**Flocking**

*Santosh Shedbalkar*

*santosh.s@digipen.edu*

Portions in RED: Copied word-for-word from <http://en.wikipedia.org/wiki/Flocking_(behavior)>

Portions in GREEN: Copied word-for-word from <http://harry.me/2011/02/17/neat-algorithms---flocking/>

Portions in BLUE: Copied word-for-word from <http://www.cs.toronto.edu/~dt/siggraph97-course/cwr87/>

Portions in YELLOW: Copied word-for-word from <http://www.red3d.com/cwr/boids/>

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.

This article is about the modeling of flocking behavior. From the perspective of the mathematical modeler, "flocking" is the collective motion of a large number of self-propelled entities and is a collective animal behavior exhibited by many living beings such as birds, fish, bacteria, and insects. It is considered an emergent behavior arising from simple rules that are followed by individuals and does not involve any central coordination.

Flocking behavior was first simulated on a computer in 1986 by Craig Reynolds with his simulation program, Boids. This program simulates simple agents (boids) that are allowed to move according to a set of basic rules. The result is akin to a flock of birds, a school of fish, or a swarm of insects. The aggregate motion of a flock of birds, a herd of land animals, or a school of fish is a beautiful and familiar part of the natural world. But this type of complex motion is rarely seen in computer animation. This article explores an approach based on simulation as an alternative to scripting the paths of each bird individually. The simulated flock is an elaboration of a particle system, with the simulated birds being the particles. The aggregate motion of the simulated flock is created by a distributed behavioral model much like that at work in a natural flock; the birds choose their own course. Each simulated bird is implemented as an independent actor that navigates according to its local perception of the dynamic environment, the laws of simulated physics that rule its motion, and a set of behaviors programmed into it by the "animator." The aggregate motion of the simulated flock is the result of the dense interaction of the relatively simple behaviors of the individual simulated birds.

Each boid has direct access to the whole scene's geometric description, but flocking requires that it reacts only to flock mates within a certain small neighborhood around itself. The neighborhood is characterized by a *distance* (measured from the center of the boid) and an *angle,* measured from the boid's direction of flight. Flock mates outside this local neighborhood are ignored. The neighborhood could be considered a model of limited perception (as by fish in murky water) but it is probably more correct to think of it as defining the region in which flock mates influence a boids steering.

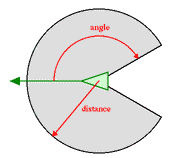
****

Figure 1. A boid's neighborhood.

A slightly more elaborate behavioral model was used in the early experiments. It included predictive obstacle avoidance and goal seeking. Obstacle avoidance allowed the boids to fly through simulated environments while dodging static objects. For applications in computer animation, a low priority goal seeking behavior caused the flock to follow a scripted path.

**Flocking Rules**

Each entity on the map, called as a “boid”, moves around while being governed by a few simple rules. Each boid starts out at the center of the map with a random velocity, and for each frame of the simulation, a new velocity is calculated using the flocking algorithm. For each boid, the algorithm uses the boid’s current velocity, its neighbor's velocities, and its position relative to its neighbors to calculate this new velocity.

Basic models of flocking behavior are controlled by three simple rules:

1. Separation - avoid crowding neighbors (short range repulsion)
2. Alignment - steer towards average heading of neighbors
3. Cohesion - steer towards average position of neighbors (long range attraction)

When these rules are used in combination display the full blown flocking behavior.

**Code:**

/\*

Function: Flock

Parameters: void

Returns: void

Description: Calculates the flocking velocity of the boid

\*/

void Boid::Flock( void )

{

CalculateNeighbors();

Vec3 accel( 0.0f, 0.0f, 0.0f );

\_separation = Separation();

\_align = Alignment();

\_cohesion = Cohesion();

accel += \_separation;

accel += \_align;

accel += \_cohesion;

\_vel += accel;

float len = Length( \_vel );

if( len > BOID\_MAX\_VEL )

{

normalize( \_vel );

\_vel \*= BOID\_MAX\_VEL;

}

}

**Separation**

While in a flock, each boid tries not to run into each other one in the flock. They try to remain separate by keeping a specified amount of space in between themselves. Each boid checks all the other boids on the map to see if the distance between them is too small, and if so, adds an inversely proportional amount to its velocity in the opposite direction.

****

Figure 2. Separation: steer to avoid crowding local flock mates.

