Crowd Simulation: Boids++

Boids Framework — A Modern take on Craig Reynolds' Boids.

DI ZHAO, University of Califonia, Davis
MUHAMMAD OSAMA, University of Califonia, Davis

Additional Key Words and Phrases: Crowd Simulation, Boids, Computer Animation

1 INTRODUCTION

Craig Reynolds program, Boids[Reynolds 1987], illustrated how flocks, herds and schools can be animated as artificial life using three simple rules. The word 'boid' originated from 'bird-oid object', which means 'bird-like object'. The simulation that Craig proposed follow the following three simple rules:

- **separation**: as the name suggest, this simply means to avoid other local objects within the simulation.
- **alignment**: is aligning the direction of the local objects to the average heading of surrounding objects.
- **cohesion**: is movement towards the average position, which in a crowd is often the center of mass of the local objects.

These three rules, very simply and beautifully modeled the flock, herds and schools animation. The contribution of our project is to extend the Boids model, effectively called **Boids++**, to include more complex rules and to provide a framework and abstraction to define other custom rules not accounted for.

2 RELATED WORKS

Simulating a crowed boids has a long history in computer animation. At the SIGGRAPH conference in 1985, Susan Amakraut, Michael GirardSusan Amakraut, and Michael Girard from Ohio State University revealed a small piece of animation "Motion studies for a work in progress entitled 'Eurythmy'". [Caldwell 1989]In their animation, several birds flying from a birdcage, crossing several cylinder obstacles and then landing on the ground. Each bird has its winds waving when flying and makes sure they don't hit each other. They call their work Eurythmy Force System which is a 3D vector based acceleration force that allows an object to move from one point to another. There is also another repellent force that avoids collision of flockmates and obstacles. The force is linear and developers only need to define the force and computers can simulate all complicate motions based on that.

Craig Reynolds' work has been extended in several ways since 1986. The basic model had rules added on top of that, which mapped things like emotions between objects [Delgado-Mata et al. 2007], these emotions could be an attraction factor or fear. An introduction of "change of leadership" was presented by Hartman and Benes[Hartman and Benes 2006].

The Media Lab of MIT also has several related motion animation research. But they are not building the system by group control but distribute models instead.[Kim 2003] Another interesting example similar to this is a popular game software called 'Creature', revealed in 1996[Wikipedia [n.d.]]. By 1999, there are more than 500,000 games been sold. The game developer Cliff D. and Grand S. make an interesting discussion about why this game is so popular and foreseeing a future of game animation and AI in their paper. The strategy of 'Creature' is maintaining a networking system in each agent(artificial object), and the agents are influenced via this network with each other under an inheritance algorithm[Cliff D. 1999].

Particle System also has similarity to crow simulation. Both two are describing a large scale number of objects' movements and each element is moving unpredictably. Particle System focuses more on phenomenons like fire, smoke, cloud, sand, etc. They are all combined with a large number of small particles but each has its unique identity. The particle can form an overall look of the system by changing its color, position, transparency, speed, etc. But the boids system is more complex than this since particle could not have a complicate motion like an animal. Also, quantum does not have a shape and individual motion. So it could be defined as a 2.5D system. Although some software like MAYA and XSI [may [n.d.]]could further implement a physical object, these designed objects still unable to form a further change like shape distortion.

To be more specific, in MAYA, collisions could not be implemented between particles, so only force field model could be implemented to keep a distance between each particle. With the development of this system, more approaches like AI are also implemented for perfection. For example in 'MUMMY RETURNS'[MR [n.d.]], an AI particle simulation is designed. Several core ideas could be referred and adapted to crowed simulation as well.

