Iris Publishers

ISBN: 978-1-946628-17-6

Emergent Population Behavior of Agent-Based Model

In honor of Professor Zhenbang, Kuang to celebrate his Ninetieth Birthday

Emergent Population Behavior of Agent-Based Model

Jiujiang Zhu*

School of Civil Engineering and Architecture, Wuyi University, China

*Corresponding to: jiujiangzhu@hotmail.com

Author Biography

Prof. Jiujiang Zhu, a distinguished mathematician, has significantly contributed to various research areas, including solid and fluid mechanics, materials science, image analysis, computational biology, computational fluid dynamics, and GPU parallel competing. His unique perspective, enriched by his thermodynamics and continuum mechanics background, has been recognized by the academic community. In 1997, the Chinese Academy of Sciences awarded him the prestigious "One Hundred Excellent Young Scientists" award, acknowledging his mathematical modeling and numerical simulation expertise. He is the Editor-in-Chief of "Global-Journal-of- Engineering-Sciences."

Published by

Iris Publishers

United States

Date: July 15, 2024

ISBN: 978-1-946628-17-6

List of Contents

Chapter 1

Cell Adhesion Mediates Clonal Formation of Stem and Differentiated Cell Populations

Chapter 2

Agent-based Modelling of Cell-cell Interactions for *in vitro* Vascular Formation and Cancer Cell Growth

Chapter 3

Ellipsoid Particle Dynamics Simulation on Multi-Core Processors Platforms

Chapter 4

Tensegrity Model for Living Cells and Living Tissue

Chapter 5

Simulation of Bacterial Cell Swimming Based on Lattice-Boltzmann Method and Agent-Based Model

Table of Figures

Cell Adhesion Mediates Clonal Formation of Stem and Differentiated Cell Populations

Abstract

A 3-dimensional agent based biophysical model is proposed in this paper. The model explores how the collective cell migration and spatial-temporal pattern formation originate in the behaviour of a collection of individual cells, each of which responds to a number of physical forces act on it, such as specific and non-specific adhesion forces between cells and cell-substrate; repulsive force between cells and cell-substrate, resistance force between cells, cell-ECM and cell-substrate. The proposed model is used to study the clonal formation of stem and differentiated cell population for *in vitro* cell culture. The proliferation and differentiation of stem cells is controlled by ligand-receptor signaling of leukemia inhibitory factor and transcription factor Oct-4 expressing. The probability of cell apoptosis and stem cell differentiation are governed by exogenous cytokine and endogenous factor stimulation in a dose-dependent manner, a modified ligand-receptor signaling threshold hypothesis taking into account the regulation of transcriptional factor is used for cell fate decision. The simulation result showed that spatial-temporal organization and clonal formation of stem cell populations are mainly mediated by cell adhesion difference between stem cells and differentiated cells.

Introduction

Embryonic stem cells can be maintained in culture indefinitely to an undifferentiated state. Due to their ability to self-renew and differentiate into a variety of other cell types, stem cells and their derivatives act as a repair system for biological tissue and body. Therefore, the utilization of embryonic stem cells provides an inexhaustible cell source in basic research, clinical research and therapeutic applications. For *in vitro* studies, embryonic stem cells were cultured in the presence of a leukemia inhibitory factor (LIF) to maintain their undifferentiated state. Because embryonic stem cells survive poorly as single cells, embryoid body formation was generally initiated from different sizes colony pieces until transferred to suspension culture. Therefore, clonal evolution of stem and differentiated cell populations play an important role for *in vitro* cell culture experiment, so does for *in vive* pattern formation of biological tissue. It has been demonstrated previously [1] that population dynamics of embryonic stem cells could be controlled by exogenous cytokine stimulations and cell intrinsic parameters. The population behavior could be predicted by ligand/receptor signaling threshold (LIST) model [2]. However, the previous model only concentrated on how stem cells generate substantial numbers of relatively pure stem cells or differentiated descendent cells. Actually, clonal formation of stem cells not only depends upon the number of cell outcomes but also interaction forces between cells and between cell and environment, especially adhesion forces. Adhesion forces between cells play a predominant ruler in pattern formation and morphogenesis, such as cell aggregate, self-assembly and cell sorting [3-8]. Adhesion force between cells and adhesion force between cell and substrate contribute to the formation of multicellular spheroid through expression E-cadherin and beta 1 integrin respectively [9]. The aim of this paper is to explore how mechanical force acts on cells mediate clonal formation of stem cells in consistent with cell signaling. The novelty of present paper lies in the following aspects. 1. We use over damping type differential equations approach, at same time, activation force is introduced to model cell motility and random walk like locomotion. 2. Penalty element is introduced to model cell division. 3. Cell fate decision is modified to dosage dependent manner relate to exogenous cytokine stimulation. 4. Our simulation shows multicellular patterning requires proper ratios of model parameters. 5. Our simulation also shows stem cells colon formation and evolution is mainly mediated by adhesion forces. This paper is organized as follows: Section 2 presents the theory and method we used, which consists of two parts. In the first part section 2 we investigate forces act on cells and mechanical model of cell migration and cell division. In the second part of section 2 we study the rules related to cell fate decision and how cell proliferation, differentiation and apoptosis respond to cell signaling. In section 3 we present our simulation results. Section 4 is conclusion.

Theory and Method

Forces and Mechanical Model

Mechanical model of cell migration has proven a valuable means of understanding the relative importance of underlying interactions and principles in determining cell migratory behaviour. Many off-lattice models for the motion of individual cells have been proposed, such as Monte-Carlo simulation [10,11]; agent-based model [12]; Voronoi-Delaunay model [13]; Langevin stochastic differential equations approach [14]; Over-damping type equations approach [15-17]. Cell active interaction with substrate (or extracellular matrix, other cells) provides cell random walk behaviour. Even if in the presence of concentration gradients of chemoattractants, in long term whole population move toward gradient direction, but individual cell still migrate in more or less random manner. This random local motion is essential for cell-cell interaction.

Most previous models did not pay attention to this issue. In the paper by Galle et al. [18], random walk did be taken into consideration, but it was modeled through Langevin stochastic differential equations. Since Reynolds number of cell migration is very small, inertial force is ignorable, therefore Langevin equation is not a good approach. We model this active random motion by an activation force. The cell motility is represented by the magnitude of activation force. In most previous models, implementation of cell proliferation was done by insert a cell in one time step, as pointed by Schaller and Meyer-Hermann [13] this may cause discontinuous events in differential equation approach. In Monte-Carlo simulation, Drasdo and Hohme [10] had proposed a process of how cell division from a perfect sphere to dumb-bell cell, and further to two sphere daughter cells. We introduce penalty element, which is used in finite element method, into cell division. Successfully adopted Drasdo and Hohme's idea in differential equation approach. The details of physical forces in our model are as follows.

Contact forces:

Following [18], in this paper, we model each individual cell as identical elastic spherical particle. Cells are assumed to be cultured over a 3-D space, which could be described in Cartesian coordinate as $z > 0$, $-l_{bsm} < x < l_{bsm}$, $-l_{bsm} < y < l_{bsm}$, where l_{bsm} is the size of basement membrane. Substrate $z = 0$ is assumed as semi-infinite elastic half-space. Each boundary of the space: $-l_{bsm} < x < l_{bsm}$ and $-l_{bsm} < y < l_{bsm}$ is simplified as a rigid body semi-infinite half-space. Extra cellular matrix (ECM) is simplified as a Newton's viscous fluid. The shape of cells is always kept as sphere. Therefore cell-cell, cell-substrate or cell-boundary could have some overlap, Hertz contact model [19] could be used to calculate the contact forces between cells (or cell-substrate, cell-boundary). Assume cell i and j with Young's modulus E_i and E_j , Poisson ratios ν_i and ν_j respectively. The position of cell i and j are $\mathbf{r}_i$ and $\mathbf{r}_j$, and the radii of cell i and j are R_i and R_j respectively. The contact deformation energy between cell i and j can be calculated in terms of Hertz model as

$$W_{ij}^{H} = \frac{2Ka^5}{5R_m^2} \tag{1}$$

In which

$$K = 4/\left[3\pi\left(k_1 + k_2\right)\right] \tag{2}$$

and

$$\begin{cases} k_1 = \dfrac{1-\nu_1^2}{\pi E_1} \\ k_2 = \dfrac{1-\nu_2^2}{\pi E_2} \end{cases} \tag{3}$$

$$R_m = \frac{R_i R_j}{R_i + R_j} \tag{4}$$

$$a = \sqrt{R_m \delta} \tag{5}$$

$$\delta = \left(R_i + R_j - \left|\mathbf{r}_i - \mathbf{r}_j\right|\right) H\left(R_i + R_j - \left|\mathbf{r}_i - \mathbf{r}_j\right|\right) \tag{6}$$

$H\left(\cdot\right)$ is Haviside function

$$H\left(x\right) = \begin{cases} 0 & \text{x<0} \\ 1 & \text{x} \geq 0 \end{cases} \tag{7}$$

Then the force act on cell i may be calculated through

$$\mathbf{F}_{j,i}^{H} = -\frac{\partial W_{ij}^{H}}{\partial \mathbf{r}_i} \tag{8}$$

Equations (1) - (8) also could be used to calculate the interaction between cell and semi-infinite half space, which equivalent to $R_j -> \infty$. The half-space could be elastic, also could be rigid body, later one corresponding to $E_j -> \infty$. For example, in the case of interaction between cell and substrate,

$$\delta = \left(R_i - z_i\right) H\left(R_i - z_i\right) \tag{9}$$

$$R_m = R_i \tag{10}$$

If assume substrate has same elastic parameters with the cell, then

$$K = \frac{2E}{3\left(1-\nu^2\right)} \tag{11}$$

Then the interaction energy between cell i and substrate may be calculated by

$$W_{s'i}^{H} = \frac{2Ka^5}{5R_i^2} \tag{12}$$

Thus, the contact force between cell and substrate i

$$\mathbf{F}_{s,i}^{H} = -\frac{\partial W_{si}^{H}}{\partial \mathbf{r}_i} \tag{13}$$

In the case of boundary condition, due to boundary is assumed as rigid body, thus $E_j -> \infty$,

$$K = \frac{4E}{3\left(1-\nu^2\right)} \tag{14}$$

The interaction energy between cell and boundary will be

$$W_{bi}^{H} = \frac{2Ka^5}{5R_i^2} \tag{15}$$

Thus, the contact force between cell and rigid boundary is

$$\mathbf{F}_{b,i}^{H} = -\frac{\partial W_{bi}^{H}}{\partial \mathbf{r}_i} \tag{16}$$

Of course, with slightly modification, boundary or substrate also could be modelled as elastic half space with different elastic constants from cell.

Adhesion forces:

Following previous reports [13,15,18], we model adhesion energy of cell-cell or cell-substrate interaction proportional to contact area, i.e. the average adhesion energy per unit contact area is a materials constant.

$$W_c^{Ad} = \varepsilon_c A_{cc} \tag{17}$$

$$W_s^{Ad} = \varepsilon_s A_{cs} \tag{18}$$

In which A_{cc} and A_{cs} are contact area between cells and between cell and substrate. The coefficients ε_c and ε_s are proportional to the receptor and ligand concentrations and the interaction energy of receptor-ligand bond [19,20]. The corresponding adhesion force could be calculated by

$$\mathbf{F}_{k,i}^{Ad} = -\frac{\partial W_k^{Ad}}{\partial \mathbf{r}_i} \quad (k = c, s) \tag{19}$$

In which $k = c$ stands for adhesion force between cells, and $k = s$ stands for adhesion force between cell and substrate. The presence of adhesion force will alter the deformation profile of cell. Therefore, the contact area between cells (or cell and substrate) should be calculated using combined Hertz contact energy and adhesion energy. As pointed out by Galle *et al.* [18] this approach is similar to Johnson-Kendall-Robert's theory [21]. However, Johnson-Kendall-Roberts model [22] result in an implicit relationship between external force and the radius of contact circle, and there are also branch solutions between them. For simplicity, in our simulation, Hertz formula is used to calculate contact area, so do subsequent repulsive force and adhesion force. In this approach, the adhesion force reduces to Derjaguin-Muoller-Toporov theory [23]. The real adhesion force may lie between JKR and MDT model, about transition from JKR and MDT, please refer to the reference paper by Carpick et al. [21] and Maugis [24]. Experiment shows that a negative critical load so called pull off force is needed to pull apart adhesion cells (or separate a adhesion cell from a surface) [25], or equivalently adhesion force not only exit when cells (or cell and substrate) are getting in touch, but also presence when they are separated within a minimum distance, which is similar to Dugdale model [21,24]. It is worth to point out that, in Derjaguin-Muoller-Toporov model, the pull off force are assumed equally distributed within contact area and outside of contact area, thus the total tearing off force is

$$F_{DMT} = 2\pi\varepsilon_k R_m \quad (k = c, s) \tag{20}$$

This adhesion force is allowed to present not only when cell is in touch but also when they are separated within a minimum distance d_0. For non-specific adhesion force, d_0 is roughly in the same order of equilibrium separation z_0 in Lennard-Jones potential [21]. In our simulation it is set as $d_0 = R_0 / 1000$. It is found that stem cells express 2–3-fold higher surface levels of β_1 integrins than transit amplifying cells, this results in adhesion force between stem cells is higher than adhesion force between transit amplifying cells [26,27]. In our simulation, we use different adhesion coefficients between stem cells, differentiated cells and between stem cell and differentiation cell.

Inertia force:

Due to the velocity of cell migration is very slow, the inertial force is much smaller than other forces, such as adhesion force and friction force. To address this point, we may give a rough estimation of Reynolds number of cell migration. Reynolds number is defined as

$$\mathrm{Re} = \frac{\rho l U}{\mu} \tag{21}$$

In which ρ is fluid (ECM) density, l is the characteristic length, U is characteristic velocity and μ is (absolute) dynamic fluid viscosity. The Reynolds number for cell migration could be estimated as following:

Let characteristic length $l = 10(\mu m)$, assume characteristic velocity U range from $5(\mu m/h)$ to $30(\mu m/\min)$, take water density $\rho = 10^3 (kg/m^3)$ and viscosity $\mu = 1\times10^{-3}(Pa\cdot s)$, then the Reynolds number will range from 1.39×10^{-9} to 5×10^{-5}. In general, the viscosity of physiological fluid is much larger than water, so corresponding Reynolds number will be much smaller. Compared with resistance and other forces, inertia force will be very small and can be reasonably ignored. Therefore, the acceleration terms should not appear in balance equations for cell migration approaching.

Resistance forces:

For simplicity, we model extracellular matrix as static fluid. As motioned in above section, the Reynolds number of cell migration is very small, so Stokes drag force is applicable to resistance force for cell moving in extracellular matrix, therefore

$$\mathbf{F}_i^f = -c_M \mathbf{v}_i \tag{22}$$

in which

$$c_M = 6\pi r_i \mu_f \tag{23}$$

Where r_i is radius of cell i, μ_f is dynamic viscosity of extracellular matrix.

When one cell contacts with other cells or substrate, some friction force will be produced. The friction force acts to cell should tangent to cell membrane, oppose to relative motion direction and proportional to the velocity of relative motion. Obviously, the friction force also depends on how large the contact surface area. Following Dallon and Othmer [15], we assume the friction force proportional to the ratio of the surface area in contact and the total surface area, *i.e.* friction force between cell and substrate is

$$\mathbf{F}_{s,i}^{\perp} = -6\pi r_i \mu_{sf} \frac{A_{is}^c}{A_{is}^t} \mathbf{v}_i^{\perp} \tag{24}$$

A_{is}^c is the surface area in contact between cell and substrate, A_{is}^t is the total area between cell and substrate, $\mathbf{v}_i^{\perp}$ stands for tangent compartment of relative velocity. Friction force between cells is

$$\mathbf{F}_{j,i}^{\perp} = -6\pi r_i \mu_{cf} \frac{A_{ij}^c}{A_{ij}^t} \left(\mathbf{v}_i^{\perp} - \mathbf{v}_j^{\perp} \right) \tag{25}$$

A_{ij}^c is the surface area in contact between cell i and j, A_{is}^t is the total area between cell i and j.

When one cell contacts with other cell or substrate, rheological contact forces along normal direction also possess viscoelastical behaviours [15,16]. We assume this viscoelasticity could be modelled as a nonlinear elastic spring parallel connected with a dashpot. The elastic force of nonlinear spring is assumed to satisfy Hertz theory Eq. (8), while the viscosity resistance force of dashpot is assumed to proportional to the surface area in contact and also proportional to the normal compartment of relative motion velocity, thus

$$\mathbf{F}_{s,i}^n \propto A_{is}^c \mathbf{v}_i^n \tag{26}$$

$$\mathbf{F}_{j,i}^n \propto A_{ij}^c \left(\mathbf{v}_i^n - \mathbf{v}_j^n \right) \tag{27}$$

For simplicity we assume the coefficients of tangent friction force and normal viscous forces are as same as each other, in overall we assume the resistance force of substrate

$$\mathbf{F}_{s,i}^r = -c_s \mathbf{v}_i \tag{28}$$

$$c_s = \mu_s A_{is} \tag{29}$$

and the resistance force of another cell

$$\mathbf{F}_{j,i}^r = -c_1 \left(\mathbf{v}_i - \mathbf{v}_j \right) \tag{30}$$

$$c_1 = \mu_1 A_{ij} \tag{31}$$

Activation force:

Contact forces, adhesion forces and resistance forces all are passive force. Apart from these passive forces, cells also have active dialogue with environment (substrate, extracellular matrix) and other cells. These dynamic dialogues control cell migration. Cell could directionally move along a chemoattractant gradient, and also could undergo random walk in the absent of chemical concentration gradient. In this paper we will not take cell chemotaxis into consideration, the main active force is cell dynamic interact with substrate. We model cell migration as random walk with certain degree velocity persistence, which represents that cells keep a memory of their previous step direction within the substrate. As shown as Figure 1 the activation force between cell and substrate is modelled as

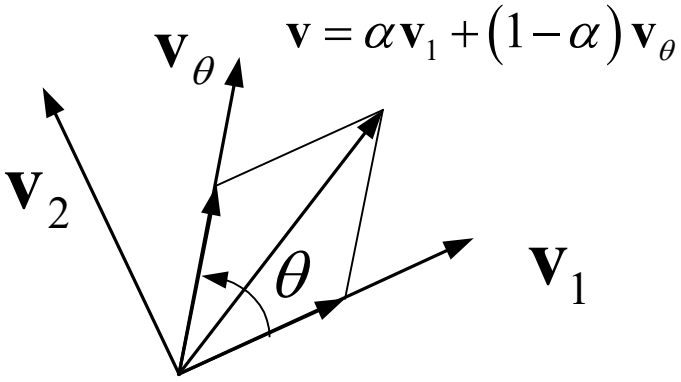


Figure 1: Schematic of cell velocity persistence.

$$\mathbf{F}^{act} = F_A \zeta(t) \left\{ \frac{\alpha\mathbf{v}_1(t) + (1-\alpha)\mathbf{v}_\theta(t)}{\left|\alpha\mathbf{v}_1(t) + (1-\alpha)\mathbf{v}_\theta(t)\right|} \right\} \tag{32}$$

It is determined by two random variables

$$\begin{cases} \zeta = \zeta(t) \\ \theta = \theta(t) \end{cases} \tag{33}$$

In Eq. (32) $\mathbf{v}_1$ is unit vector of the project of cell velocity in substrate, $\mathbf{v}_2$ is a unit vector in substrate, which is perpendicular to $\mathbf{v}_1$, $\mathbf{v}(\theta)$ is another unit vector, which is determined by

$$\mathbf{v}(\theta) = \mathbf{v}_1 \cos\theta + \mathbf{v}_2 \sin\theta \tag{34}$$

where θ is a random angle, which represent cells randomly change their migration direction. In Eq. (32), α is cell velocity persistent parameter, and F_A is the magnitude of activation force. For simplicity, when cell left substrate, we still use Eq. (32) to calculate activation force but the magnitude of activation force F_A is reduced to $F_{A3D} = F_A / 6$

Cell Cycle:

In general, the cell cycle consists of four distinct phases: G_1 phase, S phase, G_2 phase and M phase. Cells may temporarily or reversibly stop dividing and enter a state of quiescence, which is called G_0 phase. M phase is itself composed of two processes: mitosis and cytokines. In this paper, we disregard the G_0 phase. G_1 phase, S phase and G_2 phase are simplified as one interphase, i.e. cell cycle consists of two phases: I phase

and M phase. During M phase, one mother cell is divided into two daughter cells. During interphase, cells undergo cell growth, cell division is not initiated until a cell has reached a certain size. For simplicity, we assume interphase could be separated into two stages: non growth period and growth period. Cell keeps its normal volume in non-growth period and doubles its volume during growth period. As soon as reaches two times its volume, the cell enters M phase. The stem cell cycle is shown in Figure 2. Differentiation cell cycle can be shown in the same way as Figure 2, as long as stem cells in the figure are replaced by Differentiation cells accordingly.

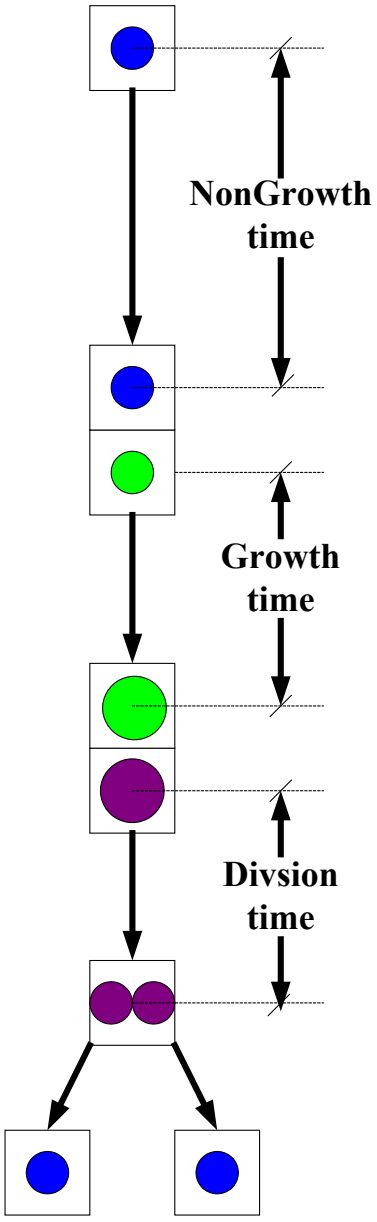


Figure 2: Stem cell cycle.

Let cell cycle duration is t_{cycle}, cell growth time is t_{gro}, cell dividing time is t_{div} and cell non growth time t_{non}, then

$$t_{cycle} = t_{gro} + t_{div} + t_{non} \tag{35}$$

In our simulation, time duration for each iteration step is $dt = 80(s)$, let iteration steps for whole cell cycle, cell growth, cell dividing and

and M phase. During M phase, one mother cell is divided into two daughter cells. During interphase, cells undergo cell growth, cell division is not initiated until a cell has reached a certain size. For simplicity, we assume interphase could be separated into two stages: non growth period and growth period. Cell keeps its normal volume in non-growth period and doubles its volume during growth period. As soon as reaches two times its volume, the cell enters M phase. The stem cell cycle is shown in Figure 2. Differentiation cell cycle can be shown in the same way as Figure 2, as long as stem cells in the figure are replaced by Differentiation cells accordingly.

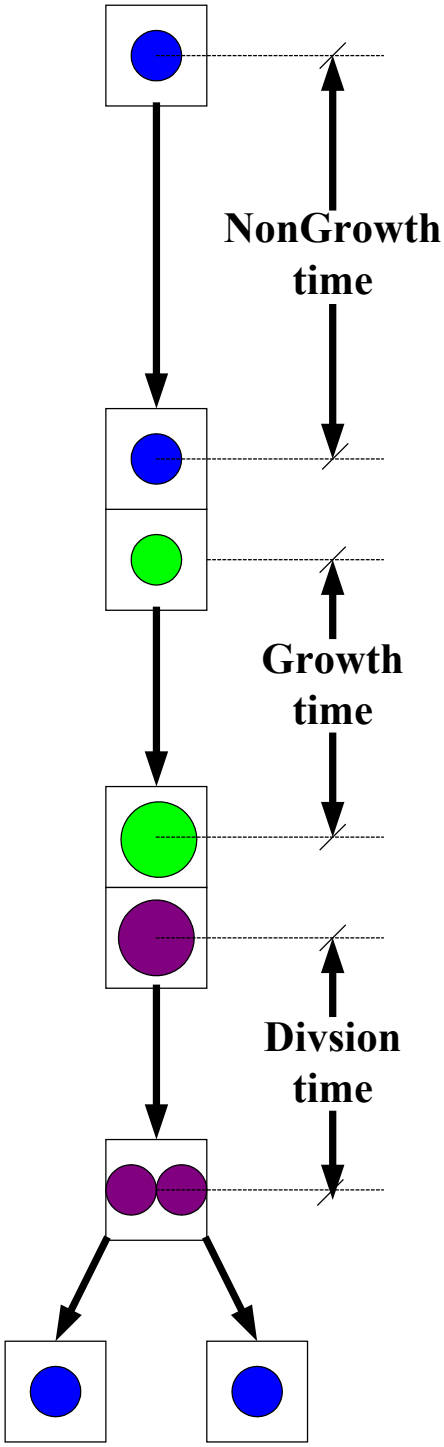


Figure 2: Stem cell cycle.

Let cell cycle duration is t_{cycle}, cell growth time is t_{gro}, cell dividing time is t_{div} and cell non growth time t_{non}, then

$$t_{cycle} = t_{gro} + t_{div} + t_{non} \tag{35}$$

In our simulation, time duration for each iteration step is $dt = 80(s)$, let iteration steps for whole cell cycle, cell growth, cell dividing and non-growth are n_{cycle}, n_{gro}, n_{div} and n_{non} respectively, then

$$\begin{cases} t_{cycle} = n_{cycle} dt \\ t_{gro} = n_{gro} dt \\ t_{div} = n_{div} dt \\ t_{non} = n_{non} dt \end{cases} \tag{36}$$

Growth time and dividing time are set as constant

$$\begin{cases} n_{gro} = 60(\text{steps}) \\ n_{div} = 60(\text{steps}) \end{cases} \tag{37}$$

While non growth time is assumed satisfy Gamma distribution. Let scope of non-growth steps is

$$n_{ns} = 600(\text{steps}) \tag{38}$$

or non-growth time scale

$$t_{ns} = 600dt \tag{39}$$

then

$$t_{non} = t_{ns} \times x \quad (0 \le x \le +\infty) \tag{40}$$

It is assumed that the distribution of x satisfies

$$p(x) = \text{Gamma}(x;\alpha) = \frac{1}{\Gamma(\alpha)} x^{\alpha-1} e^{-x} \tag{41}$$

In simulation, shape parameter is set as $\alpha = 12$. Gamma distribution random variable is generated by the method given by Gentle [28]. Cell cycle time scale

$$t_{cs} = t_{gro} + t_{div} + t_{ns} \tag{42}$$

In simulation,

$$n_{cs} = 720 \ (\text{steps}) \tag{43}$$

or

$$t_{cs} = 720 \times 80(\text{s}) = 16(hour) \tag{44}$$

Cell cycle distribution is shown in Figure 3.

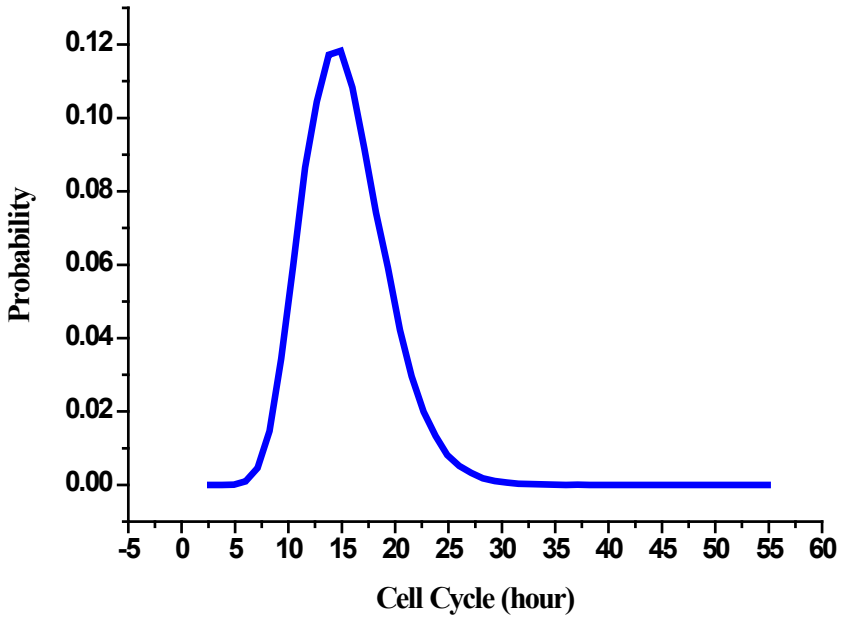


Figure 3: Cell cycle distribution.

Cell division:

Let in normal state, cell volume is V_0 and cell radius is R_0. When cell growth terminates, cell radius is R_T and cell volume

$$V_T = 2V_0 \tag{45}$$

$$\frac{4}{3}\pi R_T^3 = 2\left(\frac{4}{3}\pi R_0^3\right) \tag{46}$$

Thus, cell radius

$$R_T = (2)^{\frac{1}{3}} R_0 \tag{47}$$

During intermediate state of cell dividing, two daughter cells form a dumb bell cell (Figure 4). It is assumed that dumb bell cell conserve cell volume $2V_0$. Thus, the radius of daughter cell could be calculated as follows. Assume the radius of daughter cell is R, distance of the center of two daughter cells is d_{12}. The radius overlaps of daughter cells

$$\delta_0 = 2R - d_{12} \tag{48}$$

The volume cut off by overlap

$$V_{cut} = \int_{R-\delta_0/2}^{R} \pi\left(R^2 - z^2\right)dz = \frac{\pi\delta_0^2}{24}\left[6R - \delta_0\right] \tag{49}$$

Therefore, the volume of truncated cell is

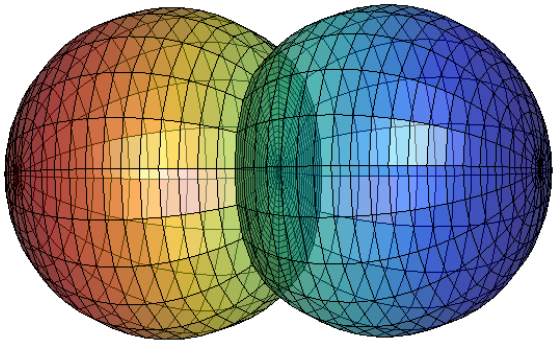


Figure 4: Dumb Bell cells.

$$V_{trouncated} = \frac{4\pi R^3}{3} - \frac{\pi\delta_0^2}{24}\left[6R - \delta_0\right] \tag{50}$$

Due to volume conservation

$$\frac{4\pi R^3}{3} - \frac{\pi\delta_0^2}{24}\left[6R - \delta_0\right] = \frac{4\pi R_0^3}{3} \tag{51}$$

therefore

$$\left(\frac{R_0}{R}\right)^3 = 1 - \frac{3}{16}\left(\frac{\delta_0}{R}\right)^2 + \frac{1}{32}\left(\frac{\delta_0}{R}\right)^3 \tag{52}$$

Substituting Eq. (48) into (52) we have

$$\left(\frac{R_0}{R}\right)^3 = \frac{\left[4 - \left(\frac{d_{12}}{R}\right)\right]\left[2 + \left(\frac{d_{12}}{R}\right)\right]^2}{32} \tag{53}$$

Denote

$$\begin{cases} y = \dfrac{R}{R_0} \\ x = \dfrac{d_{12}}{R} \end{cases} \tag{54}$$

then

$$y = \sqrt[3]{\frac{32}{(4-x)(2+x)^2}} \tag{55}$$

In which $0 \le x \le 2$, in the simulation, assume total time steps of cell division is n, then in each time step x increase by $2/n$, corresponding cell radius may be calculated by Eq. (55). The variation of y vs. x is plotted in Figure 5.

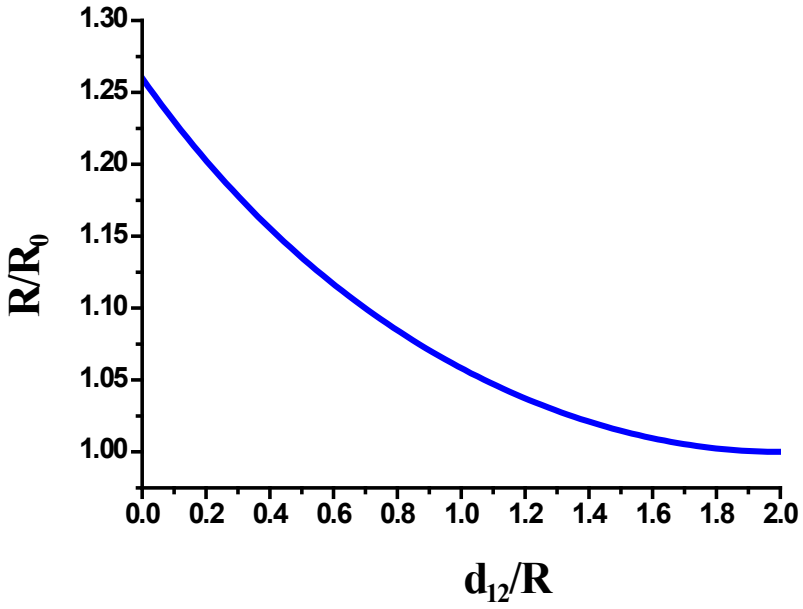


Figure 5: Sketch of dumb bell cell radius vs. distance of two daughter cells.

Penalty element:

During the period of cell division, it is assumed that the distance of centre of daughter cells is given for each time step until mitosis finishes. Let total time step of cell division is n, for each time step $x=\frac{d_{12}}{R}$ increase by $2/n$, meanwhile $y=\frac{R}{R_0}$ variation through Eq. (55). i.e. for time step i $(0\le i\le n)$

$$R=R_0\sqrt[3]{\frac{32}{\left(4-\frac{2i}{n}\right)\left(2+\frac{2i}{n}\right)^2}} \tag{56}$$

$$d_{12}=R_0\frac{2i}{n}\sqrt[3]{\frac{32}{\left(4-\frac{2i}{n}\right)\left(2+\frac{2i}{n}\right)^2}} \tag{57}$$

The distance of centre of two daughter cells are controlled by virtual penalty element, which is a nonlinear spring, the stiffness of spring is assumed to proportional to intersection area of two daughter cells.

$$k_v=\beta EA=\beta E\pi R_v^2 \tag{58}$$

The radius of intersection area R_v satisfy

$$R_v^2=R^2-\left(d_{12}/2\right)^2 \tag{59}$$

The force of penalty element

$$F_v = \sigma_v A = \beta E \varepsilon_v A = \beta E \pi R_v^2 \varepsilon_v = k_v \varepsilon_v \quad (60)$$

In which, the distance deviation

$$\varepsilon_v = \frac{|\mathbf{r}_1 - \mathbf{r}_2| - d_{12}}{R_1 + R_2} \quad (61)$$

Where R_1 and R_2 are radius of daughter cells, and $\mathbf{r}_1$ and $\mathbf{r}_2$ is the centre of daughter cells. Thus, the interaction force acts on cell 1 will be

$$\mathbf{F}^v = k_v \frac{\left(|\mathbf{r}_1 - \mathbf{r}_2| - d_{12}\right)}{\left(R_1 + R_2\right)|\mathbf{r}_1 - \mathbf{r}_2|}\left(\mathbf{r}_1 - \mathbf{r}_2\right) \quad (62)$$

In simulation k_v is chosen as $E\pi R_v^2 / 10$.

Mechanical model of cell migration:

Summarize all the forces act on cell as stated above, the motion of cell will be governed by an over dapping type of equation

$$\mathbf{F}_{j,i}^H + \mathbf{F}_{s,i}^H + \mathbf{F}_{b,i}^H + \mathbf{F}_{s,i}^{Ad} + \mathbf{F}_{j,i}^{Ad} + \mathbf{F}_i^f + \mathbf{F}_{s,i}^r + \mathbf{F}_{j,i}^r + \mathbf{F}^{act} + \mathbf{F}^v = 0 \quad (63)$$

In some previous models, such as [14], the dynamics of individual cell is modelled by Langevin equations, which are second order ordinary stochastic differential equation respect to time t. As mentioned in section 3, in this paper we ignore inertial force, so acceleration term will not enter balance equation. Therefore Eq. (63) is first order ordinary differential equation of time t. Up to now, a biophysical model has been proposed, the model consider cell migration, include cell dividing, response to a number of physical forces. As a first example, in the next section, this model will be used to investigate clonal formation and evolution of stem and differentiated cell populations.

Cell Proliferation and Differentiation

Cell signaling: Embryonic stem cells can be maintained indefinitely in a self-renewing state in the presence of leukemia inhibitory factor (LIF). Leukemia inhibitory factor signaling in embryonic stem cells acts through a cell surface receptor complex composed of binding LIF ligand to leukemia inhibitory factor receptor and binding intermediate complex to gp130 receptor [1,2,29,30]. The activated cell surface ligand-receptor signaling complexes control survival/death, self-renew/differentiation of stem cells in a dose-dependent manner. Ligand-receptor signaling threshold (LIST) model had been developed by [1,2]. Embryonic stem cells self-renew also is controlled by a POU family transcription factor Oct-4, which acts as molecular switches to activate or repress specific gene expression programmer. A critical amount of Oct-4 is required to sustain stem-cell self-renewal, and down regulation Oct-4 protein level induces stem cell differentiation. A quantitative model for corporative function of LIF-STAT3 and Oct-3/4 had been proposed by [31]. It is assumed that the relative level of Oct-4 protein reduced by 50% will induce stem cell differentiation. [1] had introduced an equivalent time delay component Oct-4 half-life $t_{1/2}$ to incorporate LIST model. In this paper, we model stem cell population dynamics by LIST [2] model with some modification. Following [1], an intrinsic population average parameter, Oct-4 half-life, is used in corporate with LIST model. In our simulation, it is set as

$$t_{1/2} = 3 * t_{cs} = 48(hour) \quad (64)$$

mass action kinetic equations of [2] LIF signaling complexes could be described as

$$L + R_1 \underset{k_{f1}}{\overset{k_{r1}}{\rightleftharpoons}} C_1 \tag{65}$$

$$C_1 + R_2 \underset{k_{f2}}{\overset{k_{r2}}{\rightleftharpoons}} C_{LIF} \tag{66}$$

In which, L is the number of LIF ligand, R_1 is the number of Unbound LIF receptor, C_1 is the number of intimidate LIF-LIFR nonsignaling complexes. is the number of unbound gp130 receptor, C_2 is the number of LIF signaling complexes. $k_{fi}\ (i = 1,2)$ is the associate rate constant and k_{ri} is dissociate rate constant, The equilibrium dissociate constant are defined as

$$k_{Di} = \frac{k_{ri}}{k_{fi}} \tag{67}$$

Then

$$k_{D1} = \frac{R_1 L}{C_1} \tag{68}$$

$$k_{D2} = \frac{R_2 C_1}{C_{LIF}} \tag{69}$$

The total number of LIFR receptors may be calculated by

$$R_{T1} = R_1 + C_1 + C_{LIF} \tag{70}$$

and the total number of gp130 receptors is

$$R_{T2} = R_2 + C_{LIF} \tag{71}$$

Assume $R_{T1}, R_{T2}, k_{D1}, k_{D2}$ and L are given, then the number of activation LIF signaling complexes is governed by

$$C_{LIF}^2 - \left[\left(R_{T1} + R_{T2} \right) + \left(1 + \frac{k_{D1}}{L} \right) k_{D2} \right] C_{LIF} + R_{T1} R_{T2} = 0 \tag{72}$$

For simplicity, in the following part of this paper, the solution of Eq. (72) will be denoted as

$$C_{LIF} = findSC\left(L, R_{T1}, R_{T2} \right) \tag{73}$$

In simulation, disassociate constant are set as

$$\begin{cases} k_{D1} = 10^{-9} \quad (M) \\ k_{D2} = 30 \times 10^{-12} (M) \end{cases} \tag{74}$$

The total number of LIFR receptors and the total number of gp130 receptors on cell surface are assumed to satisfy normal distribution. The means and variances of R_{T1} and R_{T2} are as following respectively

12)
$$\begin{cases} m_{LIFR} = 118.95 \times 10^{-12} \ (M) \\ v_{LIFR} = 13.88 \times 10^{-12} \quad (M) \\ m_{gp130} = 218.08 \times 10^{-12} (M) \\ v_{gp130} = 19.83 \times 10^{-12} \quad (M) \end{cases} \tag{75}$$

Experiment shows transit amplifying cells also has temporal proliferation ability. Normally they can proliferate 3-5 times, to model this temporal proliferation, we introduce total receptor decay facto f_{decay} , for n -th generation differentiation cell, the means of R_{T1} and R_{T2} reduced by

$$\begin{cases} m_{LIFR} = m_{LIFR} \left(1 - f_{decay}\right)^n \\ m_{gp130} = m_{LIFR} \left(1 - f_{decay}\right)^n \end{cases} \tag{76}$$

In simulation, R_{T1} and R_{T2} are randomly produced based on normal distribution according to [28] page 171. The high and low thresholds of ligands for cell survival and self-renew are assumed to be

$$\begin{cases} Low_{apop} = 0.1983 \times 10^{-12} \ (M) \\ High_{apop} = 0.3965 \times 10^{-12} (M) \\ Low_{selfr} = 0.1983 \times 10^{-12} \ (M) \\ High_{selfr} = 1.9825 \times 10^{-12} (M) \end{cases} \tag{77}$$

The high and low signaling complex thresholds of survival and self-renew are defined as

$$\begin{cases} SC_{low\_apop} = findSC\left(Low_{apop}, m_{LIFR}, m_{gp130}\right) \\ SC_{high\_apop} = findSC\left(High_{apop}, m_{LIFR}, m_{gp130}\right) \\ SC_{low\_selfr} = findSC\left(Low_{selfr}, m_{LIFR}, m_{gp130}\right) \\ SC_{high\_selfr} = findSC\left(High_{selfr}, m_{LIFR}, m_{gp130}\right) \end{cases} \tag{78}$$

The relative normalized signaling complexes for cell survival and self-renew are defined as Eq. (79) and Eq. (80) respectively

$$SC_{apop} = \frac{findSC\left(L, R_{T1}, R_{T2}\right) - SC_{low\_apop}}{SC_{high\_apop} - SC_{low\_apop}} \tag{79}$$

$$SC_{selfr} = \frac{findSC\left(L, R_{T1}, R_{T2}\right) - SC_{low\_selfr}}{SC_{high\_selfr} - SC_{low\_selfr}} \tag{80}$$

Cell fate decision: In simulation, when mitosis starts, one mother cell becomes into dumbbell cell, which will be divided into two daughter cells. Cell fate decision is carried out at the end of M-phase. After mitosis, each daughter's cell needs to decide survival or undergo apoptosis. If the daughter cell undergoes apoptosis, the cell will be deleted from the cell list. Secondly, if the mother cell is stem cell, then each survival daughter cell needs to decide to keep as stem cell or differentiate into transit amplifying cell. Cell fate decision depends upon relative signaling complexes. If $SC_{apop} < 0$, then cell will undergo apoptosis. If $SC_{apop} > 1$, then cell will be survival. For survival stem cell, if $SC_{selfr} \le 0$, two daughter cells will differentiate into transit amplifying cell. If $SC_{selfr} \ge 1$ two daughter cells all are kept as stem cell. In the intermediate case $0 < SC_{apop} < 1$ (or $0 < SC_{selfr} < 1$), in previous reference paper [1], the probability of cell survive is assumed to satisfy step function (red dot line in Figure 6)

$$p_{\text{step}}\left(x\right) = \begin{cases} 0 & \text{if } x = 0 \\ 0.5 & \text{if } 0 < x < 1 \\ 1 & \text{if } x = 1 \end{cases} \tag{81}$$

The disadvantage of this approach is obviously, as long as $0 < SC_{apop} < 1$ (or $0 < SC_{selfr} < 1$), this approach result in the output of cells is independent of signaling complexes number, i.e., cell number will keep as constant after a certain period time duration, refer to Figure 6. in [1] for details. In the case of $0 < SC_{apop} < 1$ (or $0 < SC_{selfr} < 1$), intuitively it seem that probability of cell survive should be proportional to SC_{apop} (corresponding to the green dash line in Figure 6), but actually numerical simulation shows that using this linear approach result in cell survive or not is too sensitive to the variation of . Very small changes in will cause significant changes of cell survive output. This is the reason why the previous model [1] chose the step function approach. To overcome this disadvantage, here we introduce a new approach between linear and step approach (blue solid line in Figure 6).

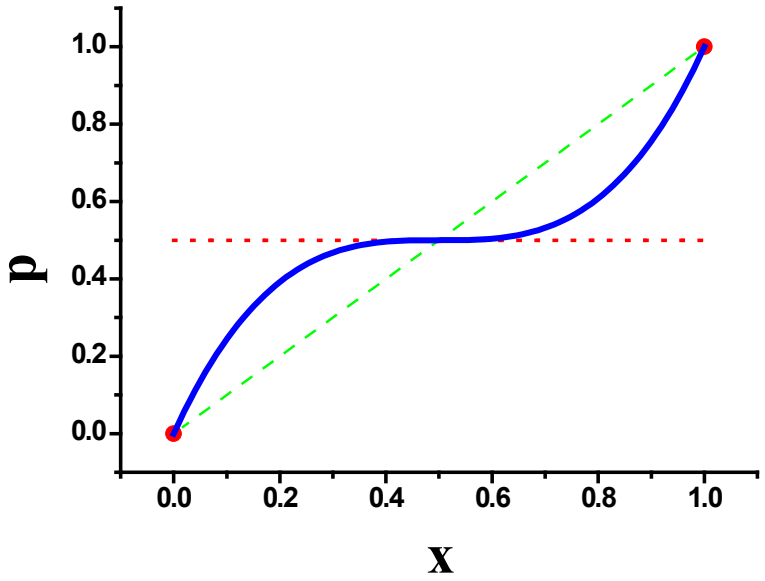


Figure 6: The probability of cell survives vs. normalized signalling complexes.

$$p(x) = 3x - 6x^2 + 4x^3 \quad (82)$$

It is easy to verify that $p(x)$ satisfies $p(0) = 0$; $p(1) = 1$; if $0 < x < 1$, then $0 < p(x) < 1$;

$$p'(x) = 3(1-2x)^2 \quad (83)$$

Thus $p'(x)$ satisfies $p'(x) \geq 0$ and $p'(0.5) = 0$ (Figure 7).

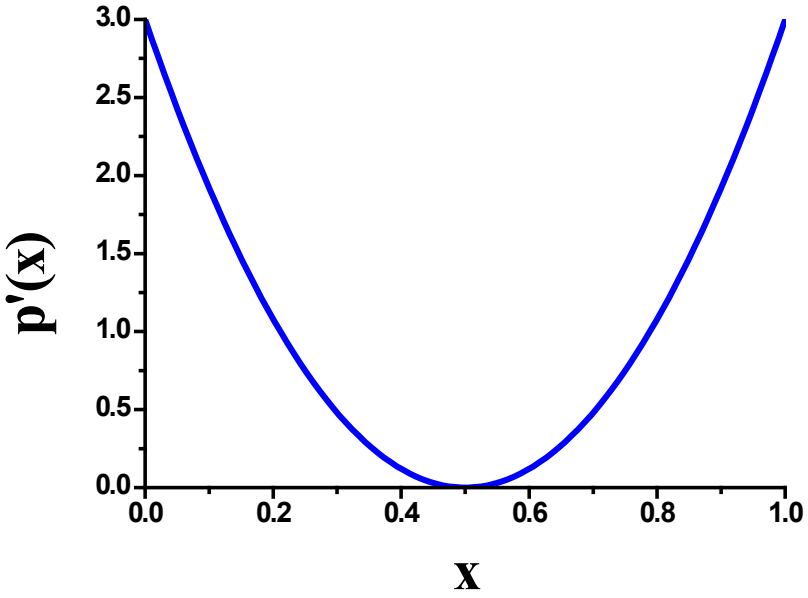


Figure 7: The derivative of the probability in Figure 6.

When $0 < SC_{apop} < 1$, a random number $0 < R_s < 1$ is produced, if $R_s < p(SC_{apop})$, daughter cell will be survival, otherwise will undergo apoptosis. If the mother cell is stem cell, cell survival decision is shown in Figure 8. If the mother cell is a differentiation cell, daughter cell survival decision can be shown in the same way as Figure 8, as long as the stem cells in the figure are replaced by differentiation cell.

As shown in Figure 9, if the mother cell is stem cell, for each survival daughter cell, first check if the cell age is larger than Oct-4 half-life. If cell age less than Oct-4 half-life, the daughter cell keeps as stem cell. Following [1], if cell age is large than Oct-4 half-life and $SC_{selfr} \geq 1$, transcriptional factor Oct4 protein will be reset to unreduced level 100%, in the simulation, it is carried out by reset the time delay of Oct-4 to 0. If cell age is large than Oct-4 half-life and $0 < SC_{selfr} < 1$, a random number $0 < R_d < 1$ is produced by computer, if $R_d < p(SC_{selfr})$, daughter cell will be stem cell, otherwise will differentiate into transit amplifying cell.

Experiments showed transit amplifying cell can only proliferate 3-5 times before apoptosis. To simulate this phenomenon, we introduce cell surface receptor decay function for differentiated cells. Assume the generation number of transit amplifying cell is m, then cell surface receptor number decay to $(1 - f_{decay})^m$ times.

Table 1: Parameters used in this paper.

Parameter	Symbol	Value
Radius of a normal cell	R_0	$5\times10^{-6}\,(m)$
Radius of cell at growth terminate	R_T	$\sqrt[3]{2}R_0$
Young modulus	E	$1000\,(Pa)$
Poisson ratio	ν	0.3
Adhesion coefficient between cell and substrate	ε_s	$3\times10^{-4}\,(N/m)$
Adhesion coefficient between stem cells	ε_{c11}	$3\times10^{-4}\,(N/m)$
Adhesion coefficient between differentiation cells	ε_{c22}	$5\times10^{-5}\,(N/m)$
Adhesion between stem and differentiation cells	ε_{c12}	$2.5\times10^{-5}\,(N/m)$
Minimum adhesion distance	d_0	$R_0/1000$
Resistance of extra cellular matrix	c_M	$3.2\,(Ns/m)$
Resistance force of substrate	μ_s	$8\times10^{10}\,(Ns/m^3)$
Resistance force of another cell	μ_1	$8\times10^{10}\,(Ns/m^3)$
Time step	dt	$80\,(s)$
Magnitude of activation force	F_A	$3\times10^{-8}\,(N)$
Velocity persistent parameter	α	0.8
Shape parameter of Gamma distribution	α_g	12
Steps of time scale of cell cycle	n_{cs}	720
Steps of time scale of non-growth time	n_{ns}	600
Steps of growth time	n_{gro}	60
Steps of dividing time	n_{div}	60
Coefficient virtual penalty element	β	0.1
Transcription factor Oct-4 half-life	$t_{1/2}$	$3\times n_{cs}\times dt$

Equilibrium dissociates constant for LIF	k_{D1}	$1\times10^{-9}(M)$
Equilibrium cross link dissociate constant	k_{D2}	$30\times10^{-12}(M)$
The mean of total number of LIFR receptors	m_{LIFR}	$118.95\times10^{-12}(M)$
The variance of total number of LIFR receptors	v_{LIFR}	$13.88\times10^{-12}(M)$
The mean of total number of gp130 receptors	m_{gp130}	$218.08\times10^{-12}(M)$
The variance of total number of gp130 receptors	v_{gp130}	$19.83\times10^{-12}(M)$
Receptor decay factor	f_{decay}	0.3
Low threshold of ligands for cell survival	Low_{apop}	$0.1983\times10^{-12}(M)$
High threshold of ligands for cell survival	$High_{apop}$	$0.3965\times10^{-12}(M)$
Low threshold of ligands for cell self-renew	Low_{selfr}	$0.1983\times10^{-12}\ (M)$
High threshold of ligands for cell self-renew	$High_{selfr}$	$1.9825\times10^{-12}(M)$

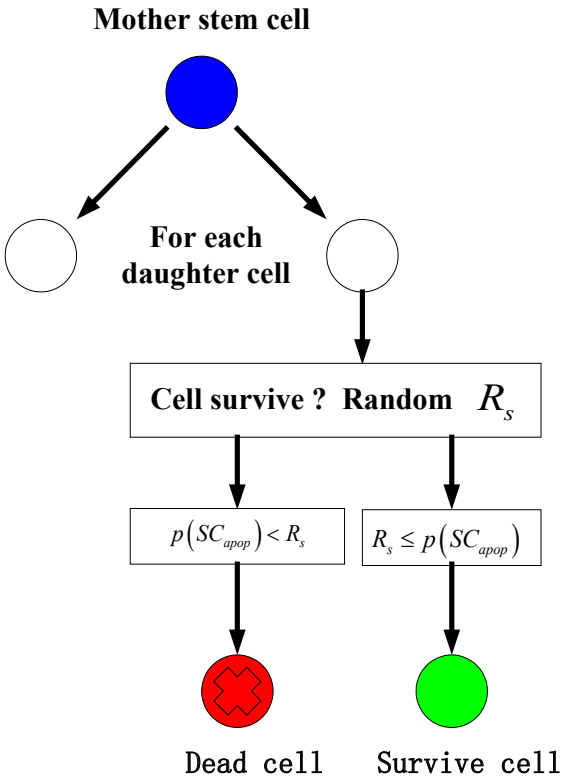


Figure 8: Survival decision of stem cell.

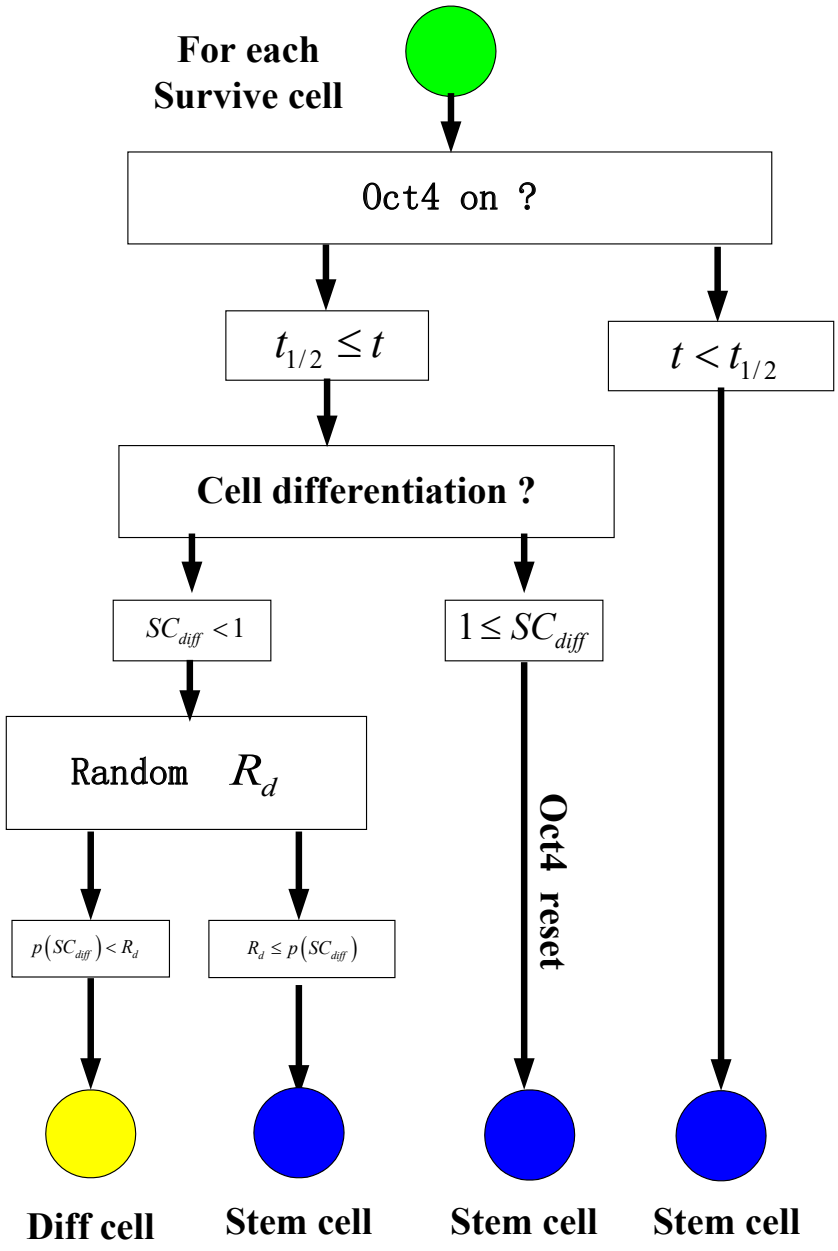


Figure 9: Differentiation decision of survival stem cell.

Results and Discussion

The parameters used in our simulation are shown in Table 1. To avoid unit conversion, the International System of Units is adopted in this paper for all parameters. As an example, we random seed 400 cell on a square of $400\mu m \times 400\mu m$. Cell type (stem cell or differentiation cell) also is chosen randomly, so in initial state, the ratio of stem cell to differentiation cell is roughly half to half. Celle could migration on or above the substrate $z = 0$. The elastic parameters of substrate are assumed to be the same as cells. The boundary of dimension for cell culture is

$x = \pm 200 \mu m$ and $y = \pm 200 \mu m$. Each boundary is assumed to be a rigid plane. The LIF ligand concentration is set as $L = 2.69 \times 10^{-12} (M)$ and $L = 500 \times 10^{-12} (M)$ respectively. Simulation results show that stem colonies emerge in a short time interval due to cell division. After Oct4- half-life, some stem cells undergo differentiation, due to adhesion force between stem cells is large than those between differentiation cells and those between stem cell and differentiation cell, differentiation cell will automatically be sorted out from stem cell colonies. This is a very important mechanism to keep stem cell colonies from breaking up. Subsequently differentiation cells distributed within the gaps between stem cell colonies or on the top of stem cell colonies. In confluent condition, some stem cell colonies are submerged among differentiation cells. In order to visualize stem cell colonies better, we provide both top view and bottom view (viewpoint from back of substrate). Figure 10 is top view when $L = 2.69 \times 10^{-12} (M)$ and iteration step equals $10,000$, while Figure 11 is its bottom view. Figure 12 is top view when $L = 500 \times 10^{-12} (M)$ and iteration step equals $5,000$, while Figure 13 is bottom view. The corresponding movies are also attached.

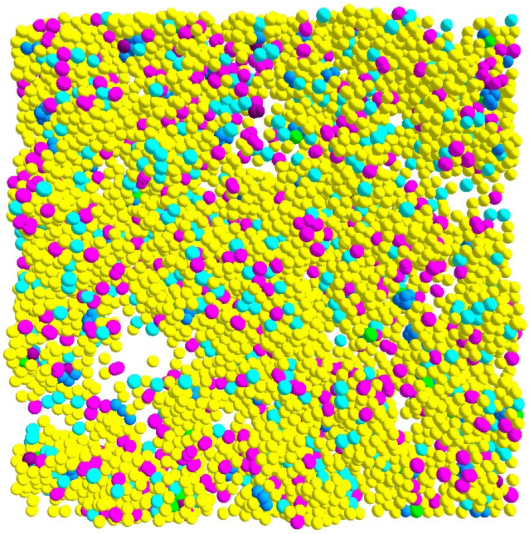


Figure 10: Top view (L=2.69, iteration=10,000).

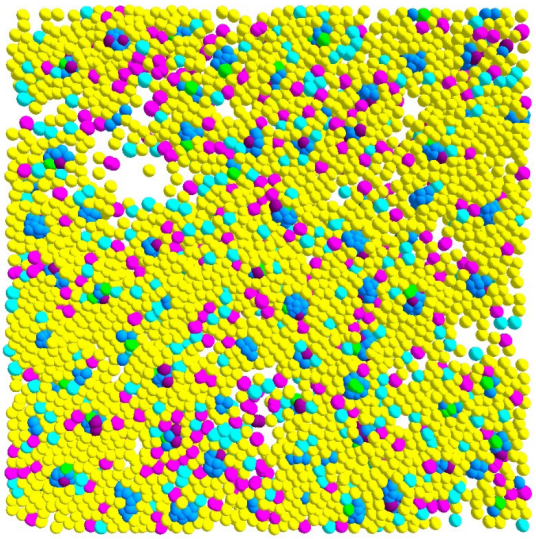


Figure 11: Bottom view (L=2.69, iteration=10,000).

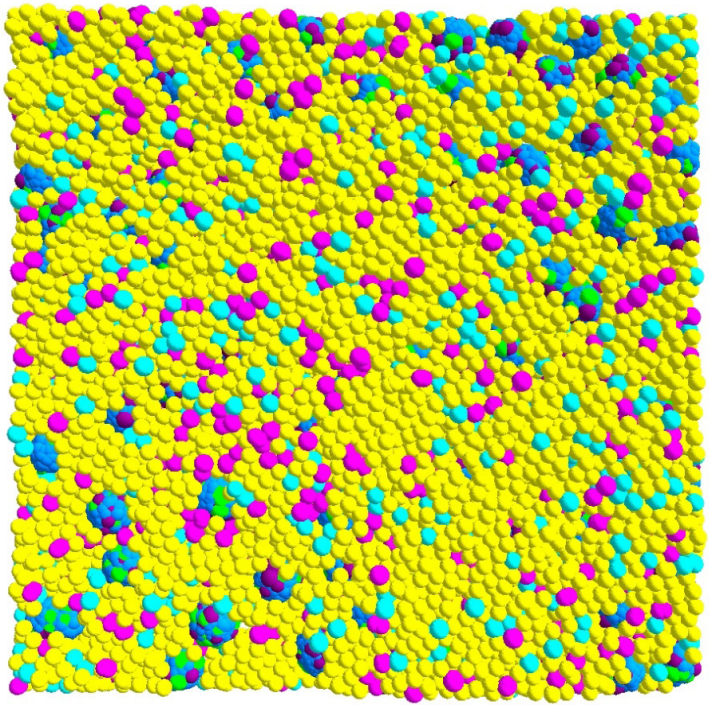


Figure 12: Top view (L=500, iteration=5,000).

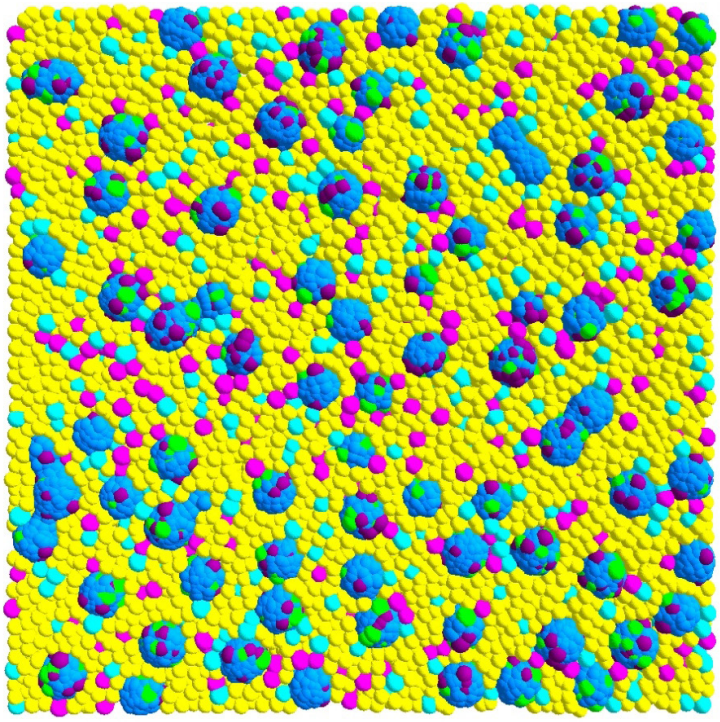


Figure 13: Bottom view (L=500, iteration=5,000).

To investigate how LIF ligand concentration affects cellular population dynamics, we set $L = 1.0,\ 1.5, 2.0, 2.4,\ 2.69, 3.0, 3.5, 4.0, 500\ \left(\times 10^{-12} M\right)$ respectively. The corresponding cell outcomes are shown from Figure 14 to Figure 24. The corresponding movies were not listed here. If $L < 2.69 \times 10^{12}\,(M)$, before Oct4-half-life, stem cell number increase exponentially. Some stem cells start to differentiate from Oct4-half life, stem cell number decreases exponentially from Oct-4 half-life. Differentiation cell number decreases before Oct-4 half-life, and increases a short period, due to some stem cell differentiation, and then decrease exponentially. The total cell number decreases exponentially when differentiation cell number starts to decrease. If $L = 2.69 \times 10^{-12}\,(M)$ stem cell number roughly keeps as constant after Oct-4 half-life, and differentiation cell number and total cell number also keep as constant shortly after Oct-4 half-life, therefore $L = 2.69 \times 10^{-12}\,(M)$ serve as an equilibrium ligand concentration. If $L > 2.69 \times 10^{-12}\,(M)$, after Oct-4 half-life, stem cell number also increase exponentially but with much smaller increase factor than before Oct-4 half-life. The increase factor shows a LIF ligand concentration dosage dependent manner variation, which monotonously increases as LIF ligand concentration increases. If LIF ligand concentration is large enough, for example, $L = 500 \times 10^{-12}\,(M)$, the stem cell number almost increase as same speed before and after Oct-4 half-life. Figure 6 (D) in the reference paper by Viswanathan et al. [1] showed that if the probability of cell self-renews $p = 0.5$, there were net maintenance of stem and differentiated cells, which is corresponding to our Figure 20, i.e. in the case $L = 2.69 \times 10^{-12}\,(M)$. Figure 6 (D) in the reference paper by Viswanathan et al. [1] also showed that if $p = 0.6$ there were exponential increase in stem and differentiated cells, which is similar to our Figure 21. In the case $p < 0.5$, our Figure 17 and Figure 18 could be compared with Figure 3 in the paper by Roeder et al. [33] and Figure 1 in the paper by Kirouac and Zandstra [32].

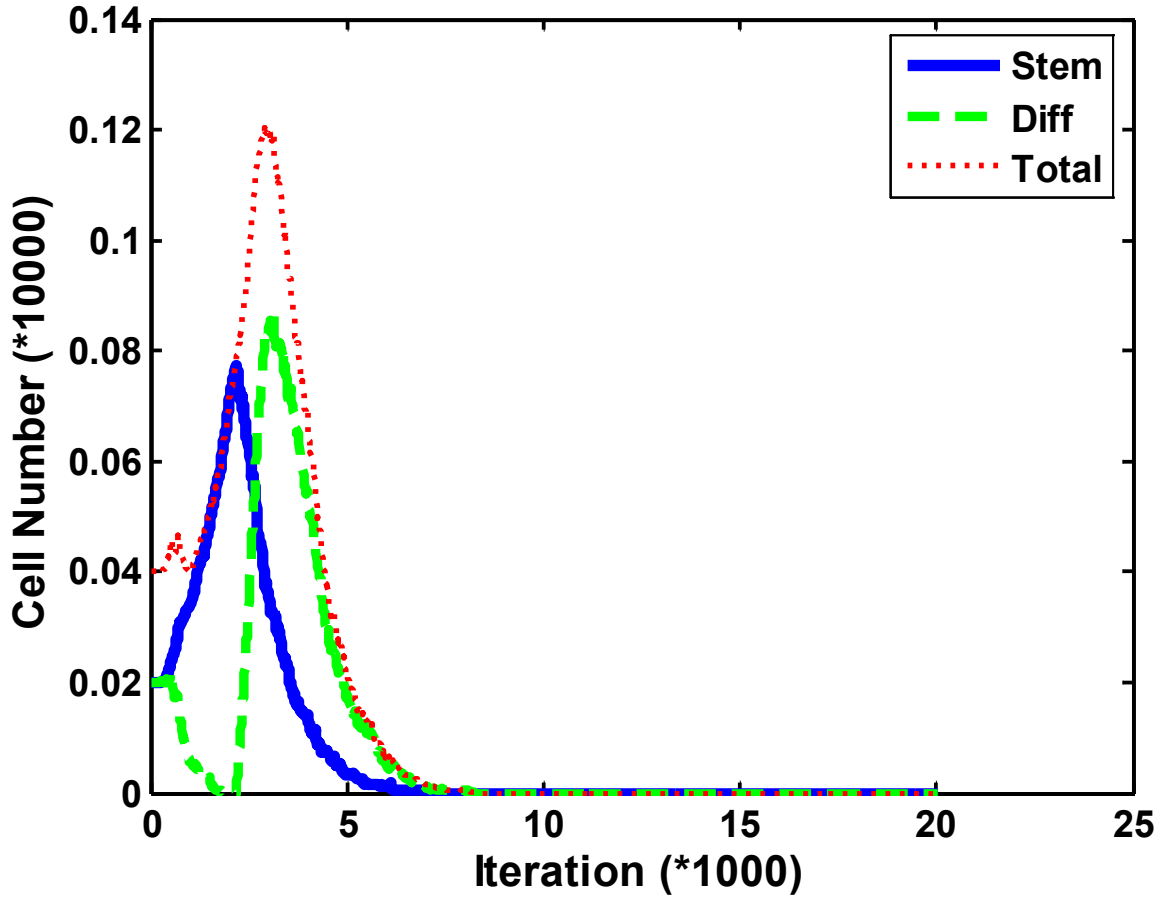


Figure 14: Cell number variation (L = 1.0).

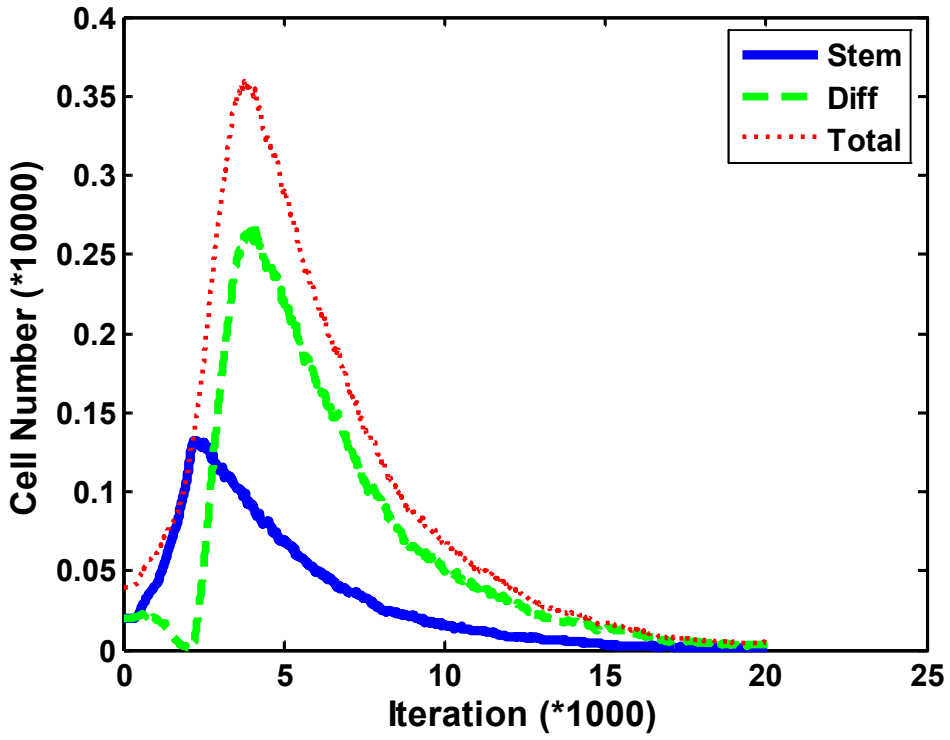


Figure 15: Cell number variation (L = 1.5).

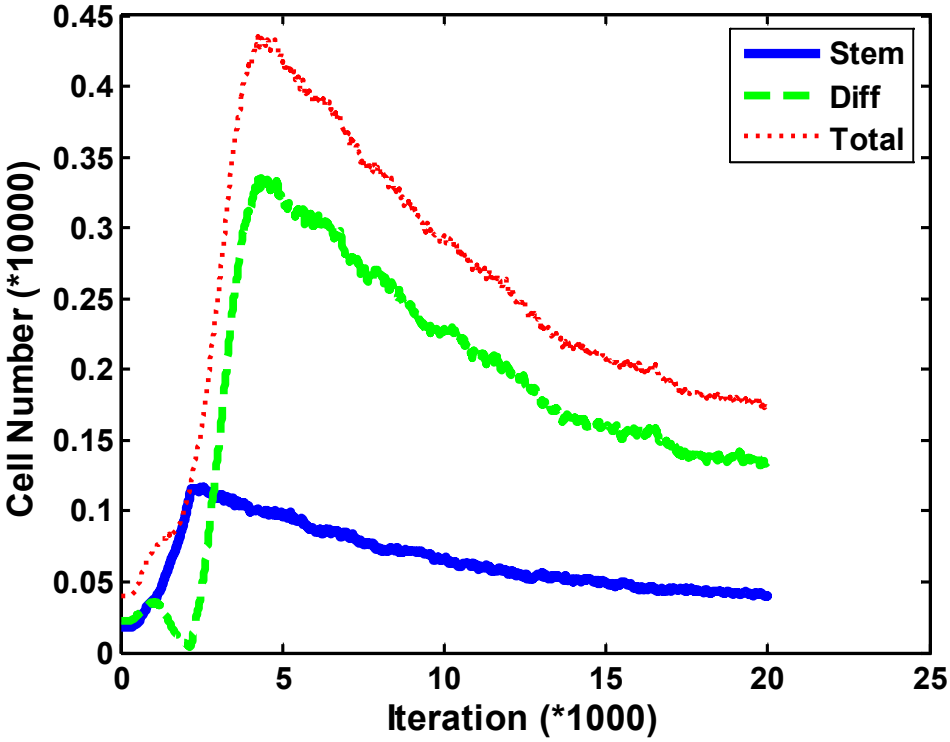


Figure 16: Cell number variation (L = 2.0).

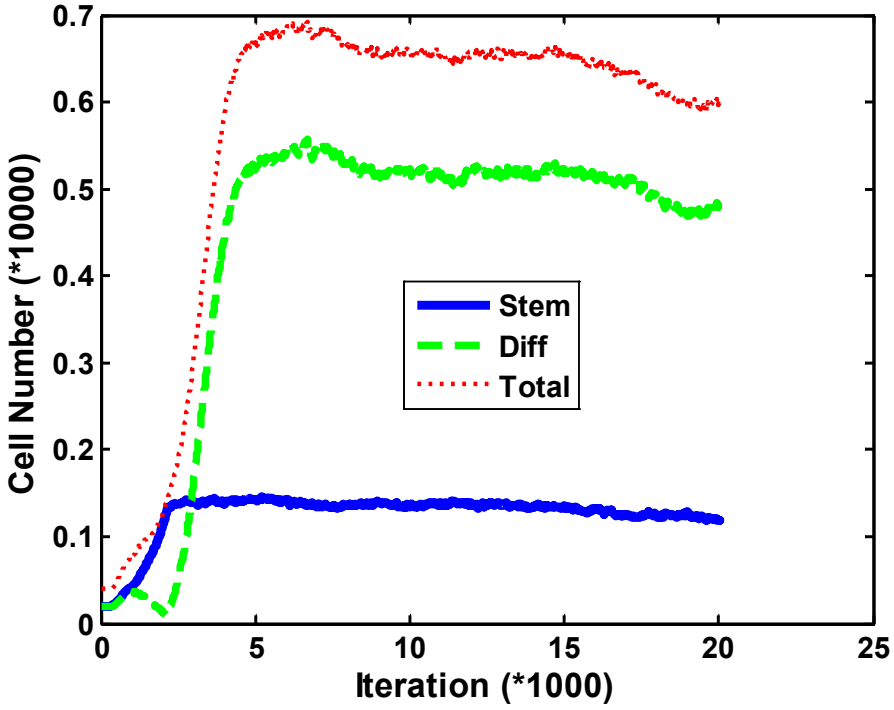


Figure 17: Cell number variation (L = 2.4).

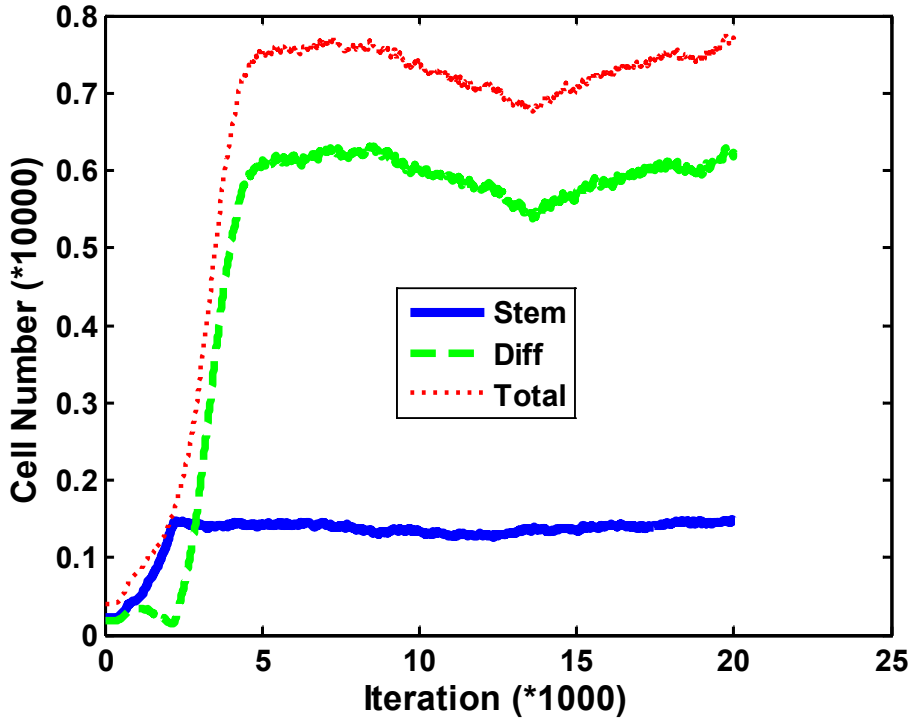


Figure 18: Cell number variation (L = 2.69).

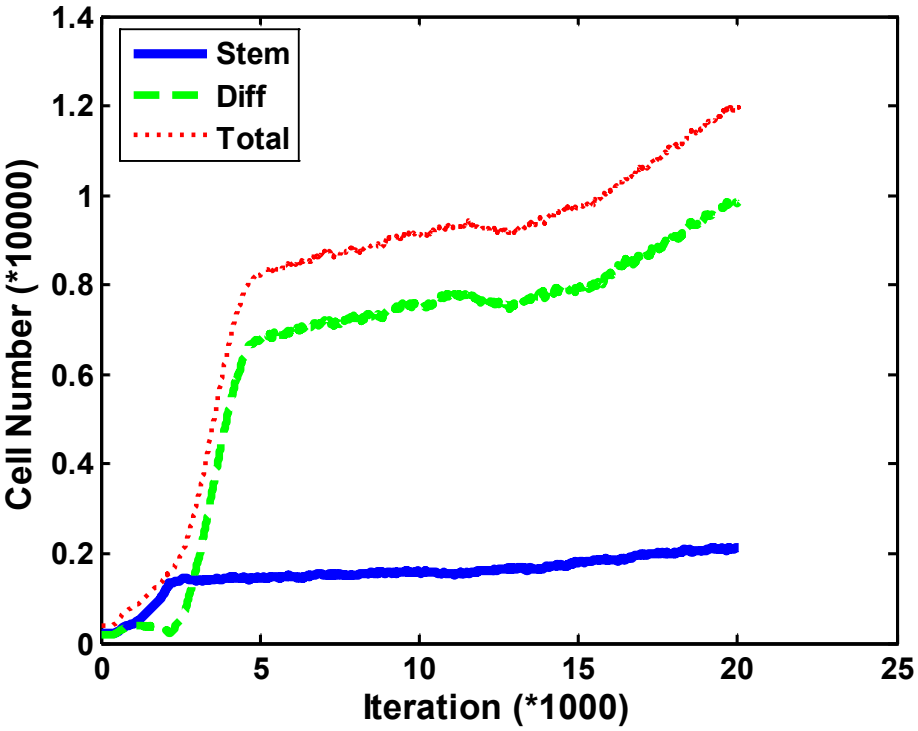


Figure 19: Cell number variation (L = 3.0).

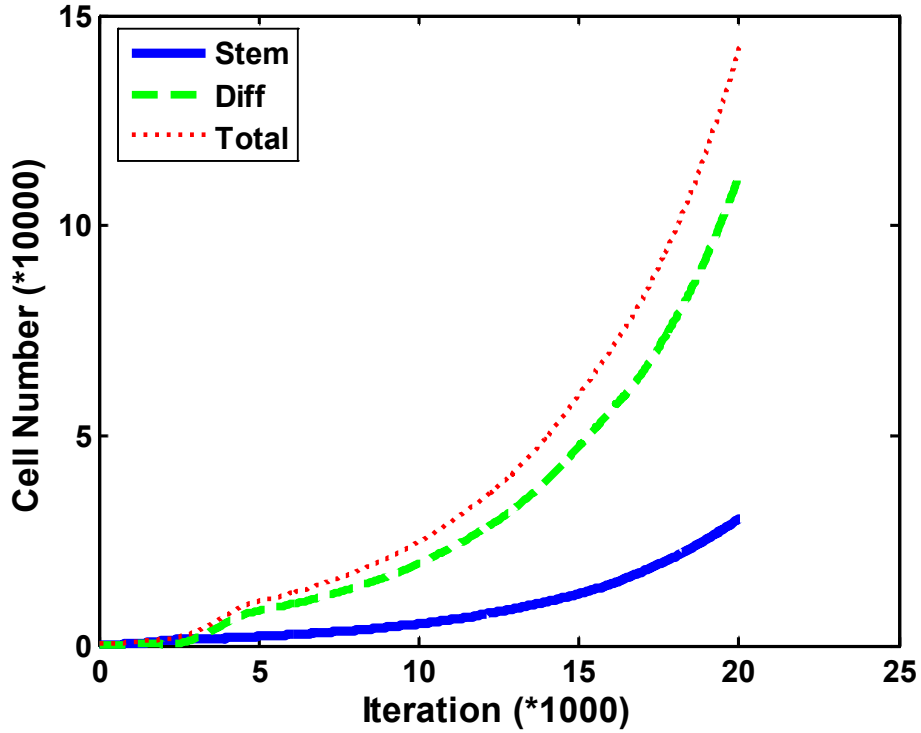


Figure 20: Cell number variation (L = 3.5).

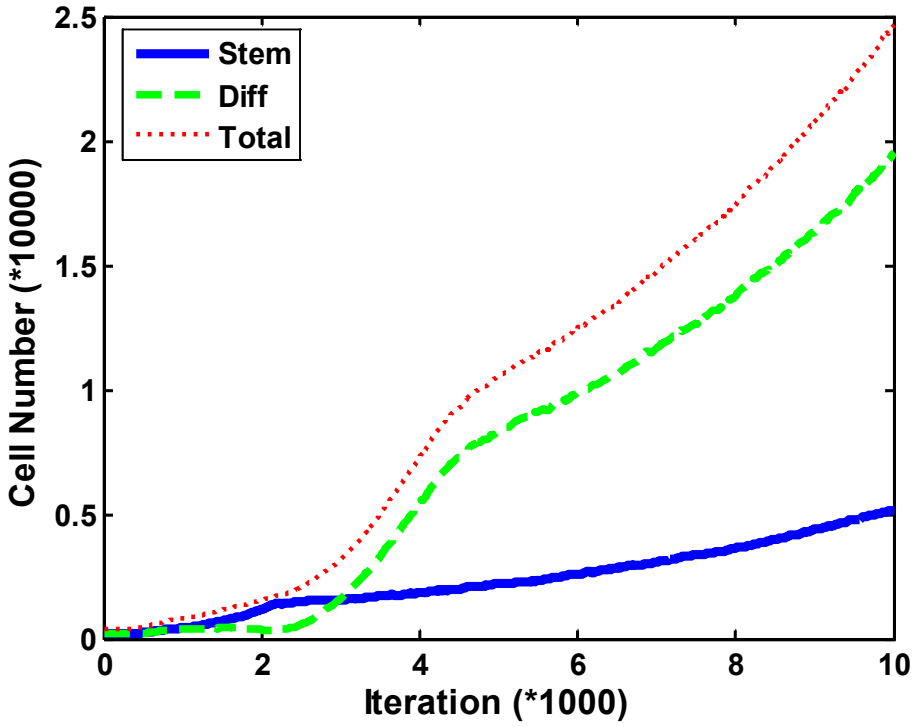


Figure 21: Cell number variation (L = 3.5).

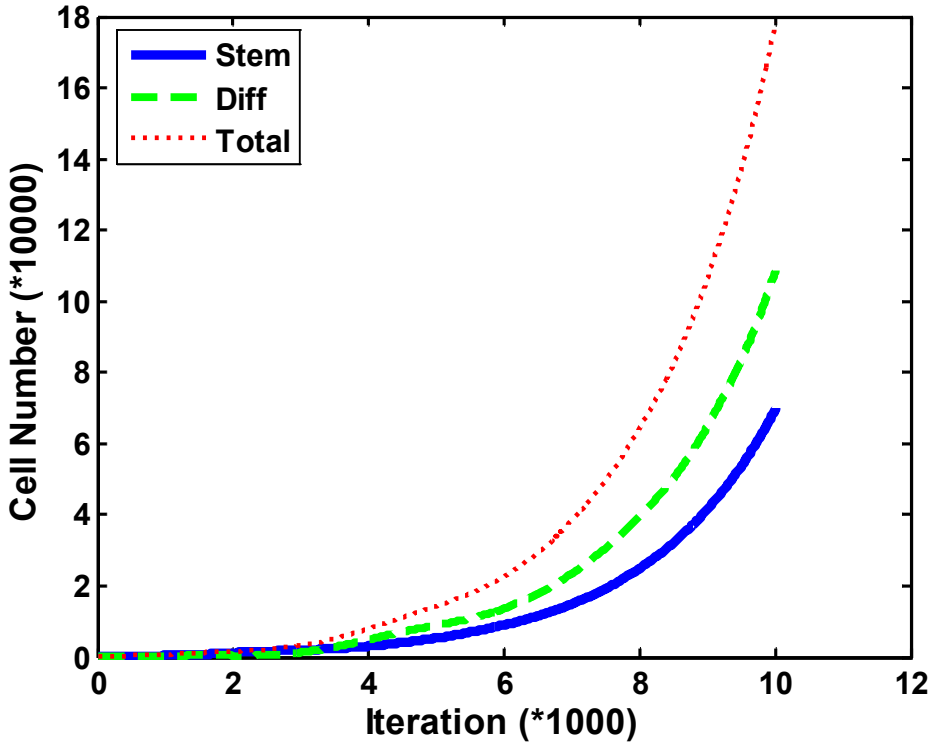


Figure 22: Cell number variation (L = 4.0).

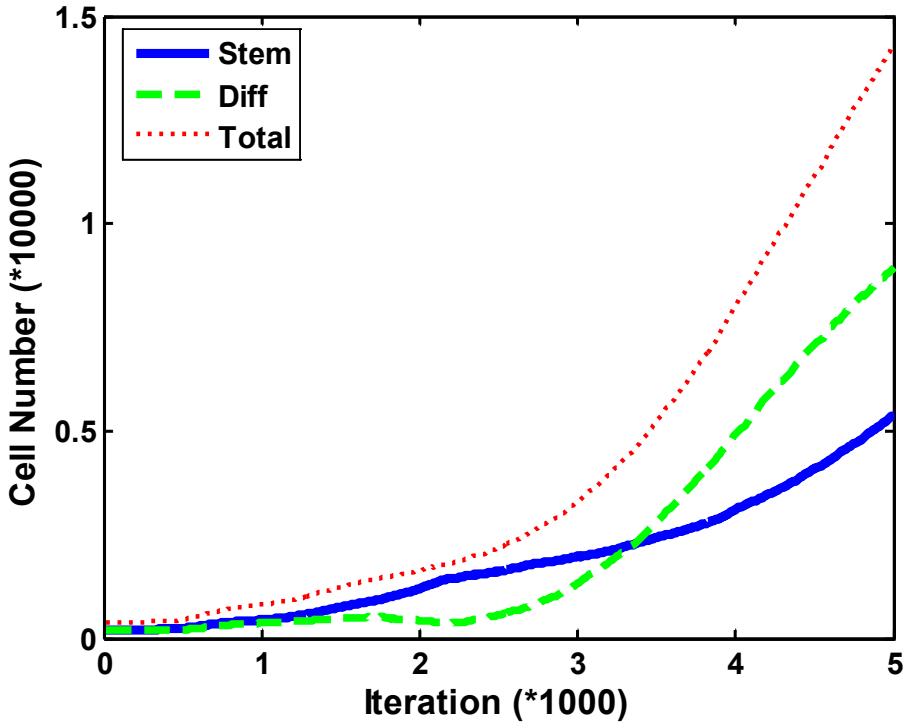


Figure 23: Cell number variation (L = 4.0).

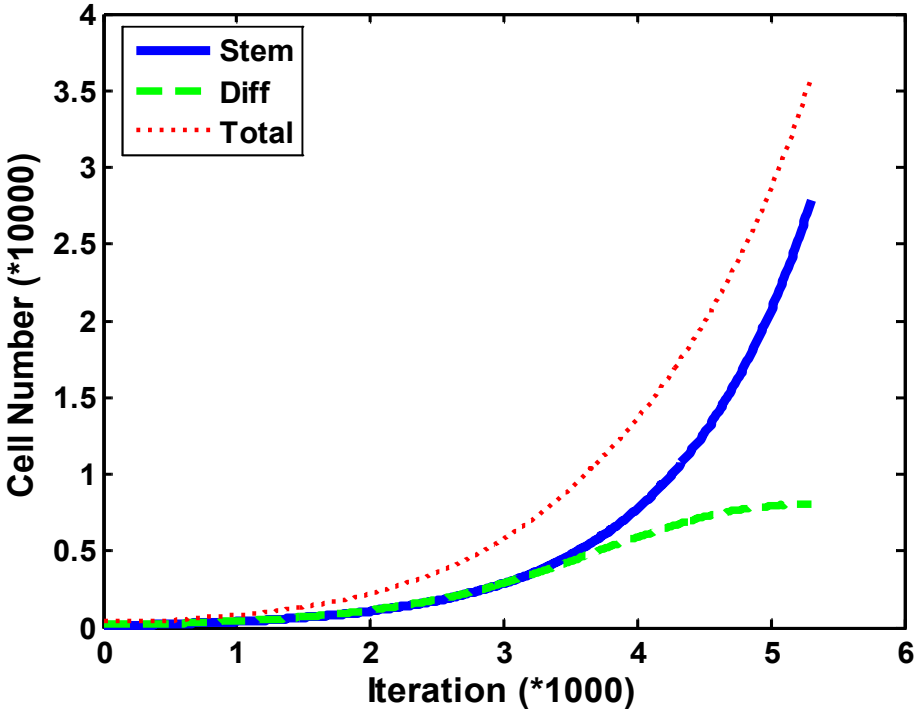


Figure 24: Cell number variation (L=500).

To explore how physical forces affect the spatial-temporal organization and self-aggregation of cells, we carried out simulation by variation of cell-specific parameters. If increase magnitude of activation forces F_A by few times, too large cell motility will produce cell diffusion effects, therefore self-assemble will not occur. If decrease adhesion coefficient between stem cells to the same level of adhesion between differentiation cells, then stem cell colonies cannot be formed. Adhesion difference between stem cells and differentiation cells play an important role in stem cell colon formation and sorting out differentiated cells. Self-aggregation and patterning of stem cells also are sensitive to the ratio of adhesion between cells and substrate to those between cells. 3-dimmension spheroids will appear by setting adhesion between cell and substrate in the same level as adhesion between stem cells. If increase adhesion coefficient between cell and substrate by 3 times or more, 2-dimension sheets, instead of 3-dimension spheroids, will form. Our simulation for colon formation of stem cells (Figure 10 – Figure 13 and movies) are similar to Figure 4 (B) in the reference paper by Viswanathan et al. [1] and Figure 1 in the reference paper by Davey and Zandstra [34]. It seems their image corresponds to a little bit lower cell density. Our simulation results are also similar to Figure 5 and 6 in the paper by Oh et al. [35] (Table 1).

Conclusion

We developed a three-dimensional agent-based, biophysical model to study clonal formation and evolution of stem and differentiated cells. The proposed model is predictive, which enables us to simulate and predict how cells migrate and division in response to exogenous cytokine stimulation and physical forces. In biological aspect, previous ligand/receptor signaling threshold model has been well modified, so that it become more flexible and predictable. It is a stochastic model in individual cell level and a deterministic model in population dynamic level. Cellular population dynamics are exactly controlled by exogenous cytokine stimulation in cooperating with intrinsic parameters, especially for intermediate signaling complexes number. In mechanical aspect, we have showed how each physical force acts on cell affect cell migration and division. Cellular mechanical interaction is the key to spatial-temporal organization and self-aggregation. Adhesion forces and cell motility play a predominant role in self-assembly of multicellular patterning. Self-aggregation and cell sorting require proper ratios of adhesion forces between cell and substrate, between cells, and cell activation force. Embryonic stem cells colon formation and evolution are mainly mediated by adhesion difference between stem cells and between differentiation cells.

References

1. Viswanathan S, Davey RE, Cheng D, Raghu RC, Lauffenburger DA, et al. (2005) Clonal evolution of stem and differentiated cells can be predicted by integrating cell-intrinsic and -extrinsic parameters. Biotechnology and Applied Biochemistry 42: 119-131.
2. Viswanathan S, Benatar T, Rose-John S, Lauffenburger DA, Zandstra PW (2002) Ligand/receptor signaling threshold (LIST) model accounts for gp130-mediated embryonic stem cell self-renewal responses to LIF and HIL-6. Stem Cells 20(2): 119-138.
3. Bischoff M, Schnabel R (2006) Global cell sorting is mediated by local cell-cell interactions in the C-elegans embryo. Developmental Biology 294: 432-444.
4. Foty RA, Steinberg MS (2005) The differential adhesion hypothesis: a direct evaluation. Developmental Biology 278(1): 255-263.
5. Kafer J, Hogeweg P, Maree AFM (2006) Moving forward moving backward: Directional sorting of chemotactic cells due to size and adhesion differences. Plos Computational Biology 2: 518-529.
6. Lecuit T (2005) Adhesion remodeling underlying tissue morphogenesis. Trends in Cell Biology 15(1): 34-42.
7. Lecuit T (2005) Cell adhesion: Sorting out cell mixing with echinoid? Current Biology 15(13): R505-R507.
8. Perez-Pomares JM, Mironov V, Guadix JA, Macias D, Markwald RR (2006) In vitro self-assembly of proepicardial cell aggregates: An embryonic vasculogenic model for vascular tissue engineering. Anatomical Record Part a-Discoveries in Molecular Cellular and Evolutionary Biology 288(7): 700-713.
9. Lin RZ, Chou LF, Chien CCM, Chang HY (2006) Dynamic analysis of hepatoma spheroid formation: roles of E-cadherin and beta 1-integrin. Cell and Tissue Research 324(3): 411-422.
10. Drasdo D, Hohme S (2005) A single-cell-based model of tumor growth in vitro: monolayers and spheroids. Physical Biology 2(3): 133-147.
11. Drasdo D, Kree R, McCaskill JS (1995) Monte-Carlo approach to tissue-cell populations. Physical Review E 52(6): 6635-6657.

12.Walker DC, Hill G, Wood SM, Smallwood RH, Southgate J (2004) Agent-based computational modeling of wounded epithelial cell monolayers. IEEE Trans Nanobioscience 3(3): 153-63.

13.Schaller G, Meyer-Hermann M (2005) Multicellular tumor spheroid in an off-lattice Voronoi-Delaunay cell model. Physical Review E Pp71.

14.Galle J, Aust G, Schaller G, Beyer T, Drasdo D (2006) Individual cell-based models of the spatial-temporal organization of multicellular systems - Achievements and limitations. Cytometry Part A 69(7): 704-710.

15.Dallon JC, Othmer HG (2004) How cellular movement determines the collective force generated by the Dictyostelium discoideum slug. Journal of Theoretical Biology 231(2): 203-222.

16.Palsson E, Othmer HG (2000) A model for individual and collective cell movement in Dictyostelium discoideum. Proceedings of the National Academy of Sciences of the United States of America 97(19): 10448-10453.

17.Zhu JJ, Coakley S, Holcombe M, MacNeil S, Smallwood RH (2006) Individual cell-based simulation of 3D multicellular spheroid self-assembly. European Cells & Materials Journal 11(Suppl 3): pp. 31.

18.Galle J, Loeffler M, Drasdo D (2005) Modeling the effect of deregulated proliferation and apoptosis on the growth dynamics of epithelial cell populations in vitro. Biophysical Journal 88(1): 62-75.

19.Hertz H (1882) Über die Berührung fester elastischer Körper (On the contact of elastic solids). Journal für die Reine und Angewandte Mathematik 92: 156–171.

20.Moy VT, Florin EL, Gaub HE (1994) Intermolecular forces and energies between ligands and receptors. Science 266(5183): 257-259.

21.Carpick RW, Ogletree DF, Salmeron M (1999) A general equation for fitting contact area and friction vs load measurements. Journal of Colloid and Interface Science 211(2): 395-400.

22.Johnson KL, Kendall K, Roberts AD (1971) Surface Energy and Contact of Elastic Solids. Proceedings of the Royal Society of London Series a-Mathematical and Physical Sciences 324(1558): 301-313.

23.Derjaguin BV, Muller VM, Toporov YP (1975) Effect of Contact Deformations on Adhesion of Particles. Journal of Colloid and Interface Science 53(2): 314-326.

24.Maugis D (1992) Adhesion of Spheres - the Jkr-Dmt Transition Using a Dugdale Model. Journal of Colloid and Interface Science 150(1): 243-269.

25.Evans E, Berk D, Leung A (1991) Detachment of Agglutinin-Bonded Red-Blood-Cells .1. Forces to Rupture Molecular-Point Attachments. Biophysical Journal 59(4): 838-848.

26.Jones PH, Harper S, Watt FM (1995) Stem-cell patterning and fate in human epidermis. Cell 80(1): 83-93.

27.Lowell S, Jones P, Le Roux I, Dunne J, Watt FM (2000) Stimulation of human epidermal differentiation by Delta-Notch signalling at the boundaries of stem-cell clusters. Current Biology 10(9): 491-500.

28.Gentle JE (2005) Random number generation and Monte Carlo methods. Springer.

29.Cartwright P, McLean C, Sheppard A, Rivett D, Jones K, Dalton S (2005) LIF/STAT3 controls ES cell self-renewal and pluripotency by a Myc-dependent mechanism. Development 132(2): 885-896.

30.Matsuda T, Nakamura T, Nakao K, Arai T, Katsuki M, et al. (1999) STAT3 activation is sufficient to maintain an undifferentiated state of mouse embryonic stem cells. Embo Journal 18(15): 4261-4269.

31.Niwa H, Miyazaki J, Smith AG (2000) Quantitative expression of Oct-3/4 defines differentiation, dedifferentiation or self-renewal of ES cells. Nature Genetics 24(4): 372-376.

32.Kirouac DC, Zandstra PW (2006) Understanding cellular networks to improve hematopoietic stem cell expansion cultures. Current Opinion in Biotechnology 17(5): 538-547.

33.Roeder I, Kamminga LM, Braesel K, Dontje B, de Haan G, et al. (2005) Competitive clonal hematopoiesis in mouse chimeras explained by a stochastic model of stem cell organization. Blood 105(2): 609-616.

34.Davey RE, Zandstra PW (2006) Spatial organization of embryonic stem cell responsiveness to autocrine gp130 ligands reveals an autoregulatory stem cell niche. Stem Cells 24(11): 2538-2548.

35.Oh SKW, Fong WJ, Teo YW, Tan HL, Padmanabhan J, et al. (2005) High density cultures of embryonic stem cells. Biotechnology and Bioengineering 91(5): 523-533.

Agent-based Modelling of Cell-cell Interactions for *in vitro* Vascular Formation and Cancer Cell Growth

Section1: Introduction

Research context

Emergence is a property exhibited by a group of typically simpler entities that arises through the interactions among them, while such a property does not exist with the single entity. For example, the formation of snowflake pattern emerges as the snowflake moves through cloud of different temperature and humidity is an emergent behaviour. Swarming is another example of emergent behaviour, in which bees, fish or birds migrate together as a group. Emergent behaviour is also a topic in complex systems studies. A complex system is a collection of parts (can be identical or different) that interact with each other and the environment and exhibit emergent behaviour. Here, I consider the formation of vascular structures in the body as a complex system which consists of an emergent pattern in interacting endothelial cells. A cancer tumour is a different but related complex system that contains various types of cells, some of which have cancer-inducing mutations. To understand the formation of a vascular structure or a cancer tumour, it is important to understand both the single cells and cell-cell interactions.

In this chapter, I want to understand how the emergent behaviours in multi-cell structures are related to low-level cell interactions. To be precise, it is the relation of inter-cell physical interaction and emergent behaviour of a group of cells that I am interested in. In particular, I hope to understand how the physical interaction amongst endothelial cells affects the pattern of vascular structure, as well as how the physical interaction among cancer cells affects the pattern of cancer cells *in vitro* but ultimately in vivo. The latter is particularly difficult because of cell heterogeneity within the tumour (see Section 2.1 for detail).

In the longer term, by studying the relation between cell-cell interaction and larger scale emergent behaviour in tissue structures, it is able to understand how tissue or organs grow, which is useful in treatment of tissues and organs. In particular, if how vascular structure is formed is understood, then it is possible to understand the factors that affect the growth of vascular structure. Similarly, the study of the relation of cancer cell interaction and tissue-level behaviour enables people to understand how tumour is formed and what affects the growth of tumour. At last, such knowledge suggests methods to control the growth of vascular structure or cancer tumour.

Furthermore, as cancer treatment drug acts differently on different cell types and different individual cells, it is difficult to predict the efficiency of drugs on controlling the growth of cancer tumour. However, studies on the effect of drugs on low-level interactions among cancer cells, may inform drug design. Note that in this chapter, as an early step to tissue growth I am limited to and so focus on *in vitro* growth in petri-dish, in which 2-dimension patterns are formed; likewise, 'cell interaction' means the cell interaction in petri-dish. It is recognized that there are two important issues of this limit. One is the cell behaviour in *in vitro* condition is not completely the same as with in vivo. Thus, there is an underlying assumption here, that the driving mechanisms of cells interaction are same (or sufficiently similar) in both *in vitro* and in vivo systems. In addition, the *in vitro* system gives experimental control, which is traded off against the biological reality in vivo. The other issue is cells grow to 2-dimensional structures (monolayers) in petri-dish, while in vivo cell growth forms 3-dimensional structures. In other studies, *in vitro* 3-dimensional experiments have been introduced, in which cells form 3-dimensional structures (spheroids). Based on the experiments in petri-dish, spheroids can help as a bridge to connect 2-dimensional *in vitro* experiments with 3-dimensional ones. The model developed in this chapter can be extended to represent such 3D structures.

To study the physical interaction among cells in vascular formation or cancer cell growth, I follow an agent-based modelling approach. First, I will find a method to represent the physical properties of cells by mathematical equations, which should include the size, shape, direction,

and position of cells. In this way the equations can show the physical variation among modelled cells. I will start with cell interactions in petri-dish; thus, the modelled cells move on substrate plane. As I will model the 3-dimension shape of cells, it will be a 3D model describing 2D cell motility. Later it can be easily extended to describe motility of cells in a 3-dimensional system.

Currently, the modelled cells move on substrate plane, thus physical interactions happen among nearby modelled cells, and space is important. With mathematical equations of physical properties of cells, I can build equations to describe the spatial relation between a pair of arbitrary modelled cells. Here I introduce the concept of contact potential, see Section 2.4.4 for detail. After that, the physical interactions among cells are modelled with a series of equations relative to the contact potential.

Using existing results of vascular formation experiments in vitro, the spatial feature of vascular network will be analysed to calibrate the physical model. It was not possible to carry out the necessary experiments to calibrate the physical model of vascular formation, and so I converted the model to a new system where I had data streams available to me.

Similar to the vascular formation model, I hope to understand the nature of the interactions among cancer cells by analysis of the patterns that they form. In *in vitro* experiments, the physical interactions among cancer cells affect the spatial feature of the pattern formed by cells. Thus, I will keep track of cell position and carry out spatial analysis on positional data. I also hope to understand the effect of a growth inhibitor drug and environmental effects, specifically hypoxia (see Section 5), on the physical interactions among cancer cells. Further, I am interested in the different responses of different cell lines that are sensitive and that are resistant to the drug in order to explore the dynamics of heterogeneous populations. Therefore, independent experiments will be carried out for each cell line, with and without drug in normal and hypoxic (oxygen starved) conditions.

The physical model will be simulated *in silico* experiments, and then the spatial distribution of cells in *in vitro* experiments will be analysed and used to calibrate the model. I will compare the pattern formed by cells with and without drug in normal and hypoxic conditions. Experiments with different treatments may require different parameter combinations in the model. I consider the differences in parameter sets when modelling the effect of drug action hypoxia and their combination.

The method of comparing *in vitro* and *in silico* experimental results is based on a set of hypotheses. See Section 1.2 for detail.

Hypothesis

The aim of this chapter was to develop an individual-based model of populations cells and inter-cell interactions. The model focused on characterising the physical properties of cells and how these drive inter-cell interaction dynamics. The biological behaviour of cells was captured in terms of changes to the physical property of cells, such as cell size and shape, and assumes that the physical model can be used to predict biological phenomenon. This assumption led to the following hypotheses:

- An *in silico* physical model can reproduce important behaviours of human cells in experimental systems.
- The model can be parameterized with biologically interpretable values to model differences arising from experimental conditions for the same biological system.
- The characterisation of cell distributions by spatial statistics enables prediction of biological phenomena using the physical model.

Chapter structure

This chapter is structured as follows:

Section 2 covers reviews of related studies about cancer cells and modelling approach. In Section 2.1, cancer is described as a complex system with six hallmarks. Then two main types of complex system modelling approaches are reviewed in Section 2.2. Also, I review the CoSMoS modelling approach which concepts are used for model repurposing from vascular formation to cancer cell growth. After that, three modelling approaches, each showing different aspects of cancer research, are detailed reviewed in Section 2.3 as case studies; and all lead to the conclusion that physical interaction plays an important role in tumour morphology. Considering the type of modelling approach, a list of issues in modelling physical interaction of cancer cells are discussed in Section 2.4, and for each issue a solution is found and concluded in the end of Section 2. In addition, in Section 2.5, the possible method to validate and calibrate the spatial distribution of cancer cells is discussed.

Following Section 2.4 and to address Hypothesis 1 and 2, a detailed physical interaction model is built in Section 3. The shape of cells is modelled in Section 3.1. With the shape and position of two modelled cells, the contact potential and adhesion potential are estimated in

Section 3.2 and 3.3 separately. In Section 3.3 the adhesion force is calculated from adhesion potential. From the contact potential, the energy is calculated by Hertz model; and from energy the contact force is calculated. This approach can be found in Section 3.4. Section 3.5 shows the calculation of contact and adhesion torque, which controls the rotation of modelled cell. The velocity and angular velocity of modelled cells are calculated from force and torque in Section 3.6; and at last, the velocity and angular velocity are used to calculate the position of modelled cells in Section 3.7. Note that as an agent-based model, the set of formulas with which to calculate all the physical quantities are the same for every modelled cell, while the actual value of quantities may vary. In section 3.8, the physical interaction between a modelled cell and the substrate is simplified as the physical interaction between the same modelled cell and its mirror image of substrate plane.

Using the physical model built in Section 3 and following the computational implementation approach discussed in Section 2.5, the physical model is first used to predict the early stage of vascular formation in Section 4.1. Although I do not have *in vitro* experimental data to calibrate the model for vascular formation, I are able to present result of *in silico* simulation with different number density of modelled cells in Section 4.1.3, with which the Hypothesis 1 is partly confirmed. In section 4.2.2, the reusability of the physical model, including additional requirements of modelling the cell cycle and population growth, is briefly summarized (drawing on a published first author journal paper). To explore Hypothesis 1 and 3, a set of eight experiments are designed afterwards (in Section 4.2.3).

Following the reusability analysis in Section 4.2.2, the cell cycle and cell population growth are modelled in Section 5.1 and 5.2 separately; and the result of experiments designed in Section 4.2.3 is shown in Section 5.3. Note that the experiments are expected to be used to calibrate both cell cycle/population growth and spatial distribution; the experimental result about spatial distribution is shown in Section 6.1. Section 5.5 demonstrates the process to calibrate the model with experimental result, which confirms the Hypothesis 1 and 2.

With population growth curve calibrated in Section 5, experiments (also designed in Section 4.2.3) are carried out to calibrate spatial distribution of modelled cells in Section 6 (this also confirms Hypothesis 3). The experimental results are images taken from experiments. The images are first passed through a pre-analysis process to enhance the quality of images in Section 6.1.1. Then in Section 6.1.2 and 6.1.3, the positions of cells are extracted from processed images. In Section 6.2, the position data is analysed by various spatial statistic tools to provide basic understanding of characteristic of patterns formed by cancer cells in experiments, from which I find that the data of control group does not reflect same number density of cells with Section 5. The control group is then removed from Section 6.3, in which spatial data is used to calibrate the model.

Section 7 concludes the three hypotheses and discusses potential next steps of future work.

Section 2: Literature Review

Introduction

In Section 1 cancer is considered as a complex system. Thus, in this section, i firstly detailed review cancer and vessel formation as complex systems, then go through modelling approaches of complex systems in Section 2.2. In Section 2.2.5, three models about various aspects of cancer are reviewed as case studies. As I aim to study the emergent behaviour of physical interaction among cells, equations will be built based on physical properties of cells discussed in Section 2.3.1. The software development approach is discussed in Section 2.4, with a computation implementation to speed up *in silico* experiment. Finally, the method of model calibration is discussed in Section 2.5.

Cancer and vessel formation as complex systems

"A complex system is any system featuring a large number of interacting components (agents, processes, etc.) whose aggregate activity is nonlinear (not derivable from the summations of the activity of individual components) and typically exhibits hierarchical self-organization under selective pressures." (Indiana University Bloomington website http://www.informatics.indiana.edu/rocha/complex/csm.html) In other words, in complex system, large number of (relatively simple) objects interact and show a new behaviour as a whole. Examples of complex systems are social systems (formed of people), the brain (formed neuronal cells), molecules (formed of atoms), and the weather (formed of air flows) (New England Complex Systems Institute (NECSI) website http://necsi.edu/guide/). In this chapter, cancer (formed of cancer cells, base tissue and vascular) and the vessel formation process (performed by endothelia cells) are referred as complex systems, with their features that suits description of the complex system explored in the following paragraphs.

Cancer: Cancer is a class of diseases where mutated cells can over-grow and over-reproduce [1]. There are six biological hallmarks of cancer, including sustaining proliferative signaling, evading growth suppressors, resisting cell death, enabling replicative immortality, inducing angiogenesis, and activating invasion and metastasis; and these capabilities enable tumour growth and spreading, termed metastasis [2-4].

The normal cells are controlled by proliferation and growth signals that ensure the normal cells enter and progress through their natural development according to a proper cell cycle. The mutated cell, however, has the capability to sustain proliferative signaling in alternative ways [3]. See Goltsov, et al. [5] for more on mutations and aberrant.

One such observed mutation in cancer cells is in the RB pathway, which decides whether a cell should proceed through its cell cycle in the normal manner. Mutations in this pathway can lead to abnormal progression through the cell cycle, and in cancer this can mean excessive proliferation. The TP53 protein can halt cell cycle in response to improper growth conditions (hypoxia, lack of space or nutrient), or trigger programmed cell death in response of cell damage [6]. In these cancer cells deregulation of this RB pathway or the TP53 protein can thus evade normal growth suppressors [7-9].

A mutation in TP53 also grants cancer cells resistance to cell death. The pre-programmed cell death works as a barrier to cancer development [10,11]. Research revealed that the apoptosis (death) mechanism is weakened in tumours high with malignancy and resistance to therapy [3]. Apart from loss of TP53 function, cancer cells have a range of other strategies to evade the apoptosis mechanism. Tumours may increase expression of anti-apoptotic regulators (Bcl-2, Bcl-) [12-14] or of survival signals (lgf1/2) [15,16]. Tumours may also "short-circuit" the extrinsic ligand-induced death pathway [3].

Another irregular behaviour of the cancer cell is replicative immortality, and this is driven by changes in the function of telomeres, structures at the end of chromosomes. The telomeres, which consist of repeated hexanucleotides [17], are involved in the capability of replicative immortality [1,3,18]. In normal cells, the telomeres shorten following cell proliferation and when the telomeres get too short the cell can no longer proliferate [132]. In this way control the number of generations the normal cells can pass through. The cancer cells, on the other hand, maintain the length of telomeres by over-expression of telomerase, and lead to immortal cells [19].

On a larger scale, tumour tissue is made up of various types of cells, including cancer cells of different types [1]. The tumour tissue has another 2 hallmarks of cancer. Unlike normal tissue, the vascular network, delivering the blood necessary for rapid cell growth, is typically activated and pervasive in tumour tissue. The blood vessels in tumour tissue are often enlarged and misshapen [20,21]. The tumour also has the capability of invasion and metastasis, which is in part the result of downregulation (decrease of quantity of a cellular component such as RNA or protein in response to external stimuli) of the cell-to-cell adhesive molecules enabling cell movement [22-24]. In addition, tumour tissue can produce a microenvironment to evade immune destruction [1].

In summary, cancer is a complex system comprising various types of normal and cancerous cells, which show irregular and complex individual and emergent behaviour on cellular and tissue scale. A key issue in cancer cell biology is to understand the relationship between dysregulation at the cellular level and tumour formation and metastasis at the tissue and body level. This is necessary to understand because anti-cancer drugs operate at the cellular level and seek to control behaviour at the system scale to affect tissue structures.

Vessel formation: The blood vessel is formed by endothelia cells. The phenomenon of vascular formation contains two main mechanisms: vasculogenesis, in which dispersive endothelial cells form vascular network; and angiogenesis, in which new vascular sprouts from an existing vascular network. In all the following parts of chapter, the phrase 'vessel formation' and 'vascular formation' refer to vasculogenesis.

As stated in [25], many *in vitro* vessel formation experiments have been undertaken with various endothelia cell lines and substrata, from which the overall process of vasculogenesis is observed and reported. The process contains following phases:

- The dispersive endothelial cells move in the direction of nearby regions with high cell densities [26,27]. It takes around 3 to 6 hours until cells contact their neighbours. Cell migration contains a small random component. Also, in this phase the speed of cells is higher than other phases.
- After cells make contact with their nearest neighbours, they elongate their shapes, and the sites connecting cells and the substrate increase. Eventually the endothelial cells form a network; [25]
- The multi-cell network slowly moves as a whole and goes through a thinning process. During this phase the structure does not deform much; [29]
- In the last phase every single endothelial cell folds up and forms the lumen of the vessel [28].

The total process takes 9-15 hours [29]. In some modelling studies the above phases 3 and 4 are considered as one [27]. In this chapter I focus on phase 1, where the shape of cell does not change, and this allows me to consider the motility of the cell and the parameters that affect it. In phase 1, the growth factor VEGF (vascular endothelial growth factor) is considered one of key factors of vasculogenesis and angiogenesis. The concentration of VEGF controls the direction of cell motility [27]. In [30], it was established that the typical size of the network is proportional to the diffusion coefficient and half-life of VEGF-A.

Further, in the *in vitro* experiment in [27], eliminating the effect of VEGF-A (by adding antibody of VEGF or removing the VEGF-A receptor) prevents the vessel formation.

In summary, in the early stage of vessel formation, large number of endothelia cells interact with each other and environments, and the result of the interaction is the net-shaped vessel structure, which is considered self-organization. Each endothelia cell is driven by the same rule, but a group of cells show different behaviour (can form the vessel). Therefore, it is considered as a complex system in this chapter.

Complex systems modelling process

Modelling process generally: As discussed in section 2.1, cancer is a complex system, which makes it challenging to understand and then manage. I need methodologies to systematically understand the key factors in cancer formation, growth, invasion and metastasis, as well as how those key factors work together. Systems models, coupled to experimental systems, can describe such interactions amongst the different factors of the system. Traditional science studies a system by dividing the system into components, and studying each component individually, i.e. a reductionist approach. However, this approach presents several difficulties for developing an understanding of the cancer system. First, not all the cells in the cancer are identical. Second, as most cells are fixed in the stroma, interaction among cells often occurs between neighbours, thus the spatial arrangement of cells must be considered. Following on from these, since cancer is so complex no single experimental approach can reflect all its characteristics; models must often based on data sets from different experiments. Finally, cancer presents the hallmarks mentioned above on both tissue scale and cell scale, the connection between cell scale behaviour and tissue scale behaviour is hard to measure [1]. Broadly, the modelling approaches for cancer can be divided into two types: data-driven and process-based [1].

Data-driven modelling: Data-driven modelling approaches normally focus on data sets arising from experiments and study the connections between them, for example correlations, regardless of the organizing principles and underlying processes that drive the system. Here, for illustration two such data-driven models are outlined: one at the tissue scale and one at the cellular scale.

Savage, et al. (2013) provides a data-driven modelling approach connecting the topological features of tumours with patient prognosis by regression analysis. In this study the Minkowski functionals: volume (V), surface area (SA), integral mean curvature (IMC) and integral total curvature (ITC) are chosen to form quantitative measurements of geometrical and topological features of 3-dimensional images of tumour. The main finding was that different tumour subtypes of the HER2 status show different responses to the anti-cancer drug tamoxifen in terms of the Minkowski functionals measurements. Additionally, the influence of four explanatory variables: tamoxifen treatment, ER status, HER2 status and node status on three morphological measurements: IMC/V, ITC/V and SA/V are analysed by linear regression. Several observations are made based on the results: the volume of both ER+ and ER- tumours is changed in response to tamoxifen treatment; tumour with differing HER2 status produce similar morphological measurements without tamoxifen treatment; HER2- tumours show a response to tamoxifen treatment in morphological measurements while HER2+ tumours do not. Finally, the tumour grade is assessed by discriminant factor analysis with Minkowski functionals as independent variables and the result shows near-perfect accuracy.

In a large, data rich study of cellular responses to sequential drug treatments, Lee et al. [31] explores the relation between patterns of drug administration and measures of levels of activity of key signaling proteins. These measures are used to define cell-type differences and drug-treatment specific differences among cell lines and for different treatment regimens by principal component analysis and partial least squares regression. To assess the synergistic actions of drugs, several drug combinations were tested in different breast cancer cells with different order and timing of drug addiction. Then it was found that addition of erlotinib at least 4 hours prior to doxorubicin causes significantly enhanced apoptotic response. As the effect of erlotinib is EGFR suppression, several experiments were designed to test if the increased apoptosis is caused by EGFR suppression. Then differentially expressed genes (DEGs) are measured on BT-20 cells with 30 minutes, 6 hours and 24 hours of erlotinib treatment. It shows that the number of DEGs dramatically increased in the groups of 6 hours and 24 hours of treatment. Compared with BT-20 cells, the number of DEGs with 24 hours treatment of HER2 cells and MDA-MB-453 cells is not significantly changed. After that, a model is built using principal component analysis (PCA) and partial least squares (PLS). From the model 4 proteins as signals are identified: cleaved caspase-8, cleaved caspase-6, phosphor-DAPK1, and phosphor-H2AX. Then the 'variable importance in the projection' (VIP) is calculated for each signal and caspase-8 is identified as the major signal. Thus caspase-8 is hypothesized as the main factor of signaling pathway that increases the sensitivity of BT-30 cells to doxorubicin, which are proved in silico test.

The two exemplary case studies show that data-driven modelling can build models on both cell and tissue scale. As data-driven modelling makes use of existing statistical models to process data sets, it is relatively easy, compared with process-based models, to build a data-driven model and explain its output. Unlike the process-based modelling approach, data-driven modelling does not require extra assumptions about unknown parts of the system. However, the relation between the data set to build data-driven model and the data produced by the model needs to be carefully examined. If the two sets of data are similar in nature, the data-driven modelling can provide high accuracy. Otherwise, data-driven modelling cannot provide valuable conclusions.

Process-based modelling: Process-based modelling, on the other hand, focuses on the key elements within the system and the way the key elements and their interactions affect the behaviour of the system. Process-based modelling includes mathematical modelling such as ordinary differential equation models and computational approaches such as individual-based models. Goltsov et al. [5] modelled the sensitivity of an intra-cellular signaling network. In this study an ODE-based model is used to study the mechanisms of resistance in RAF/MEK/ERK and PI3K/PTEN/AKT signaling network in PE04 cells. Then the model is calibrated by experimental data. From simulation of the model and experimental data, it is shown that PTEN lose or PI3CA mutation can cause cancer cells to gain resistance to drug. More tissue scaled process-based modelling approaches are reviewed in section 2.3.

There are three main types of process-based modelling: continuum, agent-based, and hybrid. The difference among these modelling types is to consider cell as continuous medium, as a particle, or as a combination of both.

Specially, individual-based modelling, or agent-based modelling, defines a set of rules that act on every agent, and the computational result of a (typically) large number of agents has the ability to show behaviours on emergent scales, for example cell-cell interactions showing emergent behaviour at the tissue scale. In current cancer biology both cells signaling, and tissue morphology are seen as very important [32,138]. However, the relationship between individual cell behaviour and tissue morphology, i.e. the tumour cell and tumour tissue, and the endothelial cell and vessel, etc., is not only hard to observe but also hard to measure. The complexity of tissue-like tumours presents the challenge of connecting molecular control mechanism with tissue-based behaviour [33].

Typical cell-based models in vessel formation studies [34] are lattice-based models that follow the energy minimization assumption. The energy minimization concept is described in section 2.2.1. The calculation of energy is a Metropolis method, as discussed in 2.3.1. Thus, instead of inter-cell dynamics being controlled by concentration of chemotaxis alone, the motility of endothelia cells also tends to reduce the total energy of the whole system. The cell-based models also produce results well agreed with *in vitro* experiments. Unlike continuum models, cell-based models are able to include a mechanism of intra-cellular signaling pathways.

The study [26] assumes that the VEGF is not autocrine and adds a VEGF-relevant signaling pathway to the model, enabling this model to reproduce early in vivo vascular assembly process. The inclusion of modelling pathways also enables cell-based models to reproduce malignant vasculogenesis by adding other irregularities in signaling pathways, i.e. mutations. Models such as [25] are used to study vessel formation in tumours.

The continuum model, on the other side, is widely used to study the motion of tissue as a whole. For instance, in [35], it introduced equations to describe tissue growth. In all these models' cells are considered as continuous medium that evenly distributed within tissue. Then the characters of tissue and interaction between tissue and environment, which are normally described by a set of partial differential equations, are studied. The continuum model can be used to represent multi-type cells; however, it is restricted that within a tissue the cells must be homogeneous, and cell colonies of different types are considered as different tissues. Also, it is easy to track the physical and biological status of each modelled agent.

The continuum models assume the endothelial cells are continuously distributed on the substrate and thus use partial differential equations to represent the rules that affect motility of endothelial cells, e.g. the momentum balance of the cells, the distribution of chemotaxins, etc. Early studies assume that the pulling action of endothelial cells is the main driving mechanism and therefore three partial differential equations are derived, representing each of conservation of cell density, extracellular matrix density and pulling force balance [128]. After that, by observing *in vitro* experimental processes, the effect of chemotaxins is added into the equations [27,30,36]. Later model studies paid more attention to the chemotaxins - precisely chemoattractant - and its effect on cell motility. The direction of cell movement, motility persistence and friction between the moving cells and substrate are modelled. There are also additional assumptions: the endothelial cells do not divide or die during the process [37]; connected cells do not overlap; and the release of chemoattractant is faster than its degradation.

The *in silico* simulation reveals the driving force of chemoattractant, supporting the assumption of the driving role of VEGF-A in the first phase of vessel formation [27]. Finally, the most recent modelling approaches consider the effect of more than one type of chemotaxins. Both chemoattractant and chemorepellent are considered in vessel formation [38], and the *in silico* result shows that chemorepellent is more effective in driving vascular formation.

As discussed in the Introduction to this section, the purpose of this modelling approach is to study how the rules that control single cell may affect the emergent behaviour of group of cells.

In general, agent-based modeling [39] may be a suitable tool to understand the cell-scale behavior between the inter-cell level and organ level dynamics; the shortage of direct experiment data, no matter *in vitro* or in vivo, precludes clear verification. The signaling-scale study

itself is subject to lack of data for model construction and validation [40]. The tissue-scale data, on the other hand, provides possible indirect support. Unfortunately, it has its own disadvantages. As most of this kind of data is collected at certain time points, it cannot display the dynamic of tissue behavior [41], which is essential in cell-scale interaction study. For example, in the early development of tumor *in vitro* experiment, there is no vessel at first, angiogenesis happens following necrosis of the center of tumor [27].

As a process-based modelling approach, agent-based modelling has its limits. Firstly, compared to the data-driven modelling approach, process-based modelling often requires assumptions on unknown parts of the system, which may introduce errors. At the same time in process-based modelling it is hard to identify errors in model formulation and simulation implementation. The potential value of process-based modelling is that it provides a way of explaining system dynamics in terms of the mechanisms that drive the system.

Agent-based modelling is an appropriate modeling approach. There are two reasons. First, there is the same set of physical laws and biological laws that control the behaviour of agents. Second, the effect of the physical and biological laws governing individual interactions is analysed from the resulting emergent behaviour of the group of agents. In the next section I model the physical aspect of agent behaviour, including the representation of agent, calculation of agent interaction and motility.

Model repurposing process: As described in Section 1 the research direction changed part-way through model development and so the model was repurposed from vascular network simulation to tumour growth simulation, the detail of which is discussed in Section 5. This repurposing is accomplished by applying the CoSMoS process. The CoSMoS process [http://www.cs.york.ac.uk/nature/cosmos/about.html] describes a principled approach to scientific modelling and simulation: it provides a structure for managing and documenting the iterative development of a simulation and gives scientists and simulation developers tools to reason - with an appropriate balance of confidence and skepticism - about how their simulation's results relate to the domain under study [42].

There are six phases in the CoSMoS modelling process, as shown in Figure 1. The first phase is 'research context', in which the high-level motivations and hypotheses are noted. The information is considered in the content of the 'domain', the subject of scientific research. The next phase is the 'domain model', in which scientific aspects about domain are understood. The domain model is the abstraction of domain. The domain model is used to develop the 'platform model', in which the simulation is designed based on understanding of the domain. Note that in the platform model, the design is also directed by the research context. The next phase is 'simulation platform', in which the simulation is encoded following the design in platform model and research context. The final phase is 'results model', in which the behaviour of simulation is observed, and the result of simulation is recorded and analysed, typically in comparison with real data.

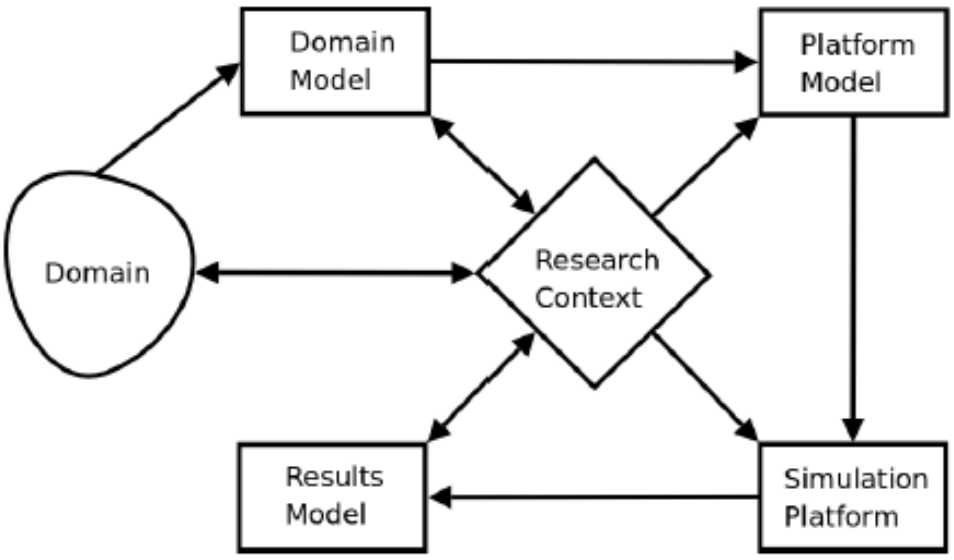


Figure 1: CoSMoS phases [95].

Here, CoSMoS is used to identify the changes that must be made to repurpose the model from vessel formation study to cancer cell interaction study. Note that neither of the models described here is built strictly under CoSMoS structure. Due to the complexity of both models, they are analysed following the CoSMoS structure phase after phase. By comparing the two models (vessel formation model and cancer cell interaction model) in each phase of CoSMoS structure, I understand the concept in models that are same and different. From the analysis, I also know the gap in my knowledge of the domain that must be filled by experimentation to calibrate the reused model. Drawing inspiration from the CoSMoS structure, I reuse the original model, including code structure, to the utmost extent, at the same time maintain confidence in the repurposed model [42].

Modelling approach as case study: In this section, three models are studied as exemplars of modelling approaches; each model focuses on one aspect of tumour morphology. All three models define rules for a single cell, and then examine the tissue-level behaviour revealed by a group of homogeneous or heterogeneous cells. Following close inspection, these three models show how the simplifications and assumptions made in the model may affect the target model behaviour, as well as which aspects are crucial to each aspect of tumour morphology study.

In [43], an agent-based model is built to explore how tumour growth is affected by glucose and cell-cell interaction in the early expansion stage. In this model, the cell cycle (see Section 2.4.1), cell proliferation and inter-cell force are modelled, in which the cell cycle can be affected by contact inhibition and the cell proliferation rate can be affected by concentration of glucose. The extracellular matrix (ECM) is not modelled.

In the model glucose is treated as a constant value outside tumour and diffuses into tumour tissue at a fixed rate. The rate of consumption of glucose by cells is also fixed. The proliferation of cells is modelled as a response to a threshold of concentration of glucose c_Q, and the apoptosis of cells as a response to another threshold of glucose c_{nec}. In this way, the model shows dependency of cell growth to concentration of glucose *C*, in which *C* is in the range of $c_{nec} \leq c \leq c_Q$. When *C* exceeds the range, the growth of cell is independent of *C*.

The distribution of cell cycle time among cells in the population is initialised as a bell-shaped curve to reflect natural variation. As the cell growth is set to be inhibited by a threshold of cell-cell force, the growth of cells with neighbours is more likely to be slowed. Thus, the real distribution of cell cycle time in simulation is deformed and has higher value with longer cycle time, as Figure 2 shows. Thus, in macroscopic view, in the model, the tumour cell growth can be delayed by stress of other cells. However, at the edge of the tumour, the modelled cells are affected by less force and perform more divisions.

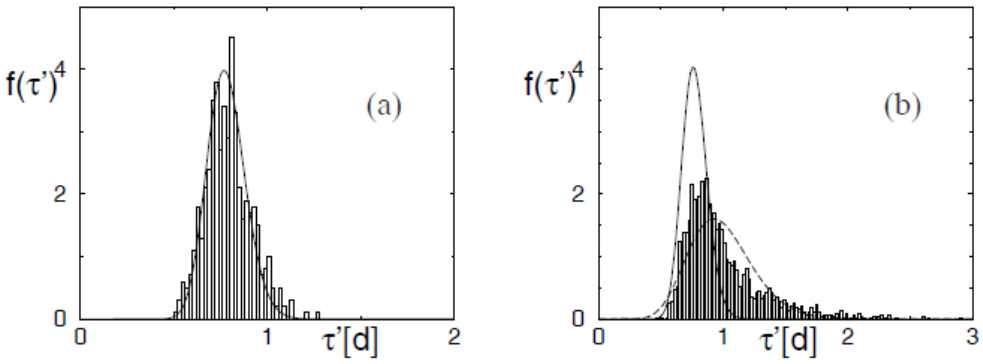


Figure 2: (a) distribution of cell cycle time in exponential growth period monolayers; (b) distribution of cell cycle time in linear growth period (dashed line).

This model is simulated, and the number of modelled cells is plotted as growth curve and compared with *in vitro* experiment data. In the experiments cells were planted to grow on petri-dish to form monolayer, or in suspension to form tumour spheroids. In monolayer the spatial distribution of tumour cells is two-dimensional, while in spheroids the distribution is three-dimensional. Both simulations and experiments are carried out with various settings of concentration of glucose. This model is a good example of simulation of tumour growth with no vascular network (angiogenesis). Also, the modelling explored how single cell behaviour (cell cycle length, proliferation) is linked to macroscopic morphology. It introduced a method to introduce the effect of contact inhibition on cell growth, which produces a simulation result that is similar to *in vitro* experiment. This modelling approach also shows how the behaviour of a model is affected by the underlying assumptions and simplifications. Although this model demonstrates a complete modelling approach, the neglect of the effect of oxygen is suspect, as oxygen also plays an important

part in tumour growth. The simplification of glucose (threshold of concentration) can be reused as oxygen except that the lack of oxygen (hypoxia) affects the cell cycle, an important part of this model. Indeed, the effect of hypoxia on tumour growth is studied in the following model.

My study aims to study the pattern formed by cancer cells, which means the cancer cells cannot be in a tight cluster, with which there is no structure to be studied. Thus, like this study, the contact inhibition to cell growth is not concerned in this chapter. However, the cell cycle length and proliferation are important regardless of the pattern, and to study cancer cell growth, these are the parts that are to be considered. This study also shows that hypoxia can result in an extended cell cycle, which should be considered in my hypoxia model. Please refer to Section 5.5.2 for more detail.

In [44], the effect of angiogenesis on tumour growth rate, tumour size and shape are explored. In this model, angiogenesis is considered as a result of cell hypoxia. The model simulated the growth of tumour cells near simplified vessel structure, as well as angiogenesis. The vascular system is considered as a source of oxygen, so simplifications are made to ignore the biological detail of vascular networks, for example, the smooth muscle of vascular walls and the blood flow inside. It is a lattice-based model including 3 types of tumour cell and ECM (modelled as a generalised cell with no cytoskeleton, no proliferation nor apoptotic capacity). The vascular system is modelled with 2 types of endothelial cells, one is the existing normal vascular cells, and the other is tumour induced vascular cells. The supply of oxygen and vascular growth factor VEGF-A are modelled as continuous concentration fields.

In this model, the normal tumour cell increases its volume and splits equally into two daughter cells when it doubles its original volume. When the normal tumour cell experiences hypoxia, it becomes a hypoxic tumour cell and secretes vascular growth factor. When the supply of oxygen is lower than a threshold the hypoxic tumour cells become necrotic tumour cells and lose volume constantly. The normal vascular cells in model have tight junction between each other. The tumour induced vascular cells, on the other hand, are initialised as inactive cells, which behave like normal vascular cells except that, when response to the vascular growth factor they turn to active neovascular cells, which proliferate and tend to move in direction of high concentration of VEGF-A.

This model is not validated with experiments. It simulates the growth of tumour with and without angiogenesis, and the simulation result is compared with other models. Without angiogenesis, the simulation result shows a slowed cell cycle with the effect of hypoxia. In this model, the structure of existing vascularization not only affects the growth rate of tumour, but also affects the shape of tumour. The tumour also shows interaction with the vascular structure, in which tumour pulls on and breaks the vascular structure, as Figure 3 shows. This model reinforces the importance of oxygen supply, and more importantly, it reveals that with proper simplification and assumption, the cell-level response has the ability to show emergent behaviour. With validation with *in vitro* or in vivo experiments this model can be more valuable.

This model suggests that vessel formation and cancer cell growth are linked through hypoxia. The future work of my study will be to simulate vessel formation together with cancer cell growth, although it is hard to operate *in vitro* experiments.

The invasion and migration of tumour is another important part of tumour study. In [24], a model is built to study the influence of different protein pathways in the process in which cancer cells are transported into the vascular system. The pathways relative to cadherins (adhesion molecules) are modelled by ordinary differential equations. The physical interaction between pairs of cells and between a cell and extracellular matrix is modelled as friction forces, interaction force and random movement. The movement of tumour cells into vessels is modelled by the Langeria equation. The modelled cell is considered to move at a low speed such that no acceleration is considered.

In this model the VE-cadherin and N-cadherin pathway is modelled. The VE-cadherin is considered to be the main factor that binds the endothelial cells together, while N-cadherin is the main factor that binds the cancer cell with an endothelial cell. The N-cadherin competes with VE-cadherin, thus the attachment of a cancer cell to an endothelial cell decreases the binding strength of this endothelial cell with other endothelial cells. In this way the cancer cell reduces the binding between endothelial cells and transports inside the vessel through the gap between unattached endothelial cells. The simulation of this model can reproduce the whole process of cancer cell attachment and transport Figure 4.

The values of many parameters in this model are from other literature. To test the model, 4 genotypes are modelled to simulate the effect of combination of pathways. The simulated result provides predictions about the length of time of transport of each genotype.

The conclusion of this study is that the transport of cancer cells into the vascular system is mainly caused by physical properties (force, velocity) of the cell. Thus, it enhanced the importance of cell-cell physical interaction to cell behaviour. It also shows another connection between physical interaction and cell behaviour (the other one is cell-cell stress and cell cycle). Therefore, physical interaction will be an important part of my study. Issues about physical models are discussed in Section 2.3.1, and the mathematical form of a physical model is built in detail in Section 3.

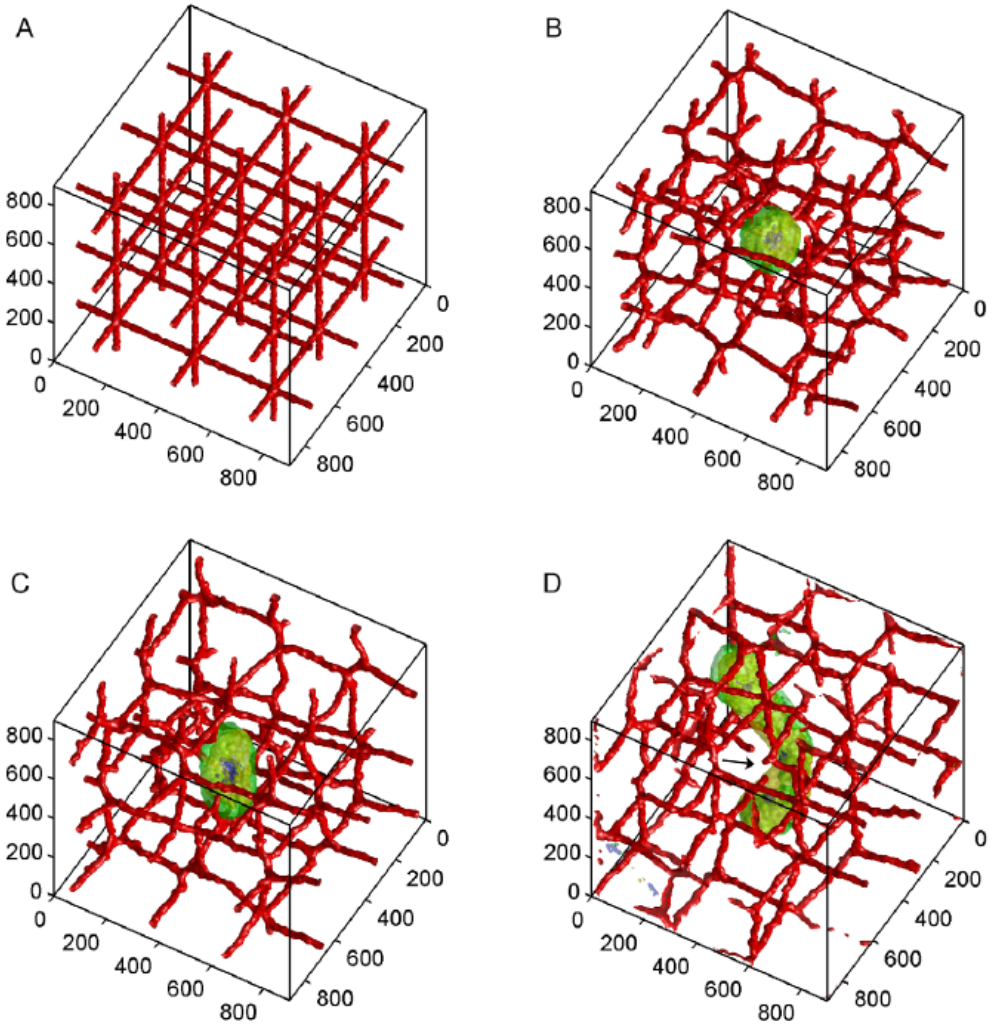


Figure 3: Shape of tumour in simulation near simplified vascular structure: (A) day 0; (B) day15; (C) day 30; (D) day 75. The tumour grows into a cylinder and finally breaks the vascular structure [44].

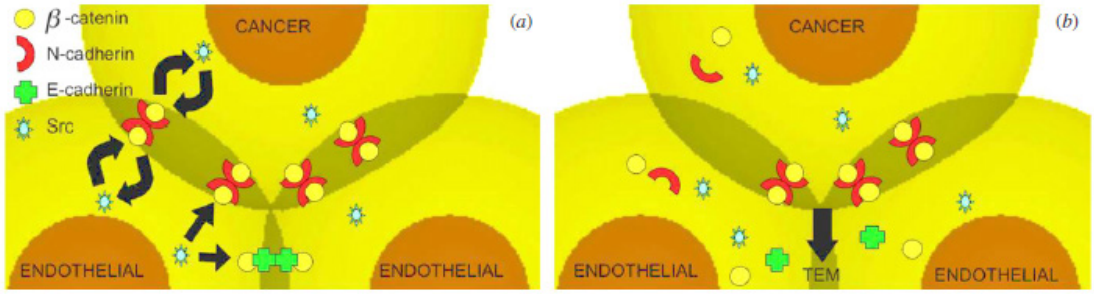


Figure 4: (a) a cancer cell make contact with pair of endothelial cells bound by VE-cadherin, and then the cancer cell is bound with endothelial cells by N-cadherin; (b) the N-cadherin in endothelial cells complete with VE-cadherin and result the loss of binds between a pair of endothelial cells [24].

As a supplementary to previous 3 papers, Chen et al. 2009 provides another point of view on the oxidation in metabolism of cancer cells. This is not exactly a model, but a study to explain the reason of different response of cancer cell to hypoxia in *in vitro* and in vivo experiments. This study provides background of the character that oxygen plays in cell metabolism; also points out that same cell may have different behaviour in different environments.

Modelling issues

The above review, combined with additional literature, leads to the need to consider a range of issues in the modelling of endothelial cell interactions and cancer cell interactions. For endothelial cells, as it does not change shape or die during vessel formation, only physical issues are considered. For cancer cells, apart from the physical part, there are also biological aspects.

Physical issues:

Space: Lattice or Off Lattice: Lattice models are a type of model in which the position of modelled cells is constrained in grids or lattice. The modelled cells can move from one lattice to another one but cannot stay in-between. In models such as [27], each lattice can contain one modelled cell; in models such as [45] several modelled cells can stay in the same lattice. The advantage of lattice models is that it takes much less time to compute simulations than off-lattice models [46]. However, the lattice spacing inherently provides a physical, artificial distance between cells. There is an assumption that cells do not overlap, although there is no justification for this. In addition, the direction and distance of cell motion are limited. The interactions between cells, and interactions between cells and environment, which is also lattice-based, is also limited (such as limit of number of cells that can have effect on each other at the same time, and the manner of inter-cell dynamic). These limitations in this exemplar are typical of those arising from lattice-based models.

There are also hybrid models comprising cells that move in a lattice structure and chemotaxins (the chemicals that can direct movement of cells) that are represented by a radiant field [27]. In this way the interaction between cell and environment (media, chemotaxins, etc.) is continuous and has an increased realism but at a computational cost.

