CCSI 3161 Assignment #1

Objectives: To develop an understanding of OpenGL animation.

Due date: Oct 3rd, at 11:59 PM.

Hand in: Electronic submission of entire Visual C project (source, compiled, project files, etc.) on Brightspace. Please zip up your *whole* project directory and submit the zip file. In addition to your project files, also include an up-to-date runnable build.

Note, before beginning your assignment, please read:

1. The coding style guideline.
2. The policy on plagiarism.

General Comments:

In this assignment you do not have to handle a window re-shape.

Boids of a Feather [20 marks total]:

(i) 2D Boids [15]

The term boids refers to simulated birds, or bird-oids. Each boid is represented by its position and its flying vector (towards which it always points). Simply draw each boid as a triangle pointing in the direction of its movement. Boids are constantly trying to:

1. stay in the flock,
2. avoid walls and obstacles and
3. avoid hitting each other.

It is *very* important to note that there is no overall control of the flock direction. It naturally emerges from the local interactions of individual boids and their nearest neighbors. You should not code an overall direction for the flock.

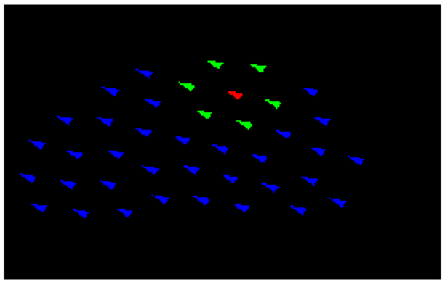
In this assignment, each boid only knows about its nearest 6 neighbors (nearest in the sense of physical distance). The neighborhood size should be set to 6, but this should be a changeable constant in your code. Remember that its 6 closest neighbors will change over time.

Each Boid tries to smoothly match its flying vector with its 6 nearest neighbors (by a small amount at each step). But Boids will not move to be within a certain small distance of any other boid. *Within a certain distance*, the closer it gets to a neighbor, the more it will push away (moves away inversely proportionally to the distance). These two conflicting tendencies will find a stable balance in flight, if coded and tweaked correctly. Learning how to carefully tweak parameters is a useful skill and something that is done extensively when working with Game Engines.

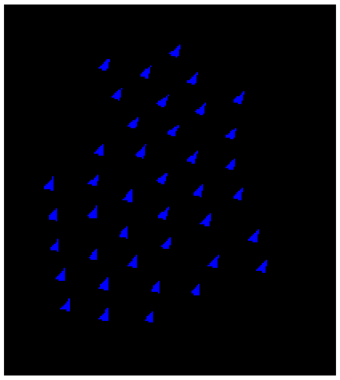
Boids also need to avoid scene edges. Boids will move in the reflected direction from scene edges. *Within a certain distance of the edge*, the closer a boid is to the edge the more the boid will push away from the edge (moves away inversely proportionally to the distance). Boids should never touch the edge or get stuck at the edge, though they might shuffle around a bit in the corners as they work things out. Edge avoidance takes precedence over the flocking behaviors so that the flock will likely break formation when avoiding edges and obstacles.

The total flock size should be set to 40, but this will be a changeable constant in your code. The 40 boids might form multiple sub-flocks at certain points, however, there will likely not be more than 2 or 3 sub-flocks most of the time (see the example executable).

The user can press keys '1' to '9'. When the user presses the '*n*th' key, you will show that boid in red and its 6 nearest boids in green (this is very useful for debugging). Pressing '0' turns it off. Other points are drawn blue.



The boids should form regular formations when in open space:



The boids should not be too clumped up or jittery in general. It is the convincing movement that is most important in this assignment, the boids should not be moving like pinballs, but like smoothly flying boids forming flocks and avoiding obstacles.

The page-up and page-down keys increase and decrease the overall speed of the flock:

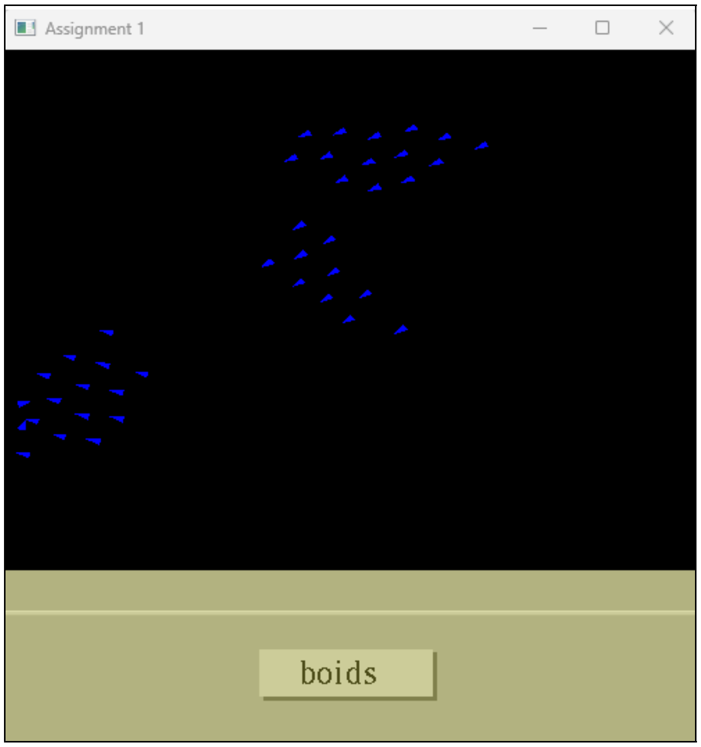
GLUT_KEY_PAGE_UP	:	increase speed
GLUT_KEY_PAGE_DOWN	:	decrease speed

But, limit the speed so that it is always greater than zero.

Hints:

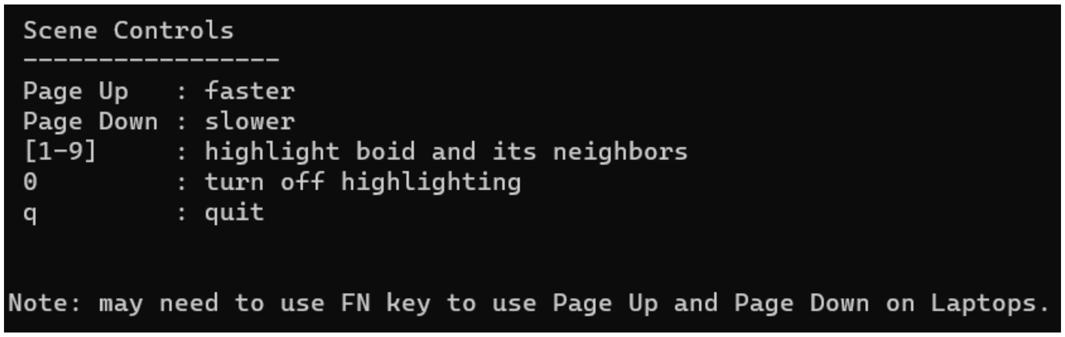
1. You should have two copies of the entire flock, one flock state that is currently being altered and the flock state at the previous time step. Make changes to the current flock but base those calculations on the previous state, so that you are not changing the previous state as you are making your calculations.
2. You need to limit the maximum change in direction so that the boids don't change too quickly. Use appropriately small step sizes and constants for smooth changes. Tweak and then tweak again, and again...
3. See the pseudo-code at the end of this assignment for the basic algorithm.
4. Feel free to use arrays with sizes defined by constants rather than trying to dynamically allocate memory.

(ii) Toggle Button [3]: Make a toggle button with an on/off state. The button should change appearance (like color) when on. The "boids" button pauses or continues the flocking simulations.



(iii) Keyboard controls [2]

Use printf to dump out a listing of all the keyboard key functions that you are supporting. Get the keyboard controls working. These are easy marks so start with this…



```
Scene Controls
-----------------
Page Up   : faster
Page Down : slower
[1-9]     : highlight boid and its neighbors
0         : turn off highlighting
q         : quit


Note: may need to use FN key to use Page Up and Page Down on Laptops.
```

Assignment 1 – Basic Algorithm

1) Randomize boid positions and directions

2) copy CurrentFlock to PreviousFlock

3) Update the CurrentFlock based on the PreviousFlock

```
for each boid in flock
{
        - find the boid's nearest 6 neighbors, N

        if  (boid is within a certain distance of wall)
        {
             -   change boid's direction vector to point a bit more away from wall
                 by an amount equal to the inverse distance to wall
                 (times some small constant)
        }
        else
        {
             - alter the boid's direction vector to be more like the average of all N

             if (distance between the boid and any neighbor in N < some distance)
             {
                   - move boid away from that neighbor by the inverse of the
                   distance  (times some small constant)
             }
        }

        -   update the boid's position along its direction vector
            (the amount will define the boid's speed)
}
```

4) repeat from 2

Note: you may need to put a cap on the maximum change of distance at each iteration, and the length of each boid's direction vector.

S. Brooks