**Code:**

/\*

Function: Separation()

Parameters: void

Returns: Separation vector

Description: Calculates the separation vector of the boid amongst its flock mates

\*/

Vec3 Boid::Separation()

{

Vec3 separation( 0.0f, 0.0f, 0.0f );

int count = 0;

Boid\* boid;

float dist;

Vec3 diff;

for(int i = 0; i < \_neighborCount; ++i )

{

boid = \_neighbors[i];

diff = \_currPos - boid->\_currPos;

dist = Length( diff );

if( dist > 0.0f && dist < \_boidSeperation )

{

normalize( diff );

diff /= sqrtf(dist);

separation += diff;

++count;

}

}

if( count > 0 )

{

separation /= (float)count;

}

return separation \* WEIGHT\_SEPERATION;

}

**Alignment**

Each boid in a flock tries to head in the same direction as the rest of the flock, which is the responsibility of the alignment portion of the algorithm. Each frame, each boid looks at the heading in which it is travelling in comparison to the headings of all its neighbors, and realigns itself to match their heading. The velocity vectors of each boid within the NEIGHBOUR\_RADIUS are averaged and the resulting vector points in the average direction of the flock, which the boid then tried to head in.

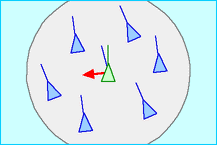
****

Figure 3. Alignment: steer towards the average heading of local flock mates.

**Code:**

/\*

Function: Alignment

Parameters: void

Returns: Alignment vector

Description: Calculates the Alignment vector of the boid

\*/

Vec3 Boid::Alignment()

{

Vec3 align( 0.0f, 0.0f, 0.0f );

int count = 0;

Boid\* boid;

float dist;

Vec3 diff;

for( int i = 0; i < \_neighborCount; ++i )

{

boid = \_neighbors[i];

diff = boid->\_currPos - \_currPos;

dist = Length( diff );

if( dist > 0.0f && dist < \_boidRadius )

{

align += boid->\_vel;

++count;

}

}

if( count > 0 )

{

align /= (float)count;

if( Length(align) > BOID\_MAX\_VEL )

{

normalize( align );

align \*= BOID\_MAX\_VEL;

}

}

return align \* WEIGHT\_ALIGNEMENT;

}

**Cohesion**

A flock is defined as a group of boids all staying close to each together, and the cohesion component of the algorithm is mainly responsible for the togetherness aspect of this. Every frame, each boid looks at the position of each other boid to see if it is within a specified NEIGHBOUR\_RADIUS, that is, it checks to see which other boids are close enough to be considered flock mates. The positions of the qualifying neighbors are averaged and the boid steers to towards that position. This way, each boid is trying to steer towards the center of the flock, resulting in them all staying close together.

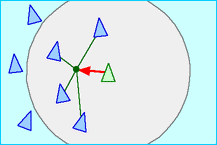


Figure 4. Cohesion: steer to move toward the average position of local flock mates.

**Code:**

/\*

Function: Cohesion

Parameters: void

Returns: Cohesion vector

Description: Calculates the Cohesion vector of the boid

\*/

Vec3 Boid::Cohesion()

{

Vec3 target( 0.0f, 0.0f, 0.0f );

int count;

Boid\* boid;

float dist;

Vec3 diff;

for( int i = 0; i < \_neighborCount; ++i )

{

boid = \_neighbors[i];

diff = boid->\_currPos - \_currPos;

dist = Length( diff );

if( dist > 0.0f && dist < \_boidRadius )

{

target += boid->\_currPos;

++count;

}

}

if( count > 0 )

{

target /= (float)count;

target = Steer( target );

}

return target \* WEIGHT\_COHESION;

}

/\*

Function: Steer

Parameters: void

Returns: Steer vector

Description: Calculates the Steer vector of the boid which moves the boid in the desired direction

\*/

Vec3 Boid::Steer( Vec3 &target )

{

Vec3 desired = target - \_currPos;

float len;

float dist = Length( desired );

if( dist > 0.0f )

{

normalize( desired );

if( dist < 100.0f )

{

desired \*= BOID\_MAX\_VEL \* dist / 100.0f;

}

else

{

desired \*= BOID\_MAX\_VEL;

}

desired -= \_vel;

len = Length( desired );

if( len > BOID\_MAX\_VEL )

{

normalize( desired );

desired \*= BOID\_MAX\_VEL;

}

return desired;

}

return Vec3( 0.0f, 0.0f, 0.0f );

}

With these three simple rules, the flock moves in an extremely realistic way, creating complex motion and interaction that would be extremely hard to create otherwise.

**Variations**

The basic model has been extended in several different ways since Reynolds proposed it. For instance, Delgado-Mata et al. extended the basic model to incorporate the effects of fear. Olfaction was used to transmit emotion between animals, through pheromones modeled as particles in a free expansion gas. Hartman and Benes introduced a complementary force to the alignment that they call the change of leadership. This steer defines the chance of the boid to become a leader and try to escape. Hemerlijk and Hildenbrandt used attraction, alignment and avoidance and extended this with a number of traits of real starlings: first, birds fly according to fixed wing aerodynamics, while rolling when turning (thus losing lift), second they coordinate with a limited number of interaction neighbors of 7 (like in real starlings), third, they try to stay above a sleeping site (like starlings do at dawn) and when they happen to move outwards the sleeping site, they return to it by turning, fourth, they move at relative fixed speed. The authors showed that the specifics of flying behavior as well as large flock size and low number of interaction partners were essential to the creation of the variable shape of flocks of starlings.