In addition, the simplicity introduced by a lattice may affect the behaviour of model and produce artefactual phenomena (depending on the shape of lattice in particular model) in simulation. For example, in (page 9 of Andreas Deutsch, 2007 (page 37 of whole book)) simulations with square and hexagonal lattice produce significant different pattern: with the square lattice, the simulation result shows strong anisotropies, while the simulation result with hexagonal lattice shows isotropies.

Off-lattice models are also possible such as those described in [47]. In this model the direction of cell movement is continuous. All the forces and torques act on the centre of cell (agent). Unlike the lattice model, there is no explicit assumption that cells do not overlap. Given an off-lattice model, there are two questions to answer: a) Are cells rigid or elastic? b) If cells are elastic, how do they represent the degree of elasticity? These two questions are considered in Sections 2.1.4 and 2.1.6 respectively. Finally, off-lattice models can be computationally demanding and to address this problem performance may be improved by implementation optimization and parallel techniques, which is discussed in Section 2.5.

Shape: Spheres and ellipses: Cell shape plays a significant role in cell behaviour [48]. Different types of cells are of various shapes and sizes. The shape and type may vary amongst the same type of cells or during the growth of a single cell. In order to model the shape of cells, certain simplifications are necessary. There are several levels of simplification. In the most simplified cases, cells are considered as identical particles in the models [45]. In some model cells are considered as polymers of several elastic components [49], which is more detailed.

There are models that consider the cells as spheres such as [50]. There are several advantages: a) as the simplest 3-dimension shape, it is relatively simple to estimate spatial relationship between spheres; b) it is convenient to apply physical rules to a spheroidal shape. Despite these benefits, the disadvantage is also obvious: as a uniform shape, a sphere cannot represent anisotropic characteristics. Anisotropic interaction, however, may be necessary for morphological phenomena such as tumour invasion and vascular formation [51]. Cell polarity is mainly caused by the asymmetrical distribution of specific proteins on cell membrane. Cell polarity contributes to directional transport of cells and enables cells to sense surrounding cells and environment [52]. The polarity pathways are associated with tumour progression [52]. In addition, polarity proteins are also linked with the Hippo pathway that regulates tissue growth based on cell-cell contact pressure [53] to limit organ size.

In order to simulate anisotropic behaviour, the simplest shape is an ellipsoid. There are a couple of examples, such as [54-56].

Inter-cell contact: Elasticity and adhesion: In Drasdo & Hohme [43] the physical interaction among tumour cells is considered as a factor affecting the distribution of the length of the cell cycle in a population of cells. Physical interaction is also considered as driving mechanism of tissue level behaviour such as tumour invasion and intravasation, i.e. invasion of cancer cells into the lymphatic or vascular systems, in [24].

Thus, the physical interaction among tumour cells plays an important role in tissue morphology as an emerging behaviour. In this section, aspects of physical interaction are discussed.

Cell elasticity is important. When they are close enough and in contact with each other, real cells deform their shape. It is thus unrealistic to model cells as rigid particles. More importantly, in [46], it is argued that cell elasticity is necessary to have force to affect motion of cells.

In some models, the cells are represented as deformable spheres with non-deformable cubical cores [46]. Although it is a good approximation for spheroidal cells, this is not possible for ellipsoid-based spheres. In order to keep a relatively simple representation of a cell while assigning it degree of elasticity, another simplification is required. The cells in the model do not deform but are permitted to overlap with each other to show degree of elasticity. The degree of elasticity is represented by the amount that two cells may overlap. In this way the representation of cell can be consistent.

As it is decided to use agent-based modelling approach, the model also keeps the potential of having each cell has different elasticity value.

Adhesive interaction is modelled in [55]. It is considered not only one of causes of cell motion, but also a key mechanism to keep cells together to maintain normal or cancer tissue [57]. It is also one of the driving forces in a number of phenomena such as cancer cell invasion [24,58]. Therefore, the adhesive interaction between cells is an important part for both cell motion and tissue morphology. As it is an inter-cell interaction that results in tissue level phenomenon, it may become the bridge to link single cell behaviour with tissue morphology. Thus, adhesive interaction is an important part of models seeking to characterise cell-cell interaction.

To implement adhesive interaction a method is needed to estimate cell-cell interaction. There is a group of methods called metropolis algorithms [59]. In these methods the spatial relationship between cells is considered to have a particular energy, and cells are considered to move to minimize the energy of the system (Nicholas Metropolis et al. 1959). The concept of minimum energy has been proved effective to explain the sorting of embryonic cells [60]. With the metropolis algorithm the energy of a cell can be estimated; then the cell velocity is estimated from the energy.

There have been approaches using Boltzmann equations [45], Johnson-Kendall-Roberts (JKR) model [46], and there are similar approaches using the Hertz-model for homogeneous elastic sphere interactions [43,46].

Contact potential of agents: In section 2.4.3, I discussed the necessity of ellipsoidal shape of modelled cell. However, when it comes to calculation of physical interaction, the ellipsoidal shape brings two problems. First, a concept is needed to describe spatial relationship between pair of modelled cells, especially how they overlap (if they do). Second, unlike spheroidal cells, a pair of ellipsoids of same position with different orientation may have different physical interaction; therefore, it is inadequate to simply use the direct distance between pair of ellipsoidal cells as the term to represent their spatial relationship.

To resolve these two problems, I introduce the concept of contact potential, which is a term to describe spatial relationship of two arbitrary ellipsoidal particles. This concept is also demanded by metropolis algorithms which are mentioned in introduction of Section 2.4.4 to estimate energy between pair of modelled cells. From energy the force and other terms to describe the physical interaction are generated.

The concept of contact potential is used to calculate physical interaction of molecular in [61]. In this study Berne and Pechukas introduced the Gaussian overlap potential (GOP) to represent a pair of elliptical particles. The GOP has a drawback, which is it can only be used for identical elliptical particles.

Then the study in John W Perram, et al. [62] analyses the contact function and compares it with Gaussian overlap potentials (GOP) and proves the contact function shows correct extension of GOP. However, it does not have good computational features [62].

In [62,63] the closest distance between two arbitrary ellipsoids on given orientations is calculated and used to generate elliptic contact potential (ECP). Although ECP gives out correct result when two ellipsoids are in tangent contact (they just contact but do not overlap), it has several drawbacks. The most important one is that ECP does not consider difference energy caused by different orientations [64]. In [64], ECP is extended to be used for heterogeneous ellipsoids and it can be used for both rigid and elastic ellipsoids.

Transfer potential to energy: In introduction of Section 2.4.4, it has been clarified that the calculation of adhesive interaction is based on the assumption that the whole system tends to reach its minimum energy status. The contact potential is not energy in itself. Thus, it needs to be transferred to energy, and then the energy is used to estimate physical quantities such as forces and velocity. According to John W Perram, et al. [62], GOP energy can be gained by replacing the square of the scaled length of spherical symmetric potential by the geometry relationship of two ellipsoids in Lennard-Jones potential [65,66].

From contact/adhesion energy to force and torque: To prevent cells from totally overlapping, the contact force is also added to balance the adhesion force. The contact force is zero if the pair of cells (agents) do not contact; then when they just connect, or in other words 'are in tangent contact', the contact force is still zero but increases as the pair of cells starts to overlap. There is an assumption here: the degree of cell overlapping represents the elasticity of cell. Therefore, by tuning the scale of contact force and adhesion force in model, the elasticity of modelled cell can be adjusted (to match experimental data if required).

From the need for contact force, it is also known that the contact potential is required in order to estimate value of contact force. Thus, both contact force and adhesion force are estimated from the contact potential, though the specific equation may be different. In the remainder of chapter, the potential which is directly used to calculate contact force is called contact potential; and the potential which is directly used to calculate adhesion force is called adhesion potential and considered derived from contact potential with certain method.

According to the definition of force in classical physics, the force is [67]. Thus, the value of contact force equals the partial differential of contact energy against displacement. Similarly, the adhesion force equals the partial differential of adhesion energy. The torque equals the partial differential of energy against the direction of movement. There is another assumption here: that the density distribution within a cell is even and therefore in my model (see later) the forces are affected on geometry centre of ellipsoidal cell (agent).

Fluid dynamics: There should be a reference that states that cells can be considered as particles moving in fluid. I assume that each cell is considered under the effect of resistance force which occurs when cell moves within the fluid [68]. When cells move in the fluid, their maximum speed is limited [69]. Note that although I do not consider acceleration, the velocity at each time step (in simulation) may vary. The forces are always balanced, so that the velocity at each time step in simulation is constant. Thus, the Stokes resistance equations [60,70] are applicable.

According to [71,72], for a symmetric ellipsoidal particle, the resistance force is in direct proportion to the viscosity coefficient. This process is straightforward but has disadvantages, the Stokes resistance formula is built for rigid particles, while in my model the cells are considered elastic, which may bring unknown errors.

Environment: Gradients and Chemotaxis: Chemotaxis, i.e. cell movement in response to some chemical gradient, can occur in cells in response to internal (to the body) sources such as growth factor or from an external source, such as drugs. In response to gradient of chemotaxins, cells may change motion status [27], or change growth status [88]. In models such as [27], the concentration of chemotaxis-inducing gradients is described by a radiant field.

Biological issues: To simulate behaviour of cancer cells, some of the biology properties, including growth, proliferation and apoptosis, of real cells need to be added together with physical part of model. In this section these parts are discussed.

The cell cycle: The cell cycle is a sequence of events that happens in a cell leading to its division and replication ([73] The cell: a molecular approach. The sixth edition). The cycle of a eukaryote cell contains three periods: interphase, mitotic phase and quiescent ([73] The cell: a molecular approach. The sixth edition) (Figure 5).

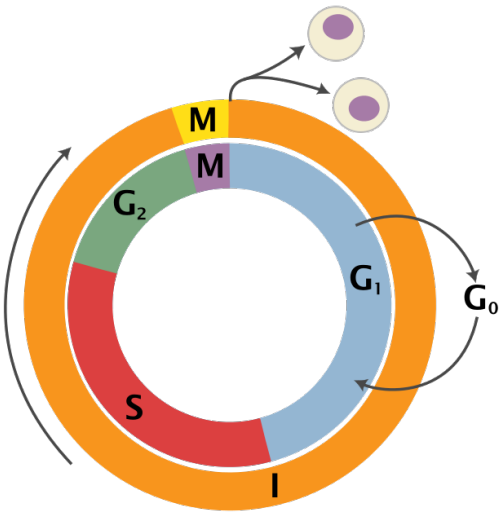


Figure 5: cell cycle phases [http://en.wikipedia.org/wiki/Cell_cycle].

The interphase can be divided into three stages: Gap 1, in which the cell increases in size, where the Gap 1 checkpoint (G1/S checkpoint) ensures that everything is ready for DNA synthesis; Synthesis stage (S stage), in which DNA replication happens; and Gap 2, in which the cell continues to grow, where the G2 checkpoint (G2/M checkpoint) ensures that the cell is ready to enter mitosis phase Figure 5.

In animal cells, in the mitotic phase (or M phase) the cell stops growing, the nuclear envelope breaks down and the chromosomes in the nucleus separate into two identical sets. Then fibers form in the cell and pull sister chromatids to opposite polar ends of the cell and form two nuclei. After that, the nuclei, cytoplasm, organelles and cell membrane are divided into two cells that contain roughly equal shares of these components. The original cell is called the mother cell; the two cells are also called daughter cells and are typically identical to each other and to the mother cell as well. The Metaphase checkpoint ensures that the cell is ready to finish division.

Quiescent phase is also called Gap 0 stage, with respect to Gap 1 and Gap 2. Note that it is not a phase in which certain cell growth or division events happens, but a resting phase where the cell stops growing. The cell may go into Gap 0 from Gap 1 stage and rest for long periods of time. Under proper condition, cell may transfer back to Gap 1 from Gap 0. The transition from and to Gap 0 may happen several times.

The apoptosis (programmed cell death) was firstly described in [98]. It is a sequence of events leading to cell change and death. It is a quick progress, normally in which the cell shrinks, the cytoplasm increases density, condensation of chromatin, nucleus breaks, ended with cell breaking down. Apoptosis can be triggered by cell injury or other apoptotic signals. During this process the cells experience a series of complex biological and chemical reactions [74]. The apoptotic pathways receiving the signals are shown in Figure 6.

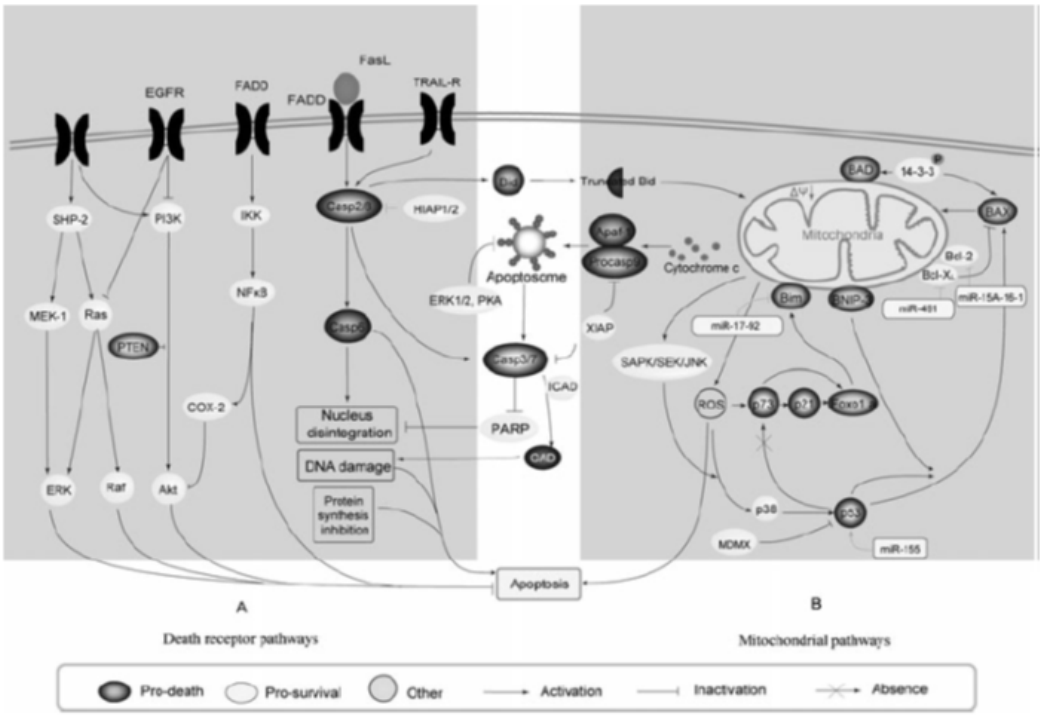


Figure 6: Two main types of pathways to induce apoptotic: death receptor pathways and mitochondrial pathways [11]. The death receptor pathways are triggered by receptors on membrane when death stimuli occur. The mitochondrial pathways are controlled by intracellular signals.

For clarity in subsequent reading, the model developed here focuses on the physical parameters of single cell and how they affect the emergent behaviour of group of cells. Thus, the size and shape of cell is the most important property to the model during the cell growth process. In the physics-based model outlined in this chapter, the cell cycle is simplified. The change in shape and size of cells will be used instead of the real biological behaviour in the cycle of modelled cells. Thus, the underlying biology is modelled phenomenologically.

The quiescent phase is simplified as an idle stage in the model, in which the size of the modelled cell does not increase. The interphase is simplified as the growing stage, in which means the size of the modelled cell increases. Finally, the mitotic phase is simplified as the split stage

in the model, in which one modelled cell divides to two identical daughter cells. Apoptosis is simplified as a stage in model as well, in which the volume of cell is considered as zero. In this way the modelled cell may transfer between simplified stages. The details of this simplified cell cycle are in Section 5.1. Also, there is an assumption here: when the modelled cell is in the growth status, the increase of its volume is linear.

Population growth: The rate that a group of cells increases their total number also represents the effect of environment on the cell. The total number of cells is an important system-level characteristic often considered in models.

A population growth curve can directly demonstrate the growth states of cells and is widely used in the research of cell growth and regulation. In (Zhao et al. 2007) it is used to show the effect of YAP. It is used in [37] to demonstrate how MagT1 restores the growth of TRPM7−/− cells. Under perfect conditions (sufficient nutrition, oxygen and space) the cell exponential growth *in vitro* experiment. Drawing number of cells over time gives the exponential curve based on e (natural logarithm) Figure 7.

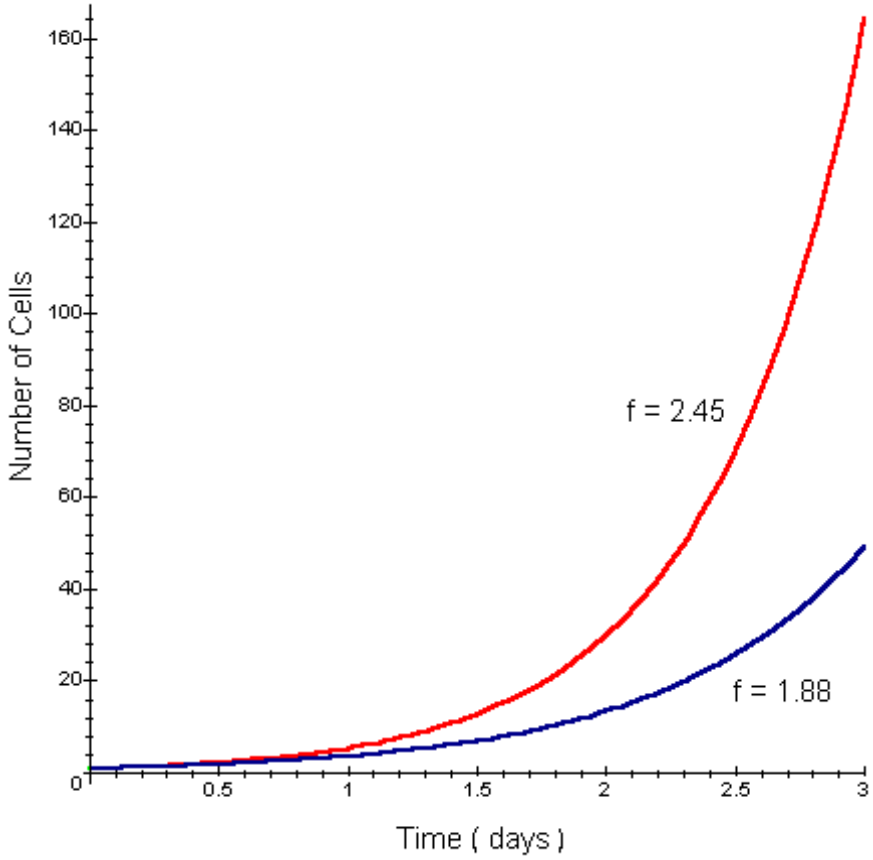


Figure 7: Exponential growth of cancer cell in 3 days. The value of f is frequency of cell cycles per unit time (1/day). (http://www.tiem.utk.edu/~-gross/bioed/webmodules/celldivision.html).

However, in real *in vitro* or in vivo growth, cell growth does not exactly follow the exponential rule because they reflect events from a heterogeneous population of cells. The cell cycle of a single cell is regulated by various factors, and different cells experience different factors and respond to those factors in different ways. For instance, on cell-cell contact, the protein kinase Hop (MST1/2) inhibits DNA replication (Zeng and Hong 2006). The cell cycle is arrested by gene p53 in Gap 1 on DNA damage [76]. A failed attempt of DNA restoration leads to programmed cell death (apoptosis). For a group of cells, the cell population is affected by both cell proliferation and apoptosis. If a group of cells are in a damaging environment (radiation, hypoxia, etc.), the effect of the environmental factors on each single cell is also demonstrated as the changes to the group population, resulting in various shape of growth curves Figure 8.

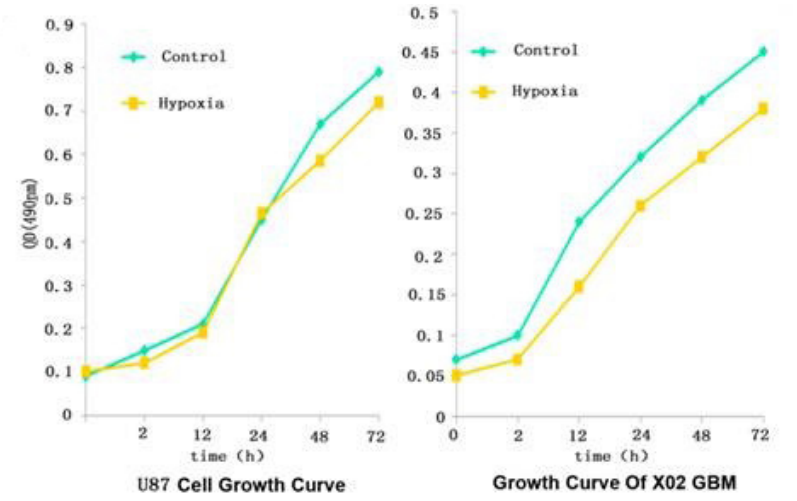


Figure 8: Growth curves of U87 cell and X02GBM cells [33]. The shape of control curve (blue) and hypoxic curve (yellow) are different for both cell lines.

Software development approach

In Sections 2.2 the modelling approach and examples were discussed. In Section 2.3 the biological and physical aspects of single cell behaviour were outlined. An agent-based model easily leads to object-orient programming (OOP) (https://en.wikipedia.org/wiki/Object-oriented_programming), in which one type of agent is represented by a 'class' that containing all data properties and functions of agent , while all the agents of the same class are 'objects'. Each agent object maintains its own properties by calling its functions, and in the *in silico* experiment a list of agent object is maintained.

In my software development approach, the physical properties of agent, such as position, velocity and force etc. Are packed into class 'Ellipsoid'. Thus, the endothelial agents are of class 'Ellipsoid'. The program starts from the first agent A, and loops through all the Ellipsoid agents to calculate the total cell-cell force and cell-cell torque on it. Then the force and torque between A and substrate are calculated and added with cell-cell force and torque to form sum force and sum torque. After that the sum force and sum torque are used to calculate the velocity and angular velocity of agent. With the initial position and direction of agent A (considered as known), and calculated velocity and angular velocity, the new position and direction of agent A is then calculated and updated. The program then turns to the next agent, and then the next agent, until all the agents are updated, as shown in Figure 9.

Cancer cells, on the other hand, because they still have all the physical properties, are of a class named 'Cell' which is 'derived' from class 'Ellipsoid' and contains all the properties and functions of class 'Ellipsoid'. The biological properties and functions are added to class 'Cell', so that the cancer cell can be aware of its age and growth status. Please refer to Appendix C: code and simulation issue for more detail of code structure.

When considering construction of agent-based models often computational limitations come into play. A key problem is that in order to calculate the physical forces impacting each agent, it is necessary to calculate those forces involved in the interactions between this agent and all other agents. Thus, as the number of agents increases, the increase in the number of calculations is not linear but squared. To reach realistic numbers in a population of agents in one simulation the running time of that simulation becomes unacceptably long.

Parallel computing technique is one potential tool to resolve this problem. By implementing these calculations concurrently, the simulation time can be significantly reduced, as Figure 12 shows. Note that the update of each agent state, including spatial position and any internal state, should happen after the all the calculations are done, therefore the calculation of physical properties of one agent does not affect its position. In other words, calculation of properties of one agent should not affect calculation of properties of other agents so there are no order effects Figure 10.

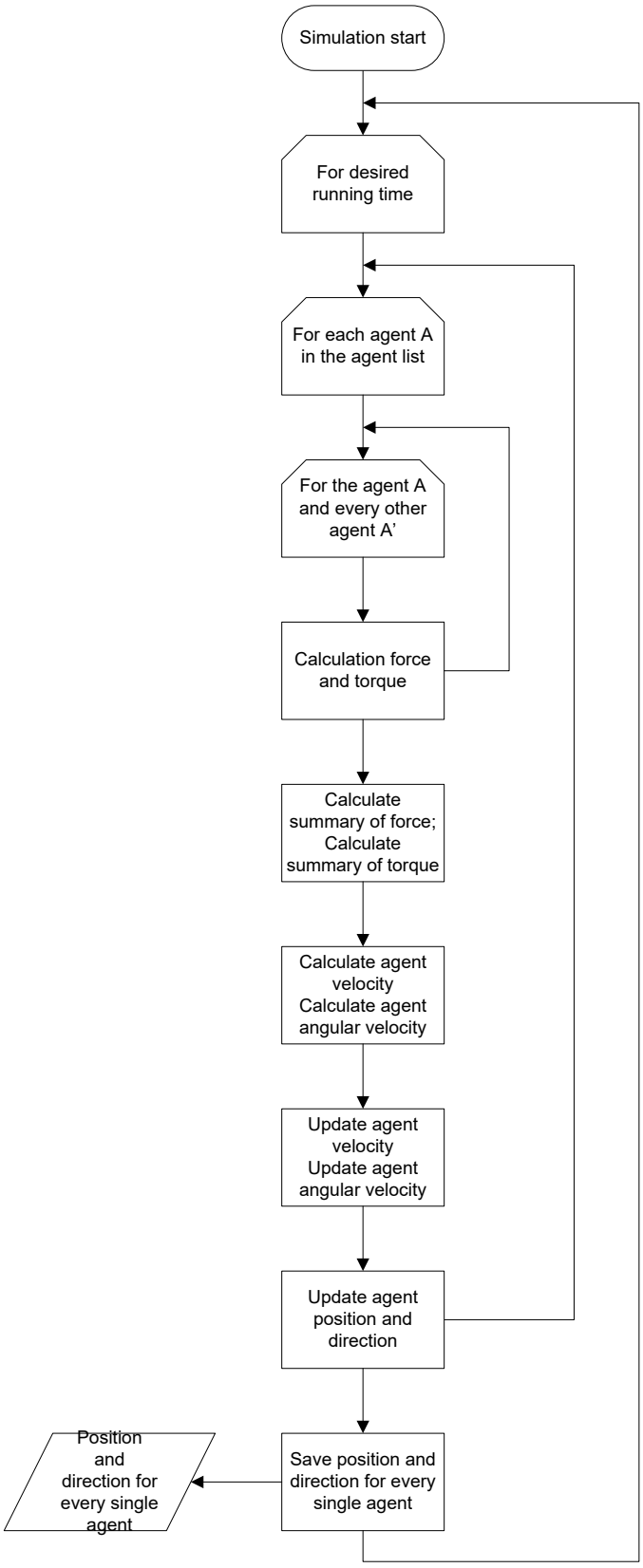


Figure 9: Process to calculate and update physical properties of all agents.

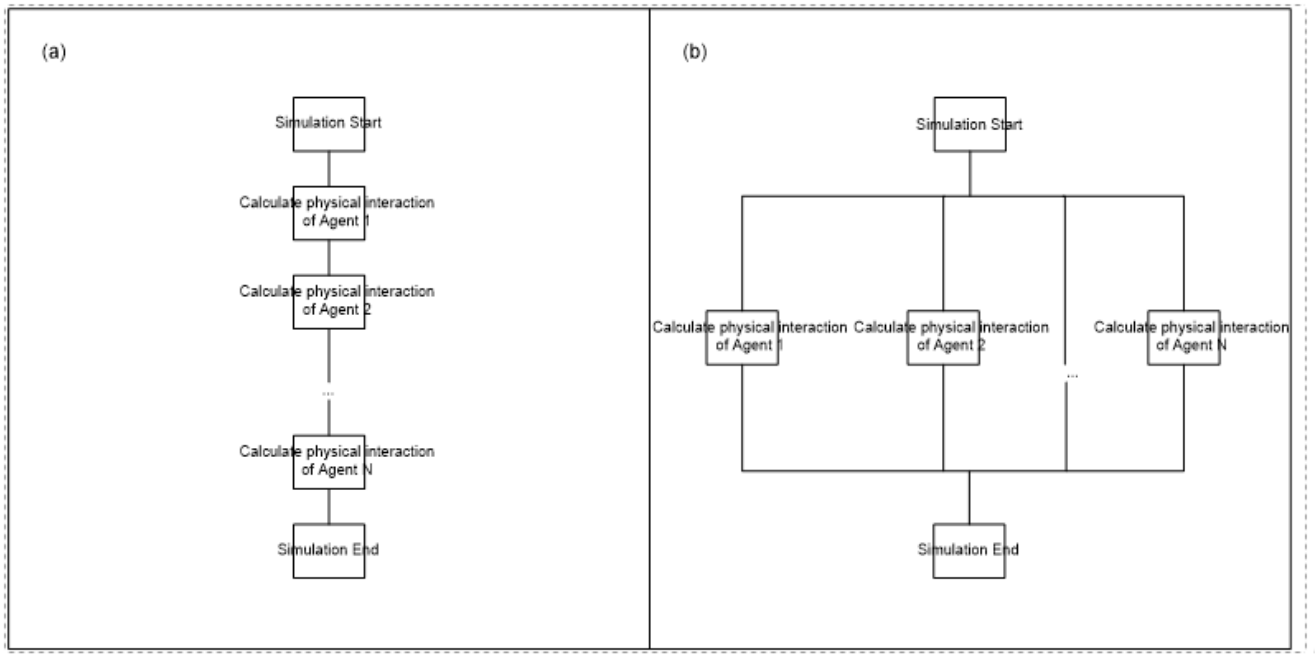


Figure 10: Calculation of physical properties with (a) and without data parallel (b).

Thread Building Blocks (TBB, https://www.threadingbuildingblocks.org/) is a parallel library released in 2006 by Intel Co. The latest version is 4.2. Unlike some other parallel interface such as MPI (http://www.mpi-forum.org/), it does not provide direct control over threads of execution [77]. Instead, TBB maps the calculation task onto threads automatically. It is useful in simulation in which a large amount of data needs to be processed in the same manner. Note TBB uses C++ code, which provides good links into the simulation code developed here.

Most importantly, the TBB library maps logical workload onto threads, which is a straightforward process for an agent-based model as the workload for each agent is already designed. At the same time, introducing TBB parallel library requires minor change of original simulation code structure. For my agent-based model, the TBB library maps calculation of each single agent to multi-core processors, as Figure 11 shows.

There is one potential problem with this solution: the TBB library is supported by Intel multi-core processors, on non-Intel processors the performance of simulation may not be as good.

Methods calibration methods

The spatial structure produced in simulation should be compared to the experimental data. In the early stage, the image of simulation can be directly compared with experimental data as a quality analysis, which helps to find mistakes in models. However it is far from enough to describe cell behaviour, so that is not efficient in comparison the simulation result of model with *in vitro* experiments, while the comparison is parts of process to fit experimental data into model [78].

In this section, quantity analysis tools are discussed, which will be used to explore the features of the spatial structure, of both experimental data and simulation result. Also, these tools will be used to calibrate simulation to experimental data. The calibration is discussed in Section 6.

Percolative transition: Percolation theory [79] is used to study the connectivity of random generated structure, in which there is a control parameter n_c. For all $n < n_c$, the whole structure contains only un-connected clusters. As the value of n increases, the structure is still un-connected, until $n = n_c$. When $n = n_c$ a transition happens, and the structure is connected.

It is necessary to study percolative transition because (a) vessel network should allow blood to travel through to carry nutrients and oxygen to tissues, thus percolation is an essential feature of vessel network; (b) the value critical exponents can be used as a feature of the system [80].

Percolative phase transition is used in Chauhan et al, 2012, in which the control parameter is the number density of cells. With low number density, the endothelial cells can only form isolated clusters, while with number density higher than a connected structure is formed, as shown in Figure 12.

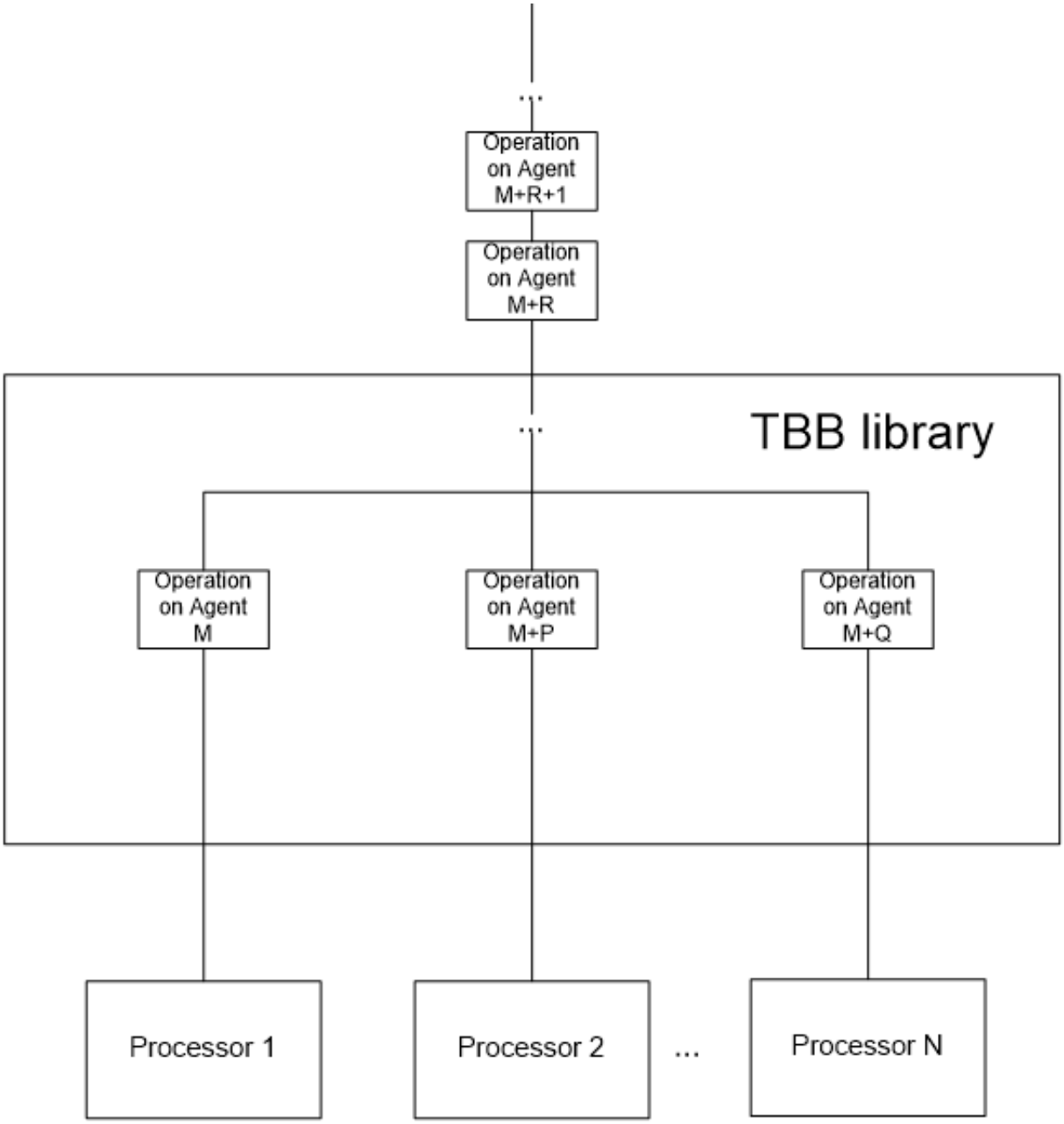


Figure 11: Concurrent operation of agent mapped onto N processors by TBB library.

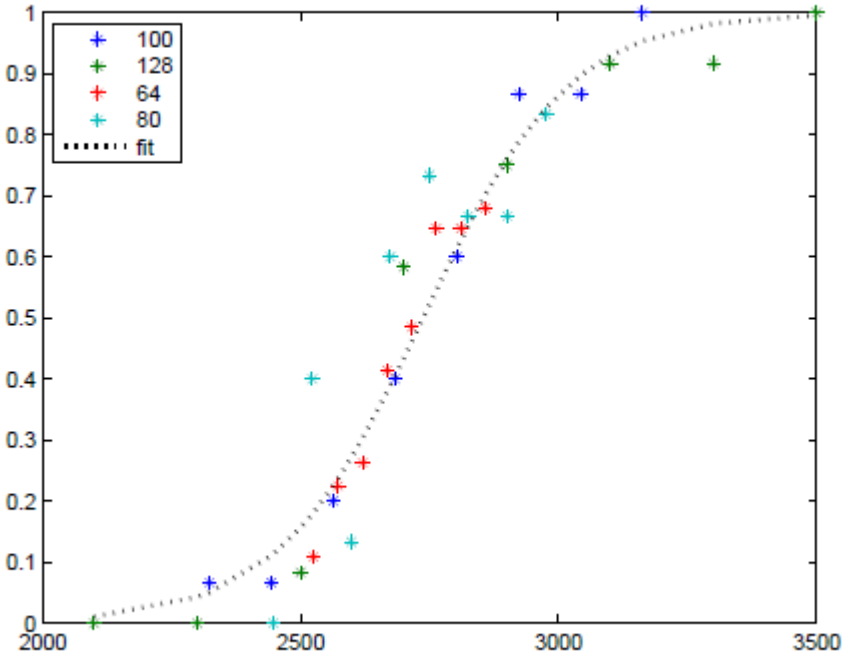


Figure 12: Percolation probability at various number density of cells [109].

X-axis is the number of cells per m^3, y-axis is probability of connection. The percolative transition happens at number of densities = 3000. '100', '128', '80' and '64' are size of observation window in four *in silico* experiments. Because all the *in silico* experiments have the same dynamic feature, all the plots can be fitted to one percolation curve.

Spatial statistic model: The history of point process can be tracked back to 1662 [81]. Point process statistics is a tool to analysis the geometry structure formed by objects that distributed in one-, two-, or three-dimension space [82]. In point process studies, the particles are abstracted as points and marks. The points are at the location of objects, while the marks contain additional information about objects, such as shape and size.

First order property: The intensity at a point in space can be considered as the mean number of events (points) in an area, i.e. a density measure. As equation (1) shows, $E(Y(d_s))$ is expected number of points, d_s is area.

$$\lambda(s) = \lim_{ds \to 0} \{ \frac{E(Y(d_s))}{d_s} \} \qquad (1)$$

The intensity is a first-order property, which measures the distribution of event (agents in this study) [83]. It is a 'local' measurement, because it cannot give information about interactions among points. The interaction among points, for example, clustered or regular, is measured by second-order properties [83].

Note that the equation in (1) is the theorical definition of the intensity, the value of intensity has to be estimated from experimental data or simulation result.

Second order properties: Ripley's K-function and pairwise function are second-order properties of the spatial structure of the points. Second-order properties measure the strength and type of interactions between points process [83]. There are higher-order properties, which are more complex and hence more difficult to estimate [82]. Therefore higher-order properties are not discussed in this chapter.

The K-function is a method to measure a distribution based on distance. For a given point, the value of the K-function equals the number of points that can be found within a circle with this point as centre and radius *d*. If the intensity at this point is known, then the K-function should be equal to intensity multiplied by area of this point.

For a point process, the K-function at distance d equals sum of the K-function of every single point in this point process at distance d multiplied by intensity then divided by area of circle with radius of *d*.

$$K(d) = \int_{\rho=0}^{d} g(\rho) 2\pi\rho d\rho \tag{2}$$

Note that the value of K-function has to be estimated from real value (gathered from experiments or simulation).

For a distribution that is complete random or stationary Poisson process, the expected number of events exponentially increases over distance. Thus, the K-function curve of a stationary Poisson process is an exponential curve, and this serves as a baseline for interpreting plots. For an arbitrary curve, the part above an exponential curve represents the clustered distribution at the corresponding distance; on the contrary, the part of curve below the exponential curve represents the dispersive distribution at the corresponding distance.

The pairwise correlation function *g* is another tool to analyse a distribution. It relates to the K-function as follows

$$g(d) = \frac{K'(d)}{2\pi d} \tag{3}$$

It is the probability of finding a pair of points at distance of d, divided by the corresponding probability for a completely random distribution (Poisson process) [84].

The pairwise correlation function has been recently used in studies about the spatial structure of cell aggression [85,86,137], by analysing data from both experiments and simulation. These studies have illustrated the potential of pairwise correlation function to be used to explore the feature of pattern of cells.

Same with the intensity and K-function, the value of pairwise correlation has to be estimated from experimental data or simulation result. The value of pairwise correlation *g(d)* is always larger than *0*. The points are considered clustered on distance d where $g(d) > 1$ and dispersed on distance *d* where $g(d) < 1$. For distance where $g(d) = 1$, the point system is randomly distributed at distance *d*.

The pairwise correlation function can be drawn with distance *d* as x-axis and value of *g(d)* at d as y-axis. The shape of pairwise correlation function can also provide information about the feature of the pattern. As shown in Figure 13, the solid line of $g(r) = 1$ represents a typical completely random distributed point process. The dashed line ($g(r) > 1$) represents a typical clustered point process, because in a clustered system all the points are within a short distance, so that it is more likely to find a pair of points with short distance than points with long distance. The dotted line, on the other hand, represents a typical regular distribution, as for regular distribution, pair of points tend to appear with certain distance, which makes the pole.

A typical point process with repulsion is shown in Figure 14, which shows multiple poles. The first pole shows the typical inter-point distance, while the second pole shows the typical distance between a point and its long-distance neighbours; the first non-zero minimum value that between the first and second poles suggests the typical distance between a point and area with small number of points [82].

Conclusion

As discussed in Section 2.2, the physical aspect of agent behaviour, including the representation of agent, calculation of agent interaction and motility will be modelled following agent-based modelling approach in the next section. Then in sections 4 and 5, the biological aspects specific to two different biological systems are discussed, including the set of biological assumptions and the experiment designed to calibrate the biological model. Based on the same experiment and the results shown in section 5, the calibrati

on of the physical aspects, in addition to the biological aspects, is discussed in section 6 using pairwise correlation function.

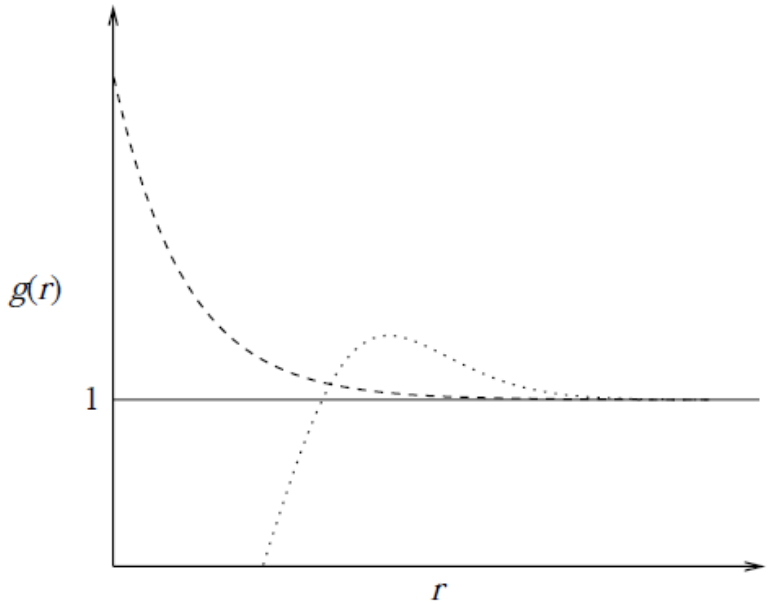


Figure 13: The typical shape of pairwise correlation functions of a completely random distribution (solid line), a clustered process (dashed line), and a regular process (dotted line) [82].

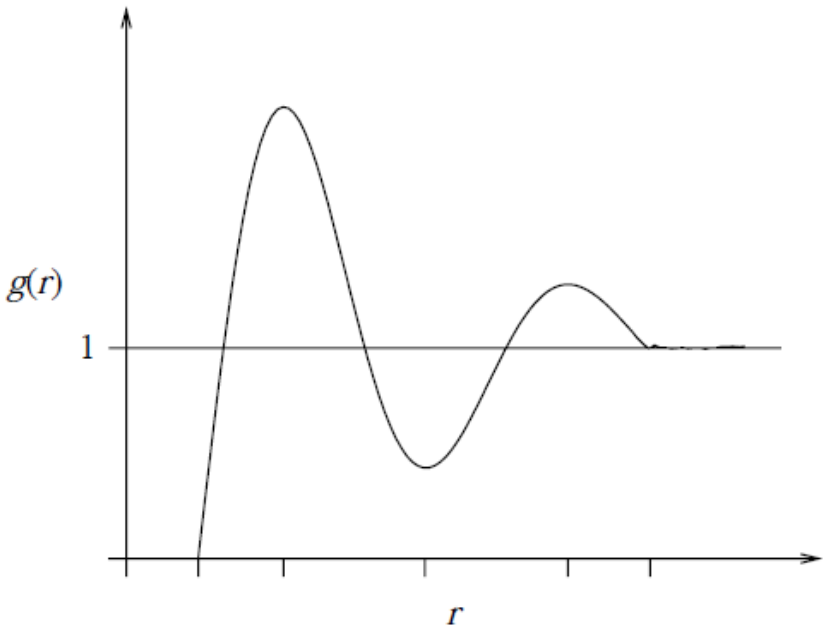


Figure 14: A typical pairwise correlation function of repulsive distribution [82].

According to Section 2.3.1, this model is built based on the assumption that agents move until the whole agent system reaches its lowest energy status. In Section 3 I begin my account of the model with the calculation of the energy of system. The agents are represented as ellipsoidal particles, and the reasoning behind this is outlined in Section 2.3.1. Therefore, in Section 3.1, I start with the mathematical representation of an ellipsoid, including its shape, size and direction. Then, taking an example of a two-agent system, a method is demonstrated in Section 3.2 to calculate the potential from the geometry information and relative position of the two ellipsoidal agents, which is an extension of derivations in [62,64].

As one of the purposes of this model is to study the effect of cell shape to migrant pattern as emergent behaviour, the particle is too simple; the polymer of elastic components is too complex. The ellipsoid is chosen to represent the geometry shape of cell. As ellipsoid is anisotropic shape, instead of distance between pair of cells, the contact potential is used to describe cell-cell interaction. Then the early simulation shows that the motion of modelled cell is non-realistic, which results in the usage of Hertz formula. As Hertz model works with spheroidal particles, an extension is made to use Hertz formula for heterogeneous and for ellipsoidal particles. The detail of extension is discussed in Section 3.

The adhesive interaction is abstracted as a force which exists between each pair of cells. The adhesive force affects both cells equally and tends to drag them to move towards each other. There is adhesion force between the cell and petri-dish as well [87]. Similarly, adhesion force between one single cell and petri dish moves the cell towards surface of petri dish. I assume that it can be affected by both environmental factors and intra-cell factors.

I not only need to estimate the cell-cell interaction, the cell-petri-dish interaction is also required to be estimated. Because the interaction of two identical cells can be considered mirror symmetry, the interaction between one cell and the substrate plane can be considered as the interaction between this cell and its mirror image. Thus, I consider it is proper to calculate the interaction between cell and petri-dish in the same way of calculating pair of cells. There are detailed ellipsoid-plane interaction formulas [71], but my method keeps the consistency of calculation of interaction of cells. At last, in the model I assume all the cell dynamic happens on surface of petri dish. Thus, a monolayer of modelled cells is arranged on the same plane. This assumption requires additional rule applied to cells: I have to assume the interaction between modelled cell and plane is much larger than interaction between pair of two cells to prevent them from piling up (it also restricts the behaviour of the model).

The real cell increases its size during growth, and it divides into two daughter cells when its volume is almost doubled. To simplify this process, the modelled cells cannot change their shape. But they can gradually increase their volume until it is exactly doubled. The scale of overall shape is maintained. In this way the modelled cells can be used to simulate growth behaviour and I can focus on the growth status of modelled cells. This part is discussed in Section 4.

In this model, there are effects of chemotaxins such as 5-FU, however I assume that it is evenly distributed in petri dish and has equal effect on every single cell (agent). Therefore, instead of having an evenly distributed radiant field, the mechanism of the drug is simplified and equally added to rules that define the behaviour of each single cell (agent). In real cell dynamic (or treatment) the chemotaxins cannot be extremely evenly distributed, the evenly distribution assumption is another simplification, which helps me to focus on the effect of certain concentration of drug to the cells.

Also in Section 2.2.5, the effect of oxygen on tumour cell metabolism is studied in [44], which slows cell cycle process and relates to secreting of chemotaxins such as VEGF-A. Thus, the effect of hypoxia should be included in the model. While [88] provides a view of oxygen metabolism from a biological point of view in Section 2.2.5, the method to model hypoxia is discussed in Section 5.

Based on discussions in Section 2, I make following assumptions:

	Explicit Assumptions	
1	All the cell dynamic happens on surface of petri dish;	To produce an intuitive pattern
2	The interaction between modelled cell and plane is much larger than interaction between pair of two cells:	Scianna *et al.* [25]
3	5-FU is evenly distributed in petri dish and has equal effect on every single cell;	For simplicity of calculation
4	Ellipsoidal shape;	R Sodt, et al. [100]. Hayashi, et al. [51]
5	The ratio of three semi-axis of ellipsoid keeps the same during the cell growth (the shape of cell does not change but the size changes);	For simplicity of calculation

6	Each cell is considered under effect of resistance force which occurs when cell moves within the fluid;	Beysens, *et al.* [68]
7	The degree of cell overlapping represents the elasticity of cell;	To keep a relatively simple representation of cell
8	The density distribution within cell is even and therefore in my model the forces are affected on geometry centre of ellipsoidal cell;	For geometric simplicity
9	The fluid in which cells grow and migrate is still, i.e. the fluid has no velocity or angular velocity.	It is the case in *in vitro* experiments
10	The growth of cell volume is linear;	For simplicity of calculation
11	No upper limit of total number of cells;	The length of experimental time is short, in which the cells have not reached the upper limit of total number.
12	The cell age obeys normal distribution;	Drasdo & Hohme [43]
	Implicit Assumptions	
13	Only contact force, adhesion force and resistance force are significant forces on the cell (density of cell is similar with water thus gravity does not need to be considered).	Grover, *et al.* [113]
14	Contact force can be computed using the Perram-Wertheim approach;	John W Perram, *et al.* [62]
15	Cells move at very low speed, so their acceleration approaches zero and the forces upon them are balanced (precondition of using Stokes resistance law);	JP Rieu, *et al.* [69], GM Walker, *et al.* 2005
16	Resistance force can be computed using Stokes' law, and the known properties of the fluid medium.	H Brenner [60]

Section 3: Inter-Cell Interaction Model

Introduction

Following the reviews in Section 2.3, physical interaction plays an important role in tumour morphology. In Section 3, the detailed physical interaction is modelled. Then in Section 4 and 5, cell cycle and population growth of two different biological systems are discussed, including the set of biological assumptions and the experiment designed to calibrate the biological model. Based on the experiment result, the cell cycle and population growth are calibrated in Section 5, and then the calibration of the physical aspects is discussed in Section 6.

As discussed in Section 2.4.1, this model is built following the agent-based process. Thus, physical interaction is presented in the form of a set of rules that act on each agent. In other words, the set of physical rules are considered as the physical behaviour of each agent, while the result of interaction considered as emergent behaviour of group of agents. In Section 2.4.3, the necessity of considering real cells as ellipsoidal particles is explained. Thus, in Section 3.1, the building of physical model starts with representing the geometry information, including its shape, size and direction, of an arbitrary ellipsoid. Although it is much easier to model all the agents as identical particles, the tumour tissue may contain various types of cells, which may in various phases of cell cycle. Due to the heterogeneous feature of tumour tissue, this model is required to be able to represent mixed agents with various shapes, size and cell cycle.

According to Section 2.4.4, this model is built based on the assumption that agents move until the whole agent system reaches its lowest energy status. Then in Section 3.2, taking example of a two-agent system, the mathematical form of potential between pair of arbitrary ellipsoidal particles is derived from the geometry information and relative position of the two agents, which is an extension of derivations in [62,64].

According to section 2.4.4, the energy can be derived from potential by using Metropolis algorithm; following this approach, and detailed in section 3.3 and 3.4, the potential is firstly transformed to contact energy and adhesion energy. According to its physical interpretation, the contact energy is transformed to contact force, and the adhesion energy is transformed to adhesion force. Similarly, the contact torque and adhesion torque are calculated from the corresponding energy in Section 3.5.The adhesion force and torque work as contraction effectors, while the contact force and torque work as repulsion effectors. Similar with [24], I assume that the cells move in a low speed that there is no acceleration, thus the sum of force is directly used to calculate velocity of agent; and the sum of torque can be used to calculate angular velocity of agent, as explained in Section 3.6. The velocity and angular velocity respectively in 3.6, which at last are transformed to position and direction of each agent, are shown in section 3.7.

As I can see, the calculation of physical interaction for each agent is relatively complex, which is another reason that parallel computing technique is necessary for large-scale simulation.

Finally, to ensure the correction of physical interpretation between all the transforms and estimate the value of constants in the formula, a dimension analysis is carried out. The correction of dimension of equations in the model is checked in Section 3.9, and the values of parameters in the model are estimated in Section 4 and 5 separately as the model is used in these two sections for different purposes.

Representation of a cell

Given my rationale for representing cells as ellipsoids, it is necessary to represent the cell geometrically. As Figure 15 shows, an ellipsoid has one centre point, and three semi-axes. Note that the ellipsoid is axisymmetric along the three semi-axes: I assume that the mass is evenly distributed over the ellipsoid so all of the forces can be considered as acting on the centre of the ellipsoid.

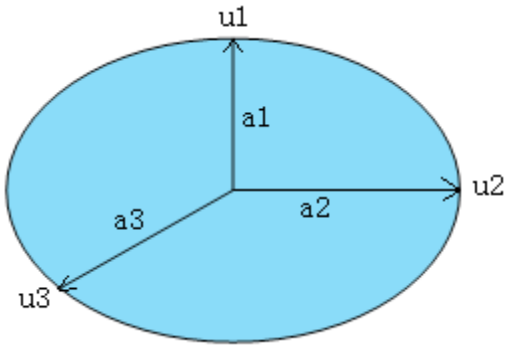


Figure 15: An ellipsoid. The length of the three semi-axes are a_1, a_2 and a_3 ($0 < a_1 < a_2 < a_3$), the direction of the three semi-axes are $\mathbf{u}_1$, F_D and d_0.

I consider the three semi- axes as three vectors in space, so that they can be written in the form of three-unit vectors to represent their direction, and three values to represent their length. The three-unit vectors are referred as $\mathbf{u}_1$, $\mathbf{u}_2$ and $\mathbf{u}_3$, and their corresponding length are referred as a_1, a_2 and a_3. From the characteristics of an ellipsoid, it is known that $\mathbf{u}_1$, $\mathbf{u}_2$ and $\mathbf{u}_3$ are orthometric. For simplicity, I label the shortest length a_1 and longest a_3, i.e. $0 < a_1 < a_2 < a_3$.

To represent the ellipsoid in a single expression for convenience, $\mathbf{u}_1$, $\mathbf{u}_2$ and $\mathbf{u}_3$ are combined together as a matrix:

$$\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3) = \begin{pmatrix} u_{11} & u_{12} & u_{13} \\ u_{21} & u_{22} & u_{23} \\ u_{31} & u_{32} & u_{33} \end{pmatrix} \tag{4}$$

In which each column represents the direction of one semi-axis. Further, a_1, a_2 and a_3 should be combined with their corresponding direction, so that the ellipsoid A can be written in the form of $\mathbf{A} = a_1^{-2}\mathbf{u}_1 \otimes \mathbf{u}_1 + a_2^{-2}\mathbf{u}_2 \otimes \mathbf{u}_2 + a_3^{-2}\mathbf{u}_3 \otimes \mathbf{u}_3$, in which $\otimes$ is the outer product. As per the matrix form used in [64].

$$\mathbf{A} = \mathbf{U}\bar{\mathbf{A}}\mathbf{U}^T = \begin{pmatrix} u_{11} & u_{12} & u_{13} \\ u_{21} & u_{22} & u_{23} \\ u_{31} & u_{32} & u_{33} \end{pmatrix} \begin{pmatrix} \frac{1}{a_1^2} & 0 & 0 \\ 0 & \frac{1}{a_2^2} & 0 \\ 0 & 0 & \frac{1}{a_3^2} \end{pmatrix} \begin{pmatrix} u_{11} & u_{21} & u_{31} \\ u_{12} & u_{22} & u_{32} \\ u_{13} & u_{23} & u_{33} \end{pmatrix} \tag{5}$$

In which $\mathbf{U}^T$ is the transposition matrix of the matrix $\mathbf{U}$, and $\overline{\mathbf{A}}$ is the eigen matrix of matrix $\mathbf{A}$. It is obvious that the matrix $\mathbf{A}$ is an orthometric matrix. Another observation is that the eigen matrix of an ellipsoid $\mathbf{A}$ with semi-axes a_1, a_2 and a_3 has the form of

$$\begin{pmatrix} \frac{1}{a_1^2} & 0 & 0 \\ 0 & \frac{1}{a_2^2} & 0 \\ 0 & 0 & \frac{1}{a_3^2} \end{pmatrix} \tag{6}$$

As matrix U represents the direction and $\overline{\mathbf{A}}$ represents the length of the three semi-axes of ellipsoid, consider an ellipsoid with its centre at the origin, all its geometry information is included by matrix A. In other words, matrix **A** represents an ellipsoid at the origin with direction **U** and semi-axes length of a_1, a_2 and a_3. This form of ellipsoid is used in all the following sections.

A point of note is the need for $\mathbf{U}^T$. The matrix **U** contains directional information. According to the cross-product rule, $\mathbf{U}^T$ is added so that the determinantal expansion of matrix form of the ellipsoid is the same as the expansion of the algebraic form of the ellipsoid.

Contact potential

For those models in which the geometry of cells is represented by a sphere, in order to tell the spatial relationship between any two cells, the distance between them is to be measured and compared with the sum of their radii. If the distance is larger than sum of their radii, then the two cells are not in contact; conversely, if the distance is less than this sum, the two cells are partly overlapped; and if the distance equals the sum, the two cells are just contact, or 'externally tangent', as Figure 16 shows.

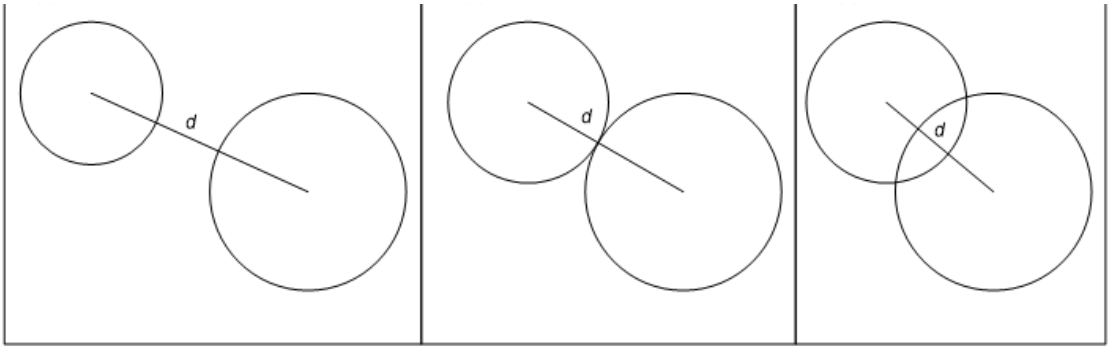


Figure 16: Two non-identical spheres. (a) For non-touching spheres, the distance between the centres is larger than the sum of radii of the spheres; (b) for the just-touching spheres, the distance between the centres equals the sum of radii; (c) for overlapping spheres, the distance between the centres is less than the sum of radii.

However, with an ellipsoid model, this simple approach is no longer valid, as orientation plays an important role in spatial relationships. As Figure 17 shows, while they are at fixed position, whether two ellipsoids are in contact or not depends on their orientation.

In order to calculate the spatial relationship of two ellipsoids, the concept of contact potential is introduced, which is discussed in section 2.2.1. The contact potential is the most important term in the model, as all the forces and torques are calculated from it. It means a certain level of energy. As shown in Figure 18, the dotted lines are potential surfaces around the ellipsoid cell, and any point on the same surface has the same potential (note the continuous space is discretized here for explanatory purposes).

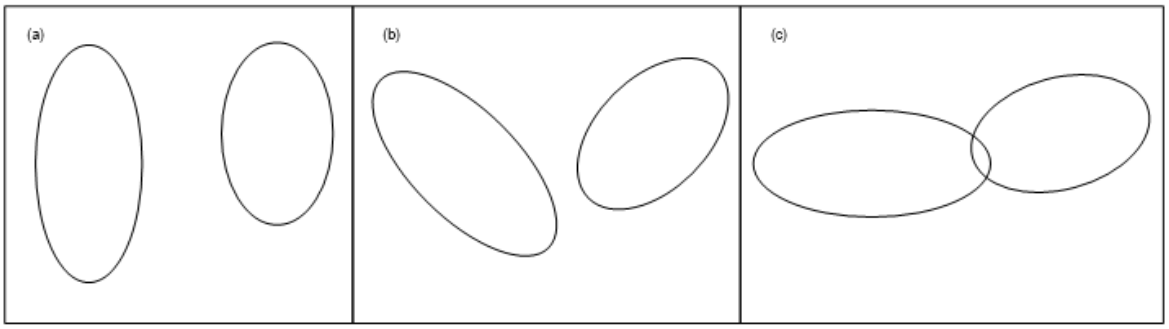


Figure 17: Two ellipsoids of arbitrary size at same position with various orientations. In (a-c) the distance between the centres of the pair ellipsoids is the same, but with various orientation the ellipsoids may be overlapped or not: (a) two ellipsoids lie in parallel, not in contact with each other.

(b) Two ellipsoids are not parallel and not in contact with each other.

(c) Two ellipsoids are not parallel and partly overlap.

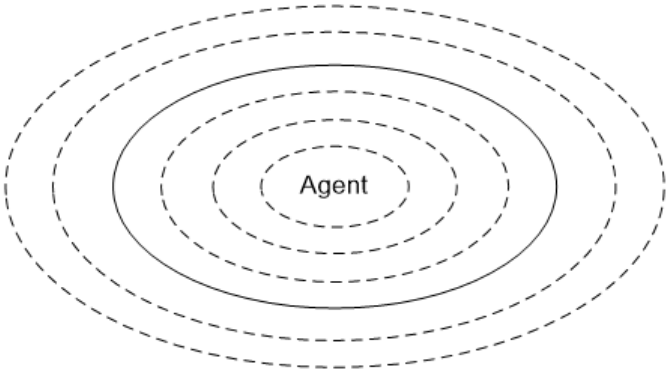


Figure 18: Potential surface of ellipsoidal agent. The solid line is the surface of the agent. The dot lines are potential surfaces.

With contact potential, the relationship between an arbitrary point in space and the ellipsoid can be described. First the value of potential of that point is calculated (as detailed below). The larger this value, the further away that point is from the ellipsoid. Inversely, the smaller this value is, the closer that point is to the ellipsoid. However potential is a relative value, which means that only the change of potential can be measured and valued. This relative nature of potential is not convenient to use for descriptions and so, for convenience here, let the centre of ellipsoid have potential value of zero, and the surface of the ellipsoid have potential value of one. Given this assumption, the contact potential is treated as follows.

Assume an ellipsoid $\mathbf{A}$ can be represented as $\mathbf{A} = a_1^{-2}\mathbf{u}_1 \otimes \mathbf{u}_1 + a_2^{-2}\mathbf{u}_2 \otimes \mathbf{u}_2 + a_3^{-2}\mathbf{u}_3 \otimes \mathbf{u}_3$. The contact potential of an arbitrary point $\mathbf{r}$ from A can be represented as $F_A(\mathbf{r}-\mathbf{r}_A) = (\mathbf{r}-\mathbf{r}_A)^T \mathbf{A}(\mathbf{r}-\mathbf{r}_A)$ [63], where $\mathbf{r}_A$ is the position vector of the centre of ellipsoid $\mathbf{A}$, and $\mathbf{r}-\mathbf{r}_A$ represents the vector starting from centre of ellipsoid $\mathbf{A}$ pointing to the arbitrary point $\mathbf{r}$. The spatial relationship between $\mathbf{r}$ and $\mathbf{A}$ can be easily determined as:

$$F_A(\mathbf{r}-\mathbf{r}_A)\begin{cases} <1 \ for \ \mathbf{r} \ \text{inside} \ A \\ =1 \ for \ \mathbf{r} \ \text{on the surface of} \ A \\ >1 \ for \ \mathbf{r} \ \text{outside} \ A \end{cases} \quad (7)$$

In which **r** is the vector that represents the position of the arbitrary point, $\mathbf{r}_A$ is the vector that represents the position of centre of ellipsoid *A* (Figure 19).

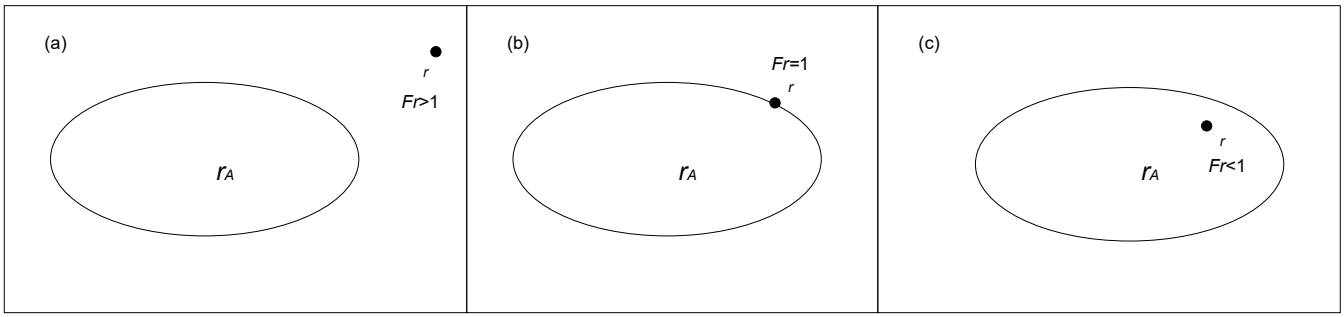


Figure 19: One point in space, r, and one ellipsoid with value of F_A (a) point outside ellipsoid, $F_A > 1$; (b) point on ellipsoid, $F_A = 1$; (c) point inside ellipsoid, $F_A < 1$.

This means that if the contact potential value of point **r** is higher than one, **r** is outside the ellipsoid *A*; if contact potential value of point **r** is lower than one, **r** is inside the ellipsoid *A*; and finally, if contact potential value equals one, **r** is on surface of the ellipsoid *A* (Figure 20).

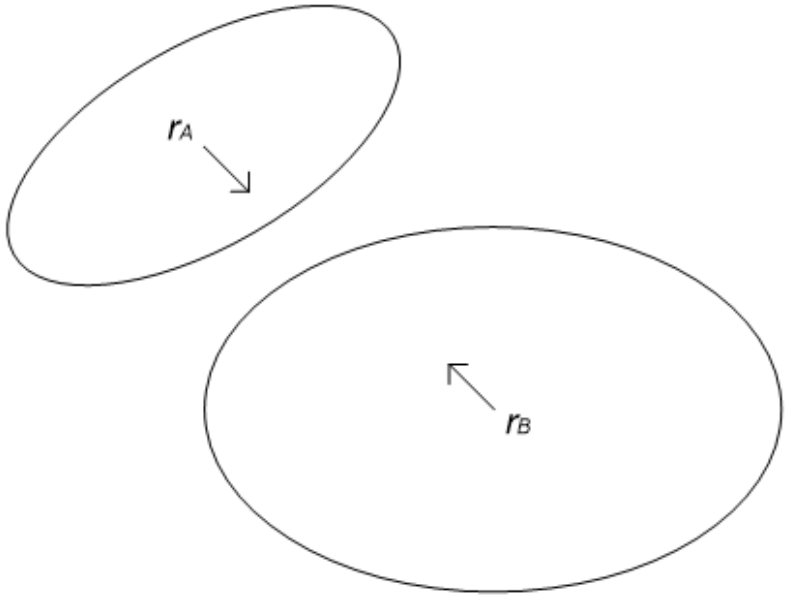


Figure 20: Two arbitrary ellipsoids and force on centre.

Now consider the spatial relationship of two arbitrary ellipsoids (cells) *A* and *B*. As discussed, their direction can be represented by matrices **U** and **V** respectively, in which the column vectors of each matrix represent the direction of semi-axis of each ellipsoid.

$$\mathbf{U} = \left(\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3\right) = \begin{pmatrix} u_{11} & u_{12} & u_{13} \\ u_{21} & u_{22} & u_{23} \\ u_{31} & u_{32} & u_{33} \end{pmatrix} \quad (8)$$

$$\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3) = \begin{pmatrix} v_{11} & v_{12} & v_{13} \\ v_{21} & v_{22} & v_{23} \\ v_{31} & v_{32} & v_{33} \end{pmatrix} \quad (9)$$

where a_1, a_2 and a_3 are the radii of ellipsoid A and b_1, b_2 and b_3 are the radii of ellipsoid B and they satisfy

$$0 < a_1 < a_2 < a_3 \quad (10)$$

$$0 < b_1 < b_2 < b_3 \quad (11)$$

Define each ellipsoid by the direction and length of all its semi-axis. Thus, A and B can be written as

$$\mathbf{A} = a_1^{-2}\mathbf{u}_1 \otimes \mathbf{u}_1 + a_2^{-2}\mathbf{u}_2 \otimes \mathbf{u}_2 + a_3^{-2}\mathbf{u}_3 \otimes \mathbf{u}_3 \quad (12)$$

$$\mathbf{B} = b_1^{-2}\mathbf{v}_1 \otimes \mathbf{v}_1 + b_2^{-2}\mathbf{v}_2 \otimes \mathbf{v}_2 + b_3^{-2}\mathbf{v}_3 \otimes \mathbf{v}_3 \quad (13)$$

The determinant calculation of both matrices **A** and **B** are larger than 0, thus it is easy to verify that the inverse matrices are

$$\mathbf{A}^{-1} = a_1^2\mathbf{u}_1 \otimes \mathbf{u}_1 + a_2^2\mathbf{u}_2 \otimes \mathbf{u}_2 + a_3^2\mathbf{u}_3 \otimes \mathbf{u}_3 \quad (14)$$

$$\mathbf{B}^{-1} = b_1^2\mathbf{v}_1 \otimes \mathbf{v}_1 + b_2^2\mathbf{v}_2 \otimes \mathbf{v}_2 + b_3^2\mathbf{v}_3 \otimes \mathbf{v}_3 \quad (15)$$

For an arbitrary point **r** in space, its potential to each ellipsoid can be written as

$$F_A(\mathbf{r} - \mathbf{r}_A) = (\mathbf{r} - \mathbf{r}_A)^T \mathbf{A}(\mathbf{r} - \mathbf{r}_A) \quad (16)$$

$$F_B(\mathbf{r} - \mathbf{r}_B) = (\mathbf{r} - \mathbf{r}_B)^T \mathbf{B}(\mathbf{r} - \mathbf{r}_B) \quad (17)$$

Next the potential of arbitrary point **r** is derived against ellipsoid **A** and **B** based on (13) and (14), thus the equation representing the potential between ellipsoid **A** and **B** should contain both $F_A(\mathbf{r} - \mathbf{r}_A)$ and $F_B(\mathbf{r} - \mathbf{r}_B)$. Similar to the equation 3.28 of [89], a decimal λ between 0 and 1 is introduced (the calculation of λ is in Appendix A), then write the potential of point **r** against ellipsoid **A** and **B** as

$$F(\mathbf{r}, \lambda) = \lambda F_A(\mathbf{r} - \mathbf{r}_A) + (1 - \lambda) F_B(\mathbf{r} - \mathbf{r}_B) \quad (18)$$

In the equation $F(\mathbf{r}, \lambda) = \lambda F_A(\mathbf{r} - \mathbf{r}_A) + (1 - \lambda) F_B(\mathbf{r} - \mathbf{r}_B)$, the decimal λ >=0, which means both ellipsoids contribute to potential of point **r**. If λ =0, it means that the point **r** is actually on center of one of ellipsoids. Note that there should be no fixed order of ellipsoid **A** and **B** in this equation, it can also be written as $F(\mathbf{r}, \lambda) = \lambda F_B(\mathbf{r} - \mathbf{r}_B) + (1 - \lambda) F_A(\mathbf{r} - \mathbf{r}_A)$, because the potential of a fixed point should be fixed and is only relative to the geometry relationship between fixed point and ellipsoid, not order of ellipsoid. Substitute (13), (14) into equation (15)

$$F(\mathbf{r},\lambda)=\lambda(\mathbf{r}-\mathbf{r}_A)^T\mathbf{A}(\mathbf{r}-\mathbf{r}_A)+(1-\lambda)(\mathbf{r}-\mathbf{r}_B)^T\mathbf{B}(\mathbf{r}-\mathbf{r}_B) \quad (19)$$

For all λ that satisfies $0\le\lambda\le1$, $F(\mathbf{r},\lambda)\ge0$, which means the value of potential of any point in space against two arbitrary ellipsoids should be equal or larger than zero. Also, for any fixed λ, F has a unique minimum value has a function of r [63]. Next this unique minimum value is estimate, then the condition that F reaches this value can be found.

To find out this minimum value, let $\frac{\partial F(\mathbf{r},\lambda)}{\partial\mathbf{r}}=0$, i.e.

$$2\lambda\mathbf{A}(\mathbf{r}-\mathbf{r}_A)+2(1-\lambda)\mathbf{B}(\mathbf{r}-\mathbf{r}_B)=0 \quad (20)$$

At the point where $\frac{\partial F(\mathbf{r},\lambda)}{\partial\mathbf{r}}=0$, function F reaches its minimum value. With the transformation focusing on ellipsoid A or B, two equations can be obtained, which are of $\mathbf{A}(\mathbf{r}-\mathbf{r}_A)$ and $\mathbf{B}(\mathbf{r}-\mathbf{r}_B)$ correspondingly

$$\mathbf{A}(\mathbf{r}-\mathbf{r}_A)=(1-\lambda)\left[\lambda\mathbf{B}^{-1}+(1-\lambda)\mathbf{A}^{-1}\right]^{-1}(\mathbf{r}_B-\mathbf{r}_A) \quad (21)$$

$$\mathbf{B}(\mathbf{r}-\mathbf{r}_B)=\lambda\left[\lambda\mathbf{B}^{-1}+(1-\lambda)\mathbf{A}^{-1}\right]^{-1}(\mathbf{r}_A-\mathbf{r}_B) \quad (22)$$

It is obvious that they have common part $\left[\lambda\mathbf{B}^{-1}+(1-\lambda)\mathbf{A}^{-1}\right]^{-1}$. Similar to the form used in [62,67], this part is taken out and denoted as

$$\mathbf{G}(\lambda)=\lambda\mathbf{B}^{-1}+(1-\lambda)\mathbf{A}^{-1} \quad (23)$$

Note that I consider λ as a fixed value, then λ is treated as a known value in function $\mathbf{G}(\lambda)=\lambda\mathbf{B}^{-1}+(1-\lambda)\mathbf{A}^{-1}$, and the geometry of ellipsoid A and B are also known. The next step is to represent the vector $\mathbf{r}$ (the position of the arbitrary point) with λ, A and B. Equation $\mathbf{A}(\mathbf{r}-\mathbf{r}_A)=(1-\lambda)\left[\lambda\mathbf{B}^{-1}+(1-\lambda)\mathbf{A}^{-1}\right]^{-1}(\mathbf{r}_B-\mathbf{r}_A)$ and $\mathbf{B}(\mathbf{r}-\mathbf{r}_B)=\lambda\left[\lambda\mathbf{B}^{-1}+(1-\lambda)\mathbf{A}^{-1}\right]^{-1}(\mathbf{r}_A-\mathbf{r}_B)$ can be transformed as

$$\mathbf{A}(\mathbf{r}-\mathbf{r}_A)=(1-\lambda)\mathbf{G}^{-1}(\lambda)(\mathbf{r}_B-\mathbf{r}_A) \quad (24)$$

$$\mathbf{B}(\mathbf{r}-\mathbf{r}_B)=\lambda\mathbf{G}^{-1}(\lambda)(\mathbf{r}_A-\mathbf{r}_B) \quad (25)$$

And from $\mathbf{A}(\mathbf{r}-\mathbf{r}_A)=(1-\lambda)\mathbf{G}^{-1}(\lambda)(\mathbf{r}_B-\mathbf{r}_A)$

$$\mathbf{r}=\mathbf{r}_A+(1-\lambda)\mathbf{A}^{-1}\mathbf{G}^{-1}(\lambda)(\mathbf{r}_B-\mathbf{r}_A) \quad (26)$$

From $\mathbf{B}(\mathbf{r}-\mathbf{r}_B)=\lambda\mathbf{G}^{-1}(\lambda)(\mathbf{r}_A-\mathbf{r}_B)$

$$\mathbf{r}=\mathbf{r}_B+\lambda\mathbf{B}^{-1}\mathbf{G}^{-1}(\lambda)(\mathbf{r}_A-\mathbf{r}_B) \quad (27)$$

Expand equation $2\lambda\mathbf{A}(\mathbf{r}-\mathbf{r}_A)+2(1-\lambda)\mathbf{B}(\mathbf{r}-\mathbf{r}_B)=0$ and get

$$\mathbf{r}=\left[\lambda\mathbf{A}+(1-\lambda)\mathbf{B}\right]^{-1}\left[\lambda\mathbf{A}\cdot\mathbf{r}_A+(1-\lambda)\mathbf{B}\cdot\mathbf{r}_B\right] \quad (28)$$

Take ellipsoid A and B as fixed variables, equation $\mathbf{r}=\left[\lambda\mathbf{A}+(1-\lambda)\mathbf{B}\right]^{-1}\left[\lambda\mathbf{A}\cdot\mathbf{r}_A+(1-\lambda)\mathbf{B}\cdot\mathbf{r}_B\right]$ tells that **r** is the function of λ. Remember that from equation $F(\mathbf{r},\lambda)=\lambda(\mathbf{r}-\mathbf{r}_A)^T\mathbf{A}(\mathbf{r}-\mathbf{r}_A)+(1-\lambda)(\mathbf{r}-\mathbf{r}_B)^T\mathbf{B}(\mathbf{r}-\mathbf{r}_B)$, $F(\mathbf{r},\lambda)\geq 0$. Then the condition that $F(\mathbf{r},\lambda)$ reaches its minimum value is to be explored.

It is known that equation $2\lambda\mathbf{A}(\mathbf{r}-\mathbf{r}_A)+2(1-\lambda)\mathbf{B}(\mathbf{r}-\mathbf{r}_B)=0$ is derived with condition $\frac{\partial F(\mathbf{r},\lambda)}{\partial\mathbf{r}}=0$* MERGEFORMAT, where $F(\mathbf{r},\lambda)$ reaches its minimum value, and equation $\mathbf{r}=\left[\lambda\mathbf{A}+(1-\lambda)\mathbf{B}\right]^{-1}\left[\lambda\mathbf{A}\cdot\mathbf{r}_A+(1-\lambda)\mathbf{B}\cdot\mathbf{r}_B\right]$ is derived from equation $2\lambda\mathbf{A}(\mathbf{r}-\mathbf{r}_A)+2(1-\lambda)\mathbf{B}(\mathbf{r}-\mathbf{r}_B)=0$. Then equation $\mathbf{r}=\left[\lambda\mathbf{A}+(1-\lambda)\mathbf{B}\right]^{-1}\left[\lambda\mathbf{A}\cdot\mathbf{r}_A+(1-\lambda)\mathbf{B}\cdot\mathbf{r}_B\right]$ is the condition that makes $F(\mathbf{r},\lambda)$ reaches its minimum value. Now it is known that $F(\mathbf{r},\lambda)$ reaches its minimum value when λ and **r** satisfy equation

$\mathbf{r}=\left[\lambda\mathbf{A}+(1-\lambda)\mathbf{B}\right]^{-1}\left[\lambda\mathbf{A}\cdot\mathbf{r}_A+(1-\lambda)\mathbf{B}\cdot\mathbf{r}_B\right]$. In addition, this minimum value $F(\mathbf{r},\lambda)$ is the potential on point **r** against ellipsoid A and B.

Then it is needed to find the expression of λ that makes $F(\mathbf{r},\lambda)=\lambda(\mathbf{r}-\mathbf{r}_A)^T\mathbf{A}(\mathbf{r}-\mathbf{r}_A)+(1-\lambda)(\mathbf{r}-\mathbf{r}_B)^T\mathbf{B}(\mathbf{r}-\mathbf{r}_B)$ reaches its minimum value. Again, note that $F(\mathbf{r},\lambda)$ is function of **r** and λ. Denote the minimum value of $F(\mathbf{r},\lambda)$ as follows

$$S(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)=\min_{\mathbf{r}}F(\mathbf{r},\lambda) \quad (29)$$

As Figure 21 shows, with 2 intersectant ellipsoids centered at $\mathbf{r_A}$ and $\mathbf{r_B}$ and arbitrary point **r**, there can be many possible value of $F(\mathbf{r},\lambda)$ at point **r** with different value of λ (but λ still should be between 0.0 and 1.0). $S(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)$ means the minimum value of all possible value of $F(\mathbf{r},\lambda)$ and is considered as potential of fixed-point **r**. From equation $F(\mathbf{r},\lambda)=\lambda(\mathbf{r}-\mathbf{r}_A)^T\mathbf{A}(\mathbf{r}-\mathbf{r}_A)+(1-\lambda)(\mathbf{r}-\mathbf{r}_B)^T\mathbf{B}(\mathbf{r}-\mathbf{r}_B)$ it is known that $F(\mathbf{r},\lambda)$ is a function of geometry information of ellipsoid A and B, **r** and λ. Although from equation $\mathbf{r}=\left[\lambda\mathbf{A}+(1-\lambda)\mathbf{B}\right]^{-1}\left[\lambda\mathbf{A}\cdot\mathbf{r}_A+(1-\lambda)\mathbf{B}\cdot\mathbf{r}_B\right]$ it is known that **r** changes with λ, the relationship between **r** and λ is non-linear and it is too complex to discuss function $F(\mathbf{r},\lambda)$ with these two variables at the same time. In addition, for a fixed-point **r** in space, the value of vector **r** is fixed. Thus I consider **r** as fixed value and do partial derivation of λ against $F(\mathbf{r},\lambda)$, so that I can get to know about how $F(\mathbf{r},\lambda)$ changes with λ.

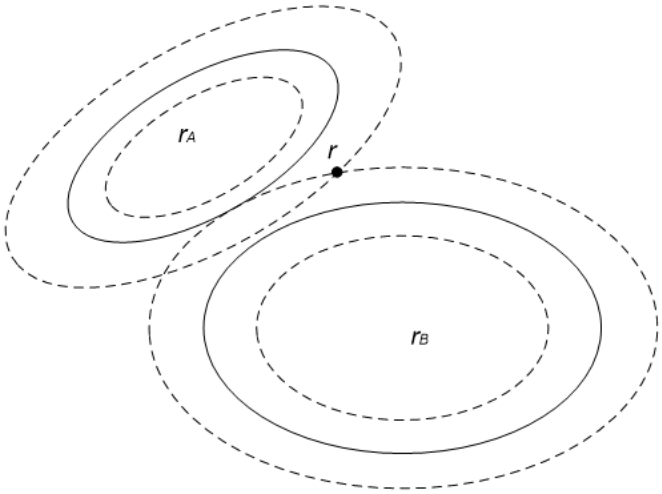


Figure 21: Two non-identical ellipsoids and one point in between that has the minimum value of F_A.

According to equation $\mathbf{A}(\mathbf{r}-\mathbf{r}_A)=(1-\lambda)\left[\lambda\mathbf{B}^{-1}+(1-\lambda)\mathbf{A}^{-1}\right]^{-1}(\mathbf{r}_B-\mathbf{r}_A)$, the potential on point $\mathbf{r}$ with relationship of ellipsoid A can be represented as

$$S_A(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)=F_A\left[\mathbf{r}(\lambda)-\mathbf{r}_A\right]=\left[\mathbf{r}(\lambda)-\mathbf{r}_A\right]^T\mathbf{A}\left[\mathbf{r}(\lambda)-\mathbf{r}_A\right] \tag{30}$$

Similarly, according to equation $\mathbf{B}(\mathbf{r}-\mathbf{r}_B)=\lambda\left[\lambda\mathbf{B}^{-1}+(1-\lambda)\mathbf{A}^{-1}\right]^{-1}(\mathbf{r}_A-\mathbf{r}_B)$, the potential on point $\mathbf{r}$ with relationship of ellipsoid B can be represented as

$$S_B(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)=F_B\left[\mathbf{r}(\lambda)-\mathbf{r}_A\right]=\left[\mathbf{r}(\lambda)-\mathbf{r}_B\right]^T\mathbf{B}\left[\mathbf{r}(\lambda)-\mathbf{r}_B\right] \tag{31}$$

From equation $S(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)=\min_r F(\mathbf{r},\lambda)$ and $F(\mathbf{r},\lambda)=\lambda(\mathbf{r}-\mathbf{r}_A)^T\mathbf{A}(\mathbf{r}-\mathbf{r}_A)+(1-\lambda)(\mathbf{r}-\mathbf{r}_B)^T\mathbf{B}(\mathbf{r}-\mathbf{r}_B)$

$$\begin{aligned}&S(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)=F\left[\mathbf{r}(\lambda),\lambda\right]\\&=\lambda F_A\left[\mathbf{r}(\lambda)-\mathbf{r}_A\right]+(1-\lambda)F_B\left[\mathbf{r}(\lambda)-\mathbf{r}_B\right]\\&=\lambda S_A(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)+(1-\lambda)S_B(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)\end{aligned} \tag{32}$$

From equation $\mathbf{A}(\mathbf{r}-\mathbf{r}_A)=(1-\lambda)\left[\lambda\mathbf{B}^{-1}+(1-\lambda)\mathbf{A}^{-1}\right]^{-1}(\mathbf{r}_B-\mathbf{r}_A)$ and $\mathbf{G}(\lambda)=\lambda\mathbf{B}^{-1}+(1-\lambda)\mathbf{A}^{-1}$

$$(\mathbf{r}-\mathbf{r}_A)^T=(1-\lambda)(\mathbf{r}_B-\mathbf{r}_A)^T\left[\mathbf{G}^{-1}(\lambda)\right]^T\left[\mathbf{A}^{-1}\right]^T \tag{33}$$

$$(\mathbf{r}-\mathbf{r}_B)^T=\lambda(\mathbf{r}_A-\mathbf{r}_B)^T\left[\mathbf{G}^{-1}(\lambda)\right]^T\left[\mathbf{B}^{-1}\right]^T \tag{34}$$

And $S(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)=\min_r F(\mathbf{r},\lambda)$ can be transformed to

$$S(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)=\lambda(1-\lambda)(\mathbf{r}_A-\mathbf{r}_B)^T\mathbf{G}^{-1}(\lambda)(\mathbf{r}_A-\mathbf{r}_B) \tag{35}$$

Equation $S(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)=\lambda(1-\lambda)(\mathbf{r}_A-\mathbf{r}_B)^T\mathbf{G}^{-1}(\lambda)(\mathbf{r}_A-\mathbf{r}_B)$ is in similar form with [62]. From conclusion in [62] it is known that $S(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)$ is nonnegative with $0\le\lambda\le1$ and zero at both end points. Now I test if $S(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)$ has single maximum value for any λ in $0\le\lambda\le1$. Note that with each fixed λ, there are many possible value of $F(\mathbf{r},\lambda)$ and $S(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)$ is the minimum value of $F(\mathbf{r},\lambda)$. The maximum value being discussed now is the highest value of $S(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)$ with all the possible value of λ that satisfies $0\le\lambda\le1$, as Figure 22 shows.

To study the feature of $F(\mathbf{r},\lambda)$ as a function, I start with feature of $\frac{d[\mathbf{r}(\lambda),\lambda]}{d\lambda}$. From equation $\mathbf{r}=\left[\lambda\mathbf{A}+(1-\lambda)\mathbf{B}\right]^{-1}\left[\lambda\mathbf{A}\cdot\mathbf{r}_A+(1-\lambda)\mathbf{B}\cdot\mathbf{r}_B\right]$ it is known that $\mathbf{r}$ is the function of λ,

and $F(\mathbf{r},\lambda)=\lambda(\mathbf{r}-\mathbf{r}_A)^R\mathbf{A}(\mathbf{r}-\mathbf{r}_A)+(1-\lambda)(\mathbf{r}-\mathbf{r}_B)^R\mathbf{B}(\mathbf{r}-\mathbf{r}_B)$. So, equation $\frac{d[\mathbf{r}(\lambda),\lambda]}{d\lambda}$ turns to

$$\frac{dF\left[\mathbf{r}(\lambda),\lambda\right]}{d\lambda}=\frac{\partial F(\mathbf{r},\lambda)}{\partial\lambda}+\mathbf{r}'(\lambda)\cdot\frac{\partial F(\mathbf{r},\lambda)}{\partial\mathbf{r}}$$
$$=\left[(\mathbf{r}-\mathbf{r}_A)^T\,\mathbf{A}(\mathbf{r}-\mathbf{r}_A)-(\mathbf{r}-\mathbf{r}_B)^T\,\mathbf{B}(\mathbf{r}-\mathbf{r}_B)\right]$$
$$+\left[\lambda(\mathbf{r}-\mathbf{r}_A)^T\cdot\mathbf{A}+(1-\lambda)(\mathbf{r}-\mathbf{r}_B)^T\cdot\mathbf{B}\right]\cdot\mathbf{r}'(\lambda) \qquad (36)$$

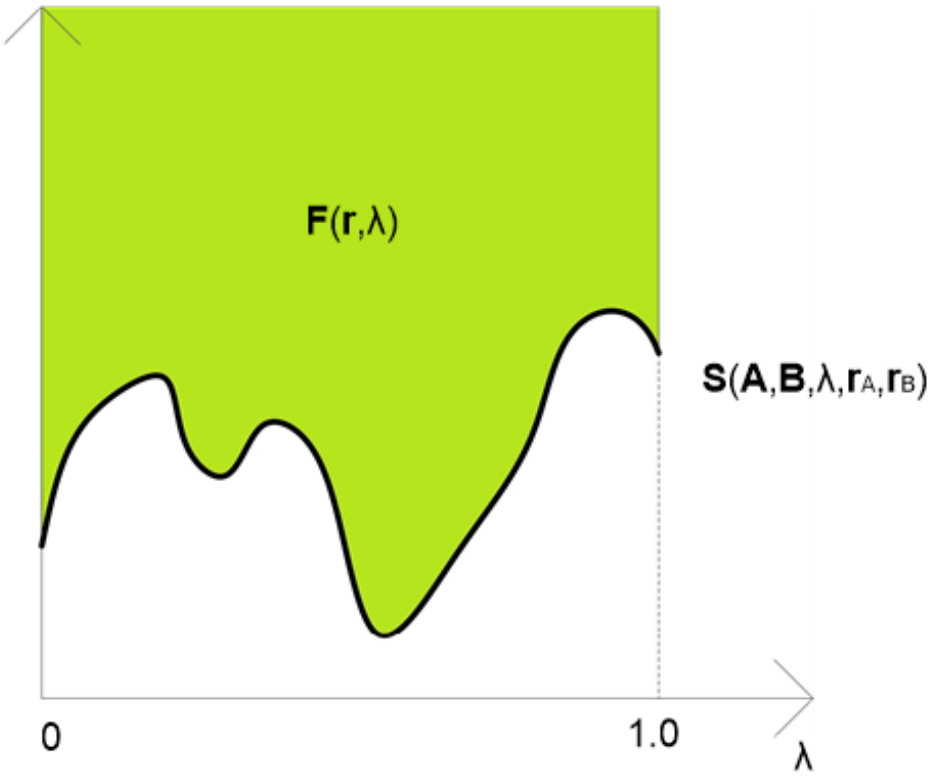


Figure 22: $S(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)$; among all the possible value of $F(\mathbf{r},\lambda)$, $S(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)=\min_r F(\mathbf{r},\lambda)$ is the minimum possible value of $F(\mathbf{r},\lambda)$ on $0\le\lambda\le1$. Value of $F(\mathbf{r},\lambda)$ and

Using $2\lambda\mathbf{A}(\mathbf{r}-\mathbf{r}_A)+2(1-\lambda)\mathbf{B}(\mathbf{r}-\mathbf{r}_B)=0$

$$\frac{dF\left[\mathbf{r}(\lambda),\lambda\right]}{d\lambda}=\left[\mathbf{r}(\lambda)-\mathbf{r}_A\right]^T\mathbf{A}\left[\mathbf{r}(\lambda)-\mathbf{r}_A\right]-\left[\mathbf{r}(\lambda)-\mathbf{r}_B\right]^T\mathbf{B}\left[\mathbf{r}(\lambda)-\mathbf{r}_B\right] \qquad (37)$$

From equation $\frac{dF\left[\mathbf{r}(\lambda),\lambda\right]}{d\lambda}=\left[\mathbf{r}(\lambda)-\mathbf{r}_A\right]^T\mathbf{A}\left[\mathbf{r}(\lambda)-\mathbf{r}_A\right]-\left[\mathbf{r}(\lambda)-\mathbf{r}_B\right]^T\mathbf{B}\left[\mathbf{r}(\lambda)-\mathbf{r}_B\right]$, it is known that $\frac{d[\mathbf{r}(\lambda),\lambda]}{d\lambda}$ can be larger or smaller than zero, which means $F(\mathbf{r},\lambda)$ increases with some value of λ and decreases with some value of λ. This does not provide much useful information about existence of maximum value of $F(\mathbf{r},\lambda)$. Thus, I derive the second derivative of $F(\mathbf{r},\lambda)$ to study $\frac{d[\mathbf{r}(\lambda),\lambda]}{d\lambda}$, i.e.

$$\frac{d^2F\left[\mathbf{r}(\lambda),\lambda\right]}{d\lambda^2}=2\left\{\left[\mathbf{r}(\lambda)-\mathbf{r}_A\right]^T\cdot\mathbf{A}-\left[\mathbf{r}(\lambda)-\mathbf{r}_B\right]^T\cdot\mathbf{B}\right\}\cdot\mathbf{r}'(\lambda) \tag{38}$$

After substituting equation (18), (30), and using (20), it can be transformed to

$$\frac{d^2F\left[\mathbf{r}(\lambda),\lambda\right]}{d\lambda^2}=2\left(\mathbf{r}_B-\mathbf{r}_A\right)^T\cdot\mathbf{G}^{-1}(\lambda)\cdot\mathbf{r}'(\lambda) \tag{39}$$

The term $\mathbf{r}'(\lambda)$ requires further expansion. At the position that $F(\mathbf{r},\lambda)$ reaches its minimum value, $\mathbf{r}(\lambda)$ satisfies $\frac{\partial F(\mathbf{r},\lambda)}{\partial\mathbf{r}}=0$. Then

$$\lambda\mathbf{A}\left(\mathbf{r}-\mathbf{r}_A\right)+(1-\lambda)\mathbf{B}\left(\mathbf{r}-\mathbf{r}_B\right)=0 \tag{40}$$

Integral equation $\lambda\mathbf{A}\left(\mathbf{r}-\mathbf{r}_A\right)+(1-\lambda)\mathbf{B}\left(\mathbf{r}-\mathbf{r}_B\right)=0$ against λ

$$\frac{\partial\left(\lambda\mathbf{A}\left(\mathbf{r}-\mathbf{r}_A\right)+(1-\lambda)\mathbf{B}\left(\mathbf{r}-\mathbf{r}_B\right)\right)}{\partial\lambda}=0$$

$$\left[\lambda\mathbf{A}+(1-\lambda)\mathbf{B}\right]\mathbf{r}'(\lambda)=-\left[\mathbf{A}\left(\mathbf{r}-\mathbf{r}_A\right)-\mathbf{B}\left(\mathbf{r}-\mathbf{r}_B\right)\right] \tag{41}$$

Or

$$\mathbf{r}'(\lambda)=-\left[\lambda\mathbf{A}+(1-\lambda)\mathbf{B}\right]^{-1}\mathbf{G}^{-1}(\lambda)\left(\mathbf{r}_B-\mathbf{r}_A\right) \tag{42}$$

Substitute equation $\mathbf{r}'(\lambda)=-\left[\lambda\mathbf{A}+(1-\lambda)\mathbf{B}\right]^{-1}\mathbf{G}^{-1}(\lambda)\left(\mathbf{r}_B-\mathbf{r}_A\right)$ back to equation $\frac{d^2F\left[\mathbf{r}(\lambda),\lambda\right]}{d\lambda^2}=2\left(\mathbf{r}_B-\mathbf{r}_A\right)^T\cdot\mathbf{G}^{-1}(\lambda)\cdot\mathbf{r}'(\lambda)$, at position where λ satisfies $\frac{\partial F(\mathbf{r},\lambda)}{\partial\mathbf{r}}=0$,

$$\frac{d^2F\left[\mathbf{r}(\lambda),\lambda\right]}{d\lambda^2}=-2\left(\mathbf{r}_B-\mathbf{r}_A\right)^T\left\{\mathbf{G}(\lambda)\left[\lambda\mathbf{A}+(1-\lambda)\mathbf{B}\right]\mathbf{G}(\lambda)\right\}^{-1}\left(\mathbf{r}_B-\mathbf{r}_A\right) \tag{43}$$

i.e.

$$\frac{d^2S\left(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B\right)}{d\lambda^2}=-2\left(\mathbf{r}_B-\mathbf{r}_A\right)^T\left\{\mathbf{G}(\lambda)\left[\lambda\mathbf{A}+(1-\lambda)\mathbf{B}\right]\mathbf{G}(\lambda)\right\}^{-1}\left(\mathbf{r}_B-\mathbf{r}_A\right) \tag{44}$$

Note that determinant calculation of both matrices **A** and **B** are larger than 0 (see equation $\mathbf{A}=a_1^{-2}\mathbf{u}_1\otimes\mathbf{u}_1+a_2^{-2}\mathbf{u}_2\otimes\mathbf{u}_2+a_3^{-2}\mathbf{u}_3\otimes\mathbf{u}_3$ and $\mathbf{B}=b_1^{-2}\mathbf{v}_1\otimes\mathbf{v}_1+b_2^{-2}\mathbf{v}_2\otimes\mathbf{v}_2+b_3^{-2}\mathbf{v}_3\otimes\mathbf{v}_3$) and $0\le\lambda\le1$, thus $1-\lambda\ge0$. Therefore $\lambda\mathbf{A}\ge0$ and $(1-\lambda)\mathbf{B}\ge0$.

Because $\lambda\mathbf{A}$ and $(1-\lambda)\mathbf{B}$ cannot be zero at the same time, finally $\left[\lambda\mathbf{A}+(1-\lambda)\mathbf{B}\right]>0$, and $\mathbf{G}(\lambda)=\lambda\mathbf{B}^{-1}+(1-\lambda)\mathbf{A}^{-1}>0$, then

$$\left\{\mathbf{G}(\lambda)\left[\lambda\mathbf{A}+(1-\lambda)\mathbf{B}\right]\mathbf{G}(\lambda)\right\}^{-1}>0$$

.

Thus $\left(\mathbf{r}_B-\mathbf{r}_A\right)^T\left\{\mathbf{G}(\lambda)\left[\lambda\mathbf{A}+(1-\lambda)\mathbf{B}\right]\mathbf{G}(\lambda)\right\}^{-1}\left(\mathbf{r}_B-\mathbf{r}_A\right)>0$, and finally $\frac{d^2S\left(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B\right)}{d\lambda^2}<0$, which means the value of $F(\mathbf{r},\lambda)$

increases more and more slowly as λ increases, $F(\mathbf{r},\lambda)$ is a concave function. It implies the uniqueness of the maximum value of $F(\mathbf{r},\lambda)$ in the interval of $\lambda \in [0,1]$, $S(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)$ is a concave function of λ, as Figure 23 shows.

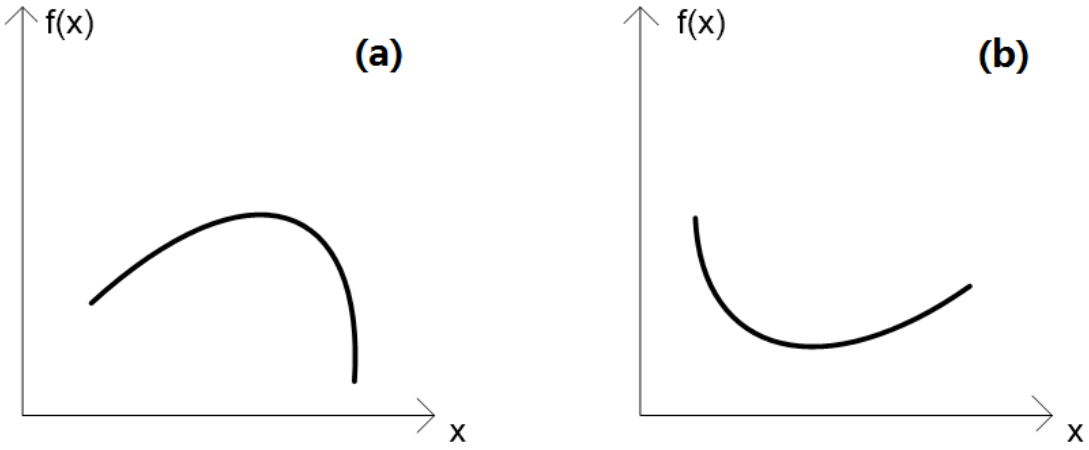


Figure 23: (a) concave function with a unique maximum value; (b) convex function with a unique minimum value.

The value of λ at this maximum value point is the value used to calculate pair potential of ellipsoid A and B. Denote this value as λ_0, then at this point λ_0 satisfies

$$\frac{dS(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)}{d\lambda}=0 \tag{45}$$

As discussed above, λ_0 is the value I look for.

According to equation $S(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)=\lambda(1-\lambda)(\mathbf{r}_A-\mathbf{r}_B)^T\mathbf{G}^{-1}(\lambda)(\mathbf{r}_A-\mathbf{r}_B)$, denote

$$\mathbf{s}=(\mathbf{r}_A-\mathbf{r}_B) \tag{46}$$

Then

$$F(\lambda)=\lambda(1-\lambda)\mathbf{s}^T\mathbf{G}^{-1}(\lambda)\mathbf{s} \tag{47}$$

In summary, similar to [62], the New Perram & Wertheim ellipsoid contact potential may be expressed as

$$\Phi(\mathbf{A},\mathbf{B},\mathbf{s})=F(\lambda_0)=\lambda_0(1-\lambda_0)\mathbf{s}^T\mathbf{G}^{-1}(\lambda_0)\mathbf{s} \tag{48}$$

In which $\Phi(\mathbf{A},\mathbf{B},\mathbf{s})$ is potential between ellipsoid A and B, and $\Phi(\mathbf{A},\mathbf{B},\mathbf{s})$ means it is a function of geometry information of A and B as well as the vector of centre of both ellipsoids.

For calculation of $\mathbf{G}(\lambda)$, first the value of λ needs to be calculated. Because

$$\mathbf{G}(\lambda)=\lambda\mathbf{B}^{-1}+(1-\lambda)\mathbf{A}^{-1},$$

Define

$$\mathbf{M} = \mathbf{B}^{1/2}\mathbf{A}^{-1}\mathbf{B}^{1/2} \tag{49}$$

$$\mathbf{t} = \mathbf{B}^{1/2}\mathbf{s} \tag{50}$$

Assume

$$\mathbf{M} = \mathbf{W}\begin{pmatrix} m_1^2 & 0 & 0 \\ 0 & m_2^2 & 0 \\ 0 & 0 & m_3^2 \end{pmatrix}\mathbf{W}^T = \begin{pmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{pmatrix}\begin{pmatrix} m_1^2 & 0 & 0 \\ 0 & m_2^2 & 0 \\ 0 & 0 & m_3^2 \end{pmatrix}\begin{pmatrix} w_{11} & w_{21} & w_{31} \\ w_{12} & w_{22} & w_{32} \\ w_{13} & w_{23} & w_{33} \end{pmatrix} \tag{51}$$

Define

$$\mathbf{q} = \mathbf{W}^T\mathbf{t} \tag{52}$$

$$g_1(x) = \frac{q_1^2\left(m_1^2x^2-1\right)}{\left(m_1^2x+1\right)^2} + \frac{q_2^2\left(m_2^2x^2-1\right)}{\left(m_2^2x+1\right)^2} + \frac{q_3^2\left(m_3^2x^2-1\right)}{\left(m_3^2x+1\right)^2} = 0 \tag{53}$$

$$\lambda = \frac{1}{1+x} \tag{54}$$

Note that

$$g_1'(x) = \frac{dg_1(x)}{dx} = 2(1+x)\left[\frac{q_1^2m_1^2}{\left(m_1^2x+1\right)^3} + \frac{q_2^2m_2^2}{\left(m_2^2x+1\right)^3} + \frac{q_3^2m_3^2}{\left(m_3^2x+1\right)^3}\right] \tag{55}$$

$$g_1'(x) > 0 \tag{56}$$

$$g_1(0) = -\left(q_1^2+q_2^2+q_3^2\right) < 0 \tag{57}$$

$$g_1(+\infty) = \left(\frac{q_1}{m_1}\right)^2 + \left(\frac{q_2}{m_2}\right)^2 + \left(\frac{q_3}{m_3}\right)^2 > 0 \tag{58}$$

Newton-Raphson method [90] could be used to find root of Equation $g_1(x) = \frac{q_1^2\left(m_1^2x^2-1\right)}{\left(m_1^2x+1\right)^2} + \frac{q_2^2\left(m_2^2x^2-1\right)}{\left(m_2^2x+1\right)^2} + \frac{q_3^2\left(m_3^2x^2-1\right)}{\left(m_3^2x+1\right)^2} = 0$

as a function,

then from value of x I can get value of λ using equation (50).The next step is calculation of $\mathbf{G}^{-1}(\lambda)$. Let

$$\mathbf{H}(\lambda) = \left[\lambda\mathbf{I} + (1-\lambda)\mathbf{B}^{1/2}\mathbf{A}^{-1}\mathbf{B}^{1/2}\right] = \left[\lambda\mathbf{I} + (1-\lambda)\mathbf{M}\right] \tag{59}$$

Then equation $\mathbf{G}(\lambda) = \lambda\mathbf{B}^{-1} + (1-\lambda)\mathbf{A}^{-1}$ turns to

$$\mathbf{G}(\lambda)=\mathbf{B}^{-1/2}\mathbf{H}(\lambda)\mathbf{B}^{-1/2} \tag{60}$$

Thus

$$\mathbf{G}^{-1}(\lambda)=\mathbf{B}^{1/2}\mathbf{H}^{-1}(\lambda)\mathbf{B}^{1/2} \tag{61}$$

Question reduces to how to calculate $\mathbf{H}^{-1}(\lambda)=\left[\lambda\mathbf{I}+(1-\lambda)\mathbf{M}\right]^{-1}$. . Denote

$$\mathbf{M}=\mathbf{W}\begin{pmatrix} m_1^2 & 0 & 0 \\ 0 & m_2^2 & 0 \\ 0 & 0 & m_3^2 \end{pmatrix}\mathbf{W}^T=\mathbf{W}\bar{\mathbf{M}}\mathbf{W}^T \tag{62}$$

Then

$$\begin{aligned}\mathbf{H}^{-1}(\lambda)&=\left[\lambda\mathbf{I}+(1-\lambda)\mathbf{W}\bar{\mathbf{M}}\mathbf{W}^T\right]^{-1}\\&=\left[\mathbf{W}\left(\lambda\mathbf{I}+(1-\lambda)\bar{\mathbf{M}}\right)\mathbf{W}^T\right]^{-1}\\&=\mathbf{W}\left[\lambda\mathbf{I}+(1-\lambda)\bar{\mathbf{M}}\right]^{-1}\mathbf{W}^T\end{aligned} \tag{63}$$

$$\mathbf{H}^{-1}(\lambda)=\mathbf{W}\begin{pmatrix} \lambda+(1-\lambda)m_1^2 & 0 & 0 \\ 0 & \lambda+(1-\lambda)m_2^2 & 0 \\ 0 & 0 & \lambda+(1-\lambda)m_3^2 \end{pmatrix}^{-1}\mathbf{W}^T \tag{64}$$

$$\mathbf{H}^{-1}(\lambda)=\mathbf{W}\begin{pmatrix} \dfrac{1}{\lambda+(1-\lambda)m_1^2} & 0 & 0 \\ 0 & \dfrac{1}{\lambda+(1-\lambda)m_2^2} & 0 \\ 0 & 0 & \dfrac{1}{\lambda+(1-\lambda)m_3^2} \end{pmatrix}\mathbf{W}^T \tag{65}$$

Finally

$$\mathbf{G}^{-1}(\lambda)=\mathbf{B}^{1/2}\mathbf{W}\begin{pmatrix} \dfrac{1}{\lambda+(1-\lambda)m_1^2} & 0 & 0 \\ 0 & \dfrac{1}{\lambda+(1-\lambda)m_2^2} & 0 \\ 0 & 0 & \dfrac{1}{\lambda+(1-\lambda)m_3^2} \end{pmatrix}\left[\mathbf{B}^{1/2}\mathbf{W}\right]^T \tag{66}$$

In this way the value of $\mathbf{G}^{-1}(\lambda)$ can be estimated with value of λ. This process is implemented by programming code and carried out in simulation.

According to equation $F_A(\mathbf{r}-\mathbf{r}_A)\begin{cases}<1 \ for \ \mathbf{r} \ \text{inside} \ A \\ =1 \ for \ \mathbf{r} \ \text{on the surface of} \ A \\ >1 \ for \ \mathbf{r} \ \text{outside} \ A\end{cases}$:

$$\Phi(\mathbf{A},\mathbf{B},\mathbf{s})\begin{cases}<1 \ for \ A \ and \ B \ overlapping \\ =1 \ for \ A \ and \ B \ externally \ tangent \\ >1 \ for \ A \ and \ B \ non-overlapping\end{cases} \quad (67)$$

According to the method to calculate $\mathbf{G}(\lambda)$, it is complex to draw graph for contact potential as a function. As a simplified condition, for two identical ellipsoidal agents with three semi-axes equals 5, 2 and 2 and in the same direction, the value of potential versus distance is shown in Figure 24.

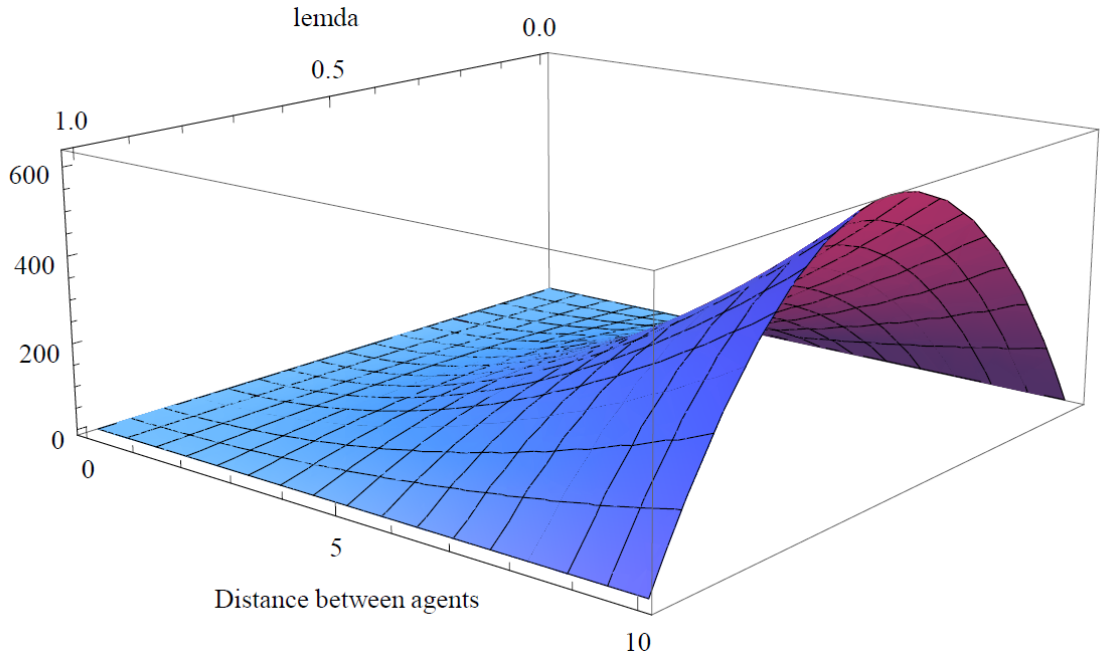


Figure 24: Potential vs. distance.

Adhesion force

"The dynamic balance between cell adhesion and cell movement is arguably the most important universal feature of adhesion in development". In addition, the type and strength of different adhesion forces plays a significant role in cell sorting [130]. The adhesion force likewise plays an important role in agent motility in my model. The velocity of agent movement is controlled by the combination of adhesion force and resistance force, while the direction of agent movement is controlled by adhesion force alone. When one agent comes into contact with other agents, the adhesion force balances with the contact force. The point at which the adhesion force and contact force balance represents the elasticity of the agent. The necessity of adhesion force in such modelled systems is discussed in section 2.1.6.

The adhesion force depends on potential. First, I discuss how to generate force from potential in general. In the model, the potential is a scalar quantity that has value but no direction: potential represents some form of energy. According to classical mechanics, force does work to move an object from one point to another point in space. In general, a force is represented by a vector in space, in the form of $\mathbf{F}(x \quad y \quad z)$. The size of force is $|\mathbf{F}|$ and the direction of force is $\frac{\mathbf{F}}{|\mathbf{F}|}$. Let the start point be $\mathbf{P}_a$ and end point be $\mathbf{P}_b$, and both $\mathbf{P}_a$ and $\mathbf{P}_b$ are vectors in space. The energy *E* equals the work done by force: $E = \int_{\mathbf{P}_a}^{\mathbf{P}_b} \mathbf{F}$. Like potential, the energy has no direction, and is a scalar quantity. Note that the value of *E* can be positive or negative, meaning the force **F** does positive work or negative work respectively. If the energy *E* and start and end point $\mathbf{P}_a$ and $\mathbf{P}_b$ are known, to calculate **F** I need to calculate the partial differential of *E* by $\mathbf{P}_a - \mathbf{P}_b$, i.e. $\mathbf{F} = \frac{\partial E}{\partial(\mathbf{P}_a - \mathbf{P}_b)}$.

Here, as I assume the mass of agents are evenly distributed, the force can be considered to work on the centre point of the ellipsoid. The position of the ellipsoid is represented by the position of its centre. Specifically, in my model, for two ellipsoids, the energy *E* equals the work done by force **F** to move an ellipsoid from its position to the position of the other ellipsoid. Similar to the previous deduction, in my model

$\mathbf{F} = \frac{\partial E}{\partial(\mathbf{A} - \mathbf{B})}$, where **A** and **B** are position of centre of the two ellipsoids.

According to 2.4.4, the Metropolis algorithms [59] is a group of methods in which the spatial relationship between cells is considered to have a particular energy, and cells are considered to move to minimize the energy of the system [59]. Here the Metropolis algorithm is used to generate energy from potential. After that the energy is partially derivative with respect to displacement to determine force. Torque is also generated from energy and the translation from energy to torque is discussed in Section 3.6. In order to simplify the problem, I begin by treating adhesion force for two identical spheres, and then generalize the conclusion to UpTo matriz, and should be in look two arbitrary spheres, and further that generalization to two arbitrary ellipsoids.

In the case of two identical spherical particles **A** and **B** with radius of R, because the sphere has no direction, according to equation

$$\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3) = \begin{pmatrix} u_1 & u_2 & u_3 \\ u_1 & u_2 & u_3 \\ u_3 & u_3 & u_3 \end{pmatrix}$$, its direction matrix is the unit matrix, and

$$\mathbf{A} = \mathbf{U}\bar{\mathbf{A}}\mathbf{U}^T = \begin{pmatrix} u_{11} & u_{12} & u_{13} \\ u_{21} & u_{22} & u_{23} \\ u_{31} & u_{32} & u_{33} \end{pmatrix} \begin{pmatrix} \frac{1}{a_1^2} & 0 & 0 \\ 0 & \frac{1}{a_2^2} & 0 \\ 0 & 0 & \frac{1}{a_3^2} \end{pmatrix} \begin{pmatrix} u_{11} & u_{21} & u_{31} \\ u_{12} & u_{22} & u_{32} \\ u_{13} & u_{23} & u_{33} \end{pmatrix}$$ becomes

$$A = B = \begin{pmatrix} \frac{1}{R^2} & 0 & 0 \\ 0 & \frac{1}{R^2} & 0 \\ 0 & 0 & \frac{1}{R^2} \end{pmatrix} \quad (68)$$

According to equation $\Phi(\mathbf{A},\mathbf{B},\mathbf{s}) = F(\lambda_0) = \lambda_0(1-\lambda_0)\mathbf{s}^T\mathbf{G}^{-1}(\lambda_0)\mathbf{s}$ the potential is therefore

$$\Phi(\mathbf{A},\mathbf{B},\mathbf{s}) = f(\lambda_0) = \lambda_0(1-\lambda_0)\mathbf{s}^T\mathbf{G}^{-1}(\lambda_0)\mathbf{s}$$
$$= \frac{(x_A - x_B)^2 + (y_A - y_B)^2 + (z_A - z_B)^2}{(2R)^2} \quad (69)$$

In which $\mathbf{r_A} = (x_\mathbf{A}, y_\mathbf{A}, z_\mathbf{A})$ and $\mathbf{r_B} = (x_\mathbf{B}, y_\mathbf{B}, z_\mathbf{B})$ are centre of the two spherical particles **A** and **B**.

In [56], the adhesion energy is modelled in the form of

$$W = \varepsilon A \quad (70)$$

In which ε is a proportional constant, *A* is the contact area of the two spheres, which is proportional to the distance between two spheres. In my model potential is proportional to square of distance between two ellipsoids, thus, to generalize the equation for ellipsoids, I use square root of potential to replace the contact area term *A* in equation $W = \varepsilon A$ and get

$$W^{ad} = \varepsilon\sqrt{\Phi(\mathbf{A},\mathbf{B},\mathbf{s})} \quad (71)$$

Note that the potential Φ is written in the form of $\Phi(\mathbf{A},\mathbf{B},\mathbf{s})$ to represent that the potential is a function of the geometry of A and B, and the distance vector between them. Since a sphere can be considered as a special type of ellipsoid with three semi-axes of same length, the potential formula can be shaped to work for spheres. Substitute equation

$$\Phi(\mathbf{A},\mathbf{B},\mathbf{s}) = f(\lambda_0) = \lambda_0(1-\lambda_0)\mathbf{s}^T\mathbf{G}^{-1}(\lambda_0)\mathbf{s}$$
$$= \frac{(x_A - x_B)^2 + (y_A - y_B)^2 + (z_A - z_B)^2}{(2R)^2}$$ to equation $W^{ad} = \varepsilon\sqrt{\Phi(\mathbf{A},\mathbf{B},\mathbf{s})}$ then

$$W^{Ad} = \varepsilon\sqrt{\frac{(x_A - x_B)^2 + (y_A - y_B)^2 + (z_A - z_B)^2}{(2R)^2}} \quad (72)$$

Then as discussed in Section 2.4.4, the value of force equals the partial differential of energy against the displacement. Thus, the adhesion force is

$$F_A^{Ad} = -\frac{\partial W^{Ad}}{\partial r_A} = -\frac{\varepsilon}{2R}\frac{\partial\sqrt{(x_A - x_B)^2 + (y_A - y_B)^2 + (z_A - z_B)^2}}{\partial r_A}$$
$$= -\frac{\varepsilon}{4R}\frac{(x_A - x_B,\quad y_A - y_B,\quad z_A - z_B)}{\sqrt{(x_A - x_B)^2 + (y_A - y_B)^2 + (z_A - z_B)^2}} \quad (73)$$

This gives

$$\left|\mathbf{F}_A^{Ad}\right| = \frac{\varepsilon}{4R} = const \quad (74)$$

This means that the size of the adhesion force for two identical spheres is a constant value. This may be counterintuitive, as one might think that the closer two particles are, the larger the adhesion force should be. However, this assumption may be driven by the phenomenon that the movement speed of an object increases over time, which only indicates that direction of the force does not change. A constant force can, of course, explain the acceleration of an object.

This is the form of adhesion energy between two identical spheres as a special case of ellipsoids in my model. The next step is to generalize the conclusion for two arbitrary spheres. In the case of two non-identical spherical particles, let R_A be radius of sphere A and R_B be radius of sphere B, similar to equation

$$A = B = \begin{pmatrix} \frac{1}{R^2} & 0 & 0 \\ 0 & \frac{1}{R^2} & 0 \\ 0 & 0 & \frac{1}{R^2} \end{pmatrix}$$

$$\mathbf{A} = \begin{pmatrix} \frac{1}{R_A^2} & 0 & 0 \\ 0 & R_A^2 & 0 \\ 0 & 0 & R_A^2 \end{pmatrix} \tag{75}$$

$$\mathbf{B} = \begin{pmatrix} \frac{1}{R_B^2} & 0 & 0 \\ 0 & R_B^2 & 0 \\ 0 & 0 & R_B^2 \end{pmatrix} \tag{76}$$

Substitute $\mathbf{A} = \begin{pmatrix} \frac{1}{R_A^2} & 0 & 0 \\ 0 & R_A^2 & 0 \\ 0 & 0 & R_A^2 \end{pmatrix}$ and $\mathbf{B} = \begin{pmatrix} \frac{1}{R_B^2} & 0 & 0 \\ 0 & R_B^2 & 0 \\ 0 & 0 & R_B^2 \end{pmatrix}$ to equation (46),

$$\Phi(\mathbf{A},\mathbf{B},\mathbf{s}) = \frac{1}{(R_A + R_B)^2}\left[(x_A - x_B)^2 + (y_A - y_B)^2 + (z_A - z_B)^2\right] \tag{77}$$

Similar to equation $W^{Ad} = \varepsilon\sqrt{\frac{(x_A - x_B)^2 + (y_A - y_B)^2 + (z_A - z_B)^2}{(2R)^2}}$,

$$W^{Ad} = \varepsilon\sqrt{\frac{(x_A - x_B)^2 + (y_A - y_B)^2 + (z_A - z_B)^2}{(R_A + R_B)^2}} \tag{78}$$

Similar to $F_A^{Ad} = -\frac{\partial W^{Ad}}{\partial r_A} = -\frac{\varepsilon}{2R}\frac{\partial\sqrt{(x_A - x_B)^2 + (y_A - y_B)^2 + (z_A - z_B)^2}}{\partial r_A} = -\frac{\varepsilon}{4R}\frac{(x_A - x_B, y_A - y_B, z_A - z_B)}{\sqrt{(x_A - x_B)^2 + (y_A - y_B)^2 + (z_A - z_B)^2}}$, adhesion force

$$\mathbf{F}_A^{Ad} = -\frac{\partial W^{Ad}}{\partial \mathbf{r}_A} = -\frac{\varepsilon}{R_A + R_B}\frac{\partial\sqrt{(x_A - x_B)^2 + (y_A - y_B)^2 + (z_A - z_B)^2}}{\partial \mathbf{r}_A} = -\frac{\varepsilon}{2(R_A + R_B)}\frac{\{x_A - x_B, y_A - y_B, z_A - z_B\}}{\sqrt{(x_A - x_B)^2 + (y_A - y_B)^2 + (z_A - z_B)^2}} \tag{79}$$

This gives

$$\left|\mathbf{F}_A^{Ad}\right| = \frac{\varepsilon}{2(R_A + R_B)} = const \tag{80}$$

Which is in similar form of equation $\left|\mathbf{F}_A^{Ad}\right| = \frac{\varepsilon}{4R} = const$. Again, the adhesion force is constant. As a generalization I assume for pair of arbitrary ellipsoidal particles:

$$W^{Ad} = \varepsilon\sqrt{\Phi(\mathbf{A}, \mathbf{B}, \mathbf{s})} \tag{81}$$

Then the adhesion force should be in form of

$$\mathbf{F}^{Ad} = \frac{\partial W^{Ad}}{\partial \mathbf{s}} = \frac{\partial W^{Ad}}{\partial \Phi}\cdot\frac{\partial \Phi}{\partial \mathbf{s}} = -2\lambda_0(1 - \lambda_0)\frac{\partial W^{Ad}}{\partial \Phi}\mathbf{G}^{-1}(\lambda_0)\mathbf{s} \tag{82}$$

From $W^{Ad} = \varepsilon\sqrt{\Phi(\mathbf{A}, \mathbf{B}, \mathbf{s})}$

$$\frac{\partial W^{Ad}}{\partial \Phi} = \frac{\partial\left(\varepsilon\sqrt{\Phi}\right)}{\partial \Phi} = \frac{1}{2}\varepsilon\Phi^{-\frac{1}{2}} \tag{83}$$

Thus, the adhesion force

$$\mathbf{F}^{Ad} = -\varepsilon\lambda_0\left(1-\lambda_0\right)\Phi^{-\frac{1}{2}}\mathbf{G}^{-1}\left(\lambda_0\right)\mathbf{s} \tag{84}$$

From $\mathbf{F}^{Ad} = -\varepsilon\lambda_0\left(1-\lambda_0\right)\Phi^{-\frac{1}{2}}\mathbf{G}^{-1}\left(\lambda_0\right)\mathbf{s}$, the adhesion force of two arbitrary ellipsoids is not constant, and its direction is always from centre of the reference ellipsoid to the centre of the other ellipsoid.

Contact force

As the adhesion force tends to make cells stick together: if there is no other force to balance it, all cells will end up completely overlapping. Contact force is the main interaction factor that prevents ellipsoid cells from overlapping. Similar to adhesion force, contact force is also calculated from energy, which is calculated from potential by using the Metropolis algorithm (Figure 25).

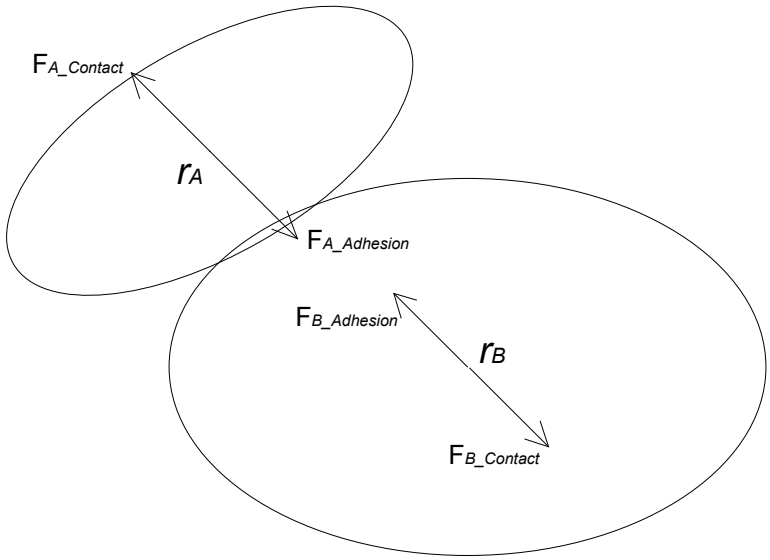


Figure 25: Contact force and adhesion force on two non-identical ellipsoids.

The Lennard-Jones Potential is the most commonly used form for two-object interaction, which can be written as the following form:

$$V\left(r\right) = 4\varepsilon\left[\left(\frac{\sigma^2}{r^2}\right)^6 - \left(\frac{\sigma^2}{r^2}\right)^3\right] \tag{85}$$

The term $\frac{r^2}{\sigma^2}$ represents the distance between a pair of spheres. Since in my model the shape of the cell is ellipsoid, I cannot use $\frac{r^2}{\sigma^2}$ directly. To represent the spatial relationship of a pair of ellipsoids, similar to [62], I may use potential to replace $\frac{r^2}{\sigma^2}$. I then get contact energy

$$U = 4\varepsilon\left[\Phi^{-6}\left(A,B,s\right) - \Phi^{-3}\left(A,B,s\right)\right] \tag{86}$$

$$U = 4\varepsilon\left[\frac{1}{\Phi^6\left(\mathbf{A},\mathbf{B},\mathbf{r}_A,\mathbf{r}_B\right)} - \frac{1}{\Phi^3\left(\mathbf{A},\mathbf{B},\mathbf{r}_A,\mathbf{r}_B\right)}\right] \tag{87}$$

Where U is the contact energy, $\Phi\left(\mathbf{A},\mathbf{B},\mathbf{r}_A,\mathbf{r}_B\right)$ is the potential, $\mathbf{r}_A$ and $\mathbf{r}_B$ are vectors that represent position of ellipsoid A and B respectively. According to Section 3.3, contact force is $\mathbf{F} = \frac{\partial U}{\partial\left(\mathbf{r}_A - \mathbf{r}_B\right)}$. Thus, the ellipsoid contact force can be written in the following form.

$$F\left(r\right) = \frac{\partial U}{\partial\left(\mathbf{r}_A - \mathbf{r}_B\right)} = \frac{\partial\left\{4\varepsilon\left[\frac{1}{\Phi^6\left(\mathbf{A},\mathbf{B},\mathbf{r}_A,\mathbf{r}_B\right)} - \frac{1}{\Phi^3\left(\mathbf{A},\mathbf{B},\mathbf{r}_A,\mathbf{r}_B\right)}\right]\right\}}{\partial\left(\mathbf{r}_A - \mathbf{r}_B\right)} \tag{88}$$

Note that $\mathbf{s} = \left(\mathbf{r}_A - \mathbf{r}_B\right)$, then $F\left(r\right) = \frac{\partial U}{\partial\left(\mathbf{r}_A - \mathbf{r}_B\right)} = \frac{\partial\left\{4\varepsilon\left[\frac{1}{\Phi^6\left(\mathbf{A},\mathbf{B},\mathbf{r}_A,\mathbf{r}_B\right)} - \frac{1}{\Phi^3\left(\mathbf{A},\mathbf{B},\mathbf{r}_A,\mathbf{r}_B\right)}\right]\right\}}{\partial\left(\mathbf{r}_A - \mathbf{r}_B\right)}$ turns to

$$F\left(r\right) = -4\varepsilon\left[-6\Phi^{-7}\left(A,B,s\right) + 3\Phi^{-4}\left(A,B,s\right)\right]\frac{\partial\Phi\left(A,B,s\right)}{\partial s} = -4\varepsilon\left[-6\Phi^{-7}\left(A,B,s\right) + 3\Phi^{-4}\left(A,B,s\right)\right]\cdot\left[2\lambda_0\left(1-\lambda_0\right)G^{-1}\left(\lambda_0\right)s\right] \tag{89}$$

Then I run an early-stage simulation including contact and adhesion force and regularly save output data including cell position, direction, velocity and angular velocity. A third-party visualization program [50]. "X-agents 3D Visualization") reads these data and draws them on screen using an OpenGL API.

Comparing Figures 26-28, I can see the movement of agents is not realistic, in which the agents are forced apart from each other by a relatively large force. This phenomenon is caused by equation

$$F\left(r\right) = -4\varepsilon\left[-6\Phi^{-7}\left(A,B,s\right) + 3\Phi^{-4}\left(A,B,s\right)\right]\frac{\partial\Phi\left(A,B,s\right)}{\partial s} = -4\varepsilon\left[-6\Phi^{-7}\left(A,B,s\right) + 3\Phi^{-4}\left(A,B,s\right)\right]\cdot\left[2\lambda_0\left(1-\lambda_0\right)G^{-1}\left(\lambda_0\right)s\right]$$

which contains $\Phi^{-7}\left(A,B,s\right)$, a term that contains high power of displacement $\mathbf{s}$. $\Phi^{-7}\left(A,B,s\right)$ makes the model dynamics highly sensitive to the value of potential, especially in the case when the value of $\Phi\left(A,B,s\right)$ is relatively small, which happens when two cells are close to each other. As shown in Figure 29, the energy dramatically increases while potential decreases. When two agents are very close to each other, a small decrease of potential results in a rapid increase of contact force which is much larger than the adhesion force, and therefore forces the agent apart at high speed. Clearly this is improper behaviour. The Lennard-Jones potential is used in [91] only to deal with non-contact particle interaction; another formula that contains square of displacement is introduced for particle contact. I also need to switch to a formula with a lower power to avoid the problem. In [24,41,58], the Hertz model is used to describe the physical interaction among cells.

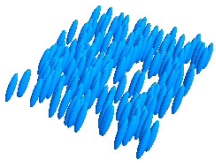


Figure 26: Starting condition of early simulation: 200 agents are randomly placed on the same substrate plane.

Figure 27: After a few loops of running of simulation, some agents have already moved relatively far away from others.

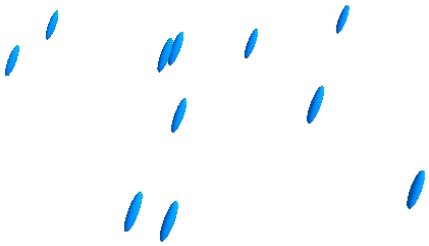


Figure 28: After 10 seconds (simulation time), most of the agents have moved beyond the observation area.

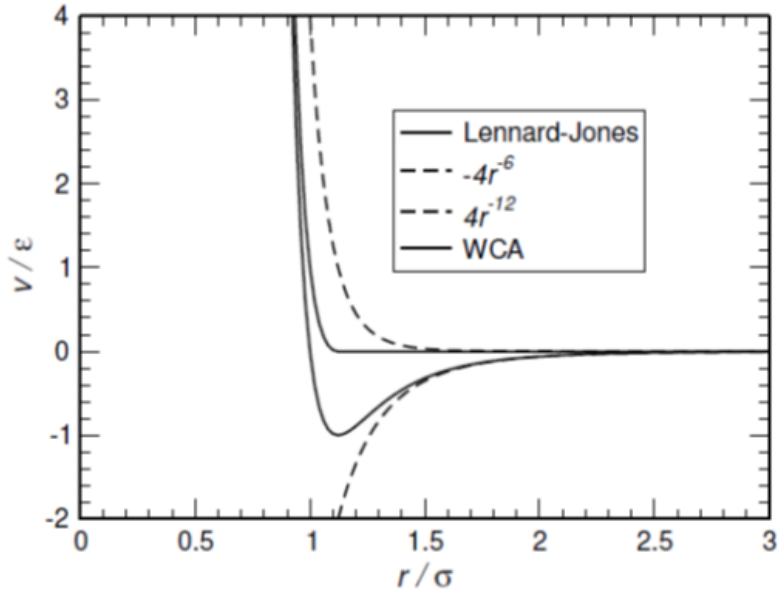


Figure 29: Energy dramatically increases while potential decreases [91].

The Hertz model was introduced by Heinrich Hertz in 1882 to solve the contact force between two elastic objects. The Hertz model is used to model the attractive and repulsive energy among elastic spheroidal cells. In the Hertz model, energy W between two spheroidal agents **A** and **B** can be represented as follows

$$W = \frac{2Ka^5}{5R_m^2} \tag{90}$$

which

$$a = \sqrt{R_m \delta} \tag{91}$$

$$\delta = R_A + R_B - \left\| \mathbf{x}_A - \mathbf{x}_B \right\| \tag{92}$$

$$K = \frac{4}{3\pi\left(k_1 + k_2\right)} \tag{93}$$

$$\begin{cases} k_1 = \dfrac{1-\nu_1^2}{\pi E_1} \\ k_2 = \dfrac{1-\nu_2^2}{\pi E_2} \end{cases} \tag{94}$$

$$R_m = \frac{R_1 R_2}{R_1 + R_2} \tag{95}$$

In which R_1 and R_2 are radii of the two spheres; ν_1 and ν_2 are Poisson's ratios of **A** and **B** respectively, and E_1 and E_2 are Young's moduli of A and B respectively; δ is level of deformation of the two spheres.

From $W = \frac{2Ka^5}{5R_m^2}$, $K = \frac{4}{3\pi\left(k_1 + k_2\right)}$, and $\begin{cases} k_1 = \frac{1-\nu_1^2}{\pi E_1} \\ k_2 = \frac{1-\nu_2^2}{\pi E_2} \end{cases}$,

it is known that the energy W is proportional to Young's moduli of both spheres A and B, and is proportional to Poisson's ratios of A and B.

Also from $W = \frac{2Ka^5}{5R_m^2}$, $a = \sqrt{R_m \delta}$, and $\delta = R_A + R_B - \left\|\mathbf{x}_A - \mathbf{x}_B\right\|$ it is known that energy W is proportional to level of deformation. A and from equation $a = \sqrt{R_m \delta}$ and $\delta = R_A + R_B - \left\|\mathbf{x}_A - \mathbf{x}_B\right\|$

, a is proportional to square root of deformation, and then from equation $W = \frac{2Ka^5}{5R_m^2}$, the energy is proportional to 2.5 power of level of deformation, comparing to the seventh power in Lennard-Jones Potential, Hertz model should have smooth curve. Thus, I decided to use the Hertz model instead of Lennard-Jones Potential.

However, the Hertz formula is not suitable for direct use with ellipsoidal agents as in my model, because an ellipsoid does not have radius R but three semi-axes which may not be equal to each other. Thus, a generalization is required. The goal of my generalization is to replace all the terms that contain R_1 and R_2 in the formula, so that when substituting the same values for the lengths of the three semi-axes of the ellipsoid into a generalised Hertz formula, the generalised Hertz formula can revert to the original Hertz formula.

As per the approach of generalizing the formula for contact potential, I begin with consideration of the Hertz formula for an ellipsoid and an arbitrary point in space, and then I apply the formula for two ellipsoids (Figure 30).

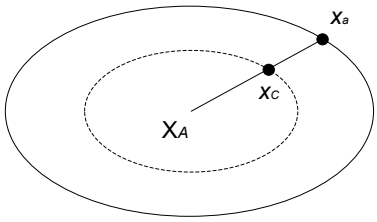


Figure 30: Ellipsoid **A** and two points $\mathbf{X}_a$ and $\mathbf{X}_c$. Point $\mathbf{X}_a$ on the surface of an ellipsoid and a point $\mathbf{X}_c$ inside an ellipsoid, and the two points are on the same line. The dotted line represents the potential surface on which point $\mathbf{X}_c$ lies.

For an ellipsoid $\mathbf{X}_A$ (here I use the position vector of the centre point of the ellipsoid to represent it) in space and one point $\mathbf{x}_c$ inside the ellipsoid. The contact force applies when two ellipsoids make contact with each other, thus I set point $\mathbf{x}_c$ inside the ellipsoid. By connecting point $\mathbf{x}_c$ and $\mathbf{X}_A$ and extending the line, I can find the point of intersection $\mathbf{x}_a$ on surface of ellipsoid. Similar to equation $F_A\left(\mathbf{r}-\mathbf{r}_A\right) = \left(\mathbf{r}-\mathbf{r}_A\right)^T \mathbf{A}\left(\mathbf{r}-\mathbf{r}_A\right)$ and $F_B\left(\mathbf{r}-\mathbf{r}_B\right) = \left(\mathbf{r}-\mathbf{r}_B\right)^T \mathbf{B}\left(\mathbf{r}-\mathbf{r}_B\right)$, the potential of point $\mathbf{x}_a$ to ellipsoid $\mathbf{X}_A$ can be written as follows

$$\Phi_a = \left(\mathbf{x}_a - \mathbf{X}_A\right)^T \mathbf{A}\left(\mathbf{x}_a - \mathbf{X}_A\right) \tag{96}$$

where Φ_a is the potential and A is the geometry matrix of ellipsoid $\mathbf{X}_A$. Similarly, for $\mathbf{x}_c$,

$$\mathbf{\Phi}_c = (\mathbf{x}_c - \mathbf{X}_A)^T \mathbf{A} (\mathbf{x}_c - \mathbf{X}_A) \tag{97}$$

Geometrically vector $(\mathbf{x}_c - \mathbf{X}_A)$ and $(\mathbf{x}_a - \mathbf{X}_A)$ are of same direction, and so I assume they satisfy

$$(\mathbf{x}_c - \mathbf{X}_A) = \mu (\mathbf{x}_a - \mathbf{X}_A) \tag{98}$$

Where μ is a real number and $0 < \mu < 1$. Thus

$$\mathbf{\Phi}_c = \mu^2 (\mathbf{x}_a - \mathbf{X}_A)^T \mathbf{A} (\mathbf{x}_a - \mathbf{X}_A) = \mu^2 \mathbf{\Phi}_a \tag{99}$$

Note that $\mathbf{x}_a$ is on the surface of the ellipsoid A, thus $\mathbf{\Phi}_a = 1$, i.e. $\mathbf{\Phi}_c = \mu^2$. So that

$$\mu = \sqrt{\mathbf{\Phi}_c} \tag{100}$$

Substitute $\mu = \sqrt{\mathbf{\Phi}_c}$ to equation $(\mathbf{x}_c - \mathbf{X}_A) = \mu (\mathbf{x}_a - \mathbf{X}_A)$,

$$\frac{(\mathbf{x}_c - \mathbf{x}_A)}{\sqrt{\mathbf{\Phi}_c}} = (\mathbf{x}_a - \mathbf{x}_A) \tag{101}$$

Thus

$$\mathbf{x}_a = \mathbf{x}_A + \frac{\mathbf{x}_c - \mathbf{x}_A}{\sqrt{\mathbf{\Phi}_c}} \tag{102}$$

Now I consider two arbitrary ellipsoids **A** and **B** in contact with each other, as Figure 31 shows. $\mathbf{X}_A$ and $\mathbf{X}_B$ are position vectors of centers of ellipsoid **A** and **B**, $\mathbf{x}_c$ is a point inside the ellipsoid, vector $\mathbf{x}_a$ and $\mathbf{x}_b$ be the points on the surface of ellipsoids. $\mathbf{x}_a$, $\mathbf{x}_c$ and $\mathbf{X}_A$ are on the same line; and $\mathbf{x}_b$, $\mathbf{x}_c$ and $\mathbf{X}_B$ are on the same line.

Similar to equation $\mathbf{x}_a = \mathbf{x}_A + \frac{\mathbf{x}_c - \mathbf{x}_A}{\sqrt{\mathbf{\Phi}_c}}$,

$$\begin{cases} \mathbf{x}_a = \mathbf{x}_A + \dfrac{\mathbf{x}_c - \mathbf{x}_A}{\sqrt{\mathbf{\Phi}_c}} \\ \mathbf{x}_b = \mathbf{x}_B + \dfrac{\mathbf{x}_c - \mathbf{x}_B}{\sqrt{\mathbf{\Phi}_c}} \end{cases} \tag{103}$$

Then I will do derivation so that all variables relative with radii of spheres in equation $W = \frac{2Ka^5}{5R_m^2}$, $a = \sqrt{R_m \delta}$, $\delta = R_A + R_B - \|\mathbf{x}_A - \mathbf{x}_B\|$

and $R_m = \frac{R_1 R_2}{R_1 + R_2}$ are replaced. Firstly, subtract the first equation by the second equation in $\begin{cases} \mathbf{x}_a = \mathbf{x}_A + \frac{\mathbf{x}_c - \mathbf{x}_A}{\sqrt{\Phi_c}} \\ \mathbf{x}_b = \mathbf{x}_B + \frac{\mathbf{x}_c - \mathbf{x}_B}{\sqrt{\Phi_c}} \end{cases}$,

$$\mathbf{x}_a - \mathbf{x}_b = \mathbf{x}_A - \mathbf{x}_B - \frac{1}{\sqrt{\Phi_c}}(\mathbf{x}_A - \mathbf{x}_B) \quad (104)$$

Thus

$$\mathbf{x}_a - \mathbf{x}_b = \left(1 - \frac{1}{\sqrt{\Phi_c}}\right)(\mathbf{x}_A - \mathbf{x}_B) \quad (105)$$

Denote

$$\mathbf{d} = \mathbf{x}_a - \mathbf{x}_b \quad (106)$$

Then equation $\mathbf{x}_a - \mathbf{x}_b = \left(1 - \frac{1}{\sqrt{\Phi_c}}\right)(\mathbf{x}_A - \mathbf{x}_B)$ turns to

$$\mathbf{d} = \left(1 - \frac{1}{\sqrt{\Phi}}\right)(\mathbf{x}_A - \mathbf{x}_B) \quad (107)$$

I consider d as the level of deformation of the two ellipsoids, thus δ in equation $\delta = R_A + R_B - \|\mathbf{x}_A - \mathbf{x}_B\|$ can be replaced by d.

$$\delta \Rightarrow d = \left(\frac{1}{\sqrt{\Phi}} - 1\right)\|\mathbf{x}_A - \mathbf{x}_B\| \quad (108)$$

Then I still need to replace R_m in $R_m = \frac{R_1 R_2}{R_1 + R_2}$, in which $R_A R_B$ and $R_A + R_B$ need to be replaced separately.