3 PROPOSAL

Using the very intuitive Unity game engine, we intend to design the first three simple rules of Boids simulation for starters. This is work already in progress by Di Zhao. We then would like to write an abstraction that can be called to represent rules for two major systems within the Boids simulation:

- **Local Animation and Rules**: describes the local movement of a boid within its own model. This can easily be mapped, animated through Blender and imported into the Unity scene we intend to design. And can encapsulate a wide range of animations and models, including simple models such as birds, fish, but also some complicated ones, such as four-legged animals like dogs or wolves, simulated as a "pack" within Boids.
- **Global Rules**: these are additional rules we intend to add within our systems. Some of the ones we have considered include the idea of a "first follower" [Sivers [n.d.]] who may have the ability to increase the search radius of a crowd within Boids and "attract" other objects to the crowd, we have also

Authors' addresses: Di Zhao, University of Califonia, Davis, devzhao@ucdavis.edu; Muhammad Osama, University of Califonia, Davis, mosama@ucdavis.edu.

considered a "coup", which depicts splitting of the Boids animation and introducing repulsive force between the splits based on some random factor. And we would also like to write a clean abstraction to add future rules if we find them interesting.

On the technical end, we are interested in looking at fast k-Nearest Neighbor (kNN) searches on the GPU and including that as our shader program to speed-up our computation. The kNN here is used for several different scenarios, including but not limited to finding other Boids to attract or avoid.

4 IMPLEMENTATION

The core idea of boids is put forward by Craig W. Reynolds in 1986 [Reynolds 1987]. There is no need to implement a leader which commands each boid for maintaining the flock. Each individual boid only need to follow the 3 principles: alignment, cohesion, and separation.

To simulate the aggregate motion, for each boid, we iterate and collect both position and velocity (speed and direction) information of all other flock-mates to calculate average velocity, average distance to other flock-mates, and central position of this flock.

4.1 Alignment

The average velocity is used for alignment; is a 3D vector with two parameters average heading and average speed. Each boid adjusts its heading towards the average heading of the whole flock so that the whole flock is always heading in the same direction. It also checks its speed with average speed. If the current boid is too fast, it slows down waiting for other flock-mates catching up, otherwise speeds up. The more important part of the alignment is the heading match process. We notice that the speed alignment does not need to be specifically implemented since the separation process would be used to adjust speed instead. We set a minimum and a maximum speed for the flock and use Clamp function to adjust the speed. Heading is more important because it directly results in whether it's jerky or smooth when the flock makes a turn. To achieve this, the following equation is implemented:

$$v_j = (\frac{1}{n}\sum_{i=1,i\neq j}^{n} v_i) \cdot S + v_j \cdot (1 - S)$$

Where v_j is the velocity of current boid, n is the number of boids in the flock. S is a parameter set to control the boid motion. Its value varies from 0 to 1. A larger S means the changing vector takes more advantage than the current status so that the flock is more tending to change. The default setting of S is 0.5. Thus, the changing vector is simplified to $v_{average} - v_j$. This change will attach to each boid by adding a force in the same direction as this changing vector. A weight value w_a is set to determine the force strength. The final force add to the boid would be like this in our project implementation:

$$f_j = (v_{avg} - v_j)_{magnitude} \cdot w_a$$

The whole process mentioned above is processed every frame for all boids.

4.2 Cohesion

Flock center position is used for cohesion. A force to the center of the flock is always added to each boid. Note that when flock crosses an obstacle, the boids may be divided into 2 or more parts then concentrate together again after then. In this case, there is no need to consider the center of each cluster when the flock is temporarily divided. Otherwise, the obstacle process of this may look like one flock initiated divide into two or more instead of an obstacle come across a single flock. The cohesion equation is:

$$v_j = ((\frac{1}{n}\sum_{i=1,i\neq j}^{n} pos_i) - pos_j) \cdot S \cdot w_c + v_j \cdot (1 - S)$$

Here pos is the position of a boid, w_c is the weight of cohesion. We could see that the further boid[j] away from the center of the flock, a larger force will be implemented.

4.3 Separation

Average boid distance is used for separation. We set a collision test for each boid in the flock to make them separate. Each boid has a private zone and we check whether other flockmates entered this zone. If so, a separation force is added each frame until leaving. The equation is like the following:

$$v_j = (\frac{1}{n}\sum_{i=1,i\neq j}^{n} \frac{v_i}{dist(pos_i, pos_j)}) \cdot S \cdot w_s + v_j \cdot (1 - S)$$

$dist(pos_i, pos_j)$ is distance vector from $boid_i$ to $boid_j$. The further $boid_i$ away from $boid_j$ the less influence will be added for separation. The private zone mentioned above is set to avoid unnecessary calculation. We consider both the position and velocity of other flock mates in this process. We notice that if only the position of two boids is considered, the flock could not be separated well in some situations (e.g. When boids are too close to each other at the beginning). This is also used for the speed matching process that we mentioned in the alignment section. A separation weight parameter w_s is also set for separation force.

