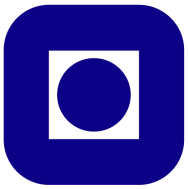

NTNU

IT3708 - Sub-symbolic AI Methods

Project 1: Flocking and Avoidance With Boids

Mathias Ose

February 17, 2015

Implementation

The simulation was implemented with Python 2.7. The framework Pygame was used to create the graphics, otherwise the enitre program is self-implemented.

Each boid is an instance of a Boid class. The world is a simple cartesian coordinate system, and each boid has fields which hold the current x, y-position and the current vx, vy velocity. Predators are a subclass of the Boid class with some different logic, and obstacles a separate class with just the position and radius properties.

The instances of these classes, as well as global parameters, are stored in a BoidWorld class which which has methods for adding and removing objects as well as simulating the flow of time. The gui code runs the main loop, and it uses the BoidWorld as an interface for getting the information it needs to draw the world as well as sending the command to tick the time forward after it is done drawing.

Many of the calculations require knowing the direction and magniture of the vector between two points. Since the boid world wraps around at the edges, the function calculating this vector needs to find the shortest relative vector out of the four possible ones.

In most cases we want boids that are closer to each other to affect each other with more force than ones that are further apart. However, the relative vector between boids has a magnitude that is greater when they are further apart. This is corrected by normalizing the vector to length 1, then multiplying by a factor between inverse proportional to the distance between the points.

The calculations of the forces are not all alike, but at some point they all involve normalization, so the returned values are all close to unit vectors. Therefore the values returned by the force calculating vectors can mostly be considered the direction of the force, with the weights applied later determining the magnitude.

Separation force

For each neighbour the relative vector between the boids is calculated and scaled by the factor described above. The sum of these vectors is a vector pointing from the boid towards the "most dangerous" area, so the negative of this vector is the one to use as the separation force.

Alignment force

The alignment force is direction of the sum of the velocities of the neighbouring boids. The velocity component vectors are summed, then the sum vector normalized.

Cohesion force

The cohesion force is also found by averaging the absolute (global) positions of the neighbouring boids. This results in a point which represents the "center of mass" of the neighbourhood. The relative vector from the boid to this point is then normalized and applied as the force.

Avoidance force

The avoidance force calculation is the most complex one. Boids should obviously avoid crashing with an obstacle, but the force shouldn't be as naïve as the separation force. If a boid can see an obstacle, but is not on a collision course, there is no reason to let the obstacle change the course.

If an obstacle is within the range where a boid can see it, the program tests if the line extending from the current velocity vector of the boid intersects the obstacle circle. If this is the case then the boid is on a collision course and should act. If the current course steers clear of the obstacle no force is applied.

The force applied is a vector normal to the currenct velocity. The coordinate of the collision point relative to the obstacle center is used to determine whether a left or right turn is best to avoid collision.

Flight force

Boids fleeing from predators works the same way as boids avoiding boids. The difference is that this force has a higher weighting when applied.

Chase

Predators chasing boids work much the same way as the cohesion force between boids, steering the predator towards the "center of mass" of the nearby boids, but the inverse distance factor is also applied here so that the predator will prioritize chasing the closer boids.

Behavior

Neighbourhood radius	15 x boid radius
Boid max speed	100
High weight	15
Low weight	1

Table 1: Some of the values used in the final simulation

Varying the three flocking weights around 10 allowed for flocking behavior to emerge while also keeping the movement quite smooth, avoiding abrupt changes like immediate stops or sharp turns at high speeds.

In the scenarios we want the high forces to dominate the low forces. To do this I chose 15 as the high value and 1 as the low value.

Scenarios

In scenario 1 the cohesion force dominates. The boids move about in tight groups, but very chaotically. Externally the "blob" of boids keeps changing directions, and internally the individual boids are slaloming left and right, circling around, and colliding with each other constantly.

In scenario 2 it is the alignment force that dominates, and the boids quickly align in the same direction in large groups. The groups start out in different directions, but as these groups meet each other, they eventually find a global direction of travel that they all follow. The flocks now move at constant speed, and the individual boids keep the same positions within the flocks.

When the separation force dominates the others in scenario 3, as you would expect the boids spread out, evenly distributing themselves on the whole screen. Interestingly, even though the alignment weight is low, after a while they stablize in a very much aligned flow, with only a little bit of wobbling to the sides. It would seem that alignment is an emergent property of strong separation. Thinking a little about it, it makes sense that with a world of limited size with wrapping, the boids can only spread out so much, and that a "rogue" boid will have no where to go but with the flow.

With both alignment and cohesion as strong forces in scenario 4, the boids flock in tight groups where they all align perfectly. The groups tend to take long oval shapes in the direction of travel rather than wide or circular shapes.

In scenario 5 the separation and cohesion forces are both strong. Since these are sort of opposite forces, the boids act very chaotic, whirring around at different speeds, changing direction all the time and not forming any permenent groupings.

In scenario 6 the separation and alignment weights are high, and the boids spread out evenly again, and quickly align in the same direction. All in all very simillar to scenario 3, with the boids being even more stable in this scenario, versus having a slight wobble in scenario 3.