It is known that the volume of sphere equals $V_{sphere} = \frac{4}{3}\pi r^3$, in which r is the radius of the sphere. From ellipsoid volume formula

$V_{ellipsoid} = \frac{4}{3}\pi a_1 a_2 a_3$. Then I consider the sphere as a special type of ellipsoid that 3 radii are the same. Thus, the radii in term $R_A R_B$ can be replaced by the radii of the ellipsoid, as equation (89), (90) and (91) show

$$R_A \Rightarrow (a_1 a_2 a_3)^{\frac{1}{3}} \quad (109)$$

$$R_B \Rightarrow (b_1 b_2 b_3)^{\frac{1}{3}} \quad (110)$$

$$R_A R_B \Rightarrow \left(a_1 a_2 a_3 b_1 b_2 b_3\right)^{\frac{1}{3}} \tag{111}$$

Note that the generalised Hertz formula should turn to normal Hertz formula with ellipsoids of 3 same radii. Thus, I cannot replace term $R_A + R_B$ in the same way with $R_A R_B$. Denote

$$\left\|\mathbf{x}_A - \mathbf{x}_B\right\| \Rightarrow R \tag{112}$$

Then

$$\delta = \left(\frac{1}{\sqrt{\Phi}} - 1\right) R \tag{113}$$

$$R_A + R_B \Rightarrow \frac{R}{\sqrt{\Phi}} \tag{114}$$

From $R_A R_B \Rightarrow \left(a_1 a_2 a_3 b_1 b_2 b_3\right)^{\frac{1}{3}}$ and $R_A + R_B \Rightarrow \frac{R}{\sqrt{\Phi}}$,

$$R_m = \frac{\left(a_1 a_2 a_3 b_1 b_2 b_3\right)^{\frac{1}{3}}}{\frac{R}{\sqrt{\Phi}}} \tag{115}$$

Substitute $\delta = \left(\frac{1}{\sqrt{\Phi}} - 1\right) R$ and $R_m = \frac{\left(a_1 a_2 a_3 b_1 b_2 b_3\right)^{\frac{1}{3}}}{\frac{R}{\sqrt{\Phi}}}$ to $a = \sqrt{R_m \delta}$,

$$\begin{aligned} a &= \sqrt{\frac{\left(a_1 a_2 a_3 b_1 b_2 b_3\right)^{\frac{1}{3}}}{\frac{R}{\sqrt{F(\mathbf{A},\mathbf{B},\mathbf{s})}}} \left(\frac{R}{\sqrt{\Phi}} - R\right)} \\ &= \left(a_1 a_2 a_3 b_1 b_2 b_3\right)^{\frac{1}{6}} \cdot \sqrt{\frac{\sqrt{\Phi}}{R} \cdot R \frac{1-\sqrt{\Phi}}{\sqrt{\Phi}}} \\ &= \left(a_1 a_2 a_3 b_1 b_2 b_3\right)^{\frac{1}{6}} \sqrt{1-\sqrt{\Phi}} \end{aligned} \tag{116}$$

Substitute $R_m = \dfrac{(a_1a_2a_3b_1b_2b_3)^{\frac{1}{3}}}{\dfrac{R}{\sqrt{\Phi}}}$ and $a = \sqrt{\dfrac{(a_1a_2a_3b_1b_2b_3)^{\frac{1}{3}}}{\dfrac{R}{\sqrt{F(\mathbf{A},\mathbf{B},\mathbf{s})}}}\left(\dfrac{R}{\sqrt{\Phi}}-R\right)}$

$= (a_1a_2a_3b_1b_2b_3)^{\frac{1}{6}} \cdot \sqrt{\dfrac{\sqrt{\Phi}}{R}} \cdot R\dfrac{1-\sqrt{\Phi}}{\sqrt{\Phi}}$ to, $W = \dfrac{2Ka^5}{5R_m^2}$,

$= (a_1a_2a_3b_1b_2b_3)^{\frac{1}{6}}\sqrt{1-\sqrt{\Phi}}$

$$W = \frac{2Ka^5}{5R_m^2} = \frac{2K\left[(a_1a_2a_3)^{\frac{1}{3}}+(b_1b_2b_3)^{\frac{1}{3}}\right]^2(a_1a_2a_3b_1b_2b_3)^{\frac{1}{6}}}{5}\cdot\left[1-\sqrt{\Phi}\right]^{\frac{5}{2}} \tag{117}$$

As discussed in Section 2.4.4, the value of force equals the partial differential of energy against displacement. So that

$$\mathbf{F}^A = -\frac{\partial W}{\partial \mathbf{s}} = -\frac{\partial W}{\partial \Phi}\frac{\partial \Phi}{\partial \mathbf{s}} \tag{118}$$

It is known that

$$\frac{\partial W}{\partial \Phi} = \frac{2K\left[(a_1a_2a_3)^{\frac{1}{3}}+(b_1b_2b_3)^{\frac{1}{3}}\right]^2(a_1a_2a_3b_1b_2b_3)^{\frac{1}{6}}}{5}\cdot\frac{5}{2}\left(1-\sqrt{\Phi}\right)^{\frac{3}{2}}\cdot\left(-\frac{1}{2}\right)\Phi^{-\frac{1}{2}} \tag{119}$$

And

$$\frac{\partial \Phi}{\partial \mathbf{s}} = 2\lambda_0(1-\lambda_0)\mathbf{G}^{-1}(\lambda_0)\mathbf{s} \tag{120}$$

Substitute equation

$\dfrac{\partial W}{\partial \Phi} = \dfrac{2K\left[(a_1a_2a_3)^{\frac{1}{3}}+(b_1b_2b_3)^{\frac{1}{3}}\right]^2(a_1a_2a_3b_1b_2b_3)^{\frac{1}{6}}}{5}\cdot\dfrac{5}{2}\left(1-\sqrt{\Phi}\right)^{\frac{3}{2}}\cdot\left(-\dfrac{1}{2}\right)\Phi^{-\frac{1}{2}}$ and $\dfrac{\partial \Phi}{\partial \mathbf{s}} = 2\lambda_0(1-\lambda_0)\mathbf{G}^{-1}(\lambda_0)\mathbf{s}$ to

$$\mathbf{F}^A = -\frac{\partial W}{\partial \mathbf{s}} = -\frac{\partial W}{\partial \Phi}\frac{\partial \Phi}{\partial \mathbf{s}},$$

$$\mathbf{F}^{con} = -\frac{\partial W}{\partial \mathbf{s}} = -\frac{\partial W}{\partial \Phi}\frac{\partial \Phi}{\partial \mathbf{s}}$$

$$= -\frac{2K\left[(a_1a_2a_3)^{\frac{1}{3}}+(b_1b_2b_3)^{\frac{1}{3}}\right]^2(a_1a_2a_3b_1b_2b_3)^{\frac{1}{6}}}{5}\cdot\frac{5}{2}\left(1-\sqrt{\Phi}\right)^{\frac{3}{2}}\cdot\left(-\frac{1}{2}\right)\Phi^{-\frac{1}{2}}\cdot\left(2\lambda_0(1-\lambda_0)\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right)$$

$$= K\left[(a_1a_2a_3)^{\frac{1}{3}}+(b_1b_2b_3)^{\frac{1}{3}}\right]^2(a_1a_2a_3b_1b_2b_3)^{\frac{1}{6}}\cdot\left(1-\sqrt{\Phi}\right)^{\frac{3}{2}}\cdot\Phi^{-\frac{1}{2}}\cdot\left(\lambda_0(1-\lambda_0)\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right) \tag{121}$$

As Figure 32 shows, the contact force calculated from Hertz model increases much slower than contact force calculated from Lenard potential, which means Hertz model can be used to present much more elastic agents, while Lenard potential describes rigid agents. Therefore, to describe the interaction between pairs of cells, Hertz model is more suitable. Equation

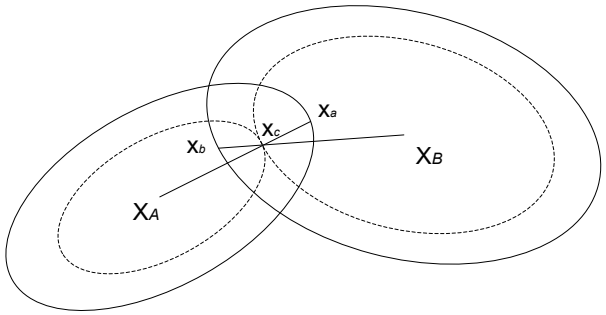


Figure 31: Two ellipsoid A and B. Two points $\mathbf{X}_a$ and $\mathbf{X}_b$ are on the surface of ellipsoid, and $\mathbf{X}_c$ is inside the ellipsoid. Point $\mathbf{X}_a$, $\mathbf{X}_A$ and $\mathbf{X}_c$ are on the same line; and the point $\mathbf{X}_a$, $\mathbf{X}_b$, and $\mathbf{X}_c$ are on the same line. The dotted line represents the potential surface on which point $\mathbf{X}_c$ lies.

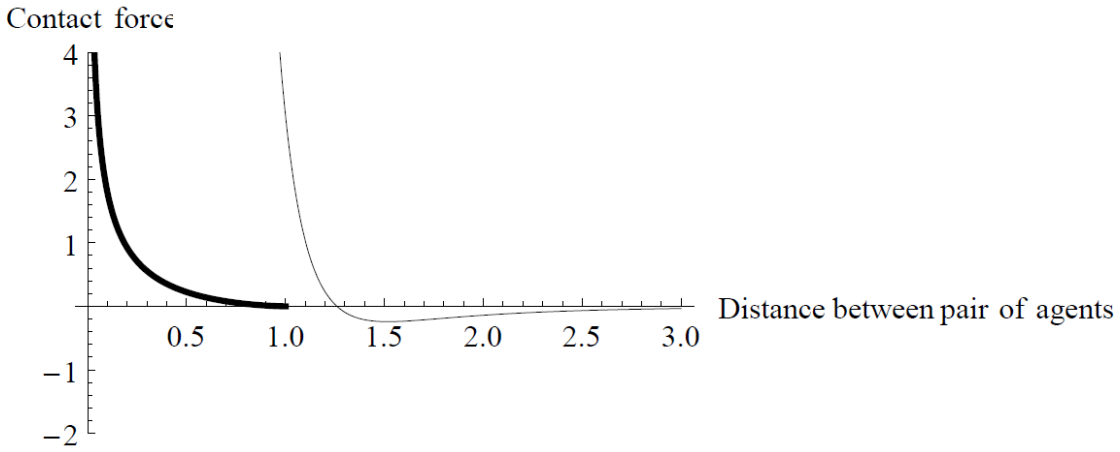


Figure 32: Size of contact force from Lenard potential (thin black line) and Hertz formula (thick black line).

$$\mathbf{F}^{con} = -\frac{\partial W}{\partial \mathbf{s}} = -\frac{\partial W}{\partial \Phi}\frac{\partial \Phi}{\partial \mathbf{s}}$$
$$= -\frac{2K\left[\left(a_1a_2a_3\right)^{\frac{1}{3}} + \left(b_1b_2b_3\right)^{\frac{1}{3}}\right]^2 \left(a_1a_2a_3b_1b_2b_3\right)^{\frac{1}{6}}}{5} \cdot \frac{5}{2}\left(1-\sqrt{\Phi}\right)^{\frac{3}{2}} \cdot \left(-\frac{1}{2}\right)\Phi^{-\frac{1}{2}} \cdot \left(2\lambda_0\left(1-\lambda_0\right)\mathbf{G}^{-1}\left(\lambda_0\right)\mathbf{s}\right)$$
$$= K\left[\left(a_1a_2a_3\right)^{\frac{1}{3}} + \left(b_1b_2b_3\right)^{\frac{1}{3}}\right]^2 \left(a_1a_2a_3b_1b_2b_3\right)^{\frac{1}{6}} \cdot \left(1-\sqrt{\Phi}\right)^{\frac{3}{2}} \cdot \Phi^{-\frac{1}{2}} \cdot \left(\lambda_0\left(1-\lambda_0\right)\mathbf{G}^{-1}\left(\lambda_0\right)\mathbf{s}\right) \quad (122)$$

is the contact force derived from the generalised Hertz formula. The calculation is complex. Like the adhesion force, the contact force should be calculated for each pair of agents. The sum of adhesion and contact force will be used to calculate agent velocity in Section 3.6.

Contact torque and adhesion torque

The contact and adhesion torque are the result of my assumption that the shape of agent is ellipsoid. The torques are used to describe the rotation of agents along an arbitrary axis, as Figure 33 shows. This axis may be coincident with one of the three semi-axes of the ellipsoid, but it also may not; also, generally speaking the axis may not pass the centre of the ellipsoid, which makes the rotation a complex problem. From the torque the angular velocity of the agent can be calculated, which represents the speed that agents rotate along the axis (Figure 33).

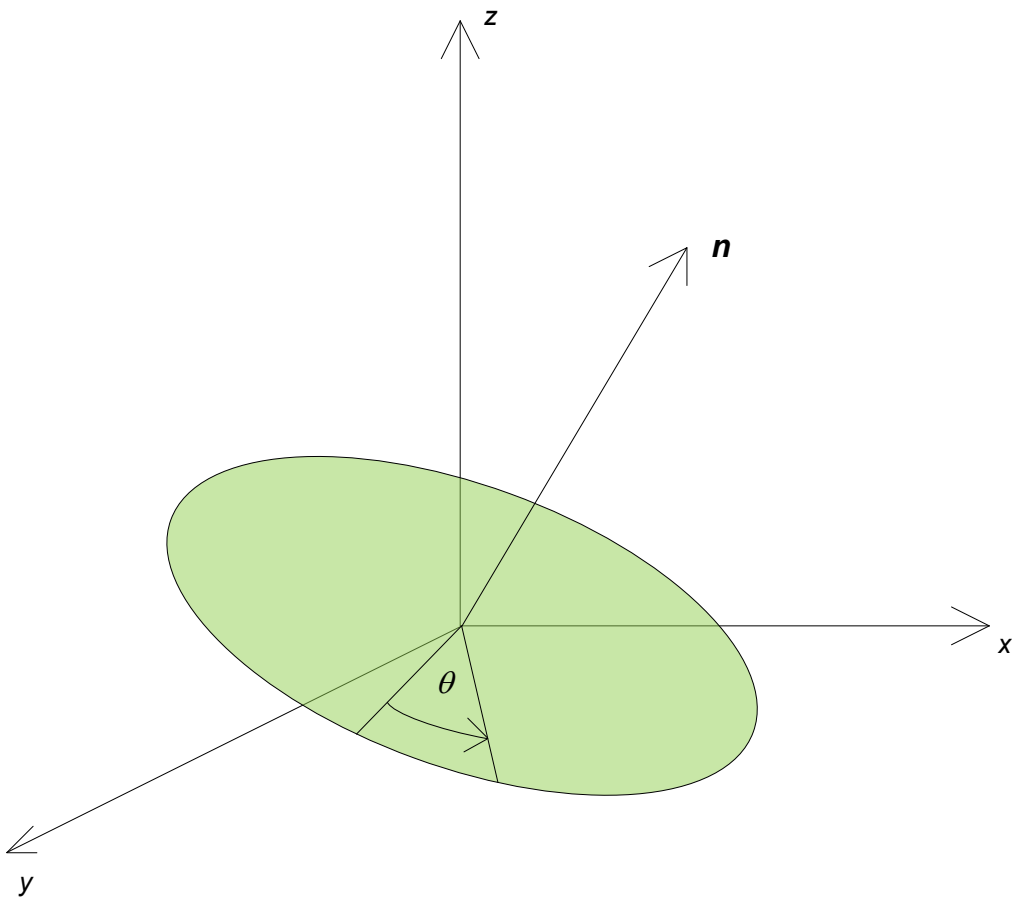


Figure 33: Rotation by angle θ along axis n.

I represent a rotation of angle θ along arbitrary axis $\mathbf{n}$ as $\mathbf{R}_{\mathbf{n}}^{\theta}$, an arbitrary vector as $\mathbf{r}$, after the rotation $\mathbf{r}$ turns to $\mathbf{r}'$. Then $\mathbf{r}'$ and $\mathbf{r}$ satisfy

$$\mathbf{r}' = \mathbf{R}_{\mathbf{n}}^{\theta}\mathbf{r} \qquad (123)$$

$\mathbf{R}_{\mathbf{n}}^{\theta}$ is a matrix as follows

$$\mathbf{R}=\begin{pmatrix} n_1^2(1-\cos\theta)+\cos\theta & n_1n_2(1-\cos\theta)-n_3\sin\theta & n_1n_3(1-\cos\theta)+n_2\sin\theta \\ n_1n_2(1-\cos\theta)+n_3\sin\theta & n_2^2(1-\cos\theta)+\cos\theta & n_2n_3(1-\cos\theta)-n_1\sin\theta \\ n_1n_3(1-\cos\theta)-n_2\sin\theta & n_2n_3(1-\cos\theta)+n_1\sin\theta & n_3^2(1-\cos\theta)+\cos\theta \end{pmatrix} \quad (124)$$

In which θ is the rotation angle; n_1, n_2, and n_3 are co-efficient.

Then $\mathbf{R}_{\mathbf{n}}^{\theta}$ can also be expressed in component form, which is a compact form of matrix form. The component form is useful in differential calculation.

$$R_{kl}=\delta_{kl}\cos\theta-\varepsilon_{klm}\sin\theta n_m+(1-\cos\theta)n_kn_l \quad (125)$$

In which ε_{klm} is a three-dimensional Levi-Civita symbol, and value of ε_{klm} satisfies

$$\varepsilon_{klm}=\begin{cases} +1 \text{ if } (k,l,m)=(1,2,3) \text{ or } (2,3,1) \text{ or } (3,1,2) \\ -1 \text{ if } (k,l,m)=(1,3,2) \text{ or } (3,2,1) \text{ or } (2,1,3) \\ 0 \text{ otherwise} \end{cases} \quad (126)$$

Similar to calculation of the forces $\mathbf{F}=\frac{\partial W}{\partial(\mathbf{A}-\mathbf{B})}$, the calculation of torque can be written as $\hat{\mathbf{o}}=\frac{\partial W}{\partial \mathbf{u}}$, in which W is the energy and **u** is the direction of the axis. First, I derive an expression for the rotation along arbitrary axis. Let $\mathbf{R}_{\mathbf{n}}^{\delta\beta}$ be an infinitesimal rotation angle $\delta\beta$ around **n** axis, here **n** is a normal direction along rotation axis, denote $\delta\hat{\mathbf{a}}=\delta\beta$, in which $\delta\hat{\mathbf{a}}$ is a vector representing rotation of $\delta\beta$ along axis **n**. Then similar to equation $R_{kl}=\delta_{kl}\cos\theta-\varepsilon_{klm}\sin\theta n_m+(1-\cos\theta)n_kn_l$, $\mathbf{R}_{\mathbf{n}}^{\delta\beta}$ can be expressed in component form as

$$R_{kl}(\delta\hat{\mathbf{a}})=\delta_{kl}-\varepsilon_{klm}\delta\beta_m=\delta_{kl}-\delta\beta\varepsilon_{klm}n_m \quad (127)$$

To get an expression of rotation $\delta\beta$, multiply ε_{kln} to both end of $R_{kl}(\delta\hat{\mathbf{a}})=\delta_{kl}-\varepsilon_{klm}\delta\beta_m=\delta_{kl}-\delta\beta\varepsilon_{klm}n_m$, and get $R_{kl}=0-2\delta\beta_n$ here the suffix 'n' is the result of calculation of Levi-Civita symbol. The inverse relationship of Equation $R_{kl}(\delta\hat{\mathbf{a}})=\delta_{kl}-\varepsilon_{klm}\delta\beta_m=\delta_{kl}-\delta\beta\varepsilon_{klm}n_m$ is

$$\delta\beta_m=\delta\beta n_m=-\frac{1}{2}\varepsilon_{klm}R_{kl}(\delta\hat{\mathbf{a}}) \quad (128)$$

Then rotated arbitrary vector **r** becomes new vector $\mathbf{r}'$, their relationship can be written as

$$\mathbf{r}'=\mathbf{r}(\boldsymbol{\beta}+\delta\boldsymbol{\beta})=\mathbf{R}_{\mathbf{n}}^{\delta\beta}\cdot\mathbf{r} \quad (129)$$

Thus, the change of direction of vector **r** is

$$\delta\mathbf{r}=\mathbf{r}'-\mathbf{r}=\mathbf{r}(\boldsymbol{\beta}+\delta\boldsymbol{\beta})-\mathbf{r}(\boldsymbol{\beta})=\left[\mathbf{R}(\delta\boldsymbol{\beta})-\mathbf{I}\right]\cdot\mathbf{r}(\boldsymbol{\beta}) \quad (130)$$

Write Eq. (130) in component form as

$$\delta r_k = r_k(\boldsymbol{\beta}+\delta\boldsymbol{\beta}) - r_k(\boldsymbol{\beta}) = -\delta\beta\varepsilon_{klm}n_m r_l \tag{131}$$

Equation (131) is the relationship of change of $\mathbf{r}$ vector and the rotation $\delta\beta$.

Eq. (131) can be written as

$$\frac{\partial r_k}{\partial\beta} = -\varepsilon_{klm}n_m r_l = \varepsilon_{klm}n_l r_m \tag{132}$$

Which means

$$\frac{\partial \mathbf{r}(\boldsymbol{\beta})}{\partial\beta} = \mathbf{n}\times\mathbf{r} \tag{133}$$

Expand vector **r** to the direction matrix of ellipsoid $\mathbf{u}_l$, the relationship between change of direction of an arbitrary ellipsoid rotated and rotation angle $\delta\beta$ is known:

$$\frac{\partial \mathbf{u}_l}{\partial\beta} = \mathbf{n}\times\mathbf{u}_l \quad (l=1,2,3) \tag{134}$$

In which **n** is rotation axis.

Or in component form

$$\frac{\partial u_{kl}}{\partial\beta} = \varepsilon_{kpq}n_p u_{ql} \tag{135}$$

Similar to equation (2.3) in [89], the torque of ellipsoid A in pair of ellipsoids satisfies

$$\mathbf{n}\cdot\boldsymbol{\tau}^A = n_p\tau_p^A = -\frac{\partial W}{\partial\beta} = -\frac{\partial W}{\partial u_{kl}}\frac{\partial u_{kl}}{\partial\beta} = -\frac{\partial W}{\partial u_{kl}}\varepsilon_{kpq}n_p u_{ql} \tag{136}$$

In which W is the energy, u_{kl} is the direction of the ellipsoid. Then

$$\tau_p^A = -\frac{\partial W}{\partial u_{kl}}\varepsilon_{kpq}u_{ql} \tag{137}$$

Similarly, the other ellipsoid in pair of ellipsoids has torque as

$$\tau_p^B = -\frac{\partial W}{\partial v_{kl}}\varepsilon_{kpq}v_{ql} \tag{138}$$

Note that $W(\mathbf{A},\mathbf{B},\mathbf{s}) = W\left[\Phi(\mathbf{A},\mathbf{B},\mathbf{s})\right]$, which means the energy is function of potential Φ. It is not convenient to calculate $\frac{\partial W}{\partial \mathbf{u}}$,

however according to characteristic of partial differential equation, $\frac{\partial W}{\partial \mathbf{u}} = \frac{\partial W}{\partial \Phi}\cdot\frac{\partial \Phi}{\partial \mathbf{u}}$. Thus

$$\tau_p^A = -\frac{\partial W}{\partial \Phi}\frac{\partial \Phi}{\partial u_{kl}}\varepsilon_{kpq}u_{ql} \tag{139}$$

$$\tau_p^B = -\frac{\partial W}{\partial \Phi}\frac{\partial \Phi}{\partial v_{kl}}\varepsilon_{kpq}v_{ql} \tag{140}$$

From Equation $\Phi(\mathbf{A},\mathbf{B},\mathbf{s}) = F(\lambda_0) = \lambda_0(1-\lambda_0)\mathbf{s}^T\mathbf{G}^{-1}(\lambda_0)\mathbf{s}$, and note that $\mathbf{G}^{-1}\mathbf{G} = \mathbf{I}$

$$\frac{\partial \Phi}{\partial \mathbf{U}} = \lambda_0(1-\lambda_0)\mathbf{s}^T\frac{\partial \mathbf{G}^{-1}(\lambda_0)}{\partial \mathbf{U}}\mathbf{s} = -\lambda_0(1-\lambda_0)\mathbf{s}^T\mathbf{G}^{-1}(\lambda_0)\frac{\partial \mathbf{G}(\lambda_0)}{\partial \mathbf{U}}\mathbf{G}^{-1}(\lambda_0)\mathbf{s} \tag{141}$$

In which U is the direction matrix of ellipsoid. Then it is needed to derive $\frac{\partial \mathbf{G}(\lambda_0)}{\partial \mathbf{U}}$. Using $\mathbf{G}(\lambda) = \lambda\mathbf{B}^{-1} + (1-\lambda)\mathbf{A}^{-1}$,

$$\frac{\partial \Phi}{\partial \mathbf{U}} = -\lambda_0(1-\lambda_0)^2\mathbf{s}^T\mathbf{G}^{-1}(\lambda_0)\frac{\partial \mathbf{A}^{-1}}{\partial \mathbf{U}}\mathbf{G}^{-1}(\lambda_0)\mathbf{s} \tag{142}$$

Or in component form

$$\frac{\partial \Phi}{\partial u_{kl}} = -\lambda_0(1-\lambda_0)^2 s_i G_{im}^{-1}(\lambda_0)\frac{\partial A_{mn}^{-1}}{\partial u_{kl}}G_{nj}^{-1}(\lambda_0)s_j \tag{143}$$

From characteristic of **A** it is known

$$\mathbf{A}^{-1} = \mathbf{U}\bar{\mathbf{A}}^{-1}\mathbf{U}^T = \begin{pmatrix} u_{11} & u_{12} & u_{13} \\ u_{21} & u_{22} & u_{23} \\ u_{31} & u_{32} & u_{33} \end{pmatrix}\begin{pmatrix} a_1^2 & 0 & 0 \\ 0 & a_2^2 & 0 \\ 0 & 0 & a_3^2 \end{pmatrix}\begin{pmatrix} u_{11} & u_{21} & u_{31} \\ u_{12} & u_{22} & u_{32} \\ u_{13} & u_{23} & u_{33} \end{pmatrix} \tag{144}$$

i.e.

$$A_{mn}^{-1} = u_{mr}\bar{A}_{rs}^{-1}u_{ns} \tag{145}$$

$$\frac{\partial A_{mn}^{-1}}{\partial u_{kl}} = \delta_{km}\delta_{lr}\bar{A}_{rs}^{-1}u_{ns} + u_{mr}\bar{A}_{rs}^{-1}\delta_{kn}\delta_{ls} \tag{146}$$

Substituting equation $\frac{\partial A_{mn}^{-1}}{\partial u_{kl}} = \delta_{km}\delta_{lr}\bar{A}_{rs}^{-1}u_{ns} + u_{mr}\bar{A}_{rs}^{-1}\delta_{kn}\delta_{ls}$ into Equation $\frac{\partial \Phi}{\partial u_{kl}} = -\lambda_0(1-\lambda_0)^2 s_i G_{im}^{-1}(\lambda_0)\frac{\partial A_{mn}^{-1}}{\partial u_{kl}} G_{nj}^{-1}(\lambda_0)s_j$

$$\frac{\partial \Phi}{\partial u_{kl}} = -\lambda_0(1-\lambda_0)^2 \left\{ \begin{array}{l} \left[s_i G_{ik}^{-1}(\lambda_0)\right]\left[\bar{A}_{ls}^{-1}u_{ns}G_{nj}^{-1}(\lambda_0)s_j\right] + \\ \left[s_i G_{im}^{-1}(\lambda_0)u_{mr}\bar{A}_{rl}^{-1}\right]\left[G_{kj}^{-1}(\lambda_0)s_j\right] \end{array} \right\} \tag{147}$$

Note that both $\mathbf{G}^{-1}$ and $\bar{\mathbf{A}}$ are symmetric matrix.

$$\frac{\partial \Phi}{\partial u_{kl}} = -2\lambda_0(1-\lambda_0)^2\left[G_{ki}^{-1}(\lambda_0)s_i\right]\left[\bar{A}_{lr}^{-1}u_{nr}G_{nj}^{-1}(\lambda_0)s_j\right] \tag{148}$$

Or in block form

$$\frac{\partial \Phi}{\partial \mathbf{U}} = -2\lambda_0(1-\lambda_0)^2\left[\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right]\otimes\left[\bar{\mathbf{A}}^{-1}\mathbf{U}^T\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right] \tag{149}$$

Substituting equation $\frac{\partial \Phi}{\partial u_{kl}} = -2\lambda_0(1-\lambda_0)^2\left[G_{ki}^{-1}(\lambda_0)s_i\right]\left[\bar{A}_{lr}^{-1}u_{nr}G_{nj}^{-1}(\lambda_0)s_j\right]$ into equation $\tau_p^A = -\frac{\partial W}{\partial u_{kl}}\varepsilon_{kpq}u_{ql}$

$$\tau_p^A = 2\lambda_0(1-\lambda_0)^2\frac{\partial W}{\partial \Phi}\varepsilon_{pqk}\left[A_{qn}^{-1}G_{nj}^{-1}(\lambda_0)s_j\right]\left[G_{ki}^{-1}(\lambda_0)s_i\right] \tag{150}$$

Similarly

$$\tau_p^B = 2\lambda_0^2(1-\lambda_0)\frac{\partial W}{\partial \Phi}\varepsilon_{pqk}\left[B_{qn}^{-1}G_{nj}^{-1}(\lambda_0)s_j\right]\left[G_{ki}^{-1}(\lambda_0)s_i\right] \tag{151}$$

Or in block form, the torques of two non-identical ellipsoids **A** and **B** are

$$\boldsymbol{\tau}^A = 2\lambda_0(1-\lambda_0)^2\frac{\partial W}{\partial \Phi}\left[\bar{\mathbf{A}}^{-1}\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right]\times\left[\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right] \tag{152}$$

$$\boldsymbol{\tau}^B = 2\lambda_0^2(1-\lambda_0)\frac{\partial W}{\partial \Phi}\left[\bar{\mathbf{B}}^{-1}\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right]\times\left[\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right] \tag{153}$$

Where W represents the energy and is the function of potential Φ (Φ is the function of geometry information of ellipsoid A and B, and distance vector s between the two ellipsoids)

$$W(\mathbf{A},\mathbf{B},\mathbf{s}) = W\left[\mathbf{\Phi}(\mathbf{A},\mathbf{B},\mathbf{s})\right] \tag{154}$$

To calculate contact torque, the energy equation $W = \frac{2Ka^5}{5R_m^2} = \frac{2K\left[(a_1a_2a_3)^{\frac{1}{3}} + (b_1b_2b_3)^{\frac{1}{3}}\right]^2 (a_1a_2a_3b_1b_2b_3)^{\frac{1}{6}}}{5} \cdot \left[1-\sqrt{\mathbf{\Phi}}\right]^{\frac{5}{2}}$ is substituted to equation

$$\boldsymbol{\tau}^{A} = 2\lambda_0(1-\lambda_0)^2 \frac{\partial W}{\partial \Phi}\left[\bar{\mathbf{A}}^{-1}\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right]\times\left[\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right]$$

$$\begin{aligned}
\boldsymbol{\tau}^{Con} &= 2\lambda_0(1-\lambda_0)^2 \frac{\partial W}{\partial \Phi}\left[\mathbf{A}^{-1}\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right]\times\left[\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right] \\
&= 2\lambda_0(1-\lambda_0)^2 \left\{ \frac{2K\left[(a_1a_2a_3)^{\frac{1}{3}} + (b_1b_2b_3)^{\frac{1}{3}}\right]^2 (a_1a_2a_3b_1b_2b_3)^{\frac{1}{6}}}{5} \cdot \frac{5}{2}\left(1-\sqrt{\Phi}\right)^{\frac{3}{2}} \cdot \left(-\frac{1}{2}\right)\Phi^{-\frac{1}{2}} \right\} \\
&\cdot\left[\mathbf{A}^{-1}\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right]\times\left[\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right] \\
&= -\lambda_0(1-\lambda_0)^2 K\left[(a_1a_2a_3)^{\frac{1}{3}} + (b_1b_2b_3)^{\frac{1}{3}}\right]^2 (a_1a_2a_3b_1b_2b_3)^{\frac{1}{6}} \cdot \left(1-\sqrt{\Phi}\right)^{\frac{3}{2}} \cdot \Phi^{-\frac{1}{2}} \\
&\cdot\left[\mathbf{A}^{-1}\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right]\times\left[\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right]
\end{aligned} \tag{155}$$

Similarly substitute adhesion potential $\frac{\partial W^{Ad}}{\partial \Phi} = \frac{\partial\left(\varepsilon\sqrt{\Phi}\right)}{\partial \Phi} = \frac{1}{2}\varepsilon\Phi^{-\frac{1}{2}}$ to $\boldsymbol{\tau}^{A} = 2\lambda_0(1-\lambda_0)^2 \frac{\partial W}{\partial \Phi}\left[\bar{\mathbf{A}}^{-1}\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right]\times\left[\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right]$, and adhesion torque can be written as

$$\boldsymbol{\tau}^{Ad} = \varepsilon\lambda_0(1-\lambda_0)^2 \Phi^{-\frac{1}{2}}\left[\mathbf{A}^{-1}\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right]\times\left[\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right] \tag{156}$$

In which $\times$ is cross produce. As Figure 34 shows, the contact and adhesion torque drive agents to rotate until agents are parallel (Figure 34).

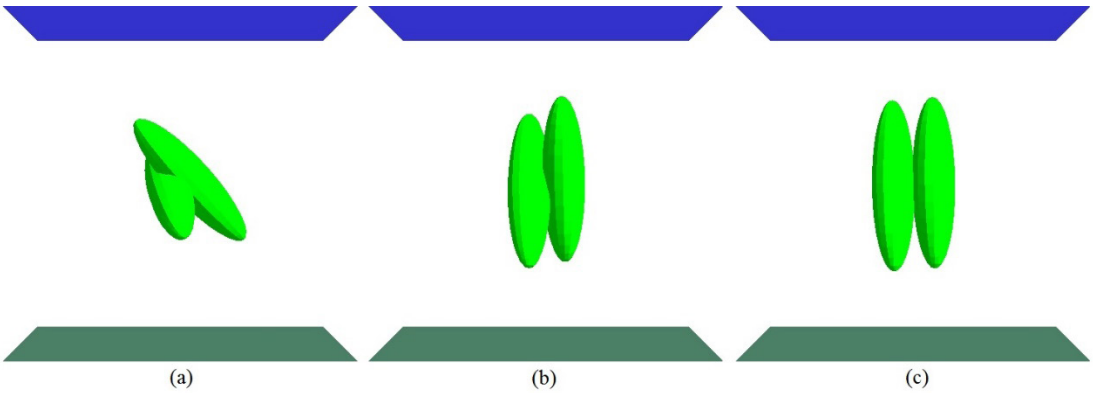


Figure 34: From (a) to (c), two agents (green ellipsoidal particle) under effect of contact and adhesion torque rotate until they are parallel.

Velocity and angular velocity

According to Section 2.3, cells can be considered as particles moving in fluid [43]. Thus, to calculate velocity and angular velocity, the Stokes resistance rule may be applied. The Stokes resistance force and torque are in the following form [72].

$$\mathbf{F}^s = -\mu \mathbf{K}^t \cdot \left(\mathbf{u} - \mathbf{u}_f \right) \tag{157}$$

$$\mathbf{T}^s = -\mu \mathbf{K}^r \cdot \left(\boldsymbol{\omega} - \boldsymbol{\omega}_f \right) \tag{158}$$

In which $\mathbf{u}$ and $\dot{\mathbf{u}}$ are the velocity and angular velocity of an ellipsoid respectively, $\mathbf{u}_f$ and $\mathbf{w}_f$ are the velocity and angular velocity of fluid respectively, and μ is the viscosity co-efficiency. As I assume the fluid in which cells grow and migrate is still, i.e. the fluid has neither velocity nor angular velocity. Thus $\mathbf{u}_f$ and $\mathbf{w}_f$ are zero vectors, and equation $\mathbf{F}^s = -\mu \mathbf{K}^t \cdot \left(\mathbf{u} - \mathbf{u}_f \right)$ and $\mathbf{T}^s = -\mu \mathbf{K}^r \cdot \left(\dot{\mathbf{u}} - \dot{\mathbf{u}}_f \right)$ turn to

$$\mathbf{F}^s = -\mu \mathbf{K}^t \mathbf{u} \tag{159}$$

$$\mathbf{T}^s = -\mu \mathbf{K}^r \boldsymbol{\omega} \tag{160}$$

$\mathbf{K}^t$ and $\mathbf{K}^r$ are Stokes resistance [60] constants that satisfy

$$\mathbf{K}^t = 16\pi \left(\frac{1}{\chi + a_1^2 \alpha_1} \mathbf{u}_1 \otimes \mathbf{u}_1 + \frac{1}{\chi + a_2^2 \alpha_2} \mathbf{u}_2 \otimes \mathbf{u}_2 + \frac{1}{\chi + a_3^2 \alpha_3} \mathbf{u}_3 \otimes \mathbf{u}_3 \right) \tag{161}$$

$$\mathbf{K}^r = 16\pi \left(\frac{a_2^2 + a_3^2}{a_2^2 \alpha_2 + a_3^2 \alpha_3} \mathbf{u}_1 \otimes \mathbf{u}_1 + \frac{a_1^2 + a_3^2}{a_1^2 \alpha_1 + a_3^2 \alpha_3} \mathbf{u}_2 \otimes \mathbf{u}_2 + \frac{a_2^2 + a_1^2}{a_2^2 \alpha_2 + a_1^2 \alpha_1} \mathbf{u}_3 \otimes \mathbf{u}_3 \right) \tag{162}$$

In which $\mathbf{u}_1$, $\mathbf{u}_2$ and $\mathbf{u}_3$ are the directions of each semi-axis of an ellipsoid cell, and a_1, a_2 and a_3 are the lengths of each semi-axis. The α_1, α_2 and α_3 can be evaluated as follows, in which a_1, a_2 and a_3 are three semi-axis of ellipsoid and satisfy $a_1 < a_2 < a_3$. Done

$$\alpha_k = \int_0^\infty \frac{d\lambda}{\left(a_k^2 + \lambda \right) \Delta\lambda} \quad (k = 1,2,3) \tag{163}$$

$$\chi = \int_0^\infty \frac{d\lambda}{\Delta\lambda} \quad (k = 1,2,3) \tag{164}$$

$$\Delta\lambda = \sqrt{\left(a_1^2 + \lambda \right)\left(a_2^2 + \lambda \right)\left(a_3^2 + \lambda \right)} \tag{165}$$

For any ellipsoid in fluid, the sum of contact force and adhesion force should balance the Stokes resistance $\mathbf{F}^s$ and therefore satisfy

$$\sum \mathbf{F} + \mathbf{F}^s = 0 \tag{166}$$

From equation $\mathbf{F}^s = -\mu \mathbf{K}^t \mathbf{u}$ and $\sum \mathbf{F} + \mathbf{F}^s = 0$,

$$\mathbf{F}^s = -\sum \mathbf{F} = -\mu \mathbf{K}^t \mathbf{u} \tag{167}$$

So that

$$\mathbf{u} = \frac{1}{\mu} \left[\mathbf{K}^t \right]^{-1} \sum \mathbf{F} \tag{168}$$

Similar to $\mathbf{F}^s = -\mu \mathbf{K}^t \mathbf{u}$, the sum of contact torque and adhesion torque should balance the Stokes resistance $\mathbf{T}^s$

$$\sum \mathbf{T} + \mathbf{T}^s = 0 \tag{169}$$

From $\mathbf{T}^s = -\mu \mathbf{K}^r \dot{\mathbf{u}}$ and $\sum \mathbf{T} + \mathbf{T}^s = 0$,

$$\boldsymbol{\omega} = \frac{1}{\mu}\left[\mathbf{K}^r\right]^{-1} \sum \mathbf{T} \tag{170}$$

In this way I get the velocity and angular velocity of the cell from the total force/torque working on it.

Cell position and direction update

Given the velocity and angular velocity of a cell and an initial position $\mathbf{P}_0$ and calculated velocity $\mathbf{u}$, the next position after time Δt can be calculated as $\mathbf{P}_1 = \mathbf{P}_0 + \mathbf{u} \cdot \Delta t$. Likewise, given the initial direction of an ellipsoid $\mathbf{R}_0$ and the calculated angular velocity $\dot{\mathbf{u}}$, the next direction after time Δt can be calculated as $\mathbf{R}_1 = (\boldsymbol{\omega} \cdot \Delta t) \cdot \mathbf{R}_0$.

Cell-substrate interaction

Consider the situation of two mirror symmetry identical agents; it can be seen that the interactions are also mirrored: the size of forces and torques are the same, and their directions are mirrored in symmetry, as Figure 35 (a) shows. Further, if I consider the planar substrate as the 'mirror' and the interaction between an agent and the substrate plane as the interaction between two mirror symmetry identical agents (shown in Figure 35 (b)), the interactions shown in Figure 35 (a) and Figure 35 (b) become exchangeable. In this chapter the interaction in Figure 35 (b) is simplified and considered as the interaction between an agent and its mirror image (Figure 35).

There are traditional equations to describe the interaction between an elastic ellipsoid and a substrate (Brenner and Gajdos, 1980). However, in this study the potential between the ellipsoidal particle and substrate is presented in similar shape of equation

$$V(r) = 4\varepsilon\left[\left(\frac{\sigma^2}{r^2}\right)^6 - \left(\frac{\sigma^2}{r^2}\right)^3\right],$$

which makes it suitable to describe a rigid particle, not an elastic particle like cell. The simplified method may not be accurate to describe the cell-substrate interaction, but by simplifying the cell-substrate interaction as the interaction between the agent and its mirror image, I gain a convenient consistency (same level of simplify) between inter-agent and agent-environment interactions. The Hertz formula can be used to calculate contact force between an agent and a substrate plane. There are potential problems with the adhesion force calculation: the properties of the substrate are not introduced to the model; thus, it is hard to calibrate the adhesion force in my model with experimental measurements. It is also hard for the model to simulate different types of agents- substrate adhesion forces or test the response the reaction of agent to different types of substrates. These problems lie outside of the scope of this chapter.

To simplify the cell-substrate calculation, cell-substrate interaction is considered as the interaction between a cell and its mirror image by the substrate plane. As Figure 35 (b) shows, **A'** is not a real cell, but the mirror image of cell **A**. So, all the force and torque on cell A between it and the substrate can be calculated between **A** and **A'**. Denote the ellipsoid as ellipsoid **A** and its mirror image ellipsoid **A'**,

$$\mathbf{A} = \begin{pmatrix} u_{11} & u_{12} & u_{13} \\ u_{21} & u_{22} & u_{23} \\ u_{31} & u_{32} & u_{33} \end{pmatrix} \begin{pmatrix} \frac{1}{a_1^2} & 0 & 0 \\ 0 & \frac{1}{a_2^2} & 0 \\ 0 & 0 & \frac{1}{a_3^2} \end{pmatrix} \begin{pmatrix} u_{11} & u_{21} & u_{31} \\ u_{12} & u_{22} & u_{32} \\ u_{13} & u_{23} & u_{33} \end{pmatrix} \tag{171}$$

As **A'** is mirror image of **A**, they are identical in size and shape, but the direction of semi-axis of **A'** has an opposite z-component.

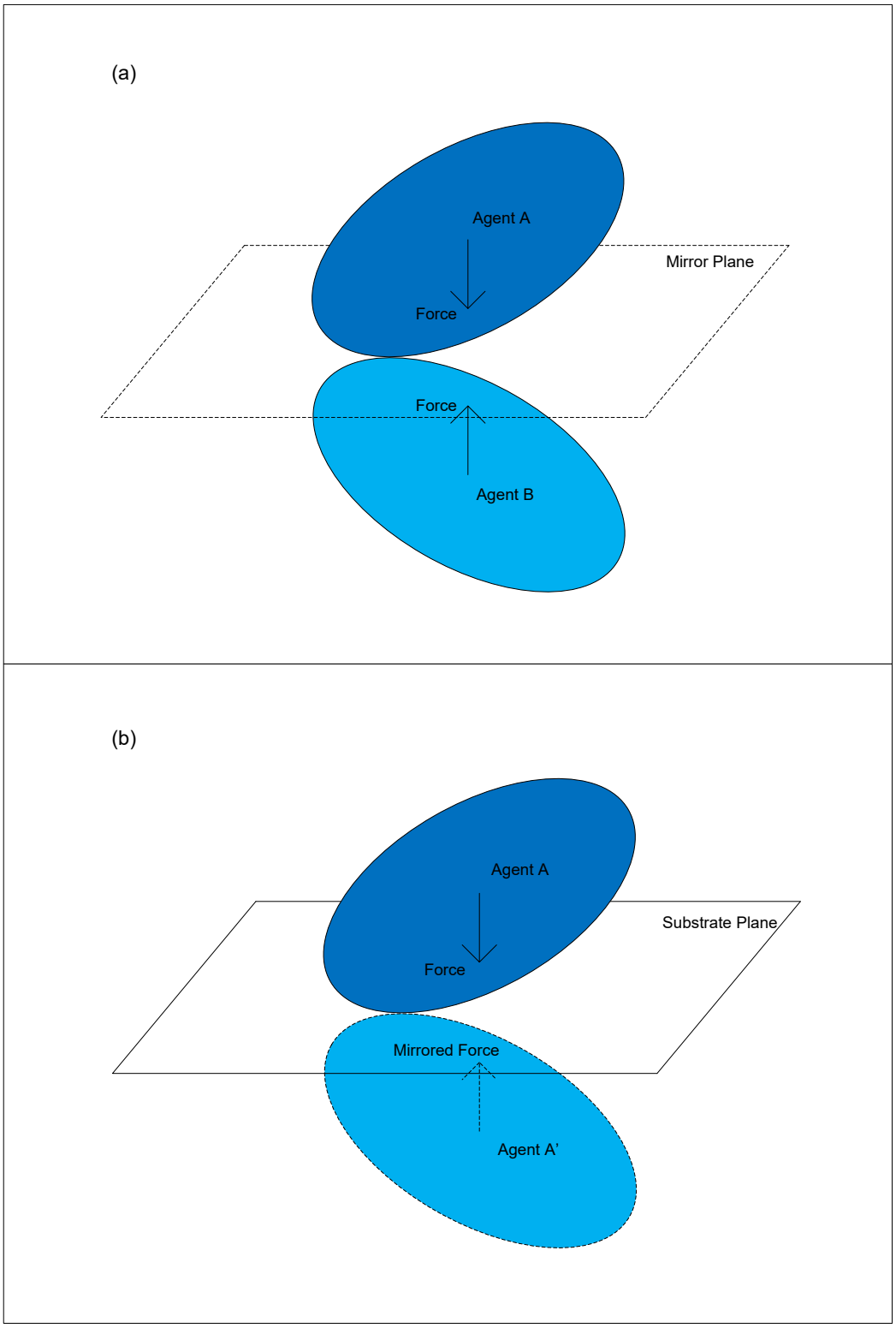


Figure 35: (a) pair of identical agents A and B, can be considered in mirror symmetry by a mirror plane; (b) an agent A interacts with the substrate plane represented as the mirror image of the agent A'.

$$\mathbf{A}'=\begin{pmatrix} u_{11} & u_{12} & u_{13} \\ u_{21} & u_{22} & u_{23} \\ -u_{31} & -u_{32} & -u_{33} \end{pmatrix}\begin{pmatrix} \frac{1}{a_1^2} & 0 & 0 \\ 0 & \frac{1}{a_2^2} & 0 \\ 0 & 0 & \frac{1}{a_3^2} \end{pmatrix}\begin{pmatrix} u_{11} & u_{21} & -u_{31} \\ u_{12} & u_{22} & -u_{32} \\ u_{13} & u_{23} & -u_{33} \end{pmatrix} \tag{172}$$

And $\mathbf{r}_A\left(x_A \quad y_A \quad z_A\right)$, $\mathbf{r}_B\left(x_A \quad y_A \quad -z_A\right)$, so

$$\mathbf{F}^{slab}=\left[\frac{5}{2}K\cdot\left(a_1a_2a_3b_1b_2b_3\right)^{\frac{5}{6}}\left(1-\sqrt{\Phi}\right)^{\frac{3}{2}}\cdot\Phi^{-\frac{1}{2}}\right]\cdot\left[\lambda_0\left(1-\lambda_0\right)\mathbf{G}^{-1}\left(\lambda_0\right)\right] \tag{173}$$

And

$$\begin{aligned}\tau_A^{slab}&=-\left[K\cdot\left(a_1a_2a_3b_1b_2b_3\right)^{\frac{5}{6}}\cdot\frac{5}{2}\left(1-\sqrt{\Phi}\right)^{\frac{3}{2}}\cdot\left(-\frac{1}{2}\right)\Phi^{-\frac{1}{2}}\right]\cdot\\ &\left(-2\right)\lambda_0\left(1-\lambda_0\right)^2\left[\mathbf{G}^{-1}\left(\lambda_0\right)\mathbf{s}\right]\otimes\left[\mathbf{A}^{-1}\mathbf{U}^T\mathbf{G}^{-1}\left(\lambda_0\right)\mathbf{s}\right]\\ &=\frac{5}{2}K\cdot\left(a_1a_2a_3b_1b_2b_3\right)^{\frac{5}{6}}\cdot\left(1-\sqrt{\Phi}\right)^{\frac{3}{2}}\cdot\Phi^{-\frac{1}{2}}\cdot\lambda_0\left(1-\lambda_0\right)^2\left[\mathbf{G}^{-1}\left(\lambda_0\right)\mathbf{s}\right]\otimes\left[\mathbf{A}^{-1}\mathbf{U}^T\mathbf{G}^{-1}\left(\lambda_0\right)\mathbf{s}\right]\end{aligned} \tag{174}$$

However, the size of force and torque between agent and substrate equals that between an agent and another agent. In the case where one agent interacts with a small number of other agents, the agent-agent force may become too large and pull the single agent away from the substrate, which does not match the *in vitro* experiment. To solve this problem, I assume the agent-substrate interaction is 5 times agent-agent interaction. This value, determined by model parameter exploration, allows agents to move across the substrate surface without detaching from the substrate itself.

Dimension analysis

Prior to implementation of the model for use in simulation, there are several parameters, such as the constant in Hertz formula, that need a real-world value. More fundamentally, it is important to determine if, in all equations, no matter whether it is for force, torque or velocity, all the parameters have their own physical interpretation which can be linked to its dimension. Thus, it is needed to ensure every physical property in the model has a correct dimension. Dimension analysis [92,93] is a method that enables exactly this. For full details please refer to Appendix B Step 1, in which dimensions of all physical entities are tested. The list of physical entities and their dimensions is shown in Table 1.

Table 1: The symbol of physical entities and their dimension. Note that for some entities the dimension is 1, which means those entities are scalar entities.

Name of physical entity	Symbol	Dimension
Contact potential variable	λ_0	1
Distance between pair of agents	$\mathbf{s}$	Δx
Shape of ellipsoidal agent	$\mathbf{A}$	$\frac{1}{\Delta x^2}$

Semi-axes of ellipsoidal agent	$a_k\,(k=1,2,3)$	Δx
Intermediate matrix of contact potential	$\mathbf{G}(\lambda)$	Δx^2
Contact potential	$\mathbf{\Phi}(\mathbf{A},\mathbf{B},\lambda)$	1
Young's modulus of elasticity	E	$\frac{\Delta x^2 \rho_c}{\Delta t^2}$
Constants of Contact force	K	$\frac{\Delta x^2 \rho_c}{\Delta t^2}$
Contact force	$\mathbf{F}^{con}$	$\frac{\rho_c \Delta x^4}{\Delta t^2}$
Contact torque	$\mathbf{T}^{con}$	$\frac{\rho_c \cdot \Delta x^5}{\Delta t^2}$
Velocity of agent	$\mathbf{U}_0$	$\frac{\Delta x}{\Delta t}$
Viscosity of environment of agent	μ	$\frac{\Delta x^2 \rho_c}{\Delta t}$
Stokes resistance constants for velocity of agent	$\mathbf{K}^t$	Δx
Angular velocity of agent	ω	$\frac{1}{\Delta t}$
Stokes resistance constants for angular velocity of agent	$\mathbf{K}^r$	Δx^3
Constants of adhesion energy	ε	$\frac{\Delta x^5 \rho_c}{\Delta t^2}$
Adhesion energy	W^{Ad}	$\frac{\rho_c \Delta x^5}{\Delta t^2}$
Adhesion force	$\mathbf{F}^{Ad}$	$\frac{\rho_c \Delta x^4}{\Delta t^2}$
Adhesion torque	$\mathbf{T}^{Ad}$	$\frac{\rho_c \cdot \Delta x^5}{\Delta t^2}$

In Step 2 of Appendix B, values are calculated for the following parameters. The values are listed in Sections 4 and 6.

Table 2: Parameters that the value to be calculated for Sections 4 and 6.

Name of Parameter	Symbol of Parameter
Unit length	Δx
Unit time	ΔT
Constant for contact force	K
Constant for adhesion force	ε

Discussion

Many values for the physical entities are set by making specific assumptions within a parameter range. This range is fixed by the definition of the entities, but the chosen value may change within this range. As such, there are various sets of parameters values for the model that should provide physically possible behaviours. Although the values estimated in this section are based on physical definitions and equation derivation, I may still need to change them during simulation, on account of the need for particular parameterizations for specific models.

Another key aspect is the balance point between adhesion force and contact force. This point determines the amount of elasticity in cells in the model. This point is based on force analysis of two identical cells, and for two non-identical cells I may observe different phenomenon, and this is not considered in the model. Also, when more than two identical cells are sufficiently close, adhesion and contact forces will balance at closer distances because the total adhesion force on each single cell is larger than the situation I analysed. These enhancements would require substantial model development time to address.

Conclusion

In this section a physical model to describe cell-cell interaction and cell-substrate interaction is presented. Dimension analysis of the equation set and estimation of values for constants is also detailed. This model provides a basis for exploration of cell-cell interactions. Since the model is agent-based, I can model the interactions of individual cells, simulating movement of cells for a certain amount of time (this time is decided by which phenomenon I want to simulate). In addition, by modifying the value of parameters, I may control the cells to produce different phenomena, which makes the model a tool to explore the relationship between physical interaction and morphological patterns that may be formed by cells. In Section 4, 6 and 7, this model is used to study two biological systems: early stage of vessel formation and cancer cell growth.

In summary, the model construction assumes the following:

Type of Assumption	Assumption	Reason
Geometry of cell	Cells are represented as ellipsoids.	R Sodt, et al. [100]. Hayashi, et al. [51]
	Cell size and shape do not change during the experiment.	
	Mass is evenly distributed within a cell.	For geometric simplicity
Force	Only contact force, adhesion force and resistance force are significant forces on the cell (density of cell is similar with water thus gravity does not need to be considered).	Grover, et al. [113]
	Contact force can be computed using the Perram-Wertheim approach.	John W Perram, et al. [62]
	Adhesion force can be modelled as a step function on distance using Hertz formula.	Ramis-Conde, et al. [24], Ramis-Conde, et al. [58] and Galle, et al. 2008
	Contact and adhesion forces balance at a defined point when cells are in contact, and the strengths of the forces can be calibrated based on this.	Based on the fact that cells do not totally overlap
	Each cell is considered under effect of resistance force which occurs when cell moves within the fluid.	Beysens, et al. [68]
	Resistance force can be computed using Stokes' law, and the known properties of the fluid medium.	Brenner, H [60]
Kinematics	Cells move at very low speed, so their acceleration approaches zero and the forces upon them are balanced.	JP Rieu, et al. [69] GM Walker, et al. 2005
Cell-substrate interaction	Interactions with the substrate can be modelled as interactions with mirrored cells.	Similarity with identical cell-cell interaction
	The interaction between modelled cell and plane is much larger than interaction between pair of two cells.	Scianna, et al. [6]

Section 4: Vessel Formation: Simulation and Result

Introduction

Along with content of Section 5 and 6, the content of this section is part of my journal paper published on Natural Computing. In Section 3 a generic model is presented to describe the physical interaction among ellipsoidal objects. The next step is to use that model to study specific biological systems in specific contexts. In order to fit the (generic) model to these systems, parameters may need to be adjusted. Additionally, and depending on the characteristic of the system, more components may need to be added, i.e. processes and associated parameters, into the model. Specifically, there is no biological process in the model: of course, different biological systems have different biological processes.

An important feature of the systems that I seek to model is that they can be described by a large number of agents that are controlled by the same underlying physical rule set and can present measurable physical behaviour at the system scale. In outline, the approach is to first develop computer code to implement and then execute the simulation, and then find a method to compare the simulation result with experimental data.

The first target system of study is vessel formation. In this system large numbers of the same type of cells interact to form an emergent and particular pattern that is sustained over a long period of time. I measure the position of cells during the pattern formation and compare the shape of network structure with experimental structures. Changing the parameters in the model affects the shape of network and this can help explain how different behaviours of cells form different vessel formations.

After initial simulations revealed that the behaviour of cells described by my model could, given particular parameterization, lead to a range of emergent structures including biologically plausible network structures, I reached the stage of calibrating the model with *in vitro* experimental data.

In this section, I first discuss the vessel formation simulation, including the impact of different parameters on the emergent structures generated by the simulation.

Micro vessels are formed as an emergent structure within the body by the aggregation of endothelial cells, which themselves are formed by differentiation from stem cells. This formation process has three stages [94]:

- Cell migration and early network formation.
- Network remodelling, where cells connect to each other.
- Further differentiation into tubular structures.

Vessel formation experiments such as [27] typically use endothelial cells and although these cells do change shape and size during the whole process, during the first phrase, which takes place between six and nine hours in vitro, the cell shape and size are the same [29]. In current literature, the experiments are undertaken in petri dishes and over the course of the experiment cells form 2D net-shaped structures.

Thus, vessel formation experiments have a number of features that inform calibration of my model:

- There is only one type of cell, and while the model can simulate behaviours of combinations of cell types, systems of single cell types are an ideal first system.
- The cells maintain the same size and shape over the timescales I am interested in.
- The cells attach to and grow on a (largely) 2D substrate.

Additionally, in the first phase of such vessel formation experiments, cells evidence polarity by aligning in particular directions. Vessel formation experiments explore how the physical interactions among cells, and their low-level physical properties, affect the larger-scale structural patterns in the resulting capillary network. The effects of varying concentrations of growth factors - which have a direct effect on the low-level physical properties of the cells - are of particular interest. In summary the vessel formation experiment is a good candidate system for model calibration.

Modelling this kind of system has value in understanding the underlying biology. Biological interaction among cells is not only hard to observe but also hard to measure. Physical properties such as cell position and orientation are relatively easy to obtain through imaging and are linked to biological mechanisms including signaling pathways that control cell-cell interactions. Therefore, by mapping the simulation of a model to experiments I can analysis the physical interaction among cells and so inform the connection between molecular control mechanisms and tissue-based structures.

The *In Silico* Experiment

Simulation process: The implementation of the vessel formation simulation using the model in Section 3 follows the structure shown in Figure 36. I have chosen an agent-based modelling approach. This allows me to define interacting rules for single cells and examine both the lower-level properties of individual cells and the higher-level behaviour of the system as a whole.

In the simulation, all the agents are stored in a list. Firstly, an agent is chosen from the list; then the simulation calculates the contact force between the chosen agent and each other agent in the list. After that all the contact forces are added up to produce the total inter-cell contact force for each cell. The total inter-cell contacts torque, total inter-cell adhesion force and total inter-cell adhesion torque are calculated in the same way.

Then the simulation calculates the physical interaction between the chosen agent and the substrate. All the forces are added up to calculate velocity according to Stokes equation, and all the torques are added up the calculate angular velocity. At last, according to the velocity and the original position of the chosen agent, its position after 1 unit time is calculated; according to the angular velocity and the original orientation of the chosen agent, its orientation after 1 unit time is calculated.

Each agent in the list is chosen one by one to calculate and update its position and direction. The process is repeated until the desired process time is reached. The flow chart of simulation is shown in Figure 36.

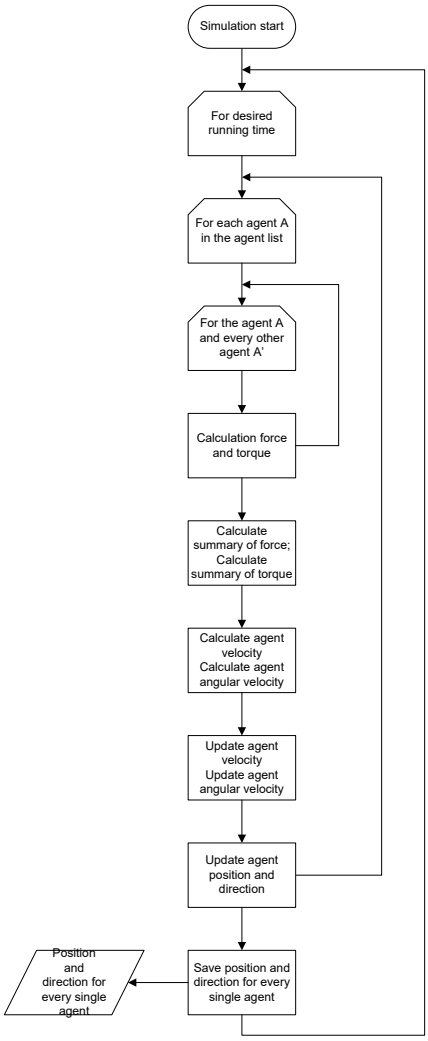


Figure 36: Simulation flow chart.

The agent-based model requires a large number of objects to show distinct emergent behaviour. At the start of the simulation, 2000 agents are randomly placed on a horizontal plane. This number is estimated from the 'middle' number density of cells in [27]. The agents are represented by identical ellipsoids with length of three semi-radius satisfying 1:1:4, as shown in Figure 38. This ratio is chosen because this long and thin shape helps to show the polarity property of cell; at the same time, it is not too long to be unrealistic comparing to real cells observed in experiment, as shown in Figure 37.

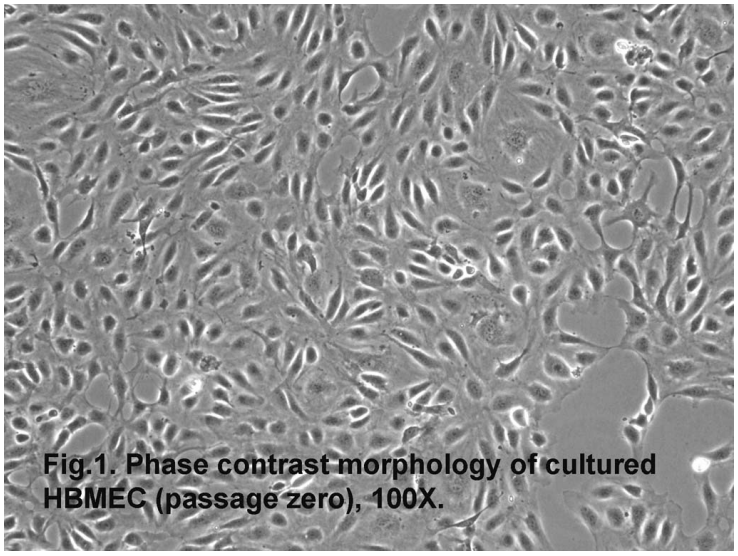


Figure 37: Shape of Human Brain Microvascular Endothelial Cells.

[http://www.sciencellonline.com/site/productimages/large/1000-1-l.jpg].

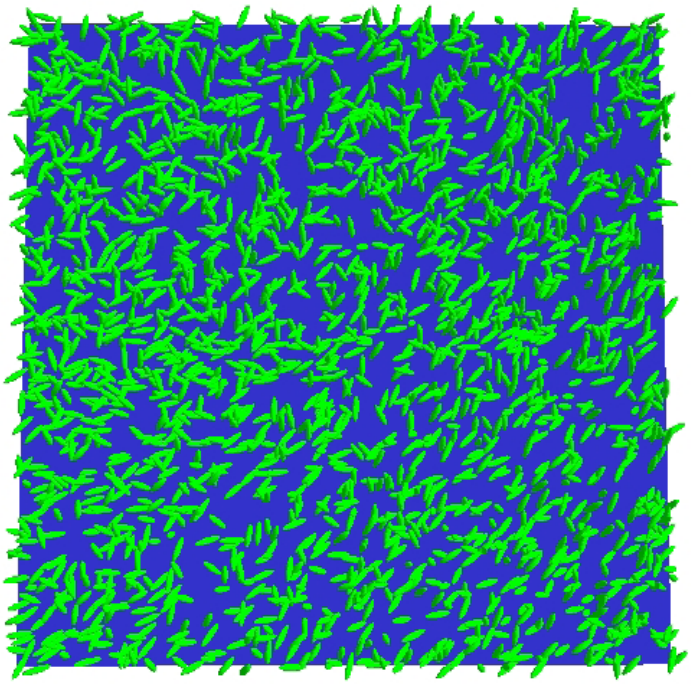


Figure 38: Starting point of a large-scale simulation, in which 2000 agents (green) are randomly placed on substrate plane (blue).

The only restriction on the starting position of cells is that they cannot be too much closer than the distance where the contact force and adhesion force balance in order to allow a realistic balance between contact force and adhesion force. This distribution is implemented in a simple manner: the horizontal position of cells is generated for one cell at a time, its value is randomly distributed within the substrate; the vertical position, on the other hand, is assigned a value that is slightly longer than the shortest semi-axis of agents. The setting of the vertical (with respect to the substrate) position of agents is to simulate the attaching behaviour of *in vitro* cells to the substrate.

The settings of positions are to ensure that the longest semi-axis of agents is horizontal to the substrate and all the agents attach to the substrate along the shortest semi-axis shortly after simulation starts (as Figure 39 shows). The direction of cells is also random. I save to a file the above-mentioned geometry information for each cell separately and store them in a list, which I denote as the 'cell list' in the following discussion, i.e. there is a list in program and each record in this list represents the geometry information of one cell. I have 2000 agents, so the list starts with 2000 records.

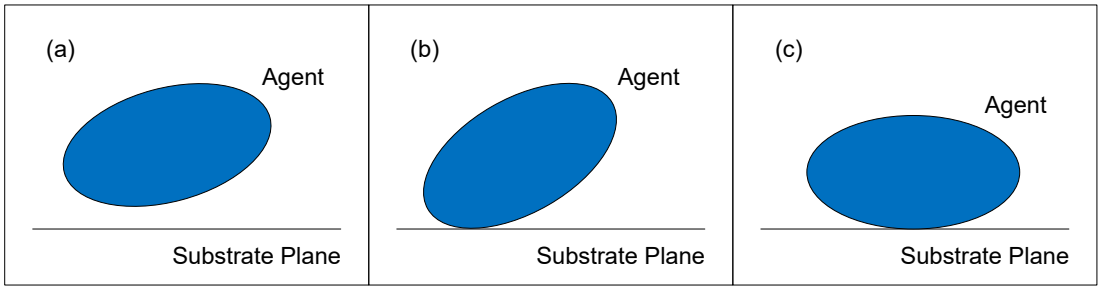


Figure 39: (a-c) The interaction between an agent and the substrate. Firstly, the agent rotates (anti-clockwise) and moves towards the substrate until they contact each other (a-b), then the agent rotates (clockwise) and moves until the longest semi-axis is parallel with the substrate plane (b-c).

Update physical information of cell: When the simulation starts, the program loops through all the cells and for every single cell calculates the contact and adhesion force and torque between it and all the other cells. The program keeps record of the sum of all the contact forces, contact torques, adhesion forces and adhesion torque for each cell, and when the looping is completed for all the cells, the program loops through all the cells again, this time to calculate its interaction with the substrate.

As discussed previously, the interaction between a single agent and the substrate is considered as the interaction between this agent and its mirror image on the surface of the substrate. In the program I do not in reality generate a mirror image and then add it to the cell list; instead, I calculate the geometry information of the mirror image and pass it into functions that calculate interactions, and then only store the calculation result. The forces between one agent and its mirror image are added together with the sum of all the forces, and the torque added to the sum of all the torques. The summary force and torque are used to estimate the velocity and angular velocity of the agent, and then with its original position and velocity the new position of agent is calculated.

Similarly, the original orientation and angular velocity are used to generate the new orientation of the agent. Note that there is an implicit entity in the calculation, which is the time step that one position/ direction update takes. The choice of time step size (i.e. unit time in the model) is an important factor. The time step must be short enough to obtain results at a comparable temporal resolution to the *in vitro* experimental data. However, smaller time steps require more calculation steps to simulate the same length of real-world time. In 3.10 I estimated the value of this time step to be one second. The one-second time step results *in silico* experiments that take an impractically long time to run. It is necessary for model and simulation testing because I can observe the movement of agents with more detail, even though each test run takes up to 120 hours to generate result over biologically meaningful timescales.

The physical information of agents is saved into text file regularly; each time the program generates one new file for all the cells to save their current physical property. As I assume the size and shape of cells do not change, I simply need to keep a record of cell position and orientation. For testing purposes, I also keep velocity and angular velocity for each cell. The interval between data recordings can be adjusted by changing the code of simulation program. Obviously, the shorter the interval is, the larger the files are to be generated, since it provides a more detailed description of cell interactions.

Data files can be imported post-simulation into a separate visualisation program when the simulation is completed. In this way I have an intuitive representation of the shape of structure formed by cells. As the coordinate system in the visualisation framework is different from the one, I am using in the model, a coordinate transform is required when cell position data is imported.

This part was tested beforehand to make sure the cells are at correct place with correct direction. The visualisation can be saved as imaged files for later use.

Simulation parameters: Simulation parameters are also calibrated as described in 3.10. However, early test simulations showed that the agents moved unrealistically rapidly, and this demonstrates that agents are affected by overly large forces. It does not mean the calculation of forces or torques is incorrect, but to investigate their motility the agents need to be slowed down. Thus, inter-agent interaction needs to be reduced. Because the contact force only takes effect when agents contact each other, the adhesion force is the main cause of agent motion. Therefore, the first step is to reduce the size of the adhesion force. However, if the contact force remains the same while adhesion force is decreased, the position where the two forces balance is changed, which means the elasticity of modelled cell is changed. To maintain the elasticity of modelled cell, the contact force and adhesion force should be reduced by the same scale.

By running simulations with various values of forces, it is observed that agents bounce away with high speed when they make contact. A plausible explanation is that the high movement speed of the agents causes a large repulsive contact force and then a high speed in the opposite direction. Thus, I determined that if both forces are reduced by an order of magnitude, the movement speed of agents is reduced to a plausible value. Considering the simplifications used during dimensional analysis (see Section 3.9 and Appendix B for full details), the oversized forces might be caused by the errors in estimation process of constants in equations. In the estimation process, several times the shape of cells is assumed to be a sphere, however in the simulation, the long and thin ellipsoidal shape is used. In other words, the methods used to calculate forces are not wrong. To solve the problem, the constants in contact and adhesion force equations are reduced by an order of magnitude. Clearly, with access to experimental data sets a fuller calibration of these values would be possible.

Table 3: Value of parameter used in vascular formation. Note that the values of K and ε are their dimensional value $\tilde{K}$ and $\tilde{\varepsilon}$. As $\tilde{K}$ and $\tilde{\varepsilon}$ do not appear in Section 3, I use K and ε for a better understanding.

Name of Parameter	Value
Unit length Δx	1.25×10^{-6} m
Unit time ΔT	1s
Density of cell ρ_c	$2000 kg/m^3$
The proportion of typical cell size and unit length	Three semi-radii of ellipsoidal agent: 5.0×10^{-6} m, 1.25×10^{-6} m, 1.25×10^{-6} m
Dynamic viscosity of fluid in which cells are moving μ	$1\times10^{-3} Pa\cdot s$
Constant for contact force K	6.4×10^{10}
Constant for adhesion force ε	3.2768×10^{10}

in silico Experiment Result

Agent movement: It is observed that agents tend to stick to each other at the longest end (as shown in Figure 40), as in biological systems. Note that there is not a biological setting in the model to drive this behaviour of agents. In my model, in addition to adhesion force there is adhesion torque, which drives nearby agents to change their directions while moving together. Additionally, the equation to calculate adhesion torque contains a term representing contact potential, which is multiplied by another term representing the distance of agents. For two fixed agents, the distance is also fixed, but the contact potential is smallest when the two agents have the longest semi-axis pointing to each other. In other words, the adhesion torque reaches its smallest value when two agents are pointing to each other along the longest semi-axis. Any other angle produces a higher adhesion torque, which turns both agents to the direction that produces lower adhesion torque. This phenomenon is an emergent behaviour of pure physical interactions which are determined by shape and positions of pair of agents. I can see that physical rules can drive models to show biology phenomenon such as the anisotropy of modelled cells.

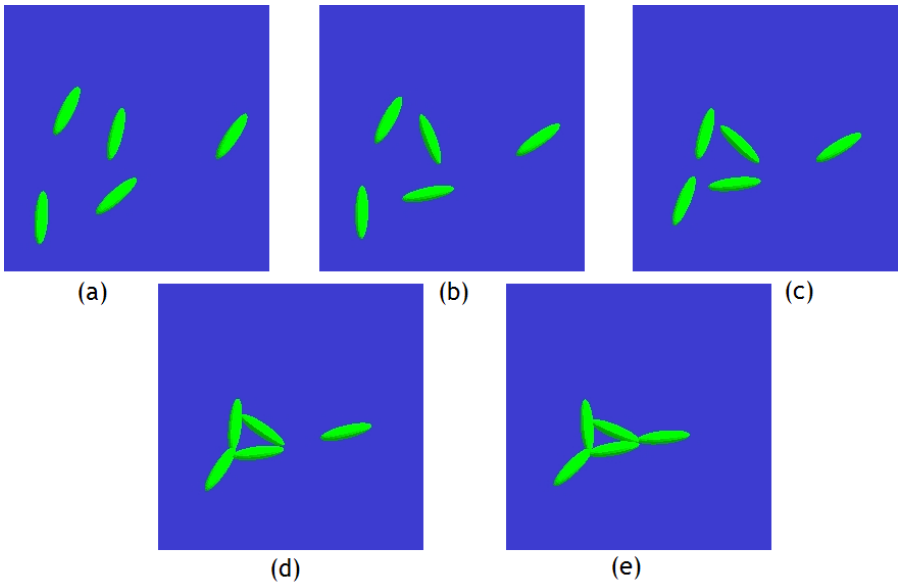


Figure 40: (a-e) In the simulation, agents turn to each other and stick along longest semi-axis.

Typical local patterns: According to this feature outlined in the previous paragraph, there are three typical types of patterns, as shown in Figure 41. Type 1: in the areas where agent density is high, agents may be able to connect with several other agents and form a star-like structure. Type 2: at intermediate densities, agents are close enough to contact each other but not close enough to form the type 1 pattern; instead, agents only connect with their nearest neighbours at both ends and form a line. Type three: in regions of low density, the distance between agents is too large so that agents cannot connect with each other but are attracted to nearby higher density area or remain as separated agents. Together these three types form a net-shaped pattern: type 1 as the junction, type 2 as the branch and type 3 as the holes of the net.

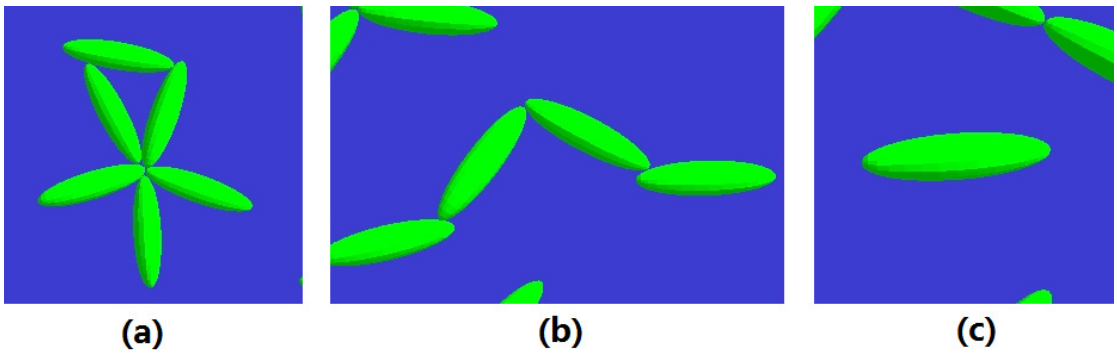


Figure 41: Three typical connect type of agents: (a): 'star' type; (b) single connection type; and (c) non-connection type.

Overall pattern: Over the course of the simulation, agents gradually move towards other nearby agents under the effect of adhesion force resulting in the pattern shown in Figure 42. The balance of adhesion force and contact force eventually stops movement. At the balancing point agents are partly overlapped, and the degree of overlapping shows the level of elasticity of agents Figure 37.

Result analysis

The pattern generated by simulation may be compared with the result vascular network of *in vitro* experiment shown in Figure 43: in both structures, there are junctions composed by group of cells (agents), and the junctions are connected by lines of cells (agents); between the

connected junctions there are holes where separate cells (agents) exist. Both structures have net-shaped features. Similar with cells, the agents tend to connect with each other along the longest semi-axis. The analysis of the measurement of structures is in section 4.1.4.

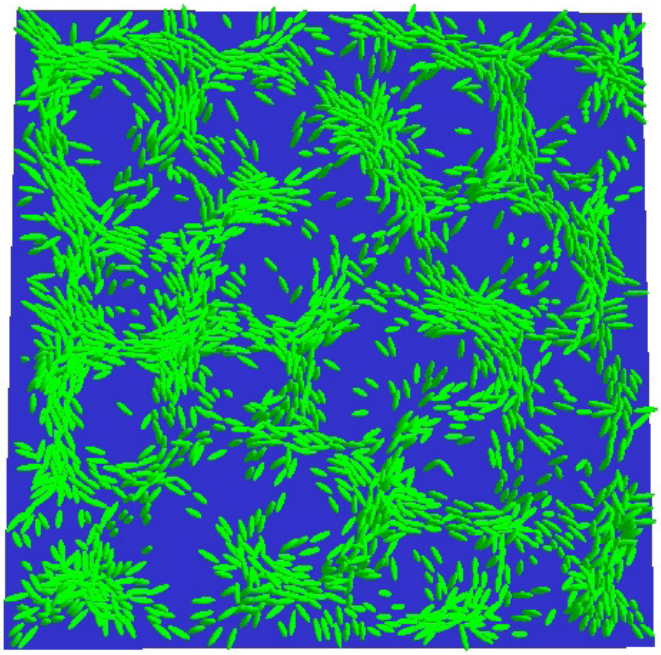


Figure 42: Network structure formed by agents (in green) on substrate plane (in blue).

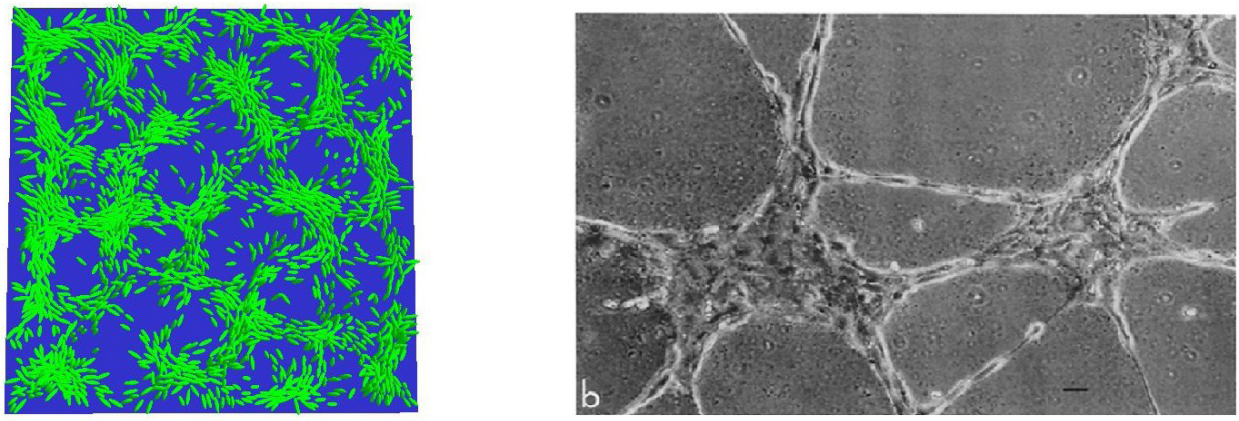


Figure 43: (a) structure generated by simulation; (b) structure generated in in vitro experiment [28].

Pattern stability: However, at this stage, the model is not calibrated with any experimental data, resulting in the adhesion force being much larger and having an effect at longer distances than it should. Agents interact in the simulation, for a period equal to 15 minutes in the real world, to the form structure shown in Figure 44 (middle). Further, as the simulation progresses the nearby cells will gather up and form cell mass and the net structure is destroyed. If I allow the simulation to continue past the state shown above, the pattern will collapse into a few large clusters of cells, as shown in Figure 45 (right). Thus, the observed network structure is transient and not stable.

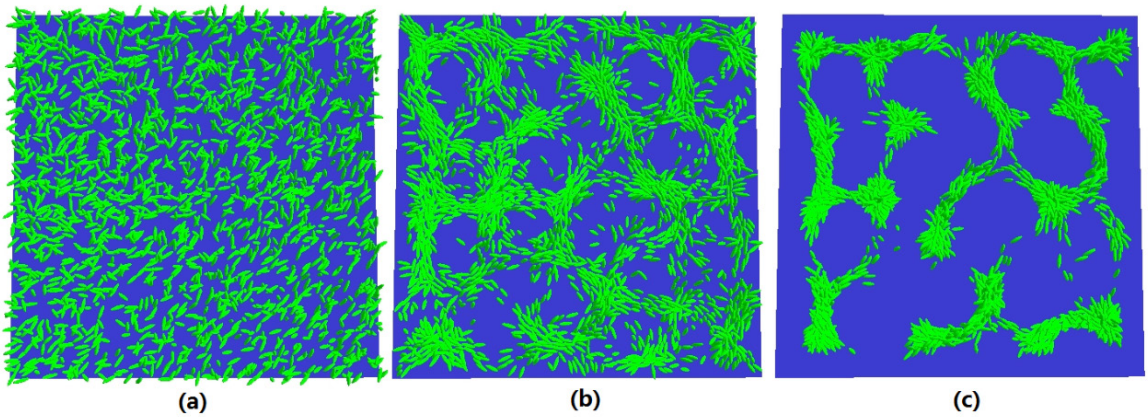


Figure 44: (a) - (c) The starting, middle and end point in of simulation. The structure is formed equal to 15 minutes in the real world from (a) to (b). The structure is collapsed in (c).

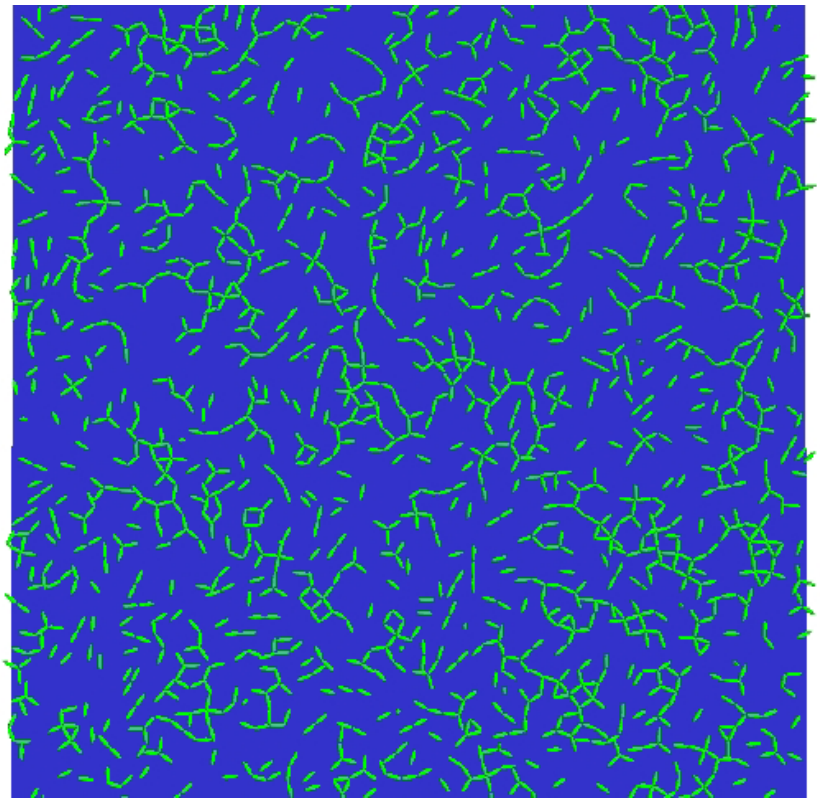


Figure 45: Structure formed by adhesion force that has a finite range of effect.

To generate a stable structure, I have to assume that the adhesion force has a finite range of effects. In this way an agent only connects with a limited number of neighbours. The adhesion force amongst one single agent and its neighbours in one direction may then be balanced by the force amongst it and its neighbours in other directions. Thus, the net-shaped structure is maintained.

By exploring various adhesion force effective range settings, I determined that the best result appeared when contact potential equaled 3. For longer effective ranges, such as 4, the vessel branches are thicker but there are fewer connections between branches; for even longer ranges such as 8, the structure collapses, as shown in Figure 46. With the adjusted model constants, a cell will not adhere to a faraway cell, and the contact force and the adhesion force balances at an appropriate distance so that two nearby cells will not overlap too much.

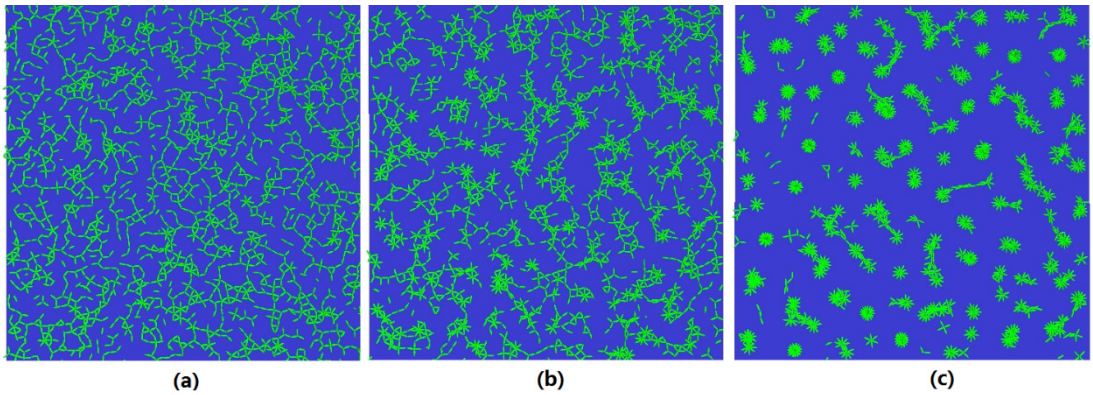


Figure 46: Simulation result of structure formed by 2400 agents with different adhesion force effective range setting. (a) Structure generated with the effective range of adhesion force equals 3; (b) structure generated with the effective range of adhesion force equals 4; (c) structure generated with the effective range of adhesion force equals 8.

This image also shows that there is no typical net structure in this simulation. It is because the cell density is too low. With more cells, the branch-shaped structure can form holes and net-shaped structure, as shown in Figure 47.

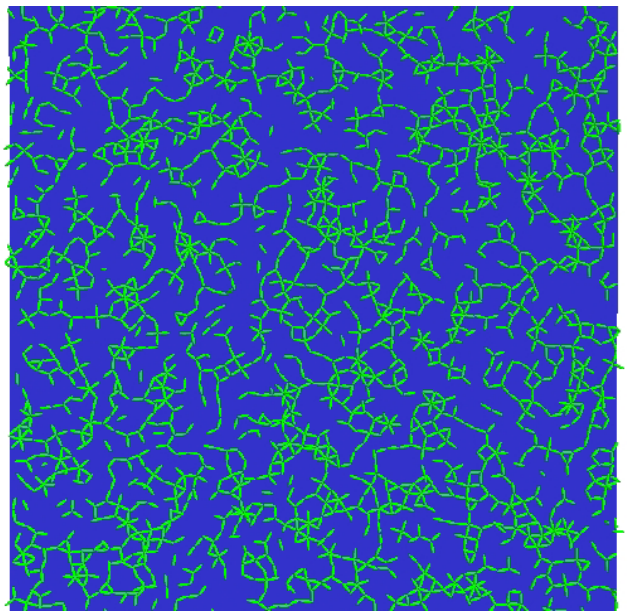


Figure 47: Simulation result with high number density of agents.

This is the simulation result started with 2400 agents. There are more holes formed with single lines of agents. Also, in both simulations shown in Figure 45 and Figure 46 (a) and (b) the structure is stable. As the contact force and the adhesion force are balanced, agents are only affected by the random force used to simulate Brownian motion. In the stable stage, the agents slightly shake at their own position and no agent connection breaks.

Result analysis

Two typical simulation results are shown in Section 4.1.3. I can see that the cells quickly moved towards the substrate and attached to the surface, as Figure 48 shows. Then the cells move towards each other and tend to connect along the longest semi-axis (as shown in Figure 40). After the network is formed, the network is stable and keeps the observed formation (Figure 49).

Figure 48: The blue part is the substrate, from (a) to (d), the green agent moves toward substrate and attach to surface.

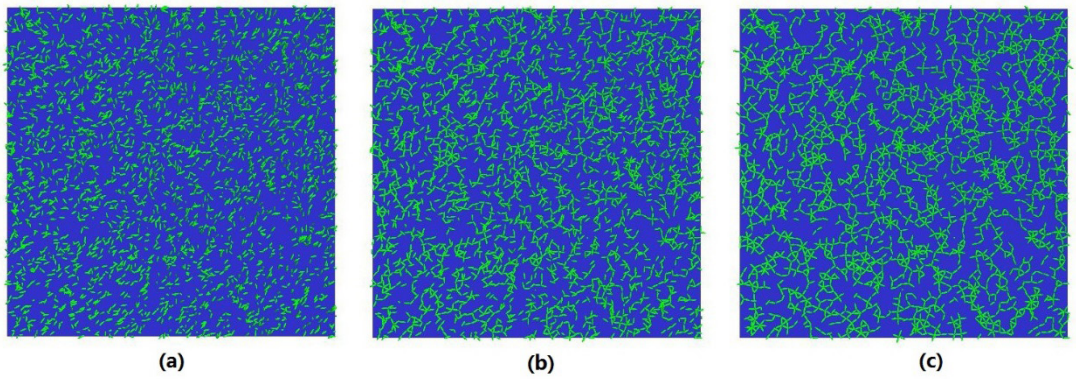


Figure 49: (a) - (c) Screenshot of start, middle and end of simulation.

By comparing the *in vitro* experiment result and *in silico* image (as shown in Figure 44), I can see that both results contain a net-shaped structure. The structure can be considered as holes that are surrounded by several cells, and conjunctions that are where cells gather up. Arguably there is a degree of similarity, but I need a quantitative method to measure both structures for further study.

By using the 'pcf' function 'pairwiseCorrelation' in R [http://127.0.0.1:25934/library/spatstat/html/pcf.html], the pairwise correlation curve is generated, as above diagram shows. In the correlation curves, x-axis represents the distance between arbitrary pair of objects, and y-axis means the chance that two agents occur at that distance.

Comparing Figure 50, I can see that both curves have a peak near the origin, which is a value higher than 1, followed by a valley with a value lower than 1; in the rest part both curves fluctuated around value 1. The experimental curve starts with a high value near the origin, while the curve produced by simulation starts with a low value. Note that Figure 50 is generated by a continuum model and Figure 50 is generated by an agent-based model, the early part of the curve will be affected by the coarse-size of cells, which is very different. As discussed previously in this section, a y-axis value above 1 means the objects tend to cluster at this distance; a y-axis value below 1 means the objects tend to scatter, or repel, at this distance. Therefore, the x-axis value of the valley represents the typical size of hole in the net-shaped structure. The minimum value of simulation is near r=20. By changing the value of parameters in the model, the position of minimum value and other features of pairwise correlation curve can be changed. The details are discussed in section 6.2.1 in the study of cancer cell interactions. Given these broad similarities, and suitable effort to explore the parameter space, including cell size, the simulation results could be better fitted to the experimental data.

In [27], the increased concentration of growth factor increases the size of hole in the pattern. Unlike this model, there is no growth factor in my model. In my model, I assume that the adhesion force plays a similar role in pattern formation. I would need to carry out simulation with various parameter settings to verify this assumption. If confirmed, I would also need to verify the quantity relationship between the concentration of growth factor and adhesion force. However as reported in Section 1, it was not possible to carry out the necessary and model-led set of experiments, and so I converted the model to a new system where I had data streams available to me.

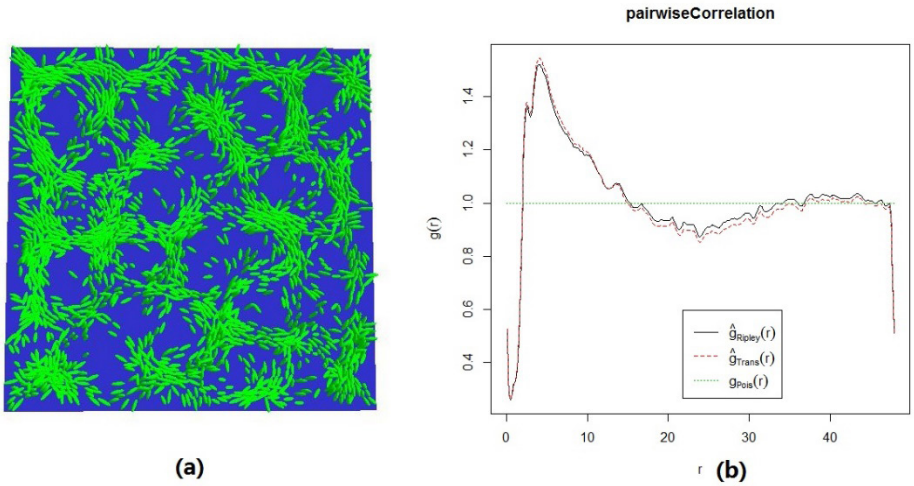


Figure 50: (a) Simulation result of vessel formation; (b) pairwise correlation function of (a).

Conclusion

In this section, simulation process and parameter setting are discussed. The initial results of simulation, of both local and overall pattern, with different parameter settings, are explored. These show that the model has the ability to form higher-scale structure emerging from the interactions among a large number of identical objects controlled by simple rules.

It is also noticed that with certain parameters the net-shaped structure collapses over time, while some parameter settings can keep the shape of structure, which shows the sensitivity of the group behaviour to changes in certain parameters in the model.

The next step is to calibrate the model with experimental data. However, because of the project scope change described in Section 1, I did not have the expected access to the source of experimental data. Hence, I hope to reuse the model in another context where I was able to design experiments and gather the necessary data.

Section 5: Predicting Population Growth Curves by Mechanisms of Drug Action and Hypoxia

Introduction

According to Section 4, the model is able to describe a system that is made of simple objects and shows different behaviour with different parameter settings. However, due to lack of experimental data, I cannot calibrate the model, and therefore want to reuse the model for another system, which has emergent behaviour, and of which I am able to design experiments and gather the necessary data.

In this section, model reuse is discussed in Section 5.1. An important step is to determine systematically to what extent the model for vessel formation may be reused in cancer cell simulations. I use the CoSMoS project [95] to consider in a structured manner how the physical interactions among cells, including the implemented software to simulate it, may be adapted to a different context. As the modes of physical interaction among cells are broadly similar between the two model systems, this seems intuitively to be an appropriate approach - but identifying and revalidating my assumptions using the CoSMoS process both exposes thinking and gives confidence to the simulation's repurposing and might enable the future reuse of the physical model in other contexts.

From repurposing analysis in Section 5.1, a conclusion is reached that the model can be used to describe cancer cell growth with cell cycle and cell population growth being modelled and added to the initial model. Then the cell cycle model to be added in my model is discussed in Section 5.1.1, and the population growth model is introduced in Section 5.1.2.

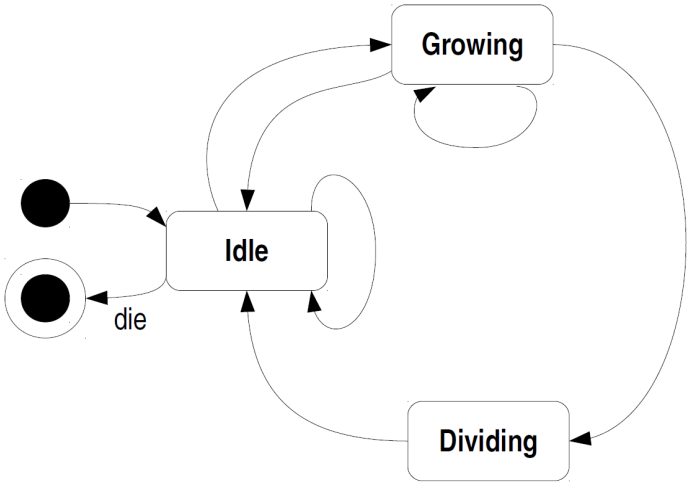


Figure 51: Simplified cell cycle of agent as a state machine [121]. The cell cycle starts from black dot and ends at white dot with black centre. Blocks are states which represent the stage of cell cycle, and arrows show the possible turnover of stage. For example, a cell can turn from Growing stage to Dividing stage because there is an arrow pointing from Growing block to Dividing block. But there is no arrow pointing from the Dividing block to the Growing block, which means cells cannot turn from Dividing stage back to Growing stage.

Then with collaboration between the author and an experimental biologist, a group of experiments are designed to calibrate the cell cycle model and population growth model (Section 5.2). I am interested in the effect of 5-FU, hypoxia and combination of 5-FU and hypoxia on cell growth. In order to model the 5-FU and hypoxia mechanisms, the first experiment is to measure the control population as the base line. In line with this, the model is first calibrated with control population. Then by considering the biological function of 5-FU and hypoxia, a plausible mechanism is added to the model for each intervention. The model initially calibrated for the control population is then able to predict the dynamics of the population in the presence of 5-FU by only invoking that mechanism associated with 5FU. Similarly, the mechanism of hypoxia is added, and the population curves are successfully predicted. At last, for independent validation of the mechanisms added for both 5-FU and hypoxia effects, both mechanisms are set to work together, and simulation result is compared with the experiment of population growth under the effects of combination of 5-FU and hypoxia. Note the model is not calibrated against that combination experiment. These experiments are carried out by out cancer cell biology experts. The experimental result is shown in section 5.3, and the detail of population growth calibration is discussed in section 5.4.

The experiments are carried out on petri dish, which means the cells can be considered as growing on a 2-dimensional plane. The simulations are done with agents on a 2-dimension plane as well (the agents are ellipsoids moving on 2D plane). In future I plan to expand this model to simulate cell growth in 3D space, and the model implementation allows this. The study in 2D space can be considered as a special condition of 3D growth, which is normally referred to as spheroidal growth. It is reasonable to assume that the growth processes in 2D and 3D have some commonalities. In fact [96] suggests that for agent-based models, 2D growth simulation is sufficient to describe generic features of population growth, unless the shape-specific data is required (from simulation). Therefore, the study of 2D population growth is important and its cell-cycle parameters may be used in future spheroid growth simulations.

Model Repurposing

The existing physical model has already demonstrated the ability to reproduce spatial patterns of cell growth resulting from physical interactions within an agent-based simulation, and I have existing tools to visualise and analyse the output from the model. I would like to reuse as much of this infrastructure as possible to build on previously tested work and so maintain simulation quality and reduce development time. I introduced concepts from CoSMoS modelling process to analyse and compare two models, please see [42] for detail. Here I just repeat the summary. I first need to identify the parts of the model that I can keep unchanged. Then, through analysis of requirements for cancer cell growth study, I list the parts of the model that need amended, including reconsidering (and re-evaluating if necessary) previous assumptions, as well as new features that need to be added into model.

Although the behaviour of cancer cells is different from that of endothelial cells, each is system of cells of the same type that are controlled by the same set of physical interaction rules. For this reason, I keep using agent-based modelling process. I consider cancer cells as ellipsoids

as well, so that the method to calculate physical properties can be reused, except for some of the parameters that need re-evaluated. The value of some parameters can be obtained from *in vitro* experimental results, as discussed in Section 5.2.

There is one parameter, however, that does need re-evaluation but cannot be obtained from analysis of experiment results, which is the time step in the simulation. As the growth and movement of cells are very slow, and more importantly, I am not able to measure cells very frequently (see next section) the time step, i.e. ΔT in Section 3, has requirement to be less than 10 minutes, while in simulations in Section 4 the value of this parameter is set as 1 second. Since this is one of basic unit entities I need to estimate the values of other constants, in the new model those other constants will need to be adjusted in line with this change in time step. According to 'Step 1' in Appendix B, the terminate velocity is within a constant range regardless of the value of ΔT. Thus ΔT can be increased from 1 second to 10 minutes, with increase of the unit length Δx.

Because cancer cells grow and die/divide, my assumption that cell size and shape do not change is no longer valid. This requires adding a couple of new mechanisms to the model. One of these mechanisms is the cell cycle, in which agents gradually change their shape or size, and all the forces and torques are calculated according to the changing shape and size. When the cell reaches a certain age, it may die or divide into two daughter cells. The chance it that dies, termed death rate, is the second mechanism to be added. Since I am using the agent-based modelling process, the death rate is added as a rule affecting all the agents. By affecting the chance that every single agent duplicates itself, the death rate affects the population of a group of agents over time. These important additions are described in Section 5.1.1 and 5.1.2.

Modelling the cell cycle: The cell cycle model part is independent from the physical part of the model, and thus can be calibrated separately and is the focus of this Section. It is also necessary to calibrate the cell cycle model first based on population growth data, as to test the physical part of model, the spatial analysis is required to analysis the pattern generated from model simulation, for which the population dynamics of the agents is crucial (see Section 6).

Cells can be considered as state machines [97]. As discussed in 4.2.2, to simulate cancer cell growth and division I need to include a cell cycle into the model, because during this time the cell will change its size, and this may affect the physical interaction. A typical cell cycle includes 3 phases: quiescent, interphase and cell division [73]. Summarized for convenience here and see section 2.4.1 for more detail on the cell cycle, in each phase the cell completes certain tasks and has to pass a checkpoint to go to the next phase. The quiescent phase or called G_0 is a resting phase. The interphase consists of three sub-phases: G_1, S and G_2. The cell increases its size in G_1, then it has to pass G_1 checkpoint to get into phase S, in which the DNA is replicated. Then the cell gets into G_2 and continues to increase its volume and passes the G_2 checkpoint to get into the mitosis phase, also called the cell division phase.

Importantly the biological activity of the cell is not the focus of the model here. I only consider the external phenomenon of cell behaviour, regardless of its internal mechanisms. I simply need to account for different physical phenomena, such as increase in cell volume, regardless of the underlying biological mechanism, such as replication of DNA. So, this biological cell cycle is abstracted to cell stages in my model, which correspond to external, physical cell behaviours.

Thus, in the model there are four stages: a Stable stage, in which cell does not change at all; a Grow stage, in which cell gradually increases its volume; a Split stage, for cell proliferation, in which a cell divides into two daughter cells; a Die stage, in which cell dies and disappears. These are simplifications of cell behaviour as described in section 2.4.1 and may be linked to underlying biological mechanisms in the way that they change the phenomenological model. For example, in real systems cells will not simply disappear when they die, some of the cells gradually shrink; while others expand and at last explode [98]. By simplifying the cell cycle to four phenomenological stages, I can focus on the physical properties of and interaction among cells, which is the key focus of the modelling.

The following state machine diagram represents the four stages and the transitions between them. The stages are in the form of circles (simple circle, double circle or black edged circle) and the transformations are represented as arrows from the start stage pointing at end stage.

Figure 51 shows the state machine that models a simplified cell cycle and drives the behaviour of the simulated cell. This represents the observed behavioural modes of the cell – quiescence, growth, reproduction and apoptosis. The black circle is the start state. It means that every agent starts then turns to the Idle stage at once. Then the arrow pointing from the Idle status to itself means that an agent may decide to stay in the Idle stage, corresponding to the cell quiescence in the real system. The agent may also decide to switch to the Growing state, corresponding to the cell growth in the *in vitro* experiments. This switch is represented by an arrow from the Idle stage to the Growing stage.

As an example of reflecting biological processes in a phenomenological model consider a particular intra-cellular signaling pathway called Hippo. Environmental signals such as local cell density may be regulated by signaling pathways such as Hippo. The cell can detect its local environment by membrane receptors: if the cell detects limited space in the surrounding tissue, the Hippo pathway is triggered to block the

transcription of gene in the cell nucleus [53]. This process is also called contact inhibition and is why in the model I keep a switch from the grow state to stable state. Cancer cells, on the other hand, may have dysregulation in this pathway and so are not controlled by the contact inhibition, so by turning on/off this switch in the model I can simulate cells with normal/cancer contact inhibition.

When an agent is in the Growing stage, the agent may remain in this state, which is represented by an arrow starting and ending on the Growing stage. This relates to agents increasing in volume in the *in vitro* experiments. In the model, when an agent stays in the Growing stage; it gradually increases its volume. I assume that the volume of an agent increases linearly in the Growing state. The rate of increase is calculated by measuring the change of volume and total time spent in the state. As discussed above, the agent may switch between the Idle and Growing state, and this defines the total growth time, in which the agent doubles its volume.

When an agent grows to twice its volume, it will transfer from the Growing stage to the Dividing stage. This is a stage that agents cannot remain in: every agent that is transferred into this stage will be divided into two daughter agents and both daughter agents will transfer to the Idle stage at once.

Finally, there is a typical length of time an agent can live, which is referred to as 'typical agent age' in the following sections. As agents may stay in the Idle stage for a period of time, or switch between Idle stage and Growing stages several times, agents may reach this typical age without being fully grown. In this case the agent will die, which is indicated by the arrow starting from the Idle stage and ending at the die stage (white circle with black dot in the centre). I also need to accommodate the fact that in real experiments the time each individual agent can live may vary, so that I assume that the real cell age is normally distributed with the typical age as the expected (mean) value with an appropriate variance (Standard deviation).

In general, an agent cannot outlive its typical age, except if it has a relatively large variance in age distribution, or if it is already in the Dividing stage. For a normal agent not being affected by any external factor, it stays in the Growing stage until reaching its full size (double its original volume). The length of the Growing stage should be the time needed for the agent to double its volume. In the model I also set a typical growing time with a degree of variability (meaning, standard deviation). However, some drugs and the hypoxia condition may arrest agents at certain stages (see below). The arrested agents die when they reach the specified age.

The length of the Dividing stage is fixed and will not be affected by drug action or agent age. This means that if an agent gets into the Dividing state, it will definitely split no matter whether it reaches the maximum age or not. The die stage is an instantaneous process. Dead agents are moved out of the group immediately. Although it is known that real agents take time to go through the apoptosis process, this stage is not the focus of my study, so I introduce this simplification.

5-fluorouracil (5-FU) is drug used to treat a number of solid tumours. 5-FU competes with the natural substrate for the enzymes in DNA synthesis. The complex formed by 5-FU and enzyme is unable to perform the normal catalytic reaction which is crucial for DNA synthesis. In this way the cell affected by 5-FU stays in DNA synthesis phase of cell cycle and is therefore 'arrested' from normal cell cycle and unable to proliferate [99].

Oxygen is required for the growth of tumour cells. Lack of oxygen supply, or hypoxia, is a common phenomenon in the inner layer of solid tumour [43]. Hypoxia also triggers several pathways such as HIFs (hypoxia inducible factors) and VEGF (vascular endothelial growth factor) [44]. The former promotes cell survival [100], and the latter induces vascular growth near tumour cells [44]. Also, both simulations in [43,44] show a slowed cell cycle with hypoxia, and study in [88] shows a slowing of tumour cell proliferation with moderate hypoxia.

Population cell growth modeling: The group population growth of cells is another mechanism that needs to be added to model, Which I implement by introducing the 'death rate'. The death rate represents the chance that an agent dies when it gets to its typical age. Although for single agent, it only controls the choice of death or division, the death rate may affect the emergent behaviour with a group of agents. As shown in Figure 52, by drawing the number of agents over time one can obtain a growth curve (see Section 2.3.1). In the model the death rate is represented by a real value between 0 and 1; 0 means 0% death rate, with which the agent will certainly divide; 1 means 100% death rate with 0% death rate the agent will certainly die (Figure 52).

I periodically counted the number of agents in all experiments and so can plot the population growth curves. By comparing experimental growth curves with simulation growth curves, I can determine how well the model is calibrated with respect to population dynamics. The death rate may be affected by the environment and particular characteristics of the cell line. Since I grow cells of a single cell line in petri dishes with homogeneous media, I assume all the agents have the same death rate, while the value of death rate may vary over time due to the change of environment (temperature fluctuations, drug effect degradations, etc.).

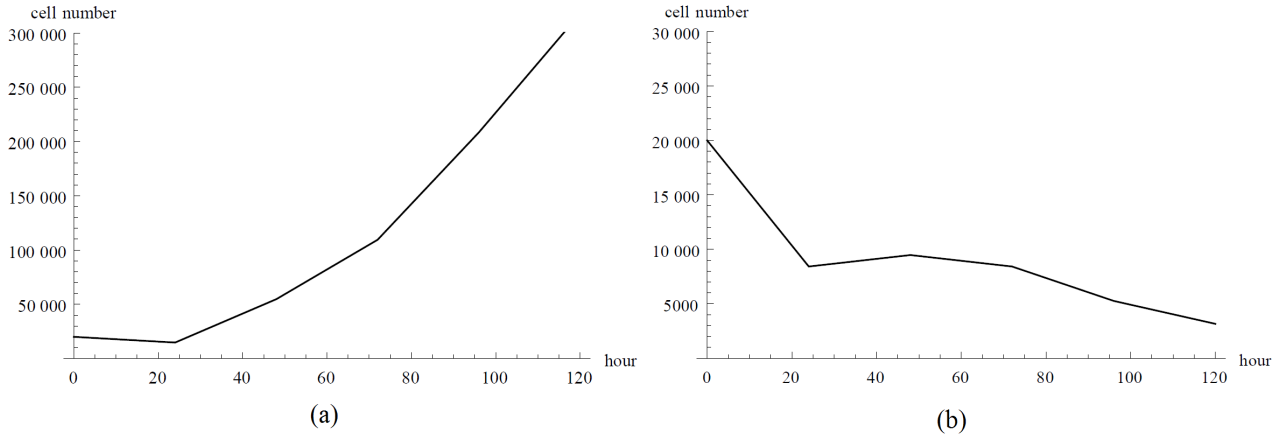


Figure 52: Typical growth curve with increase (a) and decrease (b) total number of agent (graph is drawn with experimental data of control group and combination group; to clearly show the trend of increase/decrease, the scales in (a) and (b) are different, see Section 5.5 for formal plot with same scale).

In ideal conditions the space and nutrient levels for each cell are unlimited, so that the total number of cells has no limit either and obeys an exponential rule (Antonio Brú et al. 2003). The exponential rule can be written as equation as follows.

$$\frac{dN}{dt} = r_{Grow} N \tag{175}$$

Where N is number of agents, $\frac{dN}{dt}$ is change of number of agents, and r_{Grow} is specific growth rate. If $r_{Grow} > 0$, the number of agent increases; if $r_{Grow} = 0$, the number of agents stays the same; if $r_{Grow} < 0$, number of agent decreases. Integrating (154) I can get

$$\ln N + C_1 = r_{Grow} t \tag{176}$$

$$e^{r_{Grow} t - C_1} = N \tag{177}$$

In which C_1 is a constant. To determine its value, I assume at time $t = 0$, the number of agents is N_0. So that

$$\ln N_0 + C_1 = 0 \tag{178}$$

$$C_1 = -\ln N_0 \tag{179}$$

And equation (156) becomes

$$e^{r_{Grow} t + \ln N_0} = N_t \tag{180}$$

$$e^{r_{Grow} t} \cdot e^{\ln N_0} = N_t \tag{181}$$

$$N_0 e^{r_{Grow} t} = N_t \tag{182}$$

Now I let *t* be the time that a group of agents doubles its population, and then I get $t = \ln\frac{2}{r_{Grow}} \approx \frac{0.693}{r_{Grow}}$. I call this *t* as doubling time (Figure 53).

In the real experiments, the size of petri dish is limited, and there will be not enough nutrients for an unlimited number of cells. Under this condition, the total number of cells is thus limited. I introduce a new variable called saturation density to represent the limit that an environment may support, which is represented by *K*.

Then the previous equation becomes $\frac{dN}{dt} = r_{Grow}N\left(\frac{K-N}{K}\right)$. When $N > K$, $\left(\frac{K-N}{K}\right) < 0$, number of agent decreases; when

$N = K$, $\left(\frac{K-N}{K}\right) = 0$, number of agent stays the same; when $N < K$, $\left(\frac{K-N}{K}\right) > 0$, number of agents increases.

Incorporating $\frac{dN}{dt} = r_{Grow}N\left(\frac{K-N}{K}\right)$ into model results in a growth curve shown as Figure 54.

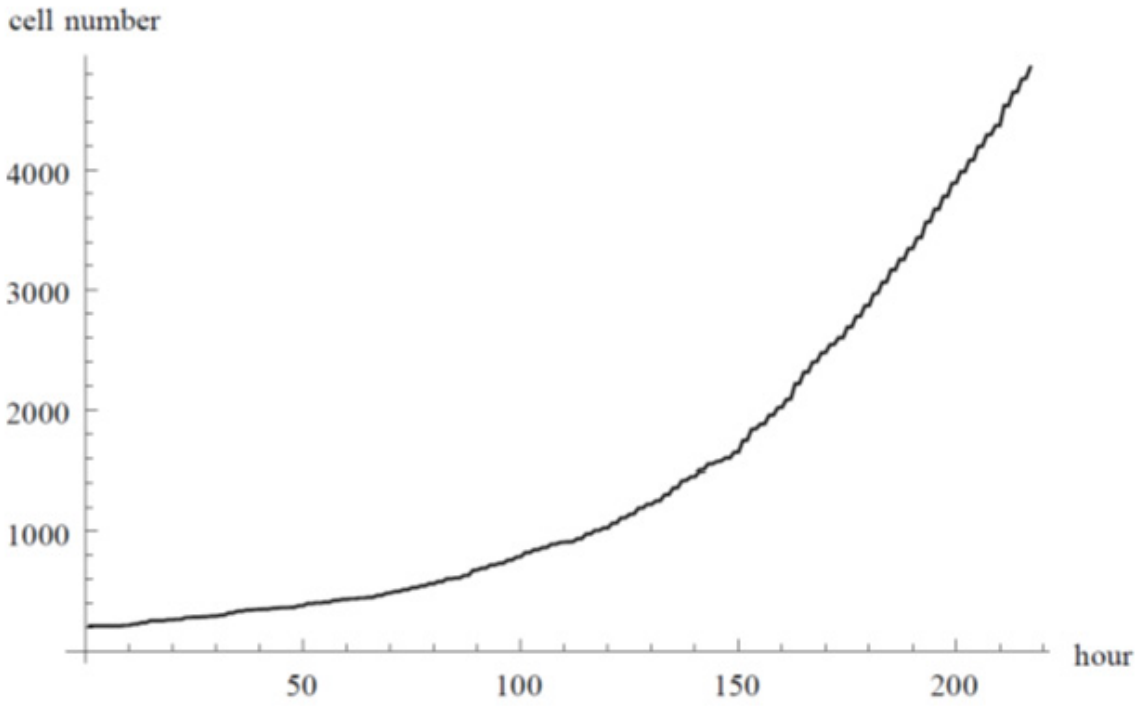


Figure 53: Typical exponential growth curve (graph is drawn with result of simulation with fixed $r_{Grow} > 0$).

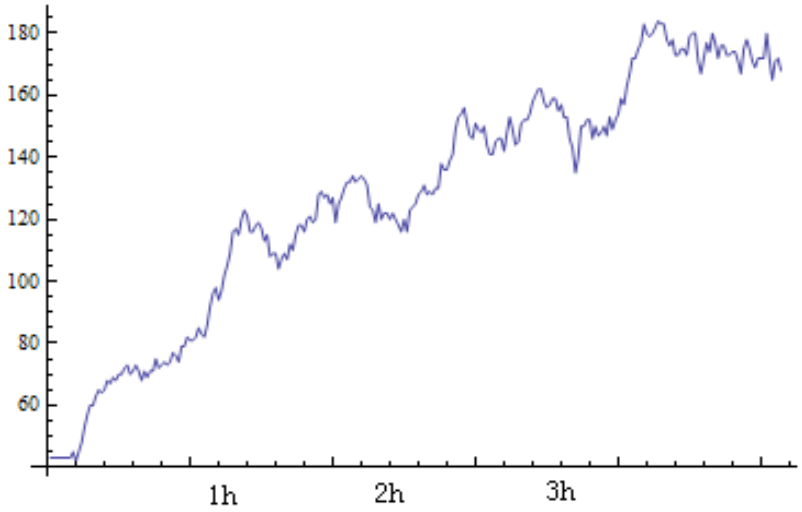


Figure 54: Typical limited growth curve with *K = 180*; x-axis is time, y-axis is number of cells; the zero value near the origin is produced by plotting software. After 3 hours, the curve waves around 180.

I assume that all the agents have the same death rate, and all agents reach the end of cell cycle at exact same time. For N agents with the same death rate r_d, there will be $N \cdot r_d$ agents die and $N \cdot (1 - r_d)$ agents dividing into two identical agents. Therefore, there will

be $2N \cdot (1 - r_d)$ agents in the next cell cycle. I can substitute this into the previous equation $\frac{dN}{dt} = \mu N$, in which $\frac{dN}{dt}$ is the change of number of agent and equals $2N \cdot (1 - r_d) - N$.

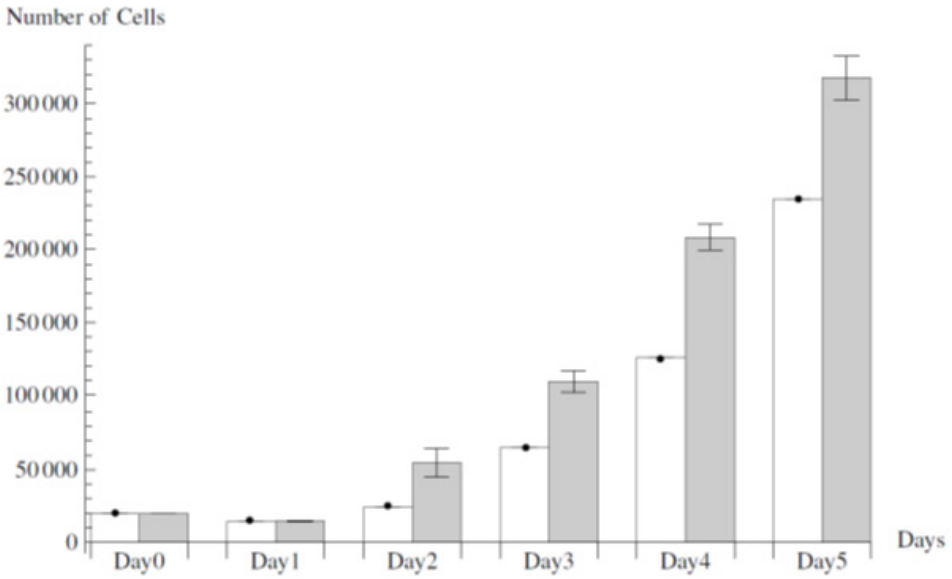


Figure 55: Population growth of simulation (white bars) and experiment (grey-scale bars). The number of cells in simulation increases slower than experiment.

Thus, I have $r_d = \frac{1 - r_{Grow}}{2}$. Because the growth rate is a population level behaviour, while the death rate is single-agent behaviour, this equation connects the two. However, the initial simulation shows that the number of agents increases more slowly than in the experimental data (Figure 55).

A possible explanation is that, in the experiment, not all the cells are in the exact same stage in the cell cycle. However, by assuming the global growth rate, any differences in the cell cycle of each cell are ignored. Then in simulation I let the agents start with random cell cycle. In this way the growth rate in simulation is slightly smaller than experiment. So that by using this method I introduce inaccuracy. To account for this, I have to avoid using doubling time. A new method is introduced in section 5.5 to calibrate the model with experimental data in a better way.

Experiment Design

Experimental set up: Experiments are conducted using cell lines: cells grown for experimental use which have well-understood properties, such as the activation of particular oncogenes, or the ability to form structures such as spheroids. In the lab, growth experiments may be conducted on a Petri dish, in which case cells can grow into a structure (see Figure 56 for example). Our biological experts helped to choose HCT116 cell line, which is sensitive to drugs and easy to form structure on petri-dish or spheroid in gel.

I am primarily interested in the shape of structures formed by tumour cells, and how drug and environmental perturbations affect this structure. To clearly show the effect of environment elements, I have one experiment with no additional treatments to serve as a control, so that results from later experiments can be compared with results from the control experiment. I choose the therapeutic agent 5FU [101] to seek to understand how the structure is affected by drug. Note that the structure is linked with cell interaction, so by comparing normal structure and 5FU structure I can analyse how this drug affects the interaction among cells. Similarly, I want to see how a hypoxic environment affects the structure by comparing normal structure and hypoxic structure. And finally, I will grow cells with 5FU in a hypoxic environment (Figure 56).

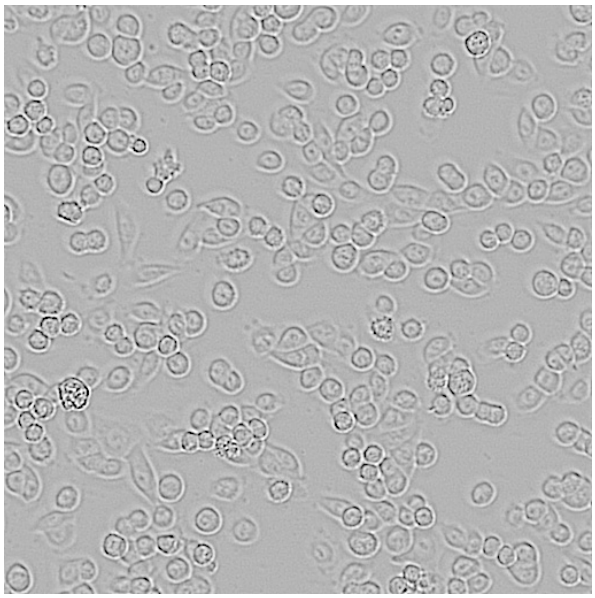


Figure 56: Cells growing on a glass plate (graph is taken from experiment carried out for Section 6).

In addition, I wish to explore the differences in behaviour of different kinds of cancer cells. Our biological experts determined to use a normal cancer cell, or wild type cell; and a mutated cell line with p53 mutation which leads to a known degree of resistance to the drug and hypoxic environment. As discussed in previous paragraph, for each cell type, four experiments will be carried out, which are one control experiment, one with drug (5-FU), one in hypoxic environment, and one with drug in hypoxic environment. Two cell lines with four experiments each, thus there are eight experiments in total:

No.	Cell Line (HCT116)		Environment		
	p53 wild type	p53 mutated	No treatment	5-FU	Hypoxia
1	√		√		
2	√			√	
3	√				√
4	√			√	√
5		√	√		
6		√		√	
7		√			√
8		√		√	√

This set of eight experiments were carried out by our biological experts and the result was analysed and used to calibrate the new model, see Section 5.2 and Section 6 for the analysis.

Data to be collected: The experiments were designed, in collaboration with cell biologists, to ensure sufficient data to allow calibration of the physical parameters. Some parameters, such as the size of cell, can be directly obtained by measurement of a certain cell line. Some parameters, on the other hand, need to be measured during the course of the experiment or processed after the experiment. The method of measurement depends on the type of data and its characteristics.

First of all, there is purely physical-related data. These data are used to calibrate the parameters of the physical model. I need data about cell size, shape and density to give these values to model cells, which as mentioned above, can be measured during experiments. I next need to know how many cells to put onto the substrate of how much area, so that the size of petri dish is also necessary. I also need length of experiment in order to control the simulation to run for a correct amount of time continually. The density and dynamic viscosity of fluid is also required, as they are related to value of constants to calculate resistance force and torque when cells are moving in experimental media.

Second, I need to add in the cell cycle model (cell growth and division etc.) and combine it with physical model, so I need data to calibrate growth-related parameters as well. I need the length of typical cell cycle of the cell line chosen, which corresponds to single cell growth curve, including a profile showing how cell size and shape can change as the cell cycle progresses. This can be determined by observing the cells over the course of the experiments. This information will need to be obtained by time-series imaging under the experimental conditions I wish to simulate. I also need the growth rate for the population per experiment, which requires me to keep a record of change of number of cells over time. With the population growth rate, I can calibrate the population growth curve. The growth rate is important because it controls the number of cells, which is essential for the correct mapping of cell distribution.

Finally, to analyse the cell interactions, I need to measure the position of the cells over time during each experiment. To get the cell position I need to take photographs of the cells over time, which will be processed by image analysis software. In this way I can obtain both the number of cells and position of each cell at the same time.

Methods to collect data during experiments: From the previous section it is known that the only data I can obtain during experiments are the change in number of cells over time, and the change in cell position over time. The experiments should contain several cell cycles so that the cell division and death can be observed. The number of cells should be counted periodically, because it is not precise enough for the model to just match the starting and ending point. Note that I need a non-destructive method to obtain this value; otherwise, the experiment cannot be continued. Similarly, I need to image the growing cells at regular intervals as well: these photographs may need to be processed before providing improved image sets. As mentioned before, I will use image analysis software to extract information from the images to obtain the position and number of cells.

I will then have directly compatible data from both experiments and simulation that can be analysed using a consistent approach. And although the structure grown in experiment and simulation may have a different appearance, I consider them match if their spatial distribution curve (as described in Section 6) is similar.

Experiment and Result

The experiment is carried out following the design in Section 5.2 by our cancer cell biology expert. There are two stages to the experiment.

In stage 1, both cell lines, HCT116 p53wt and p53mut, are used to determine the cytotoxicity of 5-FU (5-Fluorouracil) and hypoxia environment. In both stages, the hypoxia environment is mimicked by Cobalt chloride (CoCl2) and the cytotoxicity is represented by the death rate according to concentration of CoCl2. In stage 2, the 8 growth experiments are carried with the proper concentration of drugs determined in the first stage.

Stage 1: The aim of stage 1 is to determine the proper concentration of the drugs by comparing the death rate of cell with various concentrations of 5-FU and CoCl2. The concentration cannot be too high or too low, with which the death rate of cell is too high or too low respectively. In stage 1 20,000 cells were seeded in every well of a 96-well tissue culture plate and left to grow for 18 hours. Then the media was changed and a series of concentrations of drugs were added. After 24 hours, the cells were subjected to Neutral red uptake-based cell cytotoxicity assay to determine percentage cell death. The death rate of both cell lines with various concentrations is shown in Figure 57 and Figure 58 respectively. The following two figures are produced by our cancer cell biology expert, Dr Hilal Khalil.

From Figure 57, 'UT' means untreated, with which no 5-FU is added, thus the survival rate of both cell lines is 100%. Apart from the UT condition, there are 10 measurements for both cell lines. As the concentration of 5-FU increases, the survival of cell decreases. However, the measurements with 2.6, 3.9 5.8 and 8.7 micromolar show that the survival rate of p31 mutant cell line does not differ very much from UT. The survival rates of two cell lines are too similar with concentration of 13.1, 29.5, 44.4 and 66.6 micromolar, which cannot show the difference of the two cell lines. The measurements with concentration of 100.0 micromolar show a significant fall comparing with UT condition, and survival rates of two cell lines vary more than 10 10%. However the measurement of p53 mutant cell line contains a bigger error bar, which suggests bigger difference in the data of three countings. Therefore, concentration of 19.7 micromolar is the best choice. In stage two, 20.0 micromolar is used.

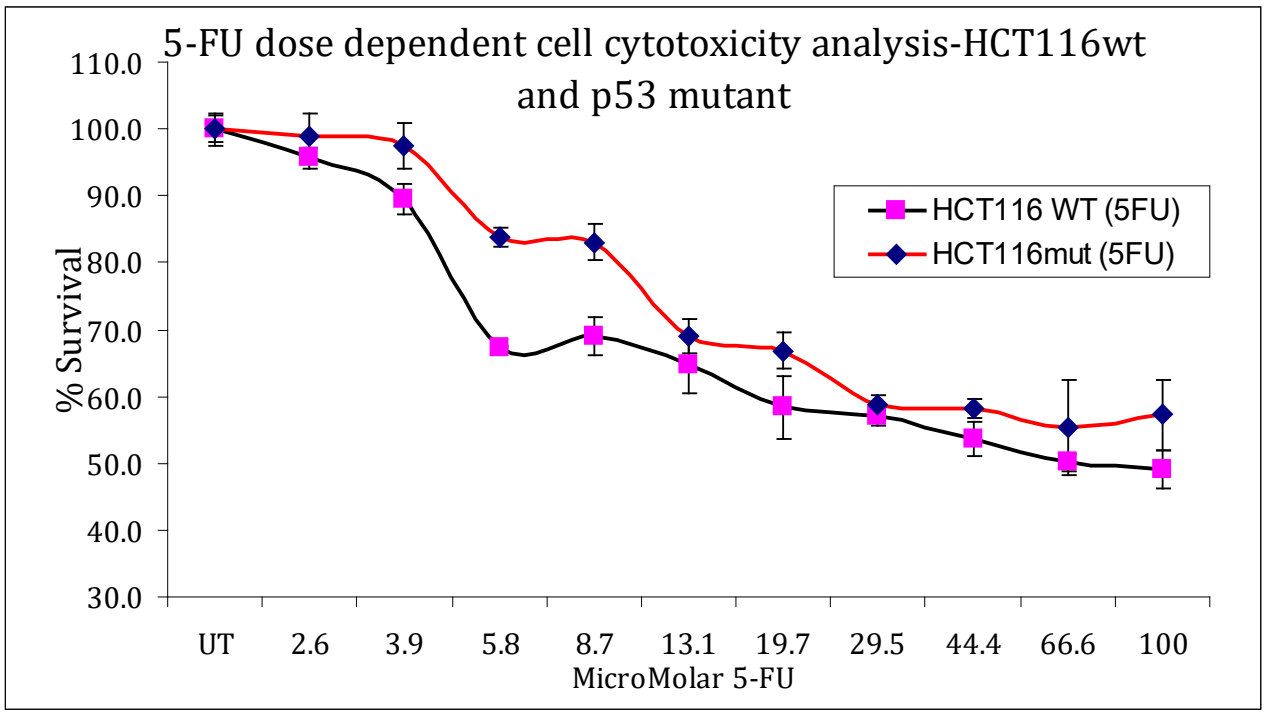


Figure 57: Survival rate of HCT116wt (black line with red dots) and p53 mutant (red line with black dots) cell lines under effect of various concentration of 5-FU. The x-axis is the concentration of 5-FU measured by MicroMolar; and the y-axis is the survival rate of cell, which is measured by %.

In Figure 58, there are also 11 measurements for both cell lines, including the UT condition (the survival with 0.0 MicroMolar). With concentrations of 25.8, 38.7, 58.1, 87.3, 131.0 and 196.7, the survival rate of p53 mutant cell line does not show significant fall compared with untreated data. With concentration of 295.4 micromolar the survival rate of both cell lines is similar, which cannot show the difference of the two cell lines against hypoxic environment. For the concentration of 1000 micromolar, the survival rate of p53 wild type (p53 wt) cell line is too low. Thus, the concentration of 443.6 and 666 micromolar can provide better results. In stage two, 500 micromolar is used.

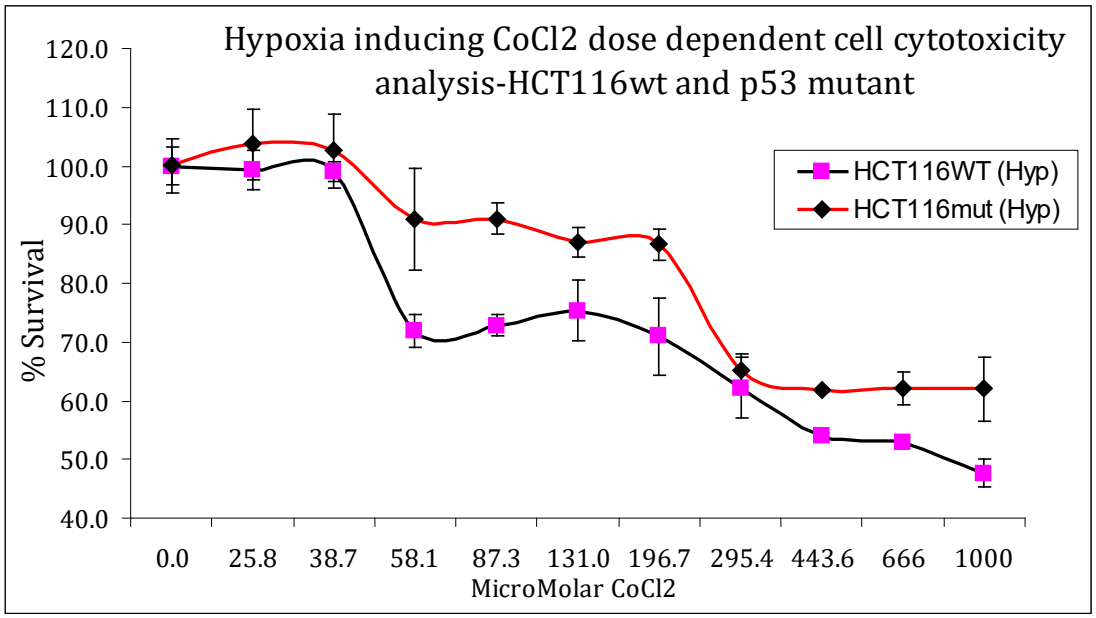


Figure 58: Survival rate of HCT116wt (black line with red dots) and p53 mutant cell lines (red line with black dots) under effect of various concentration of hypoxia drug. . The x-axis is the concentration of 5-FU measured by MicroMolar; and the y-axis is the survival rate of cell, which is measured by %.

In addition, to show the survival rate of both cell lines with combination of 5-FU and hypoxia environment, both 20.0 micromolar of 5-FU and 500 micromolar of CoCl2 are added together.

Stage 2: Stage 2 is to study the effect of drugs on growth rates. From stage 1, it is determined that the concentration of 5-FU and CoCl2 (for hypoxia) is as follows:

- 20microMolar 5-FU
- 500microMolar CoCl2 (Hypoxia)
- A combination of 20microMolar 5-FU + 500 Micromolar CoCl2

In stage 2, both cell lines are used with the concentration of drugs listed above. Exactly 20,000 cells were seeded in each well in 96-well plate. This was considered day '0'. During the next 5 days, cells in 3 out of 96 wells were sampled and counted each day. By plotting the cell population of each day, I build the population growth curve of both cell lines, which are shown in Tables 4&5. Note that 3 wells were counted for each day, all the number of cells in Tables 4&5 are average value.

Table 4: The experimental population growth curves of HCT116 p53wt.

Hrs	DAYS	HCT 116 wt							
		UT	SD	5-FU	SD	Hyp	SD	5-FU/Hyp	
0	0	20000	100	20000	100	20000	100	20000	100
24	1	14737	200	8421	546	13684	500	8421	415
48	2	54737	10000	24000	1125	11579	564	9474	564
72	3	109474	7400	63158	1456	23158	1087	8421	1087
96	4	208421	8900	77895	1147	13684	1040	5263	569
120	5	317895	15020	89474	2067	10526	789	3158	487

Table 5: The experimental population growth curves of HCT116 p53mut.

Hrs	DAYS	HCT116 mut							
		UT	SD	5-FU	SD	Hyp	SD	5-FU/Hyp	SD
0	0	20000	1850	20000	10500	20000	1050	20000	1050
24	1	17500	9090	15000	18000	20000	1200	12000	1213
48	2	55000	16054	30000	10405	27500	1547	12500	604
72	3	202500	28014	80000	2809	22500	147	12500	478
96	4	477500	25032	192500	25014	20000	287	15000	1009
120	5	812500	37000	195000	29874	20000	201	10000	743

The time-lapse images are taken on the same well every 10 minutes for 18 hours. The sample photograph is shown in Figure 59. The size of each time-lapse image is pixels, and by comparing the relative size of whole image and each cell I can therefore estimate that the diameter of cell is about 10 pixels. From the series of photographs, it is also able to estimate the length of a typical cell cycle.

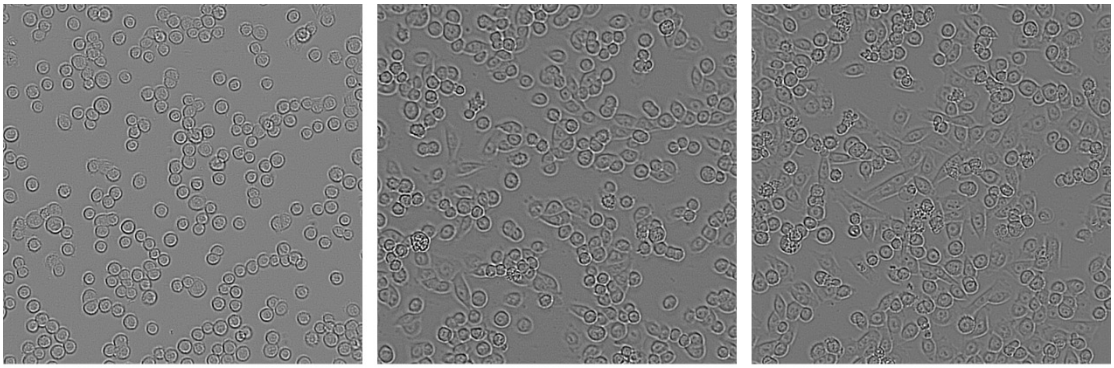


Figure 59: Time lapse images taken from growth experiment of HCT116 p53mut cell line with effect of 5FU: the left image is the start point of experiment; the middle and right images are taken of the same area with interval of 10 hours.

Model Parameterization

In the simulation, there are some other parameters that relate to experimental data. These parameters are verified as follows.

Cell size and shape: Compared to endothelial cells, the cell line used for the cancer cell interaction study has a different size and shape. Due to the importance of cell size and shape to the physical interaction, I need to modify the relevant values in the model.

The shape and size of cell can be estimated from experiment photographs using to represent the sphere radius, so that its volume is

$$V_0 = \frac{4}{3}\pi r^3 .$$

When the cell is attached to the petri dish, its shape and size may vary. I need to find a typical size and fit it to an ellipsoidal shape. Then I can use it as agent shape/size in simulation. To define the shape of an ellipsoid, I need to know the length of its three semi-axes (Figure 60). By measuring the cells in experiment photographs I can estimate the length of two semi-axis which are horizontal to the petri dish, and with the volume of un-attached agent I can calculate the last semi-axis by ellipsoid volume equation $V_e = \frac{4}{3}\pi a_1 a_2 a_3$, where a_1, a_2 and a_3 are semi-axes of ellipsoid and $a_1 < a_2 < a_3$ (in the model I have the rule that $a_1 < a_2 < a_3$, see section 3.1 for detail).

The length of three semi-axes is determined by estimating their ratio while keeping the volume of agent as $\frac{4}{3}\pi r^3$.

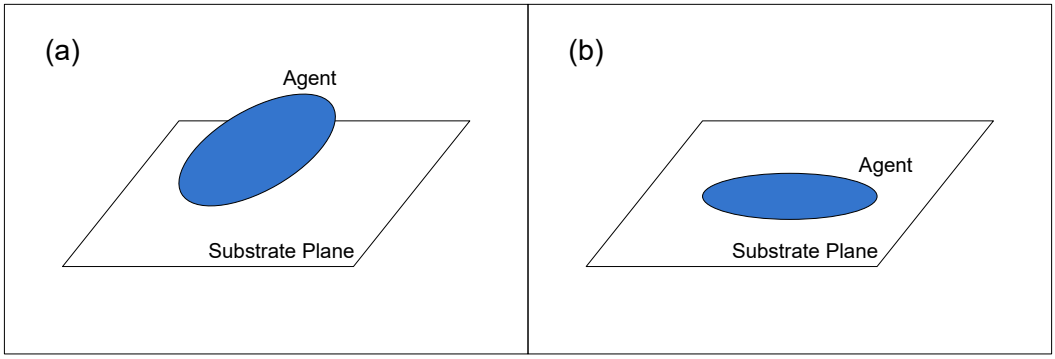


Figure 60: (a) Un-attached agent; (b) Attached agent becomes flatter.

In the experiment I can observe that cells attach to the bottom of petri-dish and the shape is generally flat. The semi-axis that is vertical to the petri dish should be much shorter than horizontal semi-axes. In addition, to afford the potential polarity of agents the two semi-axes parallel to the petri dish plane should not be equal to each other. However, from time-lapse images it is known that the ratio of these two semi-axes should not be too large. A value between 1:1 and 1:2 may be appropriate.

I let semi-axis $a_1 = \frac{1}{4}r$, $a_2 = \frac{\sqrt{2}}{2}r$, $a_3 = 2\sqrt{2}r$;

so that the volume of agent remains $\frac{4}{3}\pi r^3$.

From the photographs of experiments, I can determine the diameter of a typical HCT-116 cell is 10 μm immediately after division. Thus $r = 5\mu m$. Since the unit length $\Delta x = 1.25\mu m$, measured by the unit length, $a_1 = \Delta x$, $a_2 = 2\sqrt{2}\Delta x$, $a_3 = 8\sqrt{2}\Delta x$.

Cell age: By observing the time-lapse images, I determined that cells in the experiment divide after 12 hours. During this time the volume of the cell is doubled. Therefore, I take the value of 12 hours as the typical cell cycle in experiment. In the real experiment, the growth of cells may vary, so that I set 12 hours as the expected value and add 10% variance. Also, in a real experiment each cell may start growing from a different position in its cell cycle. To simulate this, I let each simulation agent start from a random position for its age as well. Figure 61 shows the sample growth curves with and without both the cell age variance and random agent starting up position, in which the curve with cell age variance and random agent starting up position is smooth.

I let simulated agents complete one modelled lifespan in 100-time steps. Thus 100-time steps equal 12 hours in the real experiment. As in the experiment the total number of cells is counted every 24 hours, and the total number of simulation agents is extracted every 200-time steps. Note that I output to a file the number of cell data every 10-time steps, the interval of measurement in experiment equals same simulation time to produce 20 outputs. Therefore, if curves from the experiment and simulation are drawn on same coordinate system the data point for 'Day 1' of experiment should be plotted on x-coord=20.

A setting file stores all the growth information for each cell line which is read and sent to agents at simulation start up. The following 9 settings form a typical set of information for one simulation:

(1) celltype = 1

There are several sets of built-in shape of ellipsoid in the simulation, stored in a two-dimension array. The value of 'celltype' is used to access the array to find the corresponding size of semi-axis of cell shape.

(2) numberofphase = 5

Because there are six measurements of total number of cells in our experiments, and it is assumed that in each interval the growth rate of cell is constant. There are five intervals in six measurements, thus each interval is considered as a phase of growing. This setting works together

with setting (5)-(9). The value of 'numberofphase' is read then the code knows how many phases to read, so that it can prepare space to contain growth rate of each phase.

