Physical Controller Report

Harry Donovan - HD241610

https://drive.google.com/open?id=1I5zGk7EhjqBN9vi6QOgywfug1ZDW6TGo

Contents

1 Project Proposal **2**
- 1.1 Concept Overview . . . 2
- 1.2 Software Overview . . . 2
- 1.3 Hardware Overview . . . 2

2 The Controller **3**
- 2.1 Hardware . . . 3
  - 2.1.1 Component List . . . 3

3 UML Diagrams **4**

4 Reflection **6**
- 4.1 Concept . . . 6
- 4.2 UML Diagrams . . . 6
- 4.3 Software . . . 6

1 Project Proposal

1.1 Concept Overview

Humanity has been forced into underground bunkers. Automated production facilities ensure an endless bombardment of shells. You are a defence operator, tasked with controlling a defensive turret above the bunker. A majority of the systems are automated and require only limited guidance, with the system breaking the complex hail of debris and 3D trajectories into a more manageable 2D display. Keep us safe for as long as you can; our survival is in your hands.

1.2 Software Overview

The game is inspired by the classic arcade game missile defender. It is a 2D game in which you control the rotation of a turret which is able to fire up at incoming projectiles. The spawning of enemies is controlled by one of several AI modules, with the active module changing as the game progresses.The spawned enemies can take a variety of different AI modules and one is assigned at spawn by the spawn AI. These AI modules are 'hot-swappable', so I may experiment with the enemies AI changing mid wave. I also plan to implement special 'Boss' enemies, as well as enemies with unique abilities.

1.3 Hardware Overview

I will repurpose a Sony ST-SE500 FM-AM tuner for the controller. I chose this controller as it has the needed inputs, and makes sense in-universe as recycled and repurposed equipment. I also originally intend to use the vacuum fluorescent display, however due to how the display PCB is set up this is unfortunately unrealistic. The arduino will be stored within the case, with the cable connecting it to the PC running out of the cable port that was originally for the mains power supply. I plan to repurpose one of the push-to-make buttons and the rotary encoder.

2 The Controller

2.1 Hardware

As stated in my proposal I will repurpose a Sony ST-SE500 Stereo/ FM-AM Tuner. I will be using the case in order to store the arduino, running the cable connecting the arduino to a PC through the cases' original power supply port. The rotary encoder and a single button are also on a separate PCB, so I have cut the wires that connected it to the main board and am instead running them into the arduino. The rotary encoder PCB has mounting holes connecting it to the case which are easily accessible, so I can remount it in its original position once I have it wired up to the arduino.

