INF-1100

Assignment 2

Magnus Dahl-Hansen

March 2020

1 Introduction

In this report I describe my implementation of the flocking simulator "Boids" where I try to have an object-oriented design and make use of classes, inheritance and objects. My program is written in Python with the help of the Pygame library.

1.1 Requirements

The simulation should follow three rules that makes the Boids move in a lifelike manner and support some extra features:

First rule: Boids should steer towards the average position of local flock mates

Second rule: Boids attempts to avoid hitting other Boids

Third rule: Boids should steer towards the average heading of local flock mates

Additional features: The Boids attempts to avoid obstacles and a flees from a Hoik that can eat them

2 Technical Background

When starting on this assignment one should be familiar with how classes and inheritance works.

Classes: an object contains a collection of variables that gives it a form and a behavior. A class is a container where all these variables are stored.

Inheritance: is a tool that enables a programmer to create a class that takes all the functionality from another existing class, and therefore makes it possible for a class to inherit attributes and methods from another class. A class that inherits from another class is often called the child class, whilst the one from which it inherits is called a parent class.

3 Design/implementation

I created four separate python files for my simulation, some containing classes for the objects and one containing the necessary methods and attributes for running the simulation. The Game class is mainly the class that inherits methods from other classes and is where all the methods are updated and run. In the Game class I implemented the methods for creating Boids and the collision between these and the Hoik. The Boids are created in random locations on the screen and added to a list when the left mouse button is pressed, whilst the Hoik and obstacles are created from the start. I think this is a bad design, implementing these two methods in the Game class, but I will comment on this later.

The Boid class is where I implemented most of my code since the program mostly revolves around the Boids. This class contains ten methods, where some are just basic methods that moves the Boids in a random direction, draws them and makes them loop through the edges of the screen. The more complex methods are the different rules which gives the Boids a more natural behavior and the effects of these rules can be modified in the heading of the Boid file:

First rule is the alignment rule which makes a Boid align itself with other Boids. Each Boid is given a radius for which it scans for other Boids. If other Boids are within this radius, their average velocity is calculated and the Boids attempts to change their velocity to the average velocity of the group. The next rule is cohesion. It is implemented in almost the same way as the alignment rule, except it calculates a Boid groups average position instead of velocity. A single

Boid which is in contact with a group of Boids will attempt to move itself to the average position of a group. Third rule is separation; this method makes a Boid detect if other Boids are too close to itself, if so, the Boid will change its average heading away from another Boid. In addition to these rules the flee method works in the same way as the separation method, but each Boids position is compared with the Hoiks position instead.

The obstacle class and the Hoik class are very similar, though the obstacle is just a static circle drawn in the middle of the screen whilst the Hoik is a circle that moves along the coordinates of the computer mouse.

4 Discussion

I could not get the Boids to avoid an obstacle placed in the middle of the screen, which is a major flaw of the simulation in my opinion. I wanted to implement this obstacle in the same way as the Boids avoid the Hoik, but for some reason I got an error in the command prompt telling me that the list of obstacles wasn't iterable. I was not able to figure this out and therefore I chose to comment out the lines in the code and remove the obstacle from the simulation.

Another minor flaw is when a group of Boids gets big enough, some of the Boids in the middle tends to overlap each other and therefore breaking the separation rule. The reason for this is because they either do not have time to act quickly enough and separate OR the Boids in the middle of a group gets pushed in an undesired direction by the mass/force of other surrounding Boids. This can be easily fixed by increasing the separation force on each Boid or their perception radius, but I chose not to do so since increasing the separation force makes the Boids act nervously and/or increasing the perception radius made the flock to big. I think the simulation looks nicer when the Boids are a bit closer to each other and doesn't act nervously.

I know it's a bit random that the "create_boids" and "Hoik_eat_Boid" methods are defined within the Game class and I see this as a minor design flaw. The reason why it is done so is because I struggled in the beginning to understand how class inheritance works and therefore this was an easy solution. After completing the assignment, I knew how to make one class inherit from another, but I lacked the time to fix this design issue.

5 Conclusion

This assignments task was to create a flocking simulation called Boids, which should follow three rules for flocking. I have explained my implementation and design and tested my program. It runs smoothly but have some minor errors and lacks an obstacle.

From this assignment I have learned a lot about class inheritance and ways to better design my code/implementations and will try to use this in a better way in the next project.

6 References

https://www.python-course.eu/python3_inheritance.php

https://www.programiz.com/python-programming/inheritance

http://www.kfish.org/Boids/pseudocode.html

https://www.pygame.org/docs/ref/math.html#pygame.math.Vector2

https://natureofcode.com/book/chapter-6-autonomous-agents/