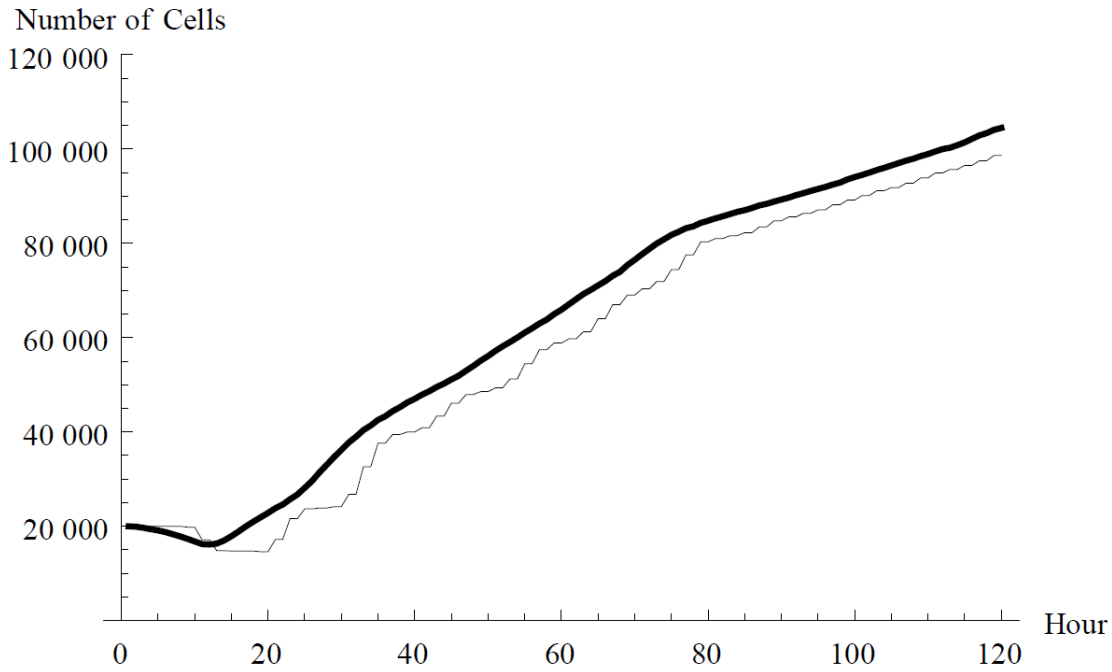


Figure 61: Stepped curve (thin line) and smooth curve (thick line) (graph is drawn with dynamic growth rate r_{Grow}. Note that the smooth curve has higher value of number of cells, because with random agent starting up position in cell cycle, some agents divide earlier and enters high growth rate period earlier; thus, more agents are produced).

(3) MaxAge=100

The "MaxAge" is the expected value of typical cell cycle length. As previously it is assumed that 100-time steps in simulation equal 12 hours in the real experiment, here the value 100 is assigned to typical cell cycle length.

(4) AgeVariance=10

In the previous section, the length of cell cycle is assumed to obey normal distribution. The "AgeVariance" is the variance of the distribution. Using 'MaxAge' and 'AgeVariance', each agent in simulation can have a different length of cell cycle.

(5) 20, 0.42920 (6) 40, 0.96362 (7) 60, 0.70711 (8) 80, 0.68990 (9) 100, 0.61751

Setting (5)-(9) are working together with setting (2) to control the probability of cell division in the simulation (the definition of probability of cell division is at the beginning of Section 5.5). The setting (2) regulates how many phases are in simulation, and settings (5)-(9) regulate the value of probability of cell division in each phase. Each one of these five settings contains two parts: the first part is the end point of current phase, and the second part is the probability of cell division in the current phase.

Take the setting (5) as an example. The value '0.42920' means the value of probability of cell division should be 0.42920; and the value '20' means this value is valid until the program produces 20 outputs.

Note that the integer part (20, 40, 60, 80, 100) has different meaning from 'MaxAge' and 'AgeVariance'. The integer value for 'MaxAge' and 'AgeVariance' means the number of time steps, while the integer value for probability of cell division means the number of output file. As there is one output file every 10-time steps, the integer 20 here actually means 20x10=200-time steps, which equals 24 hours, which fits length of interval between day0 and day1 in the experiment.

When the program produces 20 outputs, the setting (5) is useless. Thus, from that time point (200th time step, or 24th hour) the setting (6) takes effect. Similarly, the setting (6) is valid until there are 40 output files, in other words the setting (6) is valid between the 24th hour to 48th hour.

In this way these five settings define the value of probability of cell division for the simulation using data from five intervals of the *in vitro* experiments. The method to determine the value of probability of cell division from experiment is discussed in section 5.6.

In the same manner of setting (1)-(9), several sets of data can be defined to represent a unique cell type, and by assigning different cell type to agents, agents with different behaviour can be added in one single simulation.

Time frame: In the *in vitro* experiment, the time-lapse images are taken every 10 minutes. The time frame of model is modified to this value.

Dynamic viscosity of media: According to the documentation provided by the media manufacturer (http://www.lifetechnologies.com/uk/en/home/life-science/cell-culture/mammalian-cell-culture/classical-media/dmem.html), the media can be physically considered as water so that the value of dynamic viscosity does not need to be modified.

Population growth curve fitting

From the experiment data (Table 4) the population growth rate during each interval can be calculated. Note that the population of cell is counted every 24 hours in the experiment and, because of a lack of data between each measurement point; I assume the population growth rate is constant throughout each interval, although the growth rate during each interval may of course vary. The population growth rates for the four experiments are shown in Table 6.

Table 6: Population growth rate of 4 experiments with HCT wild type cell line.

HCT-116wt	UT	5-FU	Hypoxia	5-FU + Hypoxia
0-24hr	-0.305	-0.485	-0.38	-0.865
24hr-48hr	1.312	1.047	-0.167	0.118
48hr-72hr	0.693	0.968	0.693	-0.118
72hr -96hr	0.644	0.21	-0.526	-0.47
96hr -120hr	0.422	0.139	-0.262	-0.511

The population growth rate cannot be directly used in the simulation, because I use an agent-based model and what is really needed is the chance that the agent divides at the end of its cell cycle.

I introduce a new term, "probability of cell division", to the simulation to control the chance that agent successfully divides. The probability of cell division is a real number between 0.0 and 1.0, which represents the chance that one single agent divides at the end of its cycle: the maximum value means the agent will certainly divide; the minimum value means the agent will certainly die. The higher the value, the more likely it is that the agent will divide. For a group of agents, since they share the same population growth rate value, they also share the same value for the probability of cell division. From the definition of probability of cell division, it is readily seen that if its value is higher than 0.5, the population of agents increases. If the value of the probability of cell division reaches 1.0, the population of agents satisfies exponential increase, which is described in section 5.2; if the value of the probability of cell division reaches 0.0, all the agents die.

Note that the probability of cell division has a different meaning to population growth rate: the probability of cell division takes effect when cell goes to the end of the cell cycle and so affects each single cell cycle; in contrast the population growth rate is a measurement from experiment and can be the result of several cell cycles. The value derived from the experiment is the population growth rate, but the value to be used in the model is the probability of cell division. The probability of cell division is a real number between 0.0 and 1.0, while the population growth rate does not have limit, especially for one single cell cycle, the population growth rate is in the range $[-1,1]$. Thus, a conversion is required for the population growth rate and probability of cell division during single cell cycle.

Let r_{Grow} equal the population growth rate, p_{Div} equals the probability of cell division. As discussed in section 5.2, the population growth rate is calculated without using doubling time. It is known that the length of a typical cell cycle is 12 hours. Therefore, in each 24 hour-interval

there are 2 cell cycles. As discussed previously, I start with N_0 cells; let x represent the percentage of cells that divide, obviously after one cell cycle there will be $2x \cdot N_0$ cells left, because $x \cdot N_0$ cells divide to $2x \cdot N_0$ cells and the rest of cells die. After one more cell cycle there will be $4x^2 \cdot N_0$ cells left. Using N_1 to represent the number of cells after 24 hours (2 cell cycles), then the equation can be written as follows

$$N_0 \cdot 4x^2 = N_1 \tag{183}$$

By solving it I have

$$x = \sqrt{\frac{N_1}{4 \cdot N_0}} \tag{184}$$

Therefore, for each interval, with both measurements of number of cells before and after the interval I can calculate the percentage of cells that divide to two daughter cells, which I use on each agent as its chance to divide.

Also consider N_1 as N_t in the first equation, the probability of cell division p_{Div} satisfies $N_0 \cdot 4x^2 = N_t$, substitute it to the equation (161), I have

$$r_{Grow} = \frac{2}{t} \ln 2 p_{Div} \tag{185}$$

In which r_{Grow} is the population growth rate, p_{Div} is the probability of cell division for each agent, and *t* equals 24 hours. In this way I build the connection between the behaviour of each agent and the behaviour of the population. However, this connection is based on the fact that the measurement interval in the experimental result contains 2 full cell cycles, and thus is experimentally specified. For any other experiments with different cell lines and measurement intervals the connection should vary. The population growth rate r_{Grow} and probability of cell division p_{Div} (according to the equation (164)) of control population are presented in Table 7. Note that each interval of measurement in experiment contains two cell cycles, thus the population growth rate is not limited to $[-1,1]$.

Table 7: Population growth rate and probability of cell division of control group experiment.

	0-24hr	24hr-48hr	48hr-72hr	72hr -96hr	96hr -120hr
Population growth rate r_{Grow}	-0.305	1.312	0.693	0.644	0.422
probability of cell division p_{Div}	0.4292	0.96362	0.70711	0.6899	0.61751

In the simulation each agent is assigned the same probability of cell division p_{Div}. From the definition of probability of cell division p_{Div}, it is obvious that if p_{Div} is higher than 0.5, the total number of cell increases; if p_{Div} is lower than 0.5 the total number of cell decreases.

The next step is to map the growth curve of each experiment to the simulation. I map the control population first, using the probability of cell division calculated from experiment data. Then instead of using experiment data to map drug and hypoxia affected growth, I analyse the effect of these factors on the cell and introduce the mechanism into the model that changes the model behaviour in order to reach the observed population growth curve. Finally, I then test my mechanism assumptions against independent data, specifically the population growth curve for the combined drug and hypoxia condition.

Model calibration with Control population: I apply the probability of cell division in Table 7 to the simulation and run for 10 times, simulation result is shown in Table 8.

Table 8: Cell number in 10 simulations of control group.

Simulation No.	Number of Cells					
	0	24 hr	48 hr	72 hr	96 hr	120 hr
1	20001	29831	79705	149109	254143	374142
2	20002	29728	79430	146807	249908	368676
3	20003	29650	79155	147719	251822	371681
4	20002	29990	80324	150097	255907	377912
5	20003	29686	79657	147657	250946	370312
6	20002	29882	80090	148767	253311	373235
7	20003	29821	79729	147904	251395	370854
8	20001	29627	79426	147428	251350	370295
9	20003	29850	80368	148933	252525	373784
10	20001	29804	79564	147848	250390	369295
Average	20002.1	29786.9	79744.8	148226.9	252169.7	372018.6
SD	0.8756	113.3504	399.1438	974.7133	1839.781	2784.516

I draw the data as bar charts with errors bar together with experiment data. As Figure 62 shows, before calibration the simulation data does not match the experimental result.

Figure 63 shows an improved result following calibration. Compared to experiment data, the simulation result is always slightly higher. The reason for that is, in the calculation I assume all the agents have exact same cell cycle and start from time 0 but in the model, agents have variation of 10% of their cell cycles and in simulation they start from random position in cell cycle. Thus, some of the agents process to division stage earlier than theoretical situation, which lead to more daughter cells. However, the simulation result is sufficiently close to the experimental data and can be accepted as baseline of other simulations.

5-FU and the population growth curve: As discussed in Section 5.1, 5-FU disrupts DNA synthesis in tumour cells and therefore arrests tumour cells in DNA synthesis stage in cell cycle [101]. Thus, tumour cells cannot complete the cell cycle and finally die. Thus, the mechanism of 5-FU can be considered as a decrease of cell proliferation. In my model, the decrease of cell proliferation is presented by reduction of chance of cell division. Three strategies have been considered in modelling how 5-FU affected cell growth. The first is to calculate the probability of cell division from the experimental data of the 5-FU affected group, then use the value directly, as per the control population growth curve. The first strategy does not reflect mechanism of 5-FU and theoretically in this way I can map any population growth curve to model. The shortcoming is that it does not help to understand what is happening in terms of the cell responses to drug action. The second strategy is to amend the probability of cell division derived for the control condition in a way that reflects the 5-FU mechanism such that this probability works with the parameter set to fit to the control population growth curve to generate the simulation growth curve for the 5-FU condition. The third strategy is to let the probability of cell division change dynamically with the concentration of 5-FU. This strategy helps to uncover the connection between cell growth and 5-FU concentration, but it cannot help to show the difference between the controlled condition and the 5-FU condition. Therefore, I choose the second strategy for all the following simulations, in which I will try to introduce a method to amend the control data to reflect the effect of drug (and then hypoxia).

To simulate the effect of 5-FU, a new state is added: Arrested, to the state machine previously shown in Figure 51. The arrow pointing from the Growing state to the Arrested state means that an agent can be affected by 5-FU mechanism while it is growing. Once an agent is in the Arrested state it cannot turn back to the Growing state or Idle state, and that is why this new state is called 'Arrested'. An agent dies when it gets to its cell age in the Arrested state. Thus, the effect of 5-FU can be considered as an extra rate of death.

In the simulation, instead of changing the probability of cell division directly, those cells that decide to divide are given a chance to enter the Arrested state in the fate-determination phase. All the arrested cells will not divide; instead, they die after a certain time. In this way the Arrested state works as an extra chance of death (Figure 64).

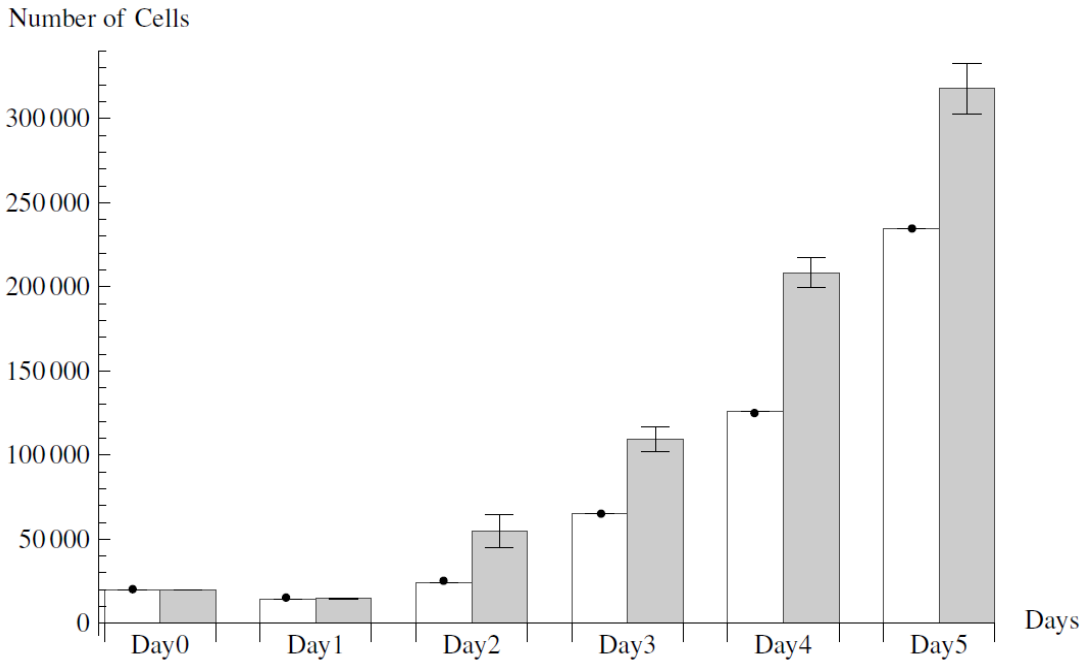


Figure 62: Control population growth curve without calibration, the white bars are simulation data, the grey bars are experimental data. The simulation ran once, so there is no error bar. I can see that the simulation population increases more slowly than the experimental population.

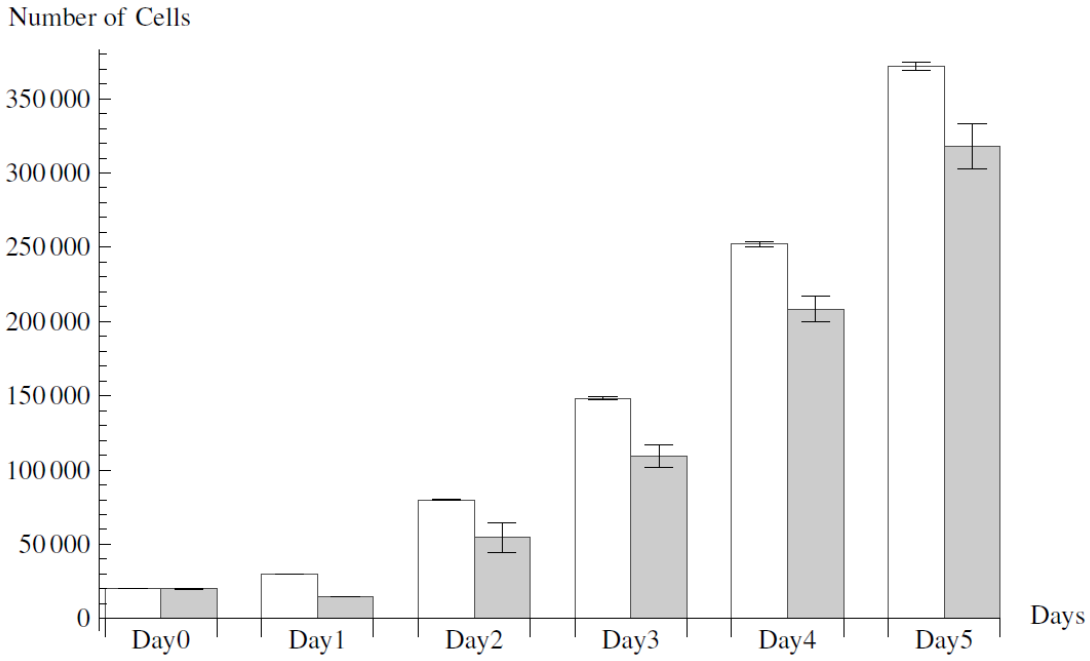


Figure 63: Control population growth curve with calibration. The white bars are average value of 10 simulations, the grey bars are experimental data. I can see that although the simulations do not match with experimental data; by adding variation to cell cycles and letting cells start from random position in cell cycle, the simulation result is closer with experimental data.

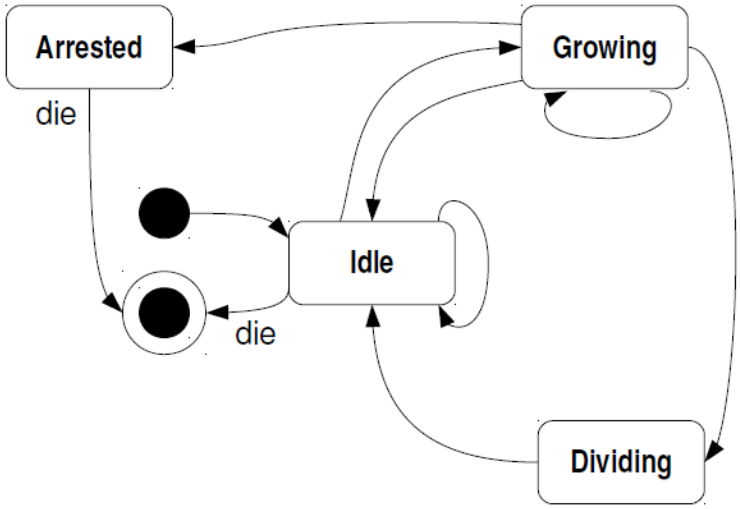


Figure 64: State of agent as state machine with mechanism of 5-FU modelled as 'Arrested' state.

In this way there are several values to determine: how much a cell has already grown before it is arrested; the percentage of cells that go into 5-FU status. In these variables, the percentage may change over time, as it depends on concentration of 5-FU which is changing over time as well, but for simplicity I consider a constant arrest rate at the moment.

If the arresting chance is too high, the total number of cells will decrease, and if it is too low the growth curve will look like the controlled growth curve. After trying several values, I get the best simulation result when it is set to 5%, as shown in Figure 65.

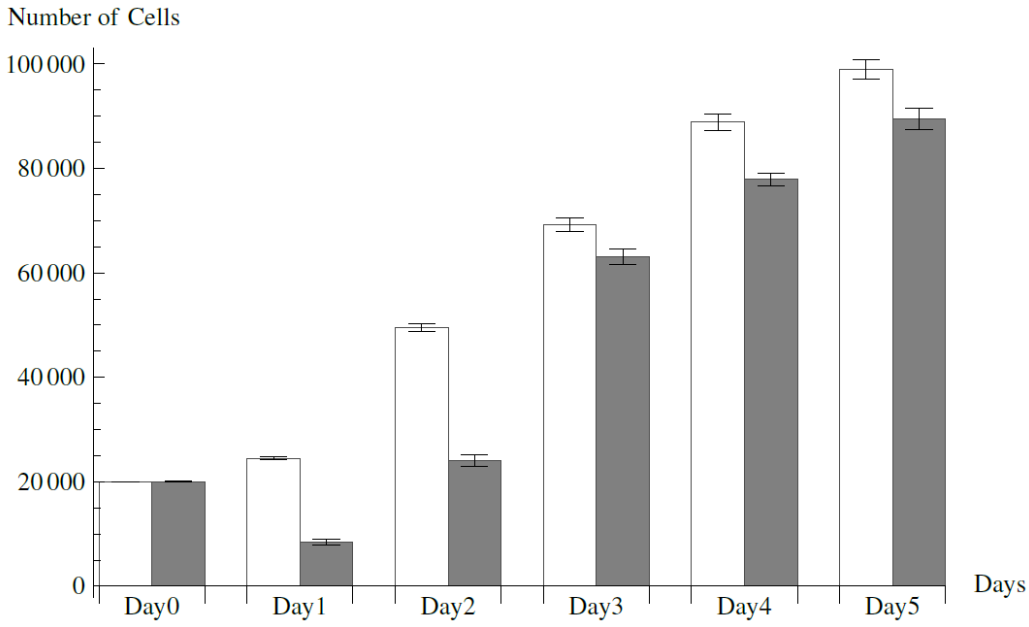


Figure 65: Simulation with 5% arrested agents compared with experimental result; the simulation bars are in white, and the experiment bars are in grey. The value of simulation data is average of 10 runs, and the experimental data is from 3 counts. The simulation and experimental data are distanced in Day1 and Day2 but become closer in Day 3, 4 and 5. Although the simulation and experimental data do not exactly match, they are efficient to show the added mechanism of 5-FU to the model.

Fitting to Hypoxia population: In a similar manner to the approach of mapping the effect of 5-FU on cell growth, I first analyse the biological effect of hypoxia on cell growth and then introduce a plausible mechanism to the model. Note that as I discussed in section 5.5, my cell cycle model is a simplified version of the underlying biology. In my model I ignore the inter-cell biological reaction and focus on cell behaviours that can be observed from outside, such as change of cell volume, cell division and cell apoptosis.

Then to introduce the mechanism of hypoxia, I first analyse its effect on cell growth, including whether it affects the length of cell cycle and probability of cell division. The first strategy used is to introduce an upper limit number of cells that may survive in a single petri-dish. Considering the petri-dish as a closed system, the oxygen and nutrient supply is limited, so that the cells planted in petri-dish cannot grow forever. This upper limit of number of cells is referred to as saturation density. Not only is the total number of cells in a single petri-dish affected by the saturation density; the rate that the number of cells change over time, i.e. population growth rate, is affected at the same time, as described in section 5.2.

The problem is that the effect of saturation density K is only significant when number of cells approaches K, while in the experiment the hypoxia acts on cells at any density and thus has a uniform effect during whole experimental process. In other words, the saturation density K has a different biological mechanism of hypoxia and is not a suitable representation to be introduced in simulation.

As described in Section 5.1, on cell level, hypoxia causes a slowed cell cycle; on tissue level, hypoxia reduces the growth of tumour [88]. Thus, to introduce the effect of hypoxia, I need to reduce the probability of cell division, which is the second strategy. I assume that cells are affected equally during the whole experiment; therefore, I have to amend the probability of cell division for all 5 intervals in the same way. As mentioned at the beginning of section 5.5, an increase in the cell population requires the probability of cell division to be higher than 0.5, and a decrease of cell population requires it to be lower than 0.5. From Figure 66 it is known that the probability of cell division from 0 to 48 hours should be lower than 0.5, then higher than 0.5 from 48 to 72 hours, and again lower than 0.5 after that, as Table 9 shows.

Table 9: Probability of cell division in experiment under control condition and method to amend the probability of cell division to fit hypoxia group.

	0-24hr	24hr-48hr	48hr-72hr	72hr -96hr	96hr -120hr
Control condition probability of cell division	0.4292	0.96362	0.70711	0.6899	0.61751
Expected hypoxia probability of cell division	<0.5	<0.5	>0.5	<0.5	<0.5
Method to amend control condition parameters	Reduce no more than 0.42920	Reduce at least 0.46362	Reduce no more than 0.20711	Reduce at least 0.18990	Reduce at least 0.11751

Table 9 also shows the method to amend the control condition parameters that may produce a well-fitting growth curve. Even without recourse to simulation it is clear from Table 9 that the requirement between 24 and 48 hours cannot be satisfied as it will make the probability of cell division between 0-24 hr. lower than zero and 48-72 hr. lower than 0.5.

For the other 4 intervals, there is a value that satisfies all requirements: the original probability of cell division for 48-72 hour is 0.70711, and to keep it more than 0.5 it cannot be reduced by more than 0.20711; for 72-96 hour and 96-120 hour the probabilities are 0.6899 and 0.61751 respectively: to decrease the cell population they should be reduced by 0.1899 at least. I choose the value 0.2 for simplicity, thus the new probability of cell division set for hypoxia affected cells is shown in Table 10, and the growth curve this set of data produces is shown in Figure 67.

Table 10: Probability of cell division in my model for hypoxic group.

Time	0-24hr	24hr-48hr	48hr-72hr	72hr -96hr	96hr -120hr
Control condition probability of cell division for simulation	0.4292	0.96362	0.70711	0.6899	0.61751
Hypoxic condition probability of cell division for simulation	0.2292	0.76362	0.50711	0.4899	0.41751

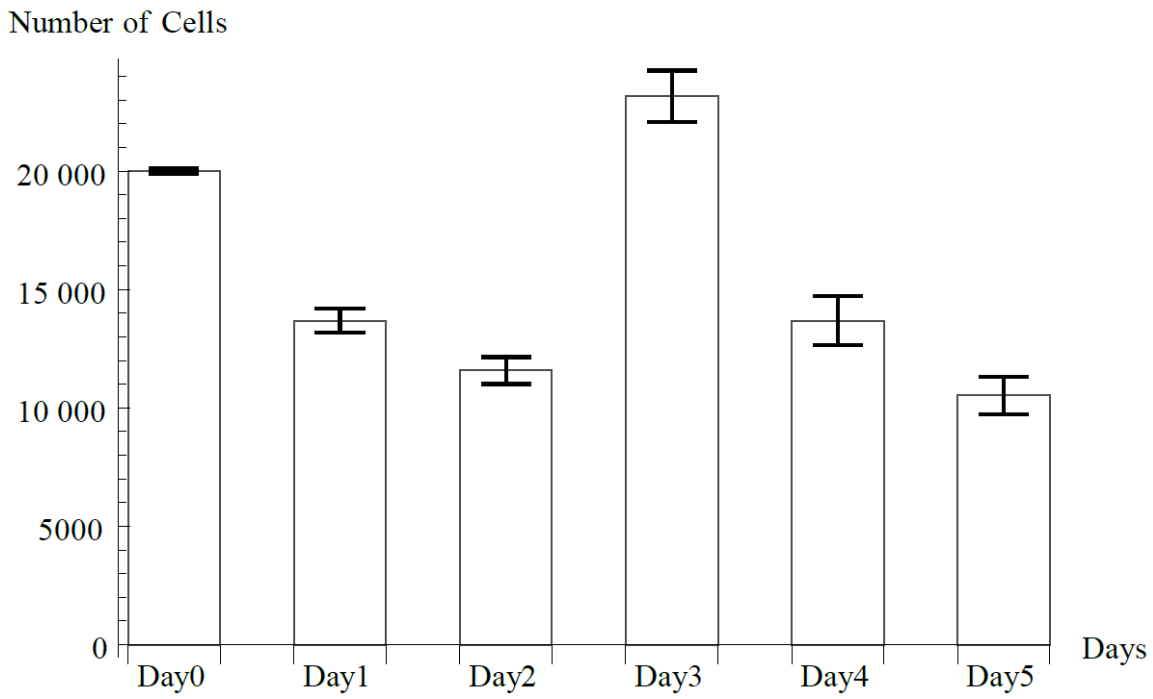


Figure 66: Growth curve of hypoxia (experimental), which is generated with 3 counts.

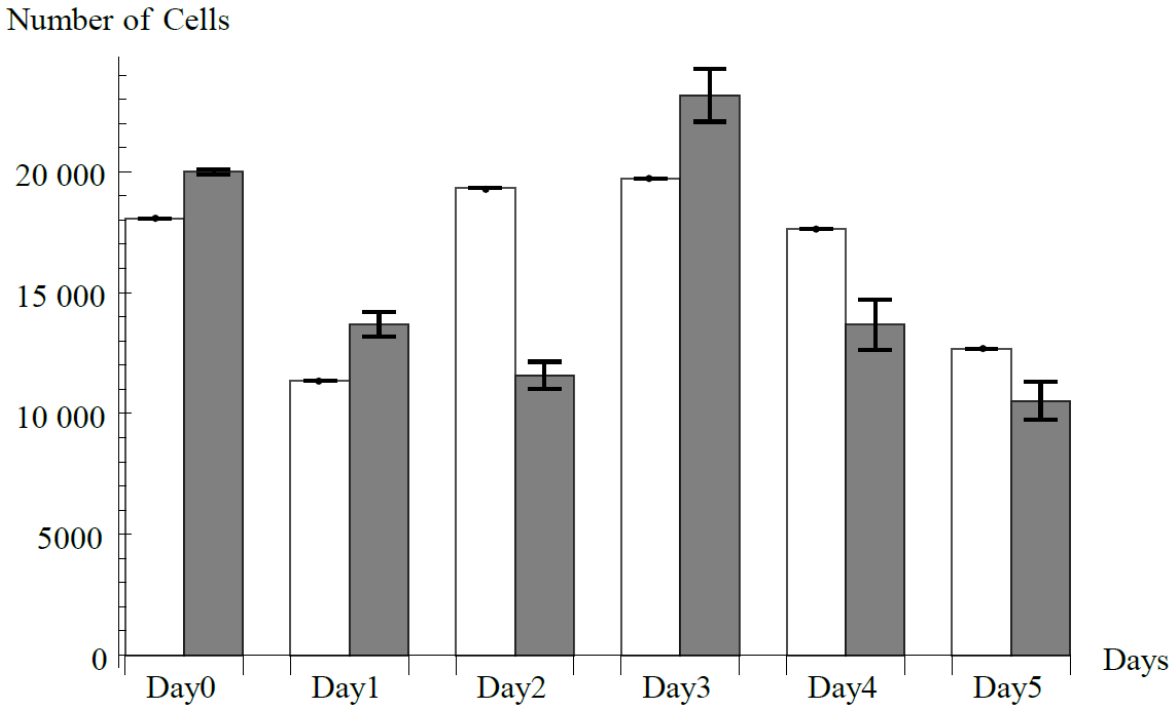


Figure 67: Hypoxia growth curve of my model and experiment result (with one type of agent). The white bars are simulation data and grey-scale bars are experimental data. As the simulation was run only once, there is not error bars for experimental data. The simulation and experimental data are very distanced on Day2.

Figure 67 shows that the growth curve from simulation and experiment are relatively close during 0-24 hr., then during 24-48 hr. the simulation produces many more agents. Thus, the simulation cell growth continues with an incorrect number of cells from 48-120 hr., so this it is not a good match either.

By observing the values in Table 10 I can predict the bad match of simulation result and experimental data, as the probability of cell division in the experiment in hypoxia group cannot be simulated by simply increasing or decreasing corresponding value in control group. In study [88] shows a slowing of tumour cell proliferation with moderate hypoxia. Thus, I decided to add a new type of agent to represent hypoxia affected cells with longer typical cell cycle time. I keep the old type of agent and let it maintain the original cycle length but further its probability of cell division, while adding a new type which has a longer cycle but keep probability of cell division in previous chart, as shown in Table 11.

The new set of growth information is written into the setting file as a new type of agent, as Table 11 shows. The 'cell type B' is the new type of agent added to simulation (the original agent is called "cell type A"). As its cell cycle length is doubled, the value of 'MaxAge' and 'AgeVariance' is also doubled, as well as the end time that each value of probability of cell division takes effect. By adjusting the proportion of these two types in the starting population, I am able to delay the time where cell population begins to increase and produce growth curves shown in Figure 68.

Table 11: Parameters relative to cell cycle and probability of cell division of two types of agents.

	Growth info for Cell Type A		Growth info for Cell Type B	
MaxAge	100		200	
AgeVariance	10		20	
numberofphase	5		5	
probability of cell division in phase 1	Phase duration from hour 0 to hour 20	0.2292	Phase duration from hour 0 to hour 40	0.4292
probability of cell division in phase 2	Phase duration from hour 20 to hour 40	0.76362	Phase duration from hour 40 to hour 80	0.96362
probability of cell division in phase 3	Phase duration from hour 40 to hour 60	0.50711	Phase duration from hour 80 to hour 120	0.70711
probability of cell division in phase 4	Phase duration from hour 60 to hour 80	0.4899	Phase duration from hour 120 to hour 160	0.6899
probability of cell division in phase 5	Phase duration from hour 80 to hour 100	0.41751	Phase duration from hour 160 to hour 200	0.61751

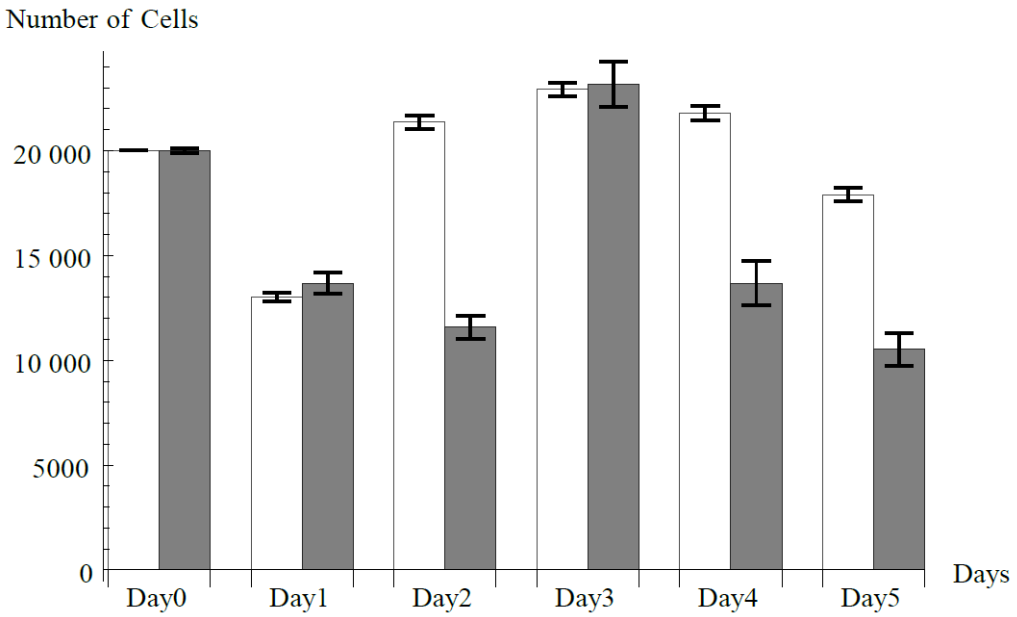


Figure 68: Hypoxia growth curve of my model and experiment result (with two types of agents) The value of simulation data is average of 10 runs, and the experimental data is from 3 counts. With two cell types, although they are still distanced on Day2, Day4 and Day5, the simulation and experimental data now match on Day1 and Day3.

Compared to Figure 67, the growth curve in Figure 68 is a better match. I can see from Figure 68 that although for most of the time points the simulated cell population level is close to the experimental results, it is still much larger than the experimental observation at the 48-hour point. It can be seen from the probability of cell division in Table 10 that for both cell types the probability of cell division is higher than 0.5, which causes the cell population to increase. However, the cell population decreases during this time in the experiment, which requires a probability of cell division lower than 0.5. As all the probability of cell division values must be altered equally under an assumption of constant drug effect, to solve the problem all the values should be decreased until the value for 24–48 hr. is lower than 0.5, which will make the probability of cell division for 0–24 hr. lower than zero and out of its possible range, and this means that this problem cannot be solved by tuning the probability of cell division. A possible solution to this fitting problem is to tune the cycle length of the new cell type, until it remains in its first two cycles between 24 and 48 hr. As the old cell type is in 3rd-4th cycle during that time, the combination of probability of cell division of both types may fit the simulation growth curve to the experimental one. This requires further work to determine the proper cycle length and proportion of the two types in the starting condition. However, the results as they stand are a sufficiently good fit.

Predicting combined 5-FU and hypoxia population: To test independently the extensions to the model included to reflect 5-FU and hypoxia mechanisms of action, I modelled and compared the simulation output with experimental data for the condition combining 5-FU and hypoxia. This means that I preserve all parameters from the previous experiments: (1) two types of cells and their cell cycle and probability of cell division setting with which I simulate the effect of hypoxia; (2) the increase in chance of cell cycle arrest chance as used in the 5-FU affected simulation. Crucially, there is no new mechanism added and no data from experiment of combination of 5-FU and hypoxia is used.

Figure 69 shows that, although the simulated and experimental data do not match, they are close enough to be considered as a good result, where with no other model adjustment I am able to predict periods of population growth and decline. I note some differences in the rates of decline and increase in the earlier stages of the simulation. According to previous simulation results, this difference is most likely driven by the hypoxia effect, because the 5-FU simulation tends to have similar dynamics over all time points, while the hypoxia simulation has higher variability during the period of most difference in Figure 68. I can see that the combined effects show a reasonably good fit to the experimental results, and I have reason to believe that, by improving the fit of the hypoxia simulation, I am able to improve the combined simulation at the same time.

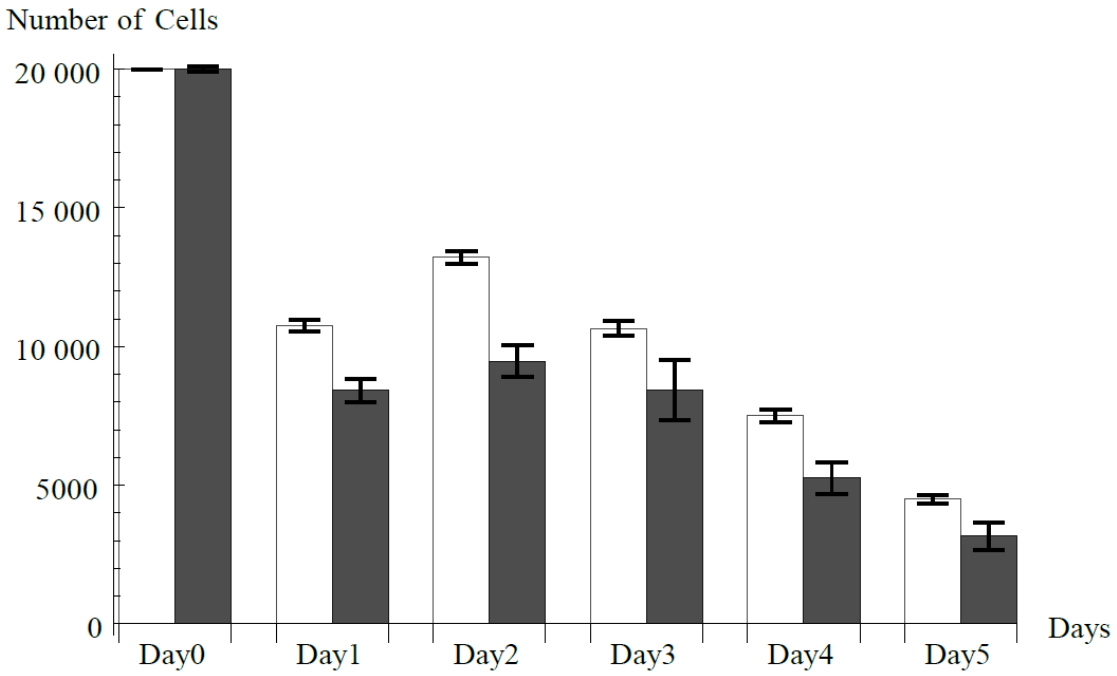


Figure 69: Simulations with combination of 5-FU and hypoxia. 5% arrested agents compared with experimental data: the simulation bars are in white, and the experiment bars are in grey. The value of simulation data is average of 10 runs, and the experimental data is from 3 counts. The simulation and experimental data do not match but can be considered as a good result.

Conclusion and Discussion

In this section, the model developed in Section 3 has been revised to provide a framework for modelling cancer cells. The main amendment is the inclusion of a cell cycle and probability of cell division, which are necessary since they are important factors of cell group population which needs to be calibrated before the group pattern formed by cancer cell.

The cell cycle representation is phenomenological, and in this section, it is calibrated to the population growth curve in the control condition experiment. According to the effect of 5-FU on cell cycle [101], by adding in an extra death rate, the mechanism of 5-FU is added to the initial model. With 5% extra death rate, the simulation of model shows a close result in comparison with experimental data.

However, the same method cannot be used to add the mechanism of hypoxia, as it not only increases the death rate of cells, but also slows their growth and hence extends the length of typical cell cycle [88,100]. The mechanism of hypoxia is modelled by including two cell types: one with higher death rate and normal cell cycle, one with normal death rate and longer cell cycle. Compared with simulation with single-cell-type, simulation of mixed cell type produces better results.

After calibration of model separately with both 5FU and hypoxia experiments, the settings of these two parts are combined together to predict the cell growth with combination experiment. The simulation of combination model shows that, although it does not match the experimental data, it is able to show a similar trend of change of cell population. From this I consider the model can produce proper cell population for spatial pattern study which will be discussed in Section 6.

In the next section I repeat this approach of calibrating to the cell spatial patterning with the built-in cell cycle calibrated in this section: the control condition is calibrated initially, followed by introducing plausible mechanisms to reflect 5FU and hypoxia separately; at last, the combined treatment of hypoxia and 5FU is predicted.

Section 6: Predicting Cell Distribution Curve

Introduction

In section 5, I mapped the population growth curve of experimental data to the model, which ensures that the simulation contains correct number of agents. In this way the inter-cell model is ready to be used to study the features of cell distribution.

The aims of this section are to:

- Find a method to characterize spatial experimental data.
- Using the method found in (1), analyse the data gathered from growth experiments to find the key properties of the distribution of cells under various environmental conditions.
- Find a method to link the spatial distribution of cells to the parameters of the physical model where the model produces similar distributions.
- Compare the parameter setting of physical models corresponding to each experiment, finding the differences between experiments and explain those differences in a biological context.

Time-lapse image process

First of all, the data gathered from the experiment is in the form of time-lapse images. According to Section 5.3, the images are taken with an interval of 10 minutes from when the HCT cells are first established in the petri dish. All of images of each growth experiment are focused on a fixed area of part of one of the 96 wells. In this section I review the process to extract the information relating to cell distribution from the time-lapse images, which is Aim (1) in the introduction of Section 6.

To characterise the cell distribution is to characterise the distribution of cell positions. Thus, to find the key properties of cell distribution, the first step is to register the positions of cells in the time-lapse images. The set of positions is then described by a point process, a spatial statistic method that can measure a distribution in space and provide quantitative descriptors of that distribution.

Pre-process: In the growth experiments, the experimental setup was not ideal due to equipment limitations. The number of cells was counted regularly, a process that perturbed the experimental setup and so may slightly affected the relative position

of petri dish and camera. Thus, some of the time-lapse images are taken under slightly different conditions. Some images vary in brightness, and some blur because of loss of focus. Also, in each image there is a large bright area and two smaller dark areas which are obviously noise. For individual cells to be recognized by image processing software, these problems need to be resolved (Figure 70).

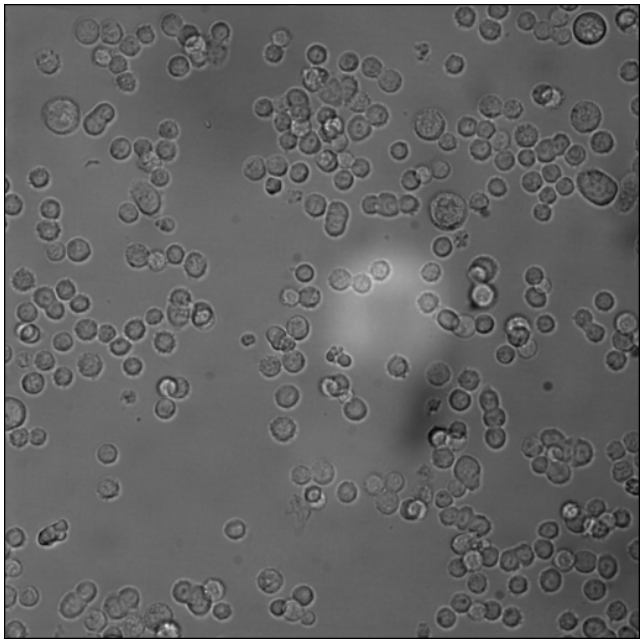


Figure 70: Blurred image with bright spot in the middle and dark spots on bottom- right and top-right part. The aim of image-process is to filter the bright spot and dark spots.

There are three operators in the software ImageMagick (http://www.imagemagick.org/) that can be used to help solve these problems. One of them is the "-level" operator. This operator requires two input values for thresholding upper and lower boundaries: 'black_point' and 'white_point'. Any part that is darker than the 'black_point' is set to be black; and any part that is brighter than the 'white_point' is set to be white. After that "-level" operator scales the greyscale of all the rest part linearly (http://www.imagemagick.org/Usage/color_mods/#level). In this way I are able to increase the contrast of image, as Figure 71 shows. Also, for a group of images taken from one experiment, by assigning the same 'black_point' and 'white_point', the whole set of images can have an even brightness.

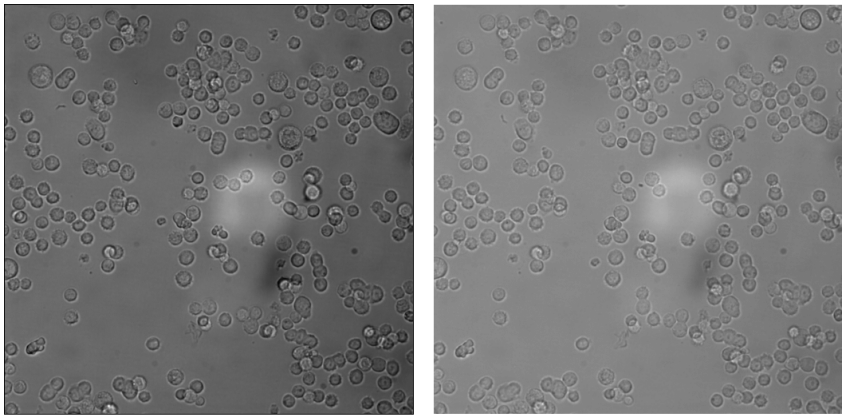


Figure 71: Before (a) and after (b) increase of contrast (-level).

Another operator is "-normalize", which expands the distribution of greyscale of each image to 0~255 (http://www.imagemagick.org/Usage/color_mods/#normalize). The darkest part of each image is converted to greyscale 255 and brightest part of each image is converted to greyscale 0, and for the rest part the greyscale is scaled so that any value between 0 and 255 can be found. In this way, the dark part of the original image looks darker, and bright part looks brighter, as Figure 72 shows.

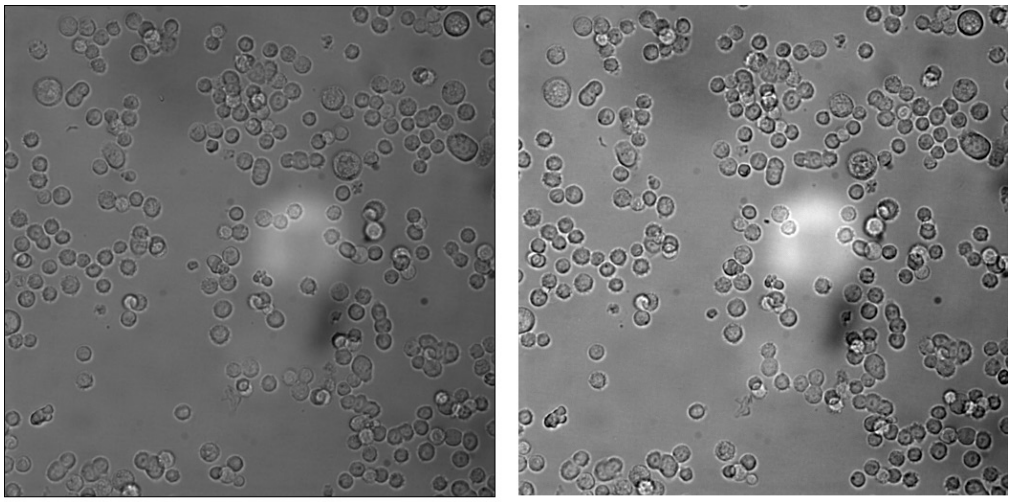


Figure 72: Before (a) and after (b) normalisation (-normalize).

Theoretically the detail of the darkest 2% and brightest 1% are lost with "-normalize" operator, which means the darkest 2% of greys in original images are turned to black and brightest 1% of greys are turned to white. From Figure 72 I can see that this loss of detail does not significantly affect the quality of the image, and importantly the contrast of the image is enhanced.

The last operator is "LoG" (http://www.imagemagick.org/Usage/convolve/#log). It is a high pass filter to get rid of the bright spot and dark spots on the image. The image before and after operation are shown in Figure 73.

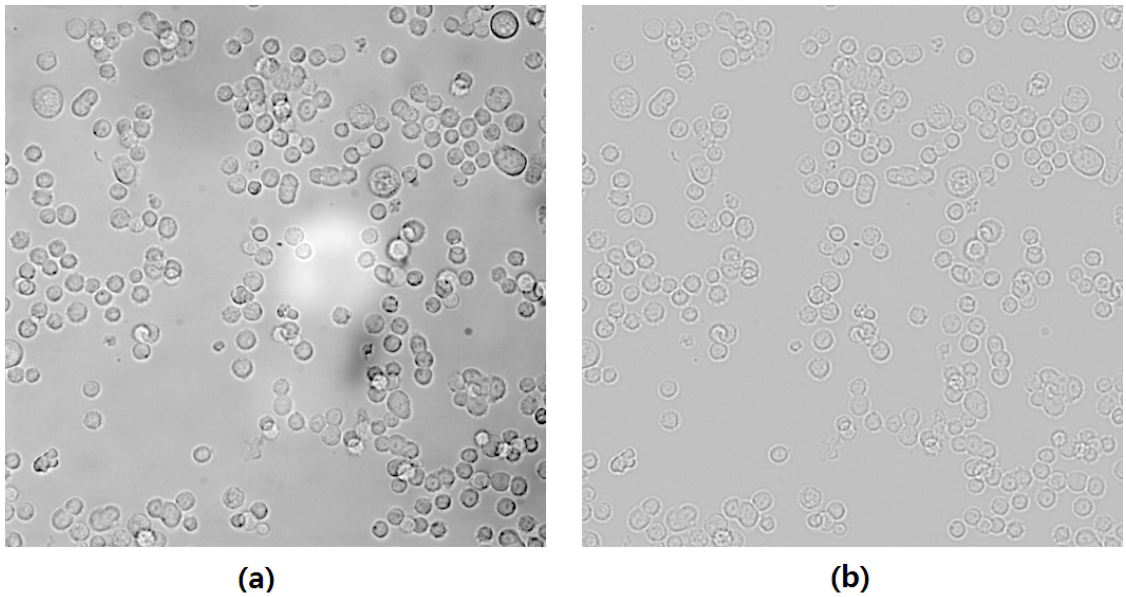


Figure 73: Before (a) and after (b) high pass filter (-LoG).

An important consideration is the order of application of these operators. Because the "-normalize" operator increases the contrast within an image, the three largest noisy areas (the dark and bright regions) will be more pronounced and thus harder to remove if "-normalize" is used before "-level". On the other hand, the operations that increase the contrast of image, such as "-normalize" and "-level", can produce noise during rescaling of the greyscale, as Figure 74 shows. Therefore the "-level" operator is the first to be applied, followed by "-normalize; "LoG" is the last operation.

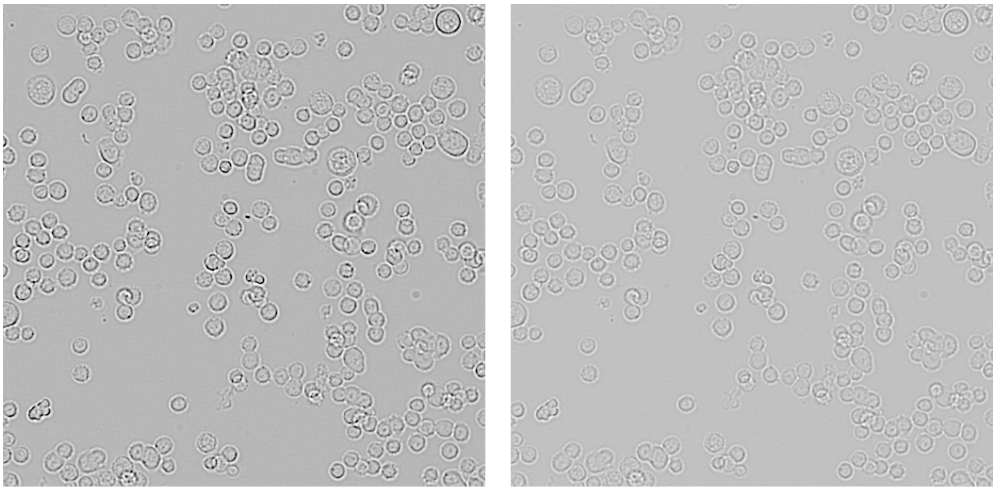


Figure 74: Result of high filter pass then normalise (a) and normalise then high filter pass (b). Cells are more obvious in (a).

Recognising cell outlines in Cell Profiler: CellProfiler is a free to use cell image analysis software, available from [http://www.cellprofiler.org/]. It provides functionalities to allow recognition of the edge of cells and recording of cell positions through a graphical user interface.

Each processing function is packaged as one module with a separate setting up process. Several modules can be combined in a specified order, so that the output of a previous module can be used as an input of a later module. In this way, several modules operating in a fixed order form a pipeline that can be saved or loaded in CellProfiler. CellProfiler automatically runs all modules in a pipeline with that fixed order.

One of the modules to load an image is always set as the first module in the pipeline. In the 'Image' module, image files or folders are dragged to the interface. The second module in the pipeline is 'Metadata'. The purpose of this module is to extract the pattern in the naming of loaded images, so that a similar name can be assigned to the corresponding output file. Then in module 'NamesandTypes', the image type is to be set as 'Greyscale image'; also, the group of images is assigned a name to be used in the following modules.

The module used after input modules is 'IdentifyPrimaryObjects'. In this module, the edge of cells is identified from input images. An identification result is shown in Figure 75: the lines in green are accepted edges; the ones in red are unacceptable edges according to their size; and the yellow lines are unaccepted edge because they touch the edge of image.

It is recognized that for some images, different values of parameter are needed in the 'IdentifyPrimaryObjects' module, which are set manually. For details of actual value, please refer to Appendix D. This non-systematic threshold adjustment is not an ideal image recognition process. However, image recognition is not the aim of this chapter: it is a pro-analysis step to retrieve data from time-lapse images for further study, thus the driving concern here is that data generation. A more controlled experimental system and more time spent investigating image analysis algorithms would overcome this limitation in the cell recognition process.

After module 'IdentifyPrimaryObjects', module 'ExportToSpreadsheet' is called to save position of identified cells to text files. I need to set the output data from module 'IdentifyPrimaryObjects', the format of name of output file which is from 'Metadata' module, and how each column of data is separated.

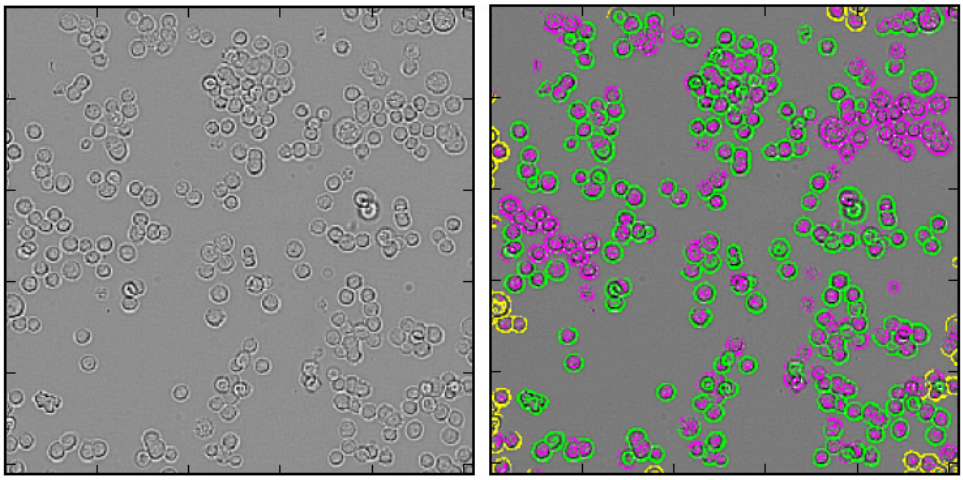


Figure 75: A time-lapse from experiment (a) being identified (b). The green parts in (b) are accepted edges of cells; the red and yellow parts in (b) are unacceptable edges of cells. Note the red part on up-right corner, where a cluster of cells is failed to be identified. This problem is discussed in Section 6.1.3.

CellProfiler parameter calibration: In Section 6.1.2 it is claimed that the parameters are chosen in order to produce the best result. Firstly, the parameters are tested, and values of parameters and number of cells found are recorded. The values are chosen so that the greatest number of cells are recognized while the lease sundries are recognized as cells. To calibrate the parameters in CellProfiler, the identified cell positions are drawn back to the original image and compare how good they fit. Before drawing cell positions onto time-lapse images, the cell position data needs to be filtered. Because sometimes the output file from CellProfiler contains duplicated cells or two cells with very close positions. In addition, the output from CellProfiler also contains image ID and cell ID that need to be skipped. A separate c++ program DrawCellPosition.sln has been implemented to remove the unnecessary columns from output data and filter out cells that are too close or are duplicated. After that the filtered positional data is drawn as a white cross on a copy of time-lapse image to show filter result (Figure 76).

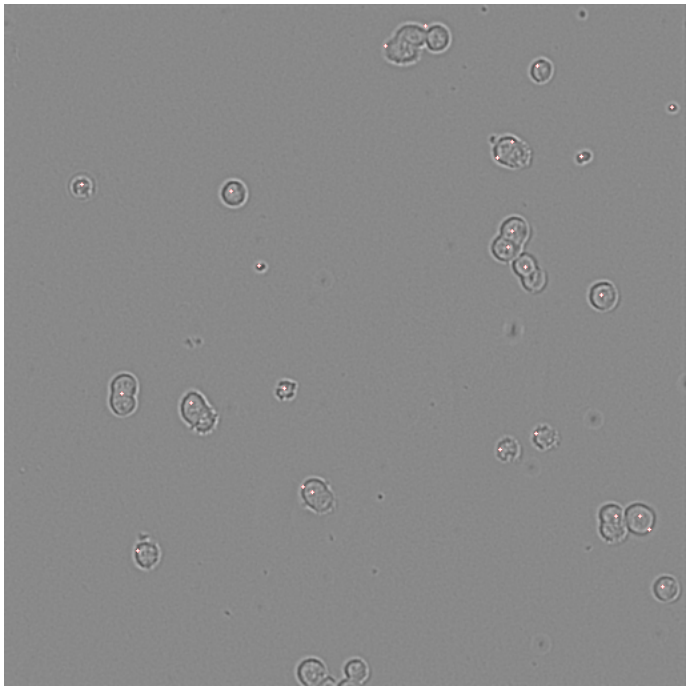


Figure 76: Time-lapse image with identified cell position drawn as white cross.

By plotting the recognized cell position on the original image, it is found that cluster of cells cannot be properly recognized. In most common circumstances, the outline of clustered cell group can be recognized, while the edge of each cell within cluster cannot be distinguished and thus the cluster is considered as one single cell positioned at the centre of the cluster (Figure 77).

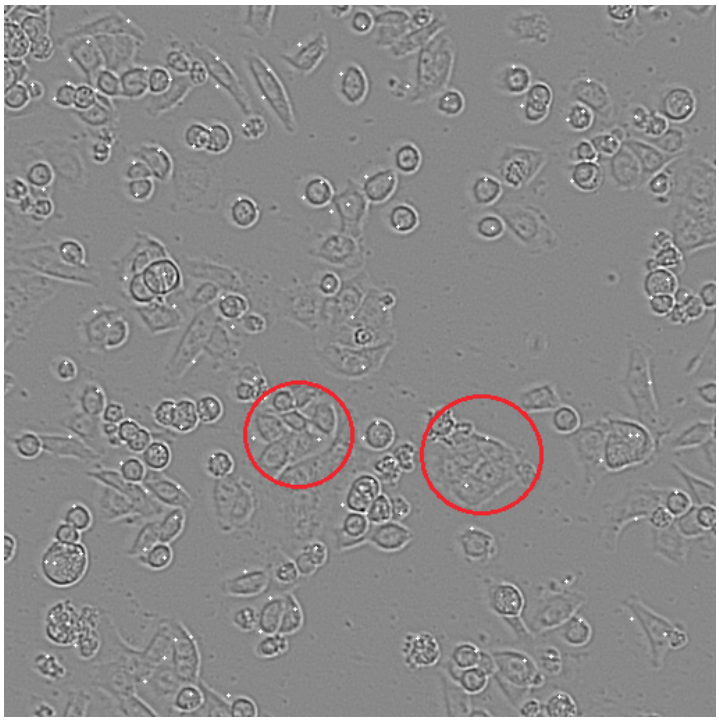


Figure 77: Cluster considered as single cell.

To investigate how much error is introduced to spatial distribution measurement by failure to recognize cells in clusters, a test is carried out in Section 6.2.3 to compare the difference of spatial distribution between the conditions that cells in clusters are recognized and are not recognized.

Cell recognition problems and manual processing: In practice, by using the approach outlined in Section 6.1.3, it was found that CellProfiler cannot distinguish between individual cells within a cluster. Although the edge of cells in clusters is clear for human eyes to identify, the edge is not a closed circle due to lighting condition and in CellProfiler non-closed circle cannot be identified as a cell. This lack of differentiation means that both the total number of cells and the measurement of the distance between cells are imprecise. In order to determine how much error recognition problems bring to the measured distribution, the position of cells of the same images (that has been processed by CellProfiler) are manually marked and recorded. The total number of cells identified by CellProfiler and by hand is shown in Table 17.

Table 12: Number of cells identified by CellProfiler and by hand.

	First Image	Middle Image	Last Image
Number of cells in control group identified by CellProfiler	29	26	38
Number of cells in control group identified manually	24	28	21
Number of cells in 5-FU group identified by CellProfiler	89	85	111
Number of cells in 5-FU group identified manually	243	240	192
Number of cells in hypoxia group identified by CellProfiler	89	173	170
Number of cells in hypoxia group identified manually	146	148	122
Number of cells in combination group identified by CellProfiler	100	108	147
Number of cells in combination group identified manually	140	144	143

From Table 12 I can see that the numbers of identified cells are similar for the control group; for other groups, the number of cells identified by CellProfiler is lower than manually identified data, especial for the 5-FU group. However, comparing with manually process, the CellProfiler can identify similar number of cells with manually for the last image of combination group, which means the performance of CellProfiler is unstable.

To test how different identified cell numbers affect the measurement of cell distribution, the pairwise correlation function for both CellProfiler and manually recognized cell position is separately calculated and plotted.

Comparing the pairwise correlation function in Figure 78, Figure 79, Figure 80 and Figure 81, it suggests that for the control group, there is not much difference between CellProfiler generated and manually generated data. However, for all the other experimental groups, the CellProfiler processed data loses the first peak of pairwise correlation, which is caused by the loss of data of the close distance between cells that are in the same cluster; and the overall height of the CellProfiler curve is slightly lower than the manually processed data, which is caused by the reduced total number of recognized cells. The reason that the control group is not significantly affected is that in the image of control group there are much fewer clusters, and within each cluster the number of cells is much lower.

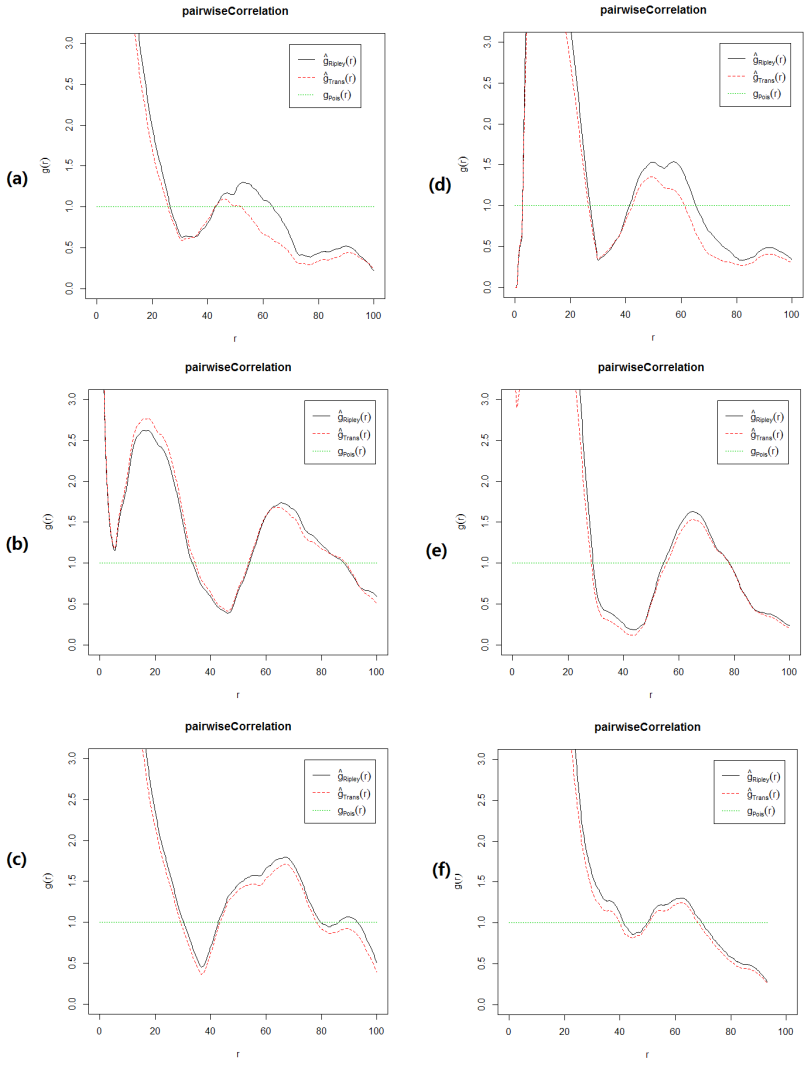


Figure 78: Pairwise correlation of the HCT wild type control group. (a) - (c) are generated with first, middle and last image recognized by CellProfiler, and (d) - (f) are generated with first, middle and last image manually recognized.

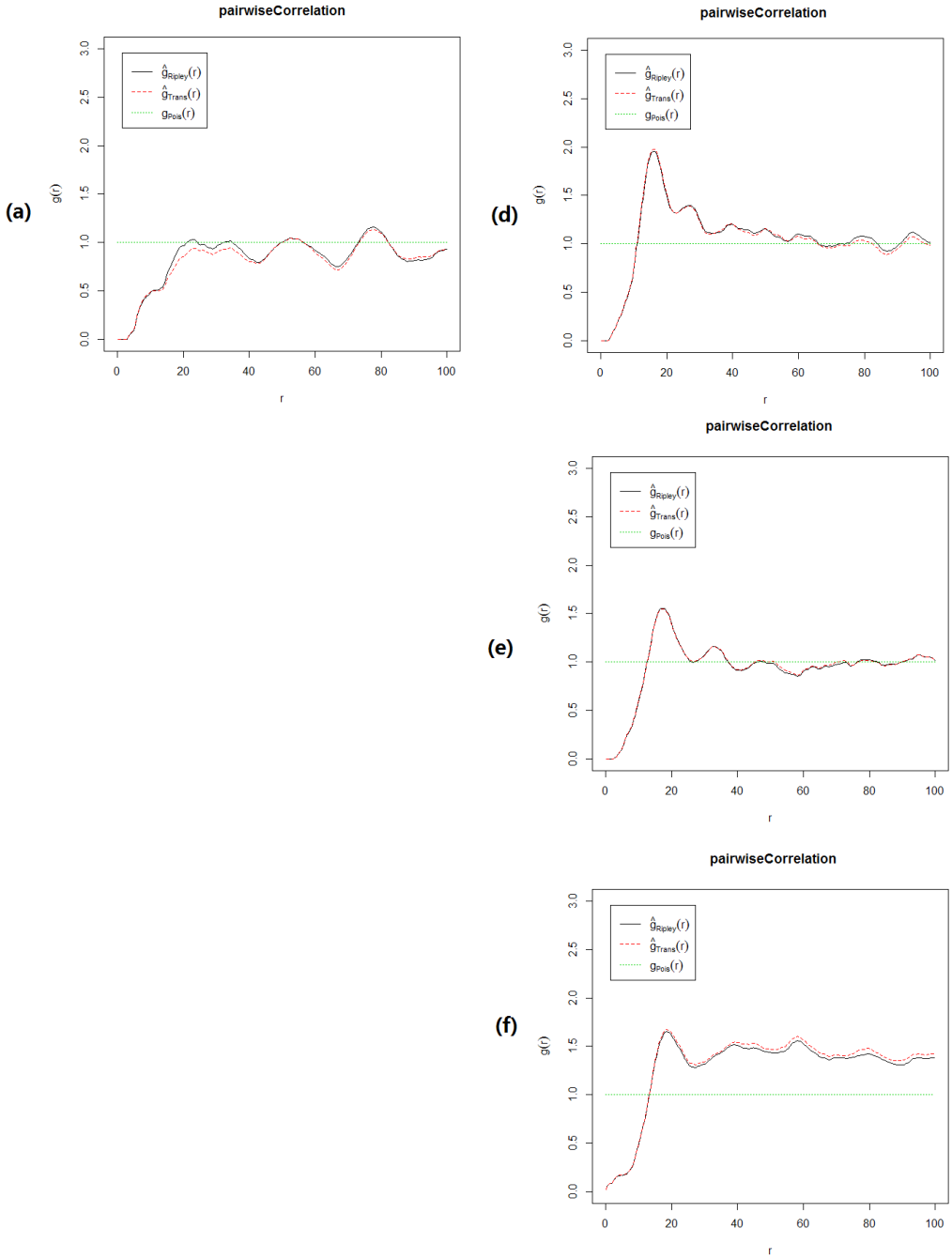


Figure 79: Pairwise correlation of HCT wild type with 5-FU. (a) is generated with the first image recognized by CellProfiler, while (d) - (f) are generated with the first, middle and last image manually recognized. Note that (b) and (c) are missing. Because from the middle and end image of 5-FU group, the CellProfiler cannot identify enough cell to produce pairwise correlation curve.

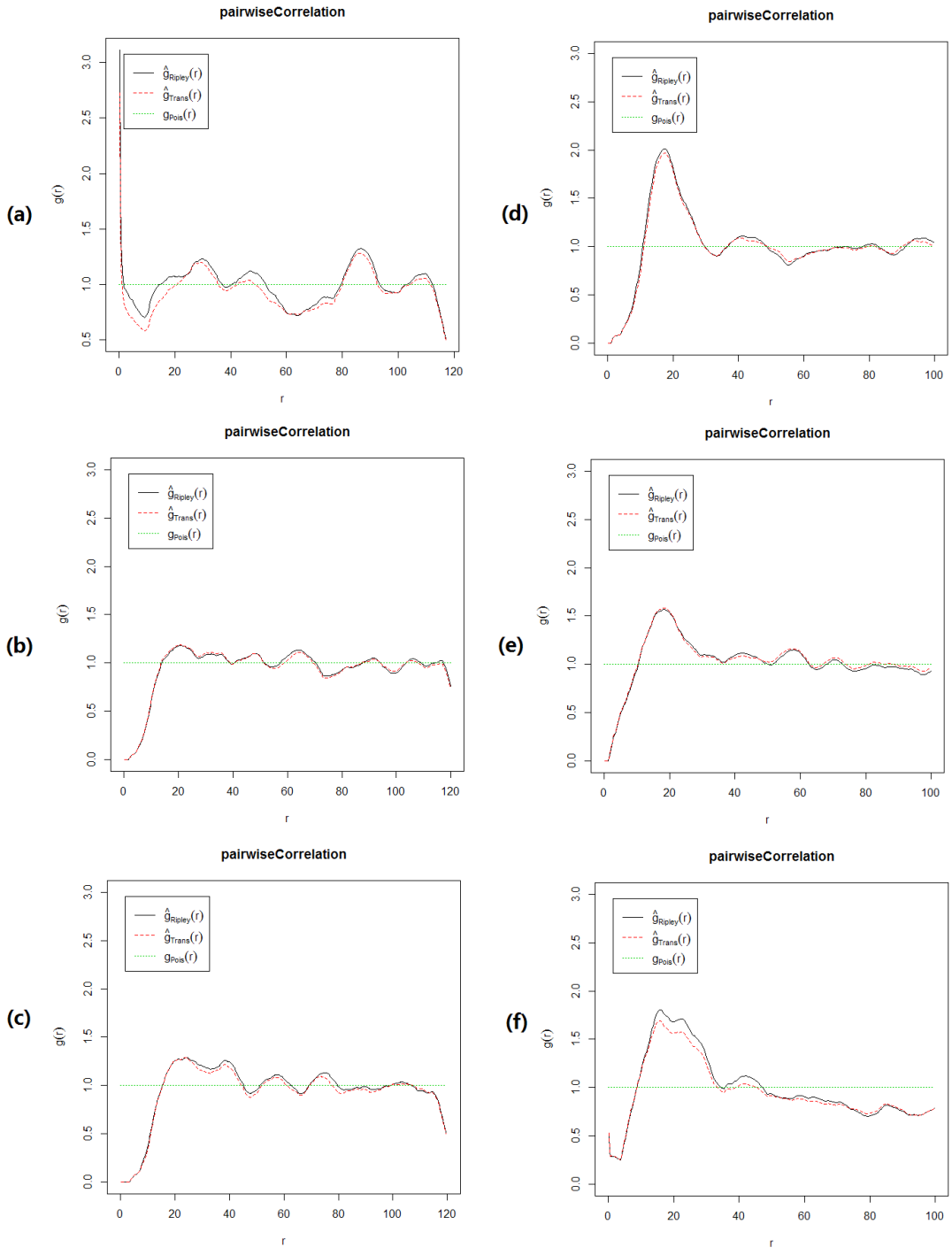


Figure 80: Pairwise correlation of the HCT wild type with hypoxia. (a) - (c) are generated with first, middle and last image recognized by CellProfiler, and (d) - (f) are generated with first, middle and last image manually recognized.

Finally, due to the imprecise result produced by CellProfiler, I choose to use manually process data for later analysis in all the following sections. Also, the conclusion of this section is obtained from the manually processed data. While not an ideal solution, image analysis is not a major focus of this chapter and so a manual analysis was judged an acceptable solution (it is discussed in Section 7, Discussion).

Fitting experimental parameters to the starting point of the simulation

Before being analysed, the experimental data and simulation result should be measured at the same scale. The diameter of cells in the control experiment is observed to be 10 μm; note in the time-lapse image the diameter of one cell is about 10 pixels. Thus, I consider in time-lapse image, each pixel represents an $1\mu m \times 1\mu m$ area, and on spatial statistical analysis curves each unit represents length of $1\mu m$. As the sizes of the images are 512×512, each image represents an $512\mu m \times 512\mu m$ area. Similarly, the result of a simulation needs to be of the same scale. The output position of agents is then expressed in terms of unit length. To unify the experimental and simulation scales, the output value should multiply value of unit length, then divide by $1\mu m$.

The result of simulation is measured by unit length, which equals 7.5×10^{-5} m. To convert the simulation, result so that it is measured in micrometers, which equals 10^{-6} m, the position data needs to be multiplied by 7.5×10^{-5} then divided by 10^{-6}, as shown in Equation (165).

$$NewPosition = \frac{OldPosition \times unitlength}{micrometer} \tag{186}$$

For each experimental group, I select images that depict the beginning of the experiment, mid-point and end of the experiment to carry out image recognition of the edges of cells, which result is later analyzed and compared with the point process.

The number density, on the other hand, should be of same value in the start point of both simulation and experimental data regardless of scale, so that the simulation starts from the same condition of experiment.

As the intensity of experimental data is known at the starting point of simulation, the number of agents should be the same as the number of cells in the first image from each experiment. In the simulation, there is a setting for the density of agents, which translates to the number of agents per unit area. The number density of agents is calculated as follows:

number density of agents = number of cells in experiment / area of substrate in simulation (187)

New unit length: 7.5e-005 m; New unit time: 60 seconds; Area of simulation substrate: 6.83-unit length * 6.83 unit length; So: 6.83*7.5e-005 m = 512.25 e-006 m -> experiment area $512\mu m \times 512\mu m$

Table 13: number density of agents in three experimental groups.

	Number of Cells	Corresponding Number Density of Agent
5-FU group	243	5.21
Hypoxia group	146	3.13
Combination of 5-FU and hypoxia group	140	3

The values of parameters such as unit length etc. are calculated for cancer cell growth as well; see Section 3.9 and Appendix B for full detail.

Table 14: Value of parameters used in simulation of cancer cell growth. Note that the values of K and ε are their dimensional value $\tilde{K}$ and $\tilde{\varepsilon}$. As $\tilde{K}$ and $\tilde{\varepsilon}$ do not appear in Section 3, I use K and ε here for a better understanding.

Name of Parameter	Value
Unit length Δx	7.5×10^{-5} m
Unit time ΔT	1 minute
Density of cell	$2000kg/m^3$

The proportion of typical cell size and unit length	Three semi-radii of ellipsoidal agent: 1.25×10^{-6} m, 5.0×10^{-6} m, 5.0×10^{-6} m
Dynamic viscosity of fluid in which cells are moving	$1\times10^{-3}\,Pa\cdot s$
Constant for contact force K	25.6×10^{12}
Constant for adhesion force ε	0.6×10^{6}

In the simulation, the initial position of all the agents is randomly generated. To test the functionality of randomized position generation, the K-function curve of starting condition of simulation is plotted and compared with K function curves of the corresponding experiment, as shown in Figures 82-84.

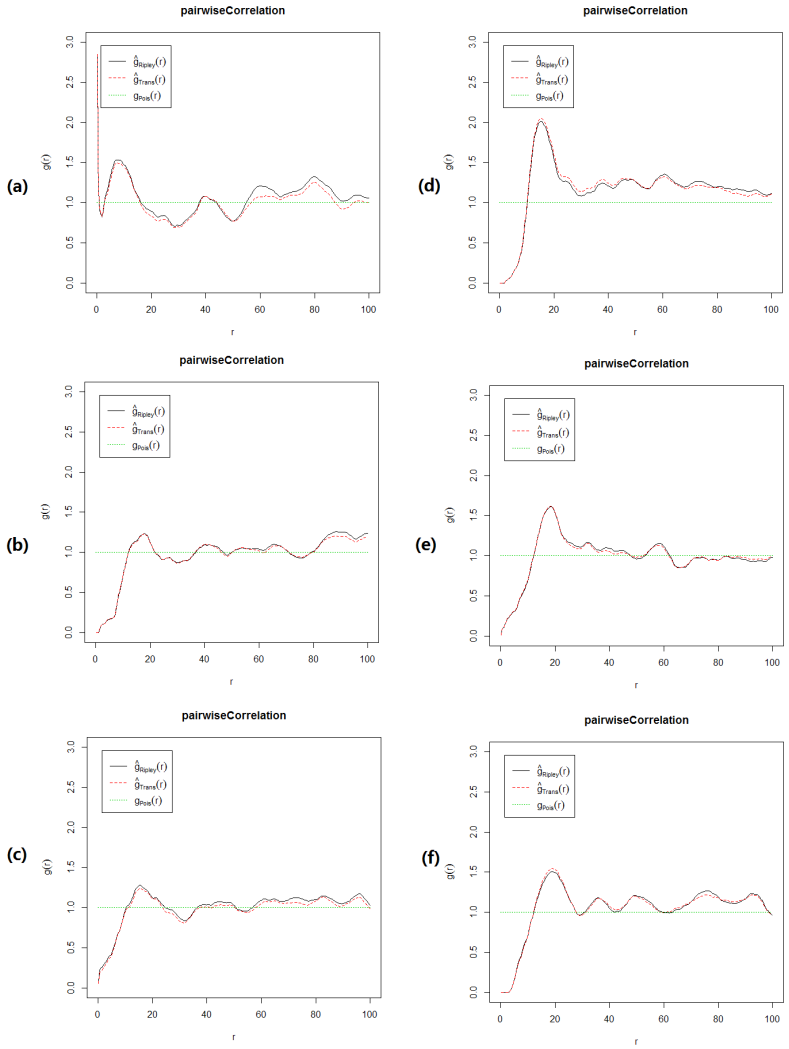


Figure 81: Pairwise correlation of the HCT wild type with combination of 5FU and hypoxia. (a) - (c) are generated with first, middle and last image recognized by CellProfiler, and (d) - (f) are generated with first, middle and last image manually recognized.

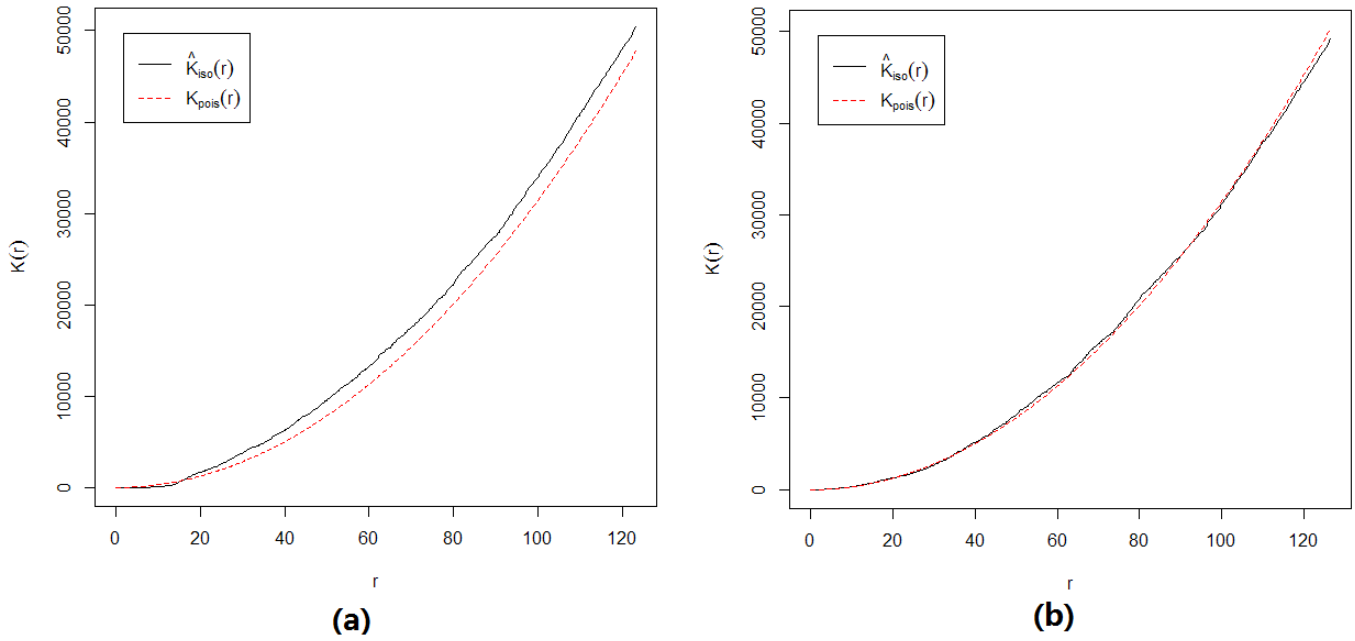


Figure 82: (a) K function of 5FU experiment group, (b) K function of 5FU simulation.

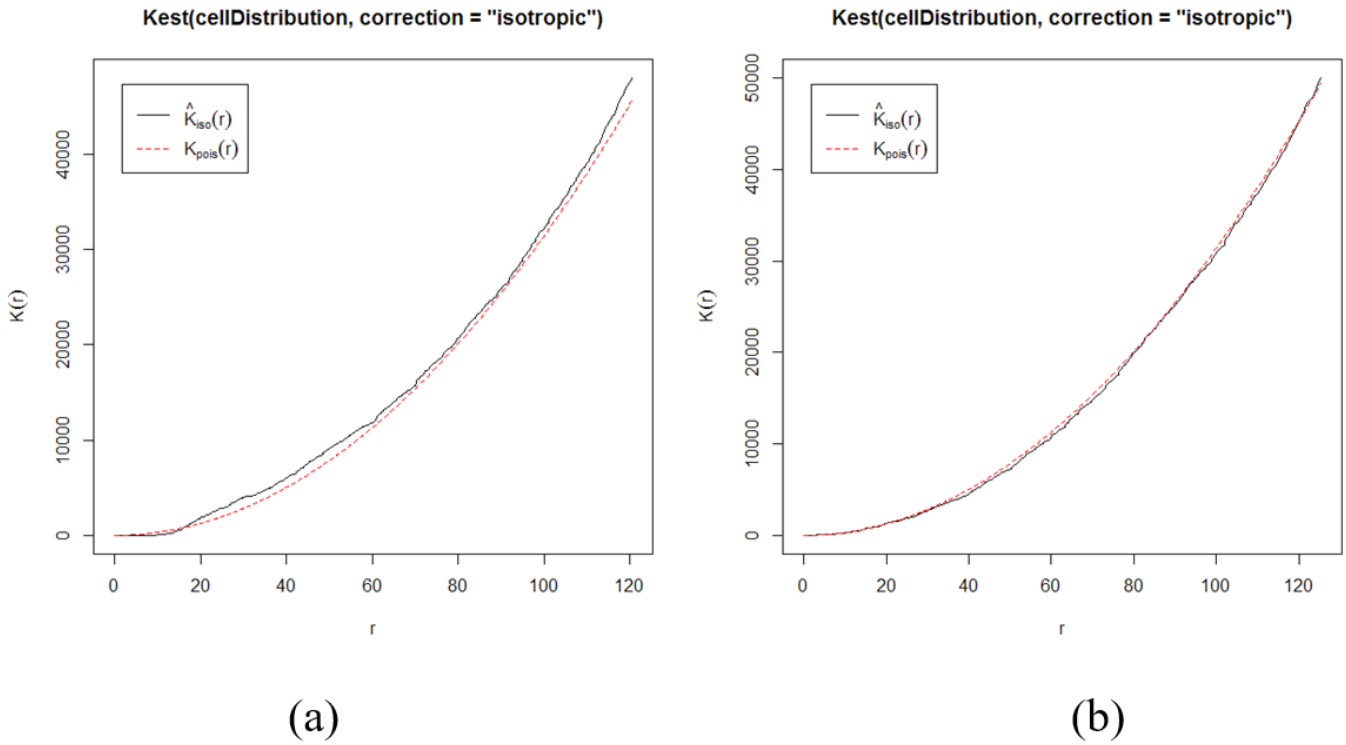


Figure 83: (a) K function of hypoxia experiment group, (b) K function of hypoxia simulation.

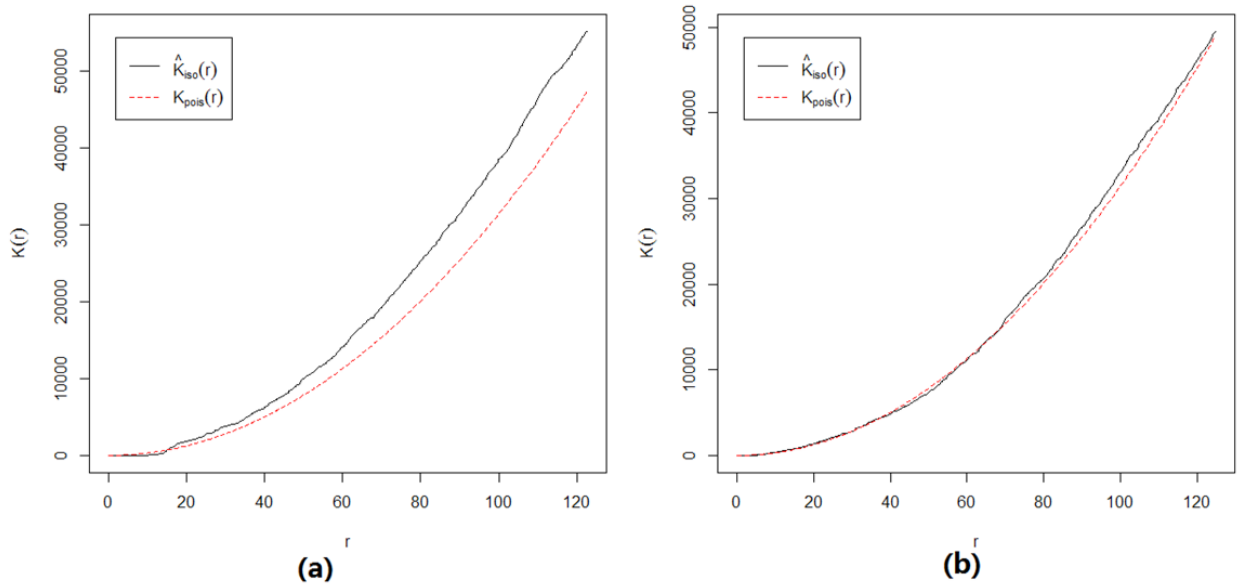


Figure 84: (a) K function of combination of 5FU and hypoxia experiment group, (b) K function of combination of 5FU and hypoxia simulation.

I also fit the starting point of 3 simulations to stationary Poisson process. The output number density of simulation is shown in Table 15.

Table 15: Number density of 3 experiments and start point of simulation.

	Number Density
Experiment group: 5-FU	0.00099
Simulation: 5-FU	0.00094
Experiment group: Hypoxia	0.00061
Simulation: Hypoxia	0.00057
Experiment group: combination of 5-FU and hypoxia	0.00058
Simulation: combination of 5-FU and hypoxia	0.00055

From the K-function curve and number density value, it is demonstrated that, at starting point, the simulation is able to generate similar distribution with experiment. The scaled simulation is ready to begin with parameter fitting.

Pairwise correlation and 4 main parameters that affect the distribution curve: Before full calibration of my model, and to inform the parameter fitting process, I carried out instructive simulations to explore the relationship between model parameters and agent distribution. By observing the shape of the pairwise correlation curve, I am able to make initial assumptions about the link between the form of the pairwise correlation curve and the value of the parameters to the inter-cell interaction model. For example, Figure 85 shows the starting point of an explorative simulation, in which the pink particles are agents and are randomly generated, the blue part is the back group representing the substrate, and the distribution of agents is shown with a top-down point of view.

In this section I will explore a couple of key parameters in the model by running simulations with various parameter settings and compare the simulation result. Then it is known how each parameter affects the pattern formed by agents. The positions of agents of each simulation result are used to generate pairwise correlation curve. Then the effect of each parameter is linked with change of shape of curve (hopefully).

With the definition of pairwise correlation function discussed in Section 6.2.1, I give meanings of each part of pairwise correlation curve as Figure 86. Note that the x-axis is unit distance between pair of agents, and y-axis is value of pairwise correlation function at each distance. The green dotted line is where pairwise correlation function . The part of curve above

dotted line means the agents are clustered at this distance; the part of curve beneath means the agents are separated at this distance; the part of curve on dotted line means the agents are randomly distributed at this distance. The distribution range means the longest range on which another agent is found. The highest point is a value generated during the process of calculation of pairwise correlation function and I believe it represents the degree of gathering of agents. These parts are considered as key features of each curve and will be compared in the following discussion.

The range that adhesion (attraction) force takes effect:

Firstly, I consider the range that adhesion force takes effect. Note that the adhesion force attracts agents together, while the contact force repulse agents to prevent them from totally overlapping. As the contact force affects agents only when they are in contact, the range of contact force is fixed and does not need to be compared. 4 simulations are run, with range of adhesion force set to 100, 10, 3 and 1 separately. The results are shown in Figures 87-90.

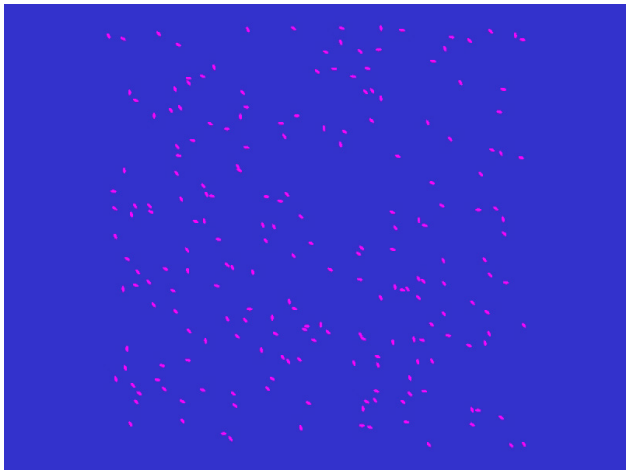


Figure 85: Starting point of simulation. Pink dots are agents which position and directly are randomly generated.

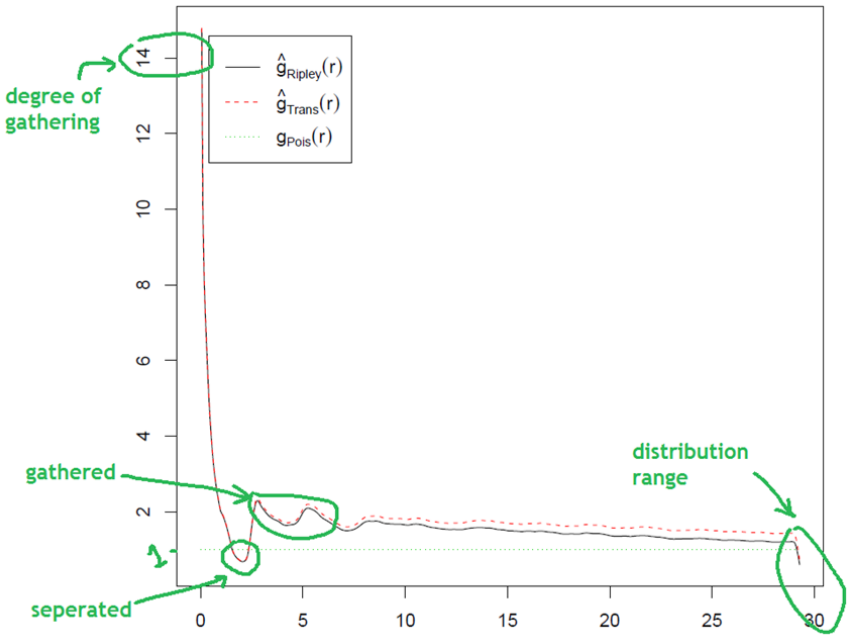


Figure 86: Interpretation of parts of pairwise correlation function.

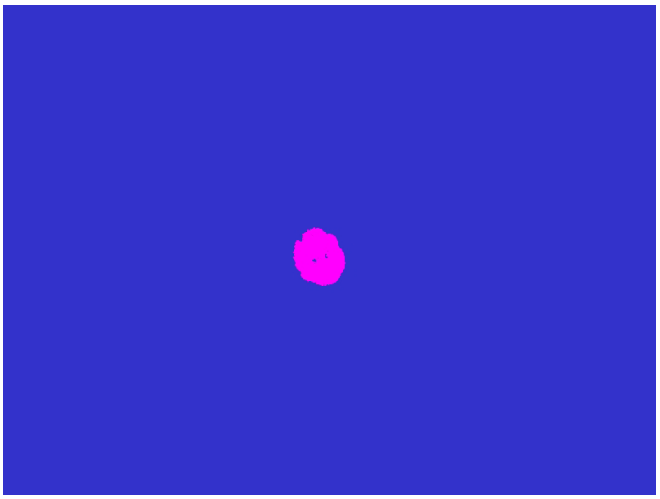


Figure 87: Pattern formed with range of adhesion force equals 100. All the agents gather and form one tight cluster.

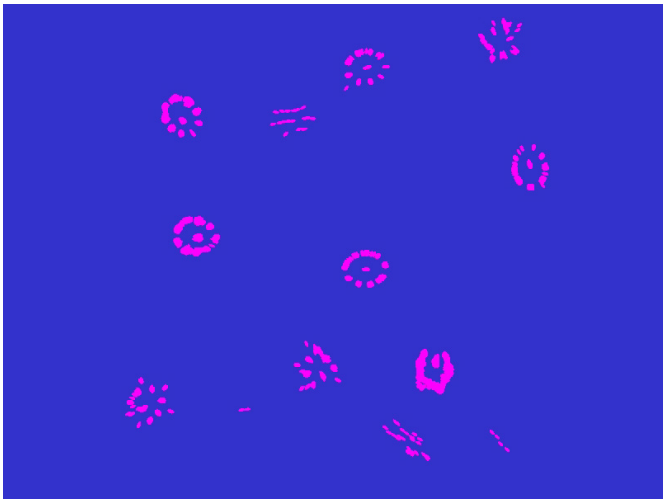


Figure 88: Pattern formed with range of adhesion force equals 10. Agents form several clusters.

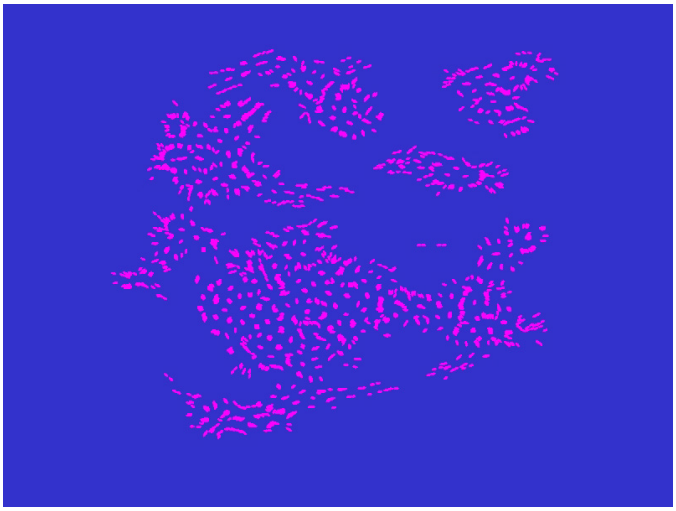


Figure 89: Pattern formed with range of adhesion force equals 3. Agents form more but smaller clusters.

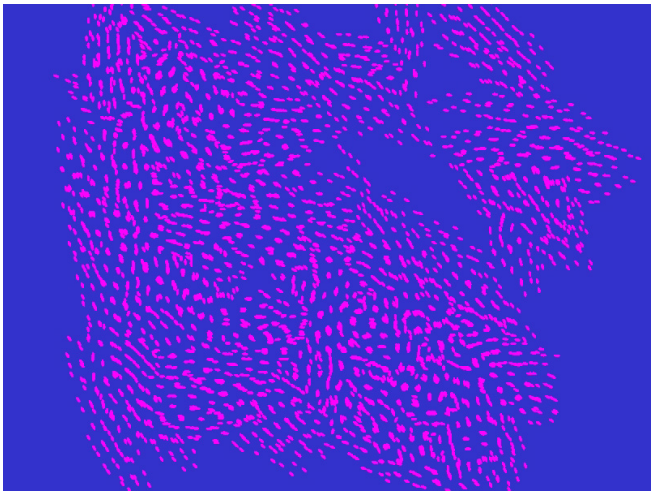


Figure 90: Pattern formed with range of adhesion force equals 1. Some agents form small clusters. Some agents are too far away from their neighbours to be attracted together and remain separated.

The corresponding pairwise correlation curves are shown in Figure 91.

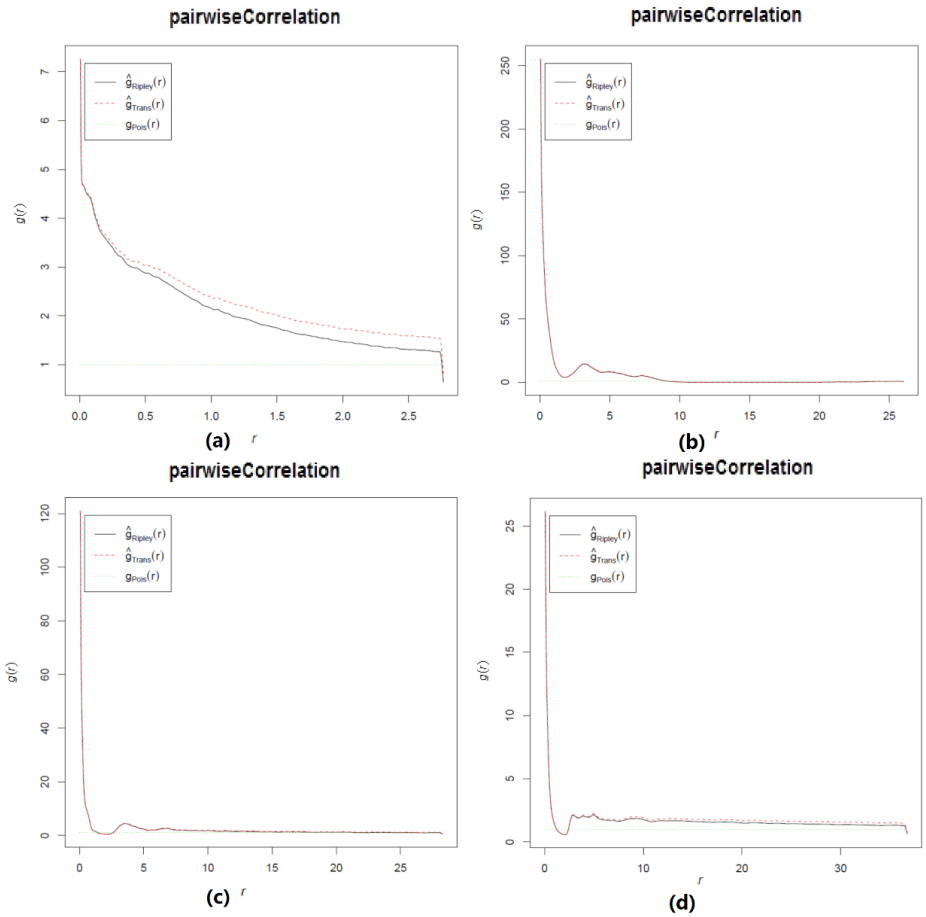


Figure 91: (a) Pairwise correlation curve with range of adhesion force equals 100; (b) pairwise correlation curve with range of adhesion force equals 10; (c) pairwise correlation curve with range of adhesion force equals 3; and (d) pairwise correlation curve with range of adhesion force equals 1.

The longer the effective range is, the more agents are stuck together. Figure 91 (a) which is produced from Figure 87 shows a short distribution range (2.5), because the adhesion force acts on a long range, all the agents gather up together and form one big cluster; and on any distance within the distribution range, the curve is above $g(r)=1$ which means agents cluster on all distance. Figure 91 (b) is produced from Figure 88, in which the effective range of adhesion force is 10 and agents gather to form several clusters. Thus, the distribution range in the curve is larger than the top left curve. The curve forms a peak near distance=5 and a lowest point near distance=2.5, which means agents tend to gather up at distance=5 and disperse at distance=2.5. Figure 91 (c) is produced from Figure 89, in which the range of adhesion force is further decreased to 3. From the curve I can see that the typical distance between pairs of agents remains the same, while the distribution range slightly increases. Figure 91 (d) is produced from Figure 90. The distribution range in this curve is larger (>30), because in this figure the range of adhesion force equals 1, only nearby agents may attract each other, and many agents are not close enough to be attracted together. The degree of gathering up decreases from Figure 91 (b), Figure 91 (c) and Figure 91 (d), as in the corresponding figures, fewer agents gather to form clusters. The positions of peak and lowest point of curve are nearly the same, thus I consider it not controlled by the range of adhesion force.

Point that adhesion force and contact force balance:

The contact force acts when agents are in contact as a repulsive force. The more two agents overlap, the larger the contact force is. The direction of contact force is always opposite to direction of adhesion force. There is a point where 2 forces are the same size and therefore balance, which controls the agents to remain static state. At static state the agents do not move closer to each other, nor move away from each other. The balancing point is represented by the value of potential of agent at balancing point. The larger the value of potential is, the less overlap agents get when they are static, which represents less elasticity of agents. On the other hand, the smaller the value of potential is, the more overlap agents get, which represents more elasticity of agents.

In the explorative simulations to test the effect of balancing point, I keep the settings for range of adhesion force = 1. Because a larger range of adhesion force forms larger clusters from which it is hard to compare the difference among balancing point settings. I choose potential value = 0.9 to represent fewer elastic agents and potential value = 0.5 to represent more elastic agents. The patterns generated with the 2 settings are shown in Figures 92 & 93 separately.

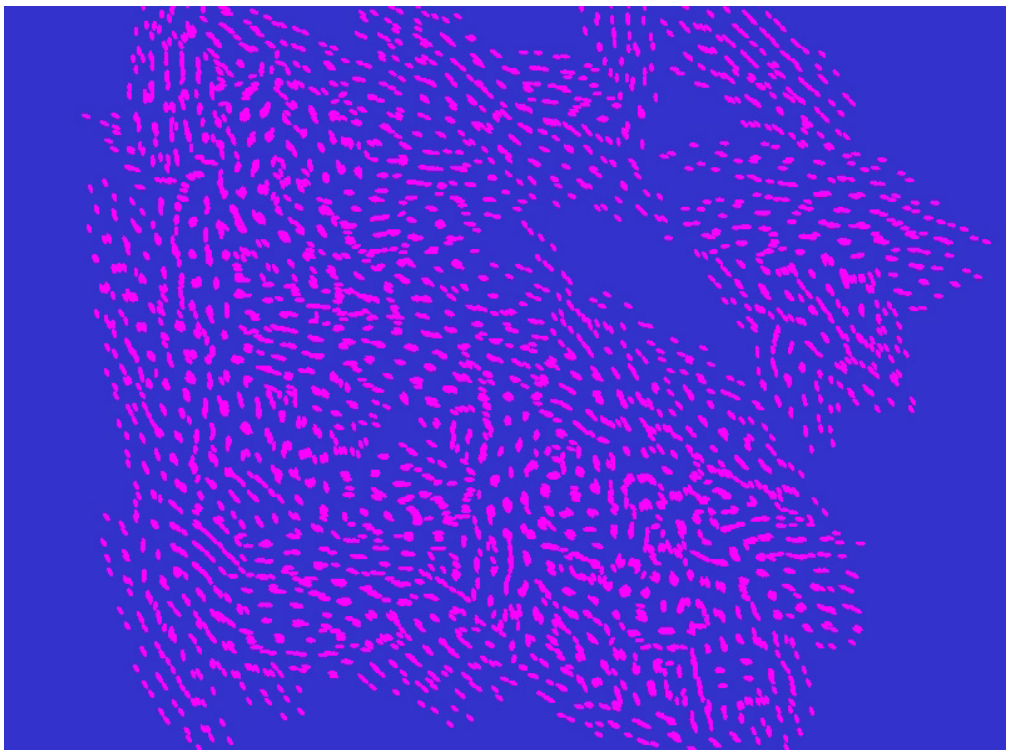


Figure 92: Balancing point: potential = 0.9; range of adhesion force equals 1.

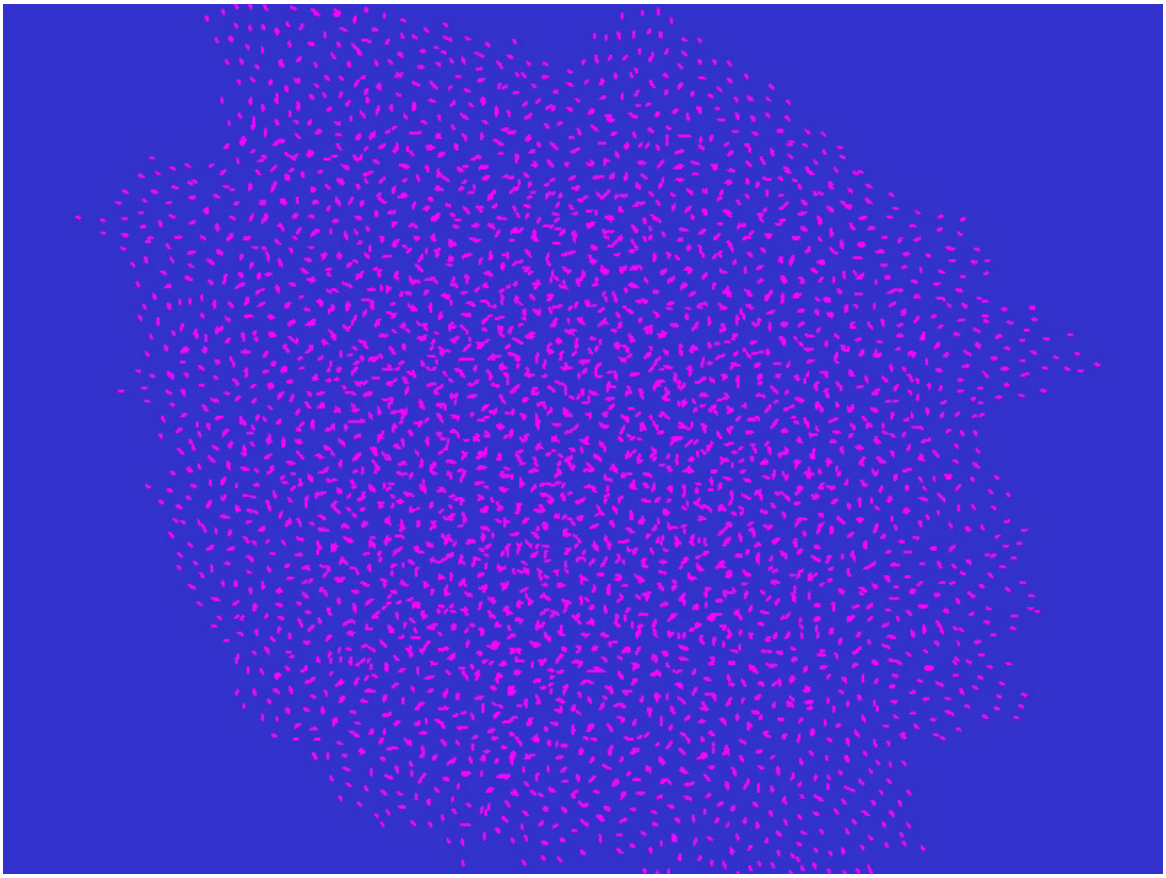


Figure 93: Potential = 0.5; range of adhesion force equals 1.

Again, I draw pairwise correlation curve to have a direct comparison of the 2 patterns, as shown in Figure 94.

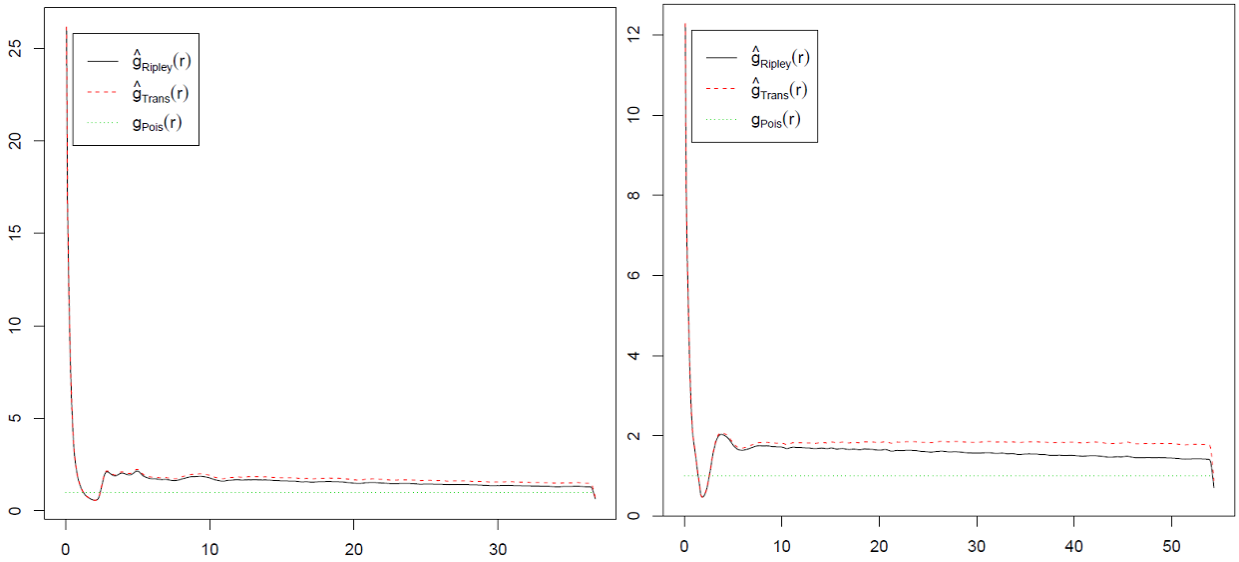


Figure 94: pairwise correlation curve of patterns with potential = 0.9 (left) and potential = 0.5 (right).

From Figure 94, I can see that when contact (repulsive) and adhesion (attractive) force balance at high potential point, the degree of gathering value increases, and the distribution range decreases. However, the position of lowest point and peak do not change. Therefore, the position of lowest point and peak are not affected by balancing point either. I assume that they are only affected by the shape of agents.

Death rate

Although the death rate is studied and fitted in Section 5, its effect on pairwise correlation curve is explored in this section. In the following 3 simulation results, the agents have 80%, 40% and 20% chance to die when they finish cell cycle. The dead agents are removed from simulation.

The pairwise correlation curve produced from Figure 95 is shown in Figure 98 (a), the pairwise correlation curve produced from Figure 96 is shown in Figure 98 (n), and the pairwise correlation curve produced from Figure 97 is shown in Figure 98 (c).

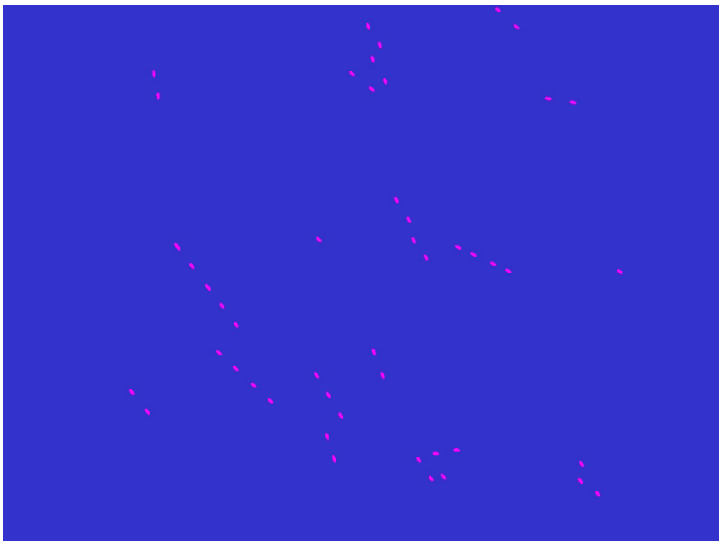


Figure 95: Pattern formed with death rate = 80%.

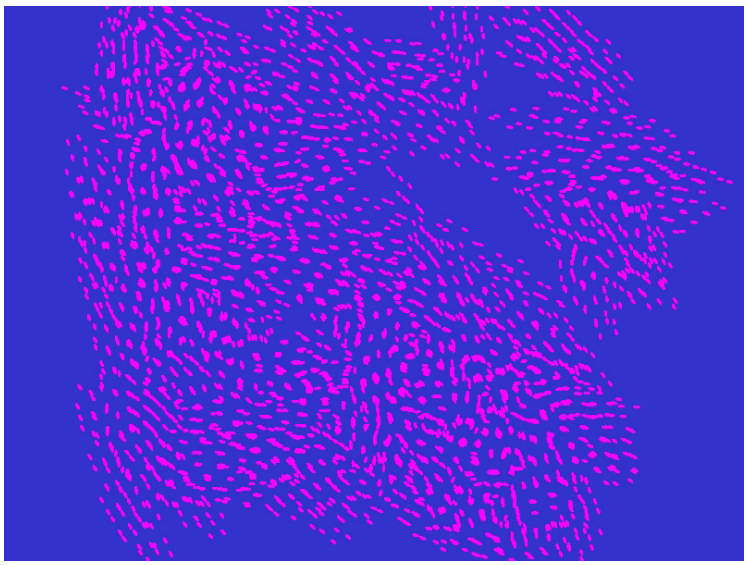


Figure 96: Pattern formed with death rate = 40%.

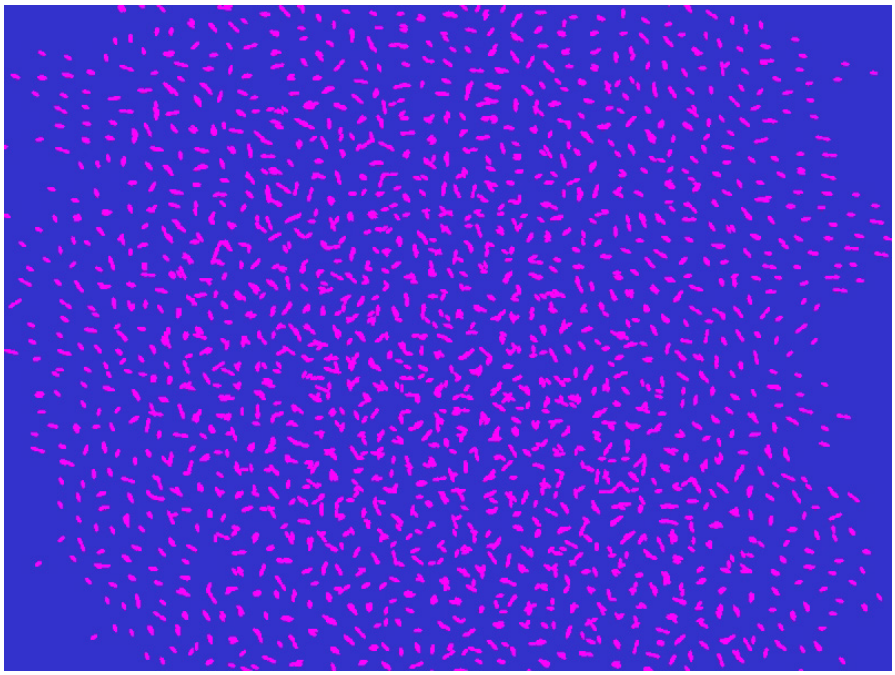


Figure 97: Pattern formed with death rate = 20%.

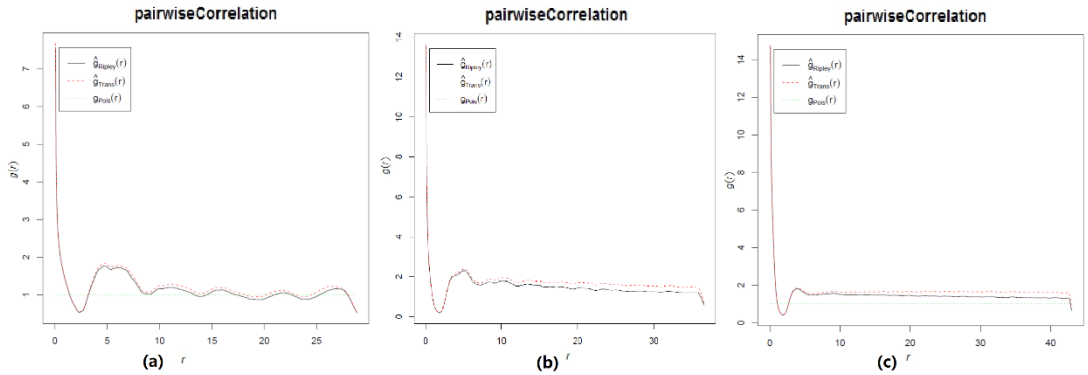


Figure 98: (a) Pairwise correlation curve of death rate = 80%; (b) pairwise correlation function of death rate = 40%; (c) pairwise correlation function of death rate = 20% (right).

From Figure 98, I can see that the higher the death rate is, the lower the degree of gathering is, and the shorter the distribution range is. This is reasonable, as with lower death rate there are more agents in the simulation result, thus for any agent there is higher chance to find another agent.

Fitting to the 5FU experiment: The time-lapse image and corresponding pairwise correlation function for the experimental data associated with the 5FU experiment is shown in Figure 99. According to Section 2.5.2, the shape of function shows that repulsion exists in the distribution of cancer cells in growth experiment with effect of 5FU. At the starting point of the experiment (Figure 99, time-lapse image (a) and pairwise correlation function (d)), there is a feature peak near distance = 20, which indicates the typical distance between typical cell and its neighbours. For distance >40 the cells tend to distribute randomly. However, it is assumed that the agents are randomly distributed at the starting point, thus the experimental data does not match the assumption. The reason may be that the time-lapse image covers a small part of one well on petri-dish, and this small area contains local structures. But the simulation will start from random distribution, otherwise it would be impossible for simulations of 5FU, hypoxia and combination group start from the same condition, and thus the difference of structure cannot be compared.

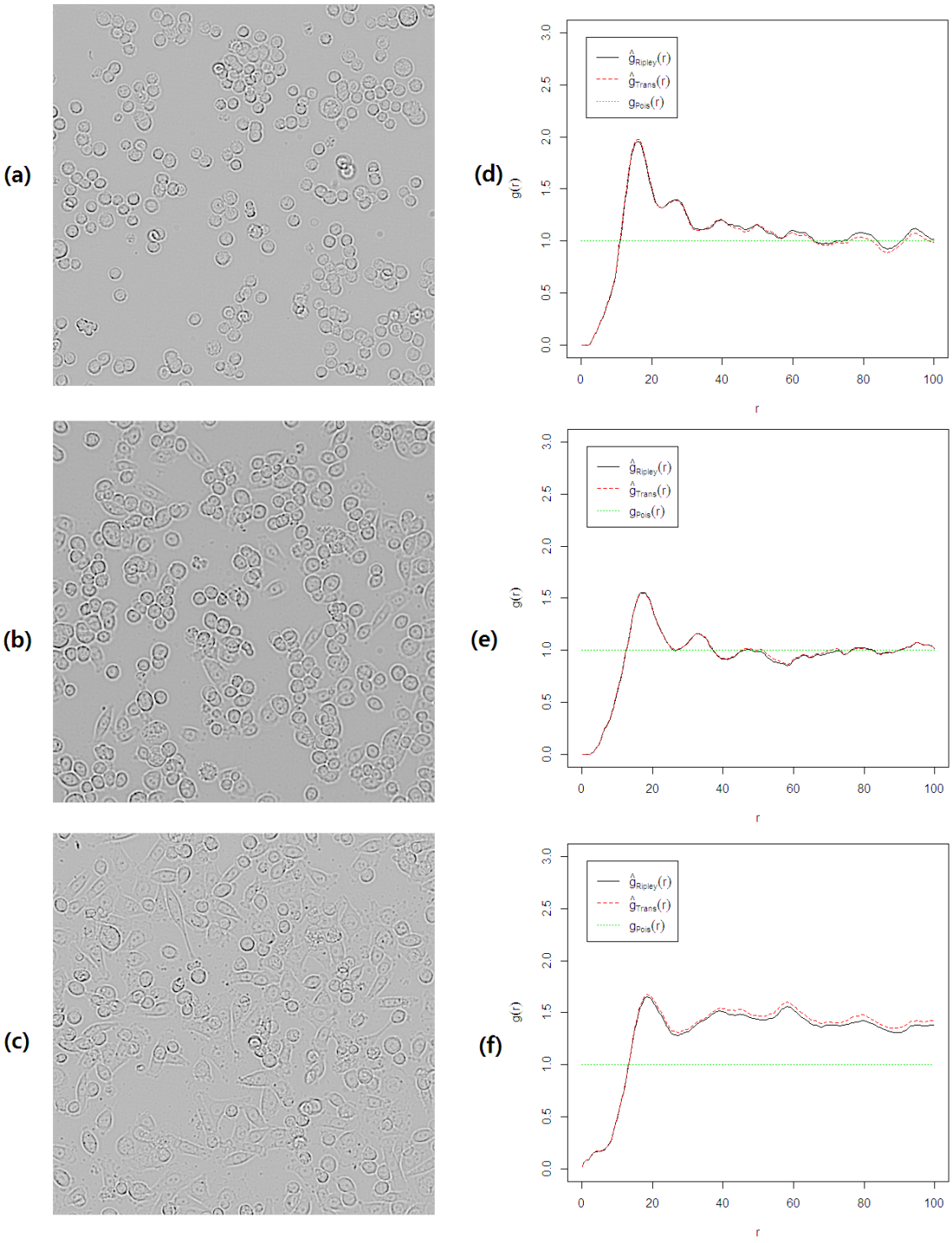


Figure 99: Time-lapse image and pairwise correlation curve of 5FU group of experiment. (a) is the time-lapse image of starting point of experiment, (d) is pairwise correlation function of (a); (b) is the time-lapse image of middle point of experiment, (e) is pairwise correlation function of (b); (c) is the time-lapse image of the ending point of experiment, (f) is pairwise correlation function of (c).

In the middle of experiment (Figure 99, time-lapse image (b) and pairwise correlation function (e)), the peak is also at distance=20 but with a reduced value, which means the number of cells in clusters is reduced. There is a second peak around

distance=35, which indicates that from a typical cell, there are groups of neighbours on this distance. Further, a valley appears at distance=60 with a value smaller than 1.0, and this means that cells tend to disperse at that distance.

At the end of experiment (Figure 99, time-lapse image (c) and pairwise correlation function (f)), the peak remains at the same distance. In addition, and in contrast to earlier time points, at all the distances greater than 20, the value of pairwise correlation is higher than 1.0, which means cells in experiment tend to gather on all distance. Figure 99 (c) shows that gaps exist, therefore the cells should not gather at all distance. However numerous cells are distributed on Figure 99 (c) with relative short distance, which is identified as one large group of cells. To display the spatial feature correctly, a circular observation window has to be applied to Figure 99 (c), as shown in Figure 100. Note that because a circular observation window is chosen, the valid range of distance in pairwise correlation function is limited.

Although the valid range of distance in pairwise correlation function is limited, it displays more detail within the range. Compared to Figure 99 (d) and (e), Figure 100 shows a deeper valley near distance=30, and the second peak is more obvious. In addition, there is a third peak near distance=60. In this way Figure 100 indicates the progress of cell cluster, which matches Figure 99 (c).

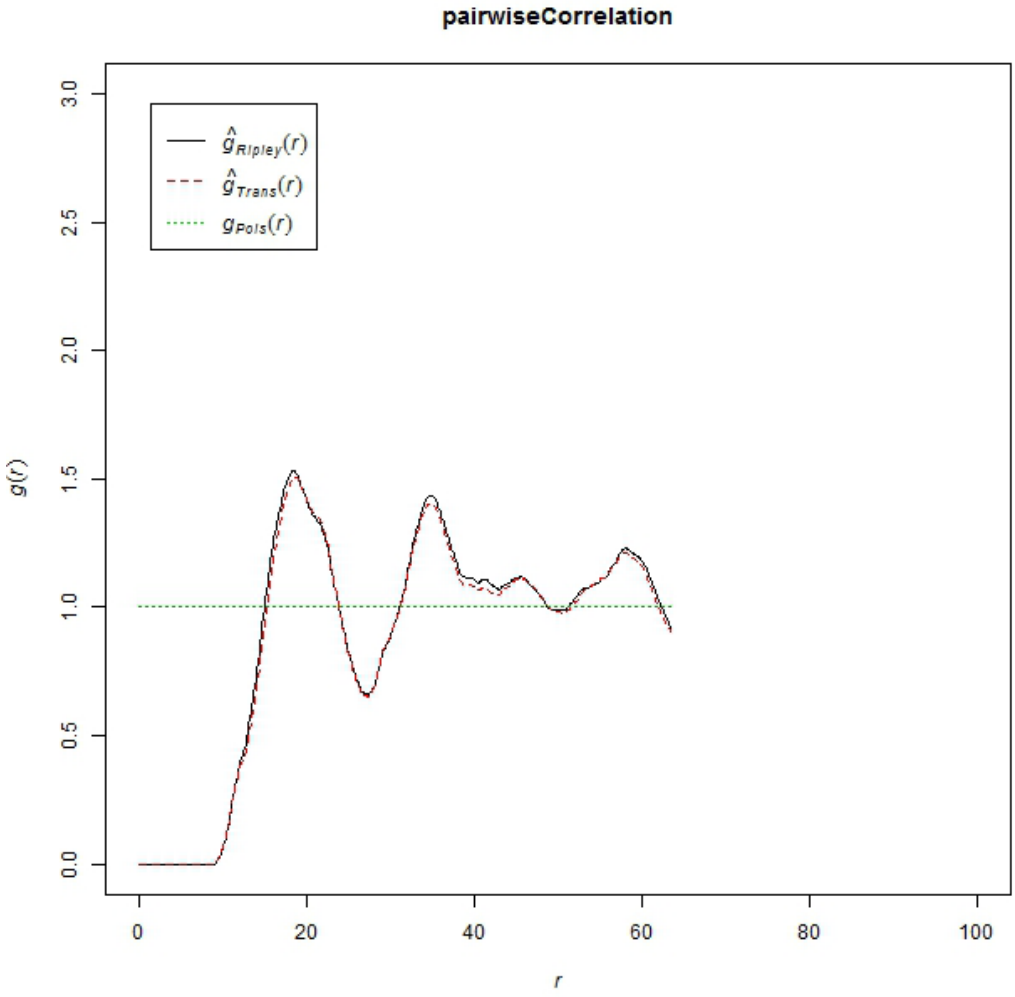


Figure 100: Pairwise correlation function of ending point of experiment with 5FU, observed with a circular window. The range of distance is limited because of the circular window.

With the same population growth setting for the 5FU experiment as discussed in section 5, I run the simulation. Firstly, I set the population in simulation so that the number density of simulation is same with experiment, which are shown in Table 16. Therefore, the simulation does not start with the same population of experiment, but with the same number density.

Table 16: The number density of experimental data and simulation data.

	Number Density
Experiment group: 5FU, starting point	0.00099
Simulation: 5-FU, starting point	0.00094
Experiment group: 5FU, end point	0.00054
Simulation: 5-FU, end point	0.00053

A set of simulation results is shown in Figure 101 From (a) - (c), it is shown that agents are more and more clustered.

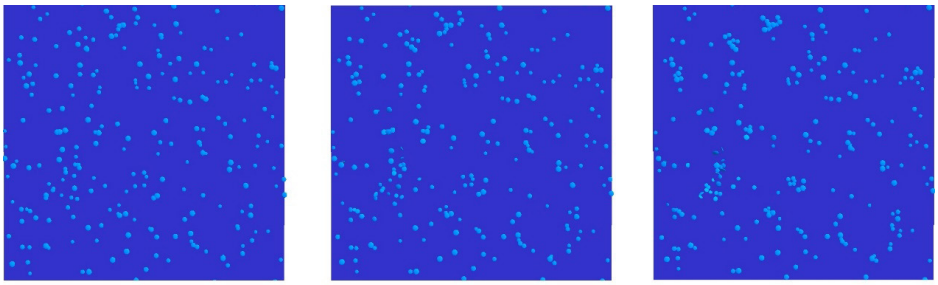


Figure 101: Simulation result of agents under effect of 5FU. The light blue part is the agents, while the dark blue part is background. (a) is the start point of simulation, (b) is the middle point of simulation, and (c) is the end point of simulation.

The pairwise correlation function of simulation data is also estimated. Unlike the experimental data, the first peak of simulation data is near distance=0, while the first peak of experimental data is observed near distance=20, as shown in Figure 102.

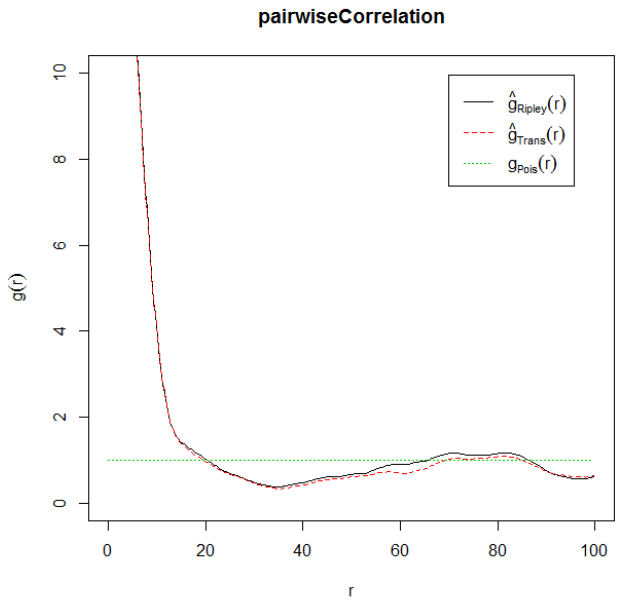


Figure 102: End point of simulation of agents with effect of 5FU. A high peak appears near distance=0.

According to the interpretation of the pairwise correlation function, values greater than 1.0 represent clustering. Analysing the position data of agents in the simulation, I find agents that are closer than 2 units in distance. However, agents should not be so close together. One possible reason is that some agents 'climb on top of' other agents in the simulation. In the cell growth experiments, because the time-lapse images are taken with top-down view, a similar phenomenon cannot be observed. Thus, a filter is introduced to process the simulation data so that agents that are too are discarded. The processed simulation data does not produce the high peak near distance=0, but there is a side effect, which is that the uniform density of simulation data is reduced and then is lower than experimental value. The pairwise correlation function of filtered simulation data is shown in Figure 103, in which the first peak appears near distance=10.

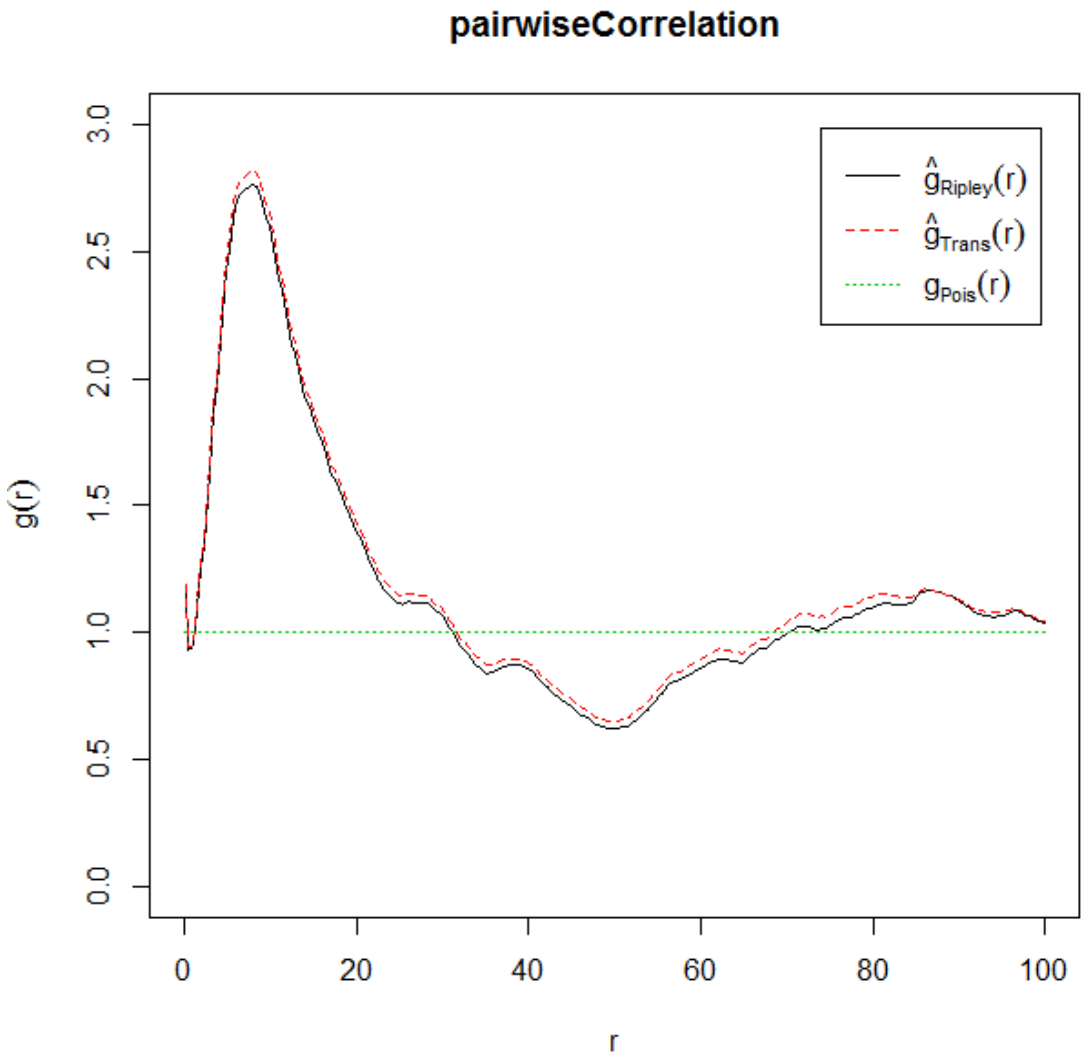


Figure 103: Pairwise correlation curve of filtered simulation data. The first peak appears near distance=10.

Running the 5FU simulation 10 times with this filter, I can draw the average pairwise correlation function, as Figure 104 shows. Note that the range of distance in Figure 104 (c) is reduced because this pairwise correlation function is generated from a circular observation window.

From Figure 104 (d), it is observed that the simulation starts from near-random distribution. I managed to reproduce a similar distribution in simulation in the middle point with the one in experiment, as shown in Figure 104 (b) and (e), both pairwise correlation functions have a peak near distance=20 and a valley near distance=60; plus, the cells and agents distribute randomly on distance>60. From Figure 104 (d) to

Figure 104 (e), more agents gather in cluster, thus in Figure 104 (f) this trend continues and results in larger clusters. Note that to keep the same ratio of x and y-axis of Figure 104 (f) with other figures, the peak of pairwise correlation function exceeds the range of y-axis.

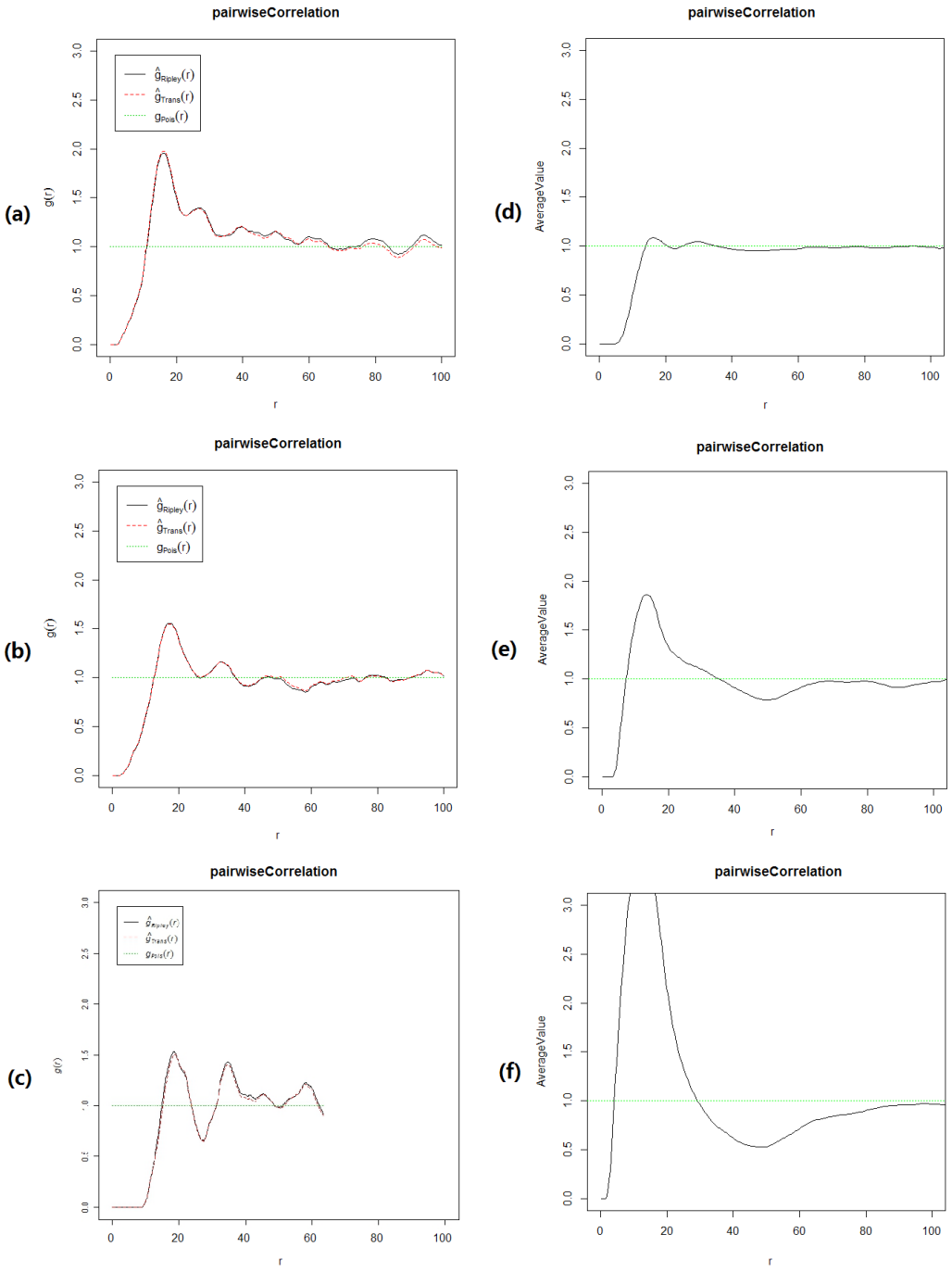


Figure 104: Pairwise correlation function of experimental data (a) - (c), and average pairwise correlation function of 10 simulations (d) - (f) of 5FU group. (a) and (d) are the starting point of experimental and simulation data; (b) and (e) are middle point of experimental and simulation data; and (c) and (f) are end point of experimental and simulation data.

As shown in Figure 104 (a), the experiment does not start from random distribution, it is a different process from Figure 104 (a) to Figure 104 (b) with Figure 104 (d) to Figure 104 (e), which is the reason that the end point of simulation (Figure 104 (f)) has different spatial feature from experimental data (Figure 104 (c)).

Fitting to the hypoxia experiment: The time-lapse images of starting, middle and end point of experiment under effect of hypoxia is shown in Figure 105 (a) - (c). The pairwise correlation function of Figure 105 (a) is shown in Figure 105 (d); the pairwise correlation function of Figure 105 (b) is shown in Figure 105 (e); and the pairwise correlation function of Figure 105 (c) is shown in Figure 105 (f).

