Itoro Esiere & Kaleb Alanis
ime257, koa239

Boids: 3D Flocking and Emergent Behavior

Goals, Implementations, and Algorithms

Our goal was to create a Boids simulator in 3D space in OpenGL. Boids is an artificial life simulation which is used to simulate the flocking of birds or fish. Essentially, the simulation entails objects in a space, which move with a velocity as a function of the entire flock. Different incentives result in the individual objects, or boids, to change their respective velocity. With simple rules, these simple agents demonstrate emergent behavior. Our code can be split into several parts. First was the Boids class. The Boids class serves as the 3D model for the boid in the simulation. It have a position, velocity, acceleration and other aesthetic characteristics. Every loop step, the acceleration updates the velocity and the velocity updates the boids location. Second is the rendering of these boids in space. Each boid has a model matrix in which given a pyramid, the axis which intersects the top point maps to the velocity of the boid. To avoid crazy jumps, we normalize and limit the magnitude of velocities and acceleration. The third implementation was the flocking heuristics. The velocity of each boid is a function of the entire flock. In our simulation, we create a weighted average of different incentives which direct a boid and change its heading. We use four incentives which are weighted into the final heading:

Cohesion
Keeps the flock together by steering each boid to move toward the average position of the other boids in its flock. We calculate the center of mass of the entire boid system. Then we add a force onto the acceleration which directs the boid's heading toward the center of mass.

Separation
Steers each boid to avoid crowding and colliding with other boids within the flock. A set desired separation distance is set. If the distance between two boids is less than this threshold, a force is applied to steer the heading away from this boid and increase this distance. The final incentive is the average of all of these avoidance forces with every boid in the system inversely proportional to the distance.

Alignment
Steers each boid towards the average direction of the flock. This flock will have an average heading vector. This incentive adds a force to the acceleration which steers the heading toward the average heading in the boid system.

Itoro Esiere & Kaleb Alanis
ime257, koa239

Boundaries When boids reach a predescribed boundary a force is applied to push them away from it and continue in their flock state. This is done by adding a force to the acceleration of a boid toward the center of the world.

The fourth implementation was the flocking modes. The three modes are natural, chaos, and return.

Natural
Natural state of the boids is when their incentives (separation, cohesion, and alignment) are weighted. This mode is the initial mode.

Chaos
Chaos state of the boids is when their incentives are randomly generated and applied.

Return
As the boids freely move out of view, you may enter this state to guide them to in front of the camera.

Bugs and Limitations

We encountered some problems along the way. We tried to implement multiple camera modes. The two additional modes were “center” and “boid”. The center camera mode was so that the user can view the flock at its center of mass. However it was possible for us to have multiple flocks and our design could not allow for us to arbitrarily pick one. The boid camera mode was so that the user could track individual boids. Due to the fact that the number of boids is large, it’d be impractical to have this mode. Another bug we had were boids flying away into infinity. We fixed this issue by applying boundaries. This is an artificial life simulation which means that we do not control the boids movements but they follow our algorithm. Since this is the case, we are forced to follow them by spanning the camera and occasionally lose track of the boids position after a period of time. The initial boids are sometimes not seen during execution of our program because position are randomly generated with srand and rand from the c++ libraries. This bug requires that we either rerun the program or left-click to generate more boids onto the screen.