Hashemi_Yashar_LabReport4

Yashar Hashemi 9/4/17

Title: A brief description of the assignment goal

Author: Yashar Hashemi

Background/Introduction: In this lab, I am going to create simulations of cancer cells attempting to reach a red blood cell. I hypothesize that the higher the amount of friction the cancer cells traverse through to reach the red blood cell, the slower they will go. I am going to test how three different coefficients of friction affect the cancer cell. I will be using the coefficients .1, .25 and .5. I am assuming that the cells with a friction coefficient of .1 will reach the source fastest, .25 the second fastest, and .5 the slowest. Additionally, I will be creating drag once the cancer cell enters the red blood cell as the red blood cell is composed of liquid.

Methods/Design:

- I created a Mover class
- Created a mover constructor that takes a location and a mass value
- The Mover class has a applyForce method which divides a PVector by the mass of the mover and adds it to the acceleration
- Mover class has an update function which adds the acceleration to the velocity and the velocity to the location, thus moving the mover.
- Mover class has a check edges function which inverts the velocity if the mover has reached an edge
- Mover has a display function which displays it as an ellipse with a diameter of ten
- Create an EMC class
- EMC constructor takes the velocity of a mover, calculates the friction based on the velocity and the friction coefficient, and stores the friction PVector as well as a directional PVector:

```
ECM(float co, float spd, PVector fric, PVector inf){
  //friction coefficient
  coefficient= co;
  friction =fric.get();
  //will it make the object speed up or slow down
  friction.mult(spd);
  friction.normalize();
  friction.mult(coefficient);
  influence = inf.get(); }
```

- Create a Source class

- Source class has a constructor that initializes its location as well as gives it a mass of 50 and gravitational pull of 5.

- Source class has an attractor method which returns a PVector:

```
//gets vector between location of two object
 PVector force = PVector.sub(location, m.location);
 float distance = force.mag();
 distance = constrain(distance, 5.0, 25.0);
 force.normalize();
 //using attraction/gravity formula
float strength = (G * mass * m.mass) / (distance * distance);
force.mult(strength);
return force;
```

- In setup(), I initialized an array of 25 movers at one end of the grid, each with a mass of 1, a source, an ECM and I displayed the ECM

- In draw(), had a counter to keep track of the number of iterations, I displayed the source object, and I iterated through the list of movers

- In the loop, I created a PVector to affect the direction of the movers and put in the proper parameters for the EMC changing the coefficient three times, one for each test.

- I also made an if statement that checks if one of the movers has touched the Source object. If it did, then drag is applied to the mover object and the top speed is limited to 5:

```
float speed = movers[i].velocity.mag();
//velocity^2 * coefficient
float dragMagnitude = .05 *speed *speed;
PVector drag = movers[i].velocity.get();
drag.mult(-1);
drag.normalize();
drag.mult(dragMagnitude);
movers[i].applyForce(drag);
movers[i].topspeed = 5;
```

- Finally, in the draw() function I used the attract method to attract the mover to the source (attract()), I applied the attract force the the movers in the array(applyForce()), I applied the EMC's friction to the movers in the array (applyForce()), I applied the EMC's directional influence to the movers in the array (applyForce()), I updated the movers in the array (update()), I used the checkEdges() function to make sure they don't go off the screen and I displayed all the movers in the array.

Results: I have successfully created a mover object that gets attracted to a source and responds to My hypothesis turned out to be correct. My results show that the with a higher friction coefficient, or a higher amount of friction, it takes longer for cancer cells to reach red blood cells. The figures below further demonstrate my results:

Figure 1: When the ECM has a friction coefficient of .1, it took 82 iterations for the cancer cell to reach the red blood cell.

Figure 2: When the ECM has a friction coefficient of .25, it too 128 iterations fro the cancer cell to reach the red blood cell.

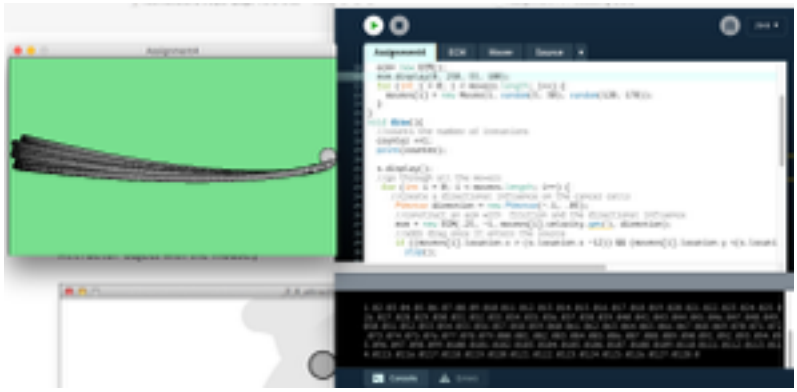
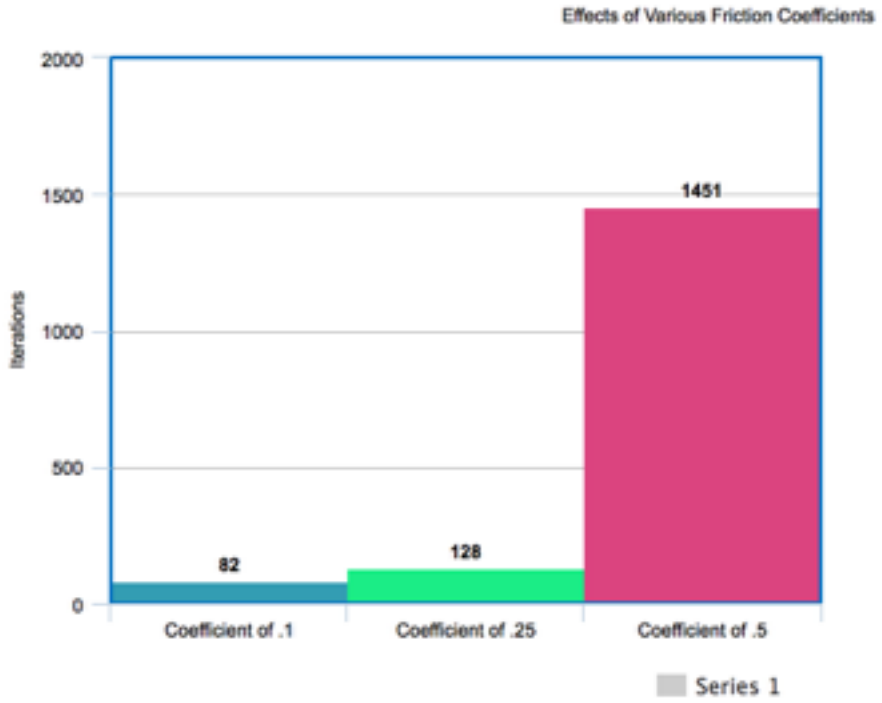


Figure 3: When the ECM had a friction coefficient of .5, it took 1451 iterations for the cancer cell to reach the red blood cell.

Figure 4: A histogram comparing the amount of iterations the cancer cell took to reach the red blood cell at various levels of friction.



Conclusion:
The cancer cells reach the red blood cell faster when there is less friction.

Next steps: Given more time, I would run more tests to give

Credit/Acknowledgements: I worked with Sadie Schiffman-Eller on this assignment. I used resources and code examples from Daniel Shiffman's book: *The Nature of Code*

Citation:

Shiffman, Daniel, Shannon Fry, and Zannah Marsh. The Nature of Code. D. Shiffman, 2012.