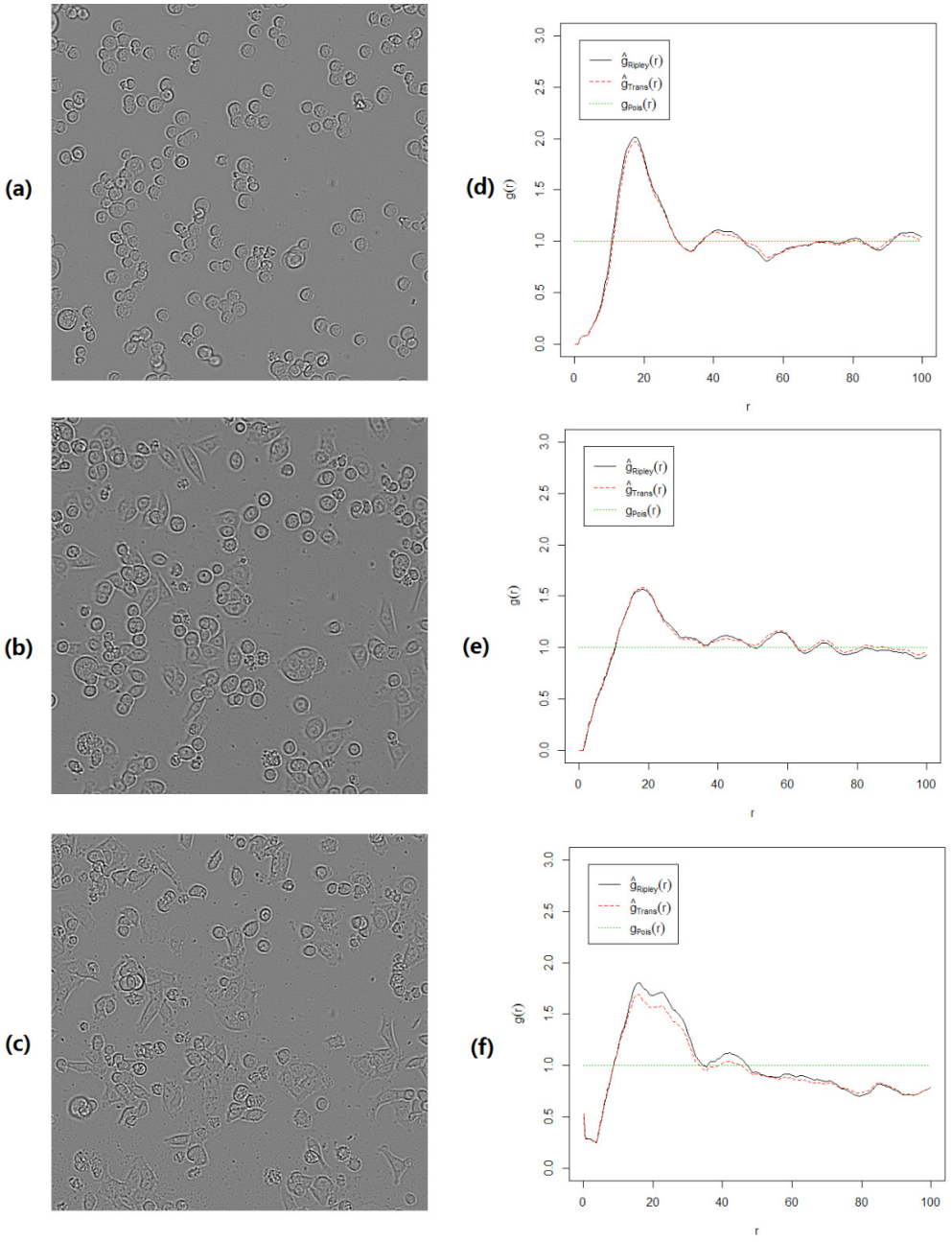


Figure 105: Time-lapse image and pairwise correlation curve of hypoxia group of experiment. (a) is the time-lapse image of starting point of experiment, (d) is pairwise correlation function of (a); (b) is the time-lapse image of middle point of experiment, (e) is pairwise correlation function of (b); (c) is the time-lapse image of the ending point of experiment, (f) is pairwise correlation function of (c).

As shown in Figure 105 (d), the experiment of hypoxia group does not start with random distribution either, like the case in 5FU group, but a repulsive distribution with the first peak near distance=20. Then from Figure 105 (d) to Figure 105 (e), the g(r) value on the first peak is reduced, which means that there are less cells in the clusters. From Figure 105 (e) to Figure 105 (f), the g(r) value of the peak slightly increases and the part of peak above the random-distribution-line (green dotted line) is also increased, which suggests that more cells gathers and form less tight clusters.

The simulation of cell growth and patterns in a hypoxic condition begins with similar number density of the experiment, as shown in Table 17. According to our biological experts, the cells under the effect of hypoxia change their metabolism, which results in slower motility. Thus, in the simulation, and by systematic variation of parameters, the speed of agent is reduced to a quarter of speed of the 5FU simulation.

Table 17: The number density of experiment and simulation of hypoxia group. Note at the start.

	Number Density
Hypoxia experiment (start)	0.00061
Hypoxia simulation (start)	0.00051

The simulation results are shown in Figure 106. As discussed in Section 6.2.2, the simulation starts from random distribution, as shown in Figure 106 (a). The red and green dots are two types of agents, as discussed in Section 5.5.3. Following the growth curve in Section 5.5.3, there are less agents in middle point of simulation (Figure 106 (b)) than the starting point of simulation (Figure 106 (a)), and less agents in the end point (Figure 106 (c)) than the middle point of simulation (Figure 106 (b)).

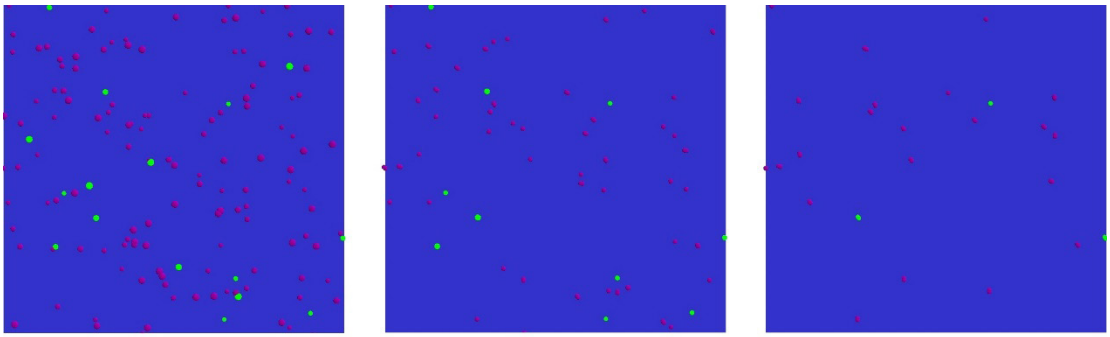


Figure 106: Simulation result of agents under effect of hypoxia. The red and green part are two types of agents, while the dark blue part is background. (a) is the start point of simulation, (b) is the middle point of simulation, and (c) is the end point of simulation.

To avoid the bias, 10 simulations are run, and the average pairwise correlation function is shown in Figure 107 together with experimental function. Figure 107 (a), (b) and (c) are the pairwise correlation function of starting, middle and end point of experimental data, while Figure 107 (d), (e) and (f) are the pairwise correlation function of starting, middle and end point of simulation data. Comparing Figure 107 (a) and (d), it is observed that unlike the experimental data, the simulation starts from a random distribution. Then although Figure 107 (e) is not in the same shape as Figure 107 (b), Figure 107 (e) shows a less tight distribution than Figure 107 (d), which is the same trend from Figure 6.2.3.3 (a) to Figure 107 (b).

The end point of simulation generates a different pairwise correlation function from the end point of experiment. The difference between Figure 107 (b) and (e), and between Figure 107 (c) and (f), may be the difference of number density of cells (agents) in experimental and simulation data, as shown in Table 18: the number density in simulation is significantly smaller than in experiment. As the growth curve in simulation of hypoxia group is calibrated with experimental data, and simulation and experimental data are close in Day 0 and Day 1, within the time range of time-lapse image (20 hours), the simulation and experimental data of number density of cell should be similar. This problem also may be caused by the small area that the time-lapse image covers, in which the number density of cells is biased.

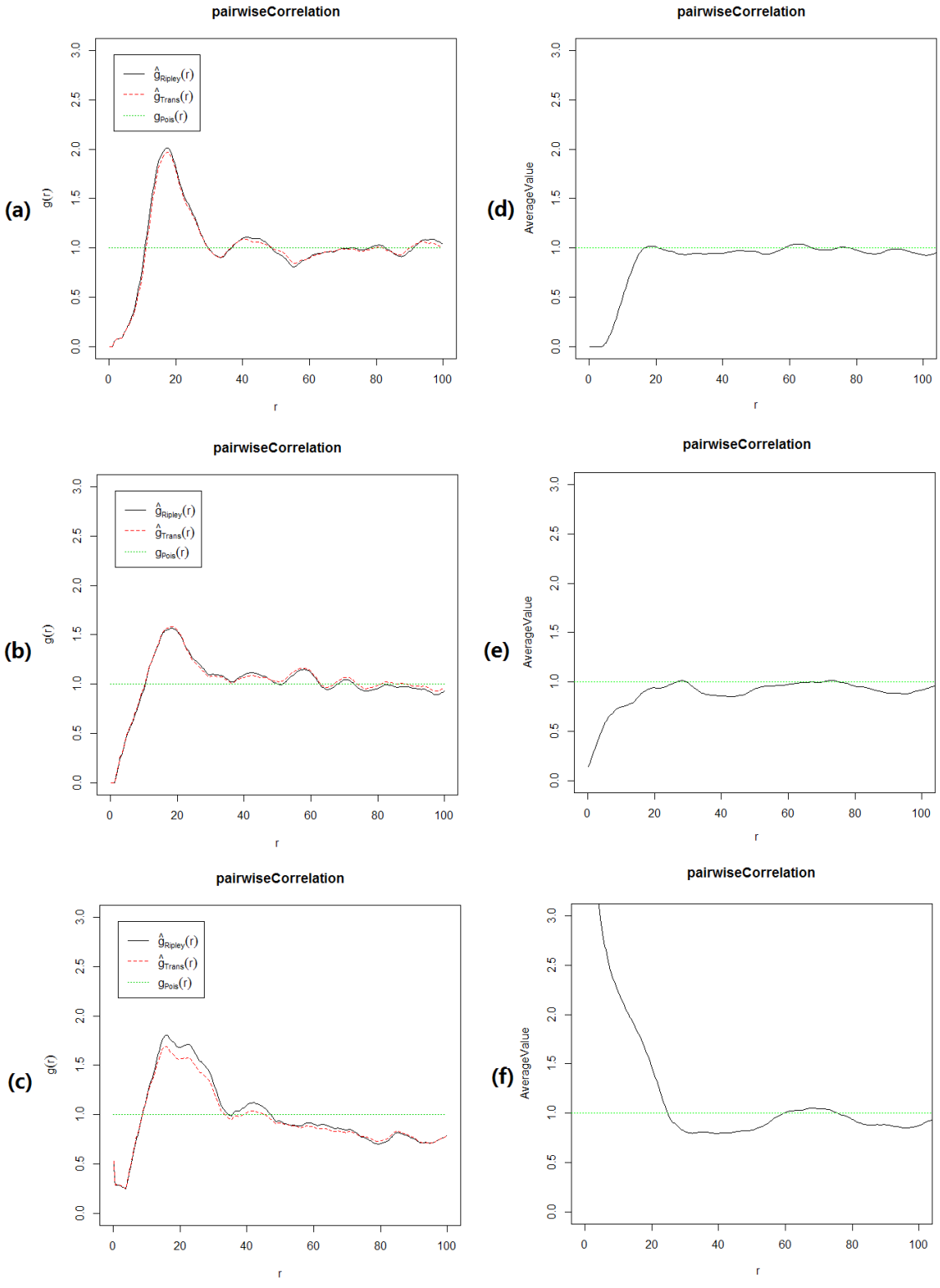


Figure 107: Pairwise correlation function of the starting, middle and end point of experimental data (a) - (c) and average of 10 simulations of hypoxia (d) - (f).

Table 18: Number density of cells (agents) in the starting, middle and end point of experiment and simulation. At the starting point, the number density is similar in experiment and simulation, while it is significantly reduced in simulation data in the middle and end point, which does not match the calibration result of growth curve in Figure 108.

	Number Density
Hypoxia group experiment: start point	0.00061
Hypoxia group simulation: start point	0.00051
Hypoxia group experiment: middle point	0.00062
Hypoxia group simulation: middle point	0.00023
Hypoxia group experiment: end point	0.00052
Hypoxia group simulation: end point	0.00012

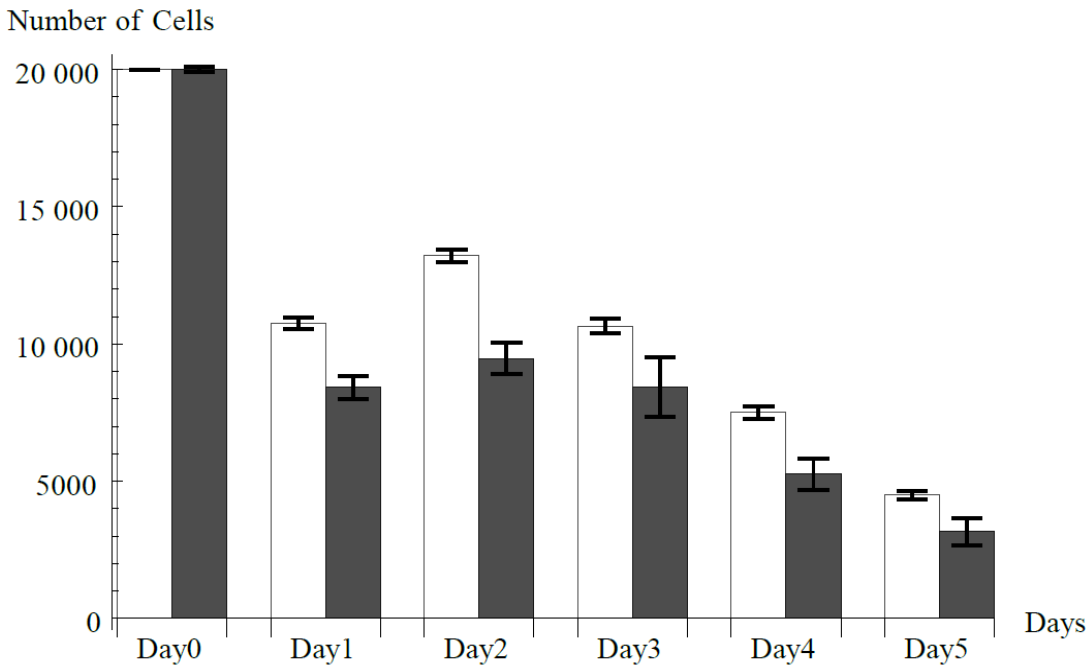


Figure 108: Hypoxia growth curve of my model (grey bars) with two types of agents and experiment result (white bars), in which the simulation and experimental data are very close in Day 0 and Day 1.

Apart from the difference in number density between experimental and simulation data, the pairwise correlation function of the end point of simulation (Figure 107 (f)) shows a different type of distribution with experimental data (Figure 107 (c)). Plus, Figure 107 (f) is not consistent with Figure 107 (d) and (e). As all agents that are closer than 10 units distance are already filtered, there should be no peak before distance=10. I assume this problem is caused by the low total number of the agents.

To test this assumption, 10 simulation results are combined together to generate a new pairwise correlation curve. The method is, keep the first simulation result unchanged; then add the x-coordinate of the second simulation result with width of simulation output so that the second simulation result is considered attached to the first one head to tail, as Figure 106 (top) shows. Similarly, add the x-coordinate of the third simulation result with double width of the simulation output so that the third simulation result is attached with the second one head to tail. Do the same thing to the rest 7 simulation results, then all 10 simulation results are attached head to tail and can be considered as one result with plenty of agents, as Figure 109 (bottom) shows. The pairwise correlation curve of the combined data is shown in Figure 110.

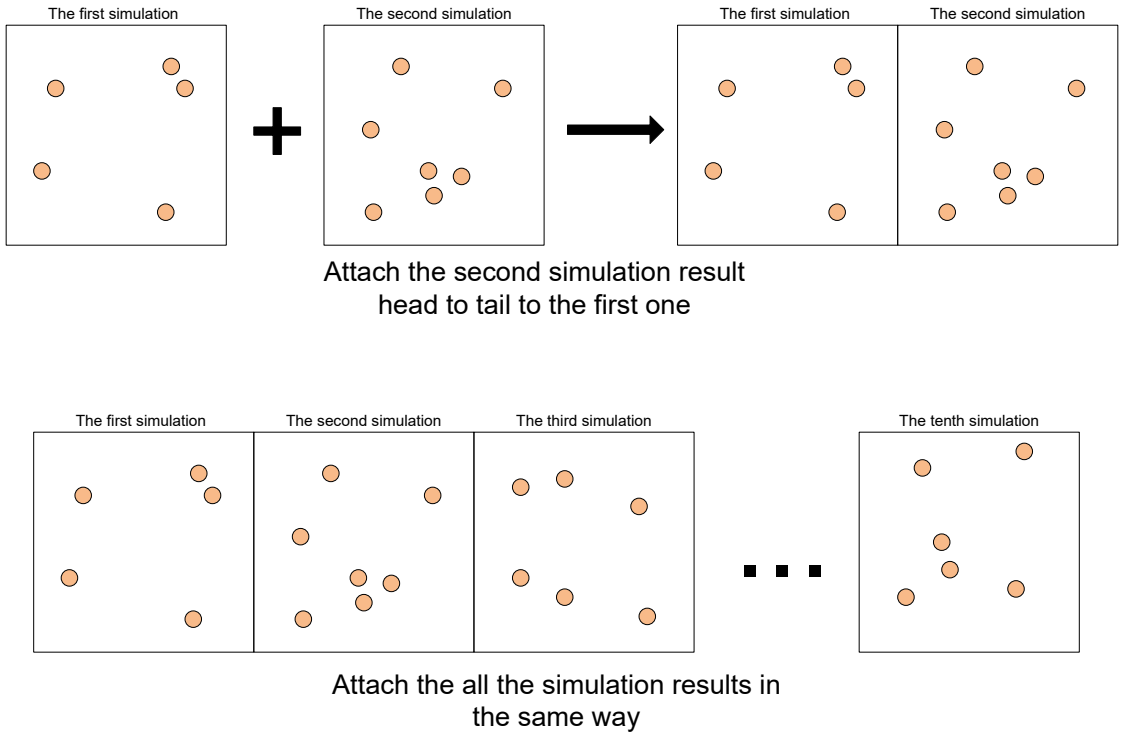


Figure 109: Combine 10 simulation files together by translating 9 of them.

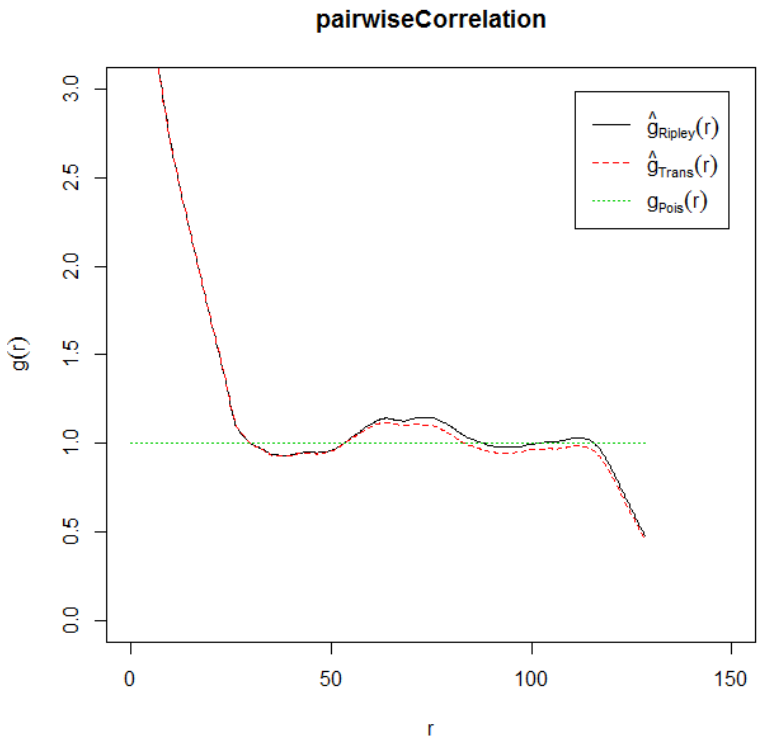


Figure 110: The pairwise correlation curve of combination of 10 simulation output files.

The peak near the origin remains in the pairwise correlation function remains, which indicates that the low total number of agent is not the reason. Thus, the second assumption is made, that the peak near the origin is caused by a low density. Compared to the number density and pairwise correlation curve of control group, I can see that number density around 0.00012 can produce a high peak near the origin in R (Figure 111).

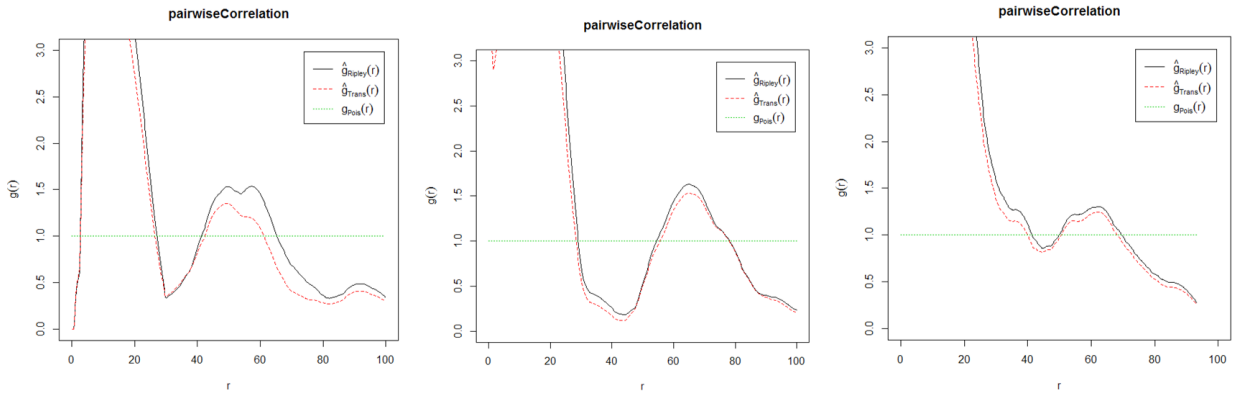


Figure 111: Sample pairwise correlation function of the start (left), middle (middle) and end (right) point of control group growth experiment.

Table 19: Number density of control group experiment.

	Number Density
Start point of control group experiment	0.00012
Middle point of control group experiment	0.00014
End point of control group experiment	0.00013

Comparison against the combined 5FU and hypoxia experiment: Following the approach of Section 5, and in spite of the difficulties faced in calibrating the model with experimental data of hypoxia group, I test the derived parameter setting for each of the 5FU and hypoxia simulations by using the average value for parameters of each simulation group is used for the simulation of combination of 5FU and hypoxia.

The time-lapse image and pairwise correlation of starting, middle and end point of experiment with combination of 5FU and hypoxia is shown in Figure 112.

Figure 112 (a) is the time-lapse image of the starting point of experiment, and 112 (d) is the pairwise correlation function of Figure 112 (a). Similar with the end point of 5FU group, as shown in Figure 113, the pairwise correlation function in 112 (d) indicates one single cluster with all the cells. In this case a circular observation window is needed to correctly display spatial feature of Figure 112 (a). The pairwise correlation function with circular observation window is shown in Figure 114.

The pairwise correlation function with circular observation window, as shown in Figure 114, indicates a peak near distance=20 and a valley near distance=30. The peak also exists in the end point of the combination group on similar distance, with value of *g(r)* slightly reduced, which means less cells are clustered. There is also a valley in the end point at similar distance.

Figure 115 shows the starting, middle and end point of simulation under effect of 5FU and hypoxia. Same with the simulation of hypoxia group, there are two types of agents in the simulation of combination group, which are represented by red and green dots. Comparing to the simulation of hypoxia group, the 5FU+hypoxia group contains less agents in the middle (Figure 115 (b)) and end point (Figure 115 (c)) (Figures 15-18).

The combination simulation is also run 10 times, and the output position of agents is also filtered according to the distance. For completeness here I show the average pairwise correlation function for each of 5FU, hypoxia and the 5FU-hypoxia combination simulations are shown in Figure 112, Figure 113 and Figure 114.

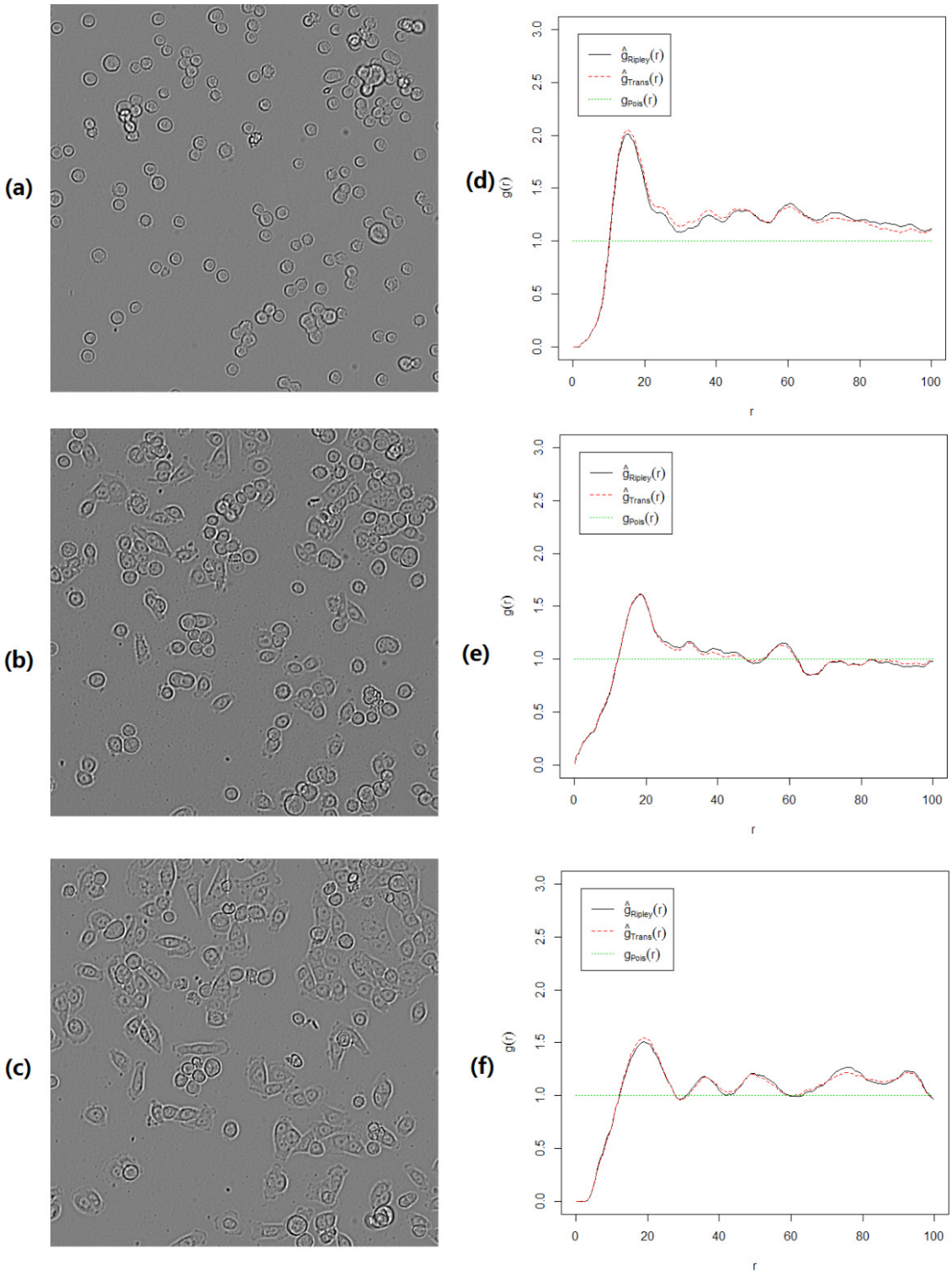


Figure 112: Time-lapse image and pairwise correlation curve of 5FU+hypoxia group of experiment. (a) is the time-lapse image of starting point of experiment, (d) is pairwise correlation function of (a); (b) is the time-lapse image of middle point of experiment, (e) is pairwise correlation function of (b); (c) is the time-lapse image of the ending point of experiment, (f) is pairwise correlation function of (c).

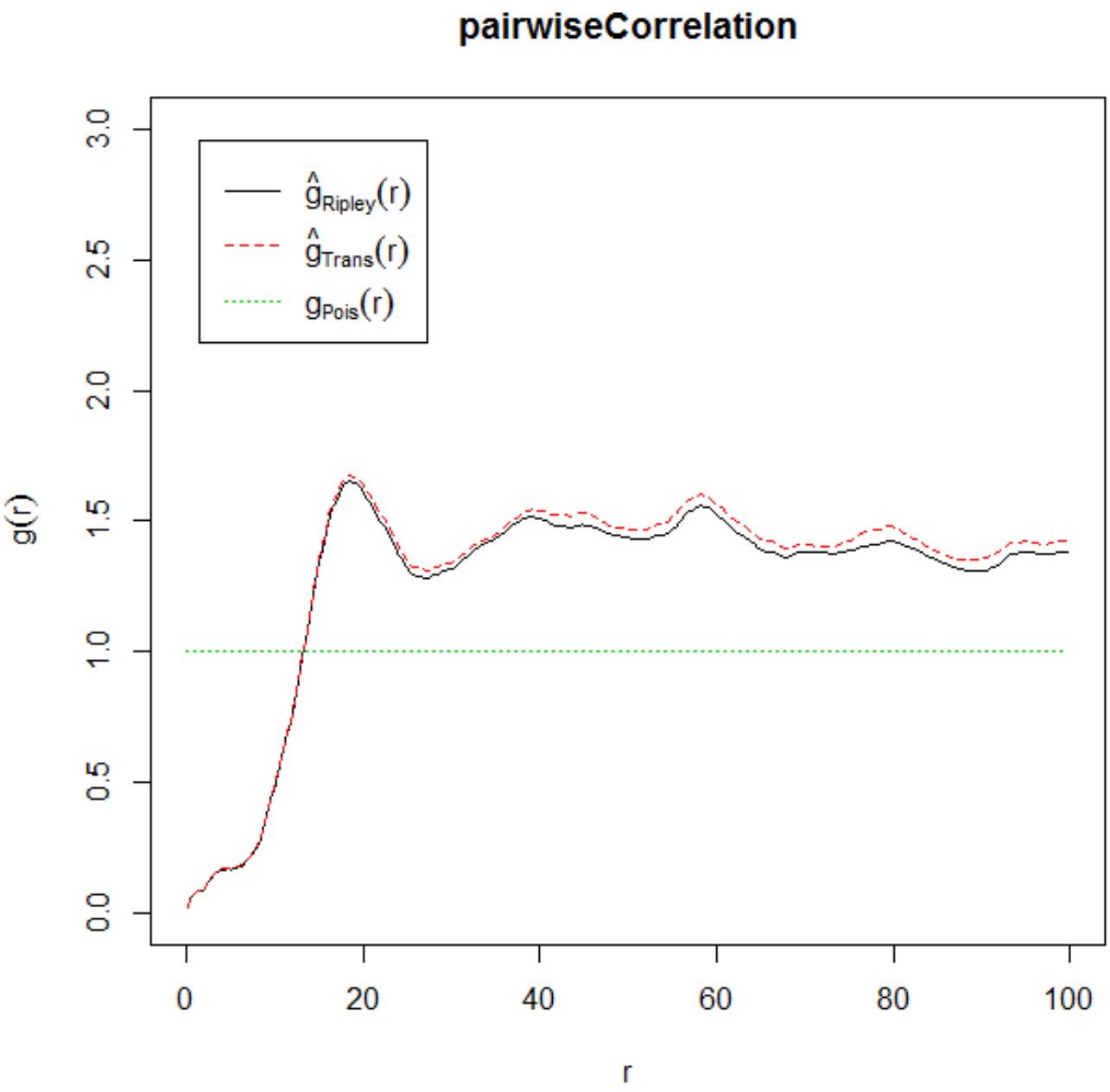


Figure 113: Pairwise correlation function of the end point of experiment with effect of 5FU. From distance=20 to distance=100, the value of *g(r)* is always larger than 1, which means all the cells form one large cluster. This means a circular observation window is needed.

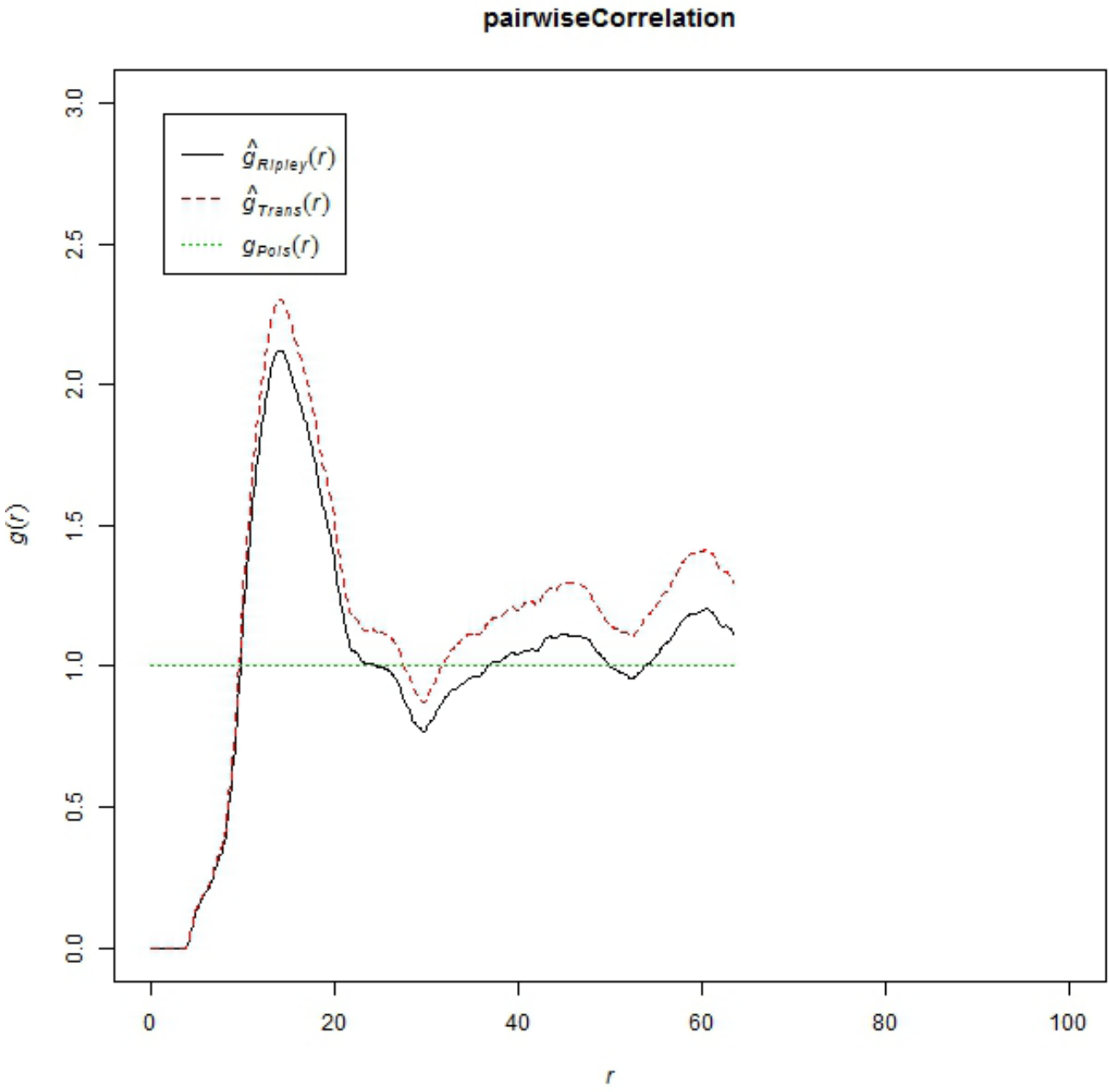


Figure 114: Pairwise correlation function of the start points of experiment with combination of 5FU and hypoxia. With a circular observation window, the range of distance is limited, however the spatial feature within the range is displayed with peak and valley.

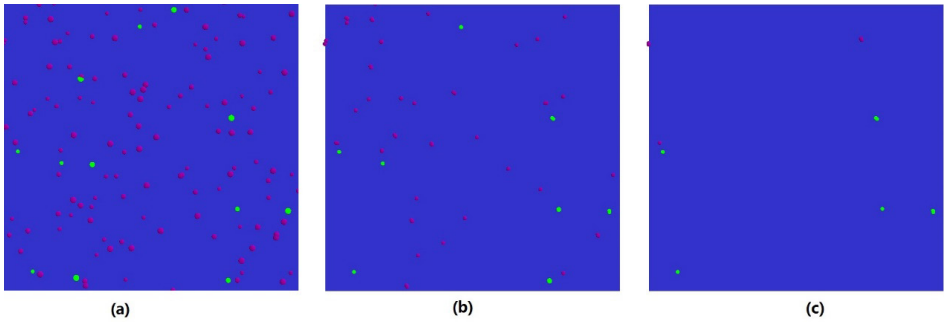


Figure 115: Simulation result of agents under effect of 5FU+hypoxia. The red and green part are two types of agents, while the dark blue part is background. (a) is the start point of simulation, (b) is the middle point of simulation, and (c) is the end point of simulation.

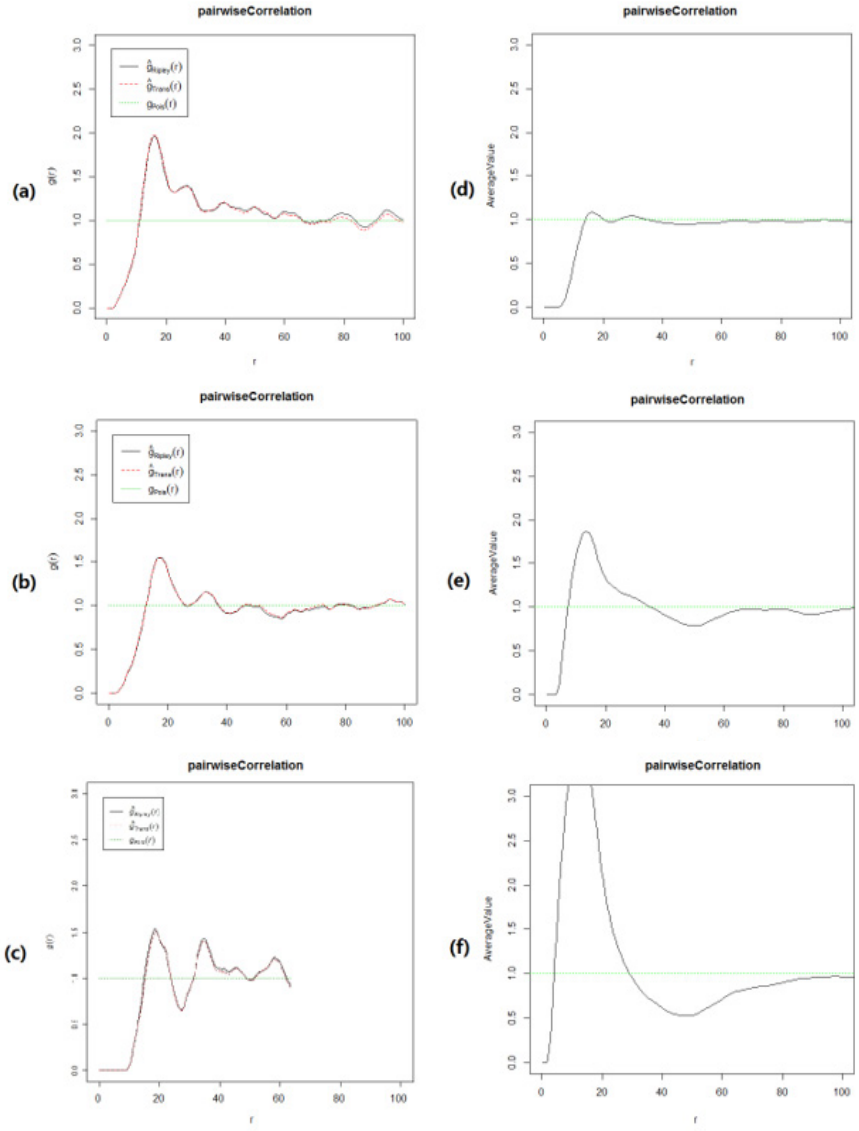


Figure 116: Pairwise correlation function of experimental data (a) - (c), and average pairwise correlation function of 10 simulations (d) - (f) of 5FU group. (a) and (d) are the starting point of experimental and simulation data; (b) and (e) are middle point of experimental and simulation data; and (c) and (f) are end point of experimental and simulation data.

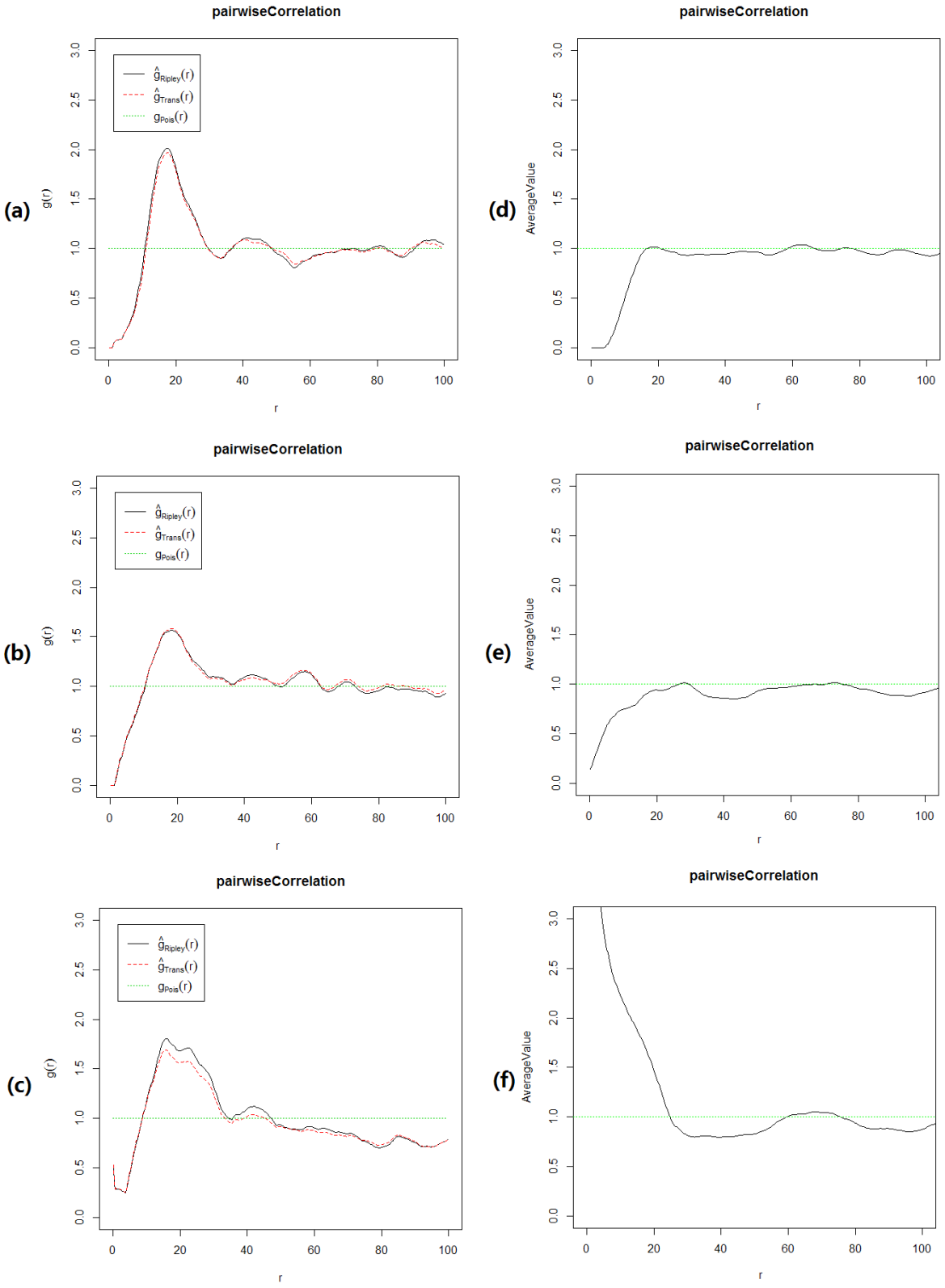


Figure 117: Pairwise correlation function of the starting, middle and end point of experimental data (a) - (c) and average of 10 simulations of hypoxia (d) - (f).

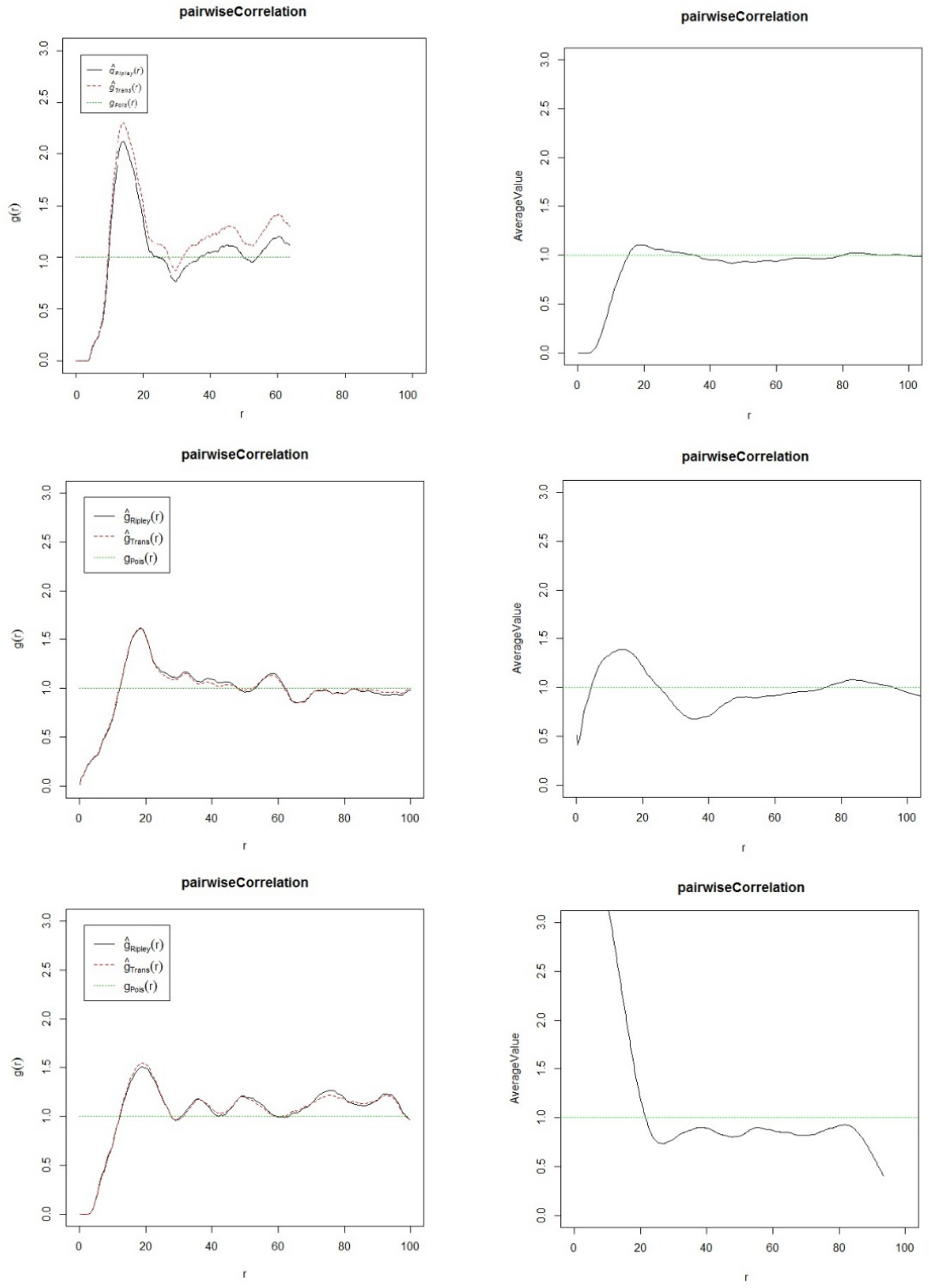


Figure 118: Pairwise correlation function of experimental data (a) - (c), and average pairwise correlation function of 10 simulations (d) - (f) of combination group (5FU+hypoxia). (a) and (d) are the starting point of experimental and simulation data; (b) and (e) are middle point of experimental and simulation data; and (c) and (f) are end point of experimental and simulation data.

From Figure 18, it can be observed that the simulation starts from random distribution, while the experimental data starts from a clustered distribution. Although the number density at the end point of simulation is too low to give out useful information, the simulation is able to reproduce the spatial feature of a peak near distance=20 and a valley near distance = 30 in the middle point.

In all three starting points, the simulation groups start from a random distribution. In the middle point of simulations, the peak of the combination simulation appears in a similar position to the 5FU simulation although with a lower peak value and can be considered as average value of 5FU and hypoxia simulations. At the end point, the combination simulation contains fewer agents than the hypoxia simulation, and the peak near the origin appears. In the end point of all three simulations, the curves are under 1.0 on distance>20. As shown in initial test simulations discussed in section 6.2.1, in the combination simulation the agents spread over a shorter range because of low number density. From the comparison of three simulation and experiment groups, it is shown that the simulation result of 5-FU group produces the most similar pairwise correlation curve with experiment; the hypoxia simulation produces the least similar curve; and the combination group shows a result that is more similar than hypoxia group but less similar than 5-FU group.

Conclusion

In section 6, I first explored the features of experimental output (time-lapse image data) and chose the pairwise correlation curve as the tool to represent the distribution of cells. Considering the experimental output as the result of cell interaction, I then run simulations with various parameter settings. The resulting pairwise correlations from simulations are presented and compared with the corresponding experimental curves. As expected, by changing simulation parameters, pairwise correlation curves with different shapes are produced. Also as expected, the range that adhesion force takes effect does affect the result pattern. I did not expect the speed of agents to be a factor in agent distribution, however after discussing with biological experts I reduced the velocity of agents as effect of drug in simulation and had a good fit with experimental result (pairwise correlation curve). Finally, the balancing point (where the contact force has same value and opposite direction of adhesion force) does not make much difference to the pairwise correlation curve. A possible reason is that the balancing point works with agents of exactly the same shape, and with agents of various cell cycles the size of each agent varies, which introduces a disturbance to the distance on which agents balance.

Besides, only when I reached the simulation phase did it become evident that the uniform density in the time-lapse images of 5FU, hypoxia and combination group in experiment is lower than expected from the population growth experiments. If I are able to repeat the growth experiment under improved conditions, I may be able to explain this.

Section 7: Conclusion and Future Work

Research Hypotheses

The aim of this chapter was to develop an individual-based model of populations cells and inter-cell interactions. The model focused on characterising the physical properties of cells and how these drive inter-cell interaction dynamics. The biological behaviour of cells was captured in terms of changes to the physical property of cells and assumes that the physical model can be used to predict biological phenomenon. This assumption led to the following hypotheses:

- An *in silico* physical model can reproduce important behaviours of experimental systems of human cells.
- The model can be parameterized with biologically interpretable values to model differences arising from experimental conditions for the same biological system.
- The characterisation of cell distributions by spatial statistics enables prediction of biological phenomena using the physical model.

To test these hypotheses, a physics-based model was developed, as detailed in Section 3, underpinned by a set of assumptions emerging from the review of literature in Section 2.

The first biological system to be studied was the early-stage formation of vascular structure by endothelial cells. This system is tested by *in silico* simulations in Section 4, in which ellipsoidal agents representing cells are placed at random positions with random directions on a substrate. In this model, endothelial cells do not divide or die in this process. Note that although no tubular structure is formed the connected agents are considered a vascular structure.

The simulation result is considered as a prediction and is compared with *in silico* simulation of another model. With spatial

statistics analysis, the typical distance between pair of agents is considered the typical size of holes in net-shaped structure. A key observation was that agents could indeed self-organize into vascular-like structures simply by manipulating both cell density and physical parameters. This work was on track to explore and ultimately test Hypotheses 1 and 2, with some early confirmation of Hypothesis 1 established. However, due to lack of data streams for *in vitro* experiments of early stage of vascular formation, as noted in Section 1, the physical model was repurposed to describe the interactions among tumour cells in Section 4.

Through a structured analysis of the physical properties of cells in both vascular formation and cancer cell interactions, I assumed that both biological systems contained similar cell-cell physical interaction. The vascular formation model was then systematically converted for use with cancer cells in Section 4 in order to fully explore Hypotheses 1 to 3.

The major difference between the early-stage study of vascular formation and cancer cell interactions (over a longer time period) is that cells may divide or die in cancer cell population growth experiments. Thus, a cell cycle is added to model, in which the parameter to determine if an agent is to divide or die is used to reflect the population growth of cancer cells under different environment conditions. The cell cycle model is discussed in Section 5 and is based on changes to the physical parameters of the cell. The predictions of model and experiment result were compared by population growth curves for a control condition (Hypothesis 1). The model was then used to predict population growth curves in a range of different experimental conditions (Hypothesis 2).

Two parameters in the model are considered to have effect on pairwise correlation curve, and the discussion is in Section 6.2.1. In consideration of Hypothesis 3, the physical model (with cell cycle part) is used to predict the spatial distribution of cells in *in vitro* experiments under different experimental conditions in Sections 6.2.2, 6.2.3 and 6.2.4.

Conclusion

The Hypothesis 3 is proved in Section 6. The characterisation of both cell distribution from experiments and predicted distribution made by the model is demonstrated by pairwise correlation curve. Figure 119 shows the time-lapse image from the experiment and corresponding pairwise correlation curve. I can see that the pairwise correlation curves of 5-FU, hypoxia and combination (experimental) group show both common and specific features. In all three groups, the first peak of pairwise correlation curves appears, around distance=20. As I explain the first peak relative with typical size of cell, and in three experiments the same type of cell is used, this result is reasonable.

From experimental images, I can see that the 5-FU group shows clusters which are looser, while the hypoxia group contains compact clusters. With data from 5-FU experimental group, the first peak of pairwise correlation curve has lower value than the hypoxia group. The experimental images also show much closer clusters in 5-FU image, while the hypoxia image contains sparse clusters. From corresponding pairwise correlation curve I can see that after the first peak the 5-FU curve stays above the line $g = 1$ which indicates random distribution, while the hypoxia curve falls below $g = 1$ after the first peak.

These features of cell distribution are used to calibrate the physical model. When the calibrated model produces predictions, they are also analysed by pairwise correlation function to be compared with experimental data. (H 3: The characterisation of cell distributions by spatial statistics enables prediction of biological phenomena using the physical model).

Hypothesis 2 is proved in Sections 5 and 6. In Section 5, the model is parameterized to predict the number of cells in cancer cell growth experiments. The key parameter in the model is the chance that agents divide into two agents when they finish the cell cycle. The control group provided data to calibrate the model as a baseline. Then the effect of 5-FU and hypoxia on chance of division was modelled and added to physical model separately. Thus, the physical model with 5-FU or hypoxia mechanism contained parameter that has biological meaning. From Figure 120 (a)-(c) I can see that as the baseline, the prediction of control group shows similar trend of increase/decrease of number of cells; the prediction of 5-FU group differs with experimental data of Day1 and Day2 but has better match with data of Day3, 4, and 5; the prediction of hypoxia group differs with data of Day2 but shows good match with experimental data of Day1 and Day3 and similar trend of increase/decrease of Day4 and 5. I also tested the parameters by adding both mechanisms of 5-FU and hypoxia to physical model together and comparing the prediction with corresponding *in vitro* experiment (Figure 120 (d)), and the prediction shows similar trend of increase/decrease with experimental data. Therefore, the model with parameters of different value can predict number of cells of same type under different condition.

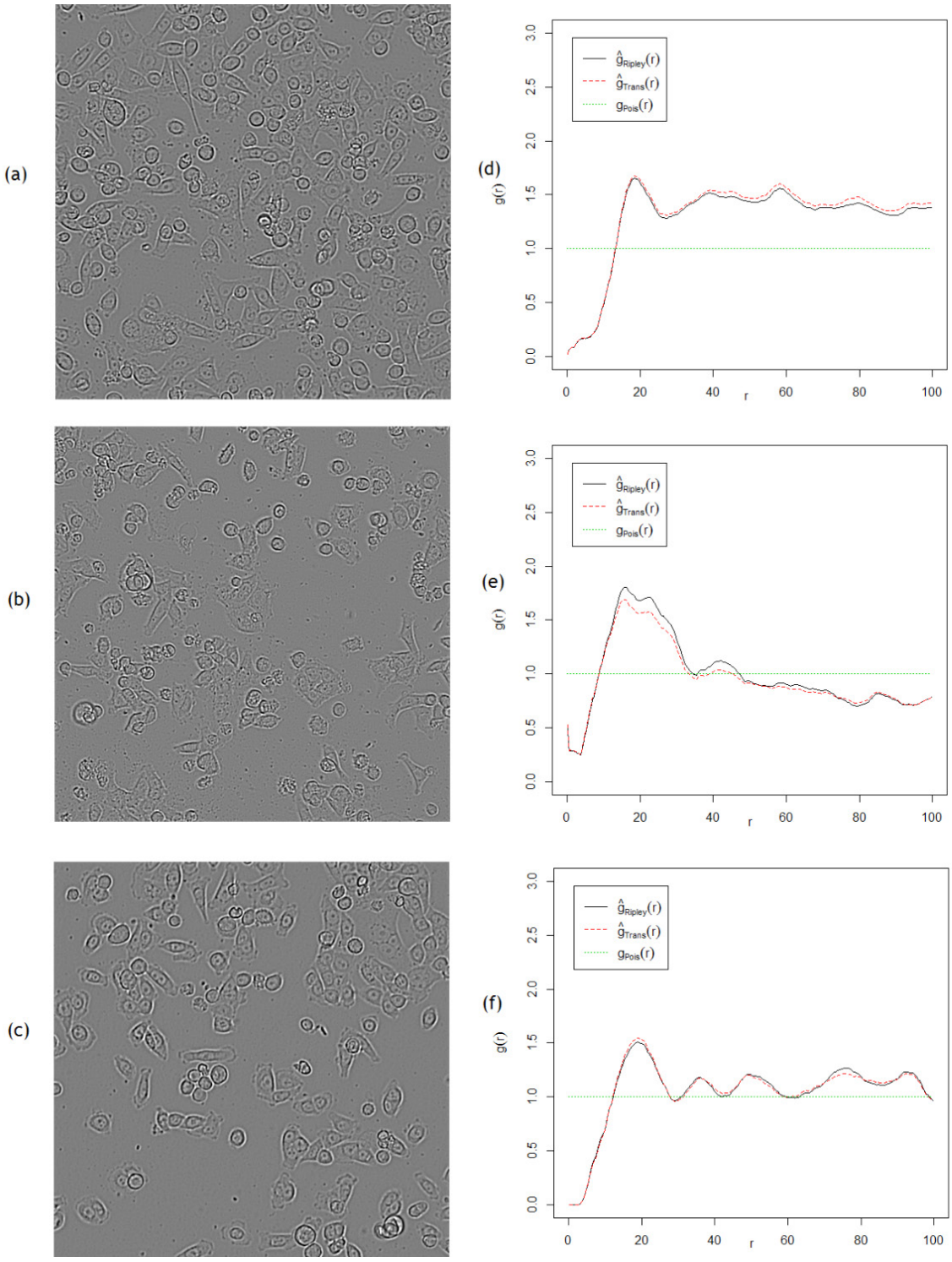


Figure 119: (a) last image of cancer cell growth experiment with 5-FU; (b) last image of cancer cell growth experiment in hypoxic condition; (c) last image of cancer cell growth experiment with 5-FU in hypoxic condition; (d) is the pairwise correlation curve of (a); (e) is the pairwise correlation curve of (b); (f) is the pairwise correlation curve of (c).

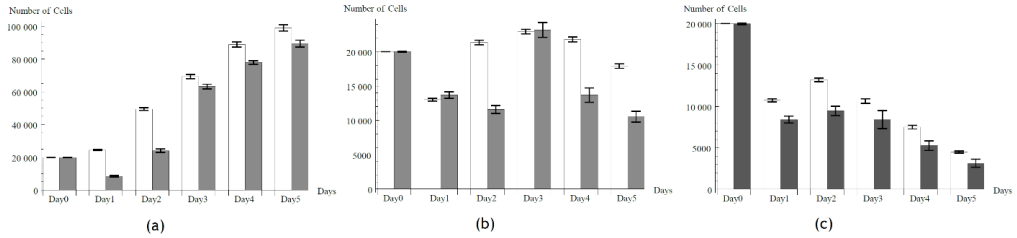


Figure 120: (a): Experimental and simulation growth curves of 5-FU group; (b): Experimental and simulation growth curves of hypoxia group; (c): Experimental and simulation growth curves of combination of 5-FU and hypoxia group. Greyscale bars are experimental data, and white bars are simulation result.

In Section 6, with mechanisms modelled in Section 5, the physical model is used to predict the distribution of cancer cells in *in vitro* experiment. The range of adhesion force, the point where adhesion force and contact force balance and the speed that agents move are considered as three key parameters. As the image from control group did not reflect the number of cells in experiment, the 5-FU group is used as the baseline (discussed in Section 6.2.4). By changing values of the three key parameters, the model predicted the distribution of cells as shown in Figure 121.

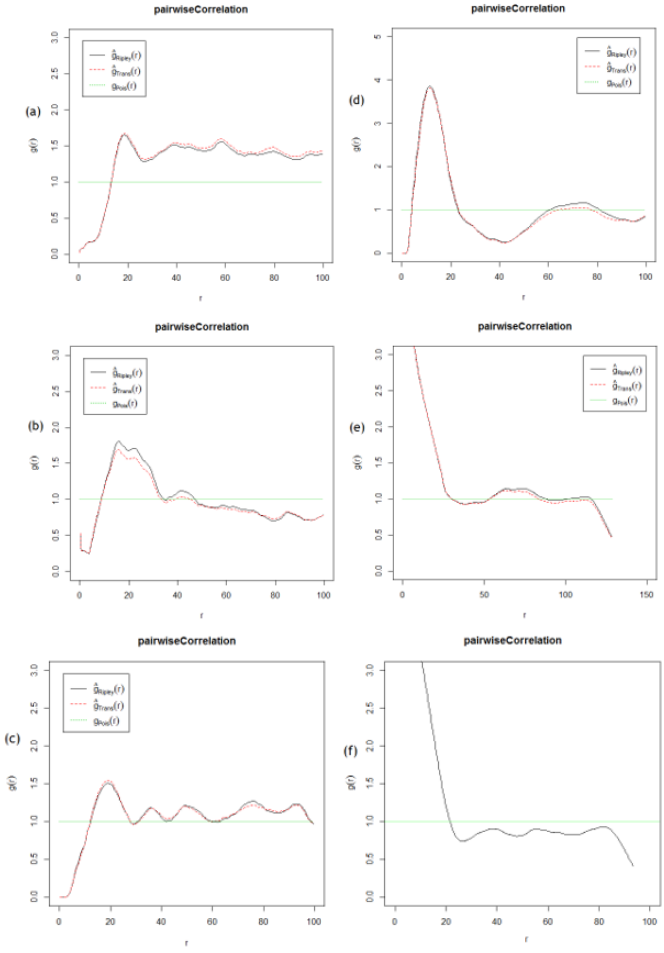


Figure 121: (a) pairwise correlation curve of last image of cancer cell growth experiment with 5-FU; (b) pairwise correlation curve of last image of cancer cell growth experiment in hypoxic condition; (c) pairwise correlation curve of last image of cancer cell growth experiment with 5-FU in hypoxic condition; (d) pairwise correlation curve of simulation of cancer cell growth with 5-FU (Note that to show actual value of peak, the scale of this figure is different from others); (e) pairwise correlation curve of simulation of cancer cell growth experiment in hypoxic condition; (f) pairwise correlation curve of simulation of cancer cell growth experiment with 5-FU in hypoxic condition.

Comparing the pairwise correlation curves produced by simulation and experimental data, I can see that the first peak in both simulation and experimental curves appears around distance = 20. The prediction of 5-FU group shows a higher first peak; although the predicted curve is lower than experimental curve after the first peak, the predicted curves show increased trend of gathering of agents on long distance (distance>60). The prediction of hypoxia group has a higher first peak and lower value on longer distance, which reflects a different feature of distribution. (H2: The model can be parameterized with biologically interpretable values to model differences arising from experimental conditions for the same biological system).

In Section 4, the prediction of vascular formation was also explored. Due to lack of dataflow, the prediction was not calibrated with any *in vitro* experiment data. However, the *in silico* simulation still shows the emergent behaviour of randomly placed identical agents. The agents connect with each other alone the longest axis and form net-shaped structure. By changing the number density of agents, the physical model can be used to generate thick but unstable structures or thin but stable structure, as shown in Figure 122.

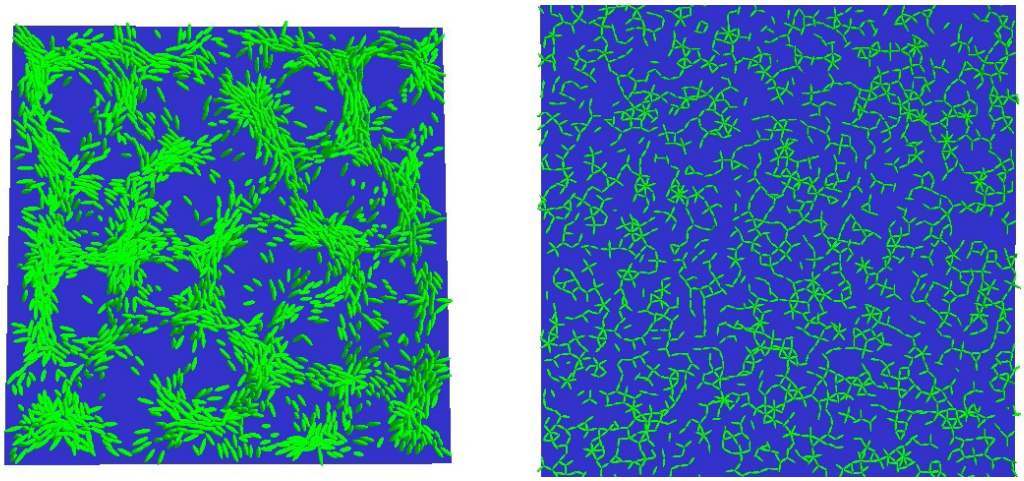


Figure 122: Thick and thin net-shaped structure.

In Section 5 the physical model successfully reproduced the growth curve of *in vitro* growth experiment of cancer cells under various conditions. In Section 6 the model is used to reproduce the distribution of cells in *in vitro* growth experiment of cancer cells under various conditions. Therefore, Hypothesis 1 is proved by Sections 4, 5 and 6. (H1: An *in silico* physical model can reproduce important behaviours of experimental systems of human cells).

Future work

As discussed in Section 2, following the process-based modeling approach the mechanism of unknown part of biology phenomenon can be assumed and modeled, and various assumptions can be made for the same phenomenon, such as cancer cell growth in hypoxic environment in Section 5. By comparing the *in silico* result of model, the assumptions can be tested and compared.

This physical model has been proved to be able to predict the vessel formation and be reused to predict the spatial distribution of cancer cells in *in vitro* growth experiment. With biological data such as cell size, cell shape, etc. this model is able to predict behaviours of various cell types, which requires physical and biological measurements from experiments. In Section 6.2.3 it was discussed about the problem of cell recognition in CellProfiler. In CellProfiler, the cells within clusters failed to be recognized, instead, the cluster was identified as a single cell. Therefore, errors in both the data about cell positions and number of cells are introduced. In future, with more accurate methods to identify and track cells, such as [102], the model can be better calibrated and produce better predictions.

With support of better image process method, the model can also be used to predict more complex phenomenon. A straightforward extension of existing cancer cell growth experiment is that, instead of planting cancer cells on petri-dish to form 2-dementional pattern, the cancer cells can be planted in centre of gel to form 3-dementional pattern (spheroids). There are also problems that may be brought about prediction of spheroidal growth. First, it is hard to directly analysis the pattern formed by spheroidal growth. Thus, instead of directly comparing the pattern and prediction, the spheroidal cancer tissue needs to be sliced. Second, in Section 5 to model the cancer growth in hypoxic condition, a second cell type is introduced to reflect the effect of hypoxia on both cell growth rate and length of cell cycle. In spheroidal growth the cancer cells in the centre of spheroids are

hypoxic, while the cells on surface remain normal. This demands a method in spatial statistics to describe the distribution of mixed types of cells, including the spatial distribution of each type and the spatial relationship between types. The marked point process ([103] Analysis of spatial correlations in marked point processes (ppt)) could be a useful tool because it is to describe the spatial distribution of mixed types of particles. Similar to the pairwise correlation function, marked correlation function produces curves that present the spatial distribution in graph [105-125].

As shown in literature review (Section 2), cancer is a complex system. Apart from predicting two cell types, as an agent-based model, it is relatively easy for this model to predict more complex systems, which can be mixed types of cancer cells, same type of cancer cells in different stage of mutation, or mixed cancer cells and vascular structure. Combining with the advanced cell tracking and identification method, and proper spatial statistic tools, this model may be used to predict complex, tissue-level phenomenon[126-139].

Ultimately, this model will be able to predict the behaviours of mixed type of cancer cells growing in 3-dimension with vascular structure, showing how spheroidal growth of cancer cells produces hypoxic condition in inner layers of cells and how vascular structure can reduce the level of hypoxia.

References

1. Bown J, Deeni Y, Savage A, Sampson A, Khalil HS, Idowu M, Zhelev N, Li Y (2015) EBTNA Book (draft).
2. Hanahan D, Weinberg RA (2000) The hallmarks of cancer. Cell 100(1): 57-70.
3. Hanahan D, Weinberg RA (2011) Hallmarks of cancer: the next generation. Cell 144(5): 646-674.
4. Luo J, Solimini NL, Elledge SJ (2009) Principles of cancer therapy: oncogene and non-oncogene addiction. Cell 136(5): 823-837.
5. Goltsov A, Faratian D, Langdon SP, Mullen P, Harrison DJ, et al. (2012) Features of the reversible sensitivity-resistance transition in PI3K/PTEN/AKT signalling network after HER2 inhibition. Cell Signal 24(2): 493-504.
6. Toettcher JE, Loewer A, Ostheimer GJ, Yaffe MB, Tidor B, et al. (2009) Distinct mechanisms act in concert to mediate cell cycle arrest. Proc Natl Acad Sci U S A 106(3): 785-790.
7. Muller PA, Vousden KH (2013) p53 mutations in cancer. Nat Cell Biol 15(1): 2-8.
8. Nevins JR (2001) The Rb/E2F pathway and cancer. Human molecular genetics 10(7): 699-703.
9. Vousden KH, Prives C (2009) Blinded by the light: the growing complexity of p53. Cell 137(3): 413-431.
10. Elmore S (2007) Apoptosis: a review of programmed cell death. Toxicol Pathol 35(4): 495-516.
11. Ouyang L, Shi Z, Zhao S, Wang FT, Zhou TT, et al. (2012) Programmed cell death pathways in cancer: a review of apoptosis, autophagy and programmed necrosis. Cell Prolif 45(6): 487-498.
12. Kang MH, Reynolds CP (2009) Bcl-2 inhibitors: targeting mitochondrial apoptotic pathways in cancer therapy. Clin Cancer Res 15(4): 1126-1132.
13. Kelly PN, Strasser A (2011) The role of Bcl-2 and its pro-survival relatives in tumourigenesis and cancer therapy. Cell Death Differ 18(9): 1414-1424.
14. Thomas S, Quinn BA, Das SK, Dash R, Emdad L, et al. (2013) Targeting the Bcl-2 family for cancer therapy. Expert Opin Ther Targets 17(1): 61-75.
15. Huang GS, Brouwer-Visser J, Ramirez MJ, Kim CH, Hebert TM, et al. (2010) Insulin-like growth factor 2 expression modulates Taxol resistance and is a candidate biomarker for reduced disease-free survival in ovarian cancer. Clin Cancer Res 16(11): 2999-3010.
16. Samani AA, Yakar S, LeRoith D, Brodt P (2007) The role of the IGF system in cancer growth and metastasis: overview and recent insights. Endocr Rev 28(1): 20-47.
17. Sadava D, Hillis D, Heller C, Berenbaum M (2011) Life: The science of biology. (9th ed.) Sunderland, MA: Sinauer Associates Inc.
18. Bodnar AG, Ouellette M, Frolkis M, Holt SE, Chiu CP, et al. (1998) Extension of life-span by introduction of telomerase into normal human cells. Science 279(5349): 349-352.
19. Aschacher T, Wolf B, Enzmann F, Kienzl P, Messner B, et al. (2015) LINE-1 induces hTERT and ensures telomere maintenance in tumour cell lines. Oncogene 35(1): 94-104.
20. Goel S, Fukumura D, Jain RK (2012) Normalization of the tumor vasculature through oncogenic inhibition: An emerging paradigm in tumor biology. Proc Natl Acad Sci U S A 109(20): E1214.
21. Jain RK (2005) Normalization of tumor vasculature: an emerging concept in antiangiogenic therapy. Science 307(5706): 58-62.
22. Canel M, Serrels A, Frame MC, Brunton VG (2013) E-cadherin–integrin crosstalk in cancer invasion and metastasis. J Cell Sci 126(2): 393-401.

23.Friedl P, Alexander S (2011) Cancer invasion and the microenvironment: plasticity and reciprocity. Cell 147(5): 992-1009.

24.Ramis-Conde I, Chaplain MA, Anderson AR, Drasdo D (2009) Multi-scale modelling of cancer cell intravasation: the role of cadherins in metastasis. Phys Biol 6(1): 016008.

25.Scianna M, Preziosi L (2013) Cellular Potts models: multiscale extensions and biological applications. CRC Press.

26.Köhn-Luque A, De Back W, Starruß J, Mattiotti A, Deutsch A, et al. (2011) Early embryonic vascular patterning by matrix-mediated paracrine signalling: a mathematical model study. PloS One 6(9): e24175.

27.Serini G, Ambrosi D, Giraudo E, Gamba A, Preziosi L, et al. (2003) Modeling the early stages of vascular network assembly. EMBO J 22(8): 1771-1779.

28.Kubota Y, Kleinman HK, Martin GR, Lawley TJ (1988) Role of laminin and basement membrane in the morphological differentiation of human endothelial cells into capillary-like structures. J Cell Biol 107(4): 1589-1598.

29.Scianna M, Bell CG, Preziosi L (2013) A review of mathematical models for the formation of vascular networks. Journal of theoretical biology 333: 174-209.

30.Gamba A, Ambrosi D, Coniglio A, De Candia A, Di Talia S, et al. (2003) Percolation, morphogenesis, and Burger's dynamics in blood vessels formation. Phys Rev Lett 90(11): 118101.

31.Lee MJ, Ye AS, Gardino AK, Heijink AM, Sorger PK, et al. (2012) Sequential Application of Anti-Cancer Drugs Enhances Cell Death by Rewiring Apoptotic Signaling Networks. Cell 149(4): 780-794.

32.Miettinen M, Lasota J (2006) Gastrointestinal stromal tumors: review on morphology, molecular pathology, prognosis, and differential diagnosis. Arch Pathol Lab Med 130(10): 1466-1478.

33.Li P, Zhou C, Xu L, Xiao H (2013) Hypoxia enhances stemness of cancer stem cells in glioblastoma: an in vitro study. Int J Med Sci 10(4): 399-407.

34.Merks RMH, Brodsky SV, Goligorksy MS, Newman SA, Glazier JA (2006) Cell elongation is key to in silico replication of in vitro vasculogenesis and subsequent remodeling. Developmental biology 289(1): 44-54.

35.King JR, Franks SJ (2004) Mathematical analysis of some multi-dimensional tissue-growth models. European Journal of Applied Mathematics 15(3): 273-295.

36.Ambrosi D, Gamba A, Serini G (2004) Cell directional and chemotaxis in vascular morphogenesis. Bull Math Biol 66(6): 1851-1873.

37.Neufeld G, Cohen T, Gengrinovitch S, Poltorak Z (1999) Vascular endothelial growth factor (VEGF) and its receptors. The FASEB Journal 13(1): 9-22.

38.Lanza V, Ambrosi D, Preziosi L (2006) Exogenous control of vascular network formation in vitro: a mathematical model. Networks and Heterogeneous Media 1(4): 621-637.

39.Bown J, Andrews PS, Deeni Y, Goltsov A, Idowu M, et al. (2012) Engineering simulations for cancer systems biology. Curr Drug Targets 13(12): 1560-1574.

40.Faratian D, Clyde RG, Crawford JW, Harrison DJ (2009) Systems pathology—taking molecular pathology into a new dimension. Nat Rev Clin Oncol 6(8): 455-464.

41.Bergers G, Hanahan D (2008) Modes of resistance to anti-angiogenic therapy. Nat Rev Cancer 8(8): 592-603.

42.Li Y, Sampson AT, Bown J, Khalil HS, Deeni Y (2014) Understanding tissue morphology: model repurposing using the CoSMoS process. Natural Computing 1-20.

43.Drasdo D, Hohme S (2005) A single-cell-based model of tumor growth in vitro: monolayers and spheroids. Phys Biol 2(3): 133-147.

44.Shirinifard A, Gens JS, Zaitlen BL, Popławski NJ, Swat M, et al. (2009) 3D multi-cell simulation of tumor growth and angiogenesis. PLoS One 4(10): e7190.

45.Deutsch A (2007) Lattice-gas cellular automaton modeling of developing cell systems. In Single-Cell-Based Models in Biology and Medicine PP: 29-51.

46.Drasdo D (2007) Center-based single-cell models: An approach to multi-cellular organization based on a conceptual analogy to colloidal particles. In Single-Cell-Based Models in Biology and Medicine PP: 171-196.

47.Dallon JC (2007) Models with lattice-free center-based cells interacting with continuum environment variables. In Single-Cell-Based Models in Biology and Medicine PP: 197-219.

48.Peng R, Yao X, Ding J (2011) Effect of cell anisotropy on differentiation of stem cells on micropatterned surfaces through the controlled single cell adhesion. Biomaterials 32(32): 8048-8057.

49.Newman TJ (2005) Modeling multi-cellular systems using sub-cellular elements. Math Biosci Eng 2(3): 613-624.

50.Jiujiang Zhu, Simon Coakley, Mike Holcombe, Rod Hose, Rod Smallwood (2007) Cell Adhesion mediates clonal formation of stem and differentiated cell populations (not published).

51.Hayashi M, Majumdar A, Li X, Adler J, Sun Z, et al. (2013) VE-PTP regulates VEGFR2 activity in stalk cells to establish endothelial cell polarity and lumen formation. Nat Commun 4: 1672.

52.Martin-Belmonte F, Perez-Moreno M (2011) Epithelial cell polarity, stem cells and cancer. Nat Rev Cancer 12(1): 23-38.

53.Liu AM, Wong KF, Jiang X, Qiao Y, Luk JM (2012) Regulators of mammalian Hippo pathway in cancer. Biochim Biophys Acta 1826(2): 357-364.

54.Painter KJ, Hillen T (2013) Mathematical modelling of glioma growth: the use of diffusion tensor imaging (DTI) data to predict the anisotropic pathways of cancer invasion. J Theor Biol 323: 25-39.

55.Palsson E (2007) A 3-D Deformable Ellipsoidal Cell Model with Cell Adhesion and Signaling. In Single-Cell-Based Models in Biology and Medicine PP: 271-299.

56.Stolarska MA, Kim Y, Othmer HG (2009) Multi-scale models of cell and tissue dynamics. Philos Trans A Math Phys Eng Sci 367(1902): 3525-3553.

57.Sundfeldt K (2003) Cell–cell adhesion in the normal ovary and ovarian tumors of epithelial origin; an exception to the rule. Mol Cell Endocrinol 202(1): 89-96.

58.Ramis-Conde I, Drasdo D, Anderson AR, Chaplain MA (2008) Modeling the influence of the E-cadherin-β-catenin pathway in cancer cell invasion: a multiscale approach. Biophys J 95(1): 155-165.

59.Metropolis N, Rosenbluth AW, Rosenbluth MN, Teller AH, Teller E (1953) Equation of state calculations by fast computing machines. The journal of chemical physics 21(6): 1087-1092.

60.Brenner H (1963) The Stokes resistance of an arbitrary particle. Chemical Engineering Science, 18(1): 1-25.

61.Berne BJ, Pechukas P (1972) Gaussian model potentials for molecular interactions. The Journal of Chemical Physics 56(8): 4213-4216.

62.Perram JW, Rasmussen J, Præstgaard E, Lebowitz JL (1996) Ellipsoid contact potential: Theory and relation to overlap potentials. Phys Rev E Stat Phys Plasmas Fluids Relat Interdiscip Topics 54(6): 6565-6572.

63.Perram JW, Wertheim MS (1985) Statistical mechanics of hard ellipsoids. I. Overlap algorithm and the contact function. Journal of Computational Physics 58(3): 409-416.

64.Paramonov L, Yaliraki SN (2005) The directional contact distance of two ellipsoids: Coarse-grained potentials for anisotropic interactions. The Journal of chemical physics 123(19): 194111.

65.Jones JE (1924) On the determination of molecular fields. I. From the variation of the viscosity of a gas with temperature. In Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences (738): 441-462.

66.Jones JE (1924) On the determination of molecular fields. II. From the equation of state of a gas. In Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences 106(738): 463-477.

67.Goldstein H, Poole JCP, Safko JL (2001) Classical Mechanics (3rd Edition). Addison-Wesley.

68.Beysens DA, Forgacs G, Glazier JA (2000) Cell sorting is analogous to phase ordering in fluids. Proc Natl Acad Sci U S A 97(17): 9467-9471.

69.Rieu JP, Upadhyaya A, Glazier JA, Ouchi NB, Sawada Y (2000) Diffusion and deformations of single hydra cells in cellular aggregates. Biophys J 79(4): 1903-1914.

70.Brenner H (1967) Coupling between the translational and rotational Brownian motions of rigid particles of arbitrary shape: II. General theory. Journal of Colloid and Interface Science 23(3): 407-436.

71.Brenner H, Gajdos LJ (1981) London-van der Waals forces and torques exerted on an ellipsoidal particle by a nearby semi-infinite slab. Canadian Journal of Chemistry 59(13): 2004-2018.

72.Derjaguin BV, Muller VM, Toporov YP (1975) Effect of contact deformations on the adhesion of particles. Journal of Colloid and Interface Science 53(2): 314-326.

73.Cooper GM, Hausman RE (2013) The Cell: A Molecular Approach. 6th Edition. 2013. Sinauer Associates: Sunderland, MA.

74.Bicknell GR, Snowden RT, Cohen GM (1994) Formation of high molecular mass DNA fragments is a marker of apoptosis in the human leukaemic cell line, U937. J Cell Sci 107(9): 2483-2489.

75.Deason-Towne F, Perraud AL, Schmitz C (2011) The Mg 2+ transporter MagT1 partially rescues cell growth and Mg 2+ uptake in cells lacking the channel-kinase TRPM7. FEBS Lett 585(14): 2275-2278.

76.Li T, Kon N, Jiang L, Tan M, Ludwig T, et al. (2012) Tumor suppression in the absence of p53-mediated cell-cycle arrest, apoptosis, and senescence. Cell 149(6): 1269-1283.

77.Reinders J (2007) Intel threading building blocks: outfitting C++ for multi-core processor parallelism. " O'Reilly Media Inc".

78.Merks RMH, Koolwijk P (2009) Modeling morphogenesis in silico and in vitro: towards quantitative, predictive, cell-based modeling. Mathematical Modelling of Natural Phenomena 4(04): 149-171.

79.Stauffer D, Aharony A (1994) Introduction to percolation theory. CRC press.

80. Cavalli F, Gamba A, Naldi G, Semplice M, Valdembri D, et al. (2007) 3D simulations of early blood vessel formation. Journal of Computational Physics 225(2): 2283-2300.

81. Daley DJ, Vere-Jones D (2003) An Introduction to the Theory of Point Processes: Volume I: Elementary Theory and Methods. Springer Science & Business Media.

82. Illian J, Penttinen A, Stoyan H, Stoyan D (2008) Statistical analysis and modelling of spatial point patterns. John Wiley & Sons.

83. Bivand RS, Pebesma EJ, Gomez-Rubio V, Pebesma EJ (2008) Applied spatial data analysis with R (Vol. 747248717). New York: Springer.

84. Baddeley A (2008) Analysing spatial point patterns in R. Technical report, CSIRO, 2010.

85. Agnew DJG, Green JEF, Brown TM, Simpson MJ, Binder BJ (2014) Distinguishing between mechanisms of cell aggregation using pair-correlation functions. J Theor Biol 352: 16-23.

86. Binder BJ, Simpson MJ (2013) Quantifying spatial structure in experimental observations and agent-based simulations using pair-correlation functions. Phys Rev E Stat Nonlin Soft Matter Phys 88(2): 022705.

87. Manoussaki D, Lubkin SR, Vemon RB, Murray JD (1996) A mechanical model for the formation of vascular networks in vitro. Acta Biotheor 44(3-4): 271-282.

88. Chen Y, Cairns R, Papandreou I, Koong A, Denko NC (2009) Oxygen Consumption Can Regulate the Growth of Tumors, a New Perspective on the Warburg Effect. PLoS One 4(9): e7033.

89. Allen MP, Germano G (2002) Rigid body potentials, forces and torques.

90. Press WH (2007) Numerical recipes 3rd edition: The art of scientific computing. Cambridge university press.

91. Allen M (2004) Introduction to molecular dynamics simulation. Computational Soft Matter: From Synthetic Polymers to Proteins 23: 1-28.

92. Barenblatt GI (1996) Scaling, self-similarity, and intermediate asymptotics: dimensional analysis and intermediate asymptotics (Vol. 14). Cambridge University Press.

93. Bhaskar R, Nigam A (1990) Qualitative physics using dimensional analysis. Artificial Intelligence 45(1): 73-111.

94. Vempati P, Popel AS, Mac Gabhann F (2011) Formation of VEGF isoform-specific spatial distributions governing angiogenesis: computational analysis. BMC Syst Biol 5(1): 59.

95. Polack FA, Andrews PS, Ghetiu T, Read M, Stepney S, et al. (2010) Reflections on the simulation of complex systems for science. In Engineering of Complex Computer Systems (ICECCS) 2010 15th IEEE International Conference on PP: 276-285.

96. Radszuweit M, Block M, Hengstler JG, Schöll E, Drasdo D (2009) Comparing the growth kinetics of cell populations in two and three dimensions. Phys Rev E Stat Nonlin Soft Matter Phys 79(5): 051907.

97. Long BL, Rekhi R, Abrego A, Jung J, Qutub AA (2013) Cells as state machines: cell behavior patterns arise during capillary formation as a function of BDNF and VEGF. J Theor Biol 326: 43-57.

98. Kerr JF, Wyllie AH, Currie AR (1972) Apoptosis: a basic biological phenomenon with wide-ranging implications in tissue kinetics. British journal of cancer 26(4): 239-257.

99. David O Morgan (2007) The cell cycle, principles of control. Yale J Biol Med 80(3): 141-142.

100. Ortmann B, Druker J, Rocha S (2014) Cell cycle progression in response to oxygen levels. Cell Mol Life Sci 71(18): 3569-3582.

101. Choudhary B, Hanski ML, Zeitz M, Hanski C (2012) Proliferation rate but not mismatch repair affects the long-term response of colon carcinoma cells to 5FU treatment. Cancer Lett 320(1): 56-64.

102. Giesen C, Wang HA, Schapiro D, Zivanovic N, Jacobs A, et al. (2014) Highly multiplexed imaging of tumor tissues with subcellular resolution by mass cytometry. Nat Methods 11(4): 417-422.

103. Eckel S, Brachat-Schwarz W, Fleischer F, Grabarnik P, Schmidt V, et al. "Analysis of spatial correlations in marked point processes (power point)".

104. Cai X, Cai J, Dong S, Deng H, Hu M (2009) [Morphology and mechanical properties of normal lymphocyte and Jurkat revealed by atomic force microscopy]. Sheng Wu Gong Cheng Xue Bao 25(7): 1107-1112.

105. Anderson A, Rejniak K (Eds.) (2007) Single-cell-based models in biology and medicine. Springer Science & Business Media.

106. Anderson PW (1972) More is different. Science 177(4047): 393-396.

107. Barleon B, Sozzani S, Zhou D, Weich HA, Mantovani A, et al. (1996) Migration of human monocytes in response to vascular endothelial growth factor (VEGF) is mediated via the VEGF receptor flt-1. Blood 87(8): 3336-3343.

108. Brú A, Albertos S, Subiza JL, García-Asenjo JL, Brú I (2003) The universal dynamics of tumor growth. Biophys J 85(5): 2948-2961.

109. Chauhan VPS, Stylianopoulos T, Martin JD, Popović Z, Chen O, et al. (2012) Normalization of tumour blood vessels improves the delivery of nanomedicines in a size-dependent manner. Nature Nanotechnology 7(6): 383-388.

110. Chenevert TL, McKeever PE, Ross BD (1997) Monitoring early response of experimental brain tumors to therapy using diffusion magnetic resonance imaging. Clin Cancer Res 3(9): 1457-1466.

111. Culver C, Melvin A, Mudie S, Rocha S (2011) HIF-1α depletion results in SP1-mediated cell cycle disruption and alters the cellular response to chemotherapeutic drugs. Cell Cycle 10(8): 1249-1260.

112. Greenblatt MS, Bennett WP, Hollstein M, Harris CC (1994) Mutations in The P53 Tumor-Suppressor Gene - Clues To Cancer Etiology And Molecular Pathogenesis. Cancer Res 54(18): 4855-4878

113. Grover WH, Bryan AK, Diez-Silva M, Suresh S, Higgins JM, et al. (2011) Measuring single-cell density. Proc Natl Acad Sci U S A 108(27): 10992-10996.

114. Guangqi E, Cao Y, Bhattacharya S, Dutta S, Wang E, et al. (2012) Endogenous vascular endothelial growth factor-A (VEGF-A) maintains endothelial cell homeostasis by regulating VEGF receptor-2 transcription. J Biol Chem 287(5): 3029-3041.

115. Guidolin D, Albertin G, Ribatti D (2010) Exploring in vitro angiogenesis by image analysis and mathematical modeling. Microscopy: science, technology, applications and education 2: 876-884.

116. Indiana University

117. Bloomington website http://www.informatics.indiana.edu/rocha/complex/csm.html

118. Johnson KL, Kendall K, Roberts AD (1971) Surface energy and the contact of elastic solids. In Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences 324(1558): 301-313.

119. Kreeger PK, Lauffenburger DA (2010) Cancer systems biology: a network modeling perspective. Carcinogenesis 31(1): 2-8.

120. Ladyman J, Lambert J, Wiesner K (2013) What is a complex system? European Journal for Philosophy of Science 3(1): 33-67.

121. Li Y (2012) Parallel Agent-based Cell Model in Vessel Formation. Abertay Research Student Conference.

122. Li Y, Sampson AT, Bown J, Deeni Y (2013) Understanding tissue morphology: model repurposing using the CoSMoS process. 6th Complex Systems Modelling and Simulation Workshop.

123. Ma Z, Discher DE, Finkel TH (2012) Mechanical force in T Cell receptor signal initiation. Front Immunol 3: 217.

124. Maugis D (1992) Adhesion of spheres: the JKR-DMT transition using a Dugdale model. Journal of Colloid and Interface Science 150(1): 243-269.

125. Mogilner A, Allard J, Wollman R (2012) Cell polarity: quantitative modeling as a tool in cell biology. Science 336(6078): 175-179.

126. Moran PAP (1950) Notes on continuous stochastic phenomena. Biometrika 37(1-2): 17-23.

127. Mrkvička T, Muška M, Kubečka J (2014) Two step estimation for Neyman-Scott point process with inhomogeneous cluster centers. Statistics and Computing 24(1): 91-100.

128. Namy P, Ohayon J, Tracqui P (2004) Critical conditions for pattern formation and in vitro tubulogenesis driven by cellular traction fields. J Theor Biol 227(1): 103-120.

129. New England Complex Systems Institute (NECSI) website.

130. Palsson E (2008) A 3-D model used to explore how cell adhesion and stiffness affect cell sorting and movement in multicellular systems. J Theor Biol 254(1): 1-13.

131. Parsa H, Upadhyay R, Sia SK (2011) Uncovering the behaviors of individual cells within a multicellular microvascular community. Proc Natl Acad Sci U S A 108(12): 5133-5138.

132. Passarge E (2007) Color atlas of genetics. Thieme.

133. Rasband MN (2010) The axon initial segment and the maintenance of neuronal polarity. Nature Reviews Neuroscience, 11(8): 552-562.

134. Steinberg MS (1962) Mechanism of tissue reconstruction by dissociated cells, II: Time-course of events. Science 137(3532): 762-763.

135. Steinberg MS (1962) On the mechanism of tissue reconstruction by dissociated cells, I. Population kinetics, differential adhesiveness, and the absence of directed migration. Proc Natl Acad Sci U S A 48(9): 1577-1582.

136. Steinberg MS (1962) On the mechanism of tissue reconstruction by dissociated cells, III. Free energy relations and the reorganization of fused, heteronomic tissue fragments. Proc Natl Acad Sci U S A 48(10): 1769-1776.

137. Treloar KK, Simpson MJ, Binder BJ, McElwain DS, Baker RE (2014) Assessing the role of spatial correlations during collective cell spreading. Scientific Reports 4(1): 5713.

138. Vaudry D, Stork PJS, Lazarovici P, Eiden LE (2002) Signaling pathways for PC12 cell differentiation: making the right connections. Science 296(5573): 1648-1649.

139. Vempati P, Popel AS, Mac Gabhann F (2014) Extracellular regulation of VEGF: isoforms, proteolysis, and vascular patterning. Cytokine Growth Factor Rev 25(1): 1-19.

Appendix A

Dimension Analysis

Dimension analysis requires substitution of the corresponding dimension to the right-hand side of all the main equations for forces and torques, after that the value of parameters can be estimated. As the first step, I test the correction of dimension for all equations presenting physical interactions in my model. The estimation of value of parameters is the second step.

I define three basic physical entities: unit length Δx, unit time Δt, and unit density ρ_c. Each physical entity in my model can be represented by the dimension version of itself, which is in form of the physical entity divided by its dimension. Therefore, there are two methods to find the dimension version of a physical entity. One is from the physical definition of this physical entity. For example, the contact force, as a force it can be defined by Newton's law $\mathbf{F} = m\mathbf{a}$, in which the dimension of mass *m* is volume (cubical of length) multiplied by density, or $\Delta x^3 \rho_c$; the dimension of acceleration **a** is the change of velocity in unit time, or $\frac{\Delta x}{\Delta t^2}$. Thus, from the definition of force, the dimension of contact force is $\Delta x^3 \rho_c \cdot \frac{\Delta x}{\Delta t^2}$, or $\frac{\Delta x^4 \rho_c}{\Delta t^2}$.

The other method to find dimensions of a physical entity is by deriving the equation from which this physical entity is calculated. Again, take the contact force as an example, from its formula

$$\mathbf{F}^{con} = K\left[(a_1a_2a_3)^{\frac{1}{3}} + (b_1b_2b_3)^{\frac{1}{3}}\right]^{-2}(a_1a_2a_3b_1b_2b_3)^{\frac{1}{6}} \cdot \left(1-\sqrt{\Phi}\right)^{\frac{3}{2}} \cdot \Phi^{-\frac{1}{2}} \cdot \left(\lambda_0(1-\lambda_0)\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right),$$

, the dimension of contact force can be found by writing all the entities in right hand side of equation with their dimension version. If from two methods the same dimension of entity is derived, then I consider this physical entity has correct dimension.

Step 1: Dimension test for all physical entities in the model

In the following section, I derive the dimensions for all the physical entities in my model. Although some steps look simple or meaningless, this step is to make sure the equations are not malformed during complex equation derivation in Section 3 to let a force dimensionally equal to a distance. That is why this step is important and necessary.

Firstly, the displacement between two ellipsoids **A** and **B** is $\mathbf{s} = (\mathbf{r}_A - \mathbf{r}_B)$ (defined in equation (44)), I can let Δx be unit length, so that s turns to

$$\tilde{\mathbf{s}} = \frac{\mathbf{s}}{\Delta x} \qquad (188)$$

In which $\tilde{\mathbf{s}}$ is the dimensionless version of **s**. Note that Δx is a scalar quantity and **s** is a vector, and so after the division, $\tilde{\mathbf{s}}$ is also a vector. In the following part of this section, I will use $\tilde{P}$ to represent the dimensional version for any physical quantity *P*.

Contact potential:

From its definition, it is known that the contact potential is a scalar quantity. I calculate it here step by step. From equation (46) it is known that Φ is proportional to λ_0, displacement **s** and matrix **G**. As λ_0 is a real value between zero and one, it is scalar as well. The definition of **G** in equation (20). shows it is proportional to matrix $\mathbf{A}^{-1}$ and $\mathbf{B}^{-1}$. Because of equation (11) and (12)

$$\mathbf{A}^{-1} = a_1^2\mathbf{u}_1 \otimes \mathbf{u}_1 + a_2^2\mathbf{u}_2 \otimes \mathbf{u}_2 + a_3^2\mathbf{u}_3 \otimes \mathbf{u}_3 \qquad (189)$$

$$\mathbf{B}^{-1} = b_1^2\mathbf{v}_1 \otimes \mathbf{v}_1 + b_2^2\mathbf{v}_2 \otimes \mathbf{v}_2 + b_3^2\mathbf{v}_3 \otimes \mathbf{v}_3 \qquad (190)$$

a_1, a_2, a_3 and b_1, b_2, b_3 are lengths of the ellipsoid semi-radii, so their dimension is Δx, and **u** and **v** are unit matrices, so the

dimension of $\mathbf{A}^{-1}$ and $\mathbf{B}^{-1}$ is Δx^2. Because of the definition of $\mathbf{G}(\lambda)$, the dimension $\mathbf{G}(\lambda)$ is Δx^2 as well. Therefore, the dimension of $\mathbf{G}^{-1}(\lambda_0)$ is $\frac{1}{\Delta x^2}$. So

$$\tilde{\mathbf{\Phi}}(\mathbf{A},\mathbf{B},\mathbf{s}) = \Delta x \cdot \frac{1}{(\Delta x)^2} \cdot \Delta x \cdot \mathbf{\Phi}(\mathbf{A},\mathbf{B},\mathbf{s}) = \mathbf{\Phi}(\mathbf{A},\mathbf{B},\mathbf{s}) \tag{191}$$

Thus, the dimension of $\mathbf{\Phi}(\mathbf{A},\mathbf{B},\mathbf{s})$ is 1, or I can say that $\mathbf{\Phi}(\mathbf{A},\mathbf{B},\mathbf{s})$ is non-dimensional, i.e. it is scalar.

Contact force:

The next object to test is the contact force. Consider the definition of force in general, $\mathbf{F} = m\mathbf{a}$, so the dimension of force should be a multiple of mass and acceleration. Then mass can be considered as the multiple of its volume and its density. It is easy to determine the dimension of volume is Δx^3. I let ρ_c be the unit density, so that the dimension of mass is $\rho_c \Delta x^3$. The acceleration, on the other hand, equals distance divided by square of time. I let Δt be the unit time, then the dimension of acceleration of **a** is $\frac{\Delta x}{\Delta t^2}$. And the dimension of force should be

$$\rho_c \Delta x^3 \cdot \frac{\Delta x}{\Delta t^2} = \rho_c \frac{\Delta x^4}{\Delta t^2}.$$

From equation (101),

$$\mathbf{F}^{con} = K\left[(a_1a_2a_3)^{\frac{1}{3}} + (b_1b_2b_3)^{\frac{1}{3}}\right]^{-2} (a_1a_2a_3b_1b_2b_3)^{\frac{1}{6}} \cdot \left(1-\sqrt{\mathbf{\Phi}}\right)^{\frac{3}{2}} \cdot \mathbf{\Phi}^{-\frac{1}{2}} \cdot \left(\lambda_0(1-\lambda_0)\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right) \tag{192}$$

It is known that $\tilde{a}_i = \frac{a_i}{\Delta x}(i=1,2,3)$, $\tilde{b}_i = \frac{b_i}{\Delta x}(i=1,2,3)$, $\tilde{\mathbf{\Phi}} = \mathbf{\Phi}$, $\tilde{\mathbf{G}} = \frac{\mathbf{G}}{\Delta x^2}$, so that $\tilde{\mathbf{G}}^{-1} = \frac{\mathbf{G}^{-1}}{\frac{1}{\Delta x^2}}$.

The elastic parameter *K* is defined in equation (74) as

$$\begin{cases} K = \dfrac{4}{3\pi(k_1+k_2)} \\ k_1 = \dfrac{1-\nu^2}{\pi E_1} \\ k_2 = \dfrac{1-\nu^2}{\pi E_2} \end{cases} \tag{193}$$

From their definition, dimension of the (name of physical quantity) E_1, E_2 is $\frac{\rho_c \Delta x^2}{\Delta t^2}$, dimension of ν is 1, i.e.

$$\tilde{E} = \frac{E}{\frac{\rho_c \Delta x^2}{\Delta t^2}} \tag{194}$$

Thus, the dimension of K is $\frac{\rho_c \Delta x^2}{\Delta t^2}$. Plus $\tilde{\lambda}_0 = \frac{\lambda_0}{1}$. Therefore

$$\tilde{\mathbf{F}}^{con} = \tilde{K}\left[\left(\tilde{a}_1\tilde{a}_2\tilde{a}_3\right)^{\frac{1}{3}} + \left(\tilde{b}_1\tilde{b}_2\tilde{b}_3\right)^{\frac{1}{3}}\right]^2 \cdot \left(\tilde{a}_1\tilde{a}_2\tilde{a}_3\tilde{b}_1\tilde{b}_2\tilde{b}_3\right)^{\frac{1}{6}}$$
$$\cdot\left(1-\sqrt{\tilde{\Phi}}\right)^{\frac{3}{2}} \cdot \tilde{\Phi}^{-\frac{1}{2}} \cdot \left(\tilde{\lambda}_0\left(1-\tilde{\lambda}_0\right)\tilde{\mathbf{G}}^{-1}\left(\tilde{\lambda}_0\right)\tilde{\mathbf{s}}\right)$$
$$= \frac{K}{\frac{\rho_c \Delta x^2}{\Delta t^2}}\left[\left(\frac{a_1}{\Delta x}\cdot\frac{a_2}{\Delta x}\cdot\frac{a_3}{\Delta x}\right)^{\frac{1}{3}} + \left(\frac{b_1}{\Delta x}\cdot\frac{b_2}{\Delta x}\cdot\frac{b_3}{\Delta x}\right)^{\frac{1}{3}}\right]^2 \left(\frac{a_1}{\Delta x}\cdot\frac{a_2}{\Delta x}\cdot\frac{a_3}{\Delta x}\cdot\frac{b_1}{\Delta x}\cdot\frac{b_2}{\Delta x}\cdot\frac{b_3}{\Delta x}\right)^{\frac{1}{6}}$$
$$\cdot\left(1-\sqrt{\frac{\Phi}{1}}\right)^{\frac{3}{2}} \cdot \left(\frac{\Phi}{1}\right)^{-\frac{1}{2}} \cdot \left(\lambda_0\left(1-\lambda_0\right)\frac{\mathbf{G}^{-1}}{\frac{1}{\Delta x^2}}\left(\lambda_0\right)\frac{\mathbf{s}}{\Delta x}\right)$$
$$= \frac{1}{\frac{\rho_c \Delta x^4}{\Delta t^2}} \cdot K\left[\left(a_1a_2a_3\right)^{\frac{1}{3}} + \left(b_1b_2b_3\right)^{\frac{1}{3}}\right]^2 \left(a_1a_2a_3b_1b_2b_3\right)^{\frac{1}{6}} \cdot \left(1-\sqrt{\Phi}\right)^{\frac{3}{2}} \cdot \Phi^{-\frac{1}{2}} \cdot \left(\lambda_0\left(1-\lambda_0\right)\mathbf{G}^{-1}\left(\lambda_0\right)\mathbf{s}\right)$$
$$= \frac{\mathbf{F}^{con}}{\frac{\rho_c \Delta x^4}{\Delta t^2}} \qquad (195)$$

From equation (203), the equation for contact force has no dimension problem.

Contact torque:

From definition of toque, $\boldsymbol{\tau}^{Con} = \mathbf{F} \times \mathbf{L}$, in which $\mathbf{L}$ is the force arm. The dimension of torque is $\frac{\rho_c \Delta x^5}{\Delta t^2}$. From equation (134),

$$\boldsymbol{\tau}^{Con} = -\lambda_0\left(1-\lambda_0\right)^2 K\left[\left(a_1a_2a_3\right)^{\frac{1}{3}} + \left(b_1b_2b_3\right)^{\frac{1}{3}}\right]^2 \left(a_1a_2a_3b_1b_2b_3\right)^{\frac{1}{6}} \cdot \left(1-\sqrt{\Phi}\right)^{\frac{3}{2}} \cdot \Phi^{-\frac{1}{2}}$$
$$\cdot\left[\mathbf{A}^{-1}\mathbf{G}^{-1}\left(\lambda_0\right)\mathbf{s}\right] \times \left[\mathbf{G}^{-1}\left(\lambda_0\right)\mathbf{s}\right] \qquad (196)$$

I already know that $\tilde{\lambda}_0 = \frac{\lambda_0}{1}$, $\tilde{K} = \frac{K}{\frac{\rho_c \Delta x^2}{\Delta t^2}}$, $\tilde{a}_i = \frac{a_i}{\Delta x}\left(i = 1,2,3\right)$, $\tilde{b}_i = \frac{b_i}{\Delta x}\left(i = 1,2,3\right)$, $\tilde{\boldsymbol{\Phi}} = \boldsymbol{\Phi}$, $\tilde{\mathbf{A}}^{-1} = \frac{\mathbf{A}^{-1}}{\Delta x^2}$, $\tilde{\mathbf{G}} = \frac{\mathbf{G}}{\Delta x^2}$, so that $\tilde{\mathbf{G}}^{-1} = \frac{\mathbf{G}^{-1}}{\frac{1}{\Delta x^2}}$.

So

$$\tilde{\boldsymbol{\tau}}^{Con} = \frac{\boldsymbol{\tau}^{Con}}{\frac{\rho_c \Delta x^5}{\Delta t^2}} \qquad (197)$$

From equation (205), the equation for contact torque has no dimension problem.

Adhesion energy:

Due to the definition of energy in general, $W = \mathbf{F} \cdot \mathbf{s}$. Thus the dimension of energy is $\frac{\rho_c \Delta x^5}{\Delta t^2}$. I can say the dimension of the adhesion

energy is $\frac{\rho_c \Delta x^5}{\Delta t^2}$ as well. let dimension of ε be $\frac{\rho_c \Delta x^5}{\Delta t^2}$, or I can write it in form of

$$\tilde{\varepsilon} = \frac{\varepsilon}{\frac{\rho_c \Delta x^5}{\Delta t^2}} \tag{198}$$

In which ε is dimensional quantity, and $\tilde{\varepsilon}$ is non-dimensional quantity. Plus, the dimension of $\mathbf{\Phi}(\mathbf{A}, \mathbf{B}, \mathbf{s})$ is 1, then the dimensionless version of equation can be written as

$$\tilde{W}^{Ad} = \tilde{\varepsilon}\sqrt{\tilde{\mathbf{\Phi}}(A,B,s)} = \frac{\varepsilon}{\frac{\rho_c \Delta x^5}{\Delta t^2}} \cdot \frac{\sqrt{\mathbf{\Phi}(A,B,s)}}{1}$$
$$= \frac{\varepsilon\sqrt{\mathbf{\Phi}(A,B,s)}}{\frac{\rho_c \Delta x^5}{\Delta t^2}} = \frac{W}{\frac{\rho_c \Delta x^5}{\Delta t^2}} \tag{199}$$

Therefore, the equation for adhesion energy has no dimension problem.

Adhesion force:

The dimension of force is $\frac{\rho_c \Delta x^4}{\Delta t^2}$. The definition of adhesion force is as follows

$$\mathbf{F}^{Ad} = -\varepsilon\lambda_0(1-\lambda_0)\Phi^{-\frac{1}{2}}\mathbf{G}^{-1}(\lambda_0)\mathbf{s} \tag{200}$$

It is known that $\tilde{\varepsilon} = \frac{\varepsilon}{\frac{\rho_c \Delta x^5}{\Delta t^2}}$, $\tilde{\lambda}_0 = \frac{\lambda_0}{1}$, $\tilde{\mathbf{\Phi}} = \mathbf{\Phi}$, $\tilde{\mathbf{G}}^{-1} = \frac{\mathbf{G}^{-1}}{\frac{1}{\Delta x^2}}$, $\tilde{\mathbf{s}} = \frac{\mathbf{s}}{\Delta x}$

So

$$\tilde{\mathbf{F}}^{Ad} = -\frac{\varepsilon}{\frac{\rho_c \Delta x^5}{\Delta t^2}} \cdot \lambda_0(1-\lambda_0) \cdot \mathbf{\Phi}^{-\frac{1}{2}} \cdot \frac{\mathbf{G}^{-1}}{\frac{1}{\Delta x^2}} \cdot \frac{\mathbf{s}}{\Delta x} = \frac{\mathbf{F}^{Ad}}{\frac{\rho_c \Delta x^4}{\Delta t^2}} \tag{201}$$

There is no dimension problem with adhesion force.

Adhesion torque:

The dimension of torque is $\frac{\rho_c \Delta x^5}{\Delta t^2}$. Equation for adhesion torque is as follows,

$$\boldsymbol{\tau}^{Ad} = \varepsilon\lambda_0(1-\lambda_0)^2\Phi^{-\frac{1}{2}}\left[\mathbf{A}^{-1}\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right] \times \left[\mathbf{G}^{-1}(\lambda_0)\mathbf{s}\right] \tag{202}$$

It is known that $\tilde{\varepsilon}=\dfrac{\varepsilon}{\dfrac{\rho_c\Delta x^5}{\Delta t^2}}$, $\tilde{\lambda}_0=\dfrac{\lambda_0}{1}$, $\tilde{\mathbf{\Phi}}=\mathbf{\Phi}$, , $\tilde{\mathbf{A}}^{-1}=\dfrac{\mathbf{A}^{-1}}{\Delta x^2}$, $\tilde{\mathbf{G}}^{-1}=\dfrac{\mathbf{G}^{-1}}{\dfrac{1}{\Delta x^2}}$, $\tilde{\mathbf{s}}=\dfrac{\mathbf{s}}{\Delta x}$

So

$$\tilde{\mathbf{T}}^{Ad}=-\frac{\varepsilon}{\dfrac{\rho_c\Delta x^5}{\Delta t^2}}\lambda_0(1-\lambda_0)^2\Phi^{-\frac{1}{2}}\left[\frac{\mathbf{A}^{-1}}{\Delta x^2}\cdot\frac{\mathbf{G}^{-1}}{\dfrac{1}{\Delta x^2}}\cdot\frac{\mathbf{s}}{\Delta x}\right]\times\left[\frac{\mathbf{G}^{-1}}{\dfrac{1}{\Delta x^2}}\cdot\frac{\mathbf{s}}{\Delta x}\right]=\frac{\mathbf{T}^{Ad}}{\dfrac{\rho_c\Delta x^5}{\Delta t^2}} \tag{203}$$

Thus there is no dimension problem with adhesion torque.

Stokes resistance force constant:

As a force, the dimension of Stokes resistance force should be $\dfrac{\rho_c\Delta x^4}{\Delta t^2}$.

From its definition it is known that the dimension of viscosity μ is $\dfrac{\mu\cdot\Delta t}{\rho_c\cdot\Delta x^2}$ and the dimension of velocity **u** is $\dfrac{\Delta x}{\Delta t}$. Then I need to estimate the dimension of $\mathbf{K}^t$.

For $\mathbf{K}^t$ it is known following equations (which are discussed in Section 3.6, see Section 3.6 for physical interpretation of each parameter in equations)

$$\mathbf{K}^t=16\pi\left(\frac{1}{\chi+a_1^2\alpha_1}\mathbf{u}_1\otimes\mathbf{u}_1+\frac{1}{\chi+a_2^2\alpha_2}\mathbf{u}_2\otimes\mathbf{u}_2+\frac{1}{\chi+a_3^2\alpha_3}\mathbf{u}_3\otimes\mathbf{u}_3\right) \tag{204}$$

$$\alpha_k=\int_0^\infty\frac{d\lambda}{\left(a_k^2+\lambda\right)\Delta\lambda}\quad(k=1,2,3) \tag{205}$$

$$\chi=\int_0^\infty\frac{d\lambda}{\Delta\lambda}\quad(k=1,2,3) \tag{206}$$

$$\Delta\lambda=\sqrt{\left(a_1^2+\lambda\right)\left(a_2^2+\lambda\right)\left(a_3^2+\lambda\right)} \tag{207}$$

From equation (215), dimension of λ is Δx^2. Thus equation (214) turns to

$$\chi=\int_0^\infty\frac{d\lambda}{\sqrt{\left(a_1^2+\lambda\right)\left(a_2^2+\lambda\right)\left(a_3^2+\lambda\right)}} \tag{208}$$

Therefore

$$\tilde{\chi}=\int_0^\infty\frac{d\dfrac{\lambda}{\Delta x^2}}{\sqrt{\left(\dfrac{a_1^2}{\Delta x^2}+\dfrac{\lambda}{\Delta x^2}\right)\left(\dfrac{a_2^2}{\Delta x^2}+\dfrac{\lambda}{\Delta x^2}\right)\left(\dfrac{a_3^2}{\Delta x^2}+\dfrac{\lambda}{\Delta x^2}\right)}}=\int_0^\infty\frac{\dfrac{1}{\Delta x^2}d\lambda}{\dfrac{1}{\Delta x^3}\sqrt{\left(a_1^2+\lambda\right)\left(a_2^2+\lambda\right)\left(a_3^2+\lambda\right)}}$$

$$=\frac{\int_0^\infty\dfrac{d\lambda}{\sqrt{\left(a_1^2+\lambda\right)\left(a_2^2+\lambda\right)\left(a_3^2+\lambda\right)}}}{\dfrac{1}{\Delta x}}=\frac{\chi}{\dfrac{1}{\Delta x}} \tag{209}$$

i.e. the dimension of χ is $\frac{1}{\Delta x}$.

It is known that $\tilde{a}_k = \frac{a_k}{\Delta x}\left(k=1,2,3\right)$ and $\tilde{\lambda} = \frac{\lambda}{\Delta x^2}$, so equation (213) turns to

$$\tilde{\alpha}_k = \int_0^\infty \frac{d\left(\frac{\lambda}{\Delta x^2}\right)}{\left(\frac{a_k^2}{\Delta x^2}+\frac{\lambda}{\Delta x^2}\right)\sqrt{\left(\frac{a_1^2}{\Delta x^2}+\frac{\lambda}{\Delta x^2}\right)\left(\frac{a_2^2}{\Delta x^2}+\frac{\lambda}{\Delta x^2}\right)\left(\frac{a_3^2}{\Delta x^2}+\frac{\lambda}{\Delta x^2}\right)}} \quad (k=1,2,3)$$

$$= \int_0^\infty \frac{\frac{1}{\Delta x^2}d(\lambda)}{\frac{1}{\Delta x^2}\left(a_k^2+\lambda\right)\frac{1}{\Delta x^3}\sqrt{\left(a_1^2+\lambda\right)\left(a_2^2+\lambda\right)\left(a_3^2+\lambda\right)}} = \frac{\int_0^\infty \frac{d(\lambda)}{\left(a_k^2+\lambda\right)\sqrt{\left(a_1^2+\lambda\right)\left(a_2^2+\lambda\right)\left(a_3^2+\lambda\right)}}}{\frac{1}{\Delta x^3}} = \frac{\alpha_k}{\frac{1}{\Delta x^3}} \tag{210}$$

Thus, the dimension of $\alpha_k\left(k=1,2,3\right)$ is $\frac{1}{\Delta x^3}$.

At last, with all the results above, equation (212) turns to

$$\tilde{\mathbf{K}}^t = 16\pi\left(\frac{1}{\frac{\chi}{\frac{1}{\Delta x}}+\frac{a_1^2}{\Delta x^2}\cdot\frac{\alpha_1}{\frac{1}{\Delta x^3}}}\mathbf{u}_1\otimes\mathbf{u}_1+\frac{1}{\frac{\chi}{\frac{1}{\Delta x}}+\frac{a_2^2}{\Delta x^2}\cdot\frac{\alpha_2}{\frac{1}{\Delta x^3}}}\mathbf{u}_2\otimes\mathbf{u}_2+\frac{1}{\frac{\chi}{\frac{1}{\Delta x}}+\frac{a_3^2}{\Delta x^2}\cdot\frac{\alpha_3}{\frac{1}{\Delta x^3}}}\mathbf{u}_3\otimes\mathbf{u}_3\right)$$

$$= 16\pi\left(\frac{1}{\Delta x\chi+\Delta x a_1^2\alpha_1}\mathbf{u}_1\otimes\mathbf{u}_1+\frac{1}{\Delta x\chi+\Delta x a_2^2\alpha_2}\mathbf{u}_2\otimes\mathbf{u}_2+\frac{1}{\Delta x\chi+\Delta x a_3^2\alpha_3}\mathbf{u}_3\otimes\mathbf{u}_3\right)$$

$$= \frac{1}{\Delta x}\cdot 16\pi\left(\frac{1}{\chi+a_1^2\alpha_1}\mathbf{u}_1\otimes\mathbf{u}_1+\frac{1}{\chi+a_2^2\alpha_2}\mathbf{u}_2\otimes\mathbf{u}_2+\frac{1}{\chi+a_3^2\alpha_3}\mathbf{u}_3\otimes\mathbf{u}_3\right) = \frac{\mathbf{K}^t}{\Delta x} \tag{211}$$

Thus, the dimension of $\mathbf{K}^t$ is Δx, then the dimension of $\left[\tilde{\mathbf{K}}^t\right]^{-1}$ is $\frac{1}{\Delta x}$.

Stokes resistance torque constant:

Similar to equation (212), equation (141) can be transformed as follows

$$\tilde{\mathbf{K}}^r = 16\pi\left(\frac{\frac{a_2^2}{\Delta x^2}+\frac{a_3^2}{\Delta x^2}}{\frac{a_2^2}{\Delta x^2}\cdot\frac{\alpha_2}{\frac{1}{\Delta x^3}}+\frac{a_3^2}{\Delta x^2}\cdot\frac{\alpha_3}{\frac{1}{\Delta x^3}}}\mathbf{u}_1\otimes\mathbf{u}_1+\frac{\frac{a_1^2}{\Delta x^2}+\frac{a_3^2}{\Delta x^2}}{\frac{a_1^2}{\Delta x^2}\cdot\frac{\alpha_1}{\frac{1}{\Delta x^3}}+\frac{a_3^2}{\Delta x^2}\cdot\frac{\alpha_3}{\frac{1}{\Delta x^3}}}\mathbf{u}_2\otimes\mathbf{u}_2+\frac{\frac{a_2^2}{\Delta x^2}+\frac{a_1^2}{\Delta x^2}}{\frac{a_2^2}{\Delta x^2}\cdot\frac{\alpha_2}{\frac{1}{\Delta x^3}}+\frac{a_1^2}{\Delta x^2}\cdot\frac{\alpha_1}{\frac{1}{\Delta x^3}}}\mathbf{u}_3\otimes\mathbf{u}_3\right)$$

$$= 16\pi\left(\frac{1}{\Delta x^3}\cdot\frac{a_2^2+a_3^2}{a_2^2\alpha_2+a_3^2\alpha_3}\mathbf{u}_1\otimes\mathbf{u}_1+\frac{1}{\Delta x^3}\cdot\frac{a_1^2+a_3^2}{a_1^2\alpha_1+a_3^2\alpha_3}\mathbf{u}_2\otimes\mathbf{u}_2+\frac{1}{\Delta x^3}\cdot\frac{a_2^2+a_1^2}{a_2^2\alpha_2+a_1^2\alpha_1}\mathbf{u}_3\otimes\mathbf{u}_3\right) = \frac{1}{\Delta x^3}\mathbf{K}^r \tag{212}$$

So, the dimension of $\mathbf{K}^r$ is Δx^3.

Velocity:

From its definition, it is known the dimension of velocity should be $\frac{\Delta x}{\Delta t}$. I have equation (147)

$$\mathbf{u} = \frac{1}{\mu}\left[\mathbf{K}^t\right]^{-1}\sum\mathbf{F} \tag{213}$$

I also know that $\left[\tilde{\mathbf{K}}^t\right]^{-1} = \frac{\left[\mathbf{K}^t\right]^{-1}}{\frac{1}{\Delta x}}$, $\tilde{\mathbf{F}}^{con} = \frac{\mathbf{F}^{con}}{\frac{\rho_c \Delta x^4}{\Delta t^2}}$, the dimension of viscosity μ is $\frac{\rho_c \Delta x^2}{\Delta t}$, thus

$$\tilde{\mathbf{u}}_0 = \frac{1}{\frac{\mu}{\frac{\rho_c \Delta x^2}{\Delta t}}} \cdot \frac{\left[\mathbf{K}^t\right]^{-1}}{\frac{1}{\Delta x}} \cdot \frac{\sum \mathbf{F}\left(r_{ij}\right)}{\frac{\rho_c \Delta x^4}{\Delta t^2}} = \frac{1}{\frac{\Delta x}{\Delta t}} \cdot \frac{1}{\mu}\left[\mathbf{K}^t\right]^{-1} \sum \mathbf{F}\left(r_{ij}\right) = \frac{1}{\frac{\Delta x}{\Delta t}} \cdot \mathbf{u}_0 \quad (214)$$

Thus, I can ensure that the velocity equation does not have dimension problem.

Angular velocity:

The definition of angular velocity is the angle that the reference object turned in unit time [classical mechanics]. The angle is already dimensionless, or I can say its dimension is 1. Then the dimension of angle per unit time is $\frac{1}{\Delta t}$. From equation (149):

$$\boldsymbol{\omega} = \frac{1}{\mu}\left[\mathbf{K}^r\right]^{-1} \sum \mathbf{T} \quad (215)$$

I also know that $\tilde{\mu} = \frac{\mu}{\frac{\rho_c \Delta x^2}{\Delta t}}$, $\tilde{\mathbf{K}}^r = \frac{\mathbf{K}^r}{\Delta x^3}$, $\tilde{\mathbf{T}}^{Con} = \frac{\mathbf{T}^{Con}}{\frac{\rho_c \Delta x^5}{\Delta t^2}}$, so

$$\tilde{\boldsymbol{\omega}} = \frac{1}{\frac{\mu}{\frac{\rho_c \Delta x^2}{\Delta t}}} \cdot \frac{\left[\mathbf{K}^r\right]^{-1}}{\frac{1}{\Delta x^3}} \cdot \frac{\sum \mathbf{T}_{ij}}{\frac{\rho_c \Delta x^5}{\Delta t^2}} = \frac{\frac{1}{\mu}\left[\mathbf{K}^r\right]^{-1} \sum \mathbf{T}_{ij}}{\frac{1}{\Delta t}} = \frac{\boldsymbol{\omega}}{\frac{1}{\Delta t}} \quad (216)$$

Thus I can say that the angular velocity has no dimension problem in its equation.

Now I am sure there is no dimension problem with any equations or physical quantities I are using, then I can estimate the value of all the constants. The next step is to estimate the values of parameters in the model.

Step 2: Value of parameters

Given satisfactory dimensional analysis for all the physical entities in my model, some parameters require constant values. Given a target system of cell-cell interactions in media, I can determine the physical interpretation of parameters and value ranges for these physical entities, and then from the range I may find proper value for each of them. Although my model is used for two purposes, there are common parts in the estimation of value of parameters, because for both purposes, the parameters should satisfy particular conditions, while the actual values of parameters may vary. Therefore, in this section I firstly discuss the common conditions that parameters should satisfy for both purposes, and then calculate values of parameters separately. The values of parameters being used in simulation are listed in Section 4.1.3 and Section 6.3 (some values are tuned due to simulation result).

First, the unit length, unit time and unit density are the most basic entities, because all the other entities can be represented by combination of them. The physical parameters of the model (the unit time and length, and the constants involved in computing the forces) depend on the following values:

- The typical size of a cell.

- The range of ellipsoidal shapes a cell may adopt.
- The mean density of a cell.
- The dynamic viscosity of the fluid medium.
- The maximum speed at which a cell may move in the medium.

I begin with a basic force analysis, and when analysis deals with the above values, I work out the range of suitable values. Then the values for parameters are chosen according to the requirements of each purpose. Finally, the constants in contact force and adhesion force calculation are to be calculated based on other parameters. Note that although I do not consider the gravity in the model, I consider it for common condition that parameters should satisfy. In this process I derive the range of parameters so that the value of gravity does not significantly affect the condition.

Assume two non-identical ellipsoids and they do not contact each other. According to Newton's laws, $\mathbf{F} = m\mathbf{a}$, in which m is mass of an ellipsoid, $\mathbf{F}$ is a summary of all the forces that effect on it, and $\mathbf{a}$ is the acceleration of an ellipsoid and equals $\frac{d\mathbf{v}}{dT}$, where $\mathbf{v}$ is the velocity of ellipsoid. I consider the gravity of ellipsoid for now, according to equation (147), the summary of all the forces is $m\mathbf{g} - \mu\mathbf{K}^t\mathbf{v}$, in which $m\mathbf{g}$ gravity of ellipsoidal agent is, and $-\mu\mathbf{K}^t\mathbf{v}$ is the Stokes resistance force. So

$$m\frac{d\mathbf{v}}{dT} = m\mathbf{g} - \mu\mathbf{K}^t\mathbf{v} \tag{217}$$

Note that in this equation dT is not unit time length Δt, Δt is a fixed entity and has a value, while dT means the change of velocity over time. Integral equation (201),

$$T = \frac{1}{-\frac{\mu\mathbf{K}^t}{m}} \cdot \ln(\mathbf{g} - \frac{\mu\mathbf{K}^t}{m}\mathbf{v}) + C \tag{218}$$

In which C is constant. To determine the value of C I need to consider the initial condition. At the point that $T_0 = 0$, the ellipsoidal agents are planted into substrate so that the velocity $\mathbf{v}_0$ is a zero vector. Substituting this in to (202), I have $C = \mathbf{g}$. So that

$$\lim_{T\to+\infty} \mathbf{v} = \frac{m\mathbf{g}}{\mu\mathbf{K}^t} \tag{219}$$

Now I consider the approach that a cell being put into fluid, it falls into fluid but moves increasingly slowly until it eventually stops moving because of Stokes resistance. From equation (227) it is known that the terminal velocity approaches to zero when the effect of gravity is far smaller than $\mu\left|\mathbf{K}^t\right|$, i.e.

$$mg \ll \mu\left|\mathbf{K}^t\right| \tag{220}$$

To simplify the situation, I consider cells as a sphere, this will not affect my estimation. For example, for ellipsoid $\mathbf{A}$ with semi-radii a_1, a_2 and a_3 which satisfy $a_1 \le a_2 \le a_3$, I simplify the ellipsoid as a sphere with radius of the longest semi-radius a_3, then the volume of sphere should be larger than the ellipsoid thus the mass of sphere is larger than ellipsoid as well (assume they have the same density). If the mass of sphere satisfy condition $m_{sphere}g \ll \mu\left|\mathbf{K}^t\right|$, the mass of ellipsoid satisfy $m_{ellipsoid}g < m_{sphere}g \ll \mu\left|\mathbf{K}^t\right|$, which also satisfy the condition.

Then the mass of cell can be presented as $m = \rho_e \cdot \frac{4\pi r^3}{3}$, and Stokes resistance force constant $\left|\mathbf{K}^t\right| = 6\pi r$, so that

$$\frac{4\pi r^3 \rho_e}{3} \cdot g \ll 6\pi r \cdot \mu .$$

I let the dynamic viscosity μ have the value of $1\times10^{-3}\, Pa\cdot s$ (this is dynamic viscosity of water), gravity acceleration g have the value of $10m/s^2$, plus density of cell cannot be smaller than this value of water $\rho_c = 1000kg/m^3$. With all these values I have an upper range of cell radius r: $r \ll 6.7\times10^{-4} m$.

In the following section I use the density of water as unit density. Now the terminal speed

$$v_{term} = \frac{2}{9} \cdot \frac{\rho_e g}{\mu} \cdot r^2 \qquad (221)$$

In which μ is dynamic viscosity, r is radius of ellipsoid (simplified to a sphere). From this equation it is known that if the terminal velocity is small, the density of cells should be small, and radius of cells should also be small. At the same time the dynamic viscosity μ should be relatively large so that the terminal speed can be small.

Assuming the non-dimensional terminal speed is $\tilde{v}_{term} = \frac{v_{term}}{\Delta v}$, then according to the definition of speed, the unit speed equals unit length divided by unit time, i.e. $\Delta v = \frac{\Delta x}{\Delta T}$. Note that ΔT is not Δt, because Δt is a concept used for dimension analysis only, while ΔT is the unit time in my model.

Then I will represent the radius in equation (228) by unit entities. It is straight forward to set $\Delta x = r$, i.e. the length of unit length is equal to the typical radius of cell. It is also possible to let typical radius equals several times of unit length to keep potential of representing an entity in between.

For parameters in Section 4, in vascular formation, endothelial cells show strong anisotropy. Thus, I want a long and thin shape for the agents, so that when other agents approach from different directions, the interaction is significantly different. I assume $\tilde{r} = \frac{r}{\Delta x} \in [1, \quad 4]$ at first, later I can choose a proper value from this range.

Obviously, the density of cells is bigger than water, but not too much bigger otherwise the terminal speed will increase. Assume the density of cell

$$\rho_e = 2\rho_c = 2000kg/m^3$$

Substitute all these to equation (228), and then it turns to

$$\Delta x \Delta T \in \left[1.406\times10^{-8}\cdot\tilde{v}_{term}, \quad 2.25\times10^{-7}\cdot\tilde{v}_{term}\right] \qquad (222)$$

I take typical radius of cell as $r = 5\times10^{-6} m$, so that $\Delta x \in \left[1.25\times10^{-6}, \quad 5\times10^{-6}\right] m$, and

$$\Delta T \in \left[2.812\times10^{-3}\cdot\tilde{v}_{term}, \quad 1.8\times10^{-1}\cdot\tilde{v}_{term}\right].$$

Let Δx be 1.25×10^{-6} m, then $v_{term} = 1.11\times10^{-4} m/s$, and $\Delta T \in \left[2.4971\times10^{-1}, \quad 1.5984\times10^{1}\right] s$. Let Δx be 1s, so

$$\Delta v = \frac{\Delta x}{\Delta T} = 1.25\times10^{-6} m/s .$$

Parameter values derived so far:

Unit length $\Delta x = 1.25\times10^{-6} m$; Unit time step $\Delta T = 1s$;

Unit density $\rho_c = 1000kg/m^3$; Unit speed $\Delta v = 1.25\times10^{-6} m/s$.

I next need to estimate the value of constant *K* in contact force/torque equations, and in adhesion energy, force and torque equations. Note that I need non-dimensional entities, so I need to estimate their value then divide by their dimensions.

As the Young's' modulus for cell is 0.471 ± 0.081 kPa [104], so for two same cells, $E_1 = E_2 = (0.471\pm0.081)\times10^3$ Pa. And

$v_1 = v_2 = \frac{1}{3}$, so

$$K = \frac{3}{4}E = 0.35325\times10^3 \quad (223)$$

Because $\tilde{K} = \frac{K\Delta T^2}{\rho_c \cdot \Delta x^2}$,

$$\tilde{K} = 2.2608\times10^{11} \quad (224)$$

Assume ellipsoid *A* and *B* are spheres with radius of *r*.

$$\varepsilon = K[r+r]^2 r\cdot\left(1-\sqrt{\Phi}\right)^{\frac{3}{2}} \quad (225)$$

$$\frac{\varepsilon}{K} = 4r^3\cdot\left(1-\sqrt{\Phi}\right)^{\frac{3}{2}} \quad (226)$$

From (234), it is known that the bigger Φ is, the smaller $\frac{\varepsilon}{K}$ is. And

$$\lim_{\Phi\to 1}\left(\frac{\varepsilon}{K}\right) = 0 \quad (227)$$

Now I assume the adhesion force balances with contact force when contact potential is 0.9216 (which is square of 0.96). Then

$$\varepsilon = K[r+r]^2 r\cdot(1-0.96)^{\frac{3}{2}} \quad (228)$$

$$\frac{\varepsilon}{K} = 4\cdot\left(5\times10^{-6}\right)^3\cdot(0.2)^3 = 4\times10^{-18} \quad (229)$$

So that

$$\varepsilon = 1.413\times10^{-15} \quad (230)$$

As I use dimensional value of parameters in model

$$\tilde{\varepsilon} = \frac{\varepsilon \Delta T^2}{\rho_c \Delta x^5} = 4.6301 \times 10^{11} \quad (231)$$

For parameters in Section 6, due to the interval of two time-lapse images are taken is 10 minutes, and in my simulation, I output the position of agents every 10 simulation loops (See Figure 36 for detail), I let ΔT equal to 1 minute. I assume the density of cancer cells is not significantly different from endothelial cells, and the viscosity of environment is also nearly the same. Then the derivation about terminal speed remains same,

$$v_{term} = 1.11 \times 10^{-4} m/s \quad (232)$$

I assume the value of unit speed remains the same, i.e.

$$\Delta v = \frac{\Delta x}{\Delta T} = 1.25 \times 10^{-6} m/s \quad (233)$$

And then $\Delta x = \Delta v \cdot \Delta T = 7.5 \times 10^{-5}$ m. From time-lapse images of control group, I see that the cancer cells round circles from a top-down view; the radius of circle is about 5.0×10^{-6} m. Thus I set three semi-radii of agent as 1.25×10^{-6} m, 5.0×10^{-6} m, 5.0×10^{-6} m, so that the agent is a thin slice sticking on substrate.

I also assume Young's modulus of cancer cell is not significantly different from endothelial cell. Thus, for cancer cell I also have

$$K = \frac{3}{4} E = 0.35325 \times 10^3 \quad (234)$$

And because $\tilde{K} = \frac{K \Delta T^2}{\rho_c \cdot \Delta x^2}$

$$\tilde{K} = 1.1304 \times 10^{11} \quad (235)$$

Because of equation (237), $\frac{\varepsilon}{K} = 4 \times 10^{-18}$, thus

$$\varepsilon = 1.413 \times 10^{-15} \quad (236)$$

Then from $\tilde{\varepsilon} = \frac{\varepsilon \Delta T^2}{\rho_c \Delta x^5}$

$$\tilde{\varepsilon} = 1.072 \times 10^{15} \quad (237)$$

Appendix B

Code Structure and Simulation Issue

Simulation class

class Ellipsoid;	Contains all the basic physical aspect of model, including all the force/torque/velocity/angular velocity that mentioned above.
class Cell;	Derived from Ellipsoid class. And the biology functions are placed here. For now, there is no biology feature, but it will be added in future.
class Simulation;	Contains an array of object of class Cell as data member and control the simulation such as how long it will run, or in which file the data result will be.

The contact force and contact torque describe the physical interaction between cells, so that there is an Ellipsoid class to contain all physical-relative functionality. Also, the Ellipsoid class contains the method to calculate velocity and angular velocity.

Because my aim is to simulate the behaviour of a cell, there are forces and other factors that affect it. The Cell class is derived from Ellipsoid class, so that it contains the physical functions as well. The Cell class also contains cell cycle and some cell related forces and torques. Now they are kept blank, because it makes it clearer to test the physical part of the model first. Now the code is a prototype of the final model, each feature will be added in after the previous step is proved. When I operate an object of Cell class, which is what I do in the simulation, the physical work is done by functions of its parent class, the biological work is done by Cell functions (they are blank at the moment though).

Finally, a list of the Cell object is included in the Simulation class. The Simulation class reads settings from file, initializes the cell list, and controls all the calculations. It also saves the position and direction of each cell regularly.

Cell class

Class HippoCell

Data

- Firstly, HippoCell class has a mark to tell which stage it is in:

CellStage currentCellStage;

CellStage is an enum type which is defined in the header file.

- HippoCell class has a group of timers which controls how long each stage should be:

int GrowthStageLength; // how long it will take for the cell to grow to full size and turn to split //stage

int SplitStageLength; // how long the split stage should be

int MaxAge; // if cell cycle >= this number and cell not in split stage, the cell gets to die out stage.

Each of these 3 timers is generated from its range which is combined by fixed max value and min value, which is defined in header file. Here I need an algorithm to have normal distribution of stage length.

- HippoCell class also has 3 tuners for each timer; to represent the effect that environment does on cells.

int GrowthStageTuner;

int SplitStageTuner;

int MaxAgeTuner;

These can be dynamic values that change along the environment.

- There should be a percentage value. When cell grows to full size, it has a chance to split or die. This value is a decimal between 0 and 1 and should be different for different cell types.

float SplitChance;

- In the simulation there will be two types of cells, which means I will have two lists of cells, each one belongs to a cell type

and has its own physical parameter set.

- There also should be a mark to tell if the cell is affected by Hippo signal or not.
- If the cell is affected by Hippo signal, it should have 2 counters, one to represent how many contacted neighbours it has and the other to represent the average/total contact potential.

```
int ContactedNeighbor;
float AveragePotential; // only with contacted neighbors
```

Methods

- As a class derived from class Cell, HippoCell class contains all the methods that Cell class has. However, it also has special methods related to stage control and fate determination.

```
void SetCellStage( CellStage stage );
CellStage GetCellStage();
void FateDetermination(); // decide if cell should turn to any stage
```

There also should be a method for splitting, in which a new HippoCell object is built and added to the cell list.

- However, I need a group of methods so that I can manually set the length of each stage:

```
void SetGrowthStage();
void SplitStage();
void SetMaxAge();
```

Single growth control

Code part reflecting growth curve

The parameters relative with growth curve are reproduce rate of single cell, saturation density of environment (these 2 controls the population growth curve); and information about single cell volume during the cell cycle, which controls the single cell growth curve.

To store the single cell growth curve information, I have a new structure called SingleCellVolume, which contains the maximum age of cell and the start and end time of 2 slops. It should look like this:

```
Struct SingleCellVolume
{
    int m_MaxAge;
    int m_GrowthtStartTime;
    int m_GrowthEndTime;
    float m_GrowthCurveSlope;
};
```

The Single Cell Value is stored in class Simulation as an array and initialize from .txt file. When it is time to build each cell, its value is assigned into the cell as well. I keep a copy in class Simulation so that I can check how each kind of cell grows at any time during the simulation.

The saturation density is stored in class Simulation, because it is part of simulation environment. The reproduce rate, however, may differ for each type of cell, so that it is stored in each cell.

The parameters of growth curve of single and group cells are loaded in a new function called initGrowthPara(…) called from function SimulationPreStart(…) of class Simulation. The function initGrowthPara(…) reads the growth curve of single cell and reproduce rate from .txt file. The cell related data is stored.

Cell age distribution

In the experiment, cells will not split at exact time, their age abbeys the normal distribution. To implement this, I need to define the desired value (typical value) and variance of cell age. Similarly, each phrase of cell requires a typical value and variance to define.

To make cells obey the normal distribution, in the program, I generate random numbers that obey normal distribution.

TBB template

This topic is discussed in a paper for research students in 2012. Parallel_do is the simplest template to do the same operation to a list of data of foxed size. The data list is divided into smaller segments and distributed to processing threads to process concurrently. The number of concurrent threads can be set by the user or left for the library to decide. Instead of processing data serially parallel_do can process several data at the same time, and in this way the efficiency is expected to be significantly improved. Further, as the division may happen on any part of the data list, this template requires the data list to be randomly accessible so that each thread is able to find the data segment it is assigned to operate. parallel_reduce is another template used in the simulation. Its task is to apply the reduction operation to reconstitute the data list following parallel execution, i.e. to aggregate the separate calculations, waiting for all to complete, to re-integrate the data list. For the same reason parallel_reduce requires random access as well.

In the simulation, these two templates are combined together to achieve maximum improvement of efficiency: the parallel_reduce is used to sum up the force between one cell and all other cells, and parallel_do is in charge of do this operation to each cell in the cell list. During the period in which the model parameters are tuned to be realistic, the simulation is run on a machine with Intel quad-core CPU Q8300 2.50G Hz.

(From research student 2012) I measure how much the TBB parallel technique speeds up the simulation with 2800 cells, which is a typical number to show vessel pattern. Firstly, the running time of sequence code is recorded, as the following chart shows, the sequence code uses 9216 ms in each loop of all cells. An optimally parallel code will use 1.00, 0.53, 0.38, and 0.30 times of sequence code time (see Appendix for detail), by running on one, two, three and four CPUs, respectively. Thus, the ideal looping times are 9216.00 ms, 4876.77 ms, 3460.36 ms and 2752.15 ms. Once the code is paralleled by one, two, three and four CPUs the observed running time for each loop is 13328.00 ms, 8625.00 ms, 6079.00 ms, 5007.50 ms, which is 1.46, 0.95, 0.67 and 0.55 times of sequence time. And compared to parallel with one thread,

Number of Cores	Ideal Speedup	Sequence code Time (ms)	Ideal Running Time (ms)	Parallel Code Time (ms)	Parallel vs. Sequence	Actual Speedup
1	1	9126	9126	13328	1.46	1
2	0.53	9126	4876.77	8625	0.95	0.65
3	0.38	9126	3460.36	6079	0.67	0.46
4	0.3	9126	2752.15	5007.5	0.55	0.38

Sequence and Parallel code running time and speedup.

(From research student 2012) It can be seemed more clearly with a chart. As the following one shows, the purple column represents the time spent in sequential execution, the dark red column shows the ideal time spent in parallelised code, and the yellow column represents the actual time spent in parallelised code. I can see that parallel with one thread uses significantly more time than sequential code. This is because building up threads costs extra time, furthermore, the parallel code cannot have benefit from compiler optimization. However parallel with two threads reduce the looping time to almost same level of sequential code. And more CPUs make improvement to the running efficiency, three CPUs reduce running time to about 2/3, and four CPUs used about half of sequence time.

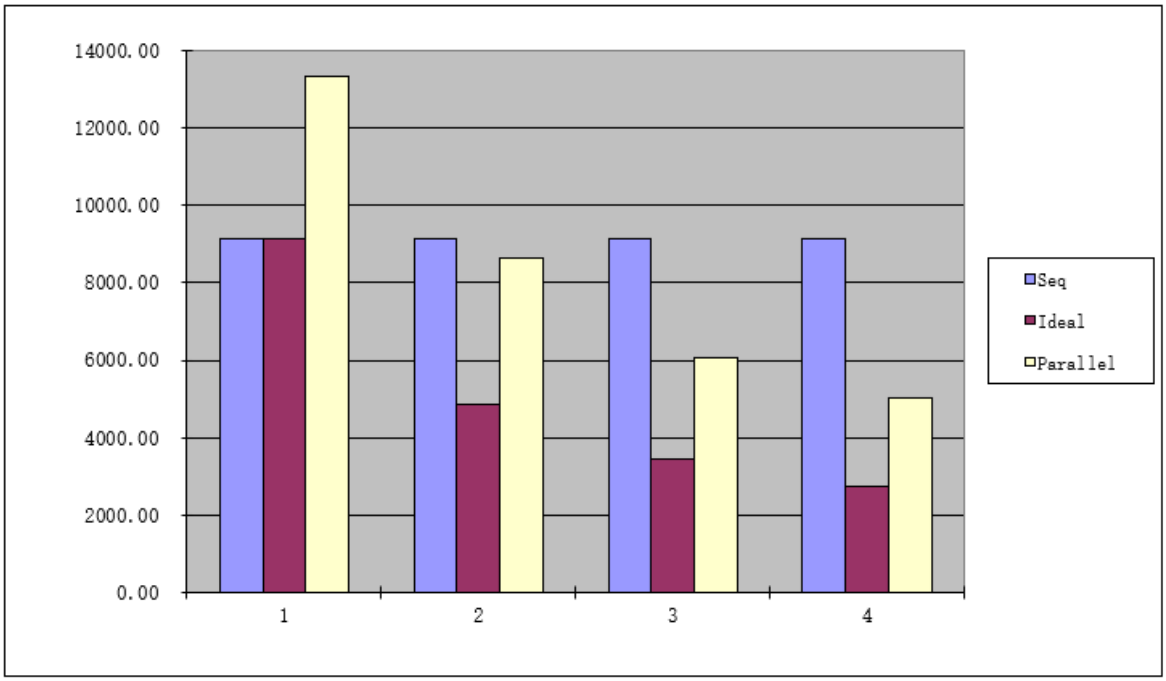


Chart 3.2: Parallel Efficiency with various numbers of CPUs.

