Lab 1 - Movement & Steering using Raylib
AI Programming for Games
COMP10068

Paul Keir

Date Issued: January 10, 2022
Instructor: Dr. Paul Keir

In this week's lab we aim first to become more familiar with the use of Microsoft Visual Studio to compile and debug elementary C++. The lab material makes use of the Raylib videogame programming library, allowing a simple route to displaying interactive graphics.

As with the Boost code from last week, you will need to edit the CMakeLists.txt file provided in the lab-steering1/code directory using Notepad++. The `CMAKE_TOOLCHAIN_FILE` variable needs to be set to the location of the vcpkg.cmake file within your Vcpkg install. To do this you can either download and unzip the Vcpkg export zip from the AI Programming for Games module's myUWS page; or install it yourself (in this case, the Vcpkg package name is simply "raylib").

After you have used CMake to configure and generate your Visual Studio solution, open it in Visual Studio and you will find three projects.

Steering Zero (steering0.cpp)

1.. Uncomment the three rotation value increments within the while loop

2.. Change the colour or position of the `raylib::Rectangle` object called `rect`

3.. Change the second (`source`) parameter of `DrawRectanglePro` to `{0,0}`. The rectangle then rotates around one of its corners rather than its centre.

4.. What does the `5` in the `DrawPoly` function call do? Try changing it to find out.

Steering One (steering1.cpp)

1.. Add a new circle to the `std::vector` of shapes using `vector::push_back` when the left mouse button is clicked. Use the mouse position as the position of the new shape you create.

2.. Use `int GetRandomValue(int min, int max)` to return a value between `min` and `max`. Use this value to create a random size for the circle you create.

3.. A `raylib::Color` can be created from 3 unsigned characters; one each for red, green and blue. Use `GetRandomValue(0,255)` to get 3 random colour components, and use them to set the colour of each new circle you add.

4.. Declare a bool value called flip outside of the while loop. Initialise it to true. Now: add a circle shape when it is true; and a square shape when it is false. Change its value after you use it with: `circle = !circle;`

5.. Add the following to display the number of shapes which have been added: `DrawText(FormatText("%i", my_shapes.size()), 9, 9, 18, RAYWHITE)`

Steering Two (steering2.cpp)

1.. Add a target for the red ship using a point obtained from the mouse position when the left button is pressed.

2.. Direct the ship towards the target point from the mouse.

3.. We will return to this code next week as we investigate steering algorithms further.