The S parameter from 3 equations have the same function and set to be the same in our implementation but it does not necessary to be the same. The 3 weight value are all set to be 1, it is does not make a significant change to the output when the value changes. We believe they are not necessary parameters. Recall the algorithm mentioned in this section is only implemented to each boid and no other general control system is added. This is built to stick to the natural flock condition.

5 OPTIMIZATION

Based on the 3 rules we further implement other optimizations of our own, and extend the model to include more complicated rules as proposed in section 3.

5.1 Performance

Iterating all flock mates for each boid per frame is too expensive and may cause a lot of lag or system crashes. We took Sebastian Lague's project as a reference and implemented a GPU based multi-thread iteration for calculating the three key parameters mentioned

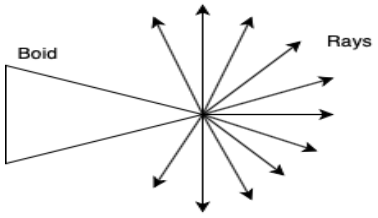
above. Each boid is assigned a single thread on the GPU and all boids run parallel on GPU and iterates simultaneously with global synchronization at the very end. Since each boid is independent of other boids, and do not update other boids in any way, the problem becomes embarrassingly parallel and very efficient on the GPU.

5.2 Realistic

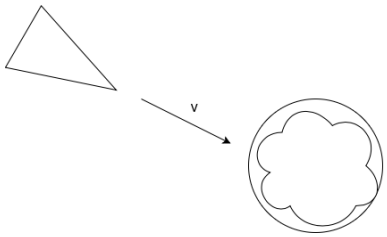
We also add a random force to each boid every 20 frames. In some situation(e.g. When the flock move straight forward for a time), the boids make no change and cause the whole flock look too ordered and becomes even sterile. So an uneven force with random direction is added to add realism to the flock motion. This force is not designed to completely break the flock order and reform again but to downgrade the precision of computer calculated simulation and make the motion more realistic. We also tried to implement a gravity to the system, but we found that this had almost no effect to the system in play.

5.3 Avoid Obstacle

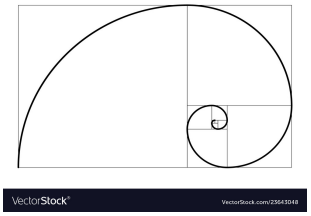
The interaction with the obstacle object is the most interesting part of the whole simulation. It makes the flock motion more random and unpredictable when it comes across an object with complex shape. Sometimes the boids further divide into several smaller groups but the whole motion looks still realistic and in later timesteps you can observe them forming back into one single flock.



The strategy to achieve this is complicated. Craig W. Reynolds mentioned two different attempts: force field model and steer-to-avoid, and we implemented the latter one since it has a better performance. We designed several rays sending outwards from the boids to detect if there are obstacles ahead. The challenge is detecting the collision of rays on the surface of an object could be very expensive if the object is polyhedral. It turns to be even more complicated to further figure out the direction for a boid to avoid the object, especially in a 3D environment. We simplify this question by casting the obstacle into a sphere. In this case for each ray sending from the boid, the obstacle could be measured as a circle on a 2D panel with this ray. With this casting process, we solve 2 problems together: the complexity of directions in 3D and arbitrary shape of obstacles.



When an obstacle is detected, a normalized force will be added to the boid on the same panel of the detection ray. This force is designed to turn the heading of the current boid and the angle it turns will be based on the golden ratio. In some simplified cases, for example, if a boid heading perpendicular to a flat surface, the boid will steer a golden spiral to avoid a collision. This is designed to avoid sharp turns and make the whole process smooth and natural. Since the flock is still maintaining during the avoid process, the avoid force is added based on the 3 rules and it is not necessary to worry about whether the force from the 3 rules may affect the performance of avoiding.