(From research student 2012) From the profiler it is known that the code took 335 ms to write data to file, 1904 ms in the main loop and 30322 ms per loop. All these data are obtained by VS2010 Ultimate Profiler with sequence version of code. So that ideally, for N number of CPUs, the paralleled time should be $t(N) = 335 + 1904 + \frac{30322}{N}$ ms. And substitute $N = 1,2,3,4$ I can get ideal running time, which are 32561 ms, 17400 ms, 12346.33 ms and 9819.35 ms, respectively. And then divide them by 32561 ms which is the total time spent in that profile analysis I get how many times the ideal parallel code should be faster.

Appendix C

Values of Module 'IdentifyPrimaryObjects' in Cell Recognisation

In Section 6.1.2, the image of experiments is imported to CellProfiler to be identified. The main process of identification is done in the module 'IdentifyPrimaryObjects'. The interface of module 'IdentifyPrimaryObjects' is shown in Figure 123.

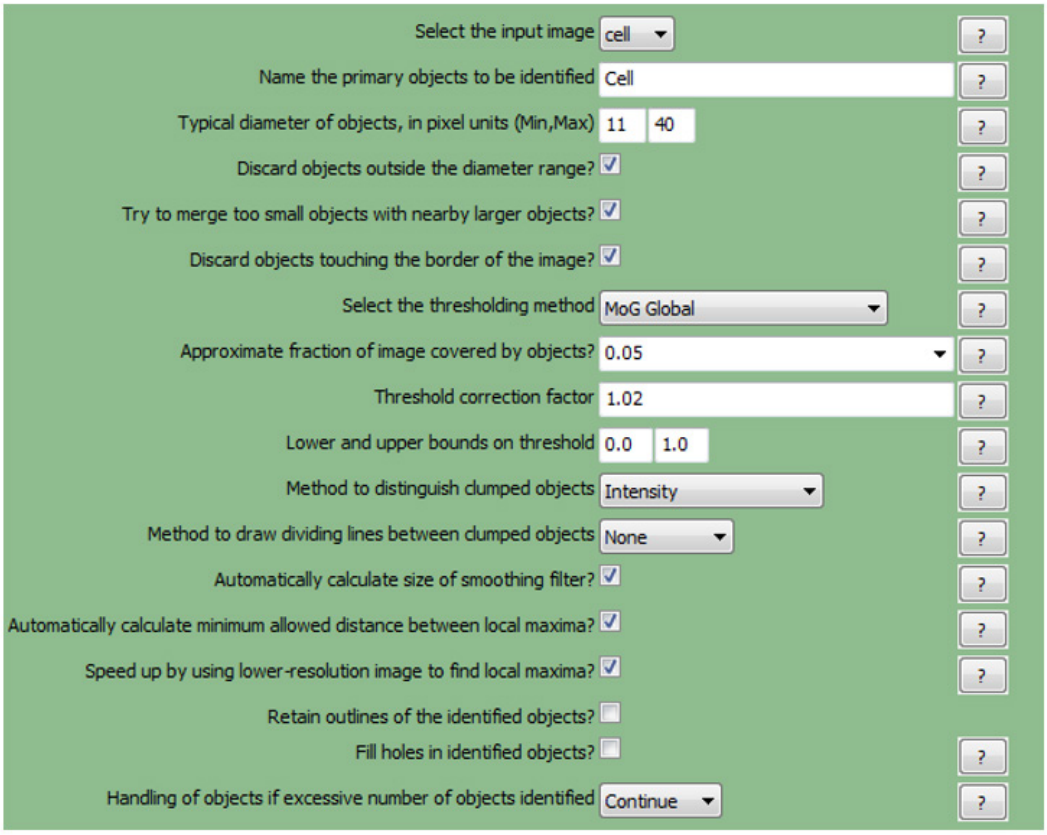


Figure 123: Interface of module 'IdentifyPrimaryObjects'.

As shown in Figure 123, the name of group of images set in 'NamesandTypes' is passed to the first setting. Here the name of the group is 'cell'. An important parameter is the range of diameter of objects, which is measured in number of pixels. From time-lapse images, it can be observed that the diameter of cells is around 10 pixels (when the shape of cell is round), so I set the lower range of diameter of object to 9, 10 or 11. The actual value may vary among images from the same experiment. The actual value is chosen to have most cells recognized while lease dust and other sundries recognized as cell. The higher range of diameter is set to 40, because on some of the images I can see the shape of cell is slender and 40 is large enough to let those cells to be recognized. These values provide the upper and lower limits of cell sizes that are necessary for cell recognition.

Another parameter that can significantly affect the result of recognition is the 'Threshold correction factor'. If a value < 1.0 is entered, the threshold is adjusted to be more lenient so that more cells are recognized; on the contrary, a value > 1.0 means the threshold is adjusted more strictly (CellProfiler document). This value highly depends on a particular image and can also vary among images from the same experiment. The actual values used are shown in the following paragraphs.

The last key parameter is 'Method to distinguish clumped objects'. CellProfiler provides four choices to be selected from. For the time-lapse image, the ''Intensity' or 'Shape' is selected according to achieve better recognition.

The setting of 'IdentifyPrimaryObjects' is shown as follows in Figure 124.

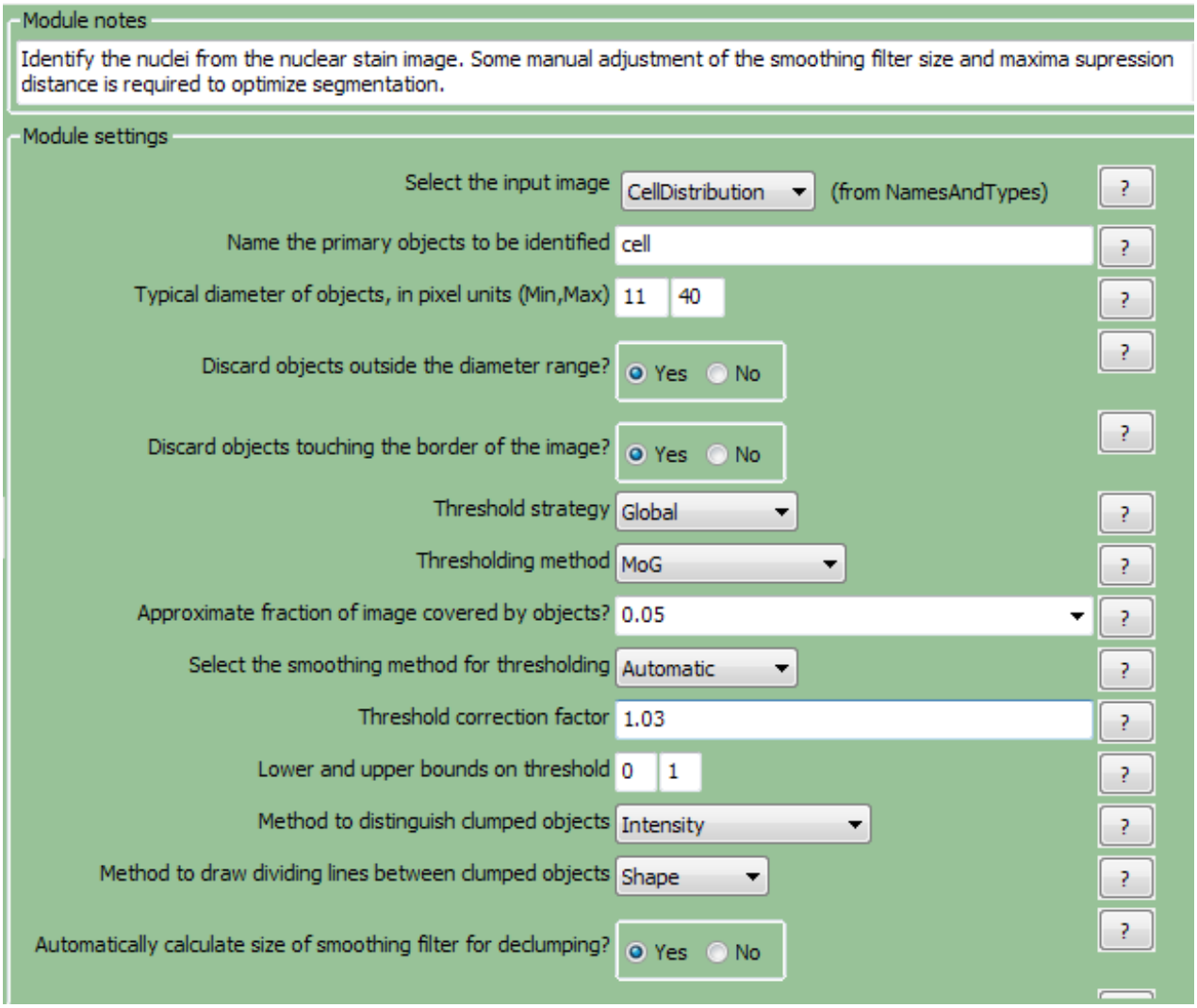


Figure 124: setting of 'IdentifyPrimaryObjects' to identify cells from time-lapse images.

However, for some images the threshold correction factor is modified to produce a better result. For example, in HCT wild type hypoxia group, for image normhpf_0000.png, normhpf_0009.png and normhpf_0019.png, the following setting is used, in which the threshold correction factor is increased from 1.0 to 1.02.

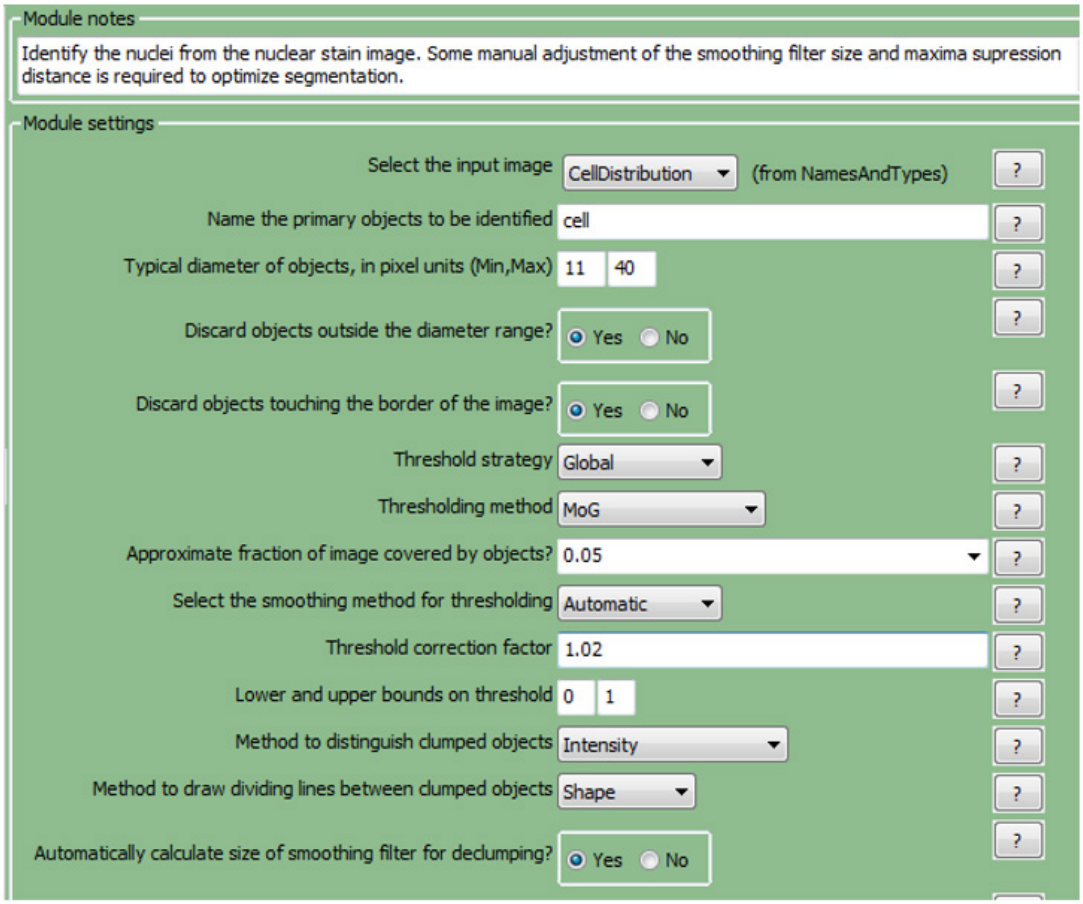


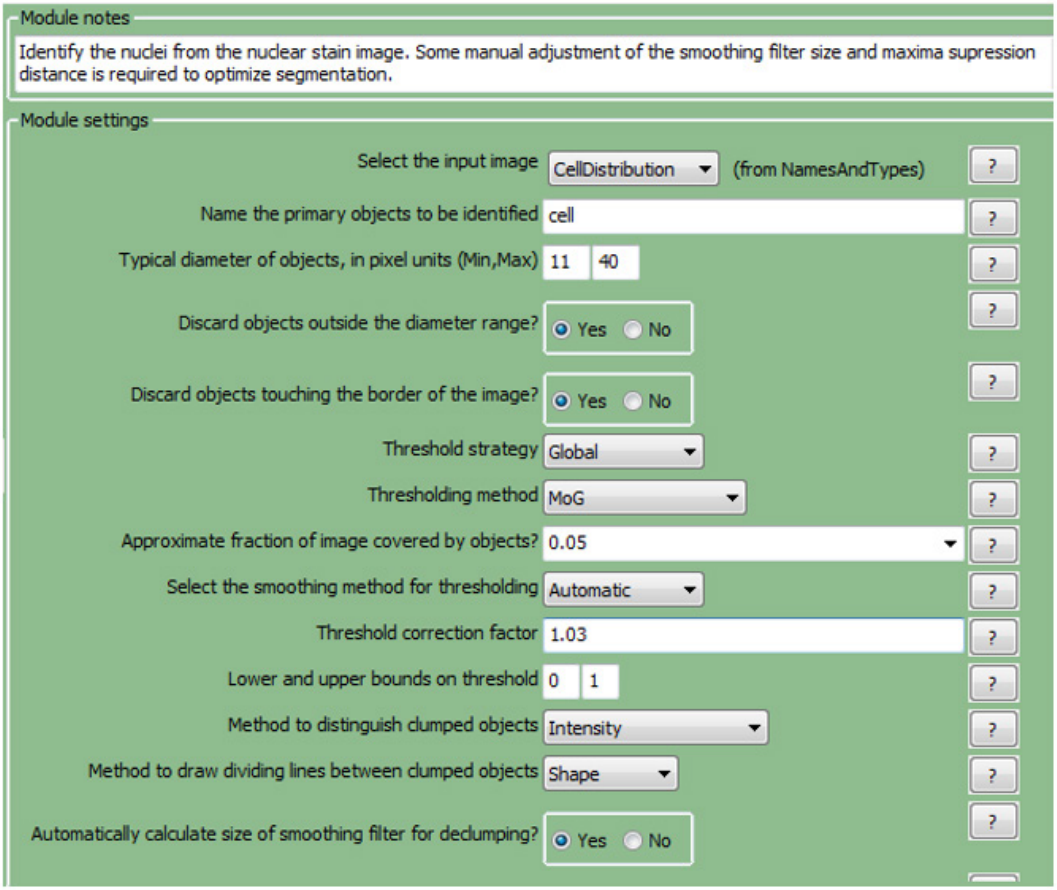
Figure 125: another setting of 'IdentifyPrimaryObjects'.

Values of Threshold correction factor for all 13 images are shown in Table 20. The variation of threshold correction factor suggests that there are still slight differences among this group of images.

Table 20: Values of Threshold correction factor for 13 images of the hypoxic group.

Name of Image	Value of Threshold Correction Factor
normhpf_0000.png	1.02
normhpf_0009.png	1.02
normhpf_0019.png	1.02
normhpf_0029.png	1.05
normhpf_0039.png	1.05
normhpf_0049.png	1.03
normhpf_0059.png	1.03
normhpf_0069.png	1.03
normhpf_0079.png	1.03
normhpf_0089.png	1.03
normhpf_0099.png	1.03
normhpf_0109.png	1.03
normhpf_0119.png	1.03

Figure 126: another setting of 'IdentifyPrimaryObjects'.



Although the change of threshold correction factor seems to be small in value, the edge reorganization is sensitive to this factor. As Figure 126 shows, by increasing or decreasing this factor by 0.01, the outcome is completely different.

In 5FU+Hypoxia group, the variation of threshold correction factor is also found, as Table 21 shows.

Table 21: Values of Threshold correction factor for 9 images of 5FU+Hypoxia group.

Name of image	Value of Threshold correction factor
normhpf_0000.png	1.02
normhpf_0010.png	1.08
normhpf_0020.png	1.05
normhpf_0030.png	1.05
normhpf_0040.png	1.05
normhpf_0050.png	1.05
normhpf_0060.png	1.05
normhpf_0070.png	1.04
normhpf_0080.png	1.04

In 5FU group, the variation of threshold correction factor is shown in Table 22.

Table 22: Values of Threshold correction factor for 12 images of 5FU group.

Name of Image	Value of Threshold Correction Factor
normhpf_0000.png	0.85
normhpf_0010.png	0.85
normhpf_0020.png	0.85
normhpf_0030.png	0.85
normhpf_0040.png	0.85
normhpf_0050.png	0.85
normhpf_0060.png	0.85
normhpf_0070.png	0.85
normhpf_0080.png	0.85
normhpf_0090.png	1.43
normhpf_01000.png	1.43
normhpf_0110.png	1.43

Ellipsoid Particle Dynamics Simulation on Multi-Core Processors Platforms

Section 1: Introduction

Aim

The aim of this chapter is to simulate the dynamic elliptic particle motion on multi-core platforms. The key issue of this chapter is to experiments these intricate theories which are based on real physical principle rather than pure artificial rule to simulate particles' movement and rotation for up to 20,000 elliptical particles by parallel computing on multi-core processors.

The expressional case of the program is a three-dimensional (3D) simulation that thousands of ellipsoids fall out of funnel like sandglass in the imaginary static fluid or in the atmosphere.

Background

Application of ellipsoidal particle simulation: Our elliptical particle model is one kind of Agent Based Model that is also called Individual Based Model or Distinct Element Model, which is a class of bottom-up computational models for simulating emergent population behaviors based on tracing activation of each individual. Individuals or agents are assigned parameters and traits to describe the interaction between each other and with the environment. Although the rule that is between agents and between agent and environment could be simple, the predicted realistic system may be surprising. For example, many spatial temporal self-organization pattern formations can be predicted See Figure 1.

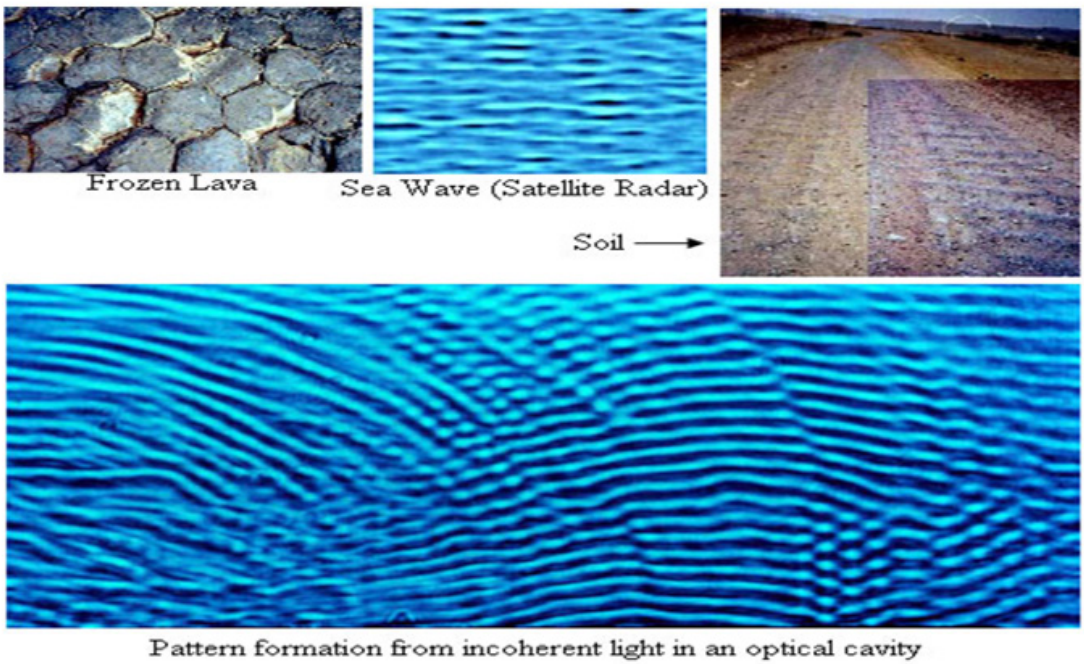


Figure 1: A picture of spatial temporal self-organization pattern formation (from Google).

Each agent is a model as an identical elastic elliptical particle. It may be used more widely than sphere because each volumetric ellipsoid has direction and shape as three different radii. Thus, it could be applied to simulate comprehensive object or phenomenon, such as liquid crystal [1] and cells' collectively migrating epithelium [2]. Both their basic elements are elastic elliptical particles although it is not exactly same in fact, see Figure 2.

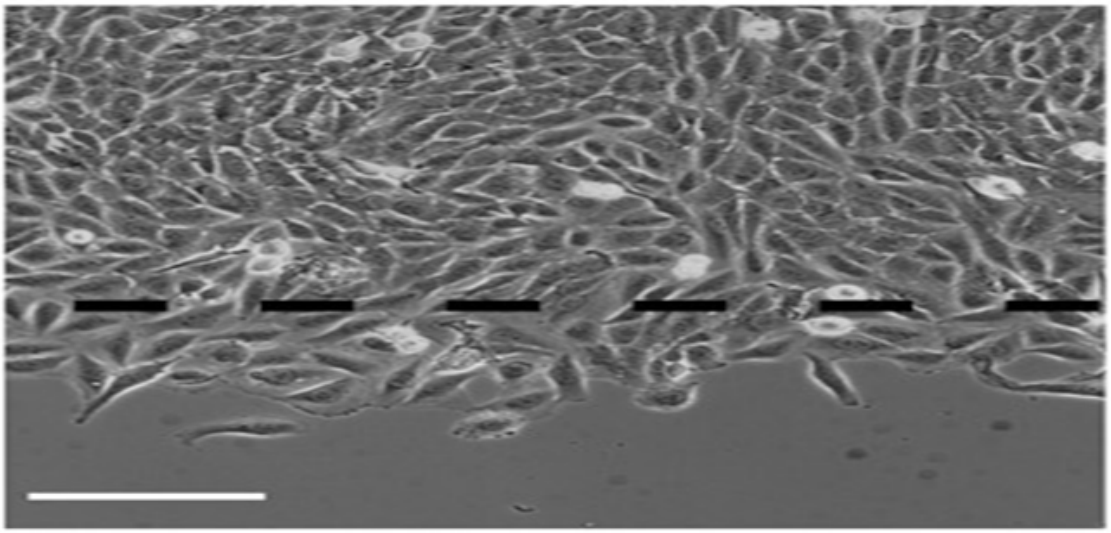


Figure 2: A picture of cells' collectively migrating epithelium (Petitjean et al., 2010).

Except what has been mentioned above, the simulation of ellipsoids also is applied in the game industry. Bullet physics engine (mainly made by Erwin Coumans) [3], a successful and professional free three dimensional (3D) multi-physics library, includes many relevant technologies of ellipsoids such as collision detection and response. Bullet has been widely used by many movie and game companies, and it is also used on many popular game platforms such as PlayStation 3, XBox 360, Nintendo Wii and iPhone [4]. The real-time clothing system is another typical applicant based on physical character of ellipsoid [5] See Figure 3.

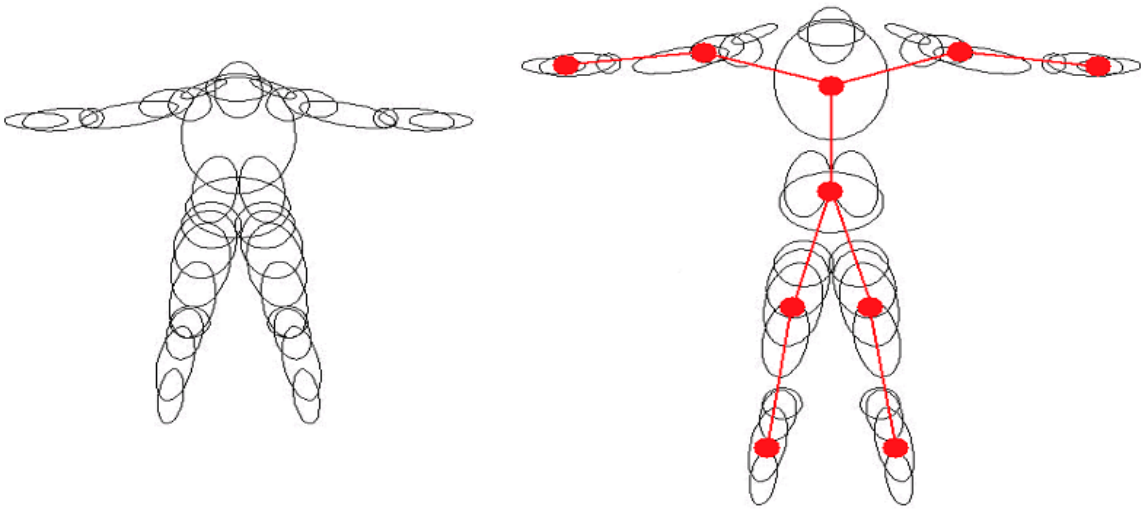


Figure 3: An ellipsoid arrangement and adjacency graph (Rudomin and Castillo).

Relevant knowledge

Models: Our every elastic ellipsoid agent has particular dynamic physical characteristics. For each particle, the force analysis shows that it is affected by contact force between itself and all other ellipsoids (E-E Contact Force) and between itself and slab (E-S Contact Force), as well as its Gravity force and the Stokes resistance force. The summary of all the forces or called total force (this name will be used in all the following sections), is used to calculate the linear acceleration of the ellipsoid. And from acceleration, linear velocity can be obtained. The similar process is implemented to calculate the direction of ellipsoid: firstly, the total torque is obtained by adding up contact torque between ellipsoid and all other ellipsoids, and between ellipsoid and the slab and the Stokes resistance torque; then the total torque is used to calculate the angular acceleration, which later is used to calculate the angular velocity by Euler equations of motion [6].

The contact force and torque between ellipsoids are the first relationship we discussed. They are modeled in various approaches by many people. For example, a well-known expression, Lennard Jones potential (L-J potential), was first proposed in 1924 by John Lennard-Jones of Bristol University. Later, Paramonov and Yaliraki (2005) applied this L-J potential to test molecules of the unique biaxial case: peropyrene $C_{26}H_{14}$ and $C_{14}H_{10}$ anthracene. And then "Berne and Pechukas introduced the Gaussian Overlap Potential (GOP) whose generalization led to the widely use Gay-Berne (GB) potential between ellipsoids" (Paramonov and Yaliraki, 2005), such as simulating liquid crystal [1]. However, Dr Zhu and his PhD student Ye Li are using Hertz model formula to simulate contact force and torque of cell migration [7] These relevant formulas also will be used in this chapter in terms of calculating contact force and torque between ellipsoids.

The situation of contact between ellipsoid particle and the nearby slab can be simplified as the contact between ellipsoid and its mirror image of the slab plane. Under this condition, this method can also be implied. The force and torque generated by Stokes resistance apply on individuals that move in fluid [8-11] indicated that the relationship between the force exerted by the fluid on the particle and translational velocities, and the connection of torque on the particle about the centre of hydrodynamic stress and rotational slip velocities. In this chapter, the Stokes formulas are generated by Brenner's (1964b, pp 635-636) [8] particular case ellipsoid, which the coupling dyadic at centre of ellipsoid and share fore at centre of ellipsoid are zero. Besides, he always provided the Stokes resistance formulas of sphere and disk.

As we assume that the ellipsoid motion and rotation in the static fluid, thus the share torque triadic at centre of ellipsoid also vanishes in terms of no velocity with fluid. Then based on the general formula (3.1) and (3.2) which are from [8] Stokes resistance force and torque formulas in this chapter may simply express as the velocity and direction of the ellipsoid rather than the fluid. After that, obviously, if the mass is enough large, the force of gravity should be computed.

Parallelism: As it is used to simulate simulation behavior, the agent-based model requires the implementation of large group of individuals. In our case, each agent requires complex physics computations. Such as using Ye Li's serial cell model program to simulate two thousand ellipsoids will cost 1.8 minutes per cycle time on double-core machine. In order to achieve speedup and simulate a large number of ellipsoids, parallelism will play an important factor in this program as efficient computation on multi-core processors. The parallel technology of Threading Building Blocks (TBB) [12, 13] offers many rich and complete methods to outfit multi-core processor parallelism in a C++ program [14] have used the technology to research some hybrid paradigms and show the advantages of task stealing techniques in TBB.

Dimensional analysis: As mentioned in section 2, there are many physical quantities in the formulas which are used to calculate the velocity and direction of each ellipsoid. But a "pure" number for a program is explicit and advantageous. Dimensionless quantities, which are a quantity without an associated Physical dimension, are widely used in mechanics, Physics, engineering, economics, and in everyday life (e.g. π). Zhuoru [15] said that for numerous complex phenomena's, theoretical analysis is not only difficult, but also hard to use mathematical expression to accurately solve. Therefore, it should combine the test to make theory in-depth analysis. Dimension analysis may be a popular method to simplify questions and find the dormant factors to indicate the direction of the experiment.

What is more, through my experience in this chapter, dimensional analysis could not only help researchers stand on a higher level to analyze objects with different performances, for instance, it explains why sometimes people could ignore the gravity or acceleration on an object (like the movement of cells); but also help programmers strictly code and flexibly debug.

Section 2: Theory and Method of Modeling Ellipsoids

In this section, the theories for ellipsoid motions will be introduced based on the elliptic geometric and physical properties over a 3D space. The mostly simulation equations are provided by Dr Jiujiang Zhu of University of Abertay Dundee.

Foundational knowledge of ellipsoid

Ellipsoid expression with matrix: First of all, the foundational knowledge of ellipsoid will be introduced. In this chapter, each ellipsoid is generally assumed to be described in Cartesian coordinate (C-O) system See Figure 4.

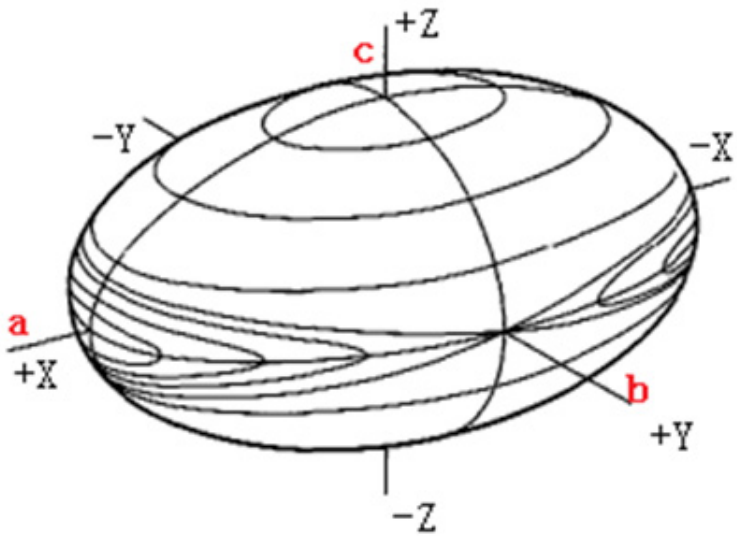


Figure 4: Ellipsoid in Cartesian coordinate.

Where graphemes 'a' and 'b' are the equatorial radii (along the x and y axes), and 'c' is the polar radius (along the z-axis). The basic equation of a standard axis-aligned ellipsoid body in C-O system is:

$$\frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}=1 \quad (1)$$

More generally, an arbitrarily oriented ellipsoid, centered at point 'v', is defined by the equation

$$(\mathbf{x}-\mathbf{v})^{\mathrm{T}}\mathbf{A}^{-1}(\mathbf{x}-\mathbf{v})=1 \quad (2)$$

Where *A* is a positive definite matrix and both x and *v* are vector points. The principal directions of the ellipsoid are defined by eigenvectors of matrix *A* and the square roots of the eigenvalues are the corresponding equatorial radii. According to the second definition, ellipsoid could be processed by a symmetric 3-by-3 matrix. Then the eigenvectors of the matrix are orthogonal (due to the spectral theorem) and represent the directions of the axes of the ellipsoid, the lengths of the semi-axes are given by the eigenvalues. Thus, the expression of ellipsoid A can be given as follows [16].

$$\mathbf{A}=a_1^{-2}\mathbf{u}_1\otimes\mathbf{u}_1+a_2^{-2}\mathbf{u}_2\otimes\mathbf{u}_2+a_3^{-2}\mathbf{u}_3\otimes\mathbf{u}_3 \quad (3)$$

Where $a_i(i=1,2,3)$ is the length of radius, and $0<a_1<a_2<a_3$; the sign $\otimes$ is outer product and

$$\mathbf{U}=\left(\mathbf{u}_1,\mathbf{u}_2,\mathbf{u}_3\right)=\begin{pmatrix}u_{11} & u_{12} & u_{13}\\ u_{21} & u_{22} & u_{23}\\ u_{31} & u_{32} & u_{33}\end{pmatrix} \tag{4}$$

It should be noticed that $\mathbf{U}$ is an orthogonal matrix, and the second index j of u_{ij} represents the number of eigenvectors and it is the orthogonal matrix. Thus, it should satisfy equation (Eq) (5), where $\mathbf{U}^T$ is the transpose of matrix U, and I is the unit matrix.

$$\mathbf{U}^T\mathbf{U}=\mathbf{U}\mathbf{U}^T=\mathbf{I}=\begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{pmatrix} \tag{5}$$

It is easy to verify that the inverse and half matrix of ellipsoids A are Eq (6) and Eq (7), respectively.

$$\mathbf{A}^{-1}=a_1^2\mathbf{u}_1\otimes\mathbf{u}_1+a_2^2\mathbf{u}_2\otimes\mathbf{u}_2+a_3^2\mathbf{u}_3\otimes\mathbf{u}_3 \tag{6}$$

$$\mathbf{A}^{1/2}=a_1^{-1}\mathbf{u}_1\otimes\mathbf{u}_1+a_2^{-1}\mathbf{u}_2\otimes\mathbf{u}_2+a_3^{-1}\mathbf{u}_3\otimes\mathbf{u}_3 \tag{7}$$

Using matrix form [17,16] to describe equations (Eqs) (3), (6) and (7), they are:

$$\mathbf{A}=\mathbf{U}\bar{\mathbf{A}}\mathbf{U}^T=\begin{pmatrix}u_{11} & u_{12} & u_{13}\\ u_{21} & u_{22} & u_{23}\\ u_{31} & u_{32} & u_{33}\end{pmatrix}\begin{pmatrix}\frac{1}{a_1^2} & 0 & 0\\ 0 & \frac{1}{a_2^2} & 0\\ 0 & 0 & \frac{1}{a_3^2}\end{pmatrix}\begin{pmatrix}u_{11} & u_{21} & u_{31}\\ u_{12} & u_{22} & u_{32}\\ u_{13} & u_{23} & u_{33}\end{pmatrix} \tag{8}$$

$$\mathbf{A}^{1/2}=\mathbf{U}\bar{\mathbf{A}}^{1/2}\mathbf{U}^T=\begin{pmatrix}u_{11} & u_{12} & u_{13}\\ u_{21} & u_{22} & u_{23}\\ u_{31} & u_{32} & u_{33}\end{pmatrix}\begin{pmatrix}\frac{1}{a_1} & 0 & 0\\ 0 & \frac{1}{a_2} & 0\\ 0 & 0 & \frac{1}{a_3}\end{pmatrix}\begin{pmatrix}u_{11} & u_{21} & u_{31}\\ u_{12} & u_{22} & u_{32}\\ u_{13} & u_{23} & u_{33}\end{pmatrix} \tag{9}$$

$$\mathbf{A}^{-1}=\mathbf{U}\bar{\mathbf{A}}^{-1}\mathbf{U}^T=\begin{pmatrix}u_{11} & u_{12} & u_{13}\\ u_{21} & u_{22} & u_{23}\\ u_{31} & u_{32} & u_{33}\end{pmatrix}\begin{pmatrix}a_1^2 & 0 & 0\\ 0 & a_2^2 & 0\\ 0 & 0 & a_3^2\end{pmatrix}\begin{pmatrix}u_{11} & u_{21} & u_{31}\\ u_{12} & u_{22} & u_{32}\\ u_{13} & u_{23} & u_{33}\end{pmatrix} \tag{10}$$

Where $\overline{\mathbf{A}}$ represents a diagonal shape matrix.

Stress Analysis

As mentioned above, the total force and torque for each ellipsoid include contact force ($\mathbf{F}_c$) and torque ($\hat{\mathbf{o}}_c$), stokes resistance force ($\mathbf{F}_s$) and torque ($\hat{\mathbf{o}}_s$) and gravity force ($\mathbf{F}_G$). In this section, the methodology of force and torque will be presented as follows except simply gravity force.

Contact potential, force and torque between ellipsoids: Elliptical contact potential (ECP), which was discussed by Perram and Wertheim (1985) [18] and Perram et al (1996) [19], will be used to represent both long range attractive interaction and shorter-range repulsive interaction. The interaction force and interaction torque among particles will be calculated by transformed Hertzian formula that depends on ECP extended coupling potential. The Hertzian formula is a method that was analyzed by Greenwood (1997) [20] and Zhu (2011) [7] to calculate the maximum contact pressure between sphere particles from simple circle particles together with an effect radius.

Contact potential: Consider two ellipsoids labeled A and B with positive semi-axes a_1, a_2, a_3 and b_1, b_2, b_3, respectively. The rotation of each ellipsoid is conveniently expressed by giving the combination of vector sets $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3$ and $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ which are orthonormal unit vectors along the principal axes of the one ellipsoid and other ellipsoids. The centers of these ellipsoids are at $\mathbf{r}_A$ and $\mathbf{r}_B$ in space coordinates (See Figure 5).

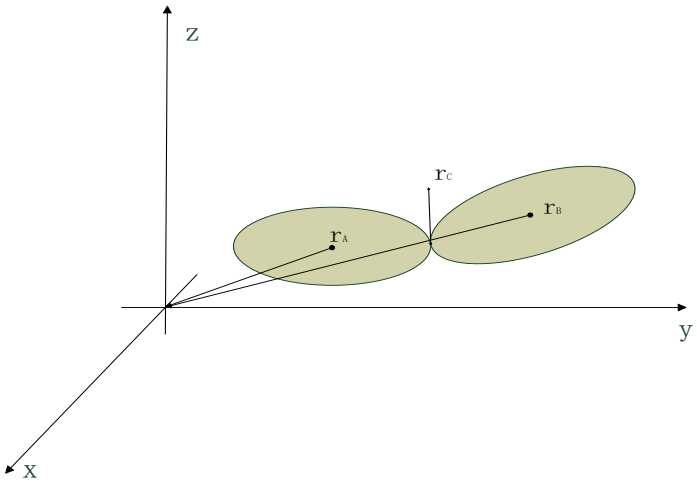


Figure 5: Relation of two ellipsoids in space coordinates.

According to Eq (2), ellipsoid A and B can be expressed as following:

$$A(\mathbf{r}) = \left(\mathbf{r} - \mathbf{r}_A\right)^T \mathbf{A}\left(\mathbf{r} - \mathbf{r}_A\right) = 1 \tag{11}$$

$$B(\mathbf{r}) = \left(\mathbf{r} - \mathbf{r}_B\right)^T \mathbf{B}\left(\mathbf{r} - \mathbf{r}_B\right) = 1 \tag{12}$$

In order to describe the relation between point and ellipsoid, denote a function $F\left(\mathbf{r} - \mathbf{r}_X\right)$ which has the relations with point.

$$F_X\left(\mathbf{r} - \mathbf{r}_X\right)\begin{cases} <1 & \mathbf{r} \text{ inside X} \\ =1 & \mathbf{r} \text{ on the surface of } X(X \to Ellipsoid) \\ >1 & \mathbf{r} \text{ outside X} \end{cases} \tag{13}$$

If $\mathbf{r} = \mathbf{r}_c$ (see in Figure 5), that means the point lines on the surface of ellipsoid. Thus, there is equation $F\left(\mathbf{r}\text{-}\mathbf{r}_X\right) = 1 = F(\mathbf{r})$, then the Eqs. (11) (12) may also denote as following:

$$F_A(\mathbf{r}\text{-}\mathbf{r}_A) = \left(\mathbf{r}\text{-}\mathbf{r}_A\right)^T \mathbf{A}\left(\mathbf{r}\text{-}\mathbf{r}_A\right) \tag{14}$$

$$F_B(\mathbf{r}\text{-}\mathbf{r}_B) = \left(\mathbf{r}\text{-}\mathbf{r}_B\right)^T \mathbf{B}\left(\mathbf{r}\text{-}\mathbf{r}_B\right) \tag{15}$$

Define $S\left(\mathbf{r},\lambda\right)$ as an affine combination of quadratic forms (Eqs.(11)(12)) [21].

$$S\left(\mathbf{r},\lambda\right) = \lambda \cdot A(\mathbf{r}) + \left(1-\lambda\right)\cdot B(\mathbf{r}) \tag{16}$$

It is equal to:

$$F\left(\mathbf{r},\lambda\right) = \lambda F_A(\mathbf{r}\text{-}\mathbf{r}_A) + \left(1-\lambda\right) F_B(\mathbf{r}\text{-}\mathbf{r}_B) \tag{17}$$

Where λ is a parameter and it satisfied $\lambda \in [0,1]$.

Take Eqs. (14)(15) into Eq. (17)

$$F\left(\mathbf{r},\lambda\right) = \lambda\left(\mathbf{r}\text{-}\mathbf{r}_A\right)^T \mathbf{A}\left(\mathbf{r}\text{-}\mathbf{r}_A\right) + \left(1-\lambda\right)\left(\mathbf{r}\text{-}\mathbf{r}_B\right)^T \mathbf{B}\left(\mathbf{r}\text{-}\mathbf{r}_B\right) \tag{18}$$

From the Eq (18), the function is non-negative in the range [0, 1], and the value of this function at point 0 and 1 is zero. Therefore, this function is a concave function and the values of it are non-negative set in [0, 1]. Such that if the function $F\left(\mathbf{r},\lambda\right)$ has a minimum, it can be described as a definition:

$$S(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B) = \min F(\mathbf{r},\lambda) \tag{19}$$

In order to find the minimum set of function $F\left(\mathbf{r},\lambda\right)$ about position $\mathbf{r}$, the vector position $\mathbf{r}$ should satisfy the equation $\frac{\partial S(\mathbf{r},\lambda)}{\partial \mathbf{r}} = 0$. It can be solved as follows.

$$2\lambda\mathbf{A}\left(\mathbf{r}\text{-}\mathbf{r}_A\right) + 2\left(1-\lambda\right)\mathbf{B}\left(\mathbf{r}\text{-}\mathbf{r}_B\right) = 0 \tag{20}$$

Simplified, then there are

$$\mathbf{A}\left(\mathbf{r}\text{-}\mathbf{r}_A\right) = \left(1-\lambda\right)\left[\lambda\mathbf{B}^{-1} + \left(1-\lambda\right)\mathbf{A}^{-1}\right]^{-1}\left(\mathbf{r}_B\text{-}\mathbf{r}_A\right) \tag{21}$$

$$\mathbf{B}\left(\mathbf{r}\text{-}\mathbf{r}_B\right) = \lambda\left[\lambda\mathbf{B}^{-1} + \left(1-\lambda\right)\mathbf{A}^{-1}\right]^{-1}\left(\mathbf{r}_A\text{-}\mathbf{r}_B\right) \tag{22}$$

Define a matrix G (Perram et al., 1996)

$$\mathbf{G}\left(\lambda\right) = \left[\lambda\mathbf{B}^{-1} + \left(1-\lambda\right)\mathbf{A}^{-1}\right] \tag{23}$$

Take Eq (23) into Eqs. (21)(22), there are

$$\mathbf{A}\left(\mathbf{r}-\mathbf{r}_A\right)=\left(1-\lambda\right)\mathbf{G}^{-1}\left(\lambda\right)\left(\mathbf{r}_B-\mathbf{r}_A\right) \tag{24}$$

$$\mathbf{B}\left(\mathbf{r}-\mathbf{r}_B\right)=\lambda\mathbf{G}^{-1}\left(\lambda\right)\left(\mathbf{r}_A-\mathbf{r}_B\right) \tag{25}$$

And from Eq (20), the $\mathbf{r}$ should be expressed as the equation (26) which is a function about constant λ

$$\mathbf{r}(\lambda)=\left[\lambda\mathbf{A}+\left(1-\lambda\right)\mathbf{B}\right]^{-1}\left[\lambda\mathbf{A}\cdot\mathbf{r}_A+\left(1-\lambda\right)\mathbf{B}\cdot\mathbf{r}_B\right] \tag{26}$$

To confirm this equation, it is same with equation (5) in [21].

Continue to transform Eqs. (24)(25), there are

$$\left(\mathbf{r}-\mathbf{r}_A\right)^T=\left(1-\lambda\right)\left(\mathbf{r}_B-\mathbf{r}_A\right)^T\left[\mathbf{G}^{-1}\left(\lambda\right)\right]^T\left[\mathbf{A}^{-1}\right]^T \tag{27}$$

$$\left(\mathbf{r}-\mathbf{r}_B\right)^T=\lambda\left(\mathbf{r}_A-\mathbf{r}_B\right)^T\left[\mathbf{G}^{-1}\left(\lambda\right)\right]^T\left[\mathbf{B}^{-1}\right]^T \tag{28}$$

From Eq. (19), it can be defined that

$$\begin{aligned}S\left(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B\right)&=F\left(\mathbf{r},\lambda\right)\Big|_{\mathbf{r}=\mathbf{r}(\lambda)}=F\left[\mathbf{r}\left(\lambda\right),\lambda\right]\\&=\lambda F_A\left[\mathbf{r}\left(\lambda\right)-\mathbf{r}_A\right]+\left(1-\lambda\right)F_B\left[\mathbf{r}\left(\lambda\right)-\mathbf{r}_B\right]\\&=\lambda S_A\left(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B\right)+\left(1-\lambda\right)S_B\left(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B\right)\end{aligned} \tag{29}$$

Then base on Eq. (26)(27)(28), the $S\left(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B\right)$ can be expressed as

$$\begin{aligned}&S\left(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B\right)=\min_{\mathbf{r}}F(\mathbf{r},\lambda)=F[\mathbf{r}(\lambda),\lambda]\\&=\lambda\left(1-\lambda\right)\left(\mathbf{r}_A-\mathbf{r}_B\right)^T\mathbf{G}^{-1}\left(\lambda\right)\left(\mathbf{r}_A-\mathbf{r}_B\right)\end{aligned} \tag{30}$$

According to the contact function (Perram et al., 1996) $F(\mathbf{A},\mathbf{B})$ for two ellipsoids is

$$F(\mathbf{A},\mathbf{B})=\{\max S(\lambda)\,|\,\lambda\in[0,1]\} \tag{31}$$

Then the contact potential may be defined as follow:

$$\begin{aligned}\Phi\left(\mathbf{A},\mathbf{B},\mathbf{r}_A,\mathbf{r}_B\right)&=\left\{\max_{\lambda}S\left(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B\right)\middle|\,\lambda\in\left[0,1\right]\right\}\\&=\left\{\max_{\lambda}\left[\min_{\mathbf{r}}F\left(\mathbf{r},\lambda\right)\right]\middle|\,\lambda\in\left[0,1\right]\right\}\\&=\left\{\max_{\lambda}\left[\lambda\left(1-\lambda\right)\left(\mathbf{r}_A-\mathbf{r}_B\right)^T\mathbf{G}^{-1}\left(\lambda\right)\left(\mathbf{r}_A-\mathbf{r}_B\right)\right]\middle|\,\lambda\in\left[0,1\right]\right\}\end{aligned} \tag{32}$$

If exit a maximum, the parameter λ must satisfy stationary condition

$$\frac{dS(\mathbf{A},\mathbf{B},\lambda,\mathbf{r}_A,\mathbf{r}_B)}{d\lambda}=0 \tag{33}$$

Decompose Eq. (33), denote

$$\begin{aligned} f(\lambda)&=\lambda(1-\lambda)\mathbf{s}^T\mathbf{G}^{-1}(\lambda)\mathbf{s}\\ g(\lambda)&=\frac{df(\lambda)}{d\lambda}=\mathbf{s}^T\mathbf{G}^{-1}(\lambda)\left[(1-\lambda)^2\mathbf{A}^{-1}-\lambda^2\mathbf{B}^{-1}\right]\mathbf{G}^{-1}(\lambda)\mathbf{s}\\ h(\lambda)&=\frac{dg(\lambda)}{d\lambda}=-2\mathbf{s}^T\{\mathbf{G}(\lambda)[\lambda\mathbf{A}+(1-\lambda)\mathbf{B}]\mathbf{G}(\lambda)\}^{-1}\mathbf{s} \end{aligned} \tag{34}$$

Where,

$$\mathbf{s}=(\mathbf{r}_A-\mathbf{r}_B) \tag{35}$$

Due to always $h(\lambda)<0$ in [0, 1], so exit a unique maximum λ_0, which satisfy stationary condition

$$g(\lambda)=0 \tag{36}$$

Therefore, the New Perram & Wertheim ellipsoid contact potential may be expressed as

$$\Phi(\mathbf{A},\mathbf{B},\mathbf{s})=f(\lambda_0)=\lambda_0(1-\lambda_0)\mathbf{s}^T\mathbf{G}^{-1}(\lambda_0)\mathbf{s} \tag{37}$$

From the final equation, the contact potential only depends on the vector position and the normalized direction matrix which consists of three vectors and these vectors stand for the direction of an ellipsoid in space coordinates. The solution how to get values of λ_0, $\mathbf{G}^{-1}(\lambda)$ and more inferred details will be shown in Appendix A which is provided by Dr Zhu and edited by Li.

Contact force and torque: As mentioned in introduction, there are several approaches to compute contact force and torque between elliptic particles. Nevertheless, other methods which were provided by Zhu et al. (2011) [7] will be discussed as follows. This approach referenced the combination of Hertz contact model [22] and above contact potential. The contact deformation energy between spherical particles (A and B) is given by Hertz model as:

$$W_{AB}^{H}=\frac{2Ka^5}{5R_m^2} \tag{38}$$

Where

$$\begin{cases} a=\sqrt{R_m\delta}\\ R_m=\dfrac{R_AR_B}{R_A+R_B}\\ K=4\Big/\left[3\pi\left(\dfrac{1-\nu_1^{\,2}}{\pi E_1}+\dfrac{1-\nu_2^{\,2}}{\pi E_2}\right)\right]\\ \delta=(R_A+R_B-|\mathbf{r}_A-\mathbf{r}_B|)\,\mathrm{H}(R_A+R_B-|\mathbf{r}_A-\mathbf{r}_B|) \end{cases} \tag{39}$$

E is Young's modulus, V is Poisson ratio, R is the radius and r are the position of particle. The function H (.) is Haviside function which is defined as follow:

$$\mathrm{H}(x)=\begin{cases}0 & x<0\\ 1 & x\geq 0\end{cases} \tag{40}$$

Thus, the contact force act on particle A could be calculated through

$$\mathbf{F}_{B,A}^{H}=-\frac{\partial W_{AB}^{H}}{\partial \mathbf{r}_{A}} \tag{41}$$

That also could be used to calculate interaction between ellipsoid as R_m and δ relate to contact potential that is indicated by the reference position. After transformation, the contact deformation energy between two ellipsoids may be expressed as

$$W=\frac{2K\left[\left(a_1a_2a_3\right)^{\frac{1}{3}}+\left(b_1b_2b_3\right)^{\frac{1}{3}}\right]^{2}\left(a_1a_2a_3b_1b_2b_3\right)^{\frac{1}{6}}}{5}\cdot\left[1-\sqrt{F\left(\mathbf{A},\mathbf{B},\mathbf{s}\right)}\right]^{\frac{5}{2}} \tag{42}$$

Note that

$$F\left(\mathbf{A},\mathbf{B},\mathbf{s}\right)=\Phi\left(\mathbf{A},\mathbf{B},\mathbf{s}\right)$$
$$\mathbf{s}=\mathbf{r}_A-\mathbf{r}_B \tag{43}$$

Therefore, the contact force between elastic ellipsoids A and B, which act on ellipsoid A may give by

$$\mathbf{F}_{A}^{con}=-\frac{\partial W}{\partial \mathbf{s}}=-\frac{\partial W}{\partial \Phi}\frac{\partial \Phi}{\partial \mathbf{s}}$$
$$=K\left[\left(a_1a_2a_3\right)^{\frac{1}{3}}+\left(b_1b_2b_3\right)^{\frac{1}{3}}\right]^{2}\left(a_1a_2a_3b_1b_2b_3\right)^{\frac{1}{6}}\cdot\left(1-\sqrt{\Phi}\right)^{\frac{3}{2}}\cdot\Phi^{-\frac{1}{2}}\cdot\left(\lambda_0\left(1-\lambda_0\right)\mathbf{G}^{-1}\left(\lambda_0\right)\mathbf{s}\right) \tag{44}$$

Similarly, the contact torque could be calculated with the same method except that the partial differential should be applied on direction of corresponding ellipsoid.

$$\hat{\mathbf{o}}^{A}=-\frac{\partial W}{\partial u_A}=-\frac{\partial W}{\partial \Phi}\frac{\partial \Phi}{\partial u_A}$$
$$=-\lambda_0\left(1-\lambda_0\right)^{2}K\left[\left(a_1a_2a_3\right)^{\frac{1}{3}}+\left(b_1b_2b_3\right)^{\frac{1}{3}}\right]^{2}\left(a_1a_2a_3b_1b_2b_3\right)^{\frac{1}{6}}$$
$$\cdot\left(1-\sqrt{\Phi}\right)^{\frac{3}{2}}\cdot\Phi^{-\frac{1}{2}}\cdot\left[\mathbf{A}^{-1}\mathbf{G}^{-1}\left(\lambda_0\right)\mathbf{s}\right]\times\left[\mathbf{G}^{-1}\left(\lambda_0\right)\mathbf{s}\right] \tag{45}$$

To summarize, the contact force and torque are also only related to the vector position the direction matrix of each fixed ellipsoid if assume the parameter K is const. More information also can be referred to in Appendix A.

Contact force and torque between an ellipsoid and a nearby boundary: Many areas within the field of colloid science are concerned with the interaction between a particle and a nearby boundary. Similarly, in order to embody the visual performance, the funnel as glyph 'V' will be used as a nearby boundary in this chapter. Generally, the boundary in space is composed of many planes. Thus, the method to calculate the contact force and torque among an ellipsoid and a nearby slab will be introduced in this section.

[23] have discussed that the contact force and torque act on a particle by a nearby wall can be regarded as the interaction between the particle and its reflection on the plane. Likewise, this chapter also applies the solution to calculate the results. Image that if the plane were regarded as the reflection of each relevant ellipsoid, the contact problem with plane will be translated contact between two ellipsoids. Thus, using the same equations to solve the contact problem with a plane would be easier. Therefore, the contact problem with a slab actually is how to calculate the reflection of an ellipsoid on a nearby plane. This method has been provided by Lucas's handout (2010) [24]. First of all, the primary work is to adjudge the collision between an ellipsoid and a given slab See Figure 6.

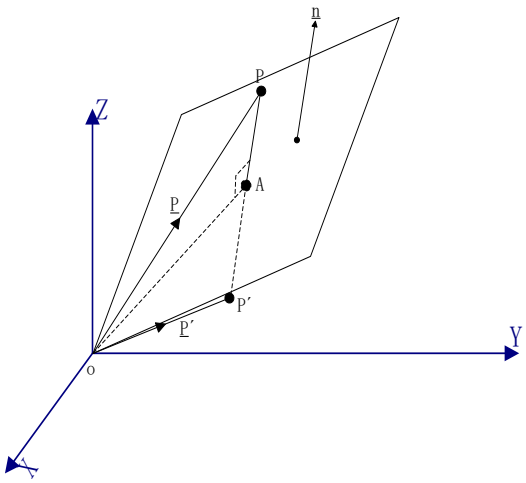


Figure 6: Reflection in a given plane.

Consider the plane passing through the origin, and then its equation is given by $ax+by+cz=0$. A point P in the plane has an image point P′ which is lying vertically opposite P on the other side of the plane through some point A on its surface, so that the $\overrightarrow{P'P}$ is parallel to the normal $\underline{n}=(a,b,c)^T$.

Hence, the shortest distance between point P and the plane is $\left|\overrightarrow{AP}\right|$, that is the projection of P at the normal direction n. So, there is $\overrightarrow{AP}=(\underline{P}\cdot\underline{n})\underline{n}\cdot$

Consider P stands for the ellipsoid, if ellipsoid contact the plane, that means the shortest distance $\left|\overrightarrow{AP}\right|$ is less or equate than its longest radius because the position of ellipsoid is the position of ellipsoidal geometric center.

Following, if the ellipsoid contacts the slab, then calculate its mirror reflection with the slab.

For an arbitrary plane in homogeneous coordinates $ax+by+cz=d$, the plane should perform a translation first to satisfy the plane pass through the origin. After applying the transformation matrix, the plane has to be translated back for the final image.

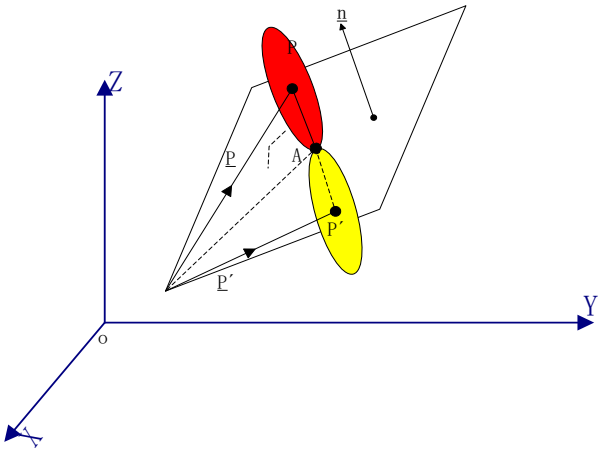


Figure 7: Reflection of ellipsoid with plane.

Based on the normal vector of plane, the transformation matrix could be given by:

$$\mathbf{T}=\begin{pmatrix} -a^2+b^2+c^2 & -2ab & -2ac & 0 \\ -2ab & a^2-b^2+c^2 & -2bc & 0 \\ -2ac & -2bc & a^2+b^2-c^2 & 0 \\ 0 & 0 & 0 & a^2+b^2+c^2 \end{pmatrix} \tag{46}$$

Choose a known point $p_1(a_1,b_1,c_1)$, which satisfies the plane equation $aa_1+bb_1+cc_1=d$. And then translate $p_1(a_1,b_1,c_1)$ to the origin by the matrix:

$$\mathbf{T}_{-p_0}=\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -a_1 & -b_1 & -c_1 & 1 \end{pmatrix} \tag{47}$$

After applying transformation matrix T, the plane should be translated back:

$$\mathbf{T}_{p_0}=\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ a_1 & b_1 & c_1 & 1 \end{pmatrix} \tag{48}$$

Then the image P′ of point P could be giving as

$$P' = P \cdot \mathbf{T}_{-p_1} \cdot \mathbf{T} \cdot \mathbf{T}_{p_1} \tag{49}$$

Note that both the point and its image are 'Vector4' type as (x, y, z, w). The relation between vector4 and general vector is giving by

$$(x, y, z, w) = (x', y', z', 1) = (x', y', z') \tag{50}$$

And then, the directions of image ellipsoid consist of three imaginary vectors which belong to three vector axes of ellipsoid. For the directional matrix of image ellipsoid, they are only related to the direction rather than translation as all of them are unit vectors and represent the body axes. Assume one of the vector axes is vector a. And n is the plane's normal vector. Thus a´ is its image vector.

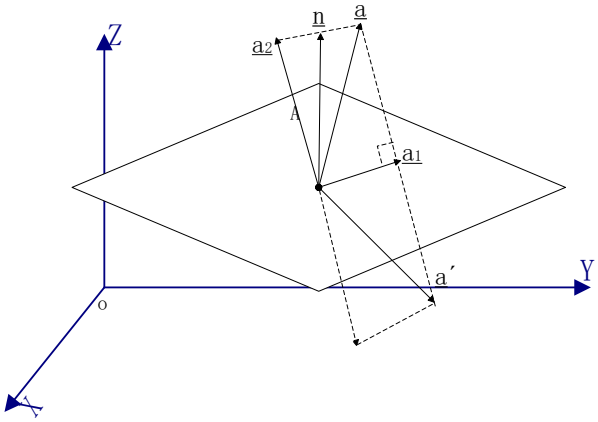


Figure 8: Image vector with a plane.

See Figure 8, the vector a1 and a2 are the projections of vector a on the plane and on the normal direction, respectively. Thus, the relation between them is giving by

$$\begin{cases} \underline{a} = \underline{a}_1 + \underline{a}_2 \\ \underline{a}' = \underline{a}_1 - \underline{a}_2 \end{cases} \tag{51}$$

In which

$$\begin{cases} \underline{a}_2 = (\underline{a} \cdot \underline{n})\underline{n} \\ \underline{a}_1 = \underline{a} - \underline{a}_2 \end{cases} \tag{52}$$

So

$$\underline{a}_1 = \underline{a} - (\underline{a} \cdot \underline{n})\underline{n} \tag{53}$$

Thus, from Eqs. (51)(52)(53), the one of imaging direction matrix vectors could be expressed as

$$\underline{a}' = \underline{a} - 2 \times (\underline{a} \cdot \underline{n})\underline{n} \tag{54}$$

Finally, the contact force and torque between an ellipsoid and a nearby plane could be calculated as two ellipsoids, one is the ellipsoid, and the other is its reflection in the nearby plane.

Stokes Resistance: Resistance may be the relevant and integrant actor in hydrodynamics. Typically, Stokes Resistance has been a generally agreed knowledge of hydrodynamic as it describes the force and torque exerted on an arbitrary rigid particle. In 1962, Brenner issued that "the existence of a unique geometrical point, through which the hydrodynamic force always acts, is established. It is pointed out that this point serves to differentiate translational and rotational particle motions" [8] As follows, the extension of fundamental formula to the ellipsoid case is discussed.

In this chapter, as assume the ellipsoids are in static fluid such that the share torque at the center of ellipsoid may vanish. (The extension part, that the ellipsoid in flowing fluid (has share torque at centre of ellipsoid) will be simulated by Fei Xue). Thus, from Brenner's general resulting equations (3.1) and (3.2) and the example case about ellipsoid [10], the total hydrodynamic force and torque on the ellipsoid in static fluid can be written as

$$\mathbf{F} = -\mu \mathbf{K}^t \cdot (\mathbf{U}_0 - \mathbf{u}_0) \tag{55}$$

$$\mathbf{T} = -\mu \mathbf{K}^r \cdot (\boldsymbol{\omega} - \boldsymbol{\omega}_f) \tag{56}$$

Where μ is the dynamic viscosity of fluid υ and ω represents velocity and angular velocity of ellipsoid respectively υ_f and ω_f are fluid velocity and its angular velocity yet both equate to zero because of static fluid in this chapter $\mathbf{K}^t$ and $\mathbf{K}^r$ are the translation dyadic and rotation dyadic at centre of the ellipsoid, respectively. And their expressions for ellipsoid may be quoted from the reference cited as follows.

$$\mathbf{K}^t = 16\pi \left(\frac{1}{\chi + a_1^2 \alpha_1} \mathbf{u}_1 \otimes \mathbf{u}_1 + \frac{1}{\chi + a_2^2 \alpha_2} \mathbf{u}_2 \otimes \mathbf{u}_2 + \frac{1}{\chi + a_3^2 \alpha_3} \mathbf{u}_3 \otimes \mathbf{u}_3 \right) \tag{57}$$

$$\mathbf{K}^r = \frac{16\pi}{3} \left(\frac{a_2^2 + a_3^2}{a_2^2 \alpha_2 + a_3^2 \alpha_3} \mathbf{u}_1 \otimes \mathbf{u}_1 + \frac{a_1^2 + a_3^2}{a_1^2 \alpha_1 + a_3^2 \alpha_3} \mathbf{u}_2 \otimes \mathbf{u}_2 + \frac{a_2^2 + a_1^2}{a_2^2 \alpha_2 + a_1^2 \alpha_1} \mathbf{u}_3 \otimes \mathbf{u}_3 \right) \tag{58}$$

$$\alpha_k = \int_0^\infty \frac{d\lambda}{\left(a_k^2 + \lambda\right) \Delta\lambda} \quad (k = 1,2,3) \tag{59}$$

$$\chi = \int_0^\infty \frac{d\lambda}{\Delta\lambda} \quad (k = 1,2,3) \tag{60}$$

In which

$$\Delta\lambda = \sqrt{\left(a_1^2 + \lambda\right)\left(a_2^2 + \lambda\right)\left(a_3^2 + \lambda\right)} \tag{61}$$

Where a_1, a_2, a_3 and $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3$ are the lengths of three semi-axes and unit vectors of axes' direction, respectively. The α_k and χ are the parameters which relate to radii of ellipsoid. And component λ is a constant parameter. Therefore, except the constant coefficients like dynamic viscosity α_k and χ, the Stokes resistance force and torque are relevant with the velocity and angular velocity of ellipsoid, respectively.

Euler Equations of Moti: So far, the mainly dynamic stress analysis of the ellipsoid has been mentioned above. As referred by Brenner in the Stokes resistance, there is a particular reference point in the body such that the problem could be split into two separate parts, one is purely translational, and the other is only rotational about the reference point. Certainly, if the point of the body is fixed in an inertial system, that is the obvious reference point. Otherwise, if the body does not have a fixed point, then the center of mass is always the reference point. For this model of ellipsoid, the center of ellipsoid not only is the fixed point, but also is a center of mass. To an ellipsoid as a particle, the translational velocity is simply to calculate based on Newton's Second Law, the equations will be directly given as:

$$\mathbf{F}_{C_{E-E}} + \mathbf{F}_{C_{E-S}} + \mathbf{F}_G + \mathbf{F}_s = m\frac{d\mathbf{v}}{dt} \tag{62}$$

$$\mathbf{v} = \mathbf{v}_o + \frac{d\mathbf{v}}{dt}\cdot \Delta t \tag{63}$$

Where each F stands for the contact force, gravity force and stokes resistance force, respectively. Letter m is the mass of an ellipsoid $\frac{dv}{dt}$ means the momentary acceleration which will be used to calculate the velocity at next step. Parameter $\tilde{\mathbf{o}}$ and $\tilde{\mathbf{o}}_0$ is the velocity of ellipsoid at next time count and at last time count separately. And Δt is the time variation among two neighboring steps.

According to the new velocity, the vector position of next step could be calculated by

$$\mathbf{P} = \mathbf{P}_0 + \mathbf{v}\cdot \Delta t \tag{64}$$

Where $\mathbf{P}_0$ is the current vector position.

Then, the difficult point of motion that has to be considered is the rotational problem. So far, the only suitable generalized coordinates for the rotational motion of rigid body are the Euler angles. However, the inertial of moments depends on the body axes and the angular momentum $\mathbf{L}$ is time independent, also as same as angular velocity which is represented by Euler angles. In order to simplify calculation, only considered the situation is that all of moments are based on body fixed coordination, which indicates the inertial of moments $\mathbf{I}$ is a set of constant values of an ellipsoid. Therefore, to find a way to transform variables between space and body fixed system is momentous.

According to transformation matrix M [6], which is described by three Euler angles, could be used to transform a vector X from space coordinates to body axes.

$$\mathbf{X}^* = \mathbf{M}\cdot\mathbf{X} \tag{65}$$

Or using its transposed matrix transform from body to space system.

$$\mathbf{X} = \mathbf{M}^T\cdot\mathbf{X}^* \tag{66}$$

In order to solve the transform matrix, Rodrigues' rotation formula [25] will be introduced as the rotation axes could be regarded as body fixed frame that is coincident with space. To rotation a vector $\mathbf{r}^*$ as angle θ about n axis, the rotation formula is given by:

$$\mathbf{r}' = \mathbf{r}^*\cos\theta + \left(\mathbf{n}\times\mathbf{r}^*\right)\sin\theta + \left(\mathbf{n}\cdot\mathbf{r}^*\right)\mathbf{n}\left(1-\cos\theta\right) \tag{67}$$

Let, $\mathbf{R}_\mathbf{n}^\theta$ be the rotate angle about n axis, here n is a normal vector along rotation axis, if arbitrary vector $\mathbf{r}^*$ is transformed into $\mathbf{r}$ through rotation $\mathbf{R}_\mathbf{n}^\theta$, then it can be denote as

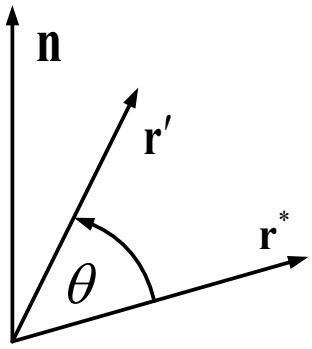


Figure 9: Rodrigues' rotation.

$$\mathbf{r}^{'} = \mathbf{R}_{\mathbf{n}}^{\theta} \cdot \mathbf{r}^{*} \tag{68}$$

Where $\mathbf{R}_{\mathbf{n}}^{\theta}$ can be expressed in component matrix form which is only related to the three normal axes and a rotation angle.

According to Eqs (66) (68), if in body frame the unit vector $\left(\mathbf{e}_1^* \quad \mathbf{e}_2^* \quad \mathbf{e}_3^*\right) = \left(\mathbf{e}_1 \quad \mathbf{e}_2 \quad \mathbf{e}_3\right)$ is coincide with space fix frame, as, it can be said that the Rodrigues' rotation matrix and transformation matrix are transposed matrix each other. (More inferred details is shown in Appendix B)

$$\mathbf{x} = \mathbf{R}_{\mathbf{n}}^{\theta} \cdot \mathbf{x}^{*} \left(\mathbf{x}^{*} = \mathbf{M} \cdot \mathbf{x}\right) \tag{69}$$

Then, Euler's equations of motion could be used to describe the momentary torque relation between space and in body fix coordination system [6]. Used Euler's equation only in body frame system, the equations could be written as:

$$\boldsymbol{\tau}^{*} = \frac{d\mathbf{L}^{*}}{dt} + \boldsymbol{\omega}^{*} \times \mathbf{L}^{*} \tag{70}$$

In which

$$\mathbf{L}^{*} = \mathbf{I}\boldsymbol{\omega}^{*} = \begin{pmatrix} I_1 & 0 & 0 \\ 0 & I_2 & 0 \\ 0 & 0 & I_3 \end{pmatrix} \begin{pmatrix} \omega_1^{*} \\ \omega_2^{*} \\ \omega_3^{*} \end{pmatrix} = \begin{pmatrix} I_1\omega_1^{*} \\ I_2\omega_2^{*} \\ I_3\omega_3^{*} \end{pmatrix} \tag{71}$$

$$I_1 = m_0 \frac{a_2^2 + a_3^2}{5}, \quad I_2 = m_0 \frac{a_3^2 + a_1^2}{5}, \quad I_3 = m_0 \frac{a_1^2 + a_2^2}{5} \tag{72}$$

In which

$$m_0 = \rho_0 V_0 = \rho_0 \frac{4}{3} \pi a_1 a_2 a_3 \tag{73}$$

Therefore, based on Eqs (70)(71)(72)(73) the momentary angular acceleration in body frame could be calculated as

$$\begin{cases} \dfrac{d\omega_1^*}{dt} = \dfrac{\tau_1^*}{I_1} + \omega_2^*\omega_3^* \dfrac{I_2 - I_3}{I_1} \\ \dfrac{d\omega_2^*}{dt} = \dfrac{\tau_2^*}{I_2} + \omega_3^*\omega_1^* \dfrac{I_3 - I_1}{I_2} \\ \dfrac{d\omega_3^*}{dt} = \dfrac{\tau_3^*}{I_3} + \omega_1^*\omega_2^* \dfrac{I_1 - I_2}{I_3} \end{cases} \tag{74}$$

And the new angular velocity in body fixed coordinates could be expressed as

$$\boldsymbol{\omega}^*(t+\Delta t) = \boldsymbol{\omega}^*(t) + \Delta t \frac{d\boldsymbol{\omega}^*(t)}{dt} \tag{75}$$

According to Eq (68), the angular velocity at next time in space system can be transformed by rotation matrix R.

$$\boldsymbol{\omega}(t+\Delta t) = \mathbf{R} \cdot \boldsymbol{\omega}^*(t+\Delta t) \tag{76}$$

And then the new rotation matrix can be calculated because the new normal axes and new angle can be computed by $\boldsymbol{\omega}(t+\Delta t)$. Through the new rotation matrix, the directions of the ellipsoid at next time also could be provided as the column vectors of rotation matrix sand for the eigenvectors.

Section 3: Parallelism with Intel Threading Building Blocks

Thinking Parallel

How should programmers think about parallel programming? As the multi-core processors machines are popular, every software developer may need to think about parallel on basis of algorithm, implementation language, and so on. Scalability of computer hardware embodies the concept that a program should increase efficiency as the number of cores increases. In our circumstance, the scale of simulation may be variable from hundreds to tens of thousands. In addition, the configuration of the machine that our program runs on could be various. Therefore, Parallelism may be a necessary solution for the complex, concentrated and time-consuming computations in the program.

Why Threading Building Blocks (TBB)?

There are many popular methods in parallelism. It is explained in the following section why TBB is used in this chapter.

Comparison with other parallel technologies: The raw thread interface, such as POSIX threads (Pthreads) or Windows threads, has been an option to support programmers to develop shared memory paralleling program. Supercomputer users, without the shared memory, mostly use Message Passing Interface (MPI) standard, which uses the assembly language to parallel and offers maximum flexibility. But its flexibility is at a high cost of programming effort, debugging time and maintenance costs [26]. By 1997, OpenMP had been attempting to fulfill the requirement of C++ programmers. It supports parallel program and helps compiler generate a program with their wishes which programmers only concentrate on paralleling program rather than take care of threading pool. Nevertheless, its commands are limited to FORTRAN and C languages. OpenMP has been referred to as "an excellent for Fortran-style code in C." yet it offers nothing specific for C++, said by Reinders (2007) [26], but now that may be no longer excellent.

Inter Threading Building Blocks was first released in 2006 by Intel cooperation, which provides support to object oriental languages. It is not only an implementation of the new C++ threading standard, but also it represents a higher level, task-based parallelism that promotes scalable data parallel programming. The Intel Company issue that "It is a widely used, award-winning C++ template library for creating reliable, portable, and scalable parallel applications." [13]. It avoids many shortcomings for programmers, such as taking care of the threading pool. Additionally, Intel Cooperation provides the reference manual [13] to guide programmers on how to use the template of the library, as TBB is

a new technique. The benefits of using TBB have been summarized by Intel:

- *Enhanced Productivity and succinct* - TBB supports C++ compiler C++ compilers to help users to get scalable and distinct parallel applications more easily with fewer lines of code.
- *Scalability with Future-proofing* - Application performance automatically improves as number of cores increases by using abstract tasks stealing,
- *Comprehensive* - TBB supports recursive parallelism and generic algorithms and supports nested structure.
- *Flexible* - TBB can be applied on a wide variety of Operating Systems and platforms

Necessary for this Chapter: From the point of requirements of this chapter, firstly, this chapter is developed based on C++ language, which is supported by TBB with its OOP library Secondly, from the point of view of hardware, the parallelism codes should be able to run easily on different computers with different number of cores. Because the larger memory the machine has, the more dynamic elliptic agents could be simulated; and the more processors it has, the faster computation will be achieved. Finally, the parallel algorithm and codes should be compact because of complexity of the physical algorithm of the program.

Importance for Games: Besides its necessity for this chapter, TBB technology is momentous in the game field. Along with the development of TBB, it is employed by more and more game designers. Intel Threading Building Blocks, The Intel Threading Building Blocks, which is an award-winning C++ template library, enables developers to not only create strong parallel applications easily and quickly, but also expand program to available multi-core processors. Therefore, Intel TBB is the most widely used for C and C + + parallel programming. Since the first TBB version was released in 2006, it was supported by Autodesk, Adobe, DreamWorks, Avid and Epic Games as well as hundreds of independent software developers. In order to help customers create, deliver and optimize highly compressed digital contents, Adobe cooperation has been using TBB in many Creative Suite 5 software applications.

Juanbing Lu, who is a Chinese senior application engineer, has many experience of software performance optimization in the game field. In the 2010 China Game Business Conference, he discussed that how to use cross-platform Intel TBB threading library to improve game performance, how to improve or to build the game multi-core architecture and how to use TBB to achieve popular multi-core model [27]. An example in Napoleon: Total War, shows that the usage of TBB technology helps to reach the near optimization of game engine for multi-core PCs, and heavily improve the performance of the game with dual and even single core processors (up to 100% performance gain) [28].

Additionally, TBB technology also can be easily applied to complicated simulation games, such as essential outdoor scene - volumetric cloud render [29]. Because the traditional game structure is not based on multi-core system design, thus many serried yet integrant calculations could not be employed. So that game players experience few realistic efficient, mostly the 2D Texture is replaced approach. In fact, most game systems do not effectively use all of the CPU's resources. Generally, Intel Company (2010) declared that Threading Building Blocks is an efficient, proper and promising technology. Compared to other parallelism structures we mentioned above, TBB is the best choice of parallelism for C++ programmers as the combination with C++ standard library and flexible nested templates.

Basic Algorithms

The most noticeable contribution of TBB is the algorithm templates which are important for parallel programmers. Loop parallelization plays the primary role in the TBB. The templates, *parallel_for, parallel_reduce* and *parallel_scan*, are generic parallel algorithms. Furthermore, TBB supports a combination of nested parallelism to enable programmers to build larger parallel components by assembling smaller parallel components. Besides, TBB provides the fundamentals of recursion and task distribution. With these three main advantages, TBB can make the resulting program work efficiently.

Initializing the library: TBB components are defined in the tbb namespace. Any thread or task scheduler that uses any algorithm template from library should have an initialized tbb::task_scheduler_init object in main function. By default, the object would call constructor to do the initialization when it is defined and calls the destructor on termination of code. The example of initialization will be shown as Figure 10.

Programmers can adjust the number of threads in the thread pool in the initialization period by using tbb::task_scheduler_init object, such as tbb::task_scheduler_init init (10). If not specified, the default value is task_scheduler_init:: automatic, which is a default value automatically adjusted by the library according to the hardware and system. Task_scheduler_init is defined in *tbb/task_scheduler_init.h* file, Similarly, other TBB components are defined in their own header file, for instance, as mentioned in the follows, *parallel_for* is defined in *tbb/parallel_for.h*, and *parallel_reduce* is defined in *tbb/parallel_reduce.h*, respectively.

```
#include "tbb/task_scheduler_init.h"
using namespace tbb;

int main() {
task_scheduler_init init;
...
 return 0;
}
```

Figure 10: Initializing the library (Reinders, 2007).

However, the new TBB version 1.26 has improved the initialization method. The programmer can skip to program and concentrate on the main parallel part, because the library will automatically adjust the init values to the configuration of machine when parallel program prepares to run.

Parallelization loop: In this section, only two general parallel templates which are *parallel_for* and *parallel_reduce* will be introduced respectively in terms of necessary for this chapter. More algorithms and instructions can be found in Intel manual reference.

parallel_for: In order to better understand what the difference is between *parallel_for* and general for loop, here are two simple examples to explain how to use *parallel_for* and compare the running speed between different.

```
int main()
{
    for(int i=0; i<10; i++)
     {
        //process the time-consuming operations, here suspend 300ms to simulate;
         Sleep(300);
         std::cout << i;
     }
    return 0;
}
```

Figure 11: Original loop codes.

This is an original loop structure and running the codes of Figure 11 on i5 CPU U520 cost 3 seconds, the result of output is "0123456789".

Replace the original loop with *parallel_for* parallel the codes See Figure 12.

The template *blocked_range<T> (begin, end, GrainSize)*, constructed the entire iteration space from begin to end as a half-open interval [begin, end), while parallel_for function divides it into subspaces for each processor. The T specifies the value type, which is general int or *size_t*. The proper degree of breakdown, which is the right level of parallel tasks to break recursion in a problem, is called grain size. Programmers should take care of the value of grain size if specified.

Reinders (2007) [26] provided Figure 13 to illustrate how grain size affects the overhead in a system. On the left, the problem is broken into 4 pieces (4X), and on the right, the problem is broken into 36 pieces (36X). Both total light areas are equal and represent the same total work to be done yet the dark grey areas are unequal and mean the overhead. Obviously, the 36X case has a relatively higher proportion of overhead than the 4X case. Programmers need to ensure that there are enough pieces to provide works for all the processors. But if the program runs on a 32-core machine then the 4X division won't be able to use all the processors, even though it has lower overhead.

```
#include <tbb/task_scheduler_init.h>
#include <tbb/blocked_range.h>
#include <tbb/parallel_for.h>
struct OpFor{
    void operator()(const tbb::blocked_range<int> &r) const
     {
        for(int i=r.begin(); i!=r.end(); ++i)
         {
            //process the time-consuming operations, here suspend 300ms to simulate;
            Sleep(300);
            std::cout << i;
         }
     }
};

int main()
 {
     tbb::task_scheduler_init init;
     OpFor f;
     tbb::parallel_for(tbb::blocked_range<int>(0,10), f);
    return 0;
 }
```

Figure 12: Use of parallel for.

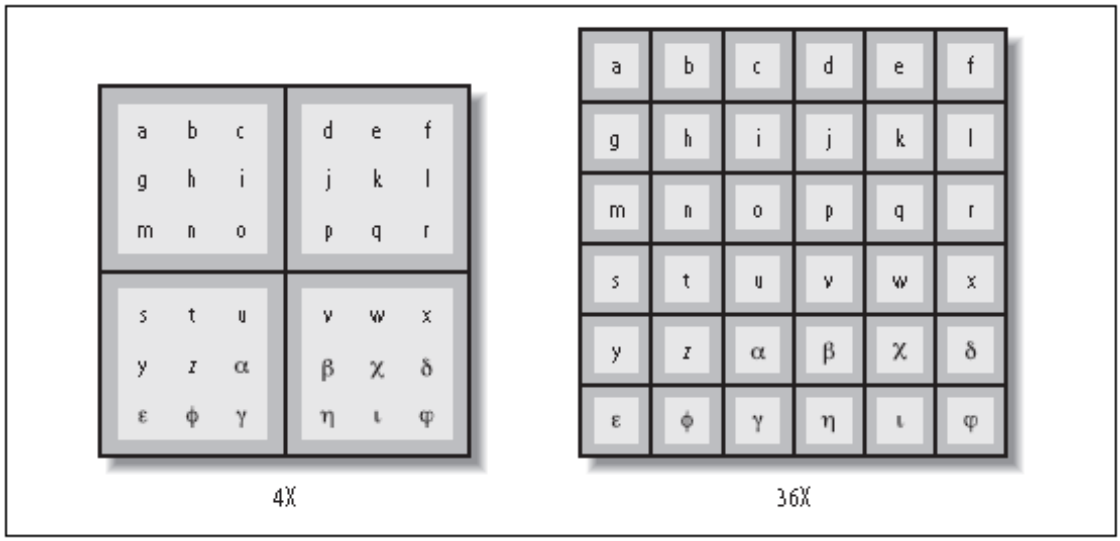


Figure 13: Packaging versus grain size, same workload (Reinders, 2007).

In Figure 12 program, the parallel loop omitted the grain size, yet a partitioner [13], which specifies how a loop template should partition its task among threads, should be supplied to the algorithm template, such as *parallel_for(blocked_range<int> (0,10), f, affinity_partitioner)*. If both grain size and partitioner are omitted like Figure 12, which means use default partitioner *auto_partitioner*. It will work out the grain size automatically for the current machine and workload and generally does a good job as far as my experience. TBB may try to divide several parts with the value of equation *(end-begin)/grainsize* to separately compute at the same time, but not really as that because the hardware is the first condition. Note that if there are more than 10,000 instructions per iteration, the value of grain size 1 will work okay, or else there may be a serious performance hit like Figure 13.

Running the Figure 12 program on the same computer cost 1563 millisecond. It is two times faster than the one without paralleling. But the output is not in sequence from 0 to 9, which means it is parallelized. Thus, if we would like to get the order result through paralleling, the lock may be limited before outputting. However, only reentrant loop body, which means there is no relation between last step of loop and next step of loop. For instance, loop body *output[i] = input[i-1] +input[i]+input[i+1]*, the

result of output at every step is not related to the result at last step. But if the loop body is relevant to the last step, it is called a non-re-executed loop. For example, recursion accumulation is a kind of non-re-executed loop. The parallel_for template could not be used with it. In this situation, the parallel_reduce template can be a solution. Then, the *parallel_reduce* template will solve this situation.

parallel_reduce: As mentioned above, there are situations that could not be paralleled by *parallel_for*. Typically, one of them is recursion summation, the original codes are shown in Figure 14.

```
int sum=0;
for(int i=0; i<10; i++)
      sum += i;
```

Figure 14: Original loop codes.

parallel_reduce will divide the *blacked_range* into several blocks to separate parallel calculation. The loop iterations in each block will be assigned based on *grainsize or partitioner*. Finally, the final result is obtained from the merger by accumulating the cumulative result of the small blocks. For instance, A+B+C+D+E+F may be calculated in serial as (((A+B)+C)+D)+E)+F),whereas the parallel calculation may compute as ((A+B+C)+(D+E+F)), which means the A+B+C and D+E+F may be calculated at the same time and then merge the result. The results of these two ways are the same but the parallel method may be faster than serial. A full example will be shown in Figure 15.

```
#include <tbb/task_scheduler_init.h>
#include <tbb/blocked_range.h>
#include <tbb/parallel_reduce.h>

struct SumFoo{
    int m_sum;

    void operator()(const tbb::blocked_range<int> &r)
     {
        for(int i=r.begin(); i!=r.end(); ++i)
         {
             Sleep(300);
             m_sum += i;
         }
     }


    // Separate a branch, ensure atomic operations
     SumFoo( OpSum& x, tbb::split) : m_sum(x.m_sum) {;}

      // merge all the branches
    void join(SumFoo& x){m_sum += x.m_sum;} |

     SumFoo():m_sum(0) {;}
};

int main()
{
     tbb::task_scheduler_init init;
     SumFoo y;
     tbb::parallel_reduce(tbb::blocked_range<int>(0,10), y, tbb::auto_partitioner());
    return 0;
}
```

Figure 15: A use of parallel reduce.

TBB defines the *parallel_reduce* in a little similar way of parallel_for as both of working space is loop through the *blocked_rang*. The main difference is that the copies of the body must be merged at the end, therefore the *operator ()* is not constant because of updating results. Another difference is that the struct/class *SumFoo* has a splitting constructor, and the *join ()* function will present for *parallel_reduce* to work.

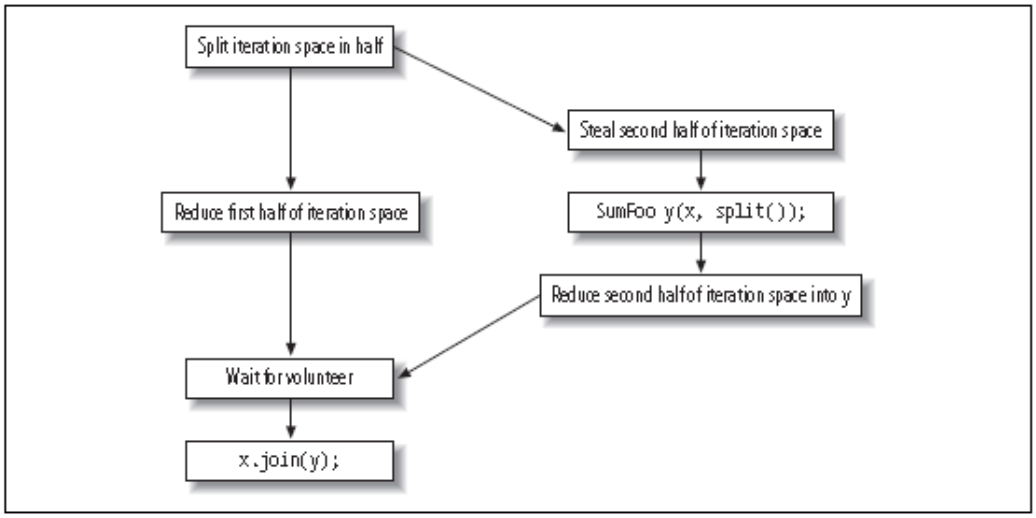


Figure 16: Split-join sequence (Reinders, 2007).

Figure 16 shows that when a worker thread is available, *parallel_reduce* will create the subtask for the processor by invoking the splitting constructor. If the task is completed, the *join* method will be used to accumulate the result of subtasks. In general, the splitting constructor does two things:

- Copies the read-only information to subtask to run the body loop
- Initializes the reduction variables to the members of the operations

The use of *grainsize and partitioner* are same as *parallel_for*. If the iterations do not take enough instructions, it is better to err on the side of large a grain size or use a partitioner to allow the library to guide the chunking of the range.

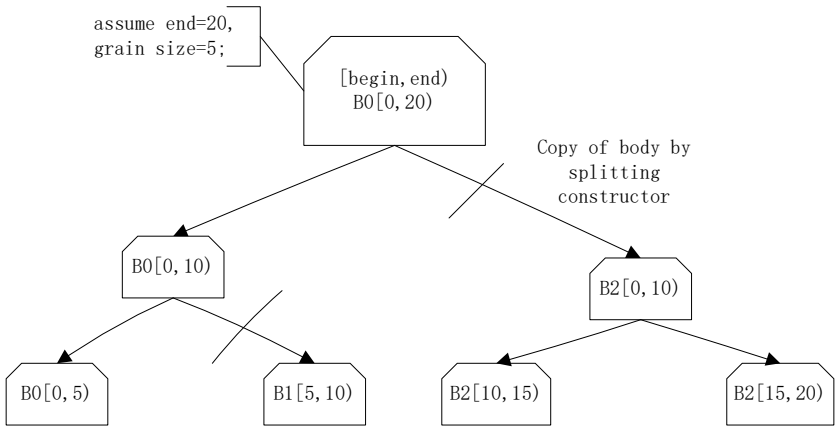


Figure 17: A Flow chart for parallel reduce.

A simple flow chart in Figure 17 describes a possible solution for the parallel computing of *parallel reduce*. The range of block is [0, 20) and the grain size is 5. Then the split constructor will divide half by half until the small block space is 5. It should be noted that the process of *parallel_reduce* may be like a binary tree yet indeterminate, because it will base on the machine 's current collecting information to guide the threads work. And the operation of join must satisfy from left to right operation as left. join(right). The computation will also be executed from left to right for each leaf. Then merge the result if there is a *join ()* method. In this example, one of solutions may be invoked as B0. join(B1) and B0. join(B2) [13].

Advanced algorithm

Besides, Threading Building Blocks also offer some algorithm templates for streams, such as *parallel_do*, *parallel_sort* and *pipeline*. In this section, *parallel_do* will be presented as an example. More details can be found in the Intel manual listed in the reference.

***parallel_do* template function:** For some loops, the loop body may add more iteration to do before the loop exits, or the loop will process work items in parallel, template *parallel_do* and *parallel_do_feeder* class can be referred.

The syntax of *parallel_do* is shown below:

```
template<typename InputIterator, typename Body>
void parallel_do(InputIterator first, InputIterator last, Body body[,
    task_group_context& group] );
```

The *parallel_do* is similar to the std:: for_each except it is parallelism. It applies the body function to parallelized process items in the half- open interval [first, last). *parallel_do* maintains an internal *parallel_do_feeder* object and additional work items can be added in body by this object if it has second argument. Note that it is not permitted to define both the one-argument and two-argument forms of *operator ()* is not permitted, only one form can be chosen in a body. More details are listed in Figure 18.

Table 15: parallel_do Requirements for Body B and its Argument Type T

Pseudo-Signature	Semantics
`B::operator() ( cv-qualifiers T& item, parallel_do_feeder<T>& feeder ) const` OR `B::operator() (cv-qualifiers T& item ) const`	Process `item`. Template `parallel_do` may concurrently invoke operator() for the same `this` but different `item`. The signature with feeder permits additional work items to be added.
`T( const T& )`	Copy a work item.
`~T::T()`	Destroy a work item.

Figure 18: parallel do requirement for Body B and its arguments type T.(Intel,2011).

It should be noted that the body of *operator ()* must be constant, and the argument type of constructor also must be constant. The following Figure 19 shows a sketch to define a two-argument form of *operator ()* with *parallel_do_feeder* object and add a new item by a method which is *parrel_do_feeder::add().*

Unlike *parallel_for or parallel_reduce*, in *parallel_do* template function, the loop iterations of each work item are assigned directly through the machine and there are no grainsize or partitioner to assign. *parallel_do* is designed to work on data structures that aren't random-access and thus aren't splitable in TBB terms. Each worker does the same loop body in the allocated range. A flow chart of process using *parallel_do* is shown in Figure 20.

```
struct MyBody {
        void operator()(item_t item,parallel_do_feeder<item_t>& feeder ) const{
 //for each new piece of work implied by item do
              {
                 item_t new_item = initializer;
                   feeder.add(new_item);
              }
    }
};
```

Figure 19: sketch for a body with the two-argument form of operator ().

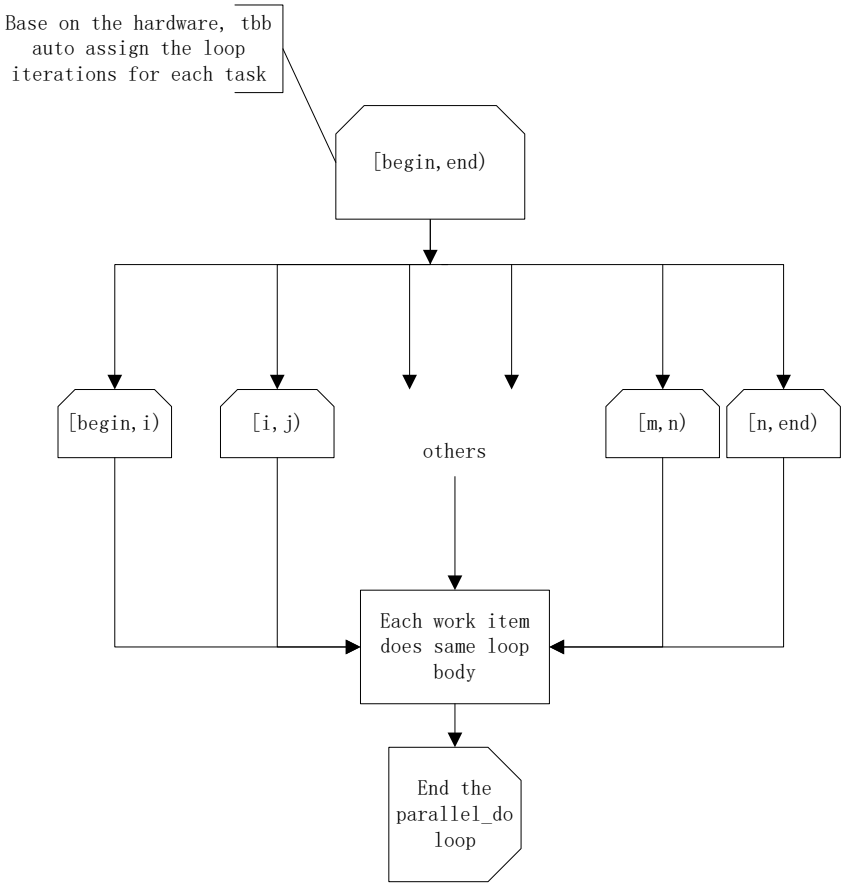


Figure 20: A flow chart of parallel do.

A simple example for *parallel_do:* This example [30] is to describe how a vector from C++ STL is paralleled by *parallel_do*. Figure 21 is a simple class, which is to add the value of private member by the member function *activate ()*.

If there is a *vector* container and it assembles 100 objects of class *OpPal*. Now in order to operate these objects in parallel, *parallel_do* will be exampled. However, the *parallel_do operator ()* must be const, which means the class object cannot be directly changed in operator body. Therefore, an intermediate class should be created as a connector which is used to create an instance to be paralleled. In other words, the second object which is *argument_type* member of *operator ()* will be used to activate the loop body. See Figure 22.

Finally, the *parallel_do* template function will activate all objects of *vector<OpPal>* type by class *OpWorker* object. See Figure 23.

```
// An unexciting class that we'll use as an example thing to be processed below.
class OpPal {
public:
    OpPal() : value(0) {
    }

    void activate() {
        ++value;
    }

    int get_value() {
        return value;
    }

private:
    int value;
};
```

Figure 21: A class used for parallel do.

```
class OpWorker {
public:
    // The const qualifier here means that calling this method doesn't
    // change the OpWorker object, but it can still change
    // the OpPal.
    void operator()(OpPal& operation) const {
        operation.activate();
    }
};
```

Figure 22: A class used to active object of class OpPal.

```
int main(int argc, char *argv[]) {

    tbb::task_scheduler_init init;

    vector<OpPal> things;

    // Add some things to our collection.
    for (int i = 0; i < 100; ++i) {
        things new_thing;
        things.push_back(new_thing);
    }
 // Activate all the things.
    OpWorker worker;
    tbb::parallel_do(things.begin(), things.end(), worker);
    return 0;
}
```

Figure 23: Example of using parallel do.

Notes on *parallel_do* scaling: The TBB manual states that use of *parallel_do* is not scalable if all of the items come from an input stream typically that does not support random access, such as *list*. At the end of this section, according to the parallelization methodology of TBB, here is a quick summary of what should be noted as follows.

- *Decomposes tasks* - analyzing the problems, find the tasks that can run concurrently.
- *Scaling* - considering overhead of managing the parallel program so that there are enough tasks to keep all the processors busy.
- *Algorithm and patterns* - how to use a proper algorithm for a program to maximize the running speed.
- *Correctness* - how the implicit synchronization inherent in TBB helps minimize the use of lock. If it must be locked, take care of deadlock and race conditions, which result from errors involving locks.
- *Random access* - thinking in terms of parallelism and concurrency (tasks that can be partition), the data structure of program should better be random accessible, so that it will maximize to reduce the time for accessing data.

Section 4: Programming

This program is based on Ye Li's inchoate cell model that is to simulate cell migration with static ellipsoid individuals to develop. However, this chapter not only changes to model the dynamic ellipsoids with motion and rotation but also uses parallel computation to save the running time.

Design Structure

Before designing the structure of a program, a programmer needs to know whether the program could be paralleled, as well as how to achieve the top efficiency, which are two key elements of parallelism. If the program does not satisfy requirements of parallel functions, it will not be effective. If the program does not satisfy the requirements of parallel functions, using parallel algorithms is not effective. As the summary of TBB above, the code and data structures will be discussed as follows.

Task analysis: Based on the first requirement of TBB functions, a well classification for the program is necessary before coding.

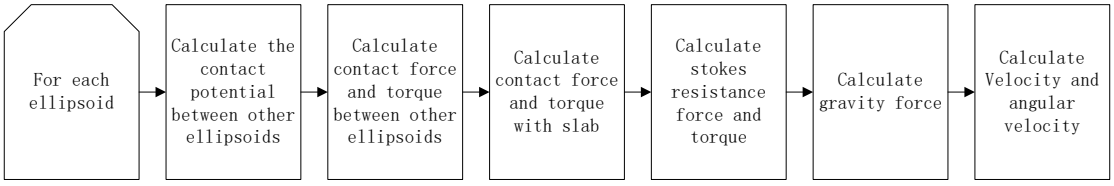


Figure 24: A list of calculation.

According to the stress analysis for each ellipsoid, Figure 24 is to describe what computations for every ellipsoid should be done in every iteration step. Based on the methodologies of each part in these computations, the first two parts are the most time-consuming calculations because of its the complexity of computing and the amount of computation. For instance, if there are 200 ellipsoids, each ellipsoid must calculate with other 199 ellipsoids. Thus, the number that 200 multiplied by 199 is the number of this kind of calculation to be run in one step times at one step will be run. So, the best arrangement of struct may use double nest loops. The extern loop should include all of these calculations in order and the inner loop is in charge of calculating potential, force and torque between ellipsoids. Note that the inner loop range is a copy of extern loop range. In order to distinguish the 200 ellipsoids of extern loop and other copied ellipsoids, each ellipsoid in extern loop is called marked ellipsoid.

After that, it is necessary to check whether every task could run concurrently. For the inner loop, according to the equations of forces and torques, the contact results of marked ellipsoids are only related to the directional distance between this ellipsoid and other 199 ellipsoids, and their direction of body axes if assume all ellipsoids are of same radii and mass. In other words, the contact results should be cumulated repeatedly through the calculation with 199 ellipsoids. Nevertheless, each calculation between marked ellipsoid and the 199 ellipsoids is individual as the computation only uses the vector position and normalized direction matrix of each relational ellipsoid. In fact, they are not transformed until the positions of ellipsoids are updated at the end of each time step. Therefore, the calculation between ellipsoids can be concurrently yet should be accumulated as a result of

each marked ellipsoid in a time step. And then for the external loop, the value of new velocity and angular velocity is depended on. That means it does the accumulation by all kinds of forces and torques, including the force and torque between ellipsoids which get from the inner loop, between ellipsoid and slab, stokes resistance and gravity.

As mentioned above, the calculation of force and torque between an ellipsoid and a slab is the same as the calculation among ellipsoids. So, the result with slab could be computed synchronously in a time step. The Gravity force for an ellipsoid is a constant value if the radii and density is fixed, obviously. The final Stokes force and torque only relate to velocity and angular velocity of marked ellipsoid at last time step if denote the dynamic viscosity and $\mathbf{K}^t, \mathbf{K}^r$ are constant parameters. Thus, all kinds of force and torque can be computed by separate marked ellipsoid. In the end, sum up these calculated forces and torques, and base on the equations of motion to compute the new velocity and angular velocity of marked ellipsoids at this time can be computed. Of course, that also can be calculated concurrently for marked ellipsoids.

Principally, the primary calculation tasks that have been listed in Figure 24 for each ellipsoid can run at the same time. Additionally, for every ellipsoid, each step contains abundant works especially in the inner loop if the number of particles is enormous. And adding a lock may be unnecessary, because there is no resource connection, as all calculations only have the relation with vector position and direction of each ellipsoid. According to Ye Li's original model calculation, the contact potential of each ellipsoid will be updated individually as a private member property after calculating the contact potential. However, for paralleling, when ellipsoid A and B's interaction was being computed at the same time as ellipsoid A and C's interaction, both threads would be updating the same contact potential of ellipsoid A and would get a nonsense result. It is a classic example of a race condition problem. Thereby, contact potential would be not a property of an ellipsoid; it's a temporary result when computing the interaction between two ellipsoids. (This race condition was inspected in my program by Sampson in 2010 [30].

Choose a container in STL: According to the requirements of paralleling, data structure plays an important role in the parallelism. TBB parallel template functions are unhappy to support sequential access because of parallelism and concurrent. Typically, in C++ STL, vector and list are the representative containers of random access and sequential access, respectively [31]. Container lists implement doubly linked list. It only supports accessing from first to last or reversed because it stands for a non-contiguous memory area and its elements linked by a pair of pointers which point to the first element and the end of the list. Such that if accessing arbitrary element like parallelism, the time for accessing data will take up the parallel efficiency and even result no working for paralleling. For instance, *parallel_reduce* template function does not work in list container in terms of splitting constructor could not get where the half pointer is in the block range. Actually, the TBB template functions use block range may be improper to parallel data in list container. However, vector container implements an array with fast random access and an ability to automatically resize when appending elements. It is assigned in a sequential memory and likes a dynamic array with index number. That will help fast random access and provide efficient paralleling if only a few operations with insert and delete in the any place of container. Therefore, vectors should be used in this chapter as efficient data access for parallel computation with TBB technology.

Choose parallel algorithms: To achieve the greatest speedup, choosing appropriate algorithms for suitable calculation is the dominant element. After task analysis, it is clear that every marked ellipsoid is a worker. Image that each ellipsoid owns a work list like Figure 24, many lists compose the whole working for all ellipsoids. Based on the usual parallel functions, *parallel_do* template function may be used to calculate the velocity and angular velocity in terms of work stream. The velocity and angular velocity depend on the total force and total torque to compute, so every kind of force and torque should be added together and changed. Note that the *operator()* method must be const, but the argument_type member of *operator()* can be changed to control the loop body as a connector. If more objects were added to the loop body, the *parallel_do_feeder* could help programmer achieve confidence.

After choosing the extern loop, the inner loop also is a major part of paralleling. Compared with extern loop, the inner loop pays attention to accumulating the contact force and torque of one marked ellipsoid and other 199 ellipsoids. It is easy to think about the parallel_reduce template function. Not only its loop body is not const but also it can merge the results of other blocks. Thus, the *parallel_reduce* may be the better way to parallel compute the contact force and torque between ellipsoids. Finally, after all of the computing, the new position and direction of each ellipsoid will be updated in order to get into the next time count. The length of this loop depends on the number of ellipsoids. So, if the number of ellipsoids is plenty, it also could be paralleled with *parallel_do*.

Based on parallelization method, the parallel structure could be expressed as Figure 25. As mentioned above, the calculation will depend on the double nest loop to parallel compute, *parallel_do* and *parallel_reduce* are used in the extern loop and inner loop, respectively. And the third loop which updates data of each ellipsoid also implements *parallel_do* to achieve speedup.

Overview of programming

According to the parallel structure, the whole structure of this chapter should stay the same. And the general program structure is shown in Figure 26.

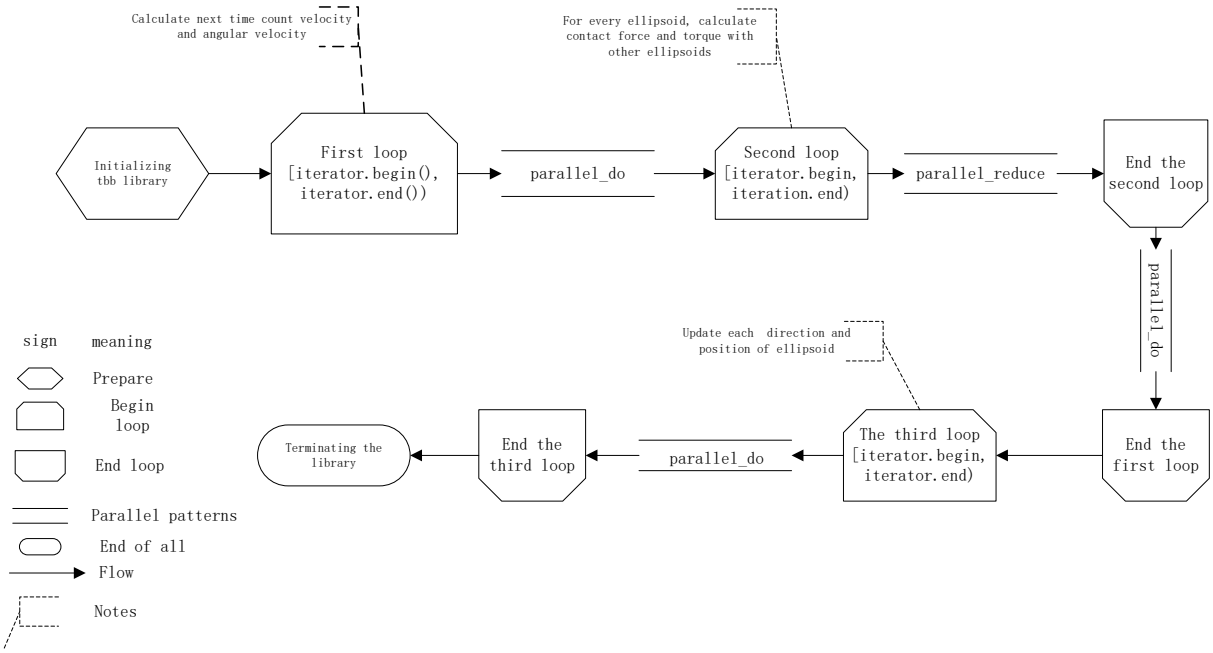


Figure 25: The structure of parallelism in this chapter.

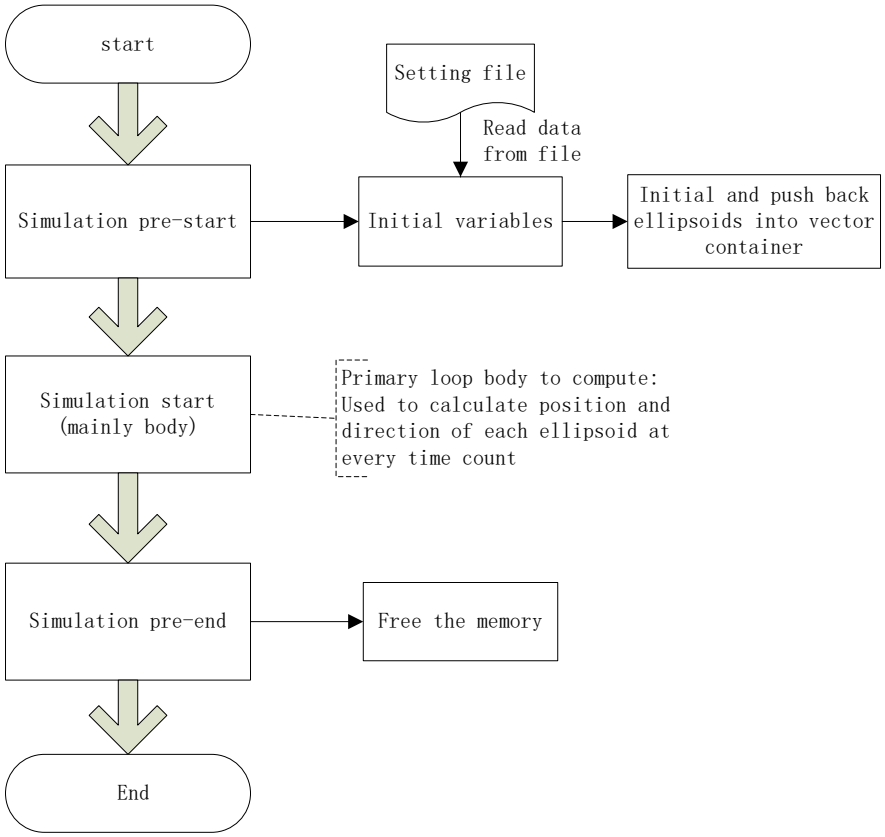


Figure 26: General structure of program.

After starting the program, firstly is preparing of simulation. All of the variables should be initialized, such as number of ellipsoids, vector point and normal of slab, building ellipsoids, and so on. And then simulation begins, it is the primary part to calculate position and direction of each ellipsoid at every step through complex but parallel computation and output the data to files in order to visualize. After completion of simulation, all of the memory resource should be kept free and exit the program. The main body of this program is the simulation part. The detailed structure can be seen in Figure 27. The time count is used to control the loop times of calculation, one time count represents the delta time for ellipsoidal motion. Programmers can set the values of parameters to save data to data files. The following parallel computing has been mentioned above except they are the loop body of time count. Note that the update loop should be paralleled at end of every time count after computing double nest loop. The reason why the time count loop is not paralleled is that it does not satisfy the condition of concurrent. The result of each ellipsoid at every time count is relevant to the new result of each ellipsoid at next time step in terms of the position, direction, velocity and angular velocity. Therefore, only this big loop could not be paralleled with TBB.

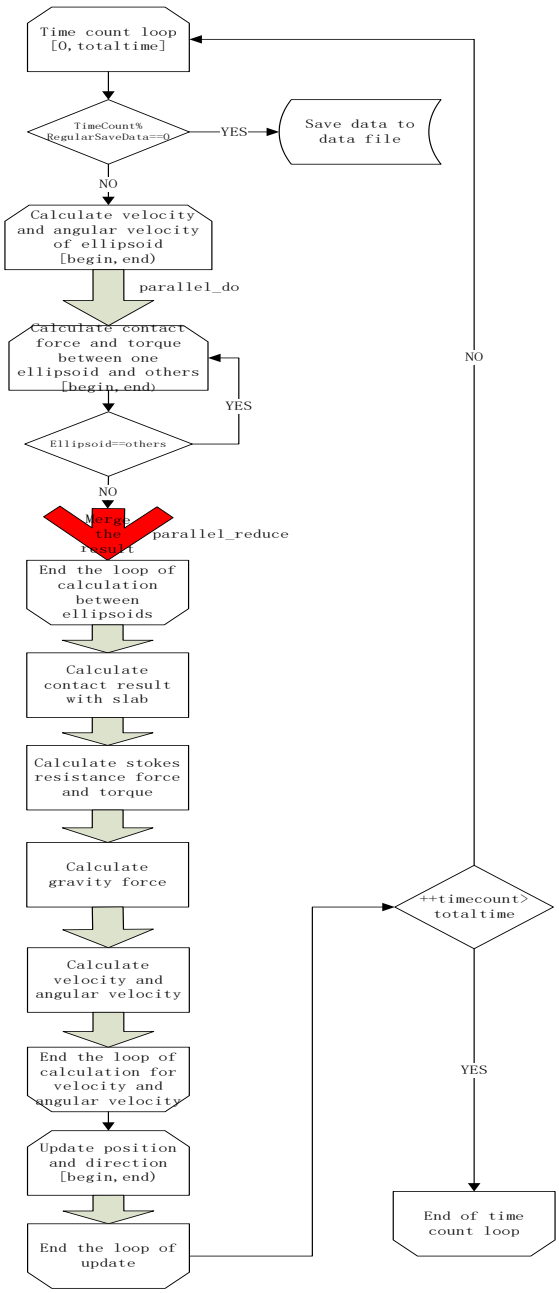


Figure 27: Structure of simulation.

Coding

This program is based on C++ STL to develop. According to the analysis before, coding this chapter is clarity. Nine classes are used in this program to complete the simulation as mentioned above. *The main()* function only supports an entry to get into the simulation class. The initialization, calling, parallelism, output and others are realized through this class. It seems like a port to collect all of the data from returned. The main structure of simulation class is like Figure 26 and the primary function is the *simulation start()*. That includes the calculating flow like Figure 27 to access other class. The ellipsoid class is the other most important part in this program because it is a computing class. All of the calculations, such as every kind of force and torque, acceleration and angular acceleration, velocity and angular velocity, and the final position and direction, will be computed in this class. The Parallel Calculate class is used to realize *parallel_reduc*e function. Then, the vector, vector4, matrix, and matrix4 are the definition of all computing signs or type of parameters. The slab is only to define the property of slab such as to get a point on the slab and to get a normal vector of the slab. The final class is utility class which contains mostly const parameters' definitions. The first three classes are particularly introduced as follows.

Simulation class: As mentioned, this class is the best performance of program structure. Except the general initialization, this class defined a container, *vector<ellipsoid*> m_Ellipsoids*, to store the data of pointer which points to the object of class ellipsoid and defined a pointer array that points to the class ellipsoid. Definitely, all the members of an ellipsoid should be assigned values at first and when new these ellipsoids, each pointer of class ellipsoid will be pushed back to the vector container. Then the container will auto call the copy constructor and the function of assignment operator from class ellipsoid to build each ellipsoid. Besides, two internal classes, calculation class and update class are included in the simulation.h. These two classes are used to apply *parallel_do* template function because the loop body of *parallel_do* must be const but the argument type could be changed. Thus, they are the middle classes as factor to call the calculation functions which belong to simulation class. A part code of *parallel_do* will be shown below.

Simulation.h

```
class Simulation
{
public:
void CalculateVelocityAndAngularVelocity(Ellipsoid* ellipsoid);
class Calculation
{
public:
Calculation( Simulation& simulation):m_Simulation(simulation){}
void operator()(Ellipsoid* ellipsoid) const {
m_Simulation.CalculateVelocityAndAngularVelocity(ellipsoid);
      }
private:
       Simulation& m_Simulation;
};
…
…
};
```

Simulation.cpp-> SimulationStart()

```
// parallel calculate force and torque ,then update velocity and angular velocity;
```

```
{
Calculation ParaCal(*this);
tbb::parallel_do( m_Ellipsoids.begin(), m_Ellipsoids.end(),ParaCal);
}
```

And then, the member function CalculateVelocityAndAngularVelocity will be invoked by paralleling. Some others function such as update, writing data to file, also are realized in this class. The completed code is written in the *simulation.h* and simulation.cpp files.

Ellipsoid class: As said earlier, this is a function class, total computing functions are here and the properties of each ellipsoid are also defined in this class through the default constructor, argument type constructor to initialize, copy constructor and assignment operator function. Coding every kind of calculating function is the theory of each computation. These constructors and operator function must be included in ellipsoid class in terms of building ellipsoid by *pushback()* function.

ParallelCalculate class: This class is only used for *parallel_reduce* function because the loop body must be put into the *operator()* function. Based on the theory of *parallel_reduce*, the class should have the split part to merge the results, default constructor to initialize, and the operator function to parallel loop body. Thus, part of the header file and source file could be coded as follows.

ParallelCalculate.h

```
class ParallelCalculate
{
public:
void operator()(const tbb::blocked_range<size_t> &Para); // TBB body operator

// parallel calculate with split
ParallelCalculate( ParallelCalculate& x, tbb::split):currentP( x.currentP),it_ellipsoid(x.it_ellipsoid),force(x.force),torque(x.torque){ }
void join( const ParallelCalculate& y);
// the constructor and initialization
ParallelCalculate(const vector<Ellipsoid*>& ellipsoid, Ellipsoid* the_it):currentP(ellipsoid),it_ellipsoid(the_it),force(0.0,0.0,0.0),torque(0.0,0.0,0.0){ }
// used for return result
Vector GetForce(){return force;}
Vector GetTorque(){return torque;}
private:
 const vector<Ellipsoid*> currentP; // for the m_Ellipsoids
 Ellipsoid* it_ellipsoid; // the current object pointer of the big loop
 Vector force;
 Vector torque;
};
```

ParallelCalculate.cpp

```
void ParallelCalculate::operator ()(const tbb::blocked_range<size_t> &Para)
{
```

```
for( size_t i = Para.begin(); i != Para.end(); ++i )
{
// the other pointer used for calculate the force and torque between two ellipsoids
vector<Ellipsoid*>::const_iterator otherEllipsoid = currentP.begin() + i;
if ( it_ellipsoid != (*otherEllipsoid) ) // if the two elipsoids are not same, then calculate.
{ // ... calculating part...
Vector ContactForce =it_ellipsoid->CalculateContactForceV2( lemda, contactPotential, G_Inverse, (*(*otherEllipsoid)) );
Vector ContactTorque =it_ellipsoid->CalculateContactTorqueV2( lemda, contactPotential, G_Inverse,(*(*otherEllipsoid)) );
force = force + ContactForce;
torque = torque + ContactTorque;
}// end for if
}//end for for-loop
}
// add the split partes together
void ParallelCalculate::join( const ParallelCalculate& y)
{
force = force + y.force; torque = torque + y.torque;
}
```

Simulation.cpp

```
// parallel compute the force and torque between ellipsoids
{
ParallelCalculate m_P(m_Ellipsoids,ellipsoid);
tbb::blocked_range<size_t> Range(0, m_Ellipsoids.size());
tbb::parallel_reduce(Range,m_P); // use tbb::auto_partitioner(); default grain size is 1.
TotalForce += m_P.GetForce(); // add the returned result to the total
TotalTorque += m_P.GetTorque ();
}
```

In simulation.cpp file, when the object of ParallelCalculate class is building, the default constructor will assign the values of parameters to its own members. And the *m_ellipsoids.size()* will return an integer which is the length of vector container, then the block range is from 0 to the returned value. TBB library will auto divide the range in half into serial small blocks in terms of information of collection by machine. At the same time, the splitting constructor will copy the necessary information to initialize the coping vector. If there is no split anymore, the *join()* method will merge the results by operation as left.join(right). The process of merging is like traversing a binary tree from left leaf to right leaf.

Compiling environment and visualization: Of cause, the TBB "library and include file" should be included in the "projects and solutions->VC++ directories, respectively. And the output data will be stored in the folder that called data under the solution directories. The visual code is modified based on a machine version which develops with OpenGL. The general structure of this visual program is that, reading a data from the output data file and drawing each ellipsoid through position and direction in order. The slabs are permitted to draw between glBegin(GL_QUADS) and glEnd() sentences. OpenGL supports programmers to directly draw four vertexes to consist of a quad, such as drawing a vertex with glVertex3f((3.1f), (2.5f), (-3.1f)) and to color these vertexes like glColor4f(0.5f,0.0f, 0.0f,0.3)(0.3 is the alpha value.) Besides, as DirectX, OpenGL also provides many other

methods to draw different polygon, such as GL_TRANGLES, GL_POINTS, GL_LINES, GL_POLYGON and so on. Definitely, others as transformation, lighting, scaling, and blending, also should be set well.

After coding the program, debugging is really important in terms of many physical equations with const parameters. Then how to calculate the scientific numerical values rather than casual assignment for parameters will be discussed in the next section.

Section 5: Debugging with Dimensional Analysis

Definition and significance of dimensional analysis

Dimensional analysis is used to measure physical variables, it is based on Joseph Fourier's idea that a physical laws like *F=ma* must be independent of its units to present. In physics, it is popular to define as" The dimension of a physical quantity is the combination of the basic physical dimensions (usually mass, length, time, electric charge, and temperature) which describe it; for example, speed has the dimension length per unit time, and may be measured in meters per second, miles per hour, or other units" [32].

A straightforward practical consequence is that any meaningful equations (and any inequality) must keep homogeneity for the left and right sides of equality. Checking this is the basic way to perform dimensional analysis. The other major application is that reasonable hypotheses about complex physical situations can be tested by experiment or by more developed theories of phenomena. That should categorize types of physical quantities and units based on their definitions to find the dominant factors. Likewise, for this chapter, on the one hand, it could confirm the correctness of equations; on the other hand, dimensional analysis could guide programmer faster get proper values of unknown constant coefficients that are also the dominant factors in many complex formulas.

The methods of dimensional analysis

ZhuoRu [15] introduced the approaches that are the following 6 steps to analyze the relation between dimensionless equation and original equation.

Step 1. Formulate appropriate basic reference dimensions

Choosing suitable reference dimensions for a project can help programmers clearer understand the meaning of each variable and equation. This chapter takes international standard units to constitute the basic reference dimensions.

$$\Delta x \to (m); \Delta t \to (s); \rho_c \to (kg/m^3)$$

Step 2. List all the variables which relate with the physical quantities and using basic reference units to express their dimensions

According to the definition or defined equation of each variable to find which variable has unit, such as the unit of length is meter, or which is constant without unit, such as PI. The following example is to show to find the dimension of a variable from an equation and use basic reference dimensions to express. Given a definition, distance S equates to velocity V multiplied by time T. (S=V.T) The unit of velocity is meter/second, and the unit of time is second, thus the physical quantity of distance is meter/second multiply by second, the result is meter. So, the dimension of distance is Δx .

Step 3. Use the variable divide its dimension which is composed of basic reference units (in step 1) to describe dimensionless variables, as $(\dim)\tilde{x} = \frac{x}{\text{dimension}}$

E.g. For distance S, its dimensionless variable could be described as $\tilde{S} = \frac{S}{\Delta x}$ or $S = \tilde{S} \cdot \Delta x$

Step 4. Take the dimensionless variables into the original equation and get the new dimensionless equation (check the homogeneity of equation)

For instance, given a free-falling equation for Height $H = \frac{1}{2} g t^2$, from the definition of height, it is easy to say the dimension of height is same with the distance S. Then, for the right side of the equation, 1/2 is a constant, and the unit of gravity acceleration g is (meter/second)/second, and time t is second. Thus combined the right-side variables together are that (meter/second)/second multiply by square second, the result is meter. So, both the dimension of the left side and right side is Δx (Satisfy the homogeneity). Using formula to explain is that

$$\left.\begin{aligned} H &= \tilde{H}\cdot\Delta x \\ g &= \frac{\tilde{g}\cdot\Delta x}{\Delta t^2} \\ t &= \tilde{t}\cdot\Delta t \\ H &= \frac{1}{2}gt^2 \end{aligned}\right\} \Rightarrow \frac{1}{2}\cdot\frac{\tilde{g}\cdot\Delta x}{\Delta t^2}\cdot\tilde{t}^2\cdot\Delta t^2 = \tilde{H}\cdot\Delta x$$

, simplified it, there is the dimensionless equation $\tilde{H} = \frac{1}{2}\tilde{g}\tilde{t}^2$.

Step 5. Find the dominant factors of equations through dimensionless equation and dimensionless variables expression and analyze their relation

From the above equation, the dominant factors for $\tilde{H}$ are the $\tilde{g}$ *and* $\tilde{t}$, if g=10 m/s/s and t=1s, then the dominant dimension factors are Δx and Δt. This example is very simple, but if it is a complex situation (introduced later), this method will become useful.

Step 6. Test the value of basic dimension, and get the value of coefficient

Assign the different values for this basic dimension, different results will be presented. Basically, people agree that using a standard velocity C which is the speed of sound in fluid to express the basic reference velocity. The velocity of sound C depends on what the medium is, here

assume that $C = \frac{\Delta x}{\Delta t} = 2000m/s$. Then if denote $\Delta x = 2\times10^{-2}$, so $\Delta t = 1\times10^{-5}$, based on the formula in step 4, the dimensionless

variables are $\tilde{g} = \frac{g\cdot\Delta t^2}{\Delta x} = \frac{10\times1\times10^{-10}}{2\times10^{-2}} = 5\times10^{-8}$, $\tilde{t} = \frac{t}{\Delta t} = 1\times10^{5}$, $\tilde{H} = 2.5\times10^{2}$. That means the height has 2.5×10^{2}

times as higher as Δx, translate to the physical quantities is $2.5\times10^{2}\times2\times10^{-2} = 5m$. (Check with original equation, it is same). More usually, programmer would like to set $\tilde{t} = 1$; it is only calculating one delta time result. So, with regard to one delta time the final result is $\tilde{H} = 0.5\times5\times10^{-8}\times1^2 = 2.5\times10^{-8}$. It represents that at the first 1×10^{-5} second, the body falls $2.5\times10^{-8}\times2\times10^{-2} = 5\times10^{-10}(m)$.

Apply to this chapter

According to the above method, that could be applied to this chapter because of many complex equations which discussed in theory. First of all, based on the first 3 steps, which are summarized, list the basic reference dimensions, describe relation between general physical quantities and these basic dimensions, and formulate the dimensionless variables. From the definition, expression and physical unit of each physical quantity that related to equations of this chapter, the dimensionless common parameters could be described with basic reference dimensions.

Table 1: Describe for common dimensionless parameters.

Physical parameter (x)	Definition or SI physical unit or expression	Dimensionless parameter ($\tilde{x}$)
Radius (R) or distance (S)	Include parameters which units are meter (m)	$\tilde{\mathbf{S}} = \frac{\mathbf{S}}{\Delta x}$ or $\tilde{R} = \frac{R}{\Delta x}$
Velocity (υ)	The unit is meter/second	$\tilde{\mathbf{v}} = \frac{\mathbf{v}\cdot\Delta t}{\Delta x}$

Acceleration (a)	The unit is meter/second/second	
Angular velocity ( )	It is radian per time (π/t)	$\tilde{\boldsymbol{\omega}} = \boldsymbol{\omega} \cdot \Delta t$
Angular acceleration (a_ω)	It is angular velocity divide time (π/t^2)	$\tilde{\mathbf{a}}_\omega = \mathbf{a}_\omega \cdot \Delta t^2$
The mass of ellipsoid (j)	$m_e = \frac{4}{3}\pi\rho_e \cdot r_1 r_2 r_3$	
Force (F)	Unit is Newton ($F = ma$)	$\tilde{\mathbf{F}} = \frac{\mathbf{F} \cdot \Delta t^2}{\rho_c \cdot \Delta x^4}$
Torque ($\boldsymbol{\tau}$)	$\boldsymbol{\tau} = \mathbf{F} \times \mathbf{s}$ (S is the distance)	$\tilde{\boldsymbol{\tau}} = \frac{\boldsymbol{\tau} \cdot \Delta t^2}{\rho_c \cdot \Delta x^5}$
Young's modulus (E)	The unit is N/m^2	$\tilde{E} = \frac{E \Delta t^2}{\rho_c \cdot \Delta x^2}$
Dynamic viscosity (μ)	The unit Pascal-second (Pa.s)or ($N \cdot s/m^2$)	$\tilde{\mu} = \frac{\mu \cdot \Delta t}{\rho_c \cdot \Delta x^2}$
Ellipsoid (A) and its Inverse ($\mathbf{A}^{-1}$)	$\mathbf{A} = a_1^{-2}\mathbf{u}_1 \otimes \mathbf{u}_1 + a_2^{-2}\mathbf{u}_2 \otimes \mathbf{u}_2 + a_3^{-2}\mathbf{u}_3 \otimes \mathbf{u}_3$ $\mathbf{A}^{-1} = a_1^{2}\mathbf{u}_1 \otimes \mathbf{u}_1 + a_2^{2}\mathbf{u}_2 \otimes \mathbf{u}_2 + a_3^{2}\mathbf{u}_3 \otimes \mathbf{u}_3$ (U is unit vector)	$\tilde{\mathbf{A}} = \mathbf{A} \cdot \Delta x^2$ $\tilde{\mathbf{A}}^{-1} = \frac{\mathbf{A}^{-1}}{\Delta x^2}$

Dimensional analysis of contact force and contact torque: Based on Contact Force and Torque equations (44)(45), and the definition of each variable in these equations, except what common physical quantities have tabled above, other particular dimensionless variables in contact force and torque could be expressed as Table 2.

Table 2: The particular dimensionless variables in contact force and torque.

Variable definition or expression	Dimensionless variable
$K = 4\Big/\left[3\pi\left(\frac{1-\nu_1^2}{\pi E_1} + \frac{1-\nu_2^2}{\pi E_2}\right)\right]$ In which ν is the Poisson ratio, and E is the Young's' modulus, here ν =1/3, square it will be smaller, thus $K = 4\Big/\left[3\left(\frac{1}{E_1} + \frac{1}{E_2}\right)\right]$	$\tilde{K} = \frac{K \cdot \Delta t^2}{\rho_c \cdot \Delta x^2}$
$\mathbf{G}(\lambda)^{-1} = \lambda\mathbf{B} + (1-\lambda)\mathbf{A}$ In which A and B are ellipsoids, λ is constant.	$\tilde{\mathbf{G}}(\lambda)^{-1} = \mathbf{G}(\lambda)^{-1} \cdot \Delta x^2$
$\Phi(\mathbf{A},\mathbf{B},\mathbf{s}) = f(\lambda_0) = \lambda_0(1-\lambda_0)\mathbf{s}^T\mathbf{G}^{-1}(\lambda_0)\mathbf{s}$ Where S is the vector distance, so both dimension of $\mathbf{s}^T$ and $\mathbf{s}$ is Δx .	$K = \frac{4E}{3(1-\nu^2)}$

According to Equations (44)(45), take the dimensionless variables into them, the dimensionless equations for contact force and torque could be written as followed.

$$\frac{\tilde{\mathbf{F}}_c \cdot \rho_c \cdot \Delta x^4}{\Delta t^2} = \frac{\tilde{K} \cdot \rho_c . \Delta x^2}{\Delta t^2} \cdot \left[\left(\tilde{a}_1 \Delta x \cdot \tilde{a}_2 \Delta x \cdot \tilde{a}_3 \Delta x \right)^{\frac{1}{3}} + \left(\tilde{b}_1 \Delta x \cdot \tilde{b}_2 \Delta x \cdot \tilde{b}_3 \Delta x \right)^{\frac{1}{3}} \right]^2$$
$$\cdot (\tilde{a}_1 \Delta x \cdot \tilde{a}_2 \Delta x \cdot \tilde{a}_3 \Delta x \cdot \tilde{b}_1 \Delta x \cdot \tilde{b}_2 \Delta x \cdot \tilde{b}_3 \Delta x)^{\frac{1}{6}}$$
$$\cdot \left(1 - \sqrt{\Phi}\right)^{\frac{3}{2}} \cdot \Phi^{-\frac{1}{2}} \cdot \left(\lambda_0 \left(1 - \lambda_0\right) \cdot \frac{\tilde{\mathbf{G}}^{-1}\left(\lambda_0\right)}{\Delta x^2} \cdot \tilde{\mathbf{s}} \cdot \Delta x \right) \tag{77}$$

$$\frac{\tilde{\mathbf{T}}_c \rho_c \Delta x^5}{\Delta t^2} = -\frac{\tilde{k} \cdot \rho_c \cdot \Delta x^2}{\Delta t^2} \left[\left(\tilde{a}_1 \Delta x \cdot \tilde{a}_2 \Delta x \cdot \tilde{a}_3 \Delta x \right)^{\frac{1}{3}} + \left(\tilde{b}_1 \Delta x \cdot \tilde{b}_2 \Delta x \cdot \tilde{b}_3 \Delta x \right)^{\frac{1}{3}} \right]^2$$
$$\cdot (\tilde{a}_1 \Delta x \cdot \tilde{a}_2 \Delta x \cdot \tilde{a}_3 \Delta x \cdot \tilde{b}_1 \Delta x \cdot \tilde{b}_2 \Delta x \cdot \tilde{b}_3 \Delta x)^{\frac{1}{6}} \cdot \lambda_0 \left(1 - \lambda_0\right)^2$$
$$\cdot \left(1 - \sqrt{\Phi}\right)^{\frac{3}{2}} \cdot \Phi^{-\frac{1}{2}} \left(\tilde{\mathbf{A}}^{-1} \Delta x^2 \frac{\tilde{\mathbf{G}}^{-1}\left(\lambda_0\right)}{\Delta x^2} \cdot \tilde{\mathbf{s}} \Delta x \right) \times \left[\frac{\tilde{\mathbf{G}}^{-1}\left(\lambda_0\right)}{\Delta x^2} \cdot \tilde{\mathbf{s}} \Delta x \right] \tag{78}$$

Simplify them,

$$\tilde{\mathbf{F}}_c = \tilde{K} \left[\left(\tilde{a}_1 \cdot \tilde{a}_2 \cdot \tilde{a}_3 \right)^{\frac{1}{3}} + \left(\tilde{b}_1 \cdot \tilde{b}_2 \cdot \tilde{b}_3 \right)^{\frac{1}{3}} \right]^2 \left(\tilde{a}_1 \tilde{a}_2 \tilde{a}_3 \tilde{b}_1 \tilde{b}_2 \tilde{b}_3 \right)^{\frac{1}{6}} \cdot \left(1 - \sqrt{\Phi}\right)^{\frac{3}{2}} \cdot \Phi^{-\frac{1}{2}}$$
$$\cdot \left(\lambda_0 \left(1 - \lambda_0\right) \tilde{\mathbf{G}}^{-1}\left(\lambda_0\right) \cdot \tilde{\mathbf{s}} \right) \tag{79}$$

$$\tilde{\mathbf{T}}_c = -\tilde{K} \cdot \left[\left(\tilde{a}_1 \cdot \tilde{a}_2 \cdot \tilde{a}_3 \right)^{\frac{1}{3}} + \left(\tilde{b}_1 \cdot \tilde{b}_2 \cdot \tilde{b}_3 \right)^{\frac{1}{3}} \right]^2 \cdot \left(\tilde{a}_1 \tilde{a}_2 \tilde{a}_3 \tilde{b}_1 \tilde{b}_2 \tilde{b}_3 \right)^{\frac{1}{6}} \cdot \lambda_0 \left(1 - \lambda_0\right)^2$$
$$\cdot \left(1 - \sqrt{\Phi}\right)^{\frac{3}{2}} \cdot \Phi^{-\frac{1}{2}} \cdot \left(\tilde{\mathbf{A}}^{-1} \tilde{\mathbf{G}}^{-1}\left(\lambda_0\right) \tilde{\mathbf{s}} \right) \times \left[\tilde{\mathbf{G}}^{-1}\left(\lambda_0\right) \tilde{\mathbf{s}} \right] \tag{80}$$

They are same as the original equations (44)(45). Then, in order to find the dominant factors of contact force and torque, denote the ellipsoid is fixed which the radii, initial direction and position are foregone, so the key factor for calculating the values of force and torque is only the parameter $\tilde{K}$.

Dimensional analysis of Stokes Resistance: Based on the Stokes resistance force and torque equations (55)-(61), the physical quantities are particular and complex matrixes. So, the process of finding the dimensions will be inferred below.

According to Equations (60)(61), the Equation. (60) can be written as

$$\chi = \int_0^\infty \frac{d\lambda}{\sqrt{(a_1^2 + \lambda)(a_2^2 + \lambda)(a_3^2 + \lambda)}} \tag{81}$$

Because radius, so command which is a constant. Plug them into Equation (81), the formula could be expressed as

$$\chi = \int_0^{\infty} \frac{d(\Delta x^2 \cdot \beta)}{\sqrt{(a_1{}^2 + \Delta x^2 \cdot \beta)(a_2{}^2 + \Delta x^2 \cdot \beta)(a_3{}^2 + \Delta x^2 \cdot \beta)}} \tag{82}$$

Replace a with dimensionless, thus

$$\chi = \int_0^{\infty} \frac{d(\Delta x^2 \cdot \beta)}{\sqrt{(\tilde{a}_1{}^2 \cdot \Delta x^2 + \Delta x^2 \cdot \beta)(\tilde{a}_2{}^2 \cdot \Delta x^2 + \Delta x^2 \cdot \beta)(\tilde{a}_3{}^2 \cdot \Delta x^2 + \Delta x^2 \cdot \beta)}} \tag{83}$$

Simplify the formula, it is

$$\chi = \frac{1}{\Delta x} \int_0^{\infty} \frac{d\beta}{\sqrt{(\tilde{a}_1{}^2 + \beta)(\tilde{a}_2{}^2 + \beta)(\tilde{a}_3{}^2 + \beta)}} \tag{84}$$

From the equation (84), it can be seen that the dimension of component χ is $\frac{1}{\Delta x}$ in terms of the dimensionless expression $\frac{d\beta}{\sqrt{(\tilde{a}_1{}^2 + \beta)(\tilde{a}_2{}^2 + \beta)(\tilde{a}_3{}^2 + \beta)}}$.

Likewise, the Equation (59) also can be written as

$$\alpha_k = \int_0^{\infty} \frac{d(\Delta x^2 \beta)}{\left(\Delta x^2 \tilde{\alpha}_k^2 + \Delta x^2 \beta\right)\sqrt{\left(\tilde{a}_1^2 \Delta x^2 + \Delta x^2 \beta\right)\left(\tilde{a}_2^2 \Delta x^2 + \Delta x^2 \beta\right)\left(\tilde{a}_3^2 \Delta x^2 + \Delta x^2 \beta\right)}} \quad (k=1,2,3) \tag{85}$$

Simplify it, the formula is

$$\alpha_k = \frac{1}{\Delta x^3} \int_0^{\infty} \frac{d\beta}{\left(\tilde{\alpha}_k^2 + \beta\right)\sqrt{\left(\tilde{a}_1^2 + \beta\right)\left(\tilde{a}_2^2 + \beta\right)\left(\tilde{a}_3^2 + \beta\right)}} \quad (k = 1,2,3) \tag{86}$$

Therefore, the dimension of α_k is $\frac{1}{\Delta x^3}$.

Using dimensionless parameter to formulate α_k and χ, they are expressed as

$$\begin{cases} \tilde{\alpha}_k = \tilde{\alpha}_k \cdot \Delta x^3 \\ \tilde{\chi} = \chi \cdot \Delta x \end{cases} \tag{87}$$

Plug Equation (87) into Equations (57)(58), there are

$$\mathbf{K}^t = 16\pi \left(\frac{1}{\frac{\tilde{\chi}}{\Delta x} + (\tilde{a}_1 \Delta x)^2 \frac{\tilde{\alpha}_1}{\Delta x^3}} \mathbf{u}_1 \otimes \mathbf{u}_1 + \frac{1}{\frac{\tilde{\chi}}{\Delta x} + (\tilde{a}_2 \Delta x)^2 \frac{\tilde{\alpha}_2}{\Delta x^3}} \mathbf{u}_2 \otimes \mathbf{u}_2 + \frac{1}{\frac{\tilde{\chi}}{\Delta x} + (\tilde{a}_3 \Delta x)^2 \frac{\tilde{\alpha}_3}{\Delta x^3}} \mathbf{u}_3 \otimes \mathbf{u}_3 \right) \tag{88}$$

$$\mathbf{K}^r = 16\pi \left(\begin{array}{l} \dfrac{(\tilde{a}_2\Delta x)^2+(\tilde{a}_3\Delta x)^2}{(\tilde{a}_2\Delta x)^2\dfrac{\tilde{\alpha}_2}{\Delta x^3}+(\tilde{a}_3\Delta x)^2\dfrac{\tilde{\alpha}_3}{\Delta x^3}}\mathbf{u}_1\otimes\mathbf{u}_1+\dfrac{(\tilde{a}_1\Delta x)^2+(\tilde{a}_3\Delta x)^2}{(\tilde{a}_1\Delta x)^2\dfrac{\tilde{\alpha}_1}{\Delta x^3}+(\tilde{a}_3\Delta x)^2\dfrac{\tilde{\alpha}_3}{\Delta x^3}}\mathbf{u}_2\otimes\mathbf{u}_2 \\ +\dfrac{(\tilde{a}_2\Delta x)^2+(\tilde{a}_1\Delta x)^2}{(\tilde{a}_2\Delta x)^2\dfrac{\tilde{\alpha}_2}{\Delta x^3}+(\tilde{a}_1\Delta x)^2\dfrac{\tilde{\alpha}_1}{\Delta x^3}}\mathbf{u}_3\otimes\mathbf{u}_3 \end{array} \right) \tag{89}$$

Simplify them, the components $\mathbf{K}^t$ and $\mathbf{K}^r$ are

$$\mathbf{K}^t = \Delta x \cdot 16\pi \left(\frac{1}{\tilde{\chi}+\tilde{a}_1^2\tilde{\alpha}_1}\mathbf{u}_1\otimes\mathbf{u}_1+\frac{1}{\tilde{\chi}+\tilde{a}_2^2\tilde{\alpha}_2}\mathbf{u}_2\otimes\mathbf{u}_2+\frac{1}{\tilde{\chi}+\tilde{a}_3^2\tilde{\alpha}_3}\mathbf{u}_3\otimes\mathbf{u}_3 \right) \tag{90}$$

$$\mathbf{K}^r = \Delta x^3 \cdot 16\pi \left(\begin{array}{l} \dfrac{\tilde{a}_2^2+\tilde{a}_3^2}{\tilde{a}_2^2\tilde{\alpha}_2+\tilde{a}_3^2\tilde{\alpha}_3}\mathbf{u}_1\otimes\mathbf{u}_1+\dfrac{\tilde{a}_1^2+\tilde{a}_3^2}{\tilde{a}_1^2\tilde{\alpha}_1+\tilde{a}_3^2\tilde{\alpha}_3}\mathbf{u}_2\otimes\mathbf{u}_2 \\ +\dfrac{\tilde{a}_2^2+\tilde{a}_1^2}{\tilde{a}_2^2\tilde{\alpha}_2+\tilde{a}_1^2\tilde{\alpha}_1}\mathbf{u}_3\otimes\mathbf{u}_3 \end{array} \right) \tag{91}$$

So from Equations (90) (91), the dimensions of components $\mathbf{K}^t$ and $\mathbf{K}^r$ are Δx and Δx^3, respectively.

