**Project Report** **Wenhan Zhu (Cosmos)**
CS 688

# Introduction

This project is a simulation that mimics agent moving behaviors for some modern RTS (Real-time strategy) games. The main techniques that some of the modern RTS games uses is to treat agents as boids. An artificial life that mimics the life of birds and fish moving together.[Reynolds]. Also a potential map is implemented to provide very basic navigation around the map.

# Implementation

There are 3 major parts of the simulation. The map, the agents and the potential map. They interact with each other and I will explain how they work when I go through their implementation details.

## Map

The map holds information about where all the objects are. There are 2 major things on the map. The first one is scene objects. They are stationary objects on the map that needs to be avoided by the agents when moving. In the simulation, there are 2 types of scene objects. One is the water fountain that is to implemented to demonstrate a particle system. Another is a scene object that loads an texture from a local file. For this scene object, there is an extra attribute that determines which part of the loaded texture is a obstacle. This makes loading textures with half passible parts possible. The second type of information the map holds are the agents. When a command is issued to the agents, the map will update the agents with the information they need.

## Agents

Agents in the simulation are implemented as boids. Boids move around with 3 type of forces interacting with each other. Separation force will keep the boids from colliding with each other. Alignment force will keep a group of boids moving at the same direction. And cohesion forces will keep boids together. There is an active state for each of the agents in the simulation. When agents are inactive, the only force they interact with each other is the separation force. So when an active agent is moving across inactive agents, the inactive agents will not be affected by the movement of the active agent. When a command is issued to a group of agents. They will be assigned a special id that indicates they are of the some group. Then the map will calculate the potential field for this group of agents. The group of agents will following the potential field to it's destination.

There are 2 types of agents in the simulation besides the very basic one. The first type is the AnimationBoid. This agent type takes in a texture map of a top down character and updates the texture based on the time and speed of the agent. As a result it creates an animation when the agent is walking. The other type of agent is a flying agent. The flying agent will not be affected by terrains on the ground or agents on the ground. It rendered a line pointing from the agent to the ground to indicate it is a fling agent. It's location is relative to it's ground position. It is clearly indicated when selecting the agent.

## Potential Field

The potential field generates map for the agents to navigate around the map. When a destination is assigned, for each block on the potential field, it will be assigned a value based on the distance it's from the destination. The more close it's to the destination, the higher the value is. Then each obstacle on the map it will create a negative value around it and decrease the potential field value and each block. As a result, at any point on the field, agent only needs to move to the highest value of the field near the agent.

There are also some other parts of the simulation that is worth mentioning.

## Agent Navigation

When agents are moving around the map not reaching the destination, the boid movement are a great fit. However, since the agents are moving in groups it hard to determine when to stop. For example, when a one of the agents in the group reaches the destination., if we set it to stop, the other agents will continue to move to the destination and make the first agent to move. This kind of works when the entire group are moving together. However, if we were to move a group of agents to one place, one of the agents arrive at a later time than the other. It will break the entire group to move to the center. To solve this problem, I implemented the active state mention earlier. When a agents reaches it's destination it's active state will set to false. Then other agents in the same group will set it's state to inactive when half of the agents near them are inactive. With this way, in the earlier situation, the agents that arrives late will stop at the edge of group and not break the entire group since all the earlier agents are set to inactive.

## Boid physics

As mentions earlier, there are 3 major forces acting on each of the agents. However, due to the purpose of the simulation. It is essential to tweak the parameters so that when the agents moving they will not collide with each other and when they are not moving they will separate to a distance. Also, the force that generates from the potential field provides the navigation direction should not be affected. It is very hard to say how much effort was put in to calculate the forces. However, the nice thing is the results looks pretty decent. There could be improvements as the result I have is still far away from what modern RTS games looks like.

This covers almost everything related to my final project. There are some markdown documentation inside the docs folder. More details can be find there.

# Future improvements

Current the simulation creates the movement of agents. It would be nice if the simulation can be extended to a full RTS game such that it can build buildings, produce armies and fight enemies. There are also some technical details that can be improved. Currently the agents interact with each other by determine the distance from center to center. I've tried to implement a version that test not the center to center distance but the actual distance. However, due to the agents having a physical model, it will break the simulation some times when 2 agents move to close and have a negative distance of each other. It will be nice if it can be tweaked to work in the future.

# Limitations

There are some limitations to the simulation, first is that the graphics are only 2D. I only realized the graphics package was 2D only when I was way down the road. I learned that it's capable of calling OpenGL commands but I found out it takes too much work to get a decent looking model. So instead of switching to a 3D package. I figured I will try some 2D graphics techniques. One of such is the sprite animation which was mentioned earlier.

Using multi-agent potential Fields in Real-time Strategy Games, Hagelback,et al. Proc. of 7th Int. Conf. on Autonomous Agents and Multiagents Systems (AAMAS 2008), Padgham, Parkes, MullerandParsons(eds), May, 12-16, 2008, Estoril, Portugal, pp. 631-638

# References


[1] Reynolds, Crag W. *Flocks, Herds, and Schools: A Distributed Behavioral Model.* SIGGRAPH '87, July 1987, pp. 25-34.

[2] Hagelback, et al. *Using multi-agent potential Fields in Real-time Strategy Games* Proc. of 7th Int. Conf. on Autonomouse Agents and Multiagent Systems (AAMAS 2008), Padgham, Parkes, MuulerandParsons(eds), May, 12-16, 2008, Estroil, Portugal, pp. 631-636


# Reference report

The main 2 reference that I end up using are the one from the original idea of boids and the other is the implementation of potential fields.

[Reynolds] In Reynolds paper, he introduced the Boids the artificial life that mimics the behavior of flocks. It is a great way to model group behavior. Related to his original works.

The idea has been used in many different fields. One of them is crowd behavior. And in recent RTS games, the results of boids in crowd behavior created some of the best agent moving behaviors in my opinion. In this simulation, I implemented boids following this paper's ideas and did some tweaking to fit my purpose.

[Hagelback] This paper focuses on how to generate a potential field for navigation. Since my intention of this project is mainly on the agent behavior side. So I chose to implement the simplest version of the potential field in this paper. I find this method of generating navigation for agents very simple and the calculation only needs to be done once when combined with boids movement.