Contents

Flocking Bird Simulation

Indian Institute of Technology Delhi

Anubhav Palway (2016CS10368) & Sunil Kumar (2016CS10314)

April 2018

1 Abstract

Here we will present the simulation of flocking behavior of the starling birds and their behavior when they are attacked by some predator. The first flocking-behaviour simulation was done by Craig Reynolds and he called his simulation program "Boids". It's still to this day the most used model for simulating flocking behaviour. We will also use this model to make simulation model. We will also look at flocking behavior of the birds when threatened by a group of predators. A set of forces are set up to describe the state of the prey, that in turn determines their behaviour in different scenarios. An effective strategy is found so all members of the flock can survive the predator attack, taking into account the advantages of the predator's greater translational velocity and the prey's higher angular velocity.

2 Introduction

The first flocking-behaviour simulation was done by Craig Reynolds in 1986 and he called his simulation program: "Boids"[2]. Boids was named after the simple agents in the system which he also called boids. The boids used a set of simple rules to determine how they would move. The three rules formatted by Reynolds in his Boids program are still the basis of modern flocking simulation and are widely used to these days. These are the three rules as they are described by Reynolds:

- **Separation:** Steer to avoid crowding local flocking.
- **Alignment:** Steer towards the average heading of local flockmate.
- **Cohesion:** Steer to move toward the average position of local flockmates.

In the subsequent sections, we will describe the basic terminology used and, the algorithms used to satisfy the three properties of the Reynolds model. Then we will introduce the saturation functions (for the range of velocities of the boids), unicycle model (used for calculating the velocities), energy constraints and the various forces which we consider for the simulation purposes and the mathematics behind it.

3 Defintions

3.1 Boid

A representation of a bird in flocking simulations. The word is most commonly used in the context of Boids, the original computer simulation of flocking behaviour by Craig Reynolds and derivative of Boids. The name "boid" corresponds to a shortened version of "bird-oid object", which refers to a bird-like object.

3.2 Flocking Behaviour

Flocking behavior is the behavior exhibited when a group of birds, called a flock, are foraging or in flight. There are parallels with the shoaling behavior of fish, the swarming behavior of insects, and herd behavior of land animals.Computer simulations and mathematical models which have been developed to emulate the flocking behaviors of birds can generally be applied also to the "flocking" behavior of other species. As a result, the term "flocking" is sometimes applied, in computer science, to species other than birds.

3.3 Emergent Behaviour

When individual objects interact with each other directly or indirectly to create much more complex results. Another characteristic of emergent behaviour is that the end result is hard to predict even if each interaction in itself is simplistic. Anthills, economics, the evolution are all examples of how agents abiding to relatively simple rules can create a system more complex than the sum of its parts. Reynolds Boids algorithm can be said to create an emergent behaviour.

4 Algorithm

4.1 An overview

This model follows this simple structure: calculate the current state, draw the state on the screen and repeat Algorithm 1 shows the structure of a typical Boids implementation. The three rules of the Boids algorithm only changes the speed and course of the boid and can be applied in any order. After the velocity has been updated, we can update the positions of the boids. We update the position by simply calculating where it would be if it fly straight for some small amount of time.

Algorithm 1 An overview

```
1: Data: A group of boids.
2: Result: Simulates flocking behaviour with an animation.
3: for all Frame do
4:     for all Boid do
5:         separation(boid);
6:         cohesion(boid);
7:         alignment(boid);
8:     for all Boid do
9:         $boid.x \leftarrow cos(boid.course(\theta)) * b.velocity * dTime$
10:        $boid.y \leftarrow sin(boid.course(\theta)) * cos(boid.course(\phi)) * b.velocity * dTime$
11:        $boid.z \leftarrow sin(boid.course(\theta)) * sin(boid.course(\phi)) * b.velocity * dTime$
12:        draw(boid)
```

4.2 Boid

The boid is the bird representation in flocking simulation model. Each boid object should at least have the following attributes to describe the state it is in.

- **Loaction:** The x, y and z coordinates of the current position of the boid.
- **Course:** The angle of the current course course of the boid.
- **Veclocity:** The speed of which the boid is traveling.

The course and speed could of course be represented by a equivalent velocity vector instead. But often more attributes is needed to make the simulation more convincing. Putting an upper limit on how fast the boids can move and turn is a common improvement.