## Measurement

Measurements of bird flocking have been made using high-speed cameras, and a computer analysis has been made to test the simple rules of flocking mentioned above. It is found that they generally hold true in the case of bird flocking, but the long range attraction rule (cohesion) applies to the nearest 5-10 neighbors of the flocking bird and is independent of the distance of these neighbors from the bird. In addition, there is an anisotropy with regard to this cohesive tendency, with more cohesion being exhibited towards neighbors to the sides of the bird, rather than in front or behind. This is no doubt due to the field of vision of the flying bird being directed to the sides rather than directly forward or backward.

Another recent study is based on an analysis of high speed camera footage of flocks above Rome, and uses a computer model assuming minimal behavioral rules.

## Algorithmic complexity

In flocking simulations, there is no central control; each bird behaves autonomously. In other words, each bird has to decide for itself which flocks to consider as its environment. Usually environment is defined as a circle (2D) or sphere (3D) with a certain radius (representing reach).

A basic implementation of a flocking algorithm has complexity O(n2) - each bird searches through all other birds to find those who falls into his environment.

Possible improvements:

* bin-lattice spatial subdivision. Entire area the flock can move in is divided into a large number of bins. Each bin stores which birds it contains. Each time a bird moves from one bin to another, lattice has to be updated.
  + Example: 2D(3D) grid in a 2D(3D) flocking simulation.
  + Complexity: O(nk), k is number of surrounding bins to consider; just when bird's bin is found in O(1).

Lee Spector, Jon Klein, Chris Perry and Mark Feinstein studied the emergence of collective behavior in evolutionary computation systems.

**Applications**

In Cologne, Germany, two biologists from the University of Leeds demonstrated a flock like behavior in humans. The group of people exhibited a very similar behavioral pattern to that of a flock, where if 5% of the flock would change direction the others would follow suit. When one person was designated as a predator and everyone else was to avoid him, the flock behaved very much like a school of fish.

Flocking has also been considered as a means of controlling the behavior of Unmanned Air Vehicles (UAVs).

Flocking is a common technology in screensavers, and has found its use in animation. Flocking has been used in many filmsto generate crowds which move more realistically. Tim Burton's *Batman Returns* (1992) featured flocking bats, and Disney's *The Lion King* (1994) included a wildebeest stampede.

Flocking behavior has been used for other interesting applications. It has been applied to automatically program Internet multi-channel radio stations. It has also been used for visualizing information and for optimization tasks

Social network simulation and modeling opinion flow. After choosing humans as the entities in the flock, the overall direction of the flock can be estimated using the rules that apply to the simple flock model, and people’s future opinions can be predicted.

Distributed systems analysis, search, and optimization. By modeling things like spatial data, network traffic, or solutions to an optimization problem as entities, the direction of the flock can be used to find clusters, where to push traffic, or optimal solutions.

**References:**

Web Sources:

[Reynolds86] Reynolds, Craig., "Boids", available online at http://www.red3d.com/cwr/boids/, 1986

[Reynolds87] Reynolds, Craig., "Flocks, Herds, and Schools: A Distributed Behavioral Model in Computer Graphics, 21(4) (SIGGRAPH '87 Conference Proceedings) pages 25-34.", available online at http://www.red3d.com/cwr/papers/1987/boids.html, 1987

[Cole] Cole, Skip., "Modeling Opinion Flow in Humans Using Boids Algorithm & Social Network Analysis", available online at http://www.gamasutra.com/view/feature/1815/modeling\_opinion\_flow\_in\_humans\_.php,

# [Anonymous] "Flocking", available online at http://en.wikipedia.org/wiki/Flocking\_%28behavior%29

# [Gurfil05] Gurfil, P., "Evaluating UAV flock mission performance using Dudek's taxonomy", available online at http://ieeexplore.ieee.org/xpl/freeabs\_all.jsp?reload=true&arnumber=1470734, 8-10 June 2005

# [Labonte09] Labonte, G., "Canadian artic Sovereignty: Local intervention by flocking UAVs", available online at http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=5356552&url=http%3A%2F%2Fieeexplore.ieee.org%2Fiel5%2F5351161%2F5356514%2F05356552.pdf%3Farnumber%3D5356552, 8-10 July 2009

[Folino02] Folino, Gianluigi., and Spezzano, Giandomenico., **"**An Adaptive Flocking Algorithm for Spatial Clustering**",** Lecture Notes in Computer Science, 2002, Volume 2439, Parallel Problem Solving from Nature — PPSN VII, Pages 924-933, available online at http://www.springerlink.com/content/c7t0fb6a54flkrw8/?MUD=MP, 2002

--