Aalto University
School of Electrical
Engineering

ELEC-A7150 C++ Programming

Steering Behaviours For Autonomous Characters

Project plan

Professor: Pasi Sarolahti

Project Assistant: Klaus Helenius

Students: Jorge Bodega - 642604
Sergio Ehijo - 645630
Marc Vila - 645591
Oriol Villamayor - 645656

Date: November 17th, 2017

1. Introduction

This project is about the simulation of the behaviour of determined autonomous characters in a certain environment. These characters will move governed by different policies or rules interacting ones with each others, where these rules might be applied for groups or for just individuals.

The idea behind this project lies with “Simple vehicle model” written by Craig Reynolds, which is and will be the basic pillar for the code written. All of the code will be developed in C++ (which is the scope of this course) and also a graphical user interface (GUI) known as SFML will be used, in order to give the user a better experience, allowing him the ability to choose different laws to govern the groups, if wanted.

The first idea to realize what autonomous characters can be is to think about fishes. In this way, there would be different species with different kind of interactions inside their own bank and also with other banks or single fishes. For this, for example, a certain fish would like to eat something, giving it the ability to go faster to the goal or as it approximates its goal, slow down, which are different examples of steering for the simple vehicle model presented in Reynolds’ paper. [1]

2. Scope and goals of the project

The aim of the project is to make an implementation of the boids flocking simulation. To accomplish that, a world will be created where these characters can move around randomly, and where the user can have a degree of what to choose in certain circumstances.

First of all, in this world there will be only to types of characters; the ***boids*** (the ones that moves around behaving like e.g. fish) and the ***blocks*** (objects that have a specific location all the time and don’t let the boids go through them). However, the main part of the project is to simulate the behaviour of the boids. So, in other words, they will have some characteristics and rules that will make them interact with the world. To get to it, some basic rules must be listed here. These rules and goals contemplate the basic requirements asked for the project; as soon as all of them are tested and working correctly, some extra rules and additions are written that will improve the project making it more enjoyable.

The world will be 2D. The principal characteristics of the boids should be:

- **Position:** two coordinates (x,y) that tell where the boid is.
- **Speed:** a vector.
- **Acceleration:** another vector.
- **Range of view:** every boid has a radius around it, and all the boids in it are its friends. It can see theirs position, speed and acceleration. Also the blocks can be in it.
- **Shape:** For the block is very simple, it will have a position, and a shape.

The **basic** rules that this project must follow are:

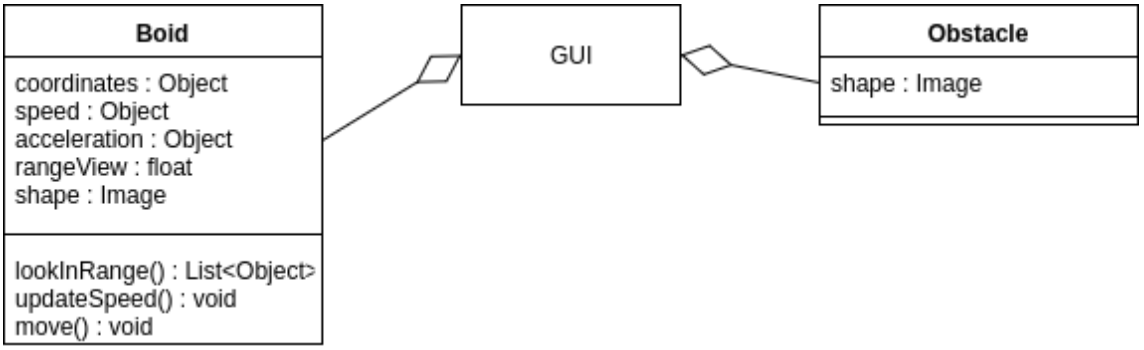
- **Avoid the obstacles:** when a boid has an obstacle in its range of view, it will change its direction to avoid it.
- **Alignment:** the boid will look the directions of its friends, it will make and average so it will move in a similar direction.
- **Cohesion:** the boid will look the positions of its friends, it will make and average so it will move towards it.
- **Crowding:** if a boid gets too close to all its friends, it goes away to avoid crowding. So this rule balances with the cohesion one, making the boid want to go near its friends, but not to much. [2]

On the other hand, some of the extra work that as a group would be excellent to achieve is:

- **Extra rules:**
  - **Noise:** boids movements are smoother.
- **Extra characters:**
  - **Reward:** the opposite of the obstacle, when a boid sees a reward, it goes directly to it. Its interesting because the intelligence of the boid should be enough to realize when an object is whether an obstacle or a reward..
  - **Species of boids:** boids that only follow the rules of cohesion, alignment and crowding with the boids of its species.
  - **Bad boid:** one species of boid that other species of boids want to avoid, so it's like a moving obstacle, but they want to be with other bad boids.

4. Architectural design

Thinking the hierarchy of classes as simple as they can be, it should be something similar to figure 1, in which there are at first three classes: one for the design of all the GUI, one for the creation of a simple vehicle object, and another one for the creation of blocks or zones which have to avoided by the individuals. The methods will not be explained deeply now because clearly with more time a deeper understanding of the problem of the project will be held and hence a much more clearer idea of what methods should go in each class and how it works. The design clearly looks like a Composite, in which different collections of vehicles and blocks references will be present in the GUI class.



5. Roles distribution

The distribution of the different roles that this project need are basically the following:

- Graphical User interface (GUI): ***Marc and Oriol.***
  This means both of them are in charge of understanding the basics of SFML.
- Working simple vehicle model: ***Jorge and Sergio.***
  Both of them are in charge to make the different algorithms present in the papers and explained previously in this document as needed for the minimal requirements.
- Group moving policies implement: ***Jorge and Sergio***
  Both of them are in charge to make the different algorithms present in the papers and explained previously in this document as needed for the minimal requirements.
- Combining SFML with C++: ***Everyone***
  At this stage, everyone is supposed to make it easy for other group participants how to connect both blocks (GUI with C++).
- Group leader: ***Sergio***

In charge of assigning and checking that the work is going as planned by the project timeline.

6. Project timeline

- 12 - 18 of November:
  - Project plan in GIT
  - Everyone knows the problem at hand, start of first iteration of coding and ideas.
- 19 - 25 of November:
  - Meeting with group advisor for feedback and follow-up plans
  - Group meetings. First results of different algorithms
- 26 of November to 1st of December:
  - Mid- term meeting: The goal is to be the best in comparison with the other groups of Steering.
  - Second results of the algorithms, start of combining GUI with C++
- 2nd of December to 14th of December:
  - Finishing the project.
  - Decision of making the extra algorithms
- 7th of December: Group feedback
- 15 dic: Final demonstration

7. References

[1] Craig Reynolds' "Simple Vehicle model" 1999
http://www.red3d.com/cwr/steer/gdc99

[2] http://natureofcode.com/book/chapter-6-autonomous-agents/