Owen Martin

Mrs. Awde

Academy of Information Technology III

26 January 2018

Quarter Two Report of Progress

At the end of my first quarter in the AOIT III course, I decided that the most effective means of advancing my CELL program was to implement a more realistic method of particulate interaction computation. During the second quarter, I have achieved this goal by learning and applying one such method commonly referred to as Verlet integration. In addition to providing a basis of cellular response to stimuli, Verlet integration yields a very efficient method of displaying the many constituent elements of the focus cell. As of this semester's conclusion, the physical environment of the CELL program has been completed, however, a need for more coherent program output has been observed.

The initial, first quarter approach of using dynamic distances between particles in a cell membrane was inappropriate. In a real unicellular organism, the particles that comprise its membrane are bound together chemically; the distances between them do not change notably whenever an outside force is applied. In the updated version of the CELL program, the distances between neighboring particles in a cell are maintained by connector objects in Verlet integration – the magnitude of which cannot be changed, but instead allow the particles to rotate around one another. It is this new behavior that allows the focus cell to maintain a stable and dynamic formation even under the forces of colliding into the program's barriers.

Verlet integration also creates the emergent property of rotational momentum in the focus cell. If a force is applied to one side of the cell that creates a spinning motion, the cell will

maintain this rotational velocity in the form of momentum, and subsequently react to a collision with a barrier differently from a normal vector collision. The individual velocities and masses of each particle in the cell membrane contribute to this overall motion because of their connections to each other; the initial velocity of a single particle will be transferred over time to every other particle, creating an average velocity that applies to the entire cell.

Just as Verlet integration utilizes each individual particle to calculate overall cell movement, the individual positions of each particle are used to paint the general shape and location of the focus cell. In order to draw the background of the cell, the positions of every particle are first passed into a polygon object constructor, which is then drawn by the program. Next, the average of every particle's *x* and *y* coordinates are used to approximate the center of the cell. The locations of every particle in the first polygon object are then shifted a specific distance towards this center, shrinking the polygon object but retaining its shape. This new object is then drawn over the initial background with a slightly different color. By employing this process repeatedly, a gradient visual representation of the cell was created.

The CELL program currently operates very smoothly but has begun to show problems with its display. While running the program, the focus cell and all other drawn material will sporadically flicker white, creating an undesired visual effect. This effect does not alter the physical processes of the program; it can be concluded that the cause of the problem lies in the program's ability to display itself through its paint method. In my attempts to research this, I have come across several alternate ways of displaying a program in Java, some of which I hope to explore in the third quarter of AOIT III. Additionally, I plan on implementing an unrestricted environment by removing the walls from the program and maintaining a central visual point

around the focus cell. Hopefully, this will pave the way for food and AI to be added to the program at a later date.