Hashemi_Yashar_LabReport2 9/20/17

Title: A Program for a moving object using Perlin Noise

Author: Yashar Hashemi

Background/Introduction:

In this lab I am designing an array of twenty cancer cells that display on screen and move using Brownian Motion 30% of the time, move towards a source object 30% of the time, and move using my own algorithm 40% of the time. I will be using Vectors for the direction, acceleration, velocity and location of my cancer cells. The purpose of this lab is to learn about vectors, vector functions, acceleration and directionality.

Methods:

- Create a Mover(Cancer) class
- Create five vectors: location, velocity, acceleration, destination and direction.
- Create two floats: mass and top speed.
- Create a Mover(float m, float x, float y) constructor that takes a location(x, y), and a mass (m) value. The Mover has an initial velocity and acceleration of (0,0), and a top speed of 5.
- Create an update(float x, float y) function which takes an x and y time parameter for the destination of the object.
- When update is called, 30% of the time, the destination PVector initializes using the parameters of update. Set the direction PVector the difference between the destination and the current location using PVector.sub(). Normalize the direction PVector and multiply it by a coefficient to make it small enough to use as acceleration.
- 30% of the time update() sets the acceleration of the to a random value that allows the Mover object to accelerate in any direction.
- 40% of the time update() accelerates the object using my own algorithm(accelerating it to the right)
- In update(), add the acceleration PVector to the velocity PVector, limit the magnitude of the velocity PVector using PVector.limit() and add the velocity PVector to the location.

- Create a checkEdges function that tests to see if the Mover object has reached the edge of the screen. If it has, it inverts the velocity
- The Mover object is displayed using a display() function which creates transparent blue ellipses of various sizes based on mass.
- Create a Source class.
- Create a constructor for the source object that stores a random x, y location.
- Display the source object as large, light gray ellipse.
- In the setup() function, I created a window with a width and a height of 1000, initialized an array of twenty Movers using a loop, set the background color to 130 on the grayscale and initialized the source object.
- In the draw() function I called the update(Source.x, Source.y), display() and checkEdges() functions from the Mover class.

Results:

Using PVectors, I have successfully written and implemented a Mover class that accelerates randomly between three different algorithms. The Mover class has the ability to accelerate towards a specific destination, randomly and in a certain direction. Additionally, I have successfully initialized and called functions on an array of Mover objects.

Conclusion: I have created an Cancer class that moves using Vectors. I gained a deeper understanding of modeling real world movement by using velocity and acceleration.

Credit/Acknowledgements: I used resources and code examples from the Processing tutorial on Objects and book: Nature of Code, Chapter 1, in constructing this program/sketch.

Citations:

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