Boids

Emily Weng and Christopher Wang

December 12, 2018

1 Motivation

In movies and video games, flocks of birds, shoals of fish, and other such crowds of animals are often used to add realism to a scene. Given that flocks of real birds can number up to 1.5 billion, manually controlling and animating the movement of each bird in a flock would be unfeasible. To solve this problem, Craig Reynolds invented the "boids" model of flocking in 1986[1].

"Boids" is the coined term for "generic simulated flocking creatures". As this definition suggests, the boids algorithm can be used to model flocks of all sorts: shoals of fish and swarms of insects. The algorithm can even be modified to model crowds of people or colonies of bacteria. One of the most famous uses of simulated flocking is the Battle of Helm's Deep in *The Lord of the Rings*, where the boids model was used to animate armies of humans and non-humans.

2 Background

The boids model of flocking specifies three fundamental rules of movement that give rise to flocking behavior:

- *separation*: a boid moves away from other boids which are too close
- *alignment*: a boid moves in the same direction as other boids
- *cohesion*: a boid moves toward the center of other boids

Because real birds are only influenced by birds in close proximity, Reynolds also specifies that a given boid is only influenced by a local neighborhood of boids, where the neighborhood is defined by some distance and some angle field of view within which the boid can "see".

The pseudo-code provided by Conrad Parker [2] implemented the above three rules. Conrad also suggests some additional behaviors, which he did not describe fully in the pseudo-code:

- *position bounds*: a boid stays within some bounding shape
- *velocity limit*: a boid moves at a velocity below some limit
- *object avoidance*: a boid avoids some object(s) (e.g., a prey boid avoids a predator boid)
- *object seeking*: a boid seeks some object(s) (e.g., a predator boid seeks a prey boid)

[1] Reynolds, C. W. (1987) Flocks, Herds, and Schools: A Distributed Behavioral Model, in Computer Graphics, 21(4) (SIGGRAPH '87 Conference Proceedings) pages 25-34.

[2] http://www.vergenet.net/ conrad/boids/

3 Approach

In this project, we simulate a flock of boids using a particle system, where each boid is a particle with a position, orientation, and velocity. We simulate separation, alignment, and cohesion as forces between particles.

To implement the boids particle system, we build on the code from problem set 3. At every time step t, we update the velocity $\vec{v}$ and position $\vec{x}$ of a boid according to the Euler method:

$$\vec{v}_t = \vec{v}_{t-1} + h \cdot \vec{a}_{t-1}$$
$$\vec{x}_t = \vec{x}_{t-1} + h \cdot \vec{v}_{t-1}$$

where

$$\vec{a}_{t-1} = \frac{\vec{F}_s}{m} + \frac{\vec{F}_a}{m} + \frac{\vec{F}_c}{m} + \cdots$$

where $\vec{F}_s$ is the separation force, $\vec{F}_a$ is the alignment force, $\vec{F}_c$ is the cohesion force, and $m = 1$.

The separation force $\vec{F}_s$ is calculated as:

```
separation(boid_position):
    neighborhood = boids within view and proximity
    f = <0, 0, 0>
    for n in neighborhood:
        separation = boid_position - n.position
        if separation.abs() < SEPARATION_DISTANCE:
            f += separation
    return f
```

The cohesion force $\vec{F}_c$ is calculated as:

```
cohesion(boid_position):
    neighborhood = boids within view and proximity
    total_position = <0, 0, 0>
    for n in neighborhood:
        total_position += n.position
    f = (total_position - boid_position) / |neighborhood|
    return f
```

The alignment force $\vec{F}_a$ is calculated similarly to the cohesion force $\vec{F}_c$, with velocity substituted for position.

We also add a force to keep the boids within some position boundaries:

```
boundary(boid_position):
    f = <0,0,0>
    if boid_position outside x_bound:
        f += x_bound - boid_position
    if boid_position outside y_bound:
        ...
    return f
```

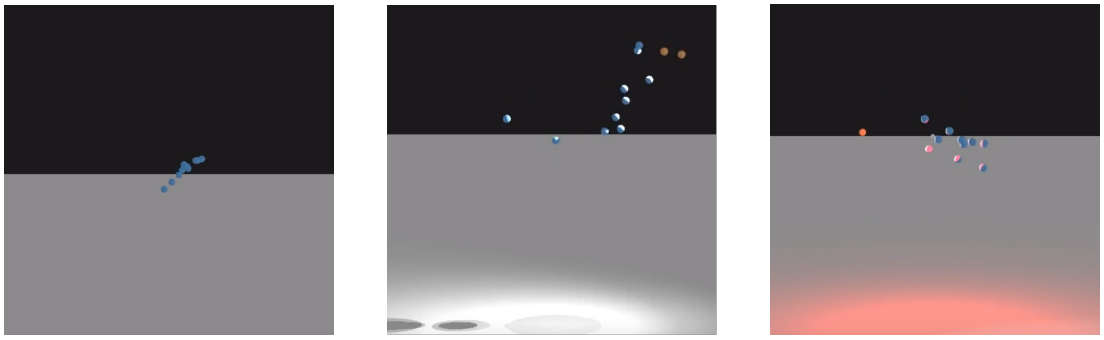

Figure 1: (left to right) a) ten prey (blue); b) two predators (orange); c) boids congregating around a target, which casts a red light

Finally, we use $\vec{v}_t = min(\vec{v}_t, \vec{v}_{limit})$ to keep their velocity capped at some upper bound. To implement object avoidance and object seeking, we modify the cohesion and separation forces between boids and objects: avoidance is essentially separation from the object(s) to avoid, and seeking is essentially cohesion toward the object(s) to seek. Then, predator and prey behavior can be implemented as special cases of object avoidance and seeking.

4 Results

We simulate flocks with ten prey and two predators (Figure 1). To render our results, we use the ray tracer from problem set 4. In the main program, we output a text file that describes the scene at each time step of the simulation. Then, we input the text files into the ray tracer to produce a frame for each scene and stitch together the scenes into a final video.

For visual effect, we attach a spotlight to each predator that points in the direction of its orientation. We implement the spotlight by creating a point light with an angular falloff (Figure 1b). We also attach a red point light to the target around which boids congregate (Figure 1c).

A video of our simulations can be seen here [3]. The boids flock exhibits the desired behaviors. At time 0:04 of the video, the boids can be seen to flock together. At time 0:16, the flock exhibits predator avoidance. At time 1:18, they congregate around a target. The rest of the video displays further extensions to the boids model, such as movement through 3-dimensional space, as well as other visual effects, such as ray-traced shadows.

5 Conclusion

In this project, we simulate a flock of birds using the boids model and render it using ray tracing. In the future, we hope to expand on this work, perhaps by rendering it through the perspective of the predator, rendering it with soft shadows, or rendering it with more striking colors. More significantly, we hope to improve upon this work by making it more efficient. As it is now, the calculation of each force loops through all the boids of a neighborhood when only one such loop is necessary in order to calculate all forces. Likewise, the calculation of certain forces can likely be combined, such as those of separation and avoidance and those of cohesion and seeking.

[3] www.youtube.com/watch?v=01Aeq0AXZ-M