ELEC-A7150 C++ Programming

Steering Behaviours for Autonomous Characters

PROJECT DOCUMENTATION

PROJECT ASSISTANT:
Klaus Helenius

STUDENTS:
Jorge Bodega Fernanz (642604)
Sergio Ehijo (645630)
Marc Vila Fábrega (645591)
Oriol Villamayor Abelló (645656)

DATE:
15/12/2017

1. Overview

What the software does, what it doesn't do? (this can be taken/updated from the project plan).
The initial idea of this project was to simulate the behaviour of determined autonomous characters in a certain environment. These characters would move governed by different policies or rules interacting ones with each others. These rules might be applied for groups or for just individuals.
For now, the software is capable of doing the following things:

- Add new boids or blocks
- Clear all boids or blocks
- Erase the last boid/block
- Play music
- Activate/deactivate how the boids interact with the blocks.

Eventhough that satisfies the initial requirements and the scope of the project, there are still some features that the actual program doesn’t handle and we would like to add:

- Some preloaded maps
- Differents behaviours of the boids

2. Software structure

Overall architecture, class relationships (diagram very strongly recommended), interfaces to external libraries.
All the software and files related to this project have been uploaded to a GIT repository since the very first time and they have been updated from there.
The structure of the GIT repository is based on different folders where all the files are grouped according to their type.
These are our folder tree:

- doc
- plan
  - PDF file.
- src
  - Gallery
    - background.jpg
  - music
    - Pokemon.ogg
  - All the “.h” and “.cpp” files

3. Instructions for building and using the software

To run the software the file, it has to be opened from the command window. Using “make main” to compile the main.cpp file, or “make clean” to remove previous compilations. Then, use “./main” and a new window will open showing the main interface of the program. This GUI interface has been configured in order to make the user have the better experience using the program in the easiest way. Everything is really intuitive, but anyway all of the controls availables are going to be explained carefully here.

4. Testing

We tested it using four different computer, two with ubuntu and two with MacOS, trying to do as bad as we can, trying to find some bad behaviours.

5. Work log

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

- *Graphical User Interface:* Marc and Oriol. Around 12 hours of working.
- *Working simple vehicle model:* Jorge and Sergio. Around 10 hours of working.
- *Group moving policies implement:* Jorge and Sergio. Around 5 hours of working.
- *Combining SFML with C++:* Everyone. Around 5 hours.