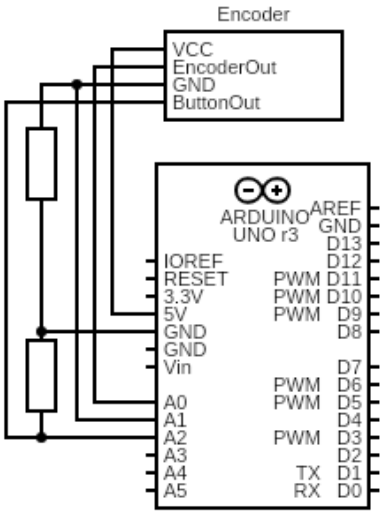

Figure 1: Circuit Diagram

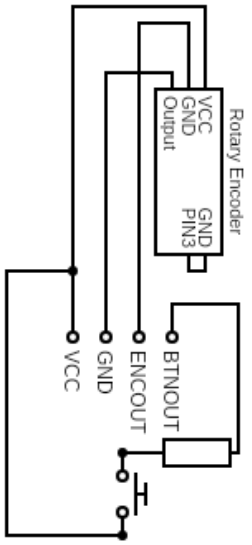

Figure 2: Circuit Diagram

2.1.1 Component List

- Arduino Uno
- Rotary Encoder
- Pull Down Resistor
- Push-To-Make Button

3 UML Diagrams

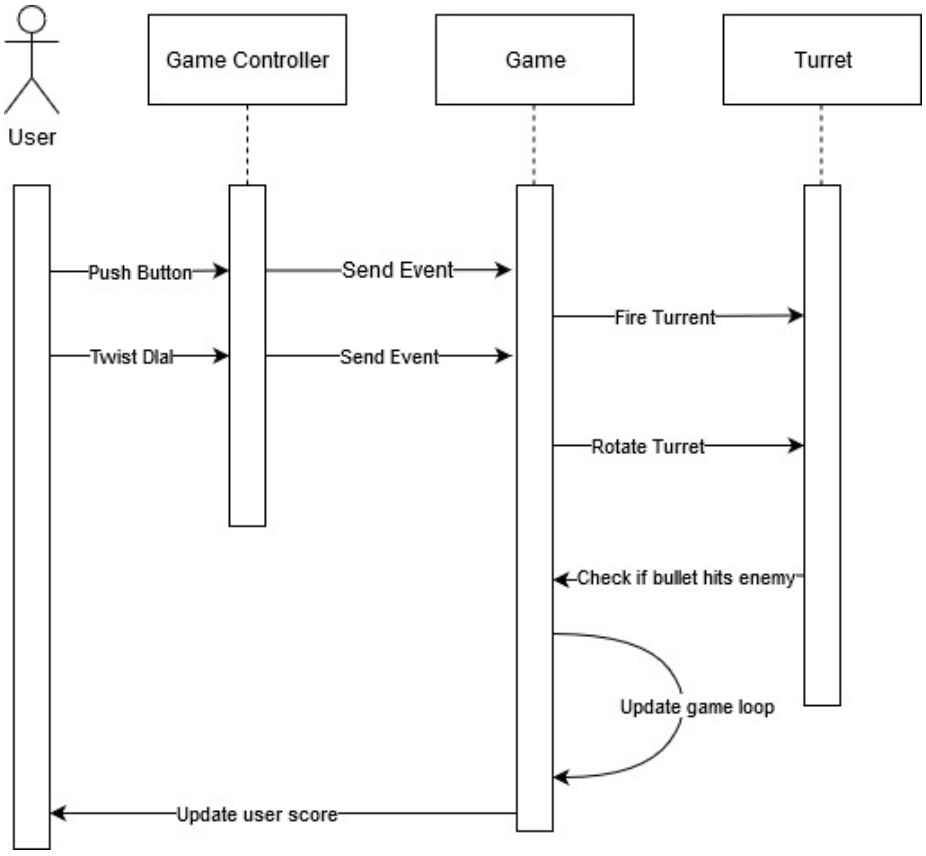

Figure 3: Use Case Diagram

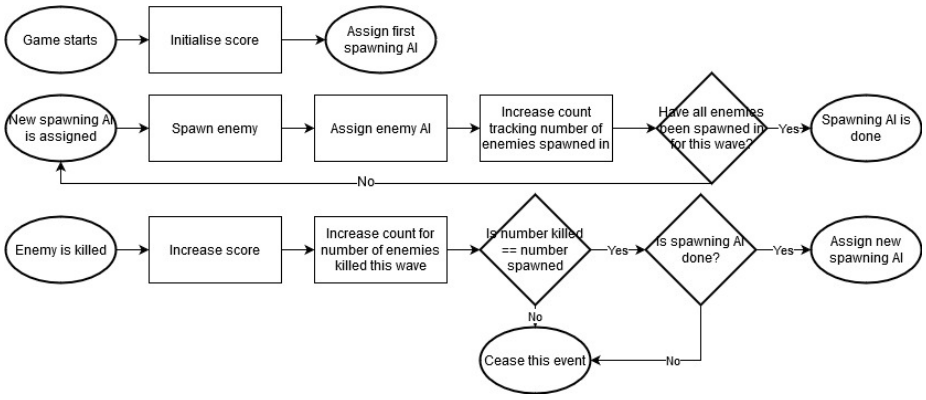

Figure 4: Game Logic Flowchart

