SOFT351

Assignment 2

Joseph Kellaway

# End User Guide

Run the “Boids.sln” Microsoft Visual Studio Solution file and click run (or the “Boids.exe” file in the “\x64\Debug” pathway) to run the game.

The user is presented with a bird’s-eye view of the winged-bear which they can move around the screen as with the previous assignment. The bear controls remain the same, with the F1 key remaining the key pressed to display controls on-screen whilst in the program. New commands added to the software are as follows:

* ‘r’ key – increase the strength of the leash affect forcing boids to stay near the default camera.
* ‘d’ key – decrease the strength of the leash affect forcing boids to stay near the default camera.
* ‘t’ key – increase the distance that the boids can move away from the default camera.
* ‘f’ key – decrease the distance that the boids can move away from the default camera.
* ‘y’ key – increase the range at which boids see nearby boids (affects turning decisions).
* ‘g’ key – decrease the range at which boids see nearby boids (affects turning decisions).
* ‘u’ key – increase the strength of cohesion (boid’s desire to be the centre of the flock).
* ‘h’ key – decrease the strength of cohesion (boid’s desire to be the centre of the flock).
* ‘i’ key – increase the strength of alignment (boid’s desire to face the same direction as the flock).
* ‘j’ key – decrease the strength of alignment (boid’s desire to face the same direction as the flock).
* ‘o’ key – increase the strength of separation (boid’s desire to maintain a minimum distance between themself and other boids).
* ‘k’ key – decrease the strength of separation (boid’s desire to maintain a minimum distance between themself and other boids).
* ‘p’ key – increase the distance that the boids desire to be away from each other.
* ‘l’ key – decrease the distance that the boids desire to be away from each other.

The default values are set in such a way that the boids appear to move around the map together somewhat whilst moving apart if they get too close to each other, avoid travelling too far away from the default camera position and flee the bear if it gets too close. Using the above commands the player can change the relevant force or length from 10% to 200% of the default value.

# Programmer’s guide

What I am looking at from a programmer’s point of view. How your program fits together. The flow through your program. This is not intended to be an exercise in formally documenting a software project.

On startup 100 Boid objects are created and stored in an array called flock. The Boid class inherits from the base class Thing3D as the boids themselves are objects with coordinates, rotations, scales etc. in the same manner that the bear is. The spawnFlock() function uses a random number generator to place the boid within a specified radius of the default camera by randomly selecting a value between the negative and positive maximum distance for the x-coordinate and then selects a random number between the negative and positive remaining value to assign to the z-coordinate. A random rotation between 0 and 2 pi radians about the x-axis is also selected to ensures that the boids don’t all start facing the same way.

Each frame the program checks for any keyboard inputs and moves the bear in the same manner as the previous assignment. After this a loop starts to calculate how each boid will turn and then move based on their environment:

1. A range check is performed on every other boid in the world to see if it is within the local neighbourhood of the boid calculating its current movement.
   1. If there are no boids nearby, the boid will move randomly. This consists of two actions:
      1. A random turning motion decided by a random number between 1 and 3 where 1 is turn left, 2 is turn right and 3 is neither (i.e. continue straight).
      2. A random speed adjustment decided by a random number between 1 and 3 that favours boids travelling at approximately half speed. If the boid is travelling under half speed then 1 and 2 speed the boid up, and 3 slows it down; if the boid is travelling over half speed then 1 speeds it up, and 2 and 3 slow it down.
   2. If there are other boids nearby, then the boid flocks:
      1. It adjusts its speed as previous described (in a. ii.).

# Additional Note to the Programmer

The random numbers generated to decide the boids spawning is done uniformly using a single random number generator in the main “Boids.cpp” file. This is so that the area permitted to the boids should be approximately evenly filled with boids.

The random numbers generated to decide boid behaviour are created uniformly by a random number generator within the boid class. This is so that whilst each boid should turn left, right and continue straight 1/3 of the time, each boid could roll to turn left at exactly the same time, but with a shared random number generator with a uniform distribution this would be less likely to happen.

# Software Engineering Issues

The software engineering issues, such as the trade-off between performance and good practice, which I have asked you to address.

Loops don’t re-do calculations

# Ownership

As anticipated in the proposal document, I started with my own submission to coursework part 1 (in fact, from a user perspective most of the functions affecting the flying bear remain unchanged). This submission had been created using a combination of my own work and the demos available on the DLE.

Added to the above are the boids themselves and various global values that affect the boids behaviour. The boid class is an extension of the Thing3D class (since they all need positions and rotations etc. anyway) with modified movement, boid-like decision making (which was researched via the provided link <http://www.red3d.com/cwr/boids/> as well as various AI lectures given as part of the course at Plymouth University).

# Evaluation

After more thoroughly considering what had been initially proposed and the intention of boids themselves, I felt the need to revise what was to be created. Initially I set out to allow the user to direct the boids by attracting them towards a playable entity, though boids having their own (albeit collective) “free will” conflicts with this idea in my opinion. Instead, the theme of predator and prey was used to allow the user to move an entity around the map and the boids will attempt to avoid it.

# Appendices