4.3 Cohesion

Steer to move toward the average position of local flockmates.[3] Cohesion is the rule that keeps the flock together, without it there would not be any flocking at all. Algorithm 2 shows how this algorithm could be implemented, it finds the average position of the neighbourhood boids and tries to move the boid towards it.

Algorithm 2 First Rule: Cohesion

```
1: Data: A boid.
2: Result: The course of the boid is updated
3:
4: goal ← (0,0);
5: neighbours ← getNeighbours(boid);
6: for all nBoid in neighbours do
7:     $goal \leftarrow goals + positionOf(nBoid)$
8:     goal ← goal / neighbours.size()
9:     steerToward(goal,boid)
```

4.4 Separation

Steer to avoid crowding local flockmates.[3] If a flocking behaviour is to be convincing it must also avoid collisions between the boids. This rule attempts steer the boid away from possible collisions. It's important to note that the distance from which the boids start to avoid each other must be less than the distance from each other(due to the cohesion rule). Otherwise no flocks would be formed.

Algorithm 3 Second Rule: Separation

```
1: *Data:* A boid.
2: *Result:* The course of the boid is updated
3:
4: goal ← (0,0);
5: neighbours ← getNeighbours(boid);
6: **for all** nBoid in neighbours **do**
7:     $goal \leftarrow goals + positionOf(Boid) - positionOf(nBoid)$;
8:     goal ← goal / neighbours.size()
9:     steerToward(goal,boid)
```

4.5 Alignment

Steer towards the average heading of local flockmates.[3] This rule tries to make the boids mimic each other's course and speed. If this rule was not used the boids would bounce around a lot and not form the beautiful flocking patterns that can be seen in real flocks.

Algorithm 4 Third Rule: Alignment

```
1: *Data:* A boid.
2: *Result:* The course and velocity of the boid is updated
3:
4: dCourse ← 0;
5: dVelocity ← 0;
6: neighbours ← getNeighbours(boid);
7: for all nBoid in neighbours do
8:     $dCourse \leftarrow dCourse + getCourse(nBoid) - getCourse(Boid)$;
9:     $dVelocity \leftarrow dVelocity + getVelocity(nBoid) - getVelocity(Boid)$;
10:    dCourse ← dCourse / neighbours.size()
11:    dVelocity ← dVelocity / neighbours.size()
12:    boid.addCourse(dCourse)
13:    boid.addVelocity(dVelocity)
```

4.6 Defining the neighbourhood

The neighbourhood of a boid is the neighbourhood it perceives and is a core part of this algorithm. The neighbourhood decides what other boids, a boid should take into account when deciding the next move. Each pseudo code of the three rules of the Boids algorithm (algorithms 2, 3 and 4) uses a function called getNeighbours(Boid) to get the neighbors of the given boid. The three rules could and should use different implementation of the get-Neighbours(Boid) method. It's important for example that the separation rule only acts on the closest boids. If separation and cohesion would work on the same neigh- bourhood they would cancel each other out. Raynolds original implementation simply defined the neighbors as the boids within a certain radius. Another possible definition of the neighbourhood is to let each boid look at the N closest boids instead. This thesis attempts to compare the two definitions to find strengths and weaknesses in both definitions and find under what circumstances either is superior.

5 Mathematics

5.1 Velocity Function

The velocities in different can be formulated as a time dependent equation system:

$$\begin{cases} \dot{x} = v \cdot cos\theta \\ \dot{y} = v \cdot sin\theta \cdot cos\phi \\ \dot{z} = v \cdot sin\theta \cdot sin\phi \\ \dot{\theta} = \omega \end{cases} \tag{1}$$

5.2 Saturation Function

Since there is an upper and lower limit for the discussed velocities, we have to limit those. It is done with the help of a saturation function.
The saturation function for the translational velocity is given by:

$$sat(v) = \begin{cases} v_{max} & \text{if } v > v_{max} \\ v & \text{if } 0 < v < v_{max} \\ 0 & \text{if } v < 0 \end{cases} \tag{2}$$

where v_{max} is the boid's maximum velocity.

Similarly, saturation function for angular velocity is:

$$sat(\omega) = \begin{cases} \omega_{max} & \text{if } \omega > \omega_{max} \\ \omega & \text{if } -\omega_{max} < \omega < \omega_{max} \\ -\omega_{max} & \text{if } \omega < -\omega_{max} \end{cases} \tag{3}$$

where ω_{max} is the boid's maximum velocity.