To sum up the inferred process, the additional dimensionless parameters in Stokes formula can be tabled below.

Table 3: The particular dimensionless variables in Stokes resistance force and torque.

Variable definition or expression	Dimensionless variable
$\chi = \int_0^\infty \dfrac{d\lambda}{\Delta\lambda} \quad (k=1,2,3)$ In which $\Delta\lambda = \sqrt{\left(a_1^2+\lambda\right)\left(a_2^2+\lambda\right)\left(a_3^2+\lambda\right)}$	$\tilde{\chi} = \chi \cdot \Delta x$
$\alpha_k = \int_0^\infty \dfrac{d\lambda}{\left(\alpha_k^2+\lambda\right)\Delta\lambda} \quad (k=1,2,3)$	$\tilde{\alpha}_k = \alpha_k \cdot \Delta x^3$
$\mathbf{F}_{k,i}^{Ad} = -\dfrac{\partial W_k^{Ad}}{\partial \mathbf{r}_i} \quad (k=c,s)$	$\tilde{\mathbf{K}}^t = \dfrac{\mathbf{K}^t}{\Delta x}$
$\mathbf{K}^r = 16\pi\left(\dfrac{a_2^2+a_3^2}{a_2^2\alpha_2+a_3^2\alpha_3}\mathbf{u}_1\otimes\mathbf{u}_1+\dfrac{a_1^2+a_3^2}{a_1^2\alpha_1+a_3^2\alpha_3}\mathbf{u}_2\otimes\mathbf{u}_2+\dfrac{a_2^2+a_1^2}{a_2^2\alpha_2+a_1^2\alpha_1}\mathbf{u}_3\otimes\mathbf{u}_3\right)$	$\tilde{\mathbf{K}}^r = \dfrac{\mathbf{K}^r}{\Delta x^3}$

Based on Table 1 and Table 3, plug related variables into Equations (55)-(61), the dimensionless equations of Stokes resistance are formulated as follows.

$$\frac{\tilde{\mathbf{F}}\cdot\rho_c\cdot\Delta x^4}{\Delta t^2}=-\frac{\tilde{\mu}\cdot\rho_c\cdot\Delta x^2}{\Delta t}\cdot\tilde{\mathbf{k}}^t\Delta x\cdot(\frac{\tilde{\mathbf{U}}_0\Delta x}{\Delta t}-\frac{\tilde{\mathbf{u}}_0\Delta x}{\Delta t}) \tag{92}$$

$$\frac{\tilde{\mathbf{T}}\cdot\rho_c\cdot\Delta x^5}{\Delta t^2}=-\frac{\tilde{\mu}\cdot\rho_c\cdot\Delta x^2}{\Delta t}\cdot\tilde{\mathbf{K}}^r\Delta x^3\cdot(\frac{\tilde{\mathbf{u}}}{\Delta t}-\frac{\tilde{\mathbf{u}}_f}{\Delta t}) \tag{93}$$

Simplify them, there are

$$\tilde{\mathbf{F}}=-\tilde{\mu}\cdot\tilde{\mathbf{K}}^t(\tilde{\mathbf{U}}_0-\tilde{\mathbf{u}}_0) \tag{94}$$

(95)

Where

$$\tilde{\mathbf{T}}=-\tilde{\mu}\cdot\tilde{\mathbf{K}}^r(\tilde{\boldsymbol{\omega}}-\tilde{\boldsymbol{\omega}}_f)$$

$$\tilde{\mathbf{K}}^t=16\pi\left(\frac{1}{\tilde{\chi}+\tilde{a}_1^2\tilde{\alpha}_1}\mathbf{u}_1\otimes\mathbf{u}_1+\frac{1}{\tilde{\chi}+\tilde{a}_2^2\tilde{\alpha}_2}\mathbf{u}_2\otimes\mathbf{u}_2+\frac{1}{\tilde{\chi}+\tilde{a}_3^2\tilde{\alpha}_3}\mathbf{u}_3\otimes\mathbf{u}_3\right) \tag{96}$$

$$\tilde{\mathbf{K}}^r=16\pi\left(\frac{\tilde{a}_2^2+\tilde{a}_3^2}{\tilde{a}_2^2\tilde{\alpha}_2+\tilde{a}_3^2\tilde{\alpha}_3}\mathbf{u}_1\otimes\mathbf{u}_1+\frac{\tilde{a}_1^2+\tilde{a}_3^2}{\tilde{a}_1^2\tilde{\alpha}_1+\tilde{a}_3^2\tilde{\alpha}_3}\mathbf{u}_2\otimes\mathbf{u}_2+\frac{\tilde{a}_2^2+\tilde{a}_1^2}{\tilde{a}_2^2\tilde{\alpha}_2+\tilde{a}_1^2\tilde{\alpha}_1}\mathbf{u}_3\otimes\mathbf{u}_3\right) \tag{97}$$

$$\tilde{\chi}=\int_0^\infty\frac{d\beta}{\sqrt{(\tilde{a}_1{}^2+\beta)(\tilde{a}_2{}^2+\beta)(\tilde{a}_3{}^2+\beta)}} \tag{98}$$

$$\tilde{\alpha}_k=\int_0^\infty\frac{d\beta}{\left(\tilde{\alpha}_k^2+\beta\right)\sqrt{\left(\tilde{a}_1^2+\beta\right)\left(\tilde{a}_2^2+\beta\right)\left(\tilde{a}_3^2+\beta\right)}}\quad(k=1,2,3) \tag{99}$$

Compared with original equations, they are the same expressions. They satisfy the homogeneity. Same with contact part, in order to find the dominant factor of Stokes resistance force and torque, assume ellipsoids are fixed in static fluid and the is constant, the dimensionless dynamic viscosity is the key factor here. Its dimensionless expression with basic reference dimensions is in Table 1.

Dimensional analysis of Gravity force: According to Table 1, dimensionless equation of gravity force could be expressed as

$$\frac{\tilde{\mathbf{G}}\cdot\rho_e\cdot\Delta x^4}{\Delta t^2}=\tilde{m}_e\rho_e\Delta x^3\cdot\frac{\tilde{\mathbf{g}}\Delta x}{\Delta t^2} \tag{100}$$

Simplify it,

$$\tilde{\mathbf{G}}=\tilde{m}_e\cdot\tilde{\mathbf{g}} \tag{101}$$

It is easy to see that if the ellipsoid is constant, which means the radii and density of ellipsoid is const, the effect factor of gravity force is dimensionless gravity acceleration. Summarize the dimension analysis of three kinds of forces and two torques, dominant factors of these dimensionless equations are $\tilde{K}, \tilde{\mu}$ and $\tilde{\mathbf{g}}$, which are related with basic reference dimensions $\Delta x, \Delta t, \rho_c$.

Test and find the proper values of dimensionless coefficients

Through the dimension analysis, the key factors have been pointed out. From the dimensionless parameter's expressions, it is clear that the basic reference dimensions are significant. This section will test the values of reference dimensions to get the proper values of dimensionless parameters, such as $\tilde{K}, \tilde{\mu}$ and $\tilde{\mathbf{g}}$. Additionally, from the above equations, the key dimensionless parameter has one same value in related force formula and torque formula. Therefore, only consider the coefficients in force equations.

Situation1: ellipsoid in static water: In this simulation, the reference density ρ_c is $1000kg/m^3$, which is water's density in international standard (IS). Compared with water, the density of ellipsoid may be bigger than ρ_c in terms of its chemical composition. If assume the ellipsoid as sand, the ellipsoidal density may be three times than water's IS density.

$$\rho_e = 3\rho_c = 3000kg/m^3 \tag{102}$$

Then the dimensionless density of ellipsoid could be expressed as

$$\rho_e = 3 \tag{103}$$

As mentioned above, the speed of sound in water C is usually used to describe the reference velocity $C = \frac{\Delta x}{\Delta t} = 2000(m/s)$. However, if using the ellipsoid to simulate the sand in fluid, the reference velocity may be unrealistic. So, define a new reference dimension is U with a new basic reference time delta T.

$$U = \frac{\Delta x}{\Delta T} \tag{104}$$

But how big the new reference velocity should be? In order to find values of delta x and delta T, simply consider the situation is that ellipsoids do not contact any others. Thus, only gravity force and stokes resistance force will exert on an ellipsoid, based on Newton Second Law to formulate these forces for an ellipsoid with physical quantities, which can be written as

$$m\frac{d\mathbf{v}}{dT} = m\mathbf{g} - \mu\mathbf{K}^t\mathbf{v} \tag{105}$$

Simplify this equation, it could be formulated as

$$dT = \int \frac{d\mathbf{v}}{\mathbf{g} - \frac{\mu\mathbf{K}^t\mathbf{v}}{m}} \tag{106}$$

So, the time T may be solved as the equation

$$T = \frac{1}{-\frac{\mu\mathbf{K}^t}{m}} \cdot \ln(\mathbf{g} - \frac{\mu\mathbf{K}^t}{m}\mathbf{v}) \tag{107}$$

Use the exponential equation to describe it,

$$xe^{-\frac{\mu\mathbf{K}^t T}{m}} = \mathbf{g} - \frac{\mu\mathbf{K}^t}{m}\mathbf{v} \tag{108}$$

Where x is an unknown number. Consider that in this case, the ellipsoid has no initial velocity. That means at the beginning, time T=0, velocity of ellipsoid $\tilde{\mathbf{o}}$ = (0, 0, 0). Thus, plug them in Equation (108), the unknown number x must satisfy

$$\mathbf{x} = \mathbf{g} \tag{109}$$

Plug Equation (109) into Equation (108), there is

$$\mathbf{g}e^{-\frac{\mu \mathbf{K}^t T}{m}} = \mathbf{g} - \frac{\mu \mathbf{K}^t}{m}\mathbf{v} \tag{110}$$

After merging the same component, Equation (110) can be formulated as

$$\mathbf{g}(1 - e^{-\frac{\mu \mathbf{K}^t T}{m}}) = \frac{\mu \mathbf{K}^t}{m}\mathbf{v} \tag{111}$$

Solved the velocity from this equation, there is

$$\mathbf{v} = \frac{m\mathbf{g}}{\mu \mathbf{K}^t}(1 - e^{-\frac{\mu \mathbf{K}^t T}{m}}) \tag{112}$$

If time $T \to \infty$, $e^{-\frac{\mu \mathbf{K}^t T}{m}} \to 0$, then the equation (112) can be expressed as

$$\mathbf{v} = \frac{m\mathbf{g}}{\mu \mathbf{K}^t} \tag{113}$$

The dimensionless velocity of ellipsoid at infinite time by reference velocity U could be expressed as

$$\tilde{\mathbf{v}} = \frac{\mathbf{v}}{U} = \frac{m\mathbf{g}}{U\mu \mathbf{K}^t} \tag{114}$$

Note that it represents the velocity at infinite time, so this dimensionless velocity is an infinite velocity. In order to simulation, assume the infinite velocity of ellipsoid in fluid is 0.001. So, the equation (114) could be written as

$$m\mathbf{g} = 0.001 \times U\mu \mathbf{K}^t \tag{115}$$

Simply estimate the values of variables, if regarding ellipsoid as sphere, then the matrix $\mathbf{K}^t = 6\pi r$, and the mass of ellipsoid is similar to sphere which is $m = 4/3 \cdot \pi \cdot \rho_e \cdot r^3$. If the fluid instantiates as water, in International Standard, the water at 20 °C has a dynamic viscosity which $\mu = 0.001 Pa.s$ [33]. Plug these equations into Equation (115), there are

$$\begin{aligned} &4/3 \cdot \pi \cdot \rho_e \cdot r^3 \cdot \mathbf{g} = 10^{-3} \cdot U \cdot 10^{-3} \cdot 6\pi r \\ &(\rho_e = 3000 kg/m^3, \mathbf{g} = 10 m/s^2) \end{aligned} \tag{116}$$

So, the reference velocity is

$$U = \frac{2}{3} \times 10^{10} \times r^2 \tag{117}$$

Using dimensionless equation of radius $r = \Delta x \cdot \tilde{r}$ to replace the physical quantity radius. Because in this chapter, the dimensionless radii of all ellipsoids are (2, 3, 4), so take the average value of radii for sphere, which is $\tilde{r} = 3$. Then there is equation $r = \Delta x \cdot \tilde{r} = 3 \times \Delta x$. Then take these values and equation (104) into Equation (117), there is expression with only basic reference dimensions $\Delta x, \Delta T$.

$$\frac{\Delta x}{\Delta T} = \frac{2}{3} \times 10^{10} \times 9 \times \Delta x^2 \tag{118}$$

Simplify this equation,

$$\Delta x \cdot \Delta T = 2.5 \times 10^{-9} \tag{119}$$

Therefore, the value of $\Delta x \cdot \Delta T$ may be set around as 2.5×10^{-9} . Basically, the radius of normal sand is about from 0.1mm to 1mm. So, if assume $\Delta x = 10^{-4}$, probably $\Delta T = 2.0 \times 10^{-5}$, thus there are

$$U = \frac{\Delta x}{\Delta T} = 5(m/s) \tag{120}$$

Thus, the reference velocity of ellipsoid in fluid is 5 meters per second. And the coefficients $\tilde{K}, \tilde{\mu}$ and $\tilde{\mathbf{g}}$ can be calculated as follow.

Dimensionless dynamic viscosity ($\mu = 0.001 Pa.s$) of water

$$\tilde{\mu} = \frac{\mu \cdot \Delta T}{\rho_c \cdot \Delta x^2} = \frac{10^{-3} \cdot 2 \times 10^{-5}}{1000 \times 10^{-8}} = 2 \times 10^{-3} \tag{121}$$

Based on Table 2, the dimensionless parameter K is related with Young's modulus E. For sand particles, Young's modulus is about 60,000pa. So, for two sands, E1=E2, such that $K = \frac{2E}{3} = 40,000 pa$, take the value into equation, the dimensionless parameter of Contact force and torque is

$$\tilde{k}_{con} = \frac{k_{con} \cdot \Delta T^2}{\rho_c \cdot \Delta x^2} = \frac{40000 \times 4 \times 10^{-10}}{1000 \times 10^{-8}} = 1.6 \tag{122}$$

The dimensionless gravity acceleration is

$$\tilde{\mathbf{g}} = \frac{\mathbf{g} \cdot \Delta T^2}{\Delta x} = \frac{10 \times 4 \times 10^{-10}}{10^{-4}} = 4 \times 10^{-5} \tag{123}$$

Situation 2: ellipsoid in air: If the ellipsoid is in the air, what will happen then? Because the dynamic viscosity depends on what the environment is, the simulation may be different as the coefficient μ . From the point of force, formulate the total forces that are contact force, Stokes resistance force and gravity force act on a fixed ellipsoid, so the dimensionless equation could be expressed based on Newton second Law as

$$\tilde{\mathbf{F}}_{con}+\tilde{\mathbf{F}}_{G}+\tilde{\mathbf{F}}_{s}=\tilde{m}\frac{d\tilde{\mathbf{o}}}{dt} \tag{124}$$

As the contact force is a conditional force in terms of the distance of each other. And the value of contact force has a big difference, which is from zero. Assume $\tilde{\mathbf{F}}_{con}=(0.0,0.0,0.0)$ that means the ellipsoid do not contact each other. Such that there are two forces, stokes resistance force and gravity. However, the dimensionless mass for an ellipsoid is constant because the dimensionless radii and dimensionless density is constant in this case. If stokes force satisfy the inequality $|\tilde{F}_S|<<|\tilde{F}_G|$, the stokes resistance may be ignored in this situation.

The dimensionless gravity force can be written as

$$\tilde{\mathbf{F}}_{G}=\tilde{m}\cdot\tilde{\mathbf{g}}=\frac{4}{3}\cdot\pi\cdot\tilde{r}_1\cdot\tilde{r}_2\cdot\tilde{r}_3\cdot\tilde{\rho}\cdot\tilde{\mathbf{g}} \tag{125}$$

Then the dimensionless formula of stokes resistance force is

$$\tilde{\mathbf{F}}_{s}=-\tilde{\mu}\cdot\tilde{\mathbf{K}}^{t}\tilde{\mathbf{v}}_{e} \tag{126}$$

As mentioned above, the dimensionless velocity of ellipsoid has infinite number. So, the dimensionless stokes force has infinite value when the dimensionless velocity achieves the infinite value.

If $\left|\tilde{\mathbf{F}}_{G}\right|>=1000\times\left|\tilde{\mathbf{F}}_{s}\right|$, the dimensionless stokes resistance force may be ignored. Hence formulate and simplify the inequality

$$\frac{\left|\tilde{\mathbf{F}}_{G}\right|}{\left|\tilde{\mathbf{F}}_{s}\right|}=\frac{\left|\frac{4}{3}\cdot\pi\cdot\tilde{r}_1\cdot\tilde{r}_2\cdot\tilde{r}_3\cdot\tilde{\rho}_e\cdot\tilde{\mathbf{g}}\right|}{\left|\tilde{\mu}\cdot\tilde{\mathbf{K}}^{t}\cdot\tilde{\mathbf{v}}_{e}\right|}>=1000 \tag{127}$$

Plug the calculated dimensionless parameters' values which are $\Delta x=1.0\times10^{-4}$, $\Delta T=2.0\times10^{-5}$, $\tilde{\mathbf{g}}=4\times10^{-5}$ and $\tilde{\mathbf{v}}_{e}=0.001$ into Equation (127)

$$\frac{\left|\tilde{\mathbf{F}}_{G}\right|}{\left|\tilde{\mathbf{F}}_{s}\right|}=\frac{200.96\times4\times10^{-5}}{\tilde{\mu}\cdot\tilde{\mathbf{K}}^{t}\times10^{-3}}>=1000 \tag{128}$$

According to Equation (96) and (98), the matrix $\tilde{\mathbf{K}}^{t}$ is only related to constants that are dimensionless radii and constant number β. So, it is also a constant, after estimating by computer, the value of is about 55. Hence Equation (128) could be simplified

$$\tilde{\mu}<1.46\times10^{-4} \tag{129}$$

That means if the dimensionless dynamic viscosity satisfies the Equation (129), the Stokes resistance can be ignored. The dynamic viscosity of air depends mostly on the temperature. At 15°C, the dynamic viscosity of air is μ =1.78×10−5 Pa.s [33], thus the dimensionless dynamic viscosity of air at 15°C is

$$\tilde{\mu}_{air}=\frac{\mu_{air}\cdot\Delta T}{\rho_c\cdot\Delta x^2}=\frac{1.78\times10^{-5}\cdot2\times10^{-5}}{1000\times10^{-8}}=3.56\times10^{-5} \tag{130}$$

Compared with critical value 1.46×10^{-4} , the value of $\tilde{\mu}_{air}$ is smaller. Thus, the Stokes resistance in air for this ellipsoid could be ignored. Nevertheless, if take the values and without strokes into the program to debug, the program throws the "buffer too small". The reason may be that there is no resistance, so the velocity of ellipsoid at a moment is too big because of contact force, such that the position of ellipsoid is out of the buffer. The solution could be to let the delta time be smaller, if the position in smaller time will be changed slowly. Therefore, if delta time has been changed, the values of other coefficients also will be changed. Let the basic reference dimensions are

$$\Delta T = 2.0\times10^{-6}, \Delta x = 1.0\times10^{-4}, U = \frac{\Delta x}{\Delta T} = \frac{1.0\times10^{-4}}{2.0\times10^{-6}} = 50 \tag{131}$$

So, the dimensionless gravity acceleration is

$$\tilde{\mathbf{g}} = \frac{\mathbf{g}\cdot\Delta \mathrm{T}^2}{\Delta x} = \frac{10\times4\times10^{-12}}{10^{-4}} = 4\times10^{-7} \tag{132}$$

Because the reference velocity U has been changed, thus the dimensionless infinite velocity of ellipsoid should be 0.0001. Plug them into Equation (127), the critical value of $\tilde{\mu}$ is

$$\tilde{\mu} < 1.46\times10^{-5} \tag{133}$$

In this situation, the value of $\tilde{\mu}_{air}$ is

$$\tilde{\mu}_{air} = \frac{\mu_{air}\cdot\Delta T}{\rho_c\cdot\Delta x^2} = \frac{1.78\times10^{-5}\cdot2\times10^{-6}}{1000\times10^{-8}} = 3.56\times10^{-6} \tag{134}$$

It is still satisfying the condition. Therefore, the Stokes resistance for the ellipsoid in the air could be ignored. To sum up the analysis before, with some conditions the Stokes resistance could be disregarded. If use the physic quantities to describe $\tilde{\mathbf{g}}, \tilde{\mu}, \tilde{\mathbf{v}}_e$, the Equation (127) is

$$\frac{\left|\tilde{\mathbf{F}}_G\right|}{\left|\tilde{\mathbf{F}}_s\right|} = A\cdot\frac{\left|\frac{\mathbf{g}\cdot\Delta \mathrm{T}^2}{\Delta x}\right|}{\left|\frac{\mu\cdot\Delta T}{\rho_c\cdot\Delta x^2}\cdot\frac{\tilde{\mathbf{o}}_e\cdot\Delta T}{\Delta x}\right|} >= 1000 (\text{where } A = \frac{\frac{4}{3}\cdot\pi\cdot\tilde{r}_1\cdot\tilde{r}_2\cdot\tilde{r}_3\cdot\tilde{\rho}_e}{\tilde{\mathbf{K}}^t}) \tag{135}$$

Put the constant parameters together, and simply it

$$\frac{\left|\tilde{\mathbf{F}}_G\right|}{\left|\tilde{\mathbf{F}}_s\right|} = B\cdot\frac{\left|\Delta x^2\right|}{\left|\mu\cdot\mathbf{v}_e\right|} >= 1000 (\text{where } B = \frac{\mathbf{g}\cdot\rho_c\cdot\frac{4}{3}\cdot\pi\cdot\tilde{r}_1\cdot\tilde{r}_2\cdot\tilde{r}_3\cdot\tilde{\rho}_e}{\tilde{\mathbf{K}}^t}) \tag{136}$$

$$\frac{\left|\Delta x^2\right|}{\left|\mu\cdot\mathbf{v}_e\right|} >= \frac{1000}{B} (\text{where } B = \frac{\mathbf{g}\cdot\rho_c\cdot\frac{4}{3}\cdot\pi\cdot\tilde{r}_1\cdot\tilde{r}_2\cdot\tilde{r}_3\cdot\tilde{\rho}_e}{\tilde{\mathbf{K}}^t}) \tag{137}$$

Solved this equation, the dynamic viscosity should satisfy Equation (138) when disregarding the Stokes resistance for an ellipsoid.

$$\mu <= \frac{B \cdot \Delta x^2}{1000 \mathbf{v}_e} (\text{where } B = \frac{\mathbf{g} \cdot \rho_c \cdot \frac{4}{3} \cdot \pi \cdot \tilde{r}_1 \cdot \tilde{r}_2 \cdot \tilde{r}_3 \cdot \tilde{\rho}_e}{\tilde{\mathbf{K}}^t} = 54807.27) \quad (138)$$

Form equation (137), it indicates that a determinate ellipsoid whether has Stokes resistance depends on the dynamic viscosity, the denoting physical infinite velocity of ellipsoid, and the basic reference length delta X. For this chapter, plug values into equation (138), which are

$$\Delta T = 2.0 \times 10^{-6}, \Delta x = 1.0 \times 10^{-4}, \mathbf{v} = 0.005(m/s), \rho_e = 3000 kg/m^3, \tilde{\mathbf{K}}^t = 55 \text{ and } \tilde{r}=(2,3,4)$$

. So

$$\mu <= 1.09 \times 10^{-4} \quad (139)$$

The dynamic viscosity of air at 15°C is $\mu_{air} = 1.78 \times 10^{-5}$, less than 1.09×10^{-4}. To the contrary, the dynamic viscosity of water is 0.001 and it is bigger than the border value, so the stokes resistance in water must be considered. From Equation (138), it also indicates that if denote $\Delta x = 1.0 \times 10^{-4}$, and ellipsoid in the air $\mu_{air} = 1.78 \times 10^{-5}$, let the Stokes resistance be disregarded, the infinite velocity of ellipsoid should satisfy

$$\mathbf{v}_e <= \frac{B \cdot \Delta x^2}{1000 \cdot \mu_{air}} = 0.03(m/s) \quad (140)$$

So, because the infinite velocity is $\tilde{\mathbf{o}} = 0.005(m/s)$, it also satisfies the condition to disregard the Stokes resistance. Importantly, the elastically of this program is performed on the dimensional analysis through the basic reference dimensions. For instance, if let the delta X delta T, and dimensionless density be smaller, the program may simulate the cell in the substrate with cell sap. Or if these basic reference dimensions are bigger, then the program may be used to calculate the collision and response between superior ellipsoidal models. Therefore, Dimensional analysis is necessary for the expendability of a program.

Section 6: Parallel Performance and Model Result

After programming the comprehensive knowledge as the theory of model, methodology of parallelism, and dimensional analysis for debugging, the performance of parallel computing and simulation of ellipsoidal motion will be summarized as follows.

Performance analysis

As introduced before, TBB parallel template functions are applied in this chapter to achieve speedup. So, to what extent the parallel computing affected the speed of calculation. The first illustration, Figure 28, is a bar diagram to compare the time-consuming of running the test program between serial code and paralleling program.

This test program is that with the augment of time-count iterations, the number of ellipsoids would increase from six to two hundred as adding one ellipsoid per hundred time-counts. So, increasing to 206 ellipsoids represents that number of time-count iterations are 20,006. The X axis of diagram is used to describe the different number of ellipsoids. And the Y axis shows that the range of time consuming is from zero to 500 minutes and unit scale is 20 minutes. Then, the red bar in this figure stands for serial code and the yellow bar represents the parallel program. So, the whole process of simulation, which the number of ellipsoids increase to 206, for paralleled program only need 44 minutes but for serial code, it costs 467 minutes. We cannot exactly say how many times speed up the parallel support as there may other threads were running on the machine when we were tracing the code. But, according to the illustration, it still indicates that the parallel technology of Threading Building Blocks really improves the computational efficiency to achieve obvious speedup because of multi threads.

The second presentation will be based on Figure 29. The aim of this chart is to illustrate the big efficiency variance of calculation between parallel and serial computing through calculating abundant number of ellipsoids; additional, it points out that as mentioned before, the parallel efficiency of TBB depends on what the hardware machine has.

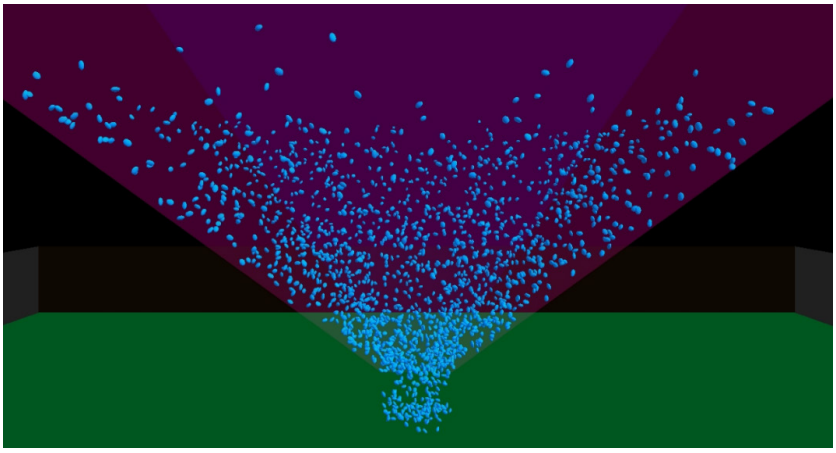


Figure 30: Simulation of 2,000 ellipsoids were falling in the hopper-1.

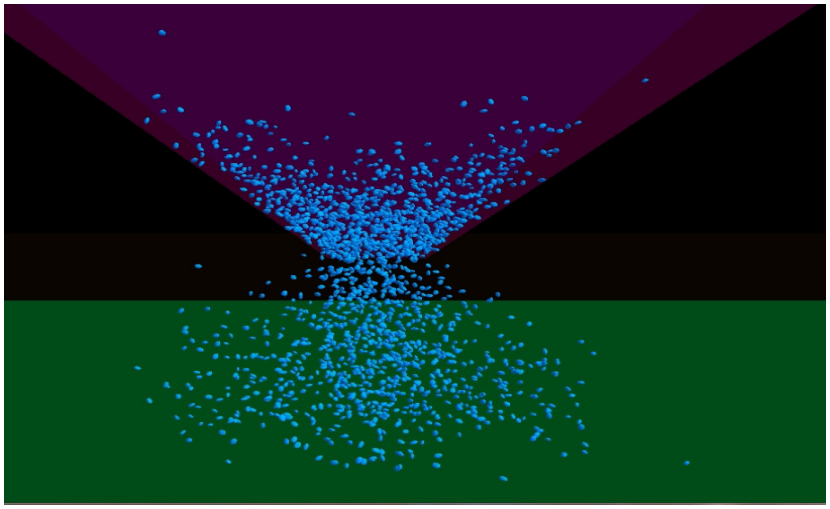


Figure 31: Simulation of 2,000 ellipsoids were falling in the hopper-2.

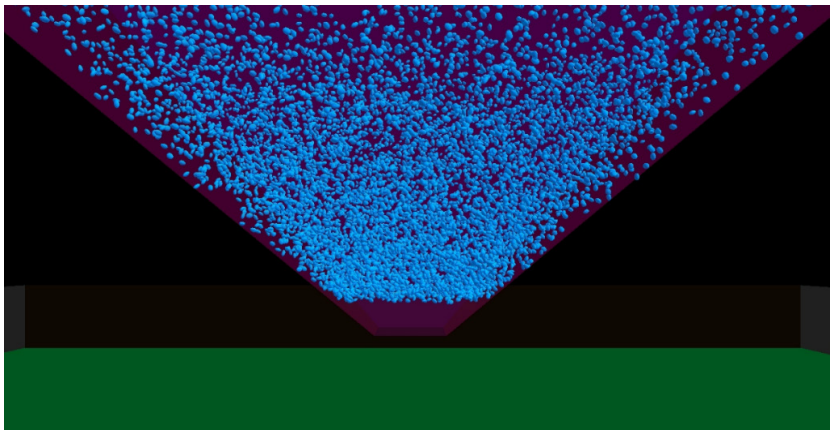


Figure 32: The initial status of 10,000 ellipsoids in the hopper.

Figure 32 shows that at the beginning of simulation, ellipsoids were random distribution in the hopper. Note that the delta time of this situation is changed, and the dimensionless dynamic viscosity of water is also included. The later status of falling has been figured out below.

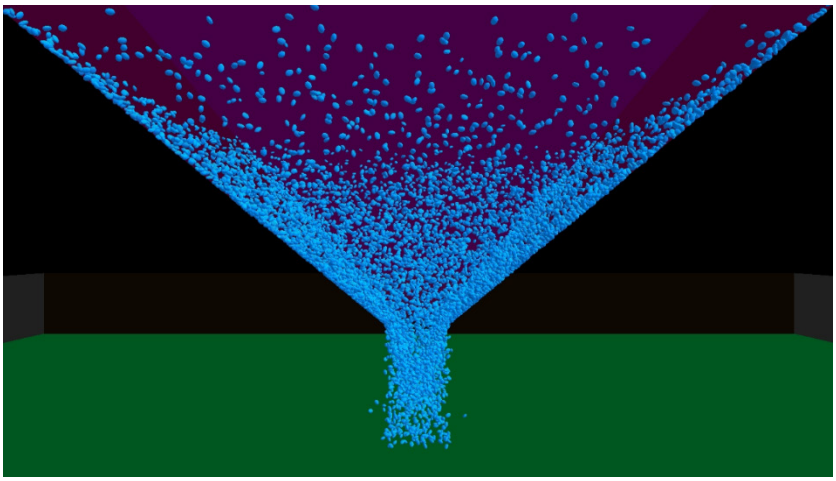


Figure 33: The middle process of falling.

This figure demonstrates the middle process of falling, in which some ellipsoid just falls out of the hopper yet does not touch the green plane. It can be seen that many ellipsoids lie on the hopper slabs to slowly slide because they have achieved the balance with the slab rather than quickly spring. Image that they are in static water, although there are not enough ellipsoids to have no gap between ellipsoids, but the sufficient stokes resistance helps them keep state of equilibrium. Just after 28,100-time counts, all of ellipsoids were falling on the green slab. As mentioned above, the equilibrium state eventually embodies in which 10,000 ellipsoids seem statically to lie on the slab See Figure 34.

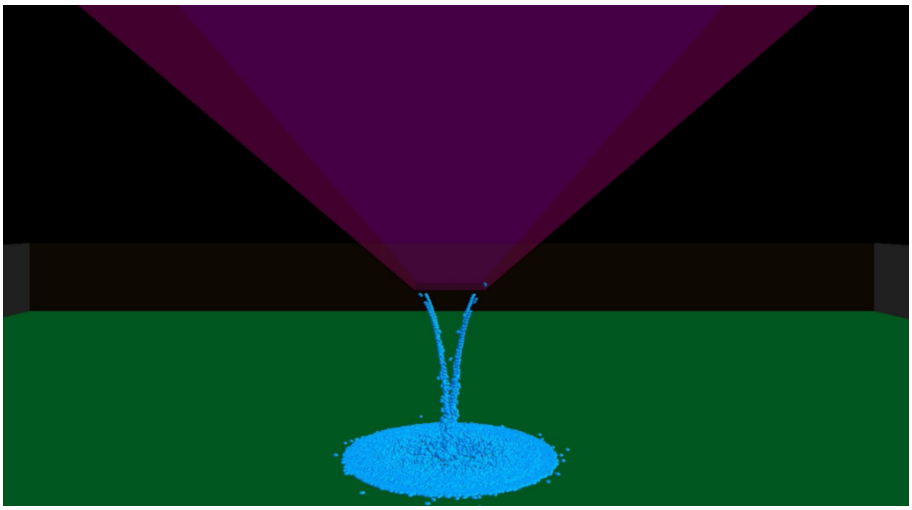


Figure 34: the final state of 10,000 ellipsoids.

As the paralleled program ran on the 8cores machine, the average speed is reduced to 3.06 second per step. According to the falling process, it is easy to see all of ellipsoids must pass through the small exit. Therefore, in the later computation, most tasks should be busier than initial because if the ellipsoids are far away each other or the contact potential is less than one (based on Equation (13)), the complex calculation of contact force and torque have no necessary to be calculated. So much time will be saved to continue other work. As this simulation, running to 14,100-time count, the mean speed of cycle time is only 1.69 which is half of the total average speed.

Section 7: Conclusion and Future Work

In general, this dissertation is to discuss the theory and methodology of modeling a mass of dynamic ellipsoids with parallel computing.

The section of theory and method of modelling ellipsoids is to introduce what and how physics acts on the ellipsoid. According to the stress analysis, the motion of an ellipsoid depends on the total force and torque which are contact force and torque, stokes resistance force and torque, and gravity force. To be more precise, it depends on the directions and position of each settled ellipsoid that would be set as known density and radii. The second part introduces the parallel method, Threading Building Blocks that is the enhanced productivity, succinct, and comprehensive parallel technology. The advantage of TBB is not only that it is easy for programmers to achieve efficient paralleling, but also with the unexpected and optimizing potential because the efficiency of paralleling with TBB depends on how powerful your machine.

Based on the first two parts to introduce the relevant theory and method, how to code the project will be discussed in section three. That mainly describes how to design the struct of program, which more proper algorithms could be applied, and the application as the part of function code. The final body part is dimensional analysis which is used to debug and test. Dimensional analysis could help programmers define the value of each const parameter. What is more? The program could be simulated as an alternative of many kinds through changing the values of basic reference dimensions. It is significant to expand the usability of this program in various application fields.

Basically, this program has achieved the primary aim of the dissertation proposal yet because of time limited, this program is still incomplete. So, after the dissertation, firstly, to parallel visualization with TBB technology, such as using pipeline function template, is the main work to try to increase the visual speed. As if the number of particles is too large, the visual process would be slow. For instance, to draw 20,000 ellipsoids in each frame cost more than one second. Secondly, try more simulations which are based on different span of basic reference dimensions, such as to simulate the water with very smaller delta X or through bigger delta X to simulate the collision and response between particular rigid bullets and elastic solid. Finally, try to apply these methodologies on any games, such as adapting some existing open-source games, and compare the advantages and disadvantages with others to learn and improve the optimizing of code.

References

1. Care CM, Cleaver DJ (2005) Computer simulation of liquid crystals. *Reports on progress in physics.* 68: 2665-2700.
2. Petitjean L, Reffay M, Grasland Mongrain E, Poujade M, Ladoux B, et al. (2010) Velocity Fields in a Collectively Migrating Epithelium. *Biophysical* 98: 1790-1800.
3. Game Physics Simulation (2012) Bullet Physics Engine.
4. Fauerby K (2003) Improved collision detection and response.
5. Rudomín I, Castillo JL Real-Time clothing: Geometry and physics. [Thesis] *Computer Science,* University of ITESM-CEM.
6. Goldstein H, Poole C, safko J (2001) The rigid body equations of motion. Classical Mechanics. 3ed., New York: Addison Wesley, pp.134-207.
7. Zhu J, Coakley S, Holcombe M, Hose R, Smallwood R (2011) Cell adhesion mediates clonal formation of stem and differentiated cell populations. University of Sheffield.
8. Brenner H (1963) The Stokes resistance of an arbitrary particle. *Chemical Engineering Science* 18(4)**:** 1-25.
9. Brenner H (1964a) The Stokes resistance of an arbitrary particle Part 2. An Extension. *Chemical Engineering Science* 19(9): 599-629.
10. Brenner H (1964b) The Stokes resistance of an arbitrary particle Part 3. Shear fields. *Chemical Engineering Science.* 19(11): 631-651.
11. Brenner H (1966) The Stokes resistance of an arbitrary particle V. Symbolic operator representation of intrinsic resistance. *Chemical Engineering Science* 21(11): 97-109.
12. Reinders J (2007) Intel Threading Building Blocks. Outfitting C++ for Multi-Core Processor Parallelism. O'Reilly Media, Inc.
13. Intel (2011) Intel@ Threading Building Blocks. reference manual. 1.26ed., Intel Cooperation.
14. Hu B, Yuan DH (2009) TBB Multi-Core Programming and Research on Its Hybrid Paradigms. *Computer Technology and Development.*
15. Zhuoru C, Chaoming J, Hongjie W, Chenming W (2004) *Engineering Fluid Mechanics 2th.* BeiJing: Higher Education of BeiJing, pp. 223-22.
16. Allen MP, Germano G (2002) Rigid body potentials, force and torques. Bristol H.H wills Physics Laboratory, University of Bristol.

17. Choi YK, Chang JW, Wang W, Kim MS, Elber G (2006) Real-Time Continuous Collision Detection for Moving Ellipsoids under Affine Deformation. report TR-2006-02. Hong Kong, The University of Hong Kong.

18. Perram JW, Wertheim MS (1985) Statistical mechanics of hard ellipsoids. I. overlap algorithm and the contact function. *Computational Physic* 58(2): 409-416.

19. Perram JW, Rasmussen J, Praestgaard EL, Lebowitz J (1996) Ellipsoid contact Potential: Theory and relation to overlap Potentials. *Physical Review E,* 54(6): 6565-6572.

20. Greenwood JA (1997) Analysis of elliptical hertzain contacts. *Tribology International* 30(3)**:** 235-237.

21. Paramonov L, Yaliraki SN (2005) The directional contact distance of two ellipsoids: Coarse-grained Potentials for anisotropic interactions. *Chemical Physics* 123(19): 194111.

22. Hertz H (1882) Über die berührung fester elastischer Körper (On the contact of rigid elastic solids). *Journal für die Reine und angewandte Mathemati*. 92**:** 156-171.

23. Brenner H. Gajdos LJ (1981) London-van der waals force and torques exerted on an ellipsoidal particle by a nearby semi-infinite slab. *CAN.J.CHEM* 59**:** 2004-2018.

24. Lucas NT (2010) Mathematics for Computer Games Part 1- Mathematical Techniques. [handout] University of Abertay Dundee. CE115a/K.

25. (2011c) WiKi Rodrigues' rotation formula.

26. Reinders J (2007) Intel Threading Building Blocks. Outfitting C++ for Multi-Core Precessor Parallelism. O'Reilly Media, Inc.

27. Lu J (2010) Use TBB and SIMD to optimize your game. *China Game Business Conference, 23 June, Shang* Hai.

28. Broadhurst R (2010) Napoleon: Total War - Better performance coming with the Empire add-on.

29. Guo S, Wu X, Lu C (2011) real-time dynamic volumetric cloud rendering on multi-core platforms Intel Software and Services Group (SSG).

30. Sampson AT (2010) Personal communication. University of Abertay Dundee.

31. Lippman SB, Lajoie J (2004) C++ Primer Third Edition (Chinese). Translated by Pan A. & Zhang L. New York: Addison Wesley. Beijing: China Electric Power Press, ISBN: 9780201164879, pp.209-240.

32. (2011a) WiKi Dimensional analysis. Wikipedia foundation, lnc.

33. (2011b) WiKi Dynamic Viscosity Wikipedia foundation, lnc.

Tensegrity Model for Living Cells and Living Tissue

Introduction

In recent years, computer simulation has become more and more popular in the field of biology. Some researchers have begun to use computers to simulate various types of cells. This dissertation attempts to use the tensegrity model to simulate living cells and tissues.

According to the theories of Donald E. Ingber [1-3], the deformation of the cell is associated with the cytoskeleton which can support the architecture of the cell and this cytoskeleton could be simulated by using tensegrity model. The tensegrity model can receive an appropriate deformation when an external loading is applied on the tensegrity model. Simulating living tissue will involve multiple cells. Each cell is living in an extra cellular matrix, and it will interact with the other cells which are nearby. With the interaction, each cell should feel several external forces and the shape of the cell should be naturally deformed. For simulating the realistic activities of living cells and tissues, the system will be under the precondition that follows the law of physics. Thus, the technique for physics simulation will be used in the project. Physics simulation is widely used in industry now, especially for simulating the dynamic of the rigid body particles. There are a number of available physics engines which are popular to use. Appendix A contains a list of some physics engines which are available. Some of them are freely available and the others need to be licensed. However, the tensegrity is one of the soft body particles which can largely deform its shapes and the physics simulation of soft body particle is much more complex than rigid body particle. Thus, this dissertation will show the development of a specific system for implementing and testing the experimental work of this paper.

The deformation of the tensegrity model is a nonlinear problem. The dissertation used the linearization analysis method to solve this problem. For simulating the interaction between the tensegrities, the dissertation used the algorithm based on Lennard-Jones Potential. For simulating the interaction between the cell and extra cellular matrix, the dissertation used the algorithm based on the Stokes resistance [4,5]. After all the calculating of simulation, this dissertation combines the rigid body motion of the tensegrity and the deformation of the tensegrity. To visualize the simulation data, this project used the Microsoft DirectX SDK (Microsoft, 2010) as the 3D visualization platform. The extensive feature set of data structures and functionality of DirectX speeded up the efficiency of the development. Finally, the images which were visualized by the Microsoft DirectX SDK [6] were made into videos by Windows Live Movie Maker.

The application was implemented in the C++ language. The reason to choose C++ as the language is that the C++ language is one of the most general computer languages used in the industry. There are also two helpful tools used in this project – MATLAB and Mathematica. MATLAB [7] is a high-level language and interactive environment that enables you to perform computationally intensive tasks faster than with traditional programming languages such as C, C++, and Fortran. In this project, MATLAB was used for visualizing the testing results of the tensegrity model. This is because building the tensegrity model requires a great many tests and MATLAB could develop a simple visualization program quickly. Mathematica [8] is a computational software program used in scientific, engineering, and mathematical fields and other areas of technical computing. It was conceived by Stephen Wolfram and is developed by Wolfram Research of Champaign, Illinois. In the project, it was used for testing the result of the expressions for calculating the deformation of the tensegrity model and it also can export the c-form code which can be directly used in C++ program.

To achieve the expected result of the simulation, this dissertation is to investigate current models of the soft body particles and finally adapt the octahedron tensegrity as sample to build the tensegrity model for simulating the living cells. The following chapter 2 (Background) will conclude the methods which were used to build the tensegrity model, to simulate the interaction between cells and the physics interaction with extra cellular matrix, and which was used in visualization part. Chapter 3 (Methodology) will explain the all the algorithms in detail which were implemented in chapter 4 (Implementation) based on the

background. Chapter 4 will represent the procedure and structure of each implemented application with figures. Chapter 5 (Result and Discussing) will state the result of two applications which implemented in chapter 4 and compare the different results to analyze and discuss them. Chapter 6 (Conclusion) will summarize all the work which has been done in this dissertation and chapter 7 (Future Work) will give a brief introduction of the methods which may be implemented in the future.

Background

This chapter's purpose is to provide a review of the existent research related to each field of this project.

The tensegrity model for cell simulation

Tensegrity is a structural principle. The term "tensegrity" was first described by the architect R. Buckminster [9]. The sculpturer Kenneth Snelson [10] first gave a visualization of the tensegrity structure. According to the definition of tensegrity, tensegrity is a structure stable in shape due to continuous tension rather than due to continuous compression.

The concept of the tensegrity model has been proposed to interpret how the cytoskeleton works. The cells and tissues may be constructed as the tensegrity structure because the tensegrity model was corroborated to have many features just like living cells, such as cell stiffening or softening, non-liner mechanical responses and high-traction force with microtubules disruption [1,2,11-16]. With plenty of research on the cellular structure, the tensegrity model can be demonstrated as a reasonable structure to describe the complex cytoskeleton of living cells. There are many structures of tensegrity that could be used as a soft body model. The octahedron tensegrity structure which is made of six struts and the cuboctahedron tensegrity which is a 12-struts tensegrity model were popular used to simulate the cytoskeleton models. To simulate the activities of living cells and living tissue, the team in Taiwan [17] prefer to use the cuboctahedron tensegrity structure to simulate the cells. The reason to choose the cuboctahedron tensegrity is that the octahedron tensegrity had limitations soon after the beginning of spreading by declining energy stored in the structure. But in this dissertation, it added the volume reservation energy into the total internal energy of the octahedron tensegrity which may have solved this problem and also, for simplicity, this dissertation finally adopted the six-strut tensegrity model (The octahedron tensegrity structure).

To compute the position of the tensegrity model, the internal energy of the tensegrity needs to be analyzed. The internal energy of the tensegrity at first considered as just the cables energy and because of the energy stored in the tensegrity losing too quickly, the volume reservation energy is added into the internal energy of the tensegrity at last. The volume reservation energy is calculated depending on the volume of the tensegrity at each frame which makes the octahedron tensegrity do not reach the limitation too quickly. The detailed analysis of it will be discussed later in chapter 3.

Lennard-Jones potential for interaction between cells

The Lennard-Jones Potential was proposed by Sir John Edward Lennard-Jones [18]. It is a mathematical approximation that describes the energy of interaction between two nonbonding molecules or atoms which are based off their distance of separation. The Lennard-Jones Potential is a comparatively good approximation because of its simplicity. It is often used to illustrate the properties of gases and to model dispersion and overlap interactions in molecular models.

In this dissertation, the application tries to simulate the tissues that are pulled apart from the top and bottom side. But at first, the cells need an appropriate energy for interaction between them. For example, there are two cells separated by a large distance. Both cells are far enough apart to where they are not interacting with each other. If the cells are placed closer together with minimal energy to make them begin interacting. The cells could continuously be attracted closer and closer till they are almost touching. At this point, the separation distance between two cells is r and the repulsion force and attraction force acting on the cell are equal. When distance between two cells is less than r, the repulsion force should be far greater than the attraction force. The scenario is very similar to the interaction between two nonbonding molecules or atoms that described by the Lennard-Jones Potential.

Thus, the algorithm for calculating the interaction force between two tensegrities (cells) can be based on the Lennard-Jones Potential.

The tensegrity model which is used to simulate the cells simplify as 12 nodes. Each node can be regarded as an atom and the interaction between two cells can be simulated by using the Lennard-Jones Potential.

Interaction with Extra Cellular Matrix (Stokes resistance force)

To simulate the Interaction with extra cellular matrix, the methods are involved with the fundamentals of fluid dynamics in biomechanical system. This dissertation is to simulate living cells and living tissues. The Stokes resistance [4,5] can be a reasonable method to simulate the moving particle in inviscid fluid. For simplicity, this paper assumes the velocity of the fluid (the extra cellular matrix) is zero. Thus, in the static equilibrium, all the forces which are acting on the tensegrity model are equal to zero.

When the cell is moving in the static fluid, the resistance forces not only block the cell as the rigid body particle, but also have an effect on the deformation of the tensegrity. The resistance force can be decomposed to several forces acting on the surface of the tensegrity model. The decomposed forces hold the opposite direction to the direction of the velocity of the tensegrity model and the magnitude is based on the area of the triangle which belongs to the surface. If the resistance force was not decomposed on the surface, the deformation caused by the interaction from the fluid could not be reflected. On the other hand, the resistance force can be decomposed onto each joint node of the tensegrity model. This can be an alternative method. With those methods to decompose the resistance force, the simulation should become more realistic.

Rigid Body Dynamics

In physics, rigid body dynamics is the study of the motion of rigid bodies. Rigid body is an idealization of a solid body and has the properties of geometry, for example a center of mass, moments of inertia and so on (Because the tensegrity model is being used to simulate cells in this dissertation, it is assumed that the inertia of the cells can be ignored). Compared with deformable bodies, rigid bodies are characterised as non-deformable. In this project, the calculating of the position of the tensegrity was divided into two steps. The tensegrity first was considered as rigid body for calculating the velocity and torque of the tensegrity. After that, to compute the deformation of the tensegrity, the tensegrity model was also regarded as a deformable body which is composed of 12 joint nodes.

The rigid body dynamics of the tensegrity model includes two parts which are the rotation of the tensegrity and the translation of the tensegrity. To describe the rotation and the translation clearly, the several coordinate systems are involved. The relationship among the different coordinates is the key to understanding how the rigid body dynamics of the tensegrity work. The detailed explanation will be discussed in chapter 3.

Soft Body Deformation

The soft body model which is used in this project is the tensegrity model. For simulating the deformation of the tensegrity which is caused by the interaction between the tensegrities or tensegrity and fluid, the positions of the 12 joint nodes of each tensegrity need to be updated in each frame. In a rigid body motion step, the forces and the torques on each node of tensegrity are renewed. As soon as the forces and torques were calculated, they could be used to calculate the deformation of the tensegrity model. According to the principle of the virtual work, the virtual work of the external force and Internal energy are equal at each frame. Thus, the new position of each node of one tensegrity could be figured out following the algorithm this dissertation used (detail in chapter 3). The result of two steps can be combined and the final position of the tensegrity can be used for the next frame.

Scientific Visualization

After simulating the dynamics and the deformation of the tensegrities, the simulation data can be visualized by the computer.

The 3D rendering framework: The experimental application and test application all used the Microsoft DirectX SDK (Microsoft, 2010) as the rendering platform. The reason for choosing the Microsoft DirectX SDK is that it offers various sets of data structure and functions which can greatly improve the efficiency of development.

Although the Microsoft DirectX SDX was chosen as the rendering platform, the application in this dissertation did not use any proprietary functionality specific to DirectX. There are also some alternative methods to implement the 3D rendering platform such as a framework based on OpenGL.

Converting coordinate systems: All the coordinate systems for physics simulation are adopted the right-hand coordinate system and the DirectX for visualizing the simulate data is adopted the left-hand coordinate system. The calculations in the left- and right-hand coordinate systems are totally different. Therefore, the coordinate used by physics simulation must convert to left hand coordinate before the program generates the simulation data which is used for visualization.

There are two methods introduced by Paul Bourke [19] which can easily convert the coordinates. The first method involves inverting the x value (any single axe will do) of all vertices in the model and camera settings. The second uses the model and camera coordinates without hange but requires flipping of any rendered image horizontally.

Methodology

Background research constituted the requirements of the method used in this project. The following sections of this dissertation are a detailed explanation for each method used in the application system.

Six-struts Tensegrity Model

The tensegrity used in this project is composed of 6 slender struts and interconnected by 24 linear elastic cables.

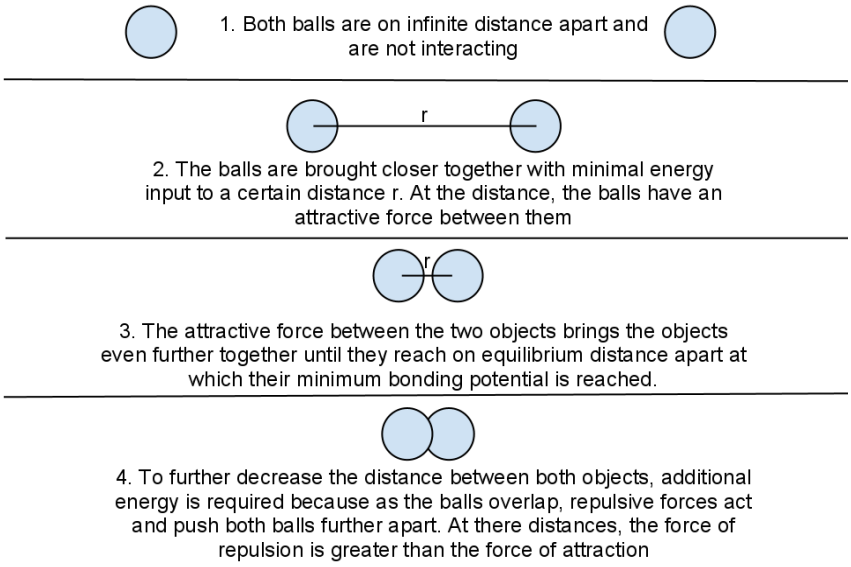


Figure 1: The interaction between two atoms based on Lennard-Jones Potential.

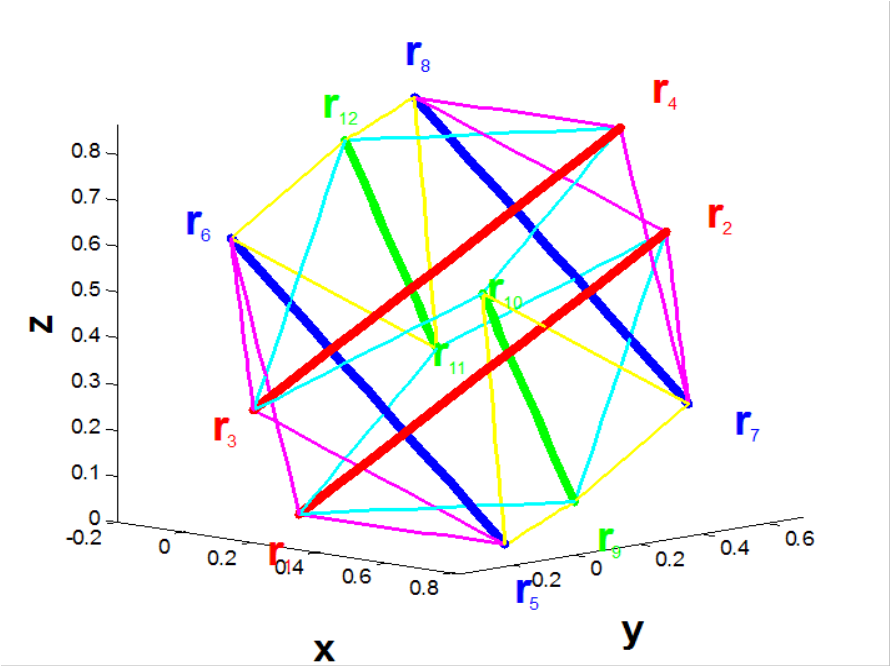


Figure 2: 3D six-strut tensegrity model visualizing by MATLAB.

There are six struts in the tensegrity structure that is used by this project and each strut has 5 degrees of freedom (explained later), which means one tensegrity has 30 degrees of freedom. The shape of the tensegrity can be represented using the position of the 12 joint nodes of the tensegrity. Each node has 3 components – x, y and z. To define a strut in 3D (three-dimensional), it is obvious that 6 components could describe the strut (The strut contains 2 nodes. Each node has 3 components.). If so, it necessitates 36 components to describe one tensegrity, which means 36 degrees of freedom. But it is not good enough. Therefore, it is necessary to introduce a Euler angle [20]. It can make the structure easier to represent.

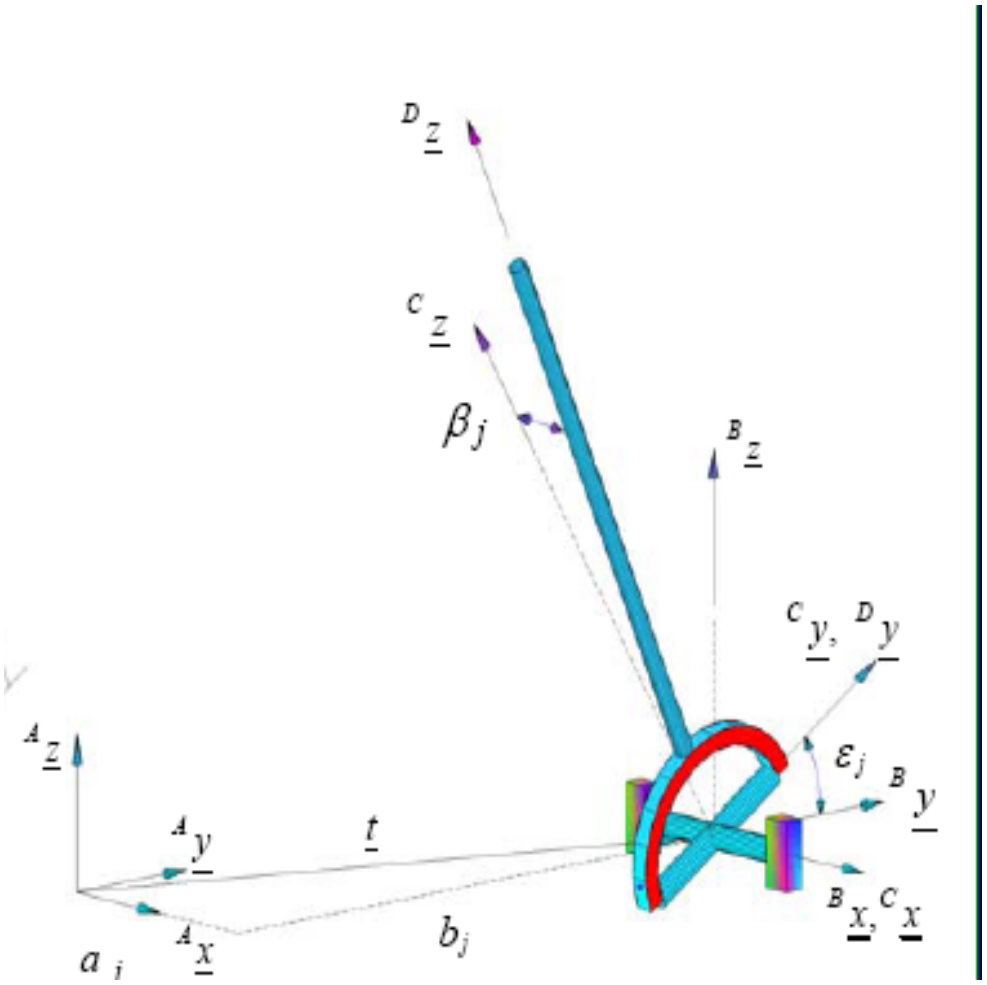


Figure 3: Euler angles used for reducing the degrees of freedom of strut.

For example, the strut r_1r_2 has two nodes ($r_1 : x_1, y_1, z_1$; $r_2 : x_2, y_2, z_2$), which means one struts has 6 components. If the Euler angles ε and β are added to describe the strut, the strut just needs to be represented using 5 components.

$$\begin{cases} x_{2*i-1} = a_i \\ y_{2*i-1} = b_i \\ z_{2*i-1} = c_i \end{cases} \quad (i = 1, 2, \cdots 6) \qquad (1)$$

$$\begin{cases} x_{2*i} = a_i + \sin\beta_i \\ y_{2*i} = b_i - \sin\varepsilon_i \cos\beta_i \\ z_{2*i} = c_i + \cos\varepsilon_i \cos\beta_i \end{cases} \quad (i = 1, 2, \cdots 6) \qquad (2)$$

Thus, the general components of one tensegrity can be indicated as below:

$$q=\{a_1,b_1,c_1,\beta_1,\varepsilon_1,a_2,b_2,c_2,\beta_2,\varepsilon_2,a_3,b_3,c_3,\beta_3,\varepsilon_3,a_4,b_4,c_4,\beta_4,\varepsilon_4,a_5,b_5,c_5,\beta_5,\varepsilon_5,a_6,b_6,c_6,\beta_6,\varepsilon_6\} \quad (3)$$

Therefore, one tensegrity can be described using 30 components. In other words, one tensegrity has 30 degrees of freedom.

The deformation coordinate does not need to consider the rigid body motion. Thus, the 6 degrees of freedom need to be fixed. In this project the 6 degrees of freedom were fixed in the way describing below:

$$\begin{cases} x_1=a_1\equiv 0 \\ y_1=b_1\equiv 0 \\ z_1=c_1\equiv 0 \\ y_5=b_3\equiv 0 \\ z_5=c_3\equiv 0 \\ z_9=c_5\equiv 0 \end{cases} \quad (4)$$

In Figure 1, it is the deformation coordinate which is used to represent how the tensegrity changes shape is shown. The six struts of the tensegrity are 5.0, 1.2, r_5r_6, r_7r_8, r_9r_{10} and $r_{11}r_{12}$. At the initial state, $\tilde{K}$ = parallel to r_3r_4. r_5r_6 parallel to r_7r_8. r_9r_{10} parallel to $r_{11}r_{12}$. r_1 is fixed on the center point of the coordinate. r_1r_5 parallel to x-axis. The y and z component of r_5 is fixed, therefore r_5 can only move along the x-axis. The z component of r_9 is fixed, hence r_9 is attached on the x-y plane.

The final general components in the deformation coordinate are:

$$q=\{\beta_1,\varepsilon_1,a_2,b_2,c_2,\beta_2,\varepsilon_2,a_3,\beta_3,\varepsilon_3,a_4,b_4,c_4,\beta_4,\varepsilon_4,a_5,b_5,\beta_5,\varepsilon_5,a_6,b_6,c_6,\beta_6,\varepsilon_6\} \quad (5)$$

Elastic energy of tensegrity:

Struts are assumed to be rigid body and cables are assumed to behave linear-elastically.

$$\xi=\frac{(l_0-l_R)}{l_0}=\frac{(l_0'-l_R')}{l_0'}=0.9 \quad (6)$$

l_0 is the initial length of the cable.

l_R is the natural length of the cable.

According to the assumption of cables, the reference force of the cable is:

$$F_R=\sqrt{\frac{8}{3}}\frac{E_cA_c}{(1-\xi)} \quad (7)$$

E_c is Young's modulus of cable.

A_c is the resting cross-section area.

The internal elastic energy of the tensegrity's cable can be expressed below:

$$
\begin{aligned}
\text{elastic} = \frac{1}{2}k\Bigg(&\left(\sqrt{(x_1-x_9)^2+(y_1-y_9)^2+(z_1-z_9)^2}-l_0\right)^2+\left(\sqrt{(x_1-x_{11})^2+(y_1-y_{11})^2+(z_1-z_{11})^2}-l_0\right)^2+\left(\sqrt{(x_2-x_9)^2+(y_2-y_9)^2+(z_2-z_9)^2}-l_0\right)^2+\\
&\left(\sqrt{(x_2-x_{11})^2+(y_2-y_{11})^2+(z_2-z_{11})^2}-l_0\right)^2+\left(\sqrt{(x_3-x_{10})^2+(y_3-y_{10})^2+(z_3-z_{10})^2}-l_0\right)^2+\left(\sqrt{(x_3-x_{12})^2+(y_3-y_{12})^2+(z_3-z_{12})^2}-l_0\right)^2+\\
&\left(\sqrt{(x_4-x_{10})^2+(y_4-y_{10})^2+(z_4-z_{10})^2}-l_0\right)^2+\left(\sqrt{(x_4-x_{12})^2+(y_4-y_{12})^2+(z_4-z_{12})^2}-l_0\right)^2+\left(\sqrt{(x_5-x_1)^2+(y_5-y_1)^2+(z_5-z_1)^2}-l_0\right)^2+\\
&\left(\sqrt{(x_5-x_3)^2+(y_5-y_3)^2+(z_5-z_3)^2}-l_0\right)^2+\left(\sqrt{(x_6-x_1)^2+(y_6-y_1)^2+(z_6-z_1)^2}-l_0\right)^2+\left(\sqrt{(x_6-x_3)^2+(y_6-y_3)^2+(z_6-z_3)^2}-l_0\right)^2+\\
&\left(\sqrt{(x_7-x_2)^2+(y_7-y_2)^2+(z_7-z_2)^2}-l_0\right)^2+\left(\sqrt{(x_7-x_4)^2+(y_7-y_4)^2+(z_7-z_4)^2}-l_0\right)^2+\left(\sqrt{(x_8-x_2)^2+(y_8-y_2)^2+(z_8-z_2)^2}-l_0\right)^2+\\
&\left(\sqrt{(x_8-x_4)^2+(y_8-y_4)^2+(z_8-z_4)^2}-l_0\right)^2+\left(\sqrt{(x_9-x_5)^2+(y_9-y_5)^2+(z_9-z_5)^2}-l_0\right)^2+\left(\sqrt{(x_9-x_7)^2+(y_9-y_7)^2+(z_9-z_7)^2}-l_0\right)^2+\\
&\left(\sqrt{(x_{10}-x_5)^2+(y_{10}-y_5)^2+(z_{10}-z_5)^2}-l_0\right)^2+\left(\sqrt{(x_{10}-x_7)^2+(y_{10}-y_7)^2+(z_{10}-z_7)^2}-l_0\right)^2+\left(\sqrt{(x_{11}-x_6)^2+(y_{11}-y_6)^2+(z_{11}-z_6)^2}-l_0\right)^2+\\
&\left(\sqrt{(x_{11}-x_8)^2+(y_{11}-y_8)^2+(z_{11}-z_8)^2}-l_0\right)^2+\left(\sqrt{(x_{12}-x_6)^2+(y_{12}-y_6)^2+(z_{12}-z_6)^2}-l_0\right)^2+\left(\sqrt{(x_{12}-x_8)^2+(y_{12}-y_8)^2+(z_{12}-z_8)^2}-l_0\right)^2\Bigg);
\end{aligned}
$$

Figure 4: The expression of the energy of the tensegrity's cables.

The Volume of tensegrity and volume reservation energy:

The surface of the tensegrity model is composed of 20 triangles. The volume of the tensegrity can be divided into 20 tetrahedral volumes which are based on the 20 triangles. The tetrahedron has 4 vertices. The triangle which is treated as the base of the tetrahedron holds three vertices of each tetrahedron and the fourth vertex for each tetrahedron is the center of the entire tensegrity. The procedure to calculate the volume of the tensegrity is shown below:

- Calculate the position of the center of the tensegrity.
- Take the three vertices of one triangle of the surface and the center point of the tensegrity to form a tetrahedron.
- Calculate the volume of each tetrahedron.
- Add the volumes of the 20 tetrahedron all together.

In step 3, the algorithm which is introduced by Math Pages [21] is the most general way to calculate the volume of tetrahedron.

After calculating the volume of the tensegrity, the volume reservation energy can be computed as:

$$W_{vol} = \frac{1}{2}k(\Delta V)^2 \tag{8}$$

If the internal energy only includes W_{cable}, the tensegrity model will soon reach the limitation when it begins to spread out. This is because the energy stored in the structure of tensegrity is losing too quickly. But, when the W_{vol} is added into the internal energy, the internal energy could be changed more slowly than before. Thus, the volume reservation energy can really improve the tensegrity to be a more stable model.

The Rigid Body Motion of Tensegrity

For visualizing the dynamic of tensegrities, the position of each tensegrity needs to be computed in each frame. The position of the tensegrity is combined from two parts. The first part is the position of the center of the tensegrity which is computed using the algorithm

from Rigid Body Dynamics. The second part is the position of each joint node of the tensegrity which is changed by the deformation of the tensegrity at each frame.

This section is to describe the Rigid Body Dynamics of the tensegrities. Rigid Body Dynamics includes two parts – rigid body rotation and rigid body translation.

Coordinate and Position:

For simulating physics dynamics, coordinates are a very crucial concept to describe the positions of all the particles. This section will introduce the basic concept and illustrate how the several coordinate systems work.

Rodrigues' Rotation: here are many ways to represent a rotation. This section will introduce the method called Rodrigues' Rotation [22] which was used in this project.

The Rodrigues' rotation formula is distinct from Euler–Rodrigues parameters and the Euler–Rodrigues formula for 3D rotation. In the theory of three-dimensional rotation, Rodrigues' rotation formula (named after Olinde Rodrigues) is an efficient algorithm for rotating a vector in space, given an axis and angle of rotation. By extension, this can be used to transform all three basis vectors to compute a rotation matrix from an axis angle representation. The Rodrigues' rotation method is more efficient than using the rotation matrix to calculate the rotated vector and converting ω and θ into a rotation matrix.

According to Rodrigues' rotation [22], to rotate a vector $\mathbf{r}^*$ angle θ about $\mathbf{n}$ axis, one gets

$$\mathbf{r}' = \mathbf{r}^* \cos\theta + (\mathbf{n} \times \mathbf{r}^*) \sin\theta + (\mathbf{n} \cdot \mathbf{r}^*)\mathbf{n}(1 - \cos\theta) \qquad (9)$$

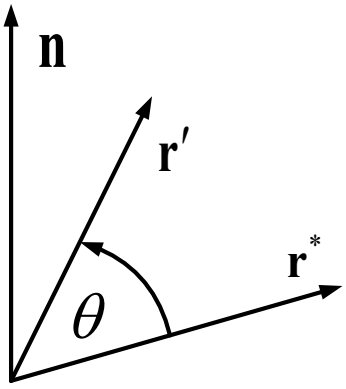


Figure 5: Rodrigues' rotation.

The component form of Eq (9) is

$$x_i' = x_i^* \cos\theta + \varepsilon_{ijk} n_j x_k^* \sin\theta + n_k x_k^* n_i (1 - \cos\theta) \qquad (10)$$

$$x_i' = \left[\delta_{ij} \cos\theta - \varepsilon_{ijk} n_k \sin\theta + n_i n_j (1 - \cos\theta)\right] x_j^* \qquad (11)$$

Let $\mathbf{R}_{\mathbf{n}}^{\theta}$ be rotation angle θ about $\mathbf{n}$ axis, here $\mathbf{n}$ is a normal vector along rotation axis, i.e. if arbitrary vector $\mathbf{r}^*$ is transformed into $\mathbf{r}'$ through rotation $\mathbf{R}_{\mathbf{n}}^{\theta}$, then it can be denoted as

$$\begin{pmatrix} x_1' \\ x_2' \\ x_3' \end{pmatrix} = \mathbf{R}_{\mathbf{n}}^{\theta} \begin{pmatrix} x_1^* \\ x_2^* \\ x_3^* \end{pmatrix} \tag{12}$$

If simply denote $\mathbf{r}' = \begin{pmatrix} x_1' \\ x_2' \\ x_3' \end{pmatrix}$ and $\mathbf{r}^* = \begin{pmatrix} x_1^* \\ x_2^* \\ x_3^* \end{pmatrix}$ then Eq. (12) may be rewritten as

$$\mathbf{r}' = \mathbf{R}_{\mathbf{n}}^{\theta} \mathbf{r}^* \tag{13}$$

then $\mathbf{R}_{\mathbf{n}}^{\theta}$ can be expressed in component form as

$$R_{kl} = \delta_{kl} \cos\theta - \sum_{m=1}^{3} \varepsilon_{klm} \sin\theta n_m + (1-\cos\theta) n_k n_l \tag{14}$$

Or

$$\mathbf{R} = \begin{pmatrix} n_1^2(1-\cos\theta)+\cos\theta & n_1 n_2(1-\cos\theta)-n_3\sin\theta & n_1 n_3(1-\cos\theta)+n_2\sin\theta \\ n_1 n_2(1-\cos\theta)+n_3\sin\theta & n_2^2(1-\cos\theta)+\cos\theta & n_2 n_3(1-\cos\theta)\text{-}n_1\sin\theta \\ n_1 n_3(1-\cos\theta)\text{-}n_2\sin\theta & n_2 n_3(1-\cos\theta)+n_1\sin\theta & n_3^2(1-\cos\theta)+\cos\theta \end{pmatrix} \tag{15}$$

where ε_{klm} is the component of permutation tensor, i.e.

$$\varepsilon_{klm} = \begin{cases} +1 \text{ if } (k,l,m) = (1,2,3) \text{ or } (2,3,1) \text{ or } (3,1,2) \\ -1 \text{ if } (k,l,m) = (1,3,2) \text{ or } (3,2,1) \text{ or } (2,1,3) \\ 0 \text{ otherwise} \end{cases} \tag{16}$$

Transformation: The transformations between the different coordinates are many and complicated. This section is to show a well-organized relationship between different coordinates.

As shown in Figure 6, there are five different coordinate systems:

- Space coordinate $O - \{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$
- Body coordinate $C - \{\mathbf{e}_1^*, \mathbf{e}_2^*, \mathbf{e}_3^*\}$
- Translation space coordinate $\mathbf{r}_1 - \{\mathbf{e}_1^*, \mathbf{e}_2^*, \mathbf{e}_3^*\}$
- Temporal space coordinate $\mathbf{r}_1 - \{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$
- Temporal body coordinate (Deformation coordinate) $\mathbf{r}_1 - \{\mathbf{e}_1^*, \mathbf{e}_2^*, \mathbf{e}_3^*\}$

Assuming arbitrary vector $\mathbf{r}$ can be write in both translation space coordinate $C - \{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$ and body coordinate $C - \{\mathbf{e}_1^*, \mathbf{e}_2^*, \mathbf{e}_3^*\}$

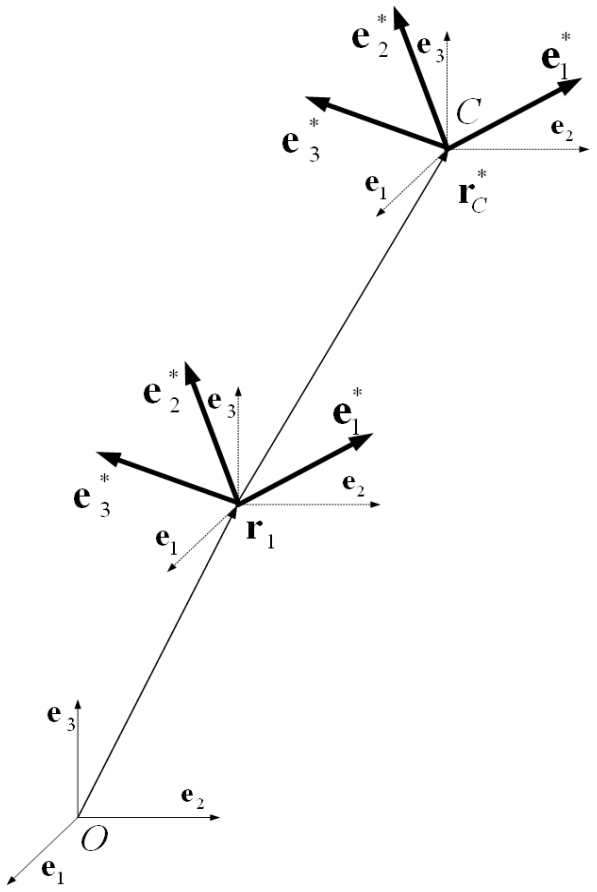


Figure 6: Coordinate systems for rigid body motion and deformation.

as

$$\mathbf{r} = \left(x_1, x_2, x_3\right)\begin{pmatrix}\mathbf{e}_1\\ \mathbf{e}_2\\ \mathbf{e}_3\end{pmatrix} = \mathbf{r}^* = \left(x_1^*, x_2^*, x_3^*\right)\begin{pmatrix}\mathbf{e}_1^*\\ \mathbf{e}_2^*\\ \mathbf{e}_3^*\end{pmatrix} \tag{17}$$

For example, a force can be decomposed as

$$\mathbf{F} = \left(F_1, F_2, F_3\right)\begin{pmatrix}\mathbf{e}_1\\ \mathbf{e}_2\\ \mathbf{e}_3\end{pmatrix} = \mathbf{F}^* = \left(F_1^*, F_2^*, F_3^*\right)\begin{pmatrix}\mathbf{e}_1^*\\ \mathbf{e}_2^*\\ \mathbf{e}_3^*\end{pmatrix} \tag{18}$$

Assuming

$$\mathbf{u}_1 = \mathbf{e}_1^* = \begin{pmatrix}R_{11}\\ R_{21}\\ R_{31}\end{pmatrix} \tag{19}$$

$$\mathbf{u}_2 = \mathbf{e}_2^* = \begin{pmatrix} R_{12} \\ R_{22} \\ R_{32} \end{pmatrix} \tag{20}$$

$$\mathbf{u}_3 = \mathbf{e}_3^* = \begin{pmatrix} R_{13} \\ R_{23} \\ R_{33} \end{pmatrix} \tag{21}$$

Or exactly

$$\mathbf{u}_1 = \mathbf{e}_1^* = (\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3) \begin{pmatrix} R_{11} \\ R_{21} \\ R_{31} \end{pmatrix} \tag{22}$$

$$\mathbf{u}_2 = \mathbf{e}_2^* = (\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3) \begin{pmatrix} R_{12} \\ R_{22} \\ R_{32} \end{pmatrix} \tag{20}$$

$$\mathbf{u}_3 = \mathbf{e}_3^* = (\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3) \begin{pmatrix} R_{13} \\ R_{23} \\ R_{33} \end{pmatrix} \tag{21}$$

i.e. the components of body basis vectors in space coordinate system.

Then the total Rodrigues' rotation matrix (compared with form (12)

$$\mathbf{R} = (\mathbf{e}_1^*, \mathbf{e}_2^*, \mathbf{e}_3^*) = \begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix} \tag{22}$$

Which is an alignment of three columns. Denote

$$\mathbf{M} = \mathbf{R}^T \tag{23}$$

Or

$$\begin{pmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{pmatrix} = \begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix}^T = \begin{pmatrix} R_{11} & R_{21} & R_{31} \\ R_{12} & R_{22} & R_{32} \\ R_{13} & R_{23} & R_{33} \end{pmatrix} \tag{24}$$

Then the transformation relationship reads

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix} \begin{pmatrix} x_1^* \\ x_2^* \\ x_3^* \end{pmatrix} \tag{25}$$

$$\begin{pmatrix} x_1^* \\ x_2^* \\ x_3^* \end{pmatrix} = \begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix}^T \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \tag{26}$$

Or in other words

$$\begin{pmatrix} \mathbf{e}_1^* \\ \mathbf{e}_2^* \\ \mathbf{e}_3^* \end{pmatrix} = \mathbf{R}^T \begin{pmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \\ \mathbf{e}_3 \end{pmatrix} = \begin{pmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{pmatrix} \begin{pmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \\ \mathbf{e}_3 \end{pmatrix} \tag{27}$$

$$\left(\mathbf{e}_1^*, \mathbf{e}_2^*, \mathbf{e}_3^*\right) = \left(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\right)\mathbf{R} = \left(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\right) \begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix} \tag{28}$$

Vis-à-vis

$$\begin{pmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \\ \mathbf{e}_3 \end{pmatrix} = \mathbf{R} \begin{pmatrix} \mathbf{e}_1^* \\ \mathbf{e}_2^* \\ \mathbf{e}_3^* \end{pmatrix} = \begin{pmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{pmatrix} \begin{pmatrix} \mathbf{e}_1^* \\ \mathbf{e}_2^* \\ \mathbf{e}_3^* \end{pmatrix} \tag{29}$$

$$\left(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\right) = \left(\mathbf{e}_1^*, \mathbf{e}_2^*, \mathbf{e}_3^*\right)\mathbf{R}^T = \left(\mathbf{e}_1^*, \mathbf{e}_2^*, \mathbf{e}_3^*\right) \begin{pmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{pmatrix} \tag{30}$$

Note that: in body coordination $C - \left\{\mathbf{e}_1^*, \mathbf{e}_2^*, \mathbf{e}_3^*\right\}$ and Temporal body coordinate $\mathbf{r}_1 - \left\{\mathbf{e}_1^*, \mathbf{e}_2^*, \mathbf{e}_3^*\right\}$, the force components are the same, so does torque. There is no difference for torque components between those two coordinate systems, although there is translation between two coordinate systems themselves.

Rigid Body Rotation:

In each iteration step we need to update $\left\{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3\right\} = \left\{\mathbf{e}_1^*, \mathbf{e}_2^*, \mathbf{e}_3^*\right\}$ in terms of torque acting on tensegrity according to

$$\mathbf{u}_i\left(t + \Delta t\right) = \Delta\mathbf{R} \cdot \mathbf{u}_i\left(t\right) \tag{31}$$

Where

$$\Delta\mathbf{R}=\mathbf{R}(\Delta\boldsymbol{\beta})$$

$$\begin{pmatrix} n_1^2(1-\cos\Delta\beta)+\cos\Delta\beta & n_1n_2(1-\cos\Delta\beta)-n_3\sin\Delta\beta & n_1n_3(1-\cos\Delta\beta)+n_2\sin\Delta\beta \\ n_1n_2(1-\cos\Delta\beta)+n_3\sin\Delta\beta & n_2^2(1-\cos\Delta\beta)+\cos\Delta\beta & n_2n_3(1-\cos\Delta\beta)-n_1\sin\Delta\beta \\ n_1n_3(1-\cos\Delta\beta)-n_2\sin\Delta\beta & n_2n_3(1-\cos\Delta\beta)+n_1\sin\Delta\beta & n_3^2(1-\cos\Delta\beta)+\cos\Delta\beta \end{pmatrix} \quad (32)$$

$$n_k=\frac{\omega_k^*}{\sqrt{\left(\omega_1^*\right)^2+\left(\omega_2^*\right)^2+\left(\omega_3^*\right)^2}} \quad (33)$$

$$\Delta\beta=\Delta t\sqrt{\left(\omega_1^*\right)^2+\left(\omega_2^*\right)^2+\left(\omega_3^*\right)^2} \quad (34)$$

$$\omega_k^*=\frac{\tau_k^*}{\eta}\left(k=1,2,3\right) \quad (35)$$

$$\boldsymbol{\tau}^*=\sum_{i=1}^{12}\boldsymbol{\tau}_i^*+\sum_{i=1}^{12}\mathbf{r}_{Ci}^*\times\mathbf{F}_i^* \quad (36)$$

In which $\mathbf{r}_{Ci}^*$ is the position difference between node i and mass center C, the component form should be expressed in bases $\left\{\mathbf{e}_1^*,\mathbf{e}_2^*,\mathbf{e}_3^*\right\}$, i.e. in coordinate $C-\left\{\mathbf{e}_1^*,\mathbf{e}_2^*,\mathbf{e}_3^*\right\}$ or $\mathbf{r}_1-\left\{\mathbf{e}_1^*,\mathbf{e}_2^*,\mathbf{e}_3^*\right\}$.

Rigid body translation:

In addition to calculating the torque in each frame, computing the velocity of the tensegrity is also important. This is because the translation of each tensegrity in each frame was calculated based on the velocity of the center of the tensegrity.

Assume

$$\mathbf{r}_{OC}=\left(x_1,x_2,x_3\right)\begin{pmatrix}\mathbf{e}_1\\ \mathbf{e}_2\\ \mathbf{e}_3\end{pmatrix} \quad (37)$$

Then

$$x_k\left(t+\Delta t\right)=x_k\left(t\right)+v_k\Delta t \quad (38)$$

v_k is the velocity of the tensegrity and the resultant force of one tensegrity is F_k.

$$v_k=\frac{F_k}{\mu} \quad (39)$$

F_k is composed of 12 forces which act on the 12 joint nodes of one tensegrity respectively.

Thus

$$F_k = \sum_{i=1}^{12} F_k^i \qquad (40)$$

Following the algorithm, which is described above, the torque and velocity of one tensegrity can be found out and the appropriate position of the rigid body of the tensegrity can be represented.

The Deformation of Tensegrity

The tensegrity model is a deformable model. One tensegrity model which this project used is composed of 12 joint nodes. Therefore, the deformation of the tensegrity model can be represented with the changing of each of its nodes. The forces and torques acting on each node have already been calculated from the computed result of the rigid body dynamics of the tensegrity. Thus, the tensegrities' position of the next frame can be computed based on these forces and torques. The process is shown below:

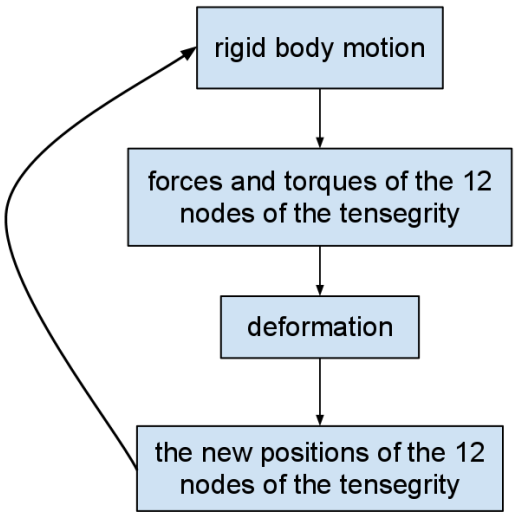


Figure 7: Procedure of the calculation of rigid body motion and deformation.

In the deformation step, the project used the linearization analysis method to calculate the new coordinate of each node in the tensegrity, in each frame. The position of each tensegrity will be changed at each frame because of the interaction forces from the other tensegrities and the fluid (extra cellular matrix). The forces acting on the tensegrity are changing in each frame. In other words, the different forces could be performed on the tensegrity over the time and the different shapes of the tensegrity could be revealed.

The key point of the deformation step is how to calculate the new coordinate of the tensegrity when the different forces are applied to it.

Assuming the global coordinate is

$$q = \left\{ q^1, r^1, \cdots q^N, r^N \right\} \qquad (41)$$

N is the total number of cells.

$$r^i = \left\{ x_0^i, y_0^i, y_0^i, e_1^i, e_2^i, e_3^i \right\} \qquad (42)$$

r^i is the coordinate for rigid body motion. q^i is the general coordinate to describe the deformation of cell i in its body fixed coordinate system.

For cell i

$$\delta W^i_{elastic} = Gp^i \cdot \left[\delta q^i\right]^{\mathrm{T}} \tag{43}$$

$$\delta W^i_{Vol} = Vp^i \cdot \left[\delta q^i\right]^{\mathrm{T}} \tag{44}$$

$$\delta W^i_{externalforce} = Ep^i \cdot \left[\delta q^i\right]^{T} \tag{45}$$

$$Ef^i = Ef^i\left(F^i_j\right) = \sum_j F^i_j \tag{46}$$

$$Ef^i = \begin{Bmatrix} EP_1, EQ_1, ER_1, EM_1, ET_1 \\ EP_2, EQ_2, ER_2, EM_2, ET_2 \\ EP_3, EQ_2, ER_2, EM_2, ET_2 \\ EP_4, EQ_2, ER_2, EM_2, ET_2 \\ EP_5, EQ_2, ER_2, EM_2, ET_2 \\ EP_6, EQ_2, ER_2, EM_2, ET_2 \end{Bmatrix} \tag{47}$$

$$Ep^i = Ep^i\left[Ef^i\left(F^i_j\right), q^i\right] \tag{48}$$

F^i_j is the interaction force which cell j applied on the cell i. Assuming the structure is currently in equilibrium state, i.e.

$$\delta W^i_{elastic} + \delta W^i_{Vol} - \delta W^i_{externalforce} = \left[Gp^i + Vp^i - Ep^i\right] \cdot \left[\delta q^i\right]^{\mathrm{T}} = 0 \tag{49}$$

$$Gp^i + Vp^i - Ep^i = 0 \tag{50}$$

In rigid body motion step, q^i remains unchanged. $r^i \to r^i + \Delta r^i$, due to rigid body motion, the interaction forces among cells need to be updated. Thus $F^i_j \to F^i_j + \Delta F^i_j$:

$$Ef^i \to \begin{Bmatrix} EP_1, EQ_1, ER_1, EM_1, ET_1 \\ EP_2, EQ_2, ER_2, EM_2, ET_2 \\ EP_3, EQ_2, ER_2, EM_2, ET_2 \\ EP_4, EQ_2, ER_2, EM_2, ET_2 \\ EP_5, EQ_2, ER_2, EM_2, ET_2 \\ EP_6, EQ_2, ER_2, EM_2, ET_2 \end{Bmatrix} + \Delta \begin{Bmatrix} EP_1, EQ_1, ER_1, EM_1, ET_1 \\ EP_2, EQ_2, ER_2, EM_2, ET_2 \\ EP_3, EQ_2, ER_2, EM_2, ET_2 \\ EP_4, EQ_2, ER_2, EM_2, ET_2 \\ EP_5, EQ_2, ER_2, EM_2, ET_2 \\ EP_6, EQ_2, ER_2, EM_2, ET_2 \end{Bmatrix} \tag{54}$$

Assume the configuration of tensegrity is increased by $q \to q + \Delta q$ (or) $q^i \to q^i + \Delta q^i$), $q + \Delta q$ satisfies

$$Gp^i\left(q^i + \Delta q^i\right) + Vp^i\left(q^i + \Delta q^i\right) - Ep^i\left[Ef^i + \Delta Ef^i, q^i + \Delta q^i\right] = 0 \tag{55}$$

Note that in Eq. (51) . Ep^i is a linear function of Ef^i, thus

$$\begin{aligned} & Ep^i\left[Ef^i + \Delta Ef^i, q^i + \Delta q^i\right] - Ep^i\left[Ef^i, q^i\right] \\ &= Ep^i\left[Ef^i + \Delta Ef^i, q^i + \Delta q^i\right] - Ep^i\left[Ef^i, q^i + \Delta q^i\right] \\ &\quad + Ep^i\left[Ef^i, q^i + \Delta q^i\right] - Ep^i\left[Ef^i, q^i\right] \end{aligned} \tag{56}$$

Or

$$\begin{aligned} &= Ep^i\left[\Delta Ef^i, q^i + \Delta q^i\right] + \frac{\partial Ep^i}{\partial q^i}\Delta q^i \\ &\approx Ep^i\left[\Delta Ef^i, q^i\right] + \frac{\partial Ep^i}{\partial q^i}\Delta q^i \end{aligned}$$

$$Ep^i\left[Ef^i + \Delta Ef^i, q^i + \Delta q^i\right] \approx Ep^i\left[Ef^i, q^i\right] + Ep^i\left[\Delta Ef^i, q^i\right] + \frac{\partial Ep^i}{\partial q^i}\Delta q^i \tag{57}$$

Denote $\Delta Ep^i = Ep^i\left[Ef^i\left(\Delta F_j^i\right), q^i\right]$, which is the external force corresponding to increment of interaction force, Eq. (55) can be rewritten as

$$\left[\frac{\partial Gp^i}{\partial q^i} + \frac{\partial Vp^i}{\partial q^i} - \frac{\partial Ep^i}{\partial q^i}\right]\Delta q^i = \Delta Ep^i + Ep^i - Gp^i - Vp^i \tag{58}$$

$$\Delta q^i = \left[\frac{\partial Gp^i}{\partial q^i} + \frac{\partial Vp^i}{\partial q^i} - \frac{\partial Ep^i}{\partial q^i}\right]^{-1}\left[\Delta Ep^i + Ep^i - Gp^i - Vp^i\right] \tag{59}$$

Incremental force:

Divide $F_j^i \to F_j^i + \Delta F_j^i$ into n step, $\delta F_j^i = \frac{1}{n}\Delta F_j^i$ $F_j^i \to F_j^i + \delta F_j^i$

$$F_j^i \to F_j^i + \delta F_j^i \to F_j^i + 2\delta F_j^i \to \cdots \to F_j^i + n\delta F_j^i = F_j^i + \Delta F_j^i \tag{60}$$

Start from the state of q^i and F_j^i

$$\delta q_1^i = \left[\frac{\partial Gp^i}{\partial q^i} + \frac{\partial Vp^i}{\partial q^i} - \frac{\partial Ep^i}{\partial q^i}\right]^{-1}\left[\delta Ep^i + Ep^i - Gp^i - Vp^i\right] \tag{61}$$

$$\delta q_1^i = \left[\frac{\partial Gp^i}{\partial q^i} + \frac{\partial Vp^i}{\partial q^i} - \frac{\partial Ep^i}{\partial q^i}\right]^{-1}\left[\frac{1}{n}\Delta Ep^i + Ep^i - Gp^i - Vp^i\right] \tag{62}$$

$$Gp^i\left(q^i+\delta q_1^i+\delta q_2^i\right)+Vp^i\left(q^i+\delta q_1^i+\delta q_2^i\right)-Ep^i\left[\left(Ef^i+\delta Ef^i\right)+\delta Ef^i,q^i+\delta q_1^i+\delta q_2^i\right]=0 \tag{63}$$

To the state of $q^i+\delta q_1^i$ and $F_j^i+\delta F_j^i$ by increasing δF_j^i, this causes the increment δq_2^i

$$\begin{aligned}
&Ep^i\left[\left(Ef^i+\delta Ef^i\right)+\delta Ef^i,q^i+\delta q_1^i+\delta q_2^i\right]-Ep^i\left[Ef^i+\delta Ef^i,q^i+\delta q_1^i\right]\\
&=Ep^i\left[\left(Ef^i+\delta Ef^i\right)+\delta Ef^i,q^i+\delta q_1^i+\delta q_2^i\right]-Ep^i\left[Ef^i+\delta Ef^i,q^i+\delta q_1^i+\delta q_2^i\right]\\
&\quad+Ep^i\left[Ef^i+\delta Ef^i,q^i+\delta q_1^i+\delta q_2^i\right]-Ep^i\left[Ef^i+\delta Ef^i,q^i+\delta q_1^i\right]\\
&=Ep^i\left[\delta Ef^i,q^i+\delta q_1^i+\delta q_2^i\right]+\frac{\partial Ep^i}{\partial q^i}\delta q_2^i\\
&\approx Ep^i\left[\delta Ef^i,q^i+\delta q_1^i\right]+\frac{\partial Ep^i}{\partial q^i}\delta q_2^i
\end{aligned} \tag{64}$$

$$\begin{aligned}
&Gp^i\left(q^i+\delta q_1^i\right)+\frac{Gp^i}{\partial q^i}\delta q_2^i+Vp^i\left(q^i+\delta q_1^i\right)+\frac{Vp^i}{\partial q^i}\delta q_2^i\\
&-\left\{Ep^i\left[Ef^i+\delta Ef^i,q^i+\delta q_1^i\right]+Ep^i\left[\delta Ef^i,q^i+\delta q_1^i\right]+\frac{\partial Ep^i}{\partial q^i}\delta q_2^i\right\}=0
\end{aligned} \tag{65}$$

$$\delta q_2^i=\left[\frac{\partial Gp^i}{\partial q^i}+\frac{\partial Vp^i}{\partial q^i}-\frac{\partial Ep^i}{\partial q^i}\right]^{-1}\left\{\begin{array}{c}Ep^i\left[Ef^i+\delta Ef^i,q^i+\delta q_1^i\right]+Ep^i\left[\delta Ef^i,q^i+\delta q_1^i\right]\\-Gp^i\left(q^i+\delta q_1^i\right)-Vp^i\left(q^i+\delta q_1^i\right)\end{array}\right\} \tag{66}$$

$$\delta q_2^i=\left[\frac{\partial Gp^i}{\partial q^i}+\frac{\partial Vp^i}{\partial q^i}-\frac{\partial Ep^i}{\partial q^i}\right]^{-1}\Bigg|_{q^i+\delta q_1^i}\left\{\begin{array}{c}Ep^i\left[Ef^i+2\delta Ef^i,q^i+\delta q_1^i\right]\\-Gp^i\left(q^i+\delta q_1^i\right)-Vp^i\left(q^i+\delta q_1^i\right)\end{array}\right\} \tag{67}$$

To the state of $q^i+\delta q_1^i+\delta q_2^i$ and $F_j^i+2\delta F_j^i$, in general

$$\begin{aligned}
&\delta q_m^i=\left[\frac{\partial Gp^i}{\partial q^i}+\frac{\partial Vp^i}{\partial q^i}-\frac{\partial Ep^i}{\partial q^i}\right]^{-1}\Bigg|_{q^i+\delta q_1^i+\cdots+\delta q_{m-1}^i}\\
&\left\{\begin{array}{c}Ep^i\left[Ef^i+m\delta Ef^i,q^i+\delta q_1^i+\cdots+\delta q_{m-1}^i\right]\\-Gp^i\left(q^i+\delta q_1^i+\cdots+\delta q_{m-1}^i\right)-Vp^i\left(q^i+\delta q_1^i+\cdots+\delta q_{m-1}^i\right)\end{array}\right\}
\end{aligned} \tag{68}$$

To the state of $q^i+\delta q_1^i+\cdots+\delta q_m^i$ and $F_j^i+m\delta F_j^i$, the coordinate and force finally arrivals to the state of $q^i+\delta q_1^i+\cdots+\delta q_n^i$

and

$$F_j^i + n\delta F_j^i = F_j^i + \Delta F_j^i .$$

For each incremental force step, it will denote initial state $q^i + \delta q_1^i + \cdots + \delta q_{m-1}^i = q_t^0$ and force $F_j^i + (m-1)\delta F_j^i = F^0$, start from this initial state to do test deformation,

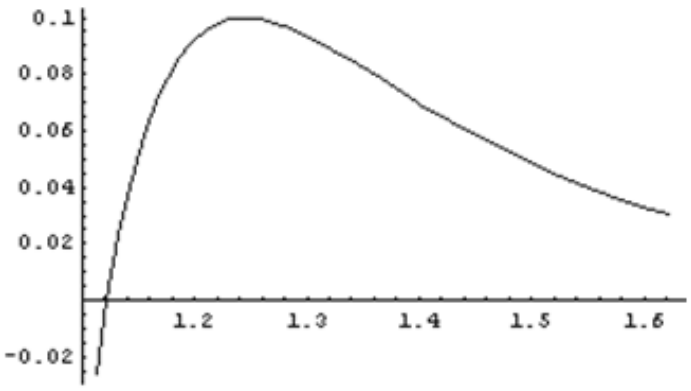
$$\delta q_t^k = \left[\frac{\partial Gp^i}{\partial q^i} + \frac{\partial Vp^i}{\partial q^i} - \frac{\partial Ep^i}{\partial q^i} \right]^{-1} \Bigg|_{q_t^k} \left\{ Ep^i \left[F^0 + \delta F_j^i, q_t^k \right] - Gp^i \left(q_t^k \right) - Vp^i \left(q_t^k \right) \right\} \tag{69}$$

$$q_t^{k+1} = q_t^k + \delta q_t^k \left(k = 0,1,\cdots K \right) \tag{70}$$

Please note that during this test process, the force $F^0 + \delta F_j^i$ keeps unchanged. Then we get modulus of the non-equilibrium force $tolf^k \left(k = 0,1,\cdots K \right)$, assume smallest modulus of the non-equilibrium force reads $tolf^l = Min_k \left\{ tolf^k \left(k = 0,1,\cdots K \right) \right\}$, then the performance for the real deformation starts from the state $q^i + \delta q_1^i + \cdots + \delta q_{m-1}^i = q_t^0$ and force $F_j^i + (m-1)\delta F_j^i = F^0$ increases to the state of $q_t^l = q_t^0 + \delta q_t^1 + \cdots + \delta q_t^l$ and force $F^0 + \delta F_j^i = F_j^i + m\delta F_j^i$. In real simulation, $\left(k = 0,1,\cdots K \right)$, K could be 5 or 10.

The Interaction between the Cells

For the realistic simulation, the interaction forces between cells need to follow an applicable rule. When the cells are far away from each other, the interaction force is almost zero. When the cells get closer and closer, the cells will attract each other. The shorter the distance between two cells, the larger the attraction force is. When the cells get much closer, the cell will not get closer anymore. Because the attraction force will decrease rapidly, and the interaction force will become the repulsion force. There is a diagram below shows how the force changes:



For this reason, the rule should be clear.

Each cell (tensegrity) has 6 struts, which means the cell has 12 nodes. The interaction force between two tensegrities could be simplified as the interaction between 12 nodes to 12 nodes.

The structure of the tensegrity shows below:

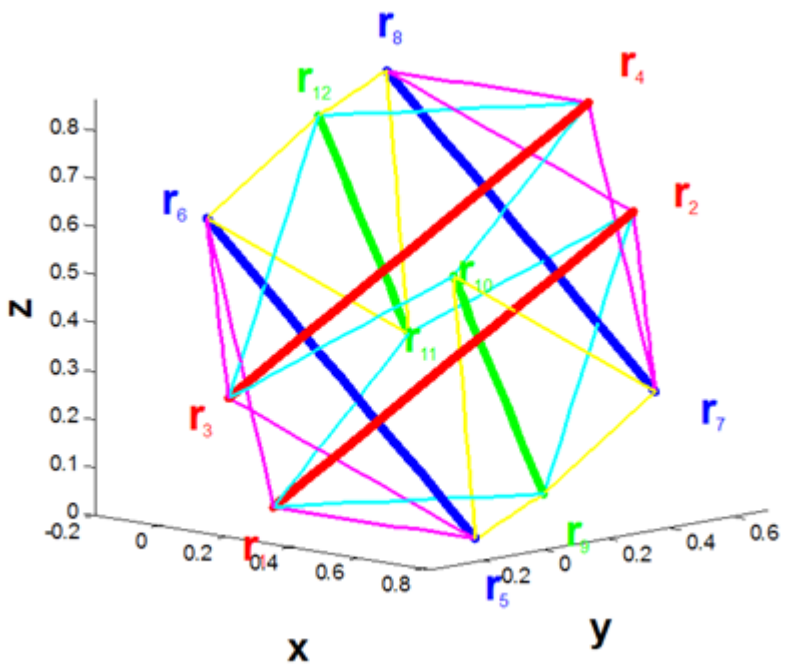


Figure 9: Structure of the octahedron tensegrity model.

For example, there are two cells (Cell 1 and Cell 2). The interaction force will be calculated following the algorithm describes below:

- Pick up one of the nodes from Cell 1 for example Node 1.
- Calculate the distances between Node 1 of Cell 1 and each of the nodes of Cell 2. Thus, there are 12 distances which are calculated in this step.
- With the values of the 12 distances, calculate the 12 interaction forces which are acting on Node1 of Cell 1 following the algorithm which involved Lennard-Jones Potential.
- Pick up the other nodes from Cell 1 (Node 2 ~ Node 12) and do the same procedure as step 1 ~ 3.

Finally, there are 144 calculated forces acting on Cell 1 and those forces are from 12 vertices of the tensegrity.

$$\mathbf{F}_{tensegrity} = \mathbf{F}_{vertex[1]} + \mathbf{F}_{vertex[2]} + \cdots + \mathbf{F}_{vertex[12]} \qquad (71)$$

The main part of the algorithm above is step 3 - Lennard-Jones Potential.

Lennard-Jones Potential is a very appropriate algorithm to rule the interaction between cells.

The common expressions of the Lennard-Jones Potential are shown below:

$$V = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \qquad (72)$$

r is the distance between two points.

ε is the depth of the potential well.

σ is the finite distance at which the inter-particle potential is zero.

V is the energy from Lennard-Jones Potential expression.

For the purpose describing the interaction force, the Lennard-Jones Potential expressions need to make some changes:

$$x = \frac{r}{\sigma} \tag{73}$$

$$V = 4\varepsilon\left[\left(\frac{1}{x}\right)^{12} - \left(\frac{1}{x}\right)^{6}\right] \tag{74}$$

The interaction force can be calculated by the potential.

$$F = \frac{dV}{dr} = \frac{dV}{dx}\frac{dx}{dr} = \frac{1}{\sigma}\frac{dV}{dx} \tag{75}$$

F is the force between two points.

$$F = \frac{1}{\sigma}\frac{dV}{dx} = 4\varepsilon\left[-12\left(\frac{1}{x}\right)^{13} + 6\left(\frac{1}{x}\right)^{7}\right] \tag{76}$$

$$F = \frac{1}{\sigma}\frac{dV}{dx} = 24\frac{\varepsilon}{\sigma}\left[\left(\frac{1}{x}\right)^{7} - 2\left(\frac{1}{x}\right)^{13}\right] \tag{77}$$

$$F = \frac{24\varepsilon}{\sigma}\left[\frac{1}{x^{7}} - \frac{2}{x^{13}}\right] \tag{78}$$

The plot of the formula (12) is shown below:

The horizontal direction of the plot indicates the distance between two points. The vertical direction of the plot represents the interaction force. The positive of the vertical direction means that the interaction force between two points is attraction force. The negative of the vertical direction means that the interaction force between two points is repulsion force.

$$F\left[x_0\right] = F\left[\sqrt[6]{2}\right] = 0 \tag{79}$$

$$F\left[r_0\right] = F\left[\sigma\sqrt[6]{2}\right] = 0 \tag{80}$$

$$\sqrt[6]{2} = 1.12246 \tag{81}$$

Thus, the algorithm could obtain the expected interaction force between cells which the experiment needs. The method described above may not be the best method, for example the tensegrity model could be also simplified as 20 triangles, but it is simple and still enough to get a good result for visualization.

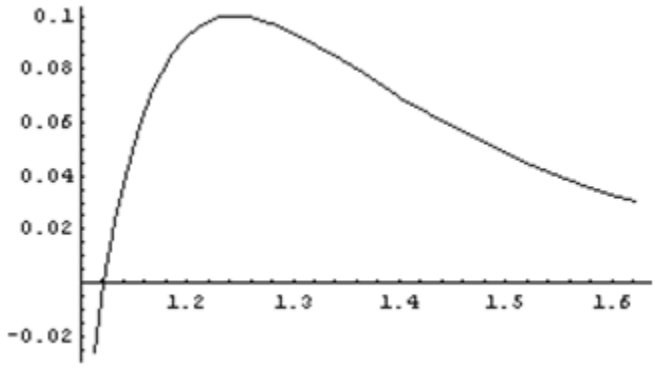


Figure 10: Plot of the expression of modified Lennard-Jones Potential.

The interaction between the cell and the extra cellular matrix

The cell will be submitted to the resistance force when it moves in the extra cellular matrix.

The project shall assume the velocity of the fluid (the extra cellular matrix) is zero. Thus, in the static equilibrium, all the forces which are acting on the tensegrity model are equal to zero.

$$\sum \mathbf{F} = 0$$

$$\mathbf{F}_{tensegrity} + \sum_{i=1}^{20} \mathbf{F}^{i}_{resistance} = 0 \tag{82}$$

$\mathbf{F}_{tensegrity}$ is the total force applied on the one tensegrity.

$\mathbf{F}^{i}_{resistance}$ is the resistance force which is acting on the triangle *i* of the tensegrity. The tensegrity is composed of 20 triangles. The resistance force acts on each triangle. Thus, $\mathbf{F}_{resistance}$ is combined with 20 resistance forces from 20 triangles. Each $\mathbf{F}^{i}_{resistance}$ is related to the area of the triangle and the normal vector of the triangle. It will be calculated using the formula (12) written below:

$$\mathbf{F}^{i}_{resistance} = -k * S^{i} \left(\mathbf{v} \cdot \mathbf{normal}^{i} \right) \mathbf{v} \tag{83}$$

S^{i} is the area of the triangle i (i = 1,2,3...,20).

$\mathbf{normal}^{i}$ is the normal vector of the triangle i (i = 1,2,3...,20).

$\mathbf{v}$ is the velocity's direction of the tensegrity and it can be calculated by $\mathbf{F}_{tensegrity}$.

$$\mathbf{v} = \mathbf{f}_{tensegrity} = \frac{\mathbf{F}_{tensegrity}}{\left| \mathbf{F}_{tensegrity} \right|} \tag{84}$$

Thus, $\mathbf{V}$ can be calculated as:

$$\mathbf{V} = V * \mathbf{v} \tag{85}$$

$$V = \frac{\mathbf{F}_{tensegrity}}{\sum_{i=1}^{20}\left(k * S^i\left(\mathbf{v}\cdot\mathbf{normal}^i\right)\mathbf{v}\right)}$$

$$V = \frac{\left|\mathbf{F}_{tensegrity}\right|}{k * \sum_{i=1}^{20} S^i\left(\mathbf{v}\cdot\mathbf{normal}^i\right)} \tag{86}$$

Converting right hand to left hand coordinate system

The visualization of the application is based on Microsoft DirectX SDK which was implemented using the left-hand coordinate. Unfortunately, the coordinates used by simulation are all employed by the right-hand coordinate. The right-hand coordinate should be converted to left hand coordinate before implementing with DirectX. Both of the methods introduced in chapter 2.6.2 are appropriate for implementing. For simplicity, the dissertation used the first method – inverting the x coordinates which is shown below:

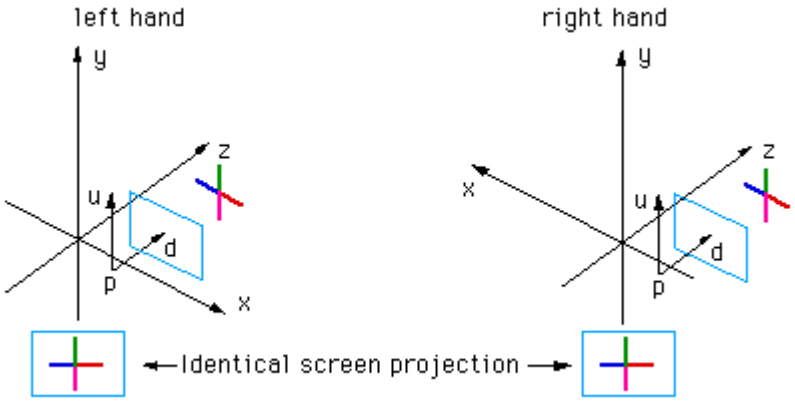


Figure 11: Inverting the x coordinates for converting coordinate systems.

The symbols p, d, and u represented the vectors position, view direction and up vector respectively.

Implementation

This chapter will illustrate how the applications were implemented.

Building the Tensegrity Model (Case A)

To simulate the multiple cells interacting with each other, the tensegrity model needs to be stable. To obtain a stable tensegrity model, a number of tests are very necessary. For this reason, the application for testing the stability of the tensegrity model was divided into two parts (Figure 12) – simulation and visualization, which made the application much easier to debug.

The simulation part was developed in the C++ language and the visualization part was developed with MATLAB.

The core algorithm to build the tensegrity model was implemented in the simulation part. The procedure of the simulation part is illustrated as below:

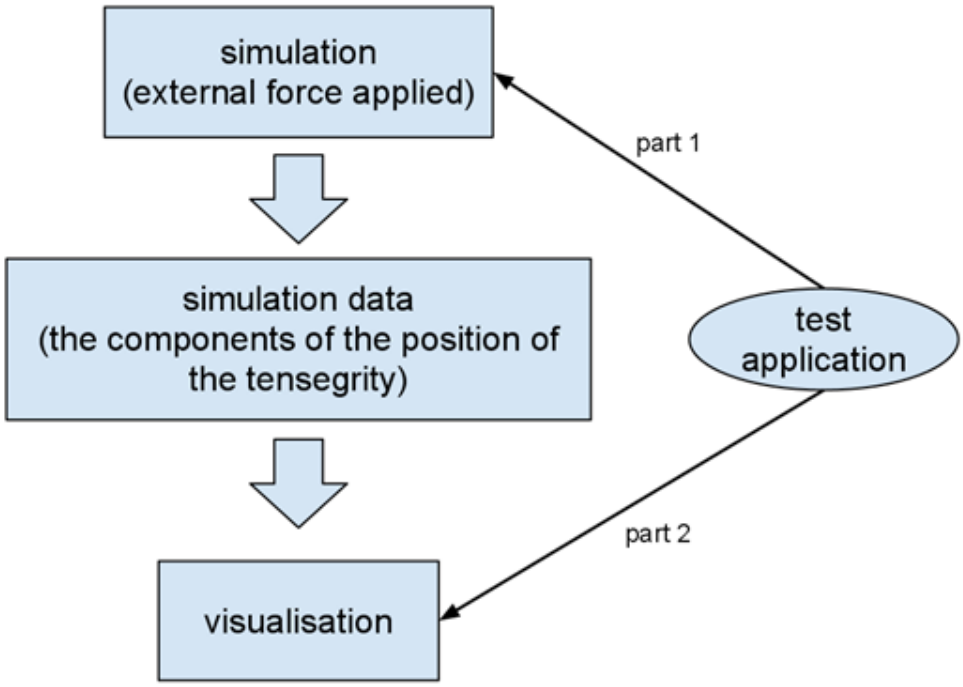


Figure 12: Structure of the test application.

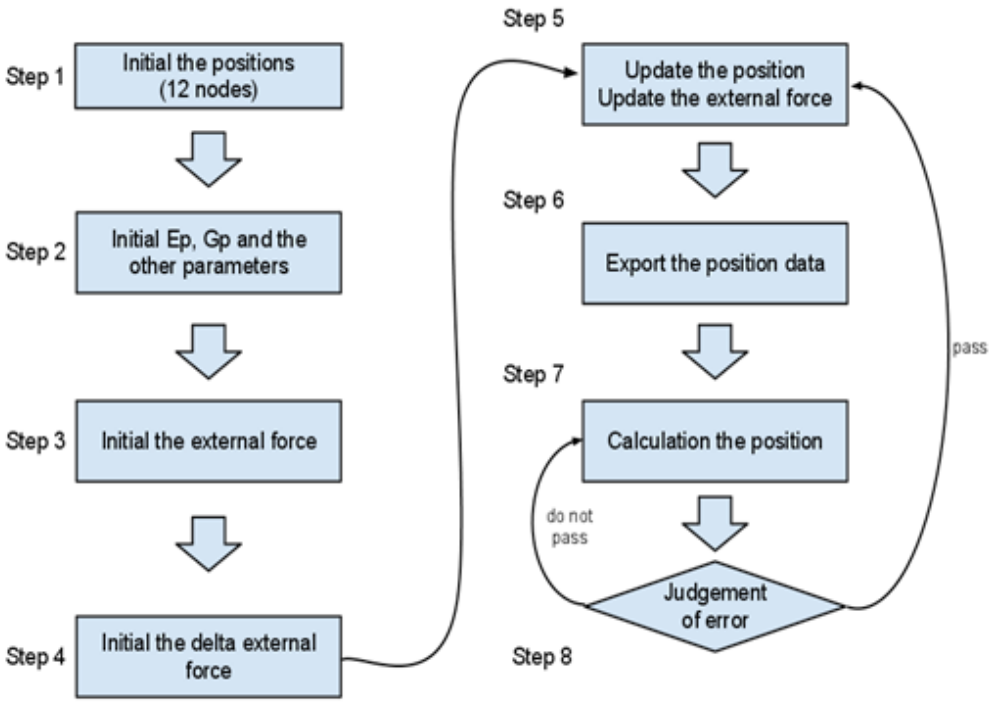


Figure 13: Procedure of the simulation of the test application.

At the step 1, the position of each node was initialed. For simplicity, the test application used a specific coordinate. The array (Position [12][3]) of the position is shown below:

```
{-0.2500,   -0.5000,        0},
{-0.2500,    0.5000,        0},
{ 0.2500,   -0.5000,        0},
{ 0.2500,    0.5000,        0},
{      0,   -0.2500,  -0.5000},
{      0,   -0.2500,   0.5000},
{      0,    0.2500,  -0.5000},
{      0,    0.2500,   0.5000},
{-0.5000,         0,  -0.2500},
{ 0.5000,         0,  -0.2500},
{-0.5000,         0,   0.2500},
{ 0.5000,         0,   0.2500}};
```

Figure 14: The initial position of the tensegrity model in test application.

These are the positions of the 12 nodes of one tensegrity which is shown above. The order of the points in the array is following the order in chapter 3.1 ($r_1, r_2, r_3 \cdots r_{12}$). But the deformation of the tensegrity model should be changed by the Deformation Coordinate $r_1 - \{\mathbf{e}_1^*, \mathbf{e}_2^*, \mathbf{e}_3^*\}$. Thus, to get the correct initial position of the tensegrity, the coordinate of the tensegrity model should change to $r_1 - \{\mathbf{e}_1^*, \mathbf{e}_2^*, \mathbf{e}_3^*\}$. The method used to change the coordinate is described below:

Firstly, using the position of the first, fifth and ninth points to compute $\mathbf{e}_3^*$.

Secondly, because the r_5 is on the x-axis in the $r_1 - \{\mathbf{e}_1^*, \mathbf{e}_2^*, \mathbf{e}_3^*\}$ (detail in chapter 3.1), that the position of the fifth point minus the position of the first point can get the $\mathbf{e}_2^*$. Then they $\mathbf{e}_1^*$ could be calculated by $\mathbf{e}_2^*$ and $\mathbf{e}_3^*$ (cross product of $\mathbf{e}_3^*$ and $\mathbf{e}_2^*$) and now the coordinate is $C - \{\mathbf{e}_1^*, \mathbf{e}_2^*, \mathbf{e}_3^*\}$. Thirdly, translate r_1 to the center point of the coordinate ($r_1 - \{\mathbf{e}_1^*, \mathbf{e}_2^*, \mathbf{e}_3^*\} \rightarrow C - \{\mathbf{e}_1^*, \mathbf{e}_2^*, \mathbf{e}_3^*\}$).

At step 2, the program initialized the Ep, Gp, Vp and some other parameters which would be used in the next step. The parameters as Ep,Gp and Vp were detailed described from forms (12) to (12) that are in chapter 3.3.

For calculating the Vp, the volume of the tensegrity is required to compute. The functions below were implemented to calculate the volume of the tensegrity:

```
//compute the volume of the tetrahedron

ComputeTetrahedronVolume(D3DXVECTOR3 *p0,

                         D3DXVECTOR3 *p1,

                         D3DXVECTOR3 *p2,

                         D3DXVECTOR3 *p3,

                         float &vol)

{

    D3DXVECTOR3 u = *p1 - *p0;

    D3DXVECTOR3 v = *p2 - *p0;

    D3DXVECTOR3 w = *p3 - *p0;

    D3DXVECTOR3 temp = D3DXVECTOR3(0.0f,0.0f,0.0f);

    D3DXVec3Cross(&temp, &u, &v);

    vol = D3DXVec3Dot(&temp, &w) / 6.0f;

    return vol;

}
```

Figure 15: The function for calculating the volume of the tetrahedron.

The code shown above followed the algorithm introduced by Math pages [21].

The code shown above followed the algorithm described in chapter 3.1.2. The array kk[60] has 60 elements which represent the 20 triangles from the surface of the tensegrity, and it is also the index of the triangles which are used in visualization part.

The expressions of the parameters such as Ep and Gp which were used in C++ program are generated from Mathematica directly. To use the expression export from Mathematica directly, the header file called mdefs.h required to be added into the test application. This header file would transform the functions' name using in the Mathematica which was running on the operating platform.

```
//compute the volume of the tensegrity

ComputeTensegrityVolume(D3DXVECTOR3* center,float& volume)

{

	volume = 0.0f;

	//20 triangles(Surface)

	for(int i = 0; i < 20; i++)

	{

		float tempVol = 0.0f;

		 //compute the volume of each tetrahedron in the tensegrity

		ComputeTetrahedronVolume(center, &vertexWorld[kk[3*i]], &vertexWorld[kk[3*i+1]],
&vertexWorld[kk[3*i+2]], tempVol);

		volume += tempVol;

	}

	return volume;



}
```

Figure 16: The function for calculating the volume of the tensegrity.

At step 3, the program initialized the external force. The external force is composed of 54 components which includes the 36 components which are from the forces that are acting on the 12 nodes and the 18 components which are from the 6 torques of the 6 struts of the tensegrity model. Normally, the external force is initialized as zero.

```
//initial the external force: the first 36 components are for the 12 forces acting on the each of nodes,

//the rest 18 components are for the 6 torques from the 6 struts.

for (int i=0; i<54; i++)

{

   externalFore[i] = 0.0;

}
```

Figure 17: The initialization of the external force in Case A.

At step 4, the program initialized the delta of the external force which defines the increment of the force between two-time steps. The different increments of the force could cause different deformations of the tensegrity model. The example below shows how the increments of the delta external force work.

```
while(loadstep < numberOfSteps)

{

------

   for (int k=0; k<54; k++)

   {

      externalFore[k] += dExternalFore[k];

   }

------

}
```

Figure 18: The delta external force working in Case A.

At step 5, the program would update the external force following the definition in step 4 and the position of the tensegrity model (12 nodes) would be updated at this step.

At step 6, the program would export the position data of the tensegrity model which was used to visualize the deformation of the tensegrity with MATLAB. The figure below describes the way for outputting the simulation data in c++:

```
//output the simulation data in each time step
int ic = loadstep/1; //the number of simulation data

char filename[80] = "data\\out" ;
char str[80];

sprintf(str, "%d", ic);

strcat(filename, str);
strcat(filename, ".txt" );

FILE *out;
out = fopen(filename,"w");
if(out==NULL)
{
	printf("    can not open %s \n" , filename);
}
//output the positions of the 12 nodes of the tensegrity model
for(int i=0;i<12; i++)
{
	fprintf(out,"%    e          %e          %e       \n",    position[i][0], position[i][1], position[i][2]);

}
printf("    loadstep = %d    iteratestep = %d    \n", ic, iteratestep);
fclose(out);
```

Figure 19: The function for output simulation data in Case A.

There is the position of the 12 nodes of the tensegrity ordered by. The format of the simulation data is shown below:

0.000000e+000	0.000000e+000	0.000000e+000
4.082483e-001	7.071068e-001	5.773503e-001
2.041241e-001	-3.535534e-001	2.886751e-001
6.123724e-001	3.535534e-001	8.660254e-001
6.123724e-001	0.000000e+000	0.000000e+000
-2.041241e-001	0.000000e+000	5.773503e-001
8.164966e-001	3.535534e-001	2.886751e-001
0.000000e+000	3.535534e-001	8.660254e-001
3.061862e-001	5.303301e-001	0.000000e+000
7.144345e-001	-1.767767e-001	5.773503e-001
-1.020621e-001	5.303301e-001	2.886751e-001
3.061862e-001	-1.767767e-001	8.660254e-001

Figure 20: The format of the simulation data in Case A.

At step 7, the new position of the tensegrity was computed. The algorithms described in chapter 3.2 and chapter 3.3 were used here. For solving the matrix in form (12), the program used Gaussian elimination method which was implemented by C++ from Numerical Recipes Third Edition [23]. After calculating the new position of the tensegrity, the parameters such as Ep and Gp would be generated again depending on the new positions. Although the new position had been calculated by the core formula (12), the errors from the calculation still need to be considered. The judgement of the errors would be dealt with in step 8.

At step 8, the errors were calculated, and the judgment of the error could be set depending on the requirement of test. If the result passed the judgement, the program would go back to step 5, which means the program continued to generate the next new position of the tensegrity model with the updated external force. If the result did not pass the judgement of error, the error would be passed to step 7 (the method detail explained in chapter 3.2) until the result passed the judgement.

```
//calculate the error(tolf)

tolf=0.0;



for (int k=0; k<24; k++)

{

        tolf += tempVec[k]*tempVec[k];

}
```

Figure 21: The function for calculating the error in Case A.

The variable temp Vec is $\Delta Ep^i + Ep^i - Gp^i - Vp^i$ which on the right side of the formula (12)

$$\left[\frac{\partial Gp^i}{\partial q^i} + \frac{\partial Vp^i}{\partial q^i} - \frac{\partial Ep^i}{\partial q^i}\right]\Delta q^i = \Delta Ep^i + Ep^i - Gp^i - Vp^i .$$

Interaction of Cells (Case B)

To simulate the activities of tissue, the application (Case B) implemented the multiple cells which interacted with each other and the extra cellular matrix. The procedure of the program is shown below:

As in the Figure 21 shown, the application is also composed of two parts-the simulation part and the visualization part. Compared with the application of Case A, Case B was trying to visualize the simulation in real time. The simulation data would not be exported as files stored on the computer. Oppositely, the simulation data would be used directly as the parameters for the calculation of the simulation of the next time step. The procedure of the simulation part is illustrated below:

At the step 1, the program initialized the position of each tensegrity in the coordinate $C - \left\{\mathbf{e}_1^*, \mathbf{e}_2^*, \mathbf{e}_3^*\right\}$ which is the Right-hand coordinate.

At the step 2, the force and torque acting on each node of the tensegrity was computed one by one. Force and torque could be computed in two ways. The first method computes the force and torque based on point. The second method is based on faces (triangles) which will be given a brief introduction in chapter 7 (Future Work).

To compute the force and torque, Case B was implemented by using the first method (based on point). The force and torque were computed one by one. For example, there are 3 tensegrities – A, B and C. Firstly, the program will calculate the force and torque acting on tensegrity A with tensegrity B. Secondly, the program will calculate it with tensegrity C. Thirdly, with the result of the first two steps, the program will calculate the total force and torque acting on each node of tensegrity A. In the first two steps, all the calculations were between two tensegrities such as tensegrity A with tensegrity B, and tensegrity A with tensegrity C. For simplicity, the tensegrity model was simplified as 12 joint nodes, and the force and torque would be calculated as the figure showing below:

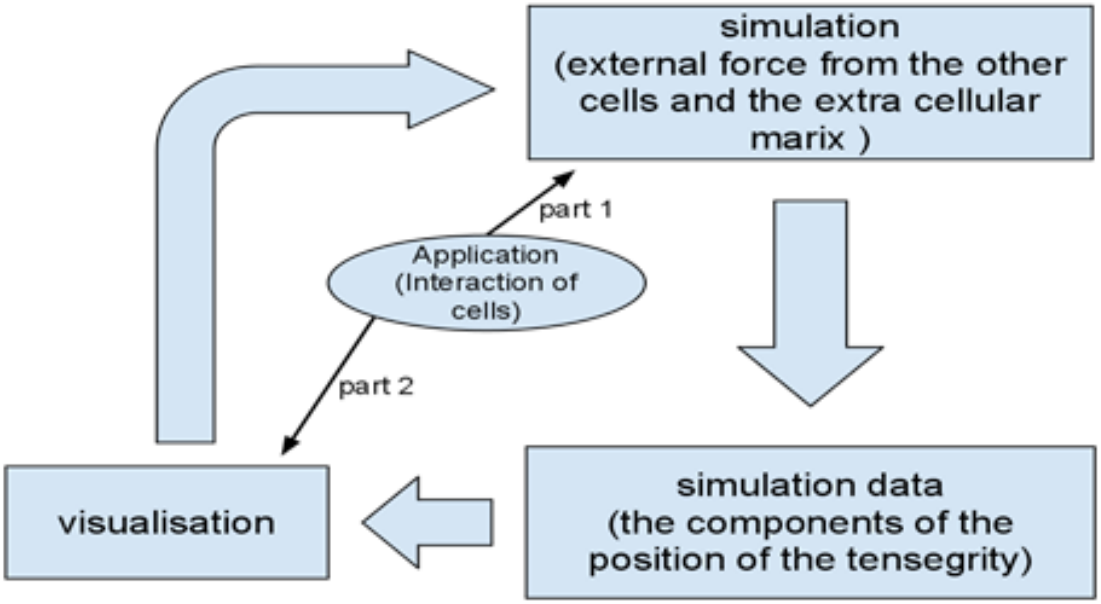


Figure 22: The structure of the application for Interaction of cells.

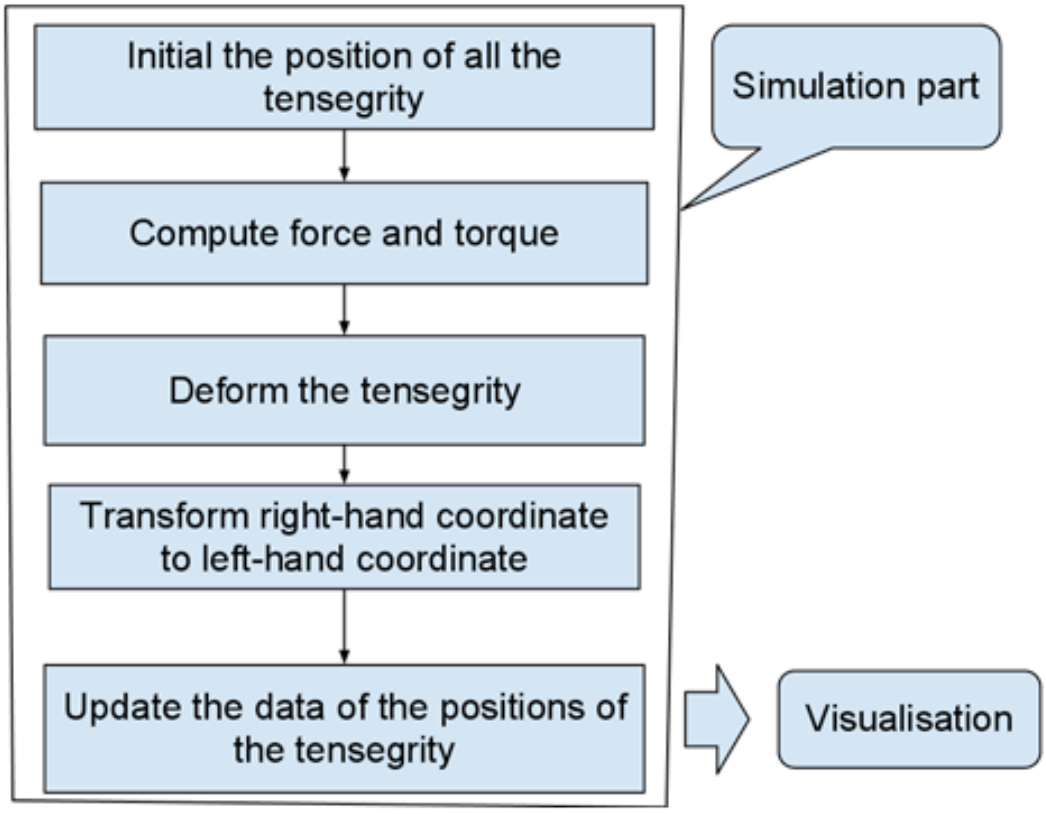


Figure 23: The procedure of the application for Interaction of cells.

```
(-0.306186f, -0.176777f, -0.433013f);

(0.102062f, 0.53033f, 0.144338f);

(-0.102062f, -0.53033f, -0.144338f);

(0.306186f, 0.176777f, 0.433013f);

(0.306186f, -0.176777f, -0.433013f);

(-0.51031f, -0.176777f, 0.144338f);

(0.51031f, 0.176777f, -0.144338f);

(-0.306186f, 0.176777f, 0.433013f);

(0.0f, 0.353553f, -0.433013f);

( 0.408248f, -0.353553f, 0.144338f);

(-0.408248f, 0.353553f, -0.144338f);

(0.0f, -0.353553f, 0.433013f);
```

Figure 24: The initial position of the application for Interaction of cells.

```
//Compute the force and torque

for(int i = 0; i<numOfTensegrity; i++)

{

      //exf is the series number of the tensegrity

      if(i != (exf-1))

      {

            ……

            //There are two method to calculate the force and torque acting on the tensegrity

            //Computed by face - 20 faces(triangles).

            //ComputeForceAndTorqueTensegrityToSelf(deltaTime, t[i], FtVertex, TorqueVertex);

            //Computed by point - 12 points(nodes)

      ComputeForceAndTorqueTensegrityToSelfByPoint(deltaTime, t[i], FtVertex, TorqueVertex);

            ……

      }

}
```

Figure 25: The function used for computing force and torque in Case B.

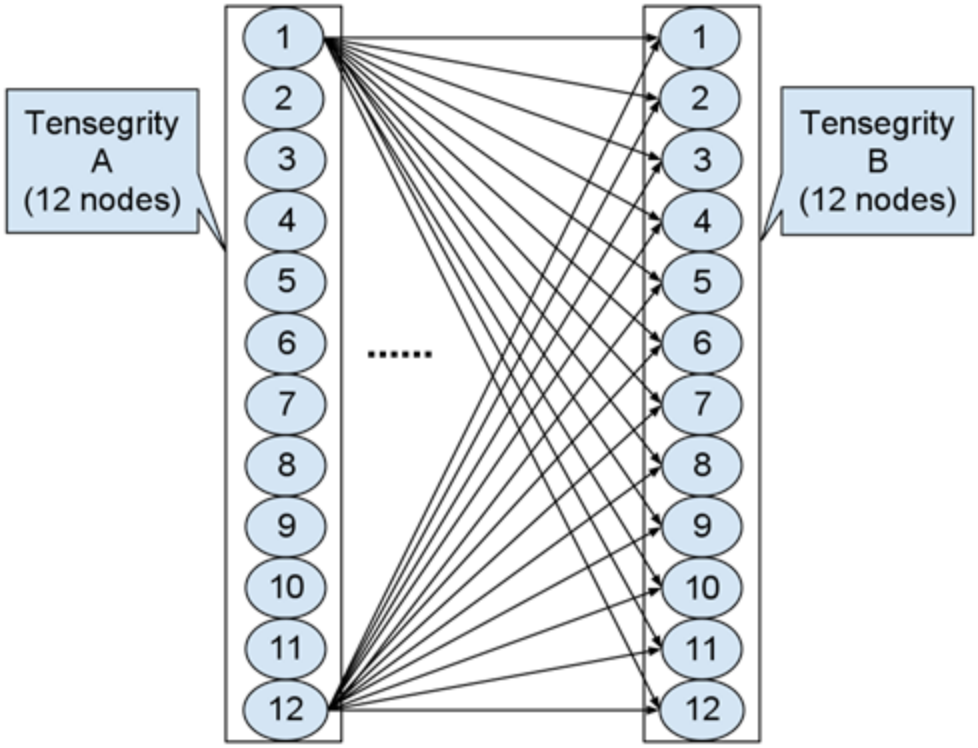


Figure 26: The interaction between two tensegrities (12 nodes).

Each tensegrity has 12 nodes. Each node of tensegrity A was used to compute the force and torque on it based on the Lennard-Jones Potential which was described in chapter 3.3 in detail.

At the step 3, the program computed the position of the tensegrity after it deformed. The main function which was implemented in this step followed the algorithm described in chapter 3.3 which was the same algorithm implemented in Case A. The force and torque which was used to calculate the deformation of the tensegrity was given from step 2.

It should be noticed that the forces and torques which are used for calculating the rigid body motion and the forces and torques used for computing the deformation of the tensegrity are not in the same coordinate. For computing the deformation, the forces and torques are required to transform the coordinates which was represented in chapter 3.2.1.2. The forces and torque would be used by the deformation part of the program after transforming the coordinate of them. The function below initialized the external force with the transformed forces and torques.

Ft Vertex is the force acting on each of the tensegrity's nodes.

Torque Vertex is the torque acting on each of the tensegrity's nodes.

At step 4, the program converted the Right-hand coordinate to the Left-hand coordinate. This is because all the coordinates used in the physics simulation were Right-hand coordinate and the 3D rendering platform which was implemented by Microsoft DirectX uses the Left-hand coordinate. The method for converting the coordinate is particularly described in chapter 3.6 and chapter 4.5.

At step 5, the new positions of the tensegrity were updated to the visualization part directly, which means the data of the positions of the tensegrity used to render were in real time.

Visualization

For visualizing the simulation data in real time, the application used the standard 3D pipeline based on DirectX. The 3D pipeline is shown below:

```
//The deformation of the tensegrity model

//initial the external force with the transformed data calculated in the rigid body motion part

double    externalFore[54];

for(int i =0; i<12;i++)

{

      //there are 12 forces(36 components) from 12 nodes

      externalFore[3*i] = FtVertex[i].x;

      externalFore[3*i+1] = FtVertex[i].y;

      externalFore[3*i+2] = FtVertex[i].z;

}



for(int i = 12, j = 0; i<18; i++)

{

      //there are 6 torques(18 components) from 6 struts(12 nodes)

      externalFore[3*i] = (TorqueVertex[2*j]+TorqueVertex[2*j+1]).x;

      externalFore[3*i+1] = (TorqueVertex[2*j]+TorqueVertex[2*j+1]).y;

      externalFore[3*i+2] = (TorqueVertex[2*j]+TorqueVertex[2*j+1]).z;

      j++;



}
```

Figure 27: The initialization of the external force in Case B.

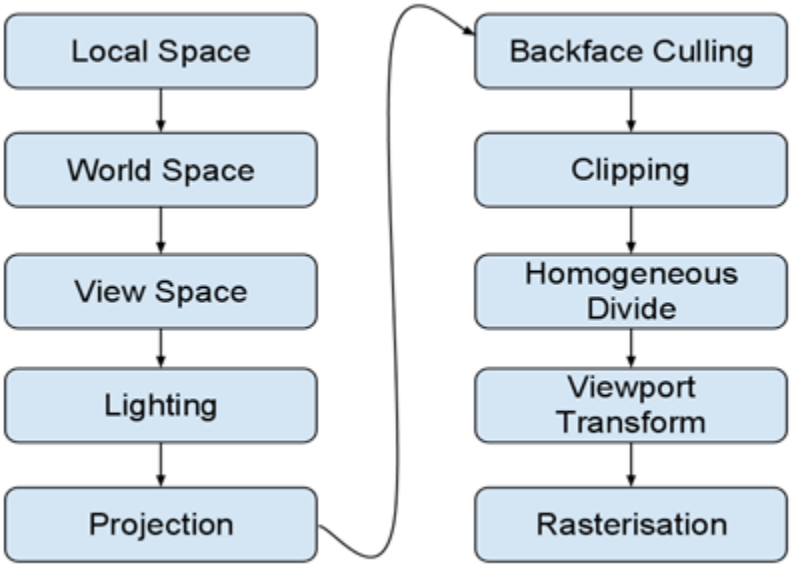


Figure 28: The 3D pipeline for visualization.

The application used the 3D pipeline which is based on the Luna's program architecture [24]. The update Scene () function and draw Scene () function from Luna is the main place that the application was used. The update Scene () function is used to update the positions of all the tensegrities and any interaction from the computer mouse and keyboard. The draw Scene () function is used to draw all of the tensegrity models in one time step. In the draw Scene () function, the shader and render function for each tensegrity were implemented. For making a video to demonstrate the result, the function which takes the screenshot of each frame was also implemented in the draw Scene () function. In the update Scene () function, all the control functions were implemented here, and all the positions of the tensegrities were updated. Updating the tensegrity has two steps. The first step is to execute the Update () function which uses the positions of all the tensegrities from the last frame to calculate the new position of the tensegrity. The value of the new position of each tensegrity is stored temporarily and the new position of each tensegrity is not updated in step 1. This is because each tensegrity needs to use the same value of all the tensegrity's positions from the last frame, which can guarantee the concurrency for real simulating. Then, step 2 uses the Update Position () function which will update the all the new positions of the tensegrities which can be used in the next frame. The procedure is shown below:

New position of tensegrity in each frame

There is a class called Tensegrity which encapsulates the function for the interaction between cells and the interaction with the fluid. The new position of the tensegrity can be calculated in two parts – the Rigid Body Dynamics part and the soft body deformation part. In the Rigid Body Dynamics part, each tensegrity is treated as a rigid body particle and the translation and rotation of the tensegrity are dealt with in this step. The forces acting on the 12 nodes of each tensegrity are calculated following the algorithm in chapter 3.4. The calculated forces were delivered to the soft body deformation function. In the soft body deformation part, with the 12 forces acting on the 12 nodes of the tensegrity, the new position after deformation could be calculated following the algorithm in chapter 3.3. The results combined with these two-part calculations becomes the final value of the tensegrities' positions for rendering in this frame.

Coordinate system conversion

The physics computing always uses the right-hand system (RHS), but the Microsoft DirectX SDK (2010) uses the left-hand system (LHS). Thus, for visualizing a correct image after the physics calculation, converting the coordinate system was necessary. The coordinate systems are shown below:

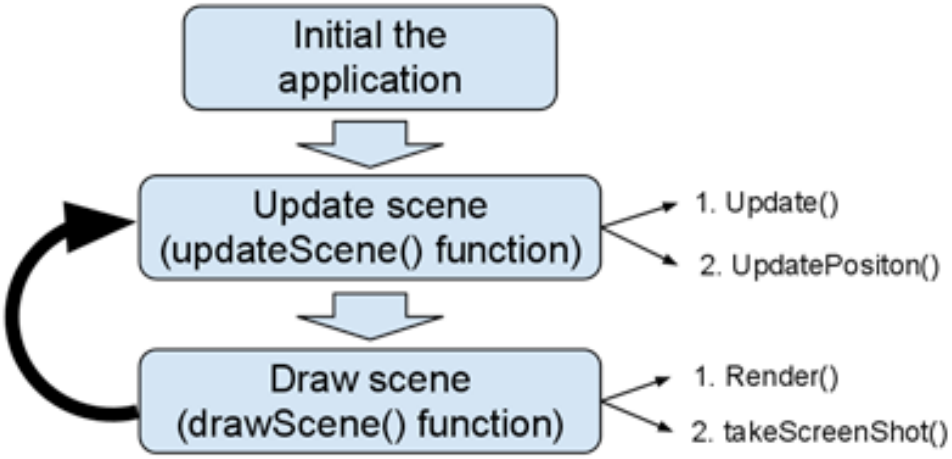


Figure 29: The structure of the application for visualization based no DirectX.

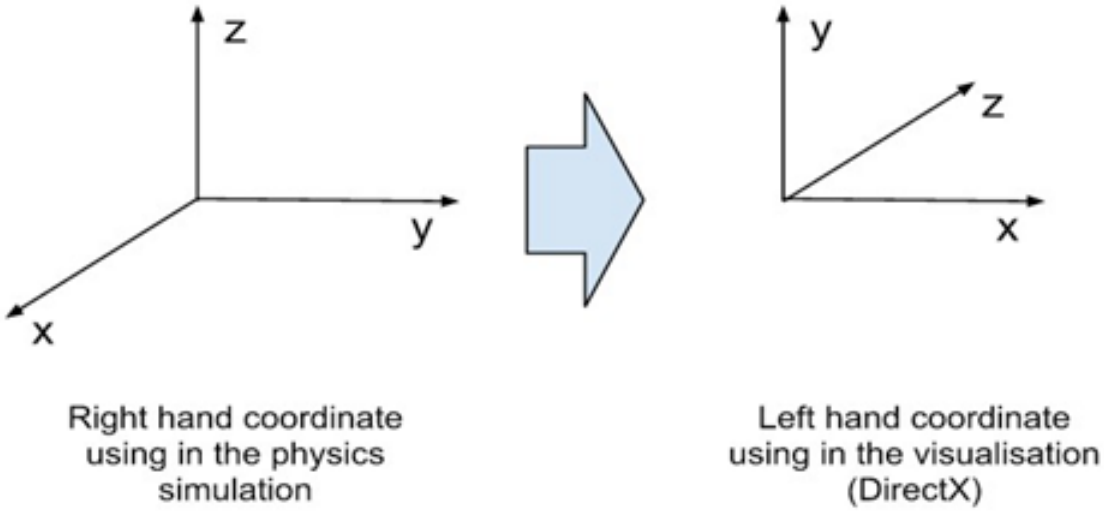


Figure 30: The right coordinate system used for physics simulation (Left) and the left-hand coordinate system used for visualization (Right).

This dissertation chose the first method (chapter 2.6.2 and chapter 3.6) to implement.

Vertex Col is one of the data structures used in the program to describe the information of vertex. It contains the position and the color of one vertex.

The code shown below is to convert the right-hand coordinate system to the left-hand coordinate system before the simulation data that generated in the simulation part of Case B used by the program of the visualization part of Case B.

```
struct VertexCol

{

    VertexCol():pos(0.0f, 0.0f, 0.0f),col(0x00000000){}

    VertexCol(float x, float y, float z, D3DCOLOR c):pos(x,y,z), col(c){}

    VertexCol(const D3DXVECTOR3& v, D3DCOLOR c):pos(v),col(c){}



    D3DXVECTOR3 pos;

    D3DCOLOR       col;

    static IDirect3DVertexDeclaration9* Decl;

};
```

Figure 31: One of the data structures of the vertex.

```
vDX[0] = VertexCol(vertexWorld[0].y, vertexWorld[0].z, -vertexWorld[0].x, WHITE);

vDX[1] = VertexCol(vertexWorld[1].y, vertexWorld[1].z, -vertexWorld[1].x, BLACK);

vDX[2] = VertexCol(vertexWorld[2].y, vertexWorld[2].z, -vertexWorld[2].x, RED);

vDX[3] = VertexCol(vertexWorld[3].y, vertexWorld[3].z, -vertexWorld[3].x, GREEN);

vDX[4] = VertexCol(vertexWorld[4].y, vertexWorld[4].z, -vertexWorld[4].x, BLUE);

vDX[5] = VertexCol(vertexWorld[5].y, vertexWorld[5].z, -vertexWorld[5].x, YELLOW);

vDX[6] = VertexCol(vertexWorld[6].y, vertexWorld[6].z, -vertexWorld[6].x, CYAN);

vDX[7] = VertexCol(vertexWorld[7].y, vertexWorld[7].z, -vertexWorld[7].x, MAGENTA);

vDX[8] = VertexCol(vertexWorld[8].y, vertexWorld[8].z, -vertexWorld[8].x, WHITE);

vDX[9] = VertexCol(vertexWorld[9].y, vertexWorld[9].z, -vertexWorld[9].x, BLACK);

vDX[10] = VertexCol(vertexWorld[10].y, vertexWorld[10].z, -vertexWorld[10].x, RED);

vDX[11] = VertexCol(vertexWorld[11].y, vertexWorld[11].z, -vertexWorld[11].x, GREEN);
```

Figure 32: Converting the physics simulation coordinate to the coordinate used for visualization.

Result and Discussing

This chapter will represent the visualized result from the simulation data and discuss the result based on the comparison of the different results with the analysis.

Result of Case A

Case A is the test application. It built the tensegrity model and tested the stability of the tensegrity model.

At first, the program set the internal energy was just from the elastic energy of cables. There are two tests which are performed when the condition (12) shown below was satisfied.

$$W_{internal} = W_{cable} \quad (87)$$

- Pull apart the tensegrity model from the top points (r_4 , r_8 and r_{12}) as the figure shows:

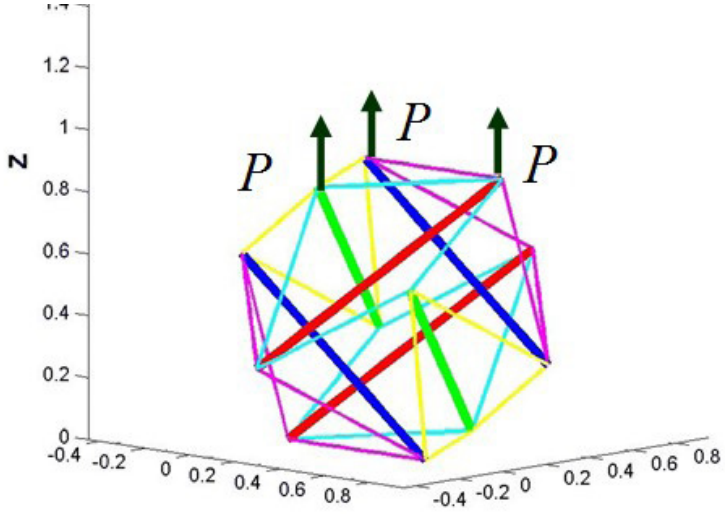


Figure 33: The tensegrity model applied the upward external force.

The external force P would increase with a predefined increment in each time step. The larger the external force was, the greater the shape of the tensegrity changed. Finally, the shape would be changed as the figure showing below:

The result demonstrated that the tensegrity model used in this project was stable with the energy just of the tension-supporting cables.

- Compress the tensegrity model on the top points (and) as the figure shows:

The external force P would increase as the same increment as the first test. The tensegrity model would crash quickly before it was compressed as a plate. This is because the internal energy stored in the tensegrity model was being lost faster and faster as the volume of the tensegrity became smaller and smaller.

Thus, for solving this problem, the volume reservation energy was added into the internal energy as form shown below:

$$W_{internal} = W_{cable} + W_{volume} \quad (88)$$

With the same external force P, the tensegrity model composed by the new internal energy could be compressed into a plate gradually. The final shape of the tensegrity would be changed as the figure shows below:

The cuboctahedron tensegrity introduced by Chen [17] could be more compressible without the volume reservation energy. But, compared with the COT (cuboctahedron tensegrity) structure of tensegrity which is made up of the 12 struts and 24 jointed nodes, the octahedron tensegrity is better because that the 6 struts tensegrity is much easier to simulate and the problem that happened in the first test of Case A

could be probably fixed by the volume reservation volume as the result in the second test. What is more, for simulating the activities of tissues, the number of the tensegrity should be as many as possible for visualization, so that the 6 struts tensegrity prefer to use in the application.

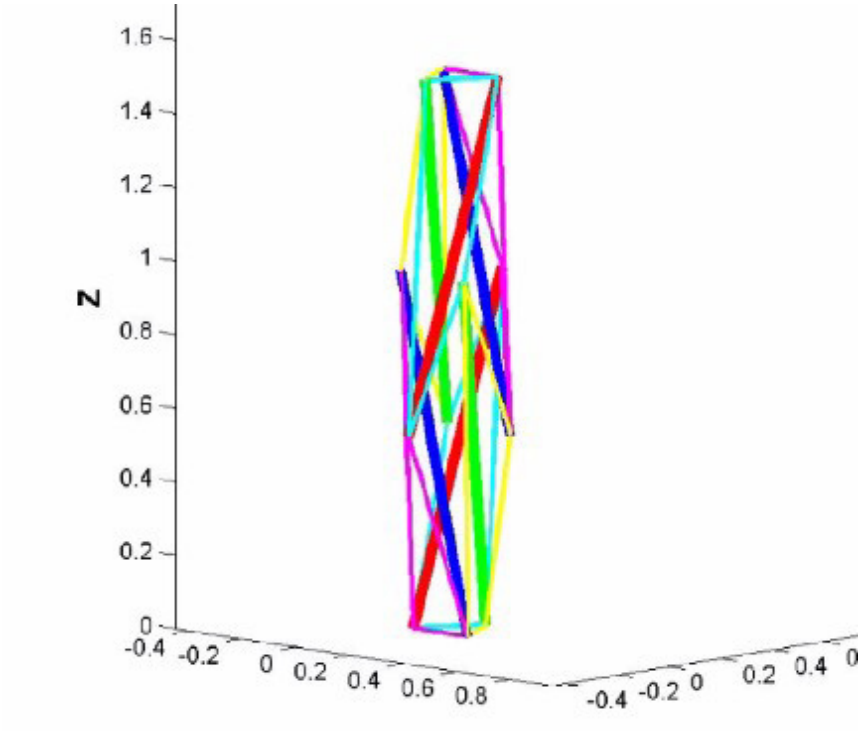


Figure 34: The deformed shape of the tensegrity model applied to the upward external force.

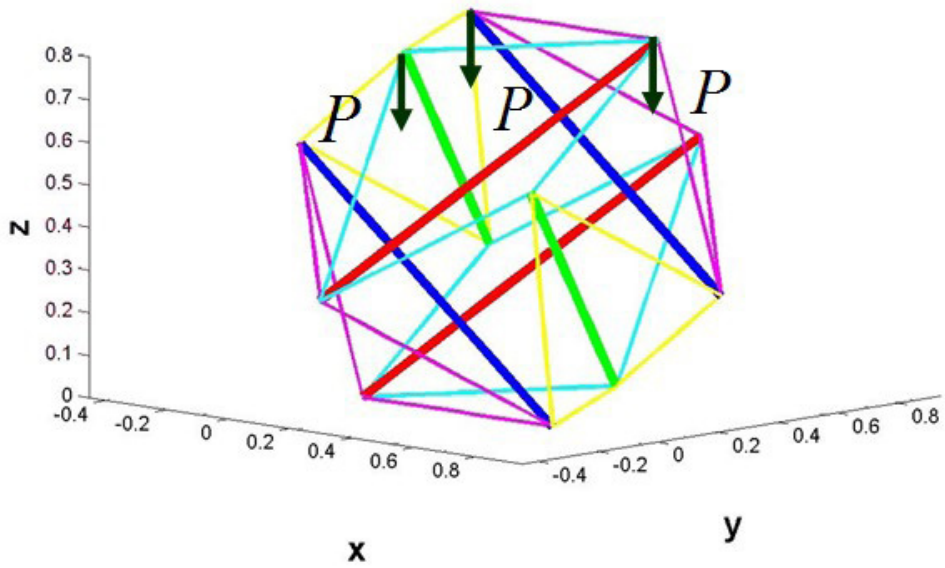


Figure 35: The tensegrity model applied the downward external force.

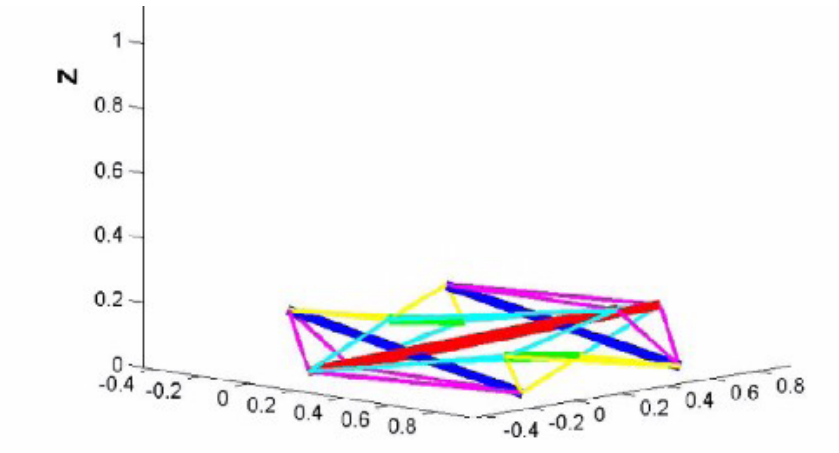


Figure 36: The deformed shape of the tensegrity model applied the downward external force.

Result of Case B

With the result for testing Case A, Case B should also get a appropriate visualization.

Case B simulated the activities of multiple cells. The number of cells (tensegrities) that the computer can simulate depended on the hardware performance, especially the capability of CPU. There are 4 samples which are implemented in Case B. The first three samples can be visualized in real time.

Sample 1 (2 Cells):

There are two simulated tensegrities rendering in real time.

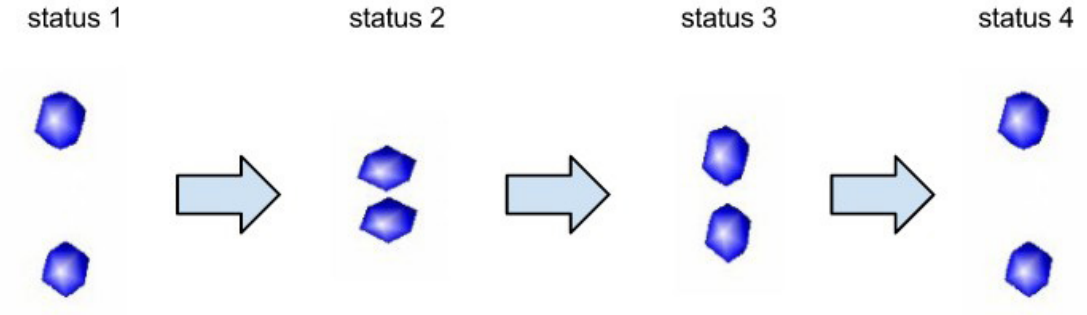


Figure 37: The procedure of the deformation of 2 tensegrities in sample 1.

At status 1, two tensegrities interact with each other, and the extra cellular matrix is assumed to be static. There is no other external loading acting on the two tensegrities. With the attraction force between them, they get closer and closer and when they get close enough, the upper tensegrity's bottom and the lower tensegrity's top will feel the repulsion from each other and the shapes of two tensegrities are deformed as shown in the status 2.

At status 2, the repulsion force acting on each tensegrity is larger than the attraction force, which means the two tensegrities will not get closer anymore. In other words, the tensegrity will remain stable during status 2. In the program of this sample, the external forces which pull apart the two tensegrities were added into the status 2. The external forces are acting on the top of the upper tensegrity and the bottom of the lower tensegrity. Two external forces are the same in value and the opposite in direction. The upper one is vertically upward, and the lower one is vertically downward.

Each of the forces is increased with a predefined increment in each time step. The shapes of the two tensegrities are changed little by little in status 3.

At the status 3, two tensegrities are pulled apart gradually and the cells have been stretched already. This is because the interaction forces are working well based on Lennard-Jones Potential. For each tensegrity, the side which is close to the other tensegrity is still attracted by the other tensegrity. The other side of the tensegrity is subjected to the external force which is big enough to deform the shape of the two tensegrities. At the status 4, the distance between two tensegrities is too far to allow them to attract each other. Thus, the tensegrities do not deform as much as the shape in status 3.

This sample can be also simulated the different initial positions of the two tensegrities. The different positions of the two tensegrities were placed, the different deformation of the tensegrity should be represented.

Sample 2 (3 Cells):

There are three simulated tensegrities rendering in real time as they deform as in the procedure shows below:

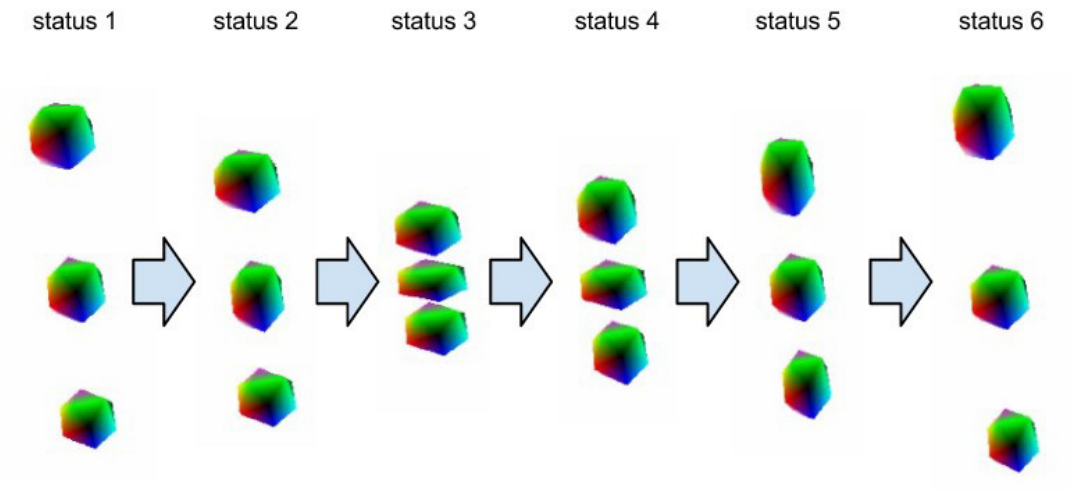


Figure 38: The procedure of the deformation of 3 tensegrities in sample 2.

At status 1, the three tensegrities are placed in three-dimensional space.

The separation distance between the upper tensegrity and the middle tensegrity, the middle tensegrity and the lower tensegrity is the same as the sample of 2 Cells. Naturally, the upper and the lower tensegrity will get close to the middle tensegrity caused by the attraction force and the forces acting on the middle tensegrity almost keep the balance, which means the middle tensegrity may not move as rigid body (Note: the shape of the middle tensegrity still deforms.).

As status 2 shows, the middle tensegrity is stretched by the attraction force from the upper tensegrity and the lower tensegrity.

At status 3, the three tensegrities have got much closer than status 2. For the middle tensegrity, the attraction forces from the upper tensegrity and the lower tensegrity becomes to the repulsion forces gradually. This is the reason that the stretched middle tensegrity becomes the compressed middle tensegrity in status 3. (Note: in status 3, the three tensegrities are not the stable and the three tensegrities would need to form a triangle in order to be stable.) At the status 4, the top of the upper tensegrity and the bottom of the lower tensegrity are applied the same external forces in the sample of 2 Cells which attempt to pull apart the tensegrities.

At the status 5, the external forces are big enough to pull apart the tensegrity. With the change of the position of the upper and lower tensegrity, the repulsion forces acting on the middle tensegrity become an attraction force and the shape of the middle tensegrity is stretched out again.

At status 6, with the movement of the upper and lower tensegrity, the separation distance of the three tensegrities is too far to make the interaction force between cells affect each other. The middle tensegrity returns to the original shape.

In this sample, the three tensegries were not in a stable state in any status. If the status 3 did not add the external force at that time, the three tensegrities should form a triangle as a stable state. This is because all of the forces acting on the three tensegrities are mutual. The tensegrities should form a geometry which should be symmetric.

Sample 3 (6 Cells):

There are 6 simulated tensegrities which are rendering in real time. The figure below shows the five statuses in the deformation of the 6 tensegrities.

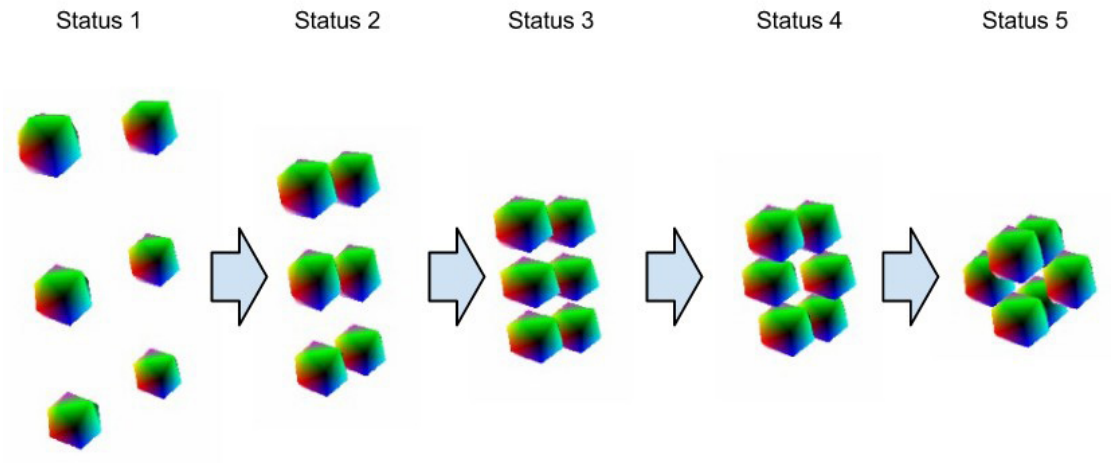


Figure 39: The procedure of the deformation of 6 tensegrities in sample 3.

At status 1, the six tensegrities are placed in three-dimensional space and the separation distance is the same as sample 2.

At status 2, the tensegrities are attracted by each other and get closer and closer. The middle two tensegrities are stretched by the attraction forces from the upper two tensegrities and the lower two tensegrities.

At status 3, the attraction forces acting on the middle two tensegrities become to the repulsion force. Thus, the shapes of the six tensegrities deformed as status 3 shows.

At the status 4, to reach the stable state, the six tensegrities continually move. The middle two tensegrities begin to move to the left and right of each side.

At the status 5, this is the stable state of the six tensegrities after interacting with each other. The six tensegrities have formed an octahedron. This is the same reason that the three tensegrities would form as a triangle if the sample 2 did not apply the external forces on the tensegrities at the status 4.

This sample is particular for testing the stable state of the tensegries. According to sample 2 (3 Cells), the tensegrities will form a symmetric shape. The 6 tensegrities form an octahedron demonstrated it again.

Sample 4 (27 Cells):

There are 27 simulated tensegrities in this sample. This sample tries to simulate stretching the tissues.

At status 1, the 27 tensegrities are placed as a cube with the same separation distance used in samples 1, 2 and 3. The tensegrities will attract each other and get closer and closer.

At status 2, the 27 tensegrities reached the stable state as a cube. The external forces which are the same as those used in sample 1 and 2 are acting on the upper 9 tensegrities and the lower 9 tensegrities.

At status 3, the 27 tensegrities are stretched as a tissue.

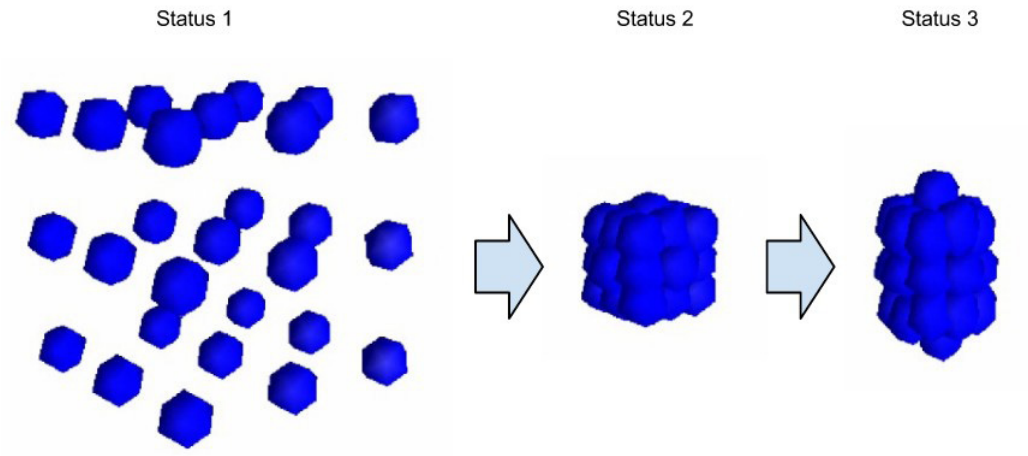


Figure 40: The procedure of the deformation of the 27 tensegrities in sample 4.

Sample 1 has 30 frames per second and sample 2 and sample 3 are above 10 frames per second. Compared with the first three samples, sample 4 just has one frame per second. It cannot be a real time application. Real time applications are wildly used in various fields such as communication techniques and computer games. It provides a real time environment to interact with users, which can easily adjust to the results generated from the application. If sample 4 (27 cells) or more cells need to be simulated in real time, parallel computing may be a good method (introduced in chapter 7.2).

For simulating the tissues with 27 tensegrities, another reason is that the center tensegrity of the 27 tensegrities can be acted upon by forces in the eight directions. This can be the most complicated condition to simulate one tensegrity. Therefore, the 27 tensegrities can be considered as a unit for simulation.

With the simulated tensegrity model, the other simulations should be also working. For example, in sample 4, if the external forces were compressed from both sides of the top and bottom of the tissues, the multilayered cells should be all squashed.

Conclusion

In chapter 1, compared with a rigid body particle, the soft body particle which can be deformed may be great to simulate the cells and tissues. The reason using the tensegrity as the soft body particle to simulate the living cells and living tissues was explained. There are also many techniques used to build the tensegrity model and simulate the interaction between them. The purpose of which is to make the behavior of the tensegrity model become more like that of a cell. It was from a suggestion from the supervisor that MATLAB and Mathematica were used to build the expressions and get the simple visual of the simulation data. The choice of C++ as the programming language and Microsoft DirectX SDK as the 3D rendering platform was for the flexibility and high efficiency to develop the application.

In chapter 2, all the important concepts which were used to implement the application were briefly introduced such as the tensegrity model for choosing the appropriate structure of the tensegrity, the Lennard-Jones Potential for the interaction between cells, the Stokes resistance for the interaction between cell and extra cellular matrix, and DirectX for visualizing the simulation data.

In chapter 3, the algorithms used to simulate the activities of the tensegrities and visualize the simulation were explained in detail.

In chapter 4, the procedures of how to implement the application were well illustrated and some main functions were explained by the code.

In chapter 5, the results of the application were represented. The methods that have been constructed in chapter 3 and implemented in chapter 4 were demonstrated here. With the demonstration of the visualization, the tensegrity model was used to simulate the living cells and living tissues successfully. On the other hand, the number of simulated tensegrities is not enough to represent more realistic living tissues. To simulate more and more tensegrities, the most direct way is to use the better hardware. But the hardware is too expensive. Thus, finding a method which can increase the speed of computing is the key to solving this problem. Nowadays, multi-core processors are wildly used in personal computers. Parallel computing becomes available to implement. Generally, the more cores the program uses the more efficient the application should be. Thus, implementing parallel computing may be a reasonable way to improve the efficiency of the simulation.

References

1. D E Ingber (2006) Cellular mechanotransduction: putting all the pieces together again, The FASEB journal: official publication of the Federation of American Societies for Experimental Biology 20(7): 811-827.
2. D E Ingber (2000) Opposing views on tensegrity as a structural framework for understanding cell mechanics, Journal of applied physiology 89(4): 1663-1670.
3. D E Ingber, Tensegrity I (2003) Cell structure and hierarchical systems biology, Journal of cell science 116(Pt 7): 1157-1173.
4. H Brenner (1963) The Stokes Resistance of an arbitrary particle. Chemical Engineering Science 18(1963): 1-25.
5. H Brenner (1964) The Stokes Resistance of an arbitrary particle-II An extension., Chemical Engineering Science 19(9): 599-629.
6. Microsoft, DirectX, 2010.
7. Matlab, MATLAB, 2011.
8. Mathematica, MATHEMATICA, 2011.
9. B Fuller, Tensegrity, Artnews Annual 4(1961): 112-127.
10. K Snelson, Snelson on the tensegrity invention, Int. J. Space Struct 11(1996): 43-48.
11. P Canadas, V M Laurent, C Oddou, D Isabey, S Wendling (2002) A cellular tensegrity model to analyse the structural viscoelasticity of the cytoskeleton. Journal of theoretical biology 218(2): 155-173.
12. M F Coughlin, D Stamenovic (1998) A tensegrity model of the cytoskeleton in spread and round cells, JOURNAL OF BIOMECHANICAL ENGINEERING-TRANSACTIONS OF THE ASME 120(6): 770-777.
13. D Stamenovic, M F Coughlin (2000) A quantitative model of cellular elasticity based on tensegrity, JOURNAL OF BIOMECHANICAL ENGINEERING-TRANSACTIONS OF THE ASME 122(1): 39-43.
14. C Sultan, D Stamenovi, D E Ingber (2004) A computational tensegrity model predicts dynamic rheological behaviors in living cells, Annals of biomedical engineering 32(4): 520-530.
15. K Y Volokh, O Vilnay, M Belsky (2000) Tensegrity architecture explains linear stiffening and predicts softening of living cells. Journal of biomechanics 33(12): 1543-1549.
16. N Wang, K Naruse, D Stamenovi, J J Fredberg, S M Mijailovich, et al, (2001) Mechanical behavior in living cells consistent with the tensegrity model, Proceedings of the National Academy of Sciences of the United States of America 98(14): 7765-7770.
17. T J Chen, C C Wu, M J Tang, J S Huang, F C Su (2010) Complexity of the Tensegrity Structure for Dynamic Energy and Force Distribution of Cytoskeleton during Cell Spreading, PLOS ONE 5(12): e14392.
18. P G Hajigeorgiou (2010) An extended Lennard-Jones potential energy function for diatomic molecules: Application to ground electronic states, Journal of Molecular Spectroscopy 263(1): 101-110.
19. P Bourke (2001) Converting between left- and right-hand coordinate systems.
20. G B Arfken, H J Weber Mathematical methods for physicists, Academic P. 1995.
21. MathPages (2011) Simplex Volumes and the Cayley-Menger Determinant.
22. C Hecker (1997) Physics, Part 4: The Third Dimension, Behind the screen pp. 15-22.
23. W H Press, Numerical Recipes: The Art of Scientific Computing, Cambridge University P.2007.
24. F D Luna (2006) Introduction to 3D game programming with DirectX 9.0c: a shader approach, Wordware Pub, Plano, Tex.

Simulation of Bacterial Cell Swimming Based on Lattice-Boltzmann Method and Agent-Based Model

Introduction

The first model, which is the real-time project, simulates the fluid flow and renders its lattice at the same time on a single NVIDIA GeForce GT 525M graphics card. The aim of this project is to render a cubic container filled with fluid and simulate the really physical fluid flow without other rigid objects in it. The size of the cube is 30*30*30. For the second model, the size of container is 104*54*54, and we added 100 bacterial cells into the fluid environment. In this model, we not only use LBM to simulate fluid flow, but also consider the interaction between fluid flow and elliptic bacterial cells. And the result of this model displays all the elliptic cells driven by fluid flow. In other words, all the elliptic rigid bodies follow the real physical fluid flow. And then, because of the complex computation in second model, we separated simulation and visualization into two steps. Firstly, the result of the simulation will be saved in .txt file separately. Secondly, visualizing both flow and bacterial cells based on Direct3D, all the data like velocity and position and other corresponding properties in terms of bacterial cells and fluid flow will be read from .txt file saved previously. Meanwhile, for the visualization in ParaView, the entire data files must follow Legacy VTK file format which is available in ParaView.

As said before, both of these models contain fluid flow. To simulate fluid flow, computational fluid dynamics method should be used. And here, both models are based in Lattice Boltzmann Method (LBM), which is one of the best choices to simulate fluid and becomes widespread use in gas or fluid flow. This method designed lots of rectangular lattices, and each lattice is labelled and will follow its special rule. Lattice-Boltzmann obeys the mass, momentum and energy conversation, and it is able to simulate fluid flow with real physics. In detail, LBM has two main processes, streaming and collision. The particles will stream from one lattice to another one according to their own velocity and direction recorded. When a collision happens, some of particle's properties should be updated through a rule. During the streaming and collision, most properties of particles have updated.

In the second model, we noticed before, the interactive effort is considered as another important part in this article. To calculate the resistance force on bacterial cells immersed in a continuous viscous fluid, Stokes resistance is the main law in this part. The total hydrodynamic force and torque are acting original position 0 of each particle. And the reaction on fluid around particle will be calculated as well. The Methodology chapter will discuss the details of this method.

Visualizing both models, we base on Microsoft DirectX SDK (Microsoft 10). In these models, lots of small particles have been seeded in fluid space in order to record fluid flow properties at corresponding position. Mainly, each particle should record instantaneous velocity in every frame. And all the particles could follow the fluid flow accurately in every frame. According to the distance and direction of each particle updated over frame, the direction and velocity could be displayed clearly. This method could record and display the flow's direction and velocity any time. Compared to the first demo, the second demo with bacterial cells also uses ParaView tool to draw streamline of fluid. Using streamline, it is clear to observe the shape of the fluid flow. But when the flow becomes stable, we cannot recognize its flow's direction and velocity. Therefore, two methods used to remedy each one's disadvantage and visualize the second model more perfectly and clearly.

The paper is organized as follows. After discussing the main aims of two projects and their related work in chapter 1, the computational model (Lattice-Boltzmann and Stokes Resistance) and Methodology about visualization will be described in chapter 2. In chapter 3, the implementation of computational model and visualization will be described in detail. In chapter 4, after displaying and stating the results of thesis projects, I will analyse and discuss them relating to the projects' main aim. And then, in chapter 5, the conclusion will discuss the contribution, advantages and disadvantages in these projects. Followed by this, the future work will be stated in chapter 6.

Project Aim and Relevance

This chapter will introduce two parts in terms of the project. First, it will describe the main efforts my thesis projects. In addition to this, these models support a general structure, which could help us extend the implementations in many domains, such as biology, geosciences, and computer games. Second, this chapter gets ready to demonstrate the related work in thesis projects, which embrace the developments and functions of all the methods implemented.

Aim of the projects

For the first project, it did a basic structure with Lattice-Boltzmann Method simulating fluid flow with real physics in a closed cubic space. Firstly, this demo is real-time, and we could observe how the fluid works with fixed accelerations. Secondly, according to the operational state of this basic model, we could make sure whether the simulation and visualization are both alright. This creates the basic and right fluid space in order to add other rigid bodies in it.

For the second project, the aim is to simulate cells with elliptic shape swimming in tube which is filled with fluid. And it offers a relatively realistic algorithm and model in biology. According to this model, we could simulate a large numbers of rigid body cells moving in blood vessel or other tissue fluid. And the interaction between rigid bodies and fluid flow will be displayed in this model.

Relevance about Lattice-Boltzmann Method

Fluid simulation is an extremely commonly used in computer graphics, which has the capability to generate the really physical animation such as water, explosion and other phenomena. Generally, there are several computational fluid dynamics methods including Eulerian grid-based methods, Smoothed Particle Hydrodynamics (SPH) method, and Lattice Boltzmann methods. For the current project, Lattice-Boltzmann Methods is the way implemented to simulate 3D fluid flow. This way has many advantages which will be discussed in detail later.

The Development of Lattice-Boltzmann Method: As we can notice, all the models' fluid simulations are based on Lattice Boltzmann Method (LBM). LBM is a class of computational fluid dynamics method for fluid simulation, and it is especially well suited to simulate flow around complex geometries [1- 3]. Over the last few years, the issue and research in terms of Lattice-Boltzmann method for fluid flow has received a lot of attention greatly improving and developing the method. In fact, LBM is developed from Lattice Gas Automata (LGA) and it provides us with a new method to research complex non-linear systems. In LGA, regular hexagon had been proposed as basic lattice model [4]. Because this model is highly symmetric, which suits the standard of Navier-Stokes equations (N-S). N-S equation describes the motion of fluid substances, and it applies Newton's second law for fluid motion. However, nevertheless, LGA still has many essential shortcomings. For instance, its distribution of particles is Fermi-Dirac distribution. Because of this, a lot of noise or errors will occur in LGA. In order to figure out these shortcomings, Boltzmann equation replacing LAG to simulate fluid has been proposed [5]. This way helps to solve the random noise problem. While, at present, LB-BGK has been proposed to solve the particle distribution more perfectly. In LB-BGK model, it uses $\frac{f_i - f_{eq}}{\tau}$ to be collision part. In this part f_{eq} represents equilibrium distribution function. In addition to this, during the transition from LGA to LBM, LBM removes the statistical noise using its ensemble average, density distribution function, instead of Boolean particle number. In the development of LBM, approximating the collision operator with the Bhatnagar-Gross-Krook (BGK) relaxation term is a critical simplification. Because of this, it makes the Lattice-Boltzmann Method more efficient and flexible to simulate fluid flow. Additionally, we know the BGK operator term refers to a collision operator used in the Boltzmann Equation and in the Lattice Boltzmann method. And this lattice BGK could make simulation more efficient. At present, there are many kinds of models that could be used in LBM in order to simulate fluid flow (see Figure 1).

During the development of Lattice-Boltzmann Method, it is a modern and relatively new technique for the simulation of fluid flow. Too many advantages of Lattice-Boltzmann method appear. Firstly, due to the simple arithmetic calculations, this does not only just help programmers design the model easily, but also it could give us an accurate solution to the complex partial differential equations. Secondly, it is very flexible to generate many complex conditions or boundaries which are able to satisfy designers' requirements. Thirdly, the distribution function depends on the last steps' properties, such as density and velocity. For this situation, this is very suitable for parallel programming. Due to these advantages discussed before, this method could help to develop fluid space quickly and accurately. Additionally, this method is a very powerful and promising method to simulate fluid flow in different domains, which are able to extend other feathers on currents project.

Implementation in different domains: As we said, Lattice-Boltzmann Method (LBM) is very suitable for fluid simulation. And then it still could create complex boundaries in order to satisfy many real-word conditions. For instance, fluid flow running around a porous square cylinder could be used by the Lattice-Boltzmann method to solve it (Babu and Narasimhan, 2010). But it does not stop here, more research using different boundary conditions has been proposed. The fluid flow moving around rotating circular cylinder also be solved by Lattice-Boltzmann Method [6].

And LBM is flexible to use in another domain. For example, according to LBM could solve the steady and unsteady flow problem [7]. It also has the capability to simulate shallow water [8]. The following figures are the results in terms of fluid flow (see figure. 2) and water wave (see figure. 3), which are based on Lattice-Boltzmann Method. Because of the flexibility of LBM, it could simulate fluid flow efficiently and realistically. Thus, we chose Lattice-Boltzmann Method as basic way to simulate fluid flow in these projects.

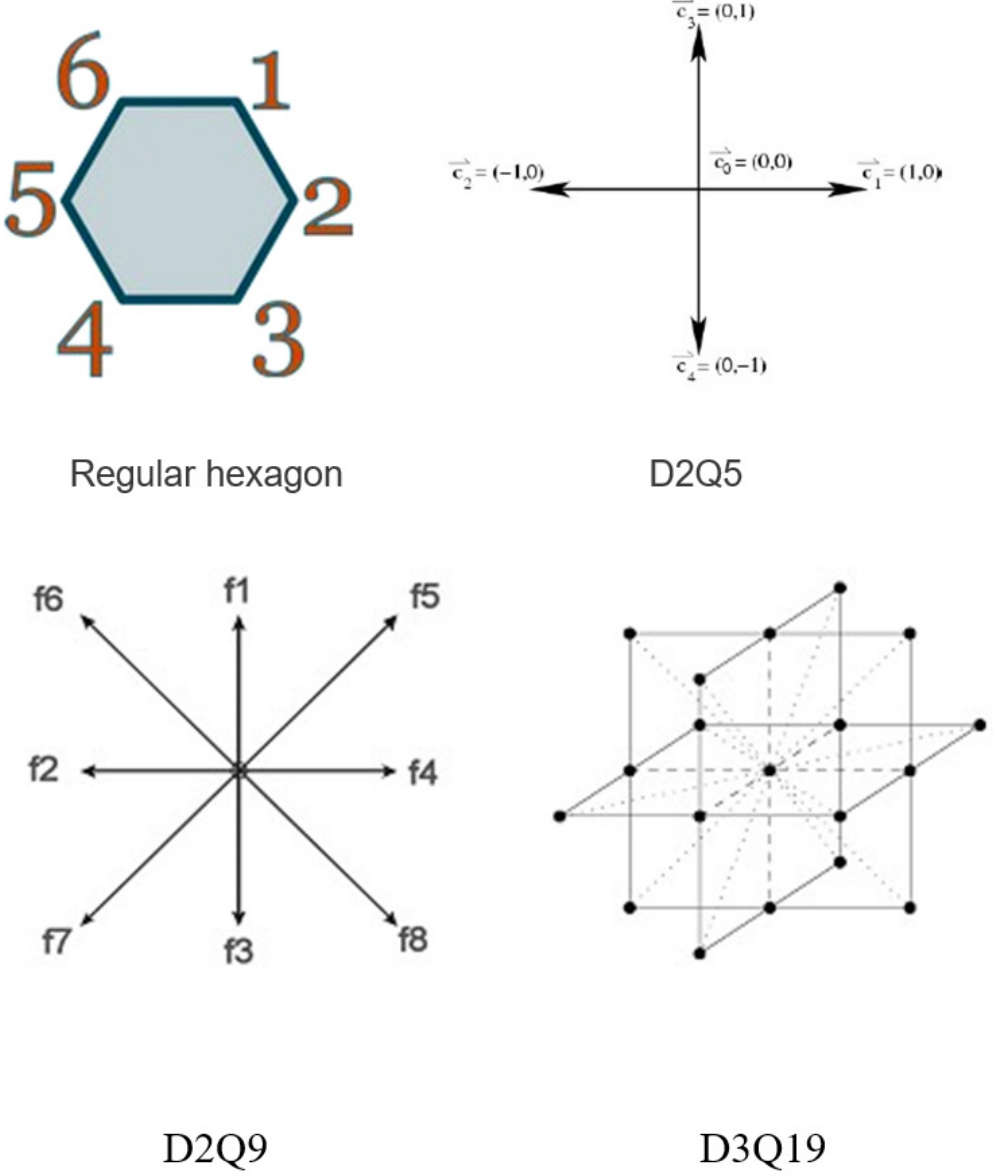


Figure 1: Four models in Lattice Boltzmann Method, DmQn models in common use represent m-dimensional and n directional velocities.

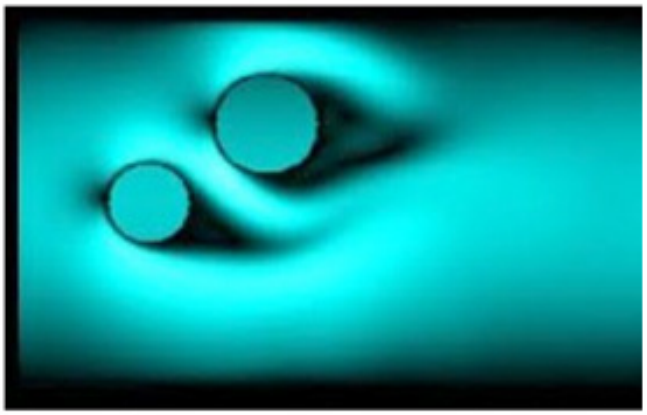


Figure 2(a): simulation of Flow Around Circle Cylinder.

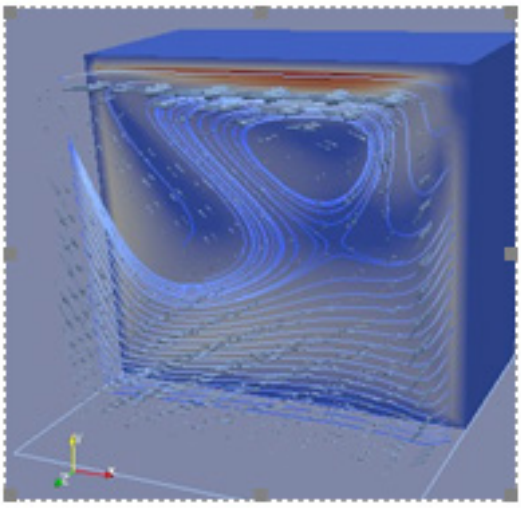


Figure 2(b): Thermal fluid simulation.

Figure 3: Water wave using Lattice-Boltzmann Method in 2D.

In computer games, there are lots of scenes from small to large scale such as droplets, swimming pool and ocean. Lattice-Boltzmann is one of the best choices implementing fluid flow in computer games. Otherwise, another popular method, smoothed particle hydrodynamics (SPH) are also able to simulate fluid dynamics excellently. But in this dissertation, the LBM is the way will be implemented and discussed. LBM could simulate them with correct physical animation. For computer games, if the human observers could not distinguish and judge whether the dynamics is correct. We could say the simulation about fluid is sufficient and successful. In previous research, the LBM has been used to simulate surface flow animation over terrain models in computer games [9]. Large-scale, deep-water simulations appropriate for oceans and lakes, which has been implemented based on LBM [10].

Components of Lattice-Boltzmann Method: LBM, On the one hand, could be considered as a simplified fictitious molecular dynamics model. What means by this is that all the space, time and velocities are discrete. On the other hand, Lattice Boltzmann Model includes two main processes, which are collision and streaming step separately. In other words, LBM models the fluid consisting of fictive particles, and such particles perform consecutive propagation and collision processes over a discrete lattice mesh. During a time, there is a particle move form current node to its neighbour node according to corresponding microscopic velocity. Generally, this process is called streaming part. Until more than one particle forms different directions reach the same nodes, all the particles velocities may be changed because of collision. Usually, this step is called collision part. In LBM, D3Q19, an efficient model in common use represents 3-dimentional and 19 directional velocities. However, nevertheless the D3Q19 has this advantage D3Q27 is what we want to make the fluid flow more realistic.

In basic LBM, to solve the issue of fluid, there are streaming, collision, viscosity, and boundary conditions as our focuses. Especially, Boundary Conditions are still debated and pop topic in LBM [2]. And then, for Boundary Conditions it offers several solutions, such as periodic boundaries, bounce-back boundaries, von Neumann boundaries, and pressure boundaries and so on. In this article, bounce-back boundaries have been chosen for this demo's fluid flow. The reason why we use bounce-back rule is that this model is thermal Lattice Boltzmann model and this method is relatively easy and plays a major role in simulating fluids characterised by complex geometries [2]. But somebody holds their opinion that bounce-back boundaries come in some variants [11] and sometimes it cannot work extremely perfectly [12]. However, nevertheless, this could get quite satisfactory results and it could work relatively efficiently. And, in addition to this bounce-back rule is quite elegant and surprisingly accurate and this rule is one of the most widely used in no-slip Boundary Condition [13]. At the earliest, the bounce-back rule was proposed by Ladd [14,15]. Ladd divided the nodes into solids nodes and fluid nodes. As their name shows, it is easy to understand that the fluid nodes are immersed in a fluid environment. On the contrary, the solid nodes hide in the interior of solid part. The bounce-back rule assumes that when the fluid particles move to and touch the solid nodes. These particles will be reflected into fluid node. And the details about how to work will be discussed in the next Methodology chapter.

Relevance about Stokes Resistance

The hydrodynamic force acting on a particle at small Reynolds number is of fundamental importance in a variety of engineering applications [16]. Previous research has characterized the stokes resistance of solid particles of arbitrary shape to translational and rotational motion in a fluid at rest at infinity[16,17]. This research gives a promising foundation to extend to uniform shearing flow.

For this dissertation, another main task is to calculate hydrodynamic force and torque acting on ellipsoid cells and interaction between bacterial cells and fluid. The important and main feature in the demo with ellipsoid cells (The dynamic ellipsoid model in static fluid is made by Guo Ni, one of members in our group) is resistance force, torque between fluid flow and elliptic cells. Here, we consider the cell moving into uniform shearing flow rather than static fluid. Therefore, shear force and torque should be considered. The stokes resistance of an arbitrary particle in uniform shear flow developed the hydrodynamic force and torque on a general body immersed in a flow [18]. Here, we shall not be concerned with the detailed structure of any of these velocity and pressure fields but only with the hydrodynamic forces and torques which they induce on the particles. But it does not stop here, according to Newton's third law, the torque effecting on fluid is equal to the torque on cells while direction is opposite. Followed by torque, the direction of couple force acting on fluid can be determined by using the right-hand grip rule. And the details of these methods will be discussed in the next chapter.

Relevance about Visualization

Fluid flow visualization makes the flow patterns visible in fluid dynamics. According to visualization, it is relatively easy and clear to obtain qualitative and quantitative information in term of flow. Generally, drawing streamline and particle tracer are the widely used and common methods visualize the patterns of fluid flow. Here, both methods have been utilized in ParaView Tool and Direct3D respectively. In addition to this, the matrix laboratory [19] is an extra step to judge the exactness of visualization in 2D. And the following words will describe the related work about visualization in detail:

Visualization in ParaView Tool: All the models have been visualized with Direct SDK. While, to visualize the fluid model with ellipsoid cells, the ParaView Tool has been used to draw the fluid streamline [20]. There are many reasons why I use ParaView in this project. Firstly, ParaView is an open source, and it is an available program for parallel and scientific visualization. Secondly, ParaView could analyze data using qualitative and quantitative techniques, and it has powerful capability to set up visualization quickly. Thirdly, ParaView could help to analyze an extremely large amount of data using distributed memory computing source. In the second project discussed before, we divided it into two main parties, simulation and visualization respectively. The reason why we are separating them is a large number of data is computed in simulation process. According to ParaView tool, it could draw streamline easily and quickly, but the data stored must follow special data format which the ParaView is able to read and write such as Legacy VTK files [21]. Briefly, using ParaView could visualize the streamline quickly and accurately. Additionally, the result of visualization using ParaView could compare the visualization using Direct3D.

Streamlines are defined as a set of curves which are instantaneously tangent to the velocities of the flow. From the definition before, streamlines are able to offer a snapshot of flow's shape of characteristics. Therefore, putting every snapshot together could create a video in order to make the change of characteristic more smoothly.

As we discussed before, if the streamline wants to be drawn successfully in ParaView, file format (Legacy format used here) must be followed. Even though Legacy format is not flexible than other format such as XML, but Legacy format is the simplest and legal one used in VTK. For the simple Legacy format, it embraces five basic parts:

- The first part is a header about the identifier and version of flies. Currently, the version number includes 1.0, 2.0, and 3.0. In this project, 2.0 has been used, and it is compatible with version 3.0. For this part, one single line head code is: "# vtk DataFile Version X" (excluding quotation marks). X means the version released by VTK.
- The second part is the title which includes a character string terminated by end-of-line character \n.
- The third part is file format. This part displays the type of file. They are ASCII or binary.
- The fourth part is the type of dataset structure. Such as

STRUCTURED_POINTS

STRUCTURED_GRID

UNSTRUCTURED_GRID

RECTILINEAR_GRID

POLYDATA

- The fifth part is one of the most important parts about the dataset attributes. The keywords are POINT_DATA or CELL_DATA.

The first three parts are fixed; while other remaining parts are flexible to choose the matching and suitable structure and then corresponding attributes will be determined. In this chapter, the STRUCTURED_POINTS is the data structure selected. For this structure, the dimensions n_x, n_y, n_z must be greater than or equal to 1. Additionally, the spacing s_x, s_y, s_z cannot be less than 0. The overview format about STRUCTURED_POINTS is displayed as follows:

DATASET STRUCTURED_POINTS

DIMENSIONS n_x n_y n_z

ORIGIN x y x

SPACING s_x s_y s_z

And the overview about the whole header in project is displayed as follows:

vtk DataFile Version 2.0

Volume example

ASCII

DATASET STRUCTURED_POINTS

DIMENSIONS 104 54 54

ASPECT_RATIO 1 1 1

ORIGIN 0 0 0

POINT_DATA 303264

VECTORS Velocity float

As mentioned before, the first three parts are mandatory and fixed. The following part is Point structure used. And then dimensions mean how many data in x, y, z components respectively. And all offsets in three dimensions are 1. The total data used in a set of data is 303264. Finally, every .vtk file will add the head before their velocity data in order to be read in ParaView successfully.

Visualization in DirectX: The visualization of both projects is based on Direct SDK, and it uses Frank D. Luna's basic shader structure as application frameworks [22]. Meanwhile, in these projects the basic particle system class [22] has been utilized to design custom particle system so that it could satisfy the requirement in Lattice-Boltzmann Model.

For these projects discussed above, utilizing particle system to simulate the fluid flow basing on Particle Image velocimetry (PIV) method in Direct3D. Using PIV method could help to measure the velocity of fluid accurately, even though negligible distortion of the fluid flow will occur sometime. As the name suggests, PIV records the position over time of small tracer particles introduced into the flow to extract the local fluid velocity [23]. And PIV method is a relatively new way to visualize flow dynamics. It is used to obtain instantaneous velocity measurements and other corresponding properties in fluid. In Particle Image velocimetry method, seeding tracer particles is always the critical component, because these particles are assumed to faithfully follow the flow dynamics according to the velocity they recorded. Importantly, the particles should be small enough so that response time of the particles to the motion of the fluid is relatively short to accurately follow flow. However, in Lattice-Boltzmann Method, there are small intervals between nodes. And if the particle is moving into this gap, it is difficult to obtain suitable instantaneous velocity. To solve this problem, bounding volumes help to set a perfect space around each node (see figure. 4 and figure. 5). In figure 4, the yellow cubic pace is bounding volume, surrounding black node. And the length of the cubic space is . According to this volume, each particle moving into this area will obtain an instantaneous velocity at the black node's position.

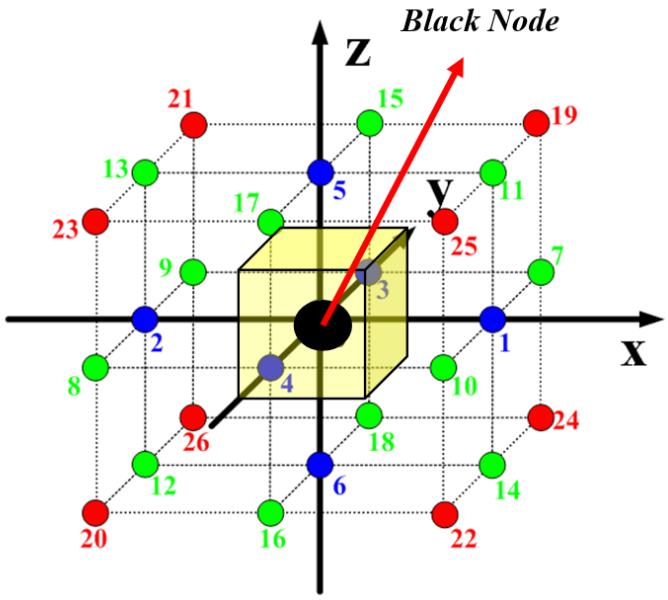


Figure 4: Three-dimensional bounding volume.

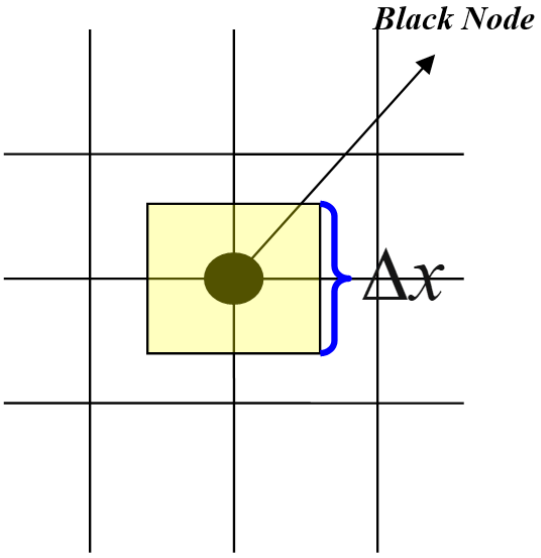


Figure 5: Two-dimensional bounding volume.

The particle system has been mentioned several times before. While, how to render particles in direct3D is still a question here. Generally, a particle is a very small object that is usually modelled as a point mathematically [22]. As we know, in direct3D, there are seven primitive types we could choose using IDirect3DDevice9::DrawPrimitive Method. To draw particle, the first way is to choose point primitive (D3DPT_POINTLIST in D3DPRIMITIVETYPE Enumeration). But this way has its disadvantages, partly because of the texture problem in point primitive, but principally because the size of point primitive is not flexible to change. However, in these projects, one of the requirements is to set a suitably small size of particle object which are able to satisfy measure the instantaneous velocity about fluid flow. Previously, in Direct3D 8.0, many programmers tried to use small billboards to replace small particle. In this way, particle objects solve the problem in the flexibility concerning size and texture, but each particle object has at least four vertex information saved in memory. If we design a particle system with a large number of particle objects, more memory will be used to store each particle's vertex variables, and this is not an efficient way to render large scenes. Point sprites, therefore, are one of the perfect ways to obtain a particle with variable size and even map entire texture to a particle. Importantly, for each particle object, we can just save one vertex rather than four in memory which are able to make it render efficiently in projects. In point sprites, the size of each point is determined by an application-specified size combined with a distance-based function computed by Direct3D (Microsoft). And the size of each particle is determined by PointScaleEnable in technique of shader. We set it as false, because the application-specified size used as screen-space size had already satisfied in these projects. And the size of each particle must be set between "MaxPointSize" and "MinPointSize" in "D3DCAPS9 structure". To render particle with texture, the parameter "PointSprinteEnable" should be set as TURE. If that is false, the coordinates of each vertex texture will be duplicated as each vertex. While, in these projects, TURE has been set and texture are able to be rasterized as a quadrilateral following four coordinates according one vertex position. For instance, the position of one particle is (x, y, z) and its size is s. simplify the expression, Particel(x, y, z, s). Therefore, the other four coordinates about vertex are (x-s/2, y-s/2, z), (x+s/2, y-s/2, z), (x-s/2, y+s/2, z), (x+s/2, y+s/2, z) respectively. The following illustrated diagram describes the method in terms of rendering clearly (see Figure 6):

Visualization in MATLAB: Even though the step about Visualization in MATLAB is important in these projects, it still has its worth to support new evidence which could prove the exactness of the method in visualization. In this part, cooperating with Dr JiuJiang ZHU, Dr Zhu complicated the thermal model using thermal Lattice-Boltzmann Method and offered the final data file for me to visualize in Direc3D and MATLAB.

MATLAB is a high-performance language for technical computing and visualization, in addition to this, it is supporting an easy-to-use environment to programme. The MATLAB are also able to help us render the streamline for the result of thermal fluid model.

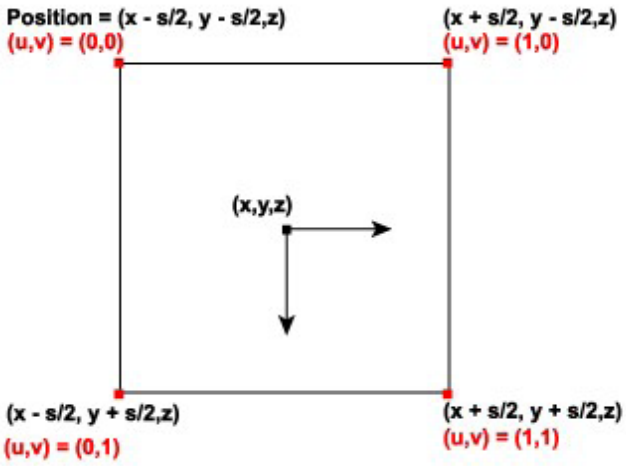


Figure 6: Texture coordinate.

The Computational Model and Methodology

This section will describe the mathematics in terms of LBM, stokes resistance. And the method of visualization will be the third component here.

Lattice Boltzmann Model (D3Q27)

First of all, these two demos use the same LBM to simulate fluid flow. The basic methods they used are almost the same. Therefore, the following words are going to discuss the basic method concerning lattice Boltzmann. As we said before, Lattice Boltzmann Method in 3D Model is common in hydrodynamic system. To make them more realistic, D3Q27 has been chosen in this chapter demos (see Figure 7). The Boltzmann equation of BGK model reads.

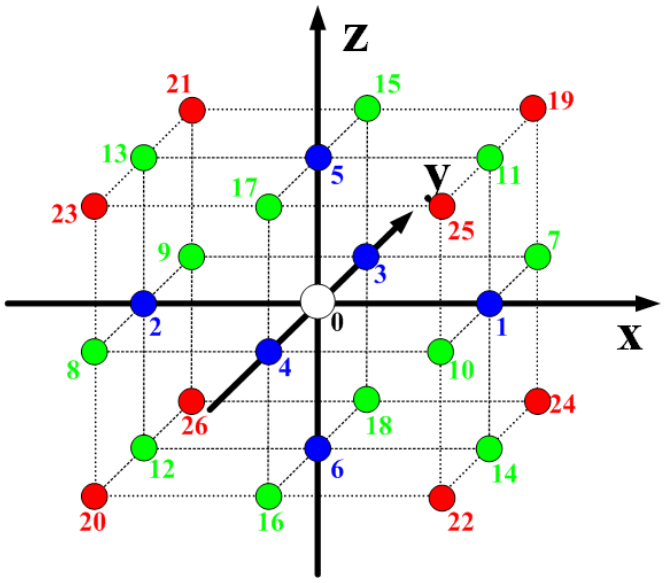


Figure 7: D3Q27 model.

$$\frac{\partial f(\xi,\mathbf{r},t)}{\partial t}+\xi.\frac{\partial f(\xi,\mathbf{r},t)}{\partial \mathbf{r}}+G\cdot\frac{\partial f(\xi,\mathbf{r},t)}{\partial \xi}\approx-\frac{f-f^{eq}}{\tau} \quad (1)$$

Where G is acceleration and λ is collision relaxation time.

In lattice, the numbers represent the microscopic velocities e_α respectively:

$$\mathbf{E}=\{e_\alpha\}\,(\alpha=0,1,\ldots,26) \quad (2)$$

In which

$$\mathbf{E}=\{\mathbf{e}_\alpha\}=\begin{pmatrix} 0&1&\bar{1}&0&0&0&0&1&\bar{1}&\bar{1}&1&1&\bar{1}&\bar{1}&1&0&0&0&0&1&\bar{1}&\bar{1}&1&\bar{1}&1&1&\bar{1}\\ 0&0&0&1&\bar{1}&0&0&1&\bar{1}&1&\bar{1}&0&0&0&0&1&\bar{1}&\bar{1}&1&1&\bar{1}&1&\bar{1}&\bar{1}&1&\bar{1}&1\\ 0&0&0&0&0&1&\bar{1}&0&0&0&0&1&\bar{1}&1&\bar{1}&1&\bar{1}&1&\bar{1}&1&\bar{1}&1&\bar{1}&1&\bar{1}&1&\bar{1} \end{pmatrix} \quad (3)$$

$\bar{1}$ stands for -1

These two equations (2) and (3) describe 27 directional velocities in this model. And the standard lattice $\xi_\alpha=\Delta x e_\alpha$, Δx is the length of space lattice. Introduce discrete variable, and Δt is time step. Then Boltzmann equation of BGK model equation may be rewritten as a discrete form as

$$f_\alpha(\mathbf{r}+\xi_\alpha\Delta t,t+\Delta t)-f_\alpha(r,t)=-\frac{f_\alpha(r,t)-f_\alpha^{eq}(r,t)}{\tilde{\tau}^*}+\int_t^{t+\Delta t}\frac{\mathbf{G}.(\xi_\alpha-\mathbf{u})}{\theta}f_\alpha^{eq}(\mathbf{r},t)\,dt \quad (4)$$

In equation (4) $f_\alpha(r,t)$ represents the frequencies (densities) at position r in special time t and

$$\tilde{\tau}^*=\tilde{\tau}+\frac{1}{2}=\frac{\lambda}{\Delta t}+\frac{1}{2} \quad (5)$$

In which and τ is collision relaxation time.

As we discussed before, LBM includes two processes, streaming and collision. Meantime, the simulation proceeds in an alternation between particle streaming and collision. And now these two parts could be described depending on equation (4) below:

Firstly: $f_\alpha(\mathbf{r}+\xi_\alpha\Delta t,t+\Delta t)-f_\alpha(r,t)$ is the streaming part. What mean by this is that the direction-specific densities will transform

to its nearest neighbour lattice points. Secondly: $\frac{f_\alpha(r,t)-f_\alpha^{eq}(r,t)}{\tilde{\tau}^*}+\int_t^{t+\Delta t}\frac{\mathbf{G}.(\xi_\alpha-\mathbf{u})}{\theta}f_\alpha^{eq}(\mathbf{r},t)\,dt$ is collision part.

After the collision happened, most of properties in term of fluid will be updated such as velocity and density around different lattice. And then the fluid particles tend to be a local equilibrium. Here, $f_\alpha^{eq}(\mathbf{r},t)$ in equation (4) represents equilibrium distribution function, its equation (6) has been displaying below:

$$f^{eq}\approx\rho W_0(\theta,\xi)\left[1+\frac{\xi.\mathbf{u}}{\theta}+\frac{(\xi.\mathbf{u})^2}{2\theta^2}-\frac{\mathbf{u}^2}{2\theta}\right]+O\left(\frac{\mathbf{u}^3}{\theta^3}\right) \quad (6)$$

In the equation (6) θ represents temperature and ρ represents fluid macroscopic density at corresponding position. To obtain the macroscopic density, the frequencies from 27 directions could be added together. See equation (7):

$$\rho=\sum_{a=0}^{26}f_a \quad (7)$$

In this part, we use dimensionless variables to alter the equation above. Because, using this way, we could create a general model without units.

Introduction to the following dimensionless variables:

$$\tilde{\mathbf{r}}=\frac{\mathbf{r}}{\Delta x},\ \tilde{t}=\frac{t}{\Delta t},\ \tilde{\mathbf{u}}=\frac{\mathbf{u}}{c},\ \tilde{\theta}=\frac{\theta}{c^2},\ \tilde{\rho}=\frac{\rho}{\rho_c},\ \tilde{f}_\alpha=\frac{f_\alpha}{\rho_c},\ \tilde{f}_\alpha^{eq}=\frac{f_\alpha^{eq}}{\rho_c},\ \tilde{\mathbf{G}}=\frac{\mathbf{G}\Delta t}{c} \tag{8}$$

In which ρ_c is reference density of system.

According to the equations (6) and (8), the Dimensionless discrete form could read

$$\tilde{f}_\alpha^{eq}\approx\tilde{\rho}W_\alpha\left[1+\frac{\mathbf{e}_\alpha\cdot\tilde{\mathbf{u}}}{\tilde{\theta}}+\frac{\left(\mathbf{e}_\alpha\cdot\tilde{\mathbf{u}}\right)^2}{2\tilde{\theta}^2}-\frac{\tilde{\mathbf{u}}^2}{2\tilde{\theta}}\right]+O\left(\tilde{\mathbf{u}}^3\right) \tag{9}$$

in which

$$\tilde{\theta}=\frac{1}{3} \tag{10}$$

$$w_\alpha=\begin{cases}8/27 & \alpha=0\\ 2/27 & \alpha=1,2\cdots,6\\ 1/54 & \alpha=7,8,\cdots 18\\ 1/216 & \alpha=19,8,\cdots 26\end{cases} \tag{11}$$

And then according to equations (4) and (8), the dimensionless equation could be described as:

$$\tilde{f}_\alpha\left(\tilde{\mathbf{r}}+\mathbf{e}_\alpha,\tilde{t}+1\right)-\tilde{f}_\alpha\left(\tilde{\mathbf{r}},\tilde{t}\right)=-\frac{\tilde{f}_\alpha\left(\tilde{\mathbf{r}},\tilde{t}\right)-\tilde{f}_\alpha^{eq}\left(\tilde{\mathbf{r}},\tilde{t}\right)}{\tilde{\tau}^*}+\int_{\tilde{t}}^{\tilde{t}+1}\frac{\tilde{\mathbf{G}}\cdot\left(\mathbf{e}_\alpha-\tilde{\mathbf{u}}\right)}{\tilde{\theta}}\tilde{f}_\alpha^{eq}\left(\tilde{\mathbf{r}},\tilde{t}\right)d\tilde{t} \tag{12}$$

Here we used rectangular method to approximate $\int_{\tilde{t}}^{\tilde{t}+1}\frac{\tilde{\mathbf{G}}\cdot\left(\mathbf{e}_\alpha-\tilde{\mathbf{u}}\right)}{\tilde{\theta}}\tilde{f}_\alpha^{eq}\left(\tilde{\mathbf{r}},\tilde{t}\right)d\tilde{t}$ part. In mathematics, the rectangular method computes an approximation to a definite integral. In this method, it tries to find the area of collection of rectangles whose heights are determined by values of the function.

$$\int_a^b f(x)dx\approx h\sum_{n=0}^{N-1}f(x_n) \tag{13}$$

In which $h=(b-a)/N$ and $x_n=a+nh$

Generally, if N gets larger, this approximation gets more accurate. In this issue, we set N=1.

Therefore, $\int_{\tilde{t}}^{\tilde{t}+1}\frac{\tilde{\mathbf{G}}\cdot\left(\mathbf{e}_\alpha-\tilde{\mathbf{u}}\right)}{\tilde{\theta}}\tilde{f}_\alpha^{eq}\left(\tilde{\mathbf{r}},\tilde{t}\right)d\tilde{t}\approx\frac{\tilde{\mathbf{G}}\cdot\left(\mathbf{e}_\alpha-\tilde{\mathbf{u}}\right)}{\tilde{\theta}}\tilde{f}_\alpha^{eq}\left(\tilde{\mathbf{r}},\tilde{t}\right)$

Finally, equation (12) could read:

$$\tilde{f}_\alpha\left(\tilde{\mathbf{r}}+\mathbf{e}_\alpha,\tilde{t}+1\right)-\tilde{f}_\alpha\left(\tilde{\mathbf{r}},\tilde{t}\right)=-\frac{\tilde{f}_\alpha\left(\tilde{\mathbf{r}},\tilde{t}\right)-\tilde{f}_\alpha^{eq}\left(\tilde{\mathbf{r}},\tilde{t}\right)}{\tilde{\tau}^*}+\frac{\tilde{\mathbf{G}}\cdot\left(\mathbf{e}_\alpha-\tilde{\mathbf{u}}\right)}{\tilde{\theta}}\tilde{f}_\alpha^{eq}\left(\tilde{\mathbf{r}},\tilde{t}\right) \quad (14)$$

Initialization about the frequencies on each lattice, we set the initial densities around each area as 1. For the unit lattice (see figure 7), each node on unit lattice has its own label as 0 to 26. This means the original node 0 has 27 directions. And then according to every direction, we could get the corresponding weights

(w_α) form equation (11). At last, to get the final 27 directional frequencies about original node 0, densities multiply each weight weights (w_α) respectively. For instance, if the direction of the frequency is 5, the result of frequency at this direction is 1*(2/27). But it does not stop here, another one we initialize is to set each velocity on every node as 0.

Bounce-back Boundary Condition in Lattice-Boltzmann Model: As mentioned before, in Lattice-Boltzmann Model there are two types of nodes. They are fluid nodes and solid nodes respectively. When the particles in fluid space are going to move from fluid nodes to fluid nodes, the "bounce-back" or reflection will happen at the time. And the reflection occurs at the middle plane of the link between fluid and solid nodes. In addition to this, the densities will be temporarily stored inside the solid parts, and they are going to re-emerge at the next time step. The following figure display the whole processes in term of bounce-back boundary condition (see figure 8):

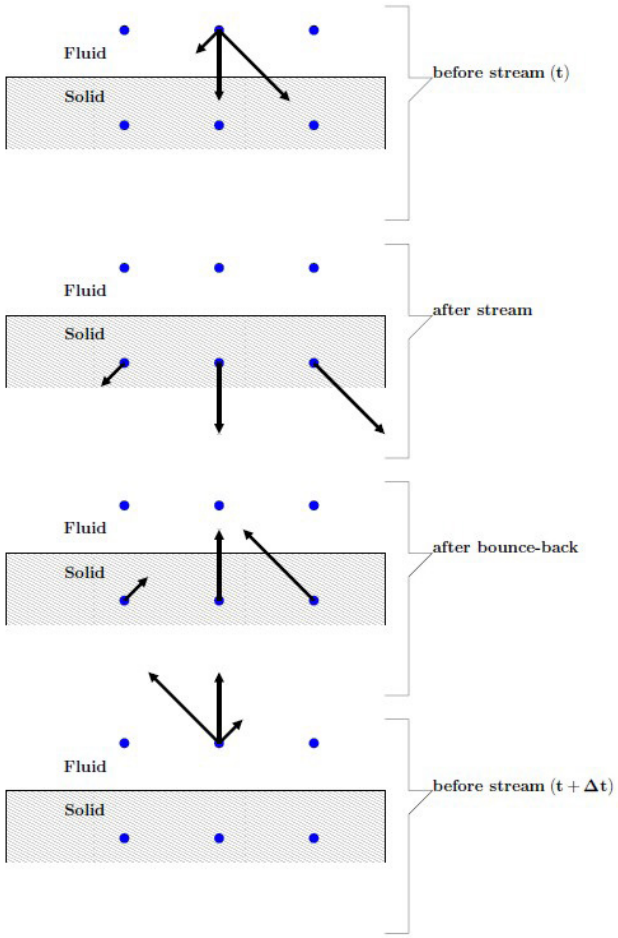


Figure 8: No-slip boundary condition (bounce-back boundary).

Figure 8 illustrates the mid-plane bounce-back algorithm in no-slip condition. And the arrows mean the direction-specific densities. And the following words will demonstrate how that working in detail is. At the first step, before streaming, the particles around mid-plane are trying to move from fluid nodes toward solid nodes. At the second step, after streaming, the particles look like moving into solid nodes. While the truth about this step is that the densities have been saved in solid nodes temporarily. At the third step, after bounce-back, all the densities have been preserved in solid nodes previously getting ready to re-emerge in fluid nodes. At the last step, we found that the densities have already moved back to fluid nodes. And image these particles meeting boundaries, they will begin to implement from the first to the fourth step again. Additionally, we could observe the feature in figure 8. After the reflection or "bounce-back" happened, the values of densities are equal, while their directions are opposite. If these processes use code to implement them, we assume the shortest arrow as density $f_{ijk}[1]$ in the first and the second step in figure 8. Meantime, at the third and the fourth step, we assume the shortest arrow as density $f_{ijk}[2]$. And (i, j, k) represent position of each node.

// general bounce-back boundary condition code:

Tem = f_{ijk} [1]; f_{ijk} [1] = f_{ijk} [2]; f_{ijk} [2] = Tem;

Stokes resistance of ellipsoid particle

Brenner, H does a lot of research on stoke resistance. And in this issue, the main part used is about the stoke resistance of ellipsoid particle. His previous contributions have dealt with the resistance on rigid particle of arbitrary shape immersing in a fluid at rest at infinity [24]. While, in a uniform shearing flow, more force and torque should be considered. Fortunately, this part about shear force and torque acting on arbitrary particle has been researched [24]. Because in this demo we use ellipsoid particle and uniform shearing flow, the contribution supported by Brenner, Howard could be utilized at the part. Form Brenner, Howard article, the form in term of the total hydrodynamic force and torque on particle written generally below:

$$\begin{aligned}\mathbf{F} &= -\mu[\mathbf{K}\cdot(\mathbf{U}_0-\mathbf{u}_0)+\mathbf{C}_0^{\dagger}\cdot(\boldsymbol{\omega}-\boldsymbol{\omega}_f)+\Phi_0:\mathbf{S}]\\ \mathbf{T}_0 &= -\mu[\mathbf{C}_0\cdot(\mathbf{U}_0-\mathbf{u}_0)+\Omega_0\cdot(\boldsymbol{\omega}-\boldsymbol{\omega}_f)+\boldsymbol{\tau}_0:\mathbf{S}]\end{aligned} \tag{15}$$

In equation (15),

$$\begin{aligned}\mathbf{F} &= \mathbf{F}_0^{'}+\mathbf{F}_0^{''}\\ \mathbf{T}_0 &= \mathbf{T}_0^{'}+\mathbf{T}_0^{''}\end{aligned} \tag{16}$$

In equation (16)

$$\begin{aligned}\mathbf{F}_0^{''} &= -\mu[\mathbf{K}\cdot(\mathbf{U}_0-\mathbf{u}_0)+\mathbf{C}_0^{\dagger}\cdot(\boldsymbol{\omega}-\boldsymbol{\omega}_f)]\\ \mathbf{T}_0^{''} &= -\mu[\mathbf{C}_0\cdot(\mathbf{U}_0-\mathbf{u}_0)+\Omega_0\cdot(\boldsymbol{\omega}-\boldsymbol{\omega}_f)]\end{aligned} \tag{17}$$

The two equations (18) have been proved by Brenner, Howard [16,17]. Followed by this, the following parts in equation is the shear force and torque. The shear force and torque are both the main parts in this project, which extend equation (17) for the uniform shearing flow.

$$\begin{aligned}\mathbf{F}_0^{'} &= -\mu\Phi_0:\mathbf{S}\\ \mathbf{T}_0^{'} &= -\mu\boldsymbol{\tau}_0:\mathbf{S}\end{aligned} \tag{18}$$

the force **F** must be independent of location of 0. **K** is the translation dyadic. And Φ_0 represents the shear-force triadic (tensor). $\mathbf{C}_0$ is the coupling dyadic at original 0. $\mathbf{C}_0^{\dagger}$ has a superscript on it, and this property of the transpose operator is defined as follows:

$$^{\dagger}(\mathbf{abcd}\ldots)=\mathbf{bacd}\ldots$$

$$(\ldots\mathbf{efgh})^{\dagger}=\ldots\mathbf{efhg}$$

In which **a, b, c** are vectors.

And $\mathbf{U}_0$ and $\mathbf{u}_0$ represent the particles' velocity and fluid flow velocity respectively. $\dot{\mathbf{u}}$ and $\dot{\mathbf{u}}_f$ represent the particles' angular velocity and fluid angular velocity respectively. Later the details about getting fluid angular velocity will be discussed.

In equation describes the total torque on the particle. $\mathbf{C}_0$ is the coupling dyadic at original 0 and Ω_0 is the rotation dyadic at 0. Additionally, τ_0 represents the shear-torque triadic (tensor) at 0. S is a 3x3 matrix in this equation displaying follow:

$$\mathbf{S}_{ij} = \frac{1}{2}\left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i}\right)(i, j = 0, 1, 2) \tag{19}$$

The issue studied is about ellipsoid, and this shape has been described by Brenner, Howard [18]. In his article, we could get the formulas concerning the translation dyadic **K**, the rotation dyadic Ω_0 below:

$$\mathbf{K} = 16\pi\left(\mathbf{i}_1 \otimes \mathbf{i}_1 \frac{1}{\chi + a_1^2\alpha_1} + \mathbf{i}_2 \otimes \mathbf{i}_2 \frac{1}{\chi + a_2^2\alpha_2} + \mathbf{i}_3 \otimes \mathbf{i}_3 \frac{1}{\chi + a_3^2\alpha_3}\right) \tag{20}$$

$$\mathbf{\Omega}_0 = \frac{16\pi}{3}\left(\mathbf{i}_1 \otimes \mathbf{i}_1 \frac{a_2^2 + a_3^2}{a_2^2\alpha_2 + a_3^2\alpha_3} + \mathbf{i}_2 \otimes \mathbf{i}_2 \frac{a_1^2 + a_3^2}{a_1^2\alpha_1 + a_3^2\alpha_3} + \mathbf{i}_3 \otimes \mathbf{i}_3 \frac{a_2^2 + a_1^2}{a_2^2\alpha_2 + a_1^2\alpha_1}\right) \tag{21}$$

In which, $\mathbf{i}_1$, $\mathbf{i}_2$, $\mathbf{i}_3$ means the right-handed triad of mutually perpendicular unit vectors, and $\mathbf{a}_1$, $\mathbf{a}_2$, $\mathbf{a}_3$ are the semi-axes of elliptic cell. And in equation (20) and (21)

$$\chi = \int_0^\infty \frac{d\lambda}{\Delta(\lambda)} \tag{22}$$

$$\alpha_i = \int_0^\infty \frac{d\lambda}{(a_i^2 + \lambda)\Delta(\lambda)} \quad (i = 1, 2, 3) \tag{23}$$

In equation (23) and (23) λ is dummy variable of integration and

$$\Delta(\lambda) = [(a_1^2 + \lambda)(a_2^2 + \lambda)(a_3^2 + \lambda)]^{\frac{1}{2}} \tag{24}$$

Because elliptic shape is considered at this project, therefore:

$$\begin{cases} \Phi_0 = 0 \\ \mathbf{C}_0 = 0 \end{cases} \tag{25}$$

$$\begin{aligned} \tau_0 = \frac{8\pi}{3}\Big[&\frac{a_3^2 - a_2^2}{a_3^2\alpha_3 + a_2^2\alpha_2}(\mathbf{i}_1 \otimes \mathbf{i}_2 \otimes \mathbf{i}_3 + \mathbf{i}_1 \otimes \mathbf{i}_3 \otimes \mathbf{i}_2) + \\ &\frac{a_1^2 - a_3^2}{a_1^2\alpha_1 + a_3^2\alpha_3}(\mathbf{i}_2 \otimes \mathbf{i}_3 \otimes \mathbf{i}_1 + \mathbf{i}_2 \otimes \mathbf{i}_1 \otimes \mathbf{i}_3) + \\ &\frac{a_2^2 - a_1^2}{a_2^2\alpha_2 + a_1^2\alpha_1}(\mathbf{i}_3 \otimes \mathbf{i}_1 \otimes \mathbf{i}_2 + \mathbf{i}_3 \otimes \mathbf{i}_2 \otimes \mathbf{i}_1)\Big] \end{aligned} \tag{26}$$

According to equation (15), it simplifies the total force and torque acting on ellipsoid particle as follows:

$$\mathbf{F} = -\mu \cdot \mathbf{K} \cdot \left(\mathbf{U}_0 - \mathbf{u}_0 \right) \tag{27}$$

$$\mathbf{T}_0 = -\mu [\mathbf{\Omega}_0 \cdot (\boldsymbol{\omega} - \boldsymbol{\omega}_f) + \boldsymbol{\tau}_0 : \mathbf{S}] \tag{28}$$

In equation (28), the properties of the operator (:) are defined as follows:

$$\mathbf{A} : \mathbf{B} = \mathbf{A}_{ij} \mathbf{B}_{ji} = \sum_{i=0}^{n} \sum_{j=0}^{m} \mathbf{A}_{ij} \mathbf{B}_{ji} \tag{29}$$

Remembering the LBM discussed in the last part, it has used non-dimensional formulas and variables. Therefore, to make sure the variables in Stokes Resistance on ellipsoid and that in LBM are non-dimensional, we need to transfer all the variables to be dimensionless. The following variables are dimensionless

$$\tilde{\mathbf{U}}_0 = \frac{\mathbf{U}_0 . \Delta t}{\Delta x} \, , \, \tilde{a}_i = \frac{a_i}{\Delta x} (i = 1, 2, 3) \, , \, \tilde{\chi} = \chi . \Delta x \, , \, \tilde{\alpha}_k = \alpha_k . \Delta x^3 \, , \, \tilde{\boldsymbol{\omega}} = \boldsymbol{\omega} . \Delta t \, , \, \tilde{\boldsymbol{\tau}}_0 = \frac{\boldsymbol{\tau}_0}{\Delta x^3} \, , \, \tilde{\mathbf{k}} = \frac{\mathbf{k}}{\Delta x} \, ,$$

(30)

$$\tilde{\mathbf{\Omega}}_0 = \frac{\mathbf{\Omega}_0}{\Delta x^3} \, , \, \tilde{\mathbf{F}} = \frac{\mathbf{F} . \Delta t^2}{\rho_c . \Delta x^4} , \tilde{\mathbf{T}} = \frac{\mathbf{T} . \Delta t^2}{\rho_c . \Delta x^5} \, , \, \tilde{\mu} = \frac{\mu}{\rho_c . \Delta x^2}$$

According to (30), the (26) and the (20) and the (21)could be described separately as follows:

$$\begin{aligned} \tilde{\boldsymbol{\tau}}_0 \cdot \Delta x^3 = \frac{8\pi}{3} [& \frac{(\tilde{a}_3 \cdot \Delta x)^2 - (\tilde{a}_2 \cdot \Delta x)^2}{(\tilde{a}_3 \cdot \Delta x)^2 \frac{\tilde{\alpha}_3}{\Delta x^3} + (\tilde{a}_2 \cdot \Delta x)^2 \frac{\tilde{\alpha}_2}{\Delta x^3}} (\mathbf{i}_1 \otimes \mathbf{i}_2 \otimes \mathbf{i}_3 + \mathbf{i}_1 \otimes \mathbf{i}_3 \otimes \mathbf{i}_2) + \\ & \frac{(\tilde{a}_1 \cdot \Delta x)^2 - (\tilde{a}_3 \cdot \Delta x)^2}{(\tilde{a}_1 \cdot \Delta x)^2 \frac{\tilde{\alpha}_1}{\Delta x^3} + (\tilde{a}_3 \cdot \Delta x)^2 \frac{\tilde{\alpha}_3}{\Delta x^3}} (\mathbf{i}_2 \otimes \mathbf{i}_3 \otimes \mathbf{i}_1 + \mathbf{i}_2 \otimes \mathbf{i}_1 \otimes \mathbf{i}_3) + \\ & \frac{(\tilde{a}_2 \cdot \Delta x)^2 - (\tilde{a}_1 \cdot \Delta x)^2}{(\tilde{a}_2 \cdot \Delta x)^2 \frac{\tilde{\alpha}_2}{\Delta x^3} + (\tilde{a}_1 \cdot \Delta x)^2 \frac{\tilde{\alpha}_1}{\Delta x^3}} (\mathbf{i}_3 \otimes \mathbf{i}_1 \otimes \mathbf{i}_2 + \mathbf{i}_3 \otimes \mathbf{i}_2 \otimes \mathbf{i}_1)] \end{aligned} \tag{31}$$

$$\tilde{\mathbf{K}} \cdot \Delta x = 16\pi \left(\mathbf{i}_1 \otimes \mathbf{i}_1 \frac{1}{\frac{\tilde{\chi}}{\Delta x} + (\tilde{a}_1 \cdot \Delta x)^2 \frac{\tilde{\alpha}_1}{\Delta x^3}} + \mathbf{i}_2 \otimes \mathbf{i}_2 \frac{1}{\frac{\tilde{\chi}}{\Delta x} + (\tilde{a}_2 \cdot \Delta x)^2 \frac{\tilde{\alpha}_2}{\Delta x^3}} + \mathbf{i}_3 \otimes \mathbf{i}_3 \frac{1}{\frac{\tilde{\chi}}{\Delta x} + (\tilde{a}_3 \cdot \Delta x)^2 \frac{\tilde{\alpha}_3}{\Delta x^3}} \right) \tag{32}$$

$$\tilde{\Omega}_0 \cdot \Delta x^3 = \frac{16\pi}{3}\left(\mathbf{i}_1 \otimes \mathbf{i}_1 \frac{(\tilde{a}_2\Delta x)_z + (\tilde{a}_3\Delta x)^2}{(\tilde{a}_2\Delta x)^2 \frac{\tilde{\alpha}_2}{\Delta x^3} + (\tilde{a}_3\Delta x)^2 \frac{\tilde{\alpha}_3}{\Delta x^3}} + \right.$$
$$\mathbf{i}_2 \otimes \mathbf{i}_2 \frac{(\tilde{a}_1\Delta x)_z + (\tilde{a}_3\Delta x)^2}{(\tilde{a}_1\Delta x)^2 \frac{\tilde{\alpha}_1}{\Delta x^3} + (\tilde{a}_3\Delta x)^2 \frac{\tilde{\alpha}_3}{\Delta x^3}} + \qquad (33)$$
$$\left.\mathbf{i}_3 \otimes \mathbf{i}_3 \frac{(\tilde{a}_2\Delta x)_z + (\tilde{a}_1\Delta x)^2}{(\tilde{a}_2\Delta x)^2 \frac{\tilde{\alpha}_2}{\Delta x^3} + (\tilde{a}_1\Delta x)^2 \frac{\tilde{\alpha}_1}{\Delta x^3}}\right)$$

And then we could get the force and torque according to Dimensionless quantity:

$$\frac{\tilde{\mathbf{F}} \cdot \rho_c \cdot \Delta x^4}{\Delta t^2} = -\frac{\tilde{\mu} \cdot \rho_c \cdot \Delta x^2}{\Delta t} \cdot \tilde{\mathbf{k}} \cdot \Delta x\left(\frac{\tilde{\mathbf{U}}_0 \cdot \Delta x}{\Delta t} - \frac{\tilde{\mathbf{u}}_0 \cdot \Delta x}{\Delta t}\right) \qquad (34)$$

$$\frac{\tilde{\mathbf{T}} \cdot \rho_c \cdot \Delta x^5}{\Delta t^2} = -\frac{\tilde{\mu} \cdot \rho_c \cdot \Delta x^2}{\Delta t} \cdot \left[\tilde{\Omega}\Delta x^3\left(\frac{\tilde{\omega}}{\Delta t} - \frac{\tilde{\omega}_f}{\Delta t}\right) + (\tilde{\tau}_0 \cdot \Delta x^3) : \frac{\tilde{\mathbf{S}}}{\Delta t}\right] \qquad (35)$$

Simplify (34)and (35) as follows:

$$\tilde{\mathbf{F}} = -\tilde{\mu} \cdot \tilde{\mathbf{K}} \cdot \left(\tilde{\mathbf{U}}_0 - \tilde{\mathbf{u}}_0\right) \qquad (36)$$

$$\tilde{\mathbf{T}}_0 = -\tilde{\mu}[\tilde{\Omega}_0 \cdot (\tilde{\omega} - \tilde{\omega}_f) + \tilde{\tau}_0 : \tilde{\mathbf{S}}] \qquad (37)$$

In equation (37), we not only need the angular velocity about ellipsoid cells, but also the angular velocity about fluid is required. Angular velocity of fluid has been mentioned previously. And the following part will discuss how to calculate angular velocity about fluid.

The angular velocity tensor is skew symmetric matrix; its elements could be present as follows:

$$W_{ij} = \frac{1}{2}\left(P_{ij} - P_{ji}\right) \qquad (38)$$

$$W_{ij} = -\varepsilon_{ijk}\omega_k \qquad (39)$$

In equation (38) $P_{ij} = \frac{\partial v_i}{\partial x_j}$ and in equation (39) ε_{ijk} is Levi-Civita symbol. The Levi-Civita symbol, also called the permutation symbol or antisymmetric symbol, is a mathematical symbol used in particular in tensor calculus. It is named after the Italian mathematician and physicist Tullio Levi-Civita. In three dimensions, the Levi-Civita symbol is defined as follows:

$$\varepsilon_{ijk} = \begin{cases} +1 \text{ if } (i,j,k) = (1,2,3) \text{ or } (2,3,1) \text{ or } (3,1,2) \\ -1 \text{ if } (i,j,k) = (1,3,2) \text{ or } (3,2,1) \text{ or } (2,1,3) \\ 0 \text{ if } i = j \text{ or } j = k \text{ or } k = i \end{cases} \qquad (40)$$

Therefore, according to equations (39) and (40), the tensor structure displayed as follows:

$$\mathbf{W}=\begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix} \quad (41)$$

(41)

As it is a skew symmetric matrix, it has a Hodge dual vector which is precisely the previous angular velocity vector **ù**

$$\boldsymbol{\omega}=\left(\omega_1,\omega_2,\omega_3\right) \quad (42)$$

(42)

Because the component of angular velocity expresses below

$$\omega_m=-\frac{1}{2}\varepsilon_{ijm}W_{ij}\left(i,j,m=1,2,3\right) \quad (43)$$

According to (40)(43), the components of angular velocity could express following:

$$\begin{cases} \omega_1=-\frac{1}{2}\varepsilon_{111}W_{11}--\frac{1}{2}\varepsilon_{121}W_{12}-\frac{1}{2}\varepsilon_{131}W_{13}-\frac{1}{2}\varepsilon_{211}W_{21} \\ -\frac{1}{2}\varepsilon_{221}W_{22}-\frac{1}{2}\varepsilon_{231}W_{23}-\frac{1}{2}\varepsilon_{311}W_{31}-\frac{1}{2}\varepsilon_{321}W_{32}-\frac{1}{2}\varepsilon_{331}W_{33} \\ \omega_2=-\frac{1}{2}\varepsilon_{112}W_{11}-\frac{1}{2}\varepsilon_{122}W_{12}-\frac{1}{2}\varepsilon_{132}W_{13}-\frac{1}{2}\varepsilon_{212}W_{21} \\ -\frac{1}{2}\varepsilon_{222}W_{22}-\frac{1}{2}\varepsilon_{232}W_{23}-\frac{1}{2}\varepsilon_{312}W_{31}-\frac{1}{2}\varepsilon_{322}W_{32}-\frac{1}{2}\varepsilon_{332}W_{33} \\ \omega_3=-\frac{1}{2}\varepsilon_{113}W_{11}--\frac{1}{2}\varepsilon_{123}W_{12}-\frac{1}{2}\varepsilon_{133}W_{13}-\frac{1}{2}\varepsilon_{213}W_{21} \\ -\frac{1}{2}\varepsilon_{223}W_{22}-\frac{1}{2}\varepsilon_{233}W_{23}-\frac{1}{2}\varepsilon_{313}W_{31}-\frac{1}{2}\varepsilon_{323}W_{32}-\frac{1}{2}\varepsilon_{333}W_{33} \end{cases} \quad (44)$$

Simplify (44)

$$\begin{cases} \omega_1=-W_{23} \\ \omega_2=-W_{13} \\ \omega_3=-W_{12} \end{cases} \quad (45)$$

And, according to these equations (38) (41) and (45)the components of angular velocity express below:

$$\begin{cases} \omega_1 = \frac{1}{2}(\frac{\partial v_3}{\partial x_2} - \frac{\partial v_2}{\partial x_3}) \\ \omega_2 = \frac{1}{2}(\frac{\partial v_3}{\partial x_1} - \frac{\partial v_1}{\partial x_3}) \\ \omega_3 = \frac{1}{2}(\frac{\partial v_2}{\partial x_1} - \frac{\partial v_1}{\partial x_2}) \end{cases} \quad (46)$$

Finally, we complete the force, torque acting on ellipsoid cells according to stoke resistance. Followed by this, the interaction between ellipsoid and fluid will be discussed. According to Newton's third law, the mutual forces of action and reaction between two bodies are equal, opposite and collinear. In this study, the torque acting on ellipsoid cells and torque acting on fluid are equal but opposite. And according to this, the torque ($\mathbf{T}_{R_f}$) effecting on fluid could be obtained. After this, a force couple could be calculated following the method as follows (see Figure. 9).

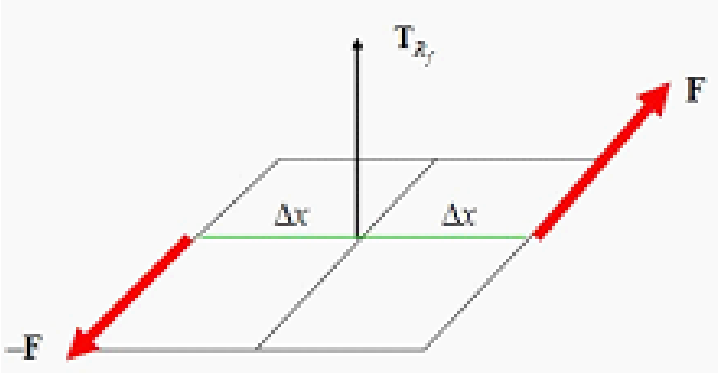


Figure 9: Couple force.

In (figure 9), the force acting on fluid could be computed with following formula:

$$\mathbf{F} = \frac{\mathbf{T}_{R_f}}{2 \cdot \Delta x} \quad (47)$$

According to non-dimensional way:

$$\tilde{\mathbf{F}} = \frac{\mathbf{F} \cdot \Delta t^2}{\rho_c \cdot \Delta x^4}, \ \tilde{\mathbf{T}}_{R_f} = \frac{\mathbf{T}_{R_f} \cdot \Delta t^2}{\rho_c \cdot \Delta x^5}$$

The equation (47) about force acting on fluid could be express as follows:

$$\frac{\tilde{\mathbf{F}} \cdot \rho_c \cdot \Delta x^4}{\Delta t^2} = \frac{\frac{\tilde{\mathbf{T}}_{R_f} \cdot \rho_c \cdot \Delta x^5}{\Delta t^2}}{2 \cdot \Delta x} \quad (48)$$

Simplify equation (48):

$$\tilde{\mathbf{F}} = \frac{\tilde{\mathbf{T}}_{R_f}}{2} \tag{49}$$

As we know, the torque acting on fluid is vector. Therefore, there are three components calculating three couple forces. All the fluid area around the ellipsoid will be driven by these couple forces.

Implementation

This chapter's main purpose is to describe the main structure of simulation and visualization. Additionally, this chapter will display the processes of simulation in term of Lattice-Boltzmann Method.

Simulation

For the first project, I implemented only Lattice-Boltzmann model (D3Q27) to simulate fluid flow on a single NVIDIA GeForce GT 525M graphics card. For this animation, 30*30*30grid has been simulated and visualized together at 24 frames per second. And the entire fluid flow want to be visible, PIV method has been used. Therefore, for this part, a large number of particles have been seeded randomly within a fixed container.

For the second project, I implemented both the Lattice-Boltzmann Model and Stokes Resistance in c++. They can be executed simultaneously and store the main properties in terms of fluid flow and ellipsoid cell. For this simulation, 104*54*54 grid with 100 ellipsoid cells has been simulated.

The following graph displays the whole processes of simulation in terms of Lattice-Boltzmann model and ellipsoid model (see Figure 10).

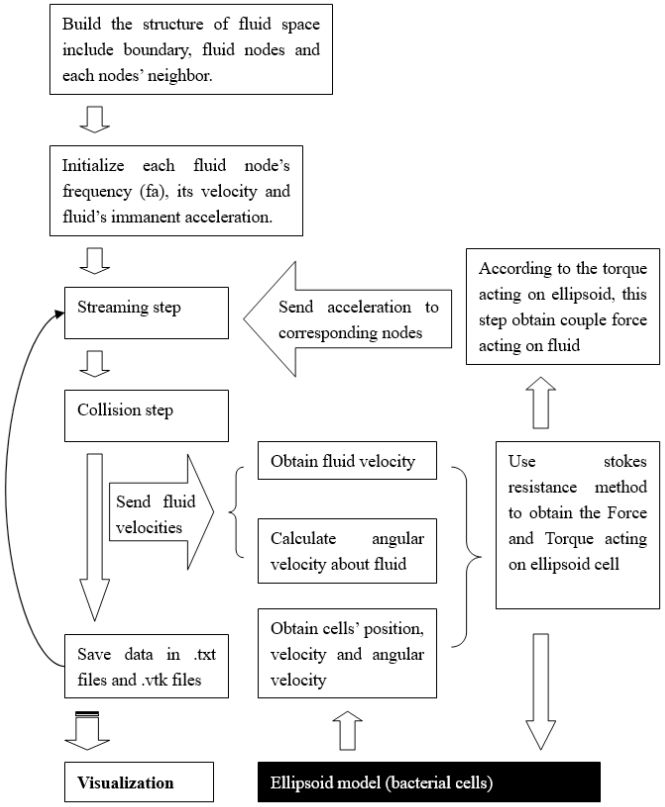


Figure 10: A Simplified Simulation Pipeline.

From the graph, it is clear to display the steps of simulation.

- Firstly, all the properties of fluid space has been built and initialized, such as node's position, initial density and velocity. After this, ellipsoid model will be initialized and it will obtain new properties about ellipsoid cells, like cells' position, velocity and angular velocity (The dynamic ellipsoid model (black part in figure.10) in static fluid is designed by Guo Ni, one of members in our group).

```
//began to setup lattice and initialization
void SoilStructure(    )
{
.....
.......

for(i=0;i<nx;i++) {
      for(j=0;j<ny;j++){
            for(k=0;k<nz;k++){
                  is_obstacle = ((31<=i && i<=40 && 0<j && j<54 && 0<k && k<=31) ||(62<=i &&
i<=72 && 0<j && j<54 && 22<=k && k<54));
                  if(2<=j && j<=insideNy    && 2<=k && k<=insideNz && (!is_obstacle))
                  {
                     node[i][j][k] = nn;
                     nn = nn + 1;
                  }
                  else
                  {
                     node[i][j][k] = -2;
                  }
            }
      }
 }   //*end to set other nodes as positive integer except for boundaries;
.............
.................
}
```

In this function above, it builds the boundaries, including the container and obstacles. The variables nx, ny, nz are the size of length, width and height respectively. Therefore, there are nx*ny*nz nodes in lattice. But, to distinguish the fluid nodes and solid nodes, all the solid nodes set as -2 (*node[i][j][k] = -2*;), while other fluid nodes set as positive number following the order list, 0, 1, 2…10, and this number could be set as ID number for each fluid node. Finally, the container and obstacle have been set which is benefitted to set each fluid node's neighbor. In this project, because of the 27 directional velocities, every node has 27 neighbor nodes including itself.

The initialization at the second step focuses on the fluid velocities and density on each node. In this project, the fluid velocities around each node also need to be initialized as 0 except for the density and frequency of each node. The density around each node is set as 1, and the initial frequency of each node is equal to the result of density multiplying weight at corresponding direction. In project, the code is a loop in order to obtain the frequency of each node at 27 directions:

```
void initLBM()
{
.....
.............
   for(int i=0;i<nn;i++)
   {
    den[i]= 1.0;
    ......
    ........
      // itinalize to f0 ... f18
    for(int j=0; j<27; j++)
    {
      f[j][i]=den[i]*wt[j]; //get the frequency of each nodes;
    }
    for(int j=0; j<3; j++)
    {
      u[j][i] = 0.0;    //initialize each nodes' speed in x, y and z direction
    }
   }
.............
.....................
}
```

All the variables about density, frequency and velocity will be initialized in *initLBM()*

function. In which, *nn* represents the number of fluid nodes.

- Secondly, the fixed acceleration will be set at proper position (up left of container) in order to drive the whole fluid space. When the fluid begins to move, the stream will happen at the same time. Followed by streaming processes, the collision will occur between every particle and the properties will be updated concerning fluid densities and velocities. To implement the collision in Lattice-Boltzmann, the following code will describe it:

```
void lbm_ns_3d(float tao, int tmax)
{
    ..........................................
    ........................
    ....
  for(int mm =0; mm<27; mm++)
  {
    float eu = (ec[mm][0]*u[0][i]+ec[mm][1]*u[1][i]+ec[mm][2]*u[2][i]);    //get e dot u
    //calculate feq0~feq26
    feq[mm] =    den[i] * wt[mm] * ( 1 + eu/c[i] +(eu/c[i])*(eu/c[i])/2.0 - uu/c[i]/2.0 );
    //calculate collision form 0~26 direction
    fc[mm][i] = tao2*f[mm][i] + ( tao1 +    tao3*accldotemu[mm]/c[i]    )*feq[mm];
  }
 //=======================end to collision================
    ........
  for(i=0;i<nn;i++)
  {
    for(int mm=0; mm<27; mm++)
    {
      mm][i] = n[ opp[ mm] ][i]>0    ?      fc[mm][ n[ opp[ mm] ][i] ]    : fc[ opp[ mm] ][i];
    }
  }
}
```

The main purpose of this part is to calculate the frequency of each fluid node in 27 directions after collision and complicate the equilibrium distribution function (9) As discussed before, the final equation (14) includes two parts. The right of the equal sign represents collision, and the remaining part is steaming. For code about collision part,

*fc[mm][i] = tao2*f[mm][i] + (tao1 + tao3*accldotemu[mm]/c[i])*feq[mm];*

In which, *fc[mm][i]* means the *i* node frequency at *mm* direction after collision. And *tao1=1/tao; tao2=1-tao1; tao3=1; tao=2 tao* represents $\tilde{\tau}^*$ in equation *(14)*. *f[mm][i]* represents the *i* node's frequency at last time at *mm* direction. *accldotemu[mm]* represents acceleration dot product the result of microscopic velocity minus macroscopic velocity ***u*** at mm direction. *feq[mm]* represents the equilibrium *fa*.

To obtain the equilibrium frequency at 27 directions, the following code tests the equilibrium equation and comes true this equation (9).

*Feq[mm] = den[i] * wt[mm] * (1 + eu/c[i] +(eu/c[i])*(eu/c[i])/2.0 – uu/c[i]/2.0);*

*den[i] * wt[mm]* means the fluid density at *i* node times weights at corresponding direction. And *eu* represents microscopic velocity dot product last macroscopic velocity at *i* node. *uu* means last macroscopic velocity dot product itself. Finally, when the last density and macroscopic velocities have been obtained, it could calculate the equilibrium frequency. Followed by this, when the frequency on each fluid node have obtain last time, it is easy to compute the current frequency on each fluid node after collision. When the collision among fluid finished, the frequency of each node will be updated according to streaming and bounce-back condition. After the collision in code, the following loop is the code to come true streaming and bounce-back condition. In which, *opp[mm]* means the opposite direction in a unit lattice (see figure.7). For instance, direction 3's opposite direction is 4, expressing *opp[3]=4.* And *n[][i]* means the neighbor node around i node. To explain this code clearly, the 2D illustration will be used below.

For instance, if we try to obtain the frequency of black node at seventh direction, we find blue node (opposite directional node) is solid node. Therefore *n[opp[mm]][i]<0*, and *f[mm][i]= fc[opp[mm]][i]*, which means the frequency of black node at seventh direction is equal to the collision frequency of black node at eighth direction. This is bounce-back boundary condition discussed before.

If we want to obtain the frequency of black node at eighth direction, the opposite direction is fluid node (*n[opp[mm]][i]>0*). Therefore, *f[mm][i]=fc[mm][n[opp[mm]][i]]*, which means the frequency of black node at eighth direction is equal to the collision frequency of green node at eighth direction. This part is the streaming part.

- Thirdly, after obtaining the properties of fluid, the fluid angular velocities are able to be calculated. And then, to obtain the stokes resistance on ellipsoid cells, we use velocities and angular velocities from cell and fluid respectively to calculate the total force and torque acting on cells.

```
void Simulation::SimulationStart()
{
   .......
   .....
  Vector StokesForce = ellipsoidPointer->CalculateStokesForce(Velocity - FluidVel);
  Vector StokesTorque = ellipsoidPointer->CalculateStokesTorque( AngularVelocity -
                                                                 FluidAngVel, FluidVel);
   ........
   ..........
}
```

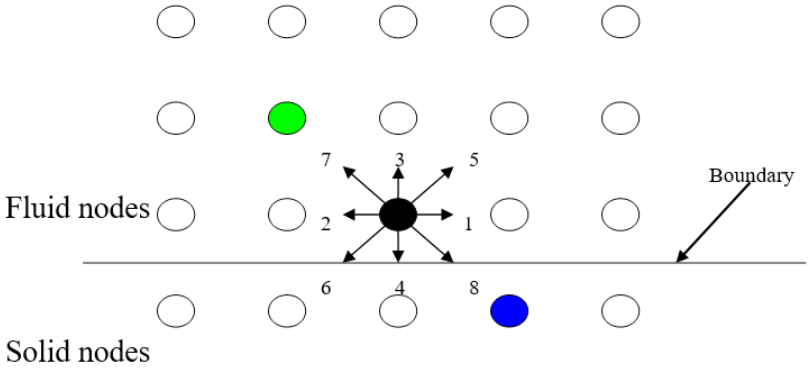


Figure 11: Bounce-Back Condition.

In the function, the code above invokes the *CalculateStokesForce(Velocity - FluidVel)* and *CalculateStokesTorque(AngularVelocity - FluidAngVel, FluidVel)* functions in order to obtain total Stokes Force and Torque acting on elliptic cells. In which the variables named *Velocity* and *FluidVel* represent elliptic cells' velocity and fluid velocity at corresponding position respectively.

- Fourthly, all the data about the properties of fluid will be saved in .txt file and .vtk files (.txt file used to visualize fluid in Direct3D, while .vtk files are used to visualize in ParaView Tool). For .vtk files, in order to obey its strict format and let the data be read in ParaView successfully, the header has been attached at the beginning of each file about fluid. The following code displayed the method to generate .vtk files:

```
void lbm_ns_3d(float tao, int tmax)
{
.....
 if(iteration%outputsteps==0)
 {
.....
     char filev[80] = "data\\veloc" ;
     char strv[80];
     sprintf_s(strv, "%d", ic);
     strcat_s(filev, strv);
     strcat_s(filev, ".vtk" );

     FILE *fv;
     if(   fopen_s( &fv, filev, "w+" ) != 0 )
     {
     ......
     }
.....
     fprintf(fv,"# vtk DataFile Version 2.0   \n");
     fprintf(fv,"Volume example   \n");
     fprintf(fv,"ASCII   \n");
     fprintf(fv,"DATASET STRUCTURED_POINTS   \n");
     fprintf(fv,"DIMENSIONS %d %d %d   \n", nx, ny, nz);
     fprintf(fv,"ASPECT_RATIO 1 1 1   \n");
     fprintf(fv,"ORIGIN 0 0 0   \n");
     fprintf(fv,"POINT_DATA %d   \n", nx*ny*nz );
     fprintf(fv,"VECTORS Velocity float   \n");
.....
      fclose(fv);
   }
     iteration=iteration+1;
.....
}
```

In which, the FILE variable, *fv*, store the information about .vtk. The main propose is to add header on each file. According to the code above, it is clear to show the header format at the beginning of real data. And in this code, the variables *nx, ny, nz* represent the size of each dimension respectively.

- Fifthly, all the new properties about fluid and elliptic cells will be sent to streaming step and ellipsoid model respectively. And the alteration will happen among them continuously.

The whole diagram displayed above offers a complete process of the second project (many ellipsoid cells immersed in fluid flow). While, for the first project (just simulates the fluid flow using Lattice-Boltzmann Method), it does not need to save data in .txt file. Because of the real-time simulation and visualization, all the data about the fluid velocity just need to be saved in temporary variables in order to finish visualization in one loop. Thus, the processes in the first project are relatively easier than those in the second project.

Visualization

Visualization in Direct3D: As discussed before, the application frameworks are based on Luna's shader frameworks and basic particle system class [22]. This framework gives a basic structure of graphic programming in shader. And it could help programmers to design their custom model or scene quickly. Additionally, the basic particle system class has been built, which has more interfaces and capabilities for extensions. This visualization in both projects utilizes the particle system class as base to design own custom particle system in order to satisfy special requirements, such as lifetime, size, position, color of each particle. Importantly, when particles have been seeded, the particle must check its own position in fluid space and obtain the corresponding instantaneous velocity. After many frames, the average velocity could be display and all the particles will faithfully follow the fluid flow. The following diagram displayed the procedure and brief structure of visualization (see figure 12):

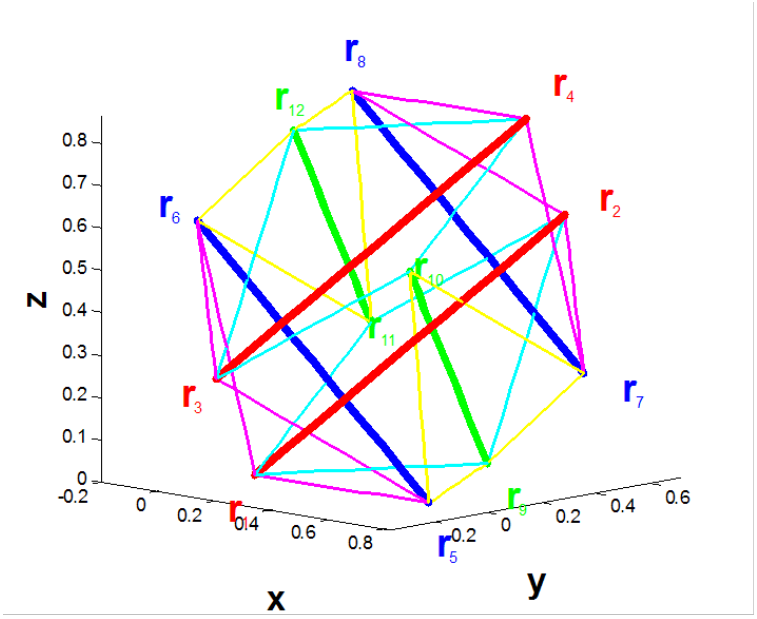


Figure 12: A Simplified visualization in Direct3D Pipeline.

The diagram above describes the main processes of visualization using DirectX. Firstly, initializing the main window, camera, and light is the basic step to visualization. In addition to this, the particles' properties will be initialized, concerning their position, size, velocity, lifetime, and color. And the size of container about fluid will be built with the size of 104*54*54. Followed by this, another rigid model, which should be initialized, is ellipsoid cells. Trying to render this model, the sphere models are able to be distorted by scale method. The proportion of three axes in ellipsoid is 1:1:4.

Secondly, all the data in terms of fluid velocities and all the properties in ellipsoid cells will be read and store in many temporary variables. And each node will record and store the instantaneous velocity around them. The following structure of code implements it.

The code above just displays one situation about trace particle at a position.

In this function PFluid::checkArea(Particle& checkPar, float dt), the instantaneous velocity about fluid could be restored in trace particles. The formal parameters, *Particle& checkPar, float dt* , represent the object of particle structure and data time respectively. And the variable, *pointPosX*, means the decimal part of x component of particle' position. And *intPosX*, means the integer part of x component. For instance, if the particle's position is p(1.3f, 1.6f, 0.8f). *pointPosX =0.3*, *intPosX=1*. And for other similar variables *pointPosY =0.6, intPosY=1. pointPosZ =0.8, intPosZ=0*. Because, in a project, all the fluid node's position is set as integer in three direction x,y,z. And the size of bounding volume is 1*1*1 (see figure 4). Due to the proper size volume, every particle in fluid space is able to obtain the corresponding instantaneous velocity. And, for instance, if the *poinPosX, poinPosY ,poinPosZ* of any trace particle are all between 0 and 0.5, trace particles will store the instantaneous velocity at position P(*intPosX, intPosY, intPosZ*). To explain it much clearer, if the particle stays at position P(5.3f, 12.2f, 2.5f) at point time t, it will obtain the instantaneous velocity of lattice node at position P(5, 12, 2). Because this particle stays into the bounding volume of lattice node at position P (5, 12, 2). But, if the poinPosX, poinPosY ,poinPosZ of any trace particle are all bigger than 0.5, trace particles will store the instantaneous velocity at position P(intPosX+1, intPosY+1, intPosZ+1). Therefore, if the particle stays at position P(5.8f, 12.6f, 2.9f) at point time t, it will obtain the instantaneous velocity of lattice node at position P(6, 13, 3).

In the code above, *mNodes[intPosX][intPosY][intPosZ]* presents the fluid node at position P. Therefore, the variable, *checkPar. initialVelocity.x*, obtained the instantaneous velocity from fluid node at position P. In the code above, the variable, *mScale*, is a const integer. It is able to scale the macroscopic velocity around each node. The reason why I need to enlarge velocity is the non-dimension velocity is very small, such as $1*10^{-4}$. As mentioned before, the non-dimension macroscopic velocity is equal to **u/c**. The formula is displayed as follow: $\tilde{\mathbf{u}} = \frac{\mathbf{u}}{c}$.

Obviously, it does not stop here, more situations have been considered in code, because we cannot make sure *poinPosX, poinPosY ,poinPosZ* of any trace particle are always between 0 and 0.5. After obtaining the velocity at corresponding node, the position of trace particle will update. The following code displayed the displacement at x, y, z direction separately.

```
checkPar.initialPos.x = checkPar.initialPos.x + checkPar.initialVelocity.x* dt;
checkPar.initialPos.y = checkPar.initialPos.y + checkPar.initialVelocity.y* dt;
checkPar.initialPos.z = checkPar.initialPos.z + checkPar.initialVelocity.z* dt;
```

Thirdly, every particle is going to check its current position so that it is possible to obtain the right instantaneous velocity around corresponding node. Meanwhile, when the ellipsoid cells receive their properties about position, directions of three axes, the cells are going to move to new position and rotation following directions. The following code describes the how to solve the problem in term if rotation of elliptic cells:

```
void Primitive::ReadDataFromFile(int ic)
{
        ……..
         spheDirection = D3DXMATRIX(dirU1.x, dirU1.y, dirU1.z, 0.0,
                                                dirU2.x, dirU2.y, dirU2.z, 0.0,
                                                dirU3.x, dirU3.y, dirU3.z, 0.0,
                                                0.0, 0.0, 0.0, 1.0);
         D3DXMatrixTranspose(&spheDirectionT, &spheDirection);
        …….
         D3DXMatrixTranslation(&spheWorld, pos.x, pos.y, pos.z);
         D3DXMatrixScaling(&spheScaling, 1.0f,1.0f,4.0f);
         D3DXMatrixMultiply(&spheScaling, &spheDirectionT, &spheScaling);
         D3DXMatrixMultiply(&spheScaling, &spheScaling, &spheDirection);
         D3DXMatrixMultiply(&spheMatrix[i], &spheScaling, &spheWorld);
         ……
        ……..
}
```

The function *void Primitive::ReadDataFromFile(int ic)* will be invoked every frame in order to obtain the position, direction of three axis about elliptic cells. After obtaining all the properties of elliptic cells, the three components of direction will be stored in vector variables, *dirU1, dirU2, dirU3*, separately. And the *pos.x, pos.y, pos.z* store the position of each elliptic cell. Followed by this, variable *spheDirection* is 4*4matrix will be created as above. The transpose of *spheDirection* matrix will be stored in another matrix variable, *spheDirectionT*. The matrix variable called *spheWorld* and *spheScaling* save the position and proportion of each axis. The main functionality of code *D3DXMatrixScaling(&spheScaling, 1.0f,1.0f,4.0f)*; is to scale the z axis of sphere to four times longer than other remaining axes. Therefore, the elliptic cell mode has been generated. In order to let elliptic cell rotated accurately and correctly, the sequence of multiplication is *spheDirectionT * spheScaling* spheDirection* spheWorld*

Fourthly, this step is to render each model concerning small particles, cubic container, and ellipsoid cells.

Fifthly, each frame's image will be stored in file according to retrieve a back buffer from the device's swap chain. And at the end of the fifth step, it will go back to read the next .txt file in order to prepare the next loop. And the alternation will work continuously.

Visualization in ParaView: This part is going to describe the steps concerning visualization in ParaView.

- Launch ParaView
- Go to open File > Open Data and choose the set of xxx.vtk files want to apply.
- Apply Filter
- Go to Filter > Streamtracer and apply it to create the xxx.pvsm datafile.
- Go to Sources > Box and draw two obstacles at proper position and apply them
- Click the .pvsm statefile and choose Display > Color > Velocity and apply it.
- Go to open File > Save animation in custom files.

First of all, after ParaView is launched, we will a set of data files named velocity.vtk and apply the filter. In ParaView the filter used for creating streamlines is the Stream Tracer filter. And the Stream Trace filter is able to generate streamline in a vector filed via a lot of seed points. After the stream trace is applied, a container will display without obstacles used in simulation. Therefore, the next step is to create two boxes as obstacles whose size is 10*54*30. The size is the same as that set in simulation. Followed by this, in order to use color to mark the magnitude of velocity, the color in Display option should be chosen and set as Velocity. Finally, to save all the images of each data file, the Save animation option in File could be selected to finish the visualization. Additionally, if all the pictures have been utilized following their sequence, an animation video is able to be created in Windows Live Movie Maker software.

Visualization in MATLAB: The name of the variable in MATLAB is similar to that in other programming languages. It also distinguishes the case of the variables. For instance, sum, Sum and SUM are three different variables. And the first character must be words. Unlike general languages, the MATLAB set a vector as a one-dimensional matrix commonly referred to as an array in other programming languages. And the array programming is also special. Its syntax is displayed as follows:

init : increment : terminator

According to the simple syntax, the loop for rendering streamline programmed as follows:

```
for m=1:1:5000;

    drawfig(m);

    str1 = sprintf('%d',m);
    S1 = ['fig\tsgt' str1 '.jpg'];
    saveas(fig,S1);

end
```

The variable m represents the No. of data file. And the whole No. of files is from 1 to 5000 with 1 increment. The second line of the code is the function for drawing streamlines. And last, each image depending on each file would be saved in "fig\tsgt" file.

And the following main code displays the method to draw streamline (the function drawfig(m)):

```
function drawfig(m)

……scan the data file including x, y……..
    [m,n]= size(V1);

    [x,y] = meshgrid(1:n,1:m);

    [sx,sy] = meshgrid(1:20:n,1:15:m);

    streamline(stream2(x,y,V1,V2,sx,sy));

……draw the scale……
    axis equal
```

The first line in function is to get the size of data. Because the components of velocity, x and y, are saved in two files separately. And the sizes of them are the same. Therefore, we only need to get the size form one of the files. To visualize the simulation in thermal model, the function streamline*(options)* could create streamline in specific options. Here, the *steam2()* function to precompute the vertex arrays, in which the arrays x, y represents the coordinates for V1,V2. and the sx,sy define the initial position of streamline. After thesis processes, the streamline about thermal model could be generated clearly.

The Results and Discussion

The Result and discussion in the first project

This project is real-time animation. And it is working on a single graphic card with 24 frames per second. In this animation, there is much acceleration at three different positions in this cubic container.

Control: The control is not the main feature and proposes in this project. The interaction's main aim is to help the user look around the container. In addition to this, the acceleration added in fluid could be controlled using keyboard in order to help user observe fluid flow. To add more acceleration in fluid, the interaction with users described as follows:

For camera:

The keyboard buttons: W: camera goes forward;

S: camera goes backward;

A: camera goes left;

D: camera goes right;

For adding acceleration:

The keyboard buttons: 8, 9, and 0: Add acceleration at proper position in fluid;

I , O, and P: Cancel the corresponding acceleration in fluid;

The following diagram displays the situation in which acceleration has been added in fluid (see Figure 13).

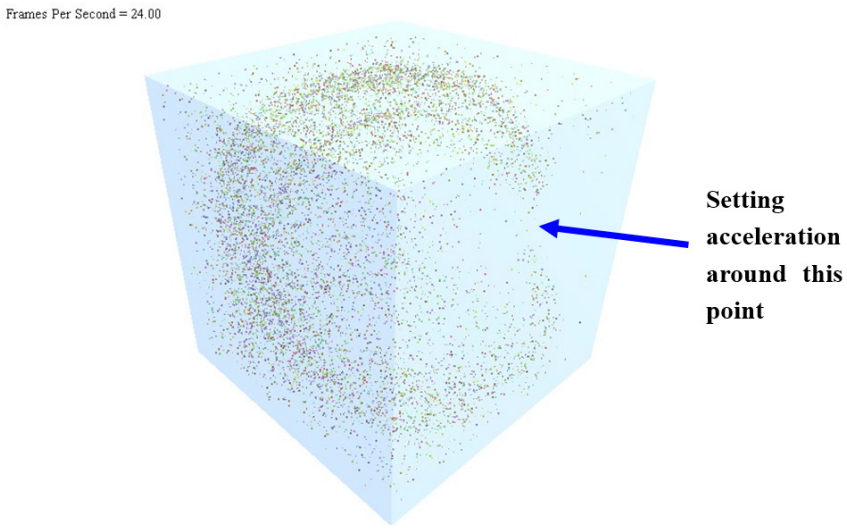


Figure 13: The snapshot of the first project.

Result of first project: The illustrated diagram above describes the result of all the particles following fluid flow and the acceleration adding round one point. From the diagram, a few particles stay around the position where have been pointed by blue arrow. But the remaining space has been fully filled with trace particles, forming a circle shape. Which means the fluid around the point with acceleration forced other fluid to move according to streaming and collision. Therefore, simulation and visualization of fluid flows are physically correct. This simulation supports a basic and credible fluid space for the next complex simulation concerning fluid flow and elliptic cells.

Discussion of first project: The first project above is a small scale, it could be simulated and visualized around 24 frames per second. The reason for using D3Q27 model is its accuracy in current project. But there is a problem about efficiency, when large scale is a requirement in another project. The following table describes the frame per second (FPS) at different size of Lattice-Boltzmann Model:

Table 1: Frame per second depending on different size of Lattice-Boltzmann Model.

Size X: Y: Z	FPS	Number of Particles
10:10:10	91	10,000
20:20:20	56	
30:30:30	24	
40:40:40	11	
50:50:50	6	
100: 100: 100	2	

When the size of Lattice-Boltzmann Model increases gradually, the FPS falls continuously. From table 1, there are ten thousand trace particles used to trace fluid flow in every size of testing model. And if the size is bigger than 30:30:30, the FPS is very small, and the simulation and visualization could not work efficiently and fluently. Therefore, Lattice-Boltzmann Model is not suitable to simulate large scale, such as big ocean and lake. Additionally, somebody also said that the Lattice-Boltzmann Method is not practical to run simulation on large systems because of the long computing time and limited memory resource [25]. But Lattice-Boltzmann is very flexible to be implemented in computer games [9]. The water wave and ocean usually appear to be realistic in computer games [10]. And Lattice-Boltzmann Method has the capability to simulate complex boundary easily [2,26,27]. But, because of the limited efficiency of the D3Q27 model, it is hard to be implemented in computer games. Fortunately, for 3D fluid simulation, there are several basic Lattice-Boltzmann Models [2], such as the fifteen-directional velocity (D3Q15), nineteen-directional velocity (D3Q19), and twenty-seven-directional velocity (D3Q29) Models [28]. According to the comparison among these three 3-D models, D3Q19 model offers a balance between computational reliability and efficiency [29]. Even though, D3Q15 is more efficient, but this model appears to be instable. Therefore, if we want to implement Lattice-Boltzmann model into computer games, D3Q19 is a better choice to improve the FPS and to simulate it accurately and correctly. And another way of boosting the frame per second is parallel programming.

The Result and discussion in the second project

For the second project, it divided simulation and visualization into two parts. Because of the large computation, the simulation is extremely expensive. And it is also a bit expensive for visualization. For this project, therefore, every image could be saved from their back buffer. And then, according to the sequence of these images, a movie is able to be created to finish this animation. As discussed above, the second project has been visualized using two methods; they are ParaView and Direct3D respectively. Several diagrams displayed below illustrate the results at different frames.

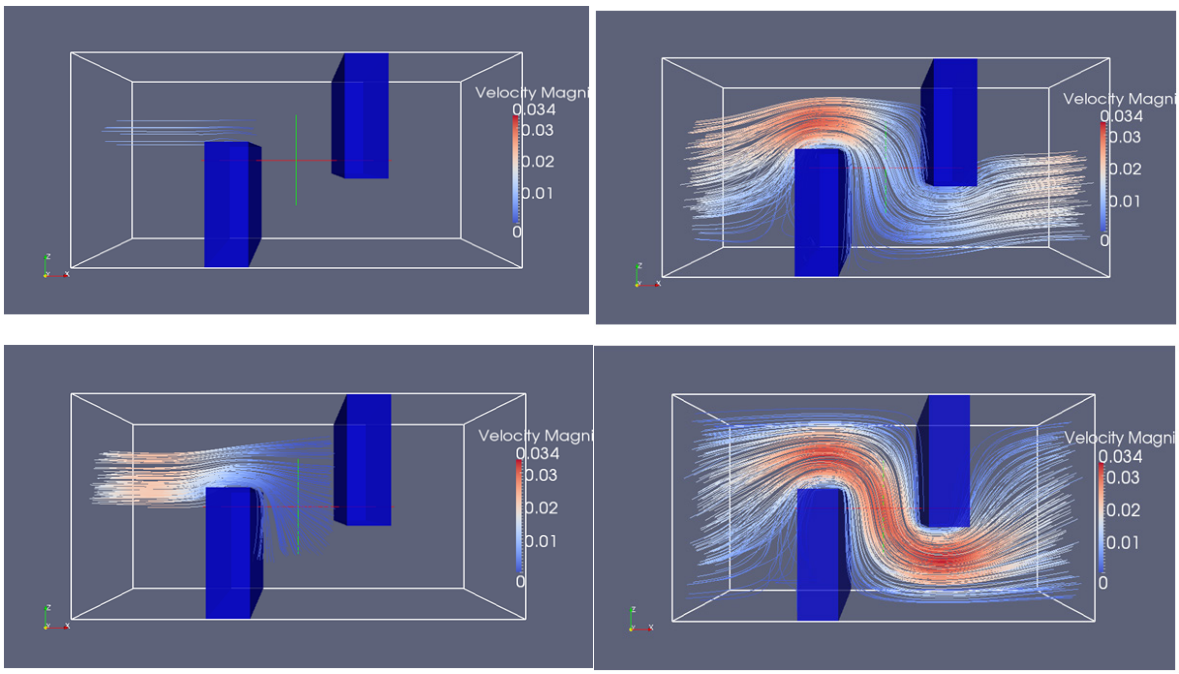


Figure 14: The result form Para View.

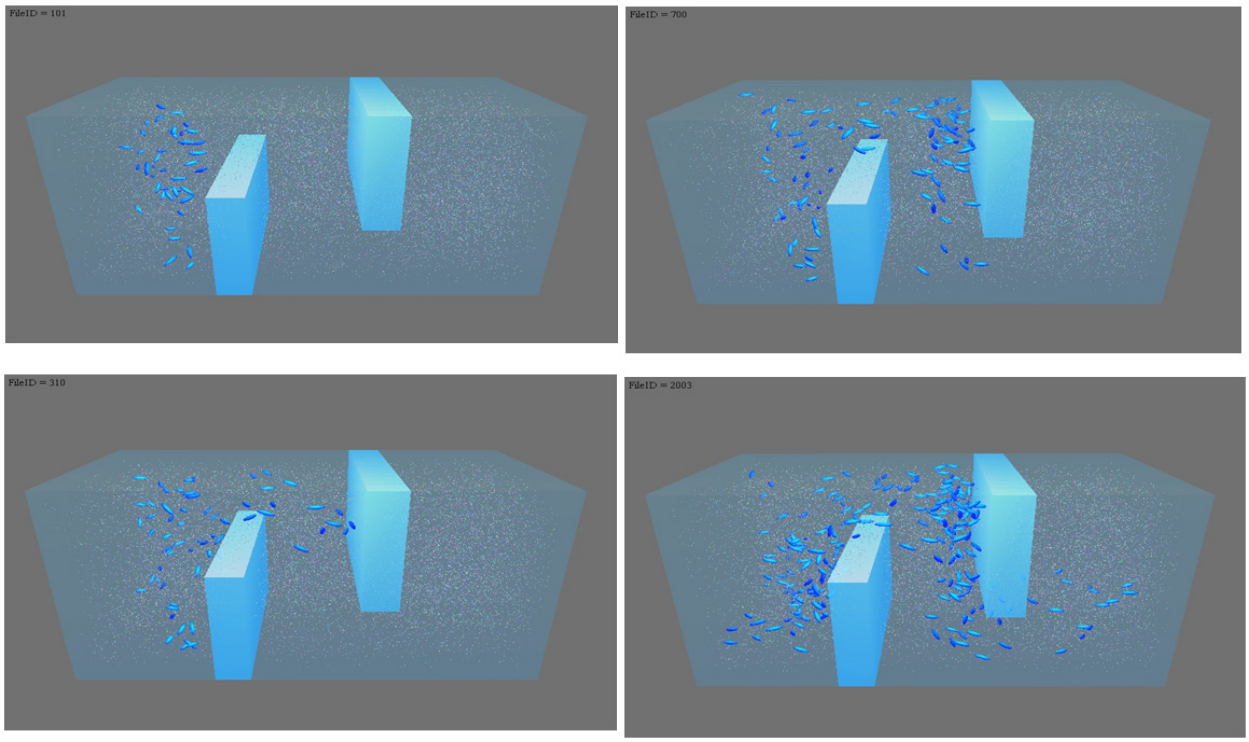


Figure 15: The result form Direct3D.

Result of Second Project: For the visualization in ParaView, it is clear to notice that the streamline runs through the whole container generally. Two blue cubic boxes in the left diagrams represent the obstacles. The fluid could run close to both obstacles instead of going through them. Meanwhile, maybe the colors about the streamline attract our attention. According to the scaleplate shown in the left serial diagrams, the red color means the velocity is relatively quick, while, for the blue color, it represents the relatively slow velocity in streamline.

Discussion of second Project: compared to the left diagram (the result in ParaView), the result in Direct3D has the same container and obstacles. While the ellipsoid cells added in this visualization support some new information and features for us. The ellipsoid cells follow the flow continuously, and the fluid flow forces all the cells to follow flow. These results prove the simulation is correct and still achieve the aim of the second project. Even though it is not easy to distinguish the path of small trace particles from the diagram above, the video in Compact Disc will show the fantastic trend about trace particles. The advantage of the visualization in Direct3D, could let us understand the moving path the velocity about fluid flow continuously.

From the result in ParaView, it is clear to observe that fluid flow bypassed two cubic obstacles perfectly in ParaView. Additionally, there is not any streamline cross the obstacles or run-away form container. Meanwhile, in this project, fixed acceleration was added on the upper left of container. On one hand, according to the streamline, it proves that the fluid around acceleration has pushed other fluid moving forward except for meeting solid objects. On the other hand, because the color of streamline represents the velocity magnitude in fluid, it is clear to find that the red color moves forward continuously. This means the path and velocity of fluid flow are both physically correct.

From the result in Direct3D, the elliptic cells have been added in fluid space and the interaction between fluid and cells have also been displayed in figure 15. The elliptic cells are able to follow the trend of the fluid flow. From the first picture in figure. 15, the cells spread on the left of container. For the second picture, all the cells met the first obstacle. And because of the Stokes Resistance on cells, the flow carried the cells to bypass the obstacle. When the fluid meets the second obstacle, the direction of fluid flow would be downward. And we find the cells have also been carried and bypassed the second obstacle. Therefore, the cells follow the fluid flow continuously due to the Stokes Resistance.

In this project, the elliptic cell is the virtual rigid body considered in this project. Generally, there are three kinds of shapes in bacterial cells, including bacilli, cocci and spirochetes [30]. The elliptic cell, here, is to simulate bacilli. Because of the shape, the rotation is able to show clearly and visibly which has capability to show the interaction from Stokes Resistance. But the ball cannot show all the features visibly. Therefore, the elliptic cell has been chosen for this project.

Here, maybe somebody will say the cell is soft body rather than rigid body. Exactly, if we use soft body, computing efficiency will be more expensive [31,32]. And soft body could be added in future work.

In this project, the 3-dimensional model with 27-directional velocity in Lattice-Boltzmann (D3Q27) has been implemented. While there are several models to simulate 3-D fluid [28], the reason why D3Q27 has been chosen as computational model is the accuracy of simulation rather than other 3d model (D3Q15, D3Q19). In 3D model (including D3Q15,D3Q19, DQ27), D3Q27 supports an accurate simulation in non-thermal fluid [33,34]. And D3Q27 is more computationally intensive [29]. Therefore, for accuracy, D3Q27 is better than the other two models. While, for efficiency, the nineteen-directional velocity model (D3Q15) and the fifteen-directional velocity model (D3Q15) are both better than D3Q27 [29,35]. Because, for the aim of this project, the real physical simulation in fluid flow is the main requirement, therefore, I will miss the efficiency. To improve efficiency and conserve accuracy, giving up the D3Q27 is not a suitable and excellent option. But, if parallel programming is utilized in this simulation, the efficiency is able to be improved and accelerated several times. Meanwhile, Lattice-Boltzmann is suitable and easy for parallel programming [36- 38].

The Result of Thermal Fluid in 2D

Thermal fluid is also visualized in two methods, Direct3D and MATLAB. These results are not the main proposal in this dissertation. But they prove that the method of visualization in Direc3D is correct and dependable.

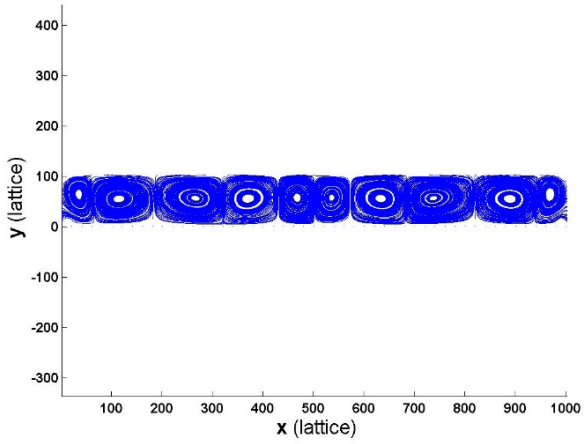


Figure 16: The result in MATLAB.

Figure 17: The result in Direct3D.

Compared to the result in Direct3D (see Figure.17), the result in MATLAB (see Figure.16) is better according to the two illustrations. But the visualization in Direct3D is better than that in MATLAB when the movie was made. Because of animation in Direct3D, is able to display tendency and velocity using trace particles. But the streamline in MATLAB just shows the general trends in fluid flow.

Conclusion

The result of the first project shows that using Lattice-Boltzmann three-dimensional model (D3Q27) could create the fluid space with small size smoothly. The interaction between each lattice node is very realistic. According to the first test, the method used here is straightforward and clear to simulate a natural fluid flow physically. And flexibility is another advantage of the Lattice-Boltzmann Method. Because the method is a computational fluid dynamic, the gas or water waves are also convenient to be simulated with LBM. But this method is somewhat expensive, if the size of fluid is very large. Therefore, it is not easy to simulate and render the real-time animation for a large scene. Especially, in computer games, the large lake or river is widely used and interaction between characters (human, animal, or rain droplet and so on) and fluid will be designed normally. Therefore, computational efficiency is a very critical problem in simulation.

The result of the second project demonstrated that ellipsoid cells swim in fluid flow. And this project not only based on Lattice-Boltzmann Method, but also the Stokes Resistance in shearing flow should be considered. According to the project, the main aim of the project is to get the result about the interaction between cells and fluid flow. Because of the dynamic fluid in space, the shearing force and torque have been added in total impact. For this project's result, on the one hand, the result in ParaView displayed the streamlines clearly. It partly helped to judge whether the fluid visualization in Direct3D is correct, but principally this result showed the whole trend of fluid flow physically and correctly. On the other hand, in Direct3D, the result showed both status in terms of cell and fluid flow continuously.

And finally, the result has shown that the ellipsoid could follow the fluid flow actually and realistically. And even though the simulation is successful and satisfied, there are still shortcomings. As mentioned before, because of the expensive computation, it is not easy to simulate large number of cells and large scene. Even though the large number of cells could be implemented successfully, it will cost too much time to simulate one frame, and this is not acceptable. Therefore, in the future work, more new methods should be considered and implemented to optimize current massive calculation.

Future Work

According to the results discussed in chapter 5, the massive calculation working in serial is a burden on simulation. Therefore, in future work, parallel program is an executable and straightforward to improve the efficiency of simulation. Recently, there are so many parallel programming methods that could be utilized including Windows threads, Message Passing Interface (MPI), OpenMP, and Inter Threading Building Blocks (TBB). Generally, supercomputer users mostly like using MPI because of the maximum flexibility in MPI. While this method will spend lots of time on debugging and it is still expensive for maintenance. For another method, OpenMP, mainly focuses on FORTRAN and C languages. But, due to the current project which is designed by C++, OpenMP is not the correct and suitable way to improve efficiency. Fortunately, the relatively new library has been launched in 2006 by Intel. This template library could support C++ perfectly. In addition to this, it is scalable, succinct, and reliable for parallel application. Thus, because of the advantages of TBB, it is extremely suitable to improve and boost the efficiency of the current project.

The project application uses rigid body (elliptical cell) within fluid flow. For the future work, another part to extend the work is to simulate the soft body (tensegrity model) swimming in dynamical fluid, which could let the simulation about cell swimming in tube more accurately and naturally.

The current project application uses an isothermal model to simulate fluid flow. And the acceleration added in current fluid is a bit constrained. Compared to Thermal Lattice Boltzmann Method, the fluid flow is able to be generated naturally according to the thermal model.

As mentioned before, Lattice-Boltzmann is able to simulate ocean and other fluid scenes physically. Therefore, in the future, implementing this method in computer games with parallel programming is one of the main tasks.

References

1. Chopard B, Marconi S (2002) Lattice Boltzmann Solid Particles in a Lattice Boltzmann Fluid. Journal of Statistical Physics 107: 23-37.
2. Illner R (2007) Lattice Boltzmann Modeling: An Introduction for Geoscientists and Engineers, Society for Industrial and Applied Mathematics.
3. Yue Hong Q (2006) Lattice Boltzmann models for Hydrodynamics. Journal of Hydrodynamics Ser B 18: 31-33.
4. Frisch U (1991) Relation between the Lattice Boltzmann-Equation and the Navier-Stokes Equations. Physica D 47: 231-232.
5. Mcnamara G, Alder B (1993) Analysis of the Lattice Boltzmann Treatment of Hydrodynamics. Physica A 194: 218-228.
6. Nemati H, Sedighi K, Farhadi M, Pirouz MM, Fattahi E (2010) Numerical simulation of fluid flow of two rotating side-by-side circular cylinders by Lattice Boltzmann method. International Journal of Computational Fluid Dynamics 24: 83-94.
7. Zhou JG (2002) A lattice Boltzmann model for the shallow water equations. Computer Methods in Applied Mechanics and Engineering 191: 3527-3539.
8. Liu H, Zhou JG, Burrows R (2010) Lattice Boltzmann simulations of the transient shallow water flows. Advances in Water Resources 33: 387-396.
9. Judice SF, Coutinho BBS, Giraldi GA (2009) Lattice Methods for Fluid Animation in Games. Computers in Entertainment 7, 56-56:29.
10. Robert G, Christopher C, Jerry T, James W (2010) Lattice Boltzmann water waves. Proceedings of the 6th international conference on Advances in visual computing Volume Part I Las Vegas NV, USA, Springer Verlag.
11. Jean Pierre B (2001) The Lattice Boltzmann Equation for Fluid Dynamics and Beyond: by Sauro Succi (Clarendon Press, Oxford, 2001) ISBN 0 19 850398 9. European Journal of Mechanics B/Fluids 22: 101.
12. Inamuro TYM, Ogino F (1995) A non-slip boundary condition for lattice Boltzmann simulations. Physics of Fluids 8: 1124.
13. Feng ZG, Michaelides EE (2009) Robust treatment of no-slip boundary condition and velocity updating for the lattice-Boltzmann simulation of particulate flows. Computers & Fluids 38: 370-381.
14. Ladd AJC (1994a) Numerical simulations of particulate suspensions via a discretized Boltzmann equation. Part 1. Theoretical foundation. Journal of Fluid Mechanics 271: 285-309.
15. Ladd AJC (1994b) Numerical simulations of particulate suspensions via a discretized Boltzmann equation. Part 2. Numerical results. Journal of Fluid Mechanics 271: 311-339.
16. Brenner H (1963) The Stokes Resistance of an arbitrary particel. Chemical Engineering Science 18: 1-25.
17. Brenner H (1964b) The Stokes Resistance of an arbitrary particel-II An extension. Chemical Engineering Science 19: 599-629.
18. Brenner H (1964c) The Stokes resistance of an arbitrary particle-III: Shear fields. Chemical Engineering Science 19: 631-651.
19. Math Works (2011)
20. Martin JP, Angelini R (2010) Making Para View Easier for the Customer: A Guide to Implementers. High Performance Computing Modernization Program Users Group Conference (HPCMP-UGC) DoD.
21. Schroeder WJ, Avila LS, Hoffman W (2000) Visualizing with VTK: a tutorial. Computer Graphics and Applications, IEEE 20: 20-27.
22. Luna FD (2006) Introduction to 3D game programming with DirectX 9.0c: a shader approach, Word ware Pub.
23. Prasad A (2000) Particle image velocimetry. Current science (Bangalore) 79: 51.
24. Brenner H (1964a) Effect of finite boundaries on the Stokes resistance of an arbitrary particle Part 2. Asymmetrical orientations. Journal of Fluid Mechanics 18: 144-158.
25. National Institute of Standards and Technology (2002) Parallel Computing.
26. Lallemand P, Luo LS (2003) Lattice Boltzmann method for moving boundaries. Journal of Computational Physics 184: 406-421.
27. Zhaoxia Y (2010) Pressure condition for lattice Boltzmann methods on domains with curved boundaries. Computers & Mathematics with Applications 59: 2168-2177.
28. He X, Doolen G (1997) Lattice Boltzmann Method on Curvilinear Coordinates System: Flow around a Circular Cylinder. Journal of Computational Physics 134: 306-315.

29. Mei R, Shyy W, YU D, Luo LS (2000) Lattice Boltzmann Method for 3-D Flows with Curved Boundary. Journal of Computational Physics 161: 680-699.

30. Science Blogs (2011) Meet the new white cell, same as the old white cell.

31. Baraff D (2003) Untangling cloth. ACM transactions on graphics 22: 862.

32. Hiroyuki O (1995) Electrophoresis of soft particles. Advances in Colloid and Interface Science 62: 189-235.

33. He XAL, LS (1997a) A priori derivation of the lattice Boltzmann equation. Physical Review E 55: R6333-R6336.

34. He XL, Li Shi (1997b) Theory of the lattice Boltzmann method: From the Boltzmann equation to the lattice Boltzmann equation. Physical Review E 56: 6811-6817.

35. Salom J (1999) Numerical Simulation of Convection Phenomena Based on Domain Decomposition Techniques and Experimental Validation. Universitat Politecnica de Catalunya Spain.

36. Qianshun C, Tong Y (2009) A Lattice Boltzmann Method for Image Denoising. Image Processing, IEEE Transactions on 18: 2797-2802.

37. Stratford K, Pagonabarraga I (2008) Parallel simulation of particle suspensions with the lattice Boltzmann method. Computers & Mathematics with Applications 55: 1585-1593.

38. Yasuda T, Satofuka N (2011) An improved entropic lattice Boltzmann model for parallel computation. Computers & Fluids 45: 187-190.

Appendix A

DirectX Software Development Kit (DXSDK_Jun10)

Downloadable from:

http://www.microsoft.com/download/en/details.aspx?id=6812

Visual Studio 2010 Professional Edition

Downloadable from:

https://www.dreamspark.com/products/Product.aspx?productid=25

MathWorks MATLAB R2009b

Downloadable from:

http://www.mathworks.co.uk/products/new_products/latest_features.html

ParaView

Downloadable from:

http://www.paraview.org/paraview/resources/software.html

The framework of Direct3D from FRANK D. LUNA

Downloadable from:

http://www.jblearning.com/catalog/9781598220162”

GLOBAL JOURNAL OF ENGINEERING SCIENCES

Acknowledgement: This research is supported by the project "High Education Funding of Guangdong Province: 2018KZDXM072"

Iris Publishers