5.4 Prey and Predator

To test the durability of our system, we introduced a prey and predator into our system to simulate a real natural scene.

The prey randomly picks a point in the 3D scene and move toward that point. To make the motion of prey smooth and real, we implement a force from the position of prey to the target point on prey each frame. When the prey reaches the target, another point will be randomly picked. The flock chases the prey in a similar way. A force heading the prey is added on every boid each frame together with 3 rules. To make the scene more simple we set the speed of prey always greater than the boids so the flock never catches up.

We also added a predator to chase the flock. The predator is always heading the center of the flock and the strategy is exactly the same as boid chasing prey. Each boid gets an updated predator position each frame. If the distance of predator is less than an avoid radius an escape force will be added toward an opposite direction of the predator on boid. The difference between avoiding predators and obstacles is the boids do not follow three rules when escaping predators. We notice that three rules should not be implemented to simulate avoid predators since alignment and cohesion always playing a dominant role in boid motion. Even we set an escape weight and set it 10 times more than the weights for three rules, the flock still does not show a distinct escaping motion while predator approaches. In our implementation, we only reserve the separation part and set a significantly higher speed to boid, in order to match a similar escaping scene from the natural world.

5.5 Local Animations

To add to the realism, we are able to import any custom blender model into our 3D scene. So the same simulation can be used for birds or fish or any sort of flock motion, and also incorporate handcrafted local animations for each model while animating the larger flock. Since our system uses a GPU to process some of the more compute intensive tasks, it is easily able to handle the minute local animations that further add realism.

There's also further research needed to automate such realism for a flock simulation, similar to what have been done for human crowd simulation with the little random arm and shoulder gestures.

6 CONCLUSIONS

We develop a framework that extends Craig Reynolds' [Reynolds 1987] boids simulation for simulating real flock like movements. We took the basic boids principles and extended it to show more complicated set of rules, used the GPU to introduce performance optimizations to better simulate the problem, and further expanded on the realism by adding local animations to the boids within the flock. Our work is available on the github repository found at https://github.com/neoblizz/boids.

REFERENCES

MAYA [n.d.]. . MAYA. https://knowledge.autodesk.com/support/maya/learn-explore/caas/CloudHelp/cloudhelp/2015/ENU/Maya/files/GUID-2AB4F791-4F12-4BCA-BA13-7A2F128FF3F4-htm.html

FANDOM [n.d.]. . FANDOM. https://mummy.fandom.com/wiki/The_Mummy_Returns_(Video_game)

Ph.D. Caldwell, Craig Bemreuter. 1989. *A physically-based simulation approach to three-dimensional computer animation.* OSU.

Grand S. Cliff D. 1999. The Creatures global digital ecosystem. *NCBI* (1999).

Carlos Delgado-Mata, Jesús Ibáñez-Martínez, Simon Bee, Rocio Ruiz-Rodarte, and Ruth Aylett. 2007. On the Use of Virtual Animals with Artificial Fear in Virtual Environments. *New Generation Comput.* 25 (02 2007), 145–169. https://doi.org/10.1007/s00354-007-0009-5

Christopher Hartman and Bedrich Benes. 2006. Autonomous boids: Research Articles. *Computer Animation and Virtual Worlds* 17 (07 2006), 199–206. https://doi.org/10.1002/cav.123

Hana Kim. 2003. Multimodal animation control. *MIT* (2003).

Craig W. Reynolds. 1987. Flocks, Herds and Schools: A Distributed Behavioral Model. *SIGGRAPH Comput. Graph.* 21, 4 (Aug. 1987), 25–34. https://doi.org/10.1145/37402.37406

Derek Sivers. [n.d.]. *First Follower: Leadership Lessons from Dancing Guy.* Youtube. https://www.youtube.com/watch?v=fW8amMCVAJQ

Wikipedia. [n.d.]. *Creatures (1996 video game).* wiki. https://en.wikipedia.org/wiki/Creatures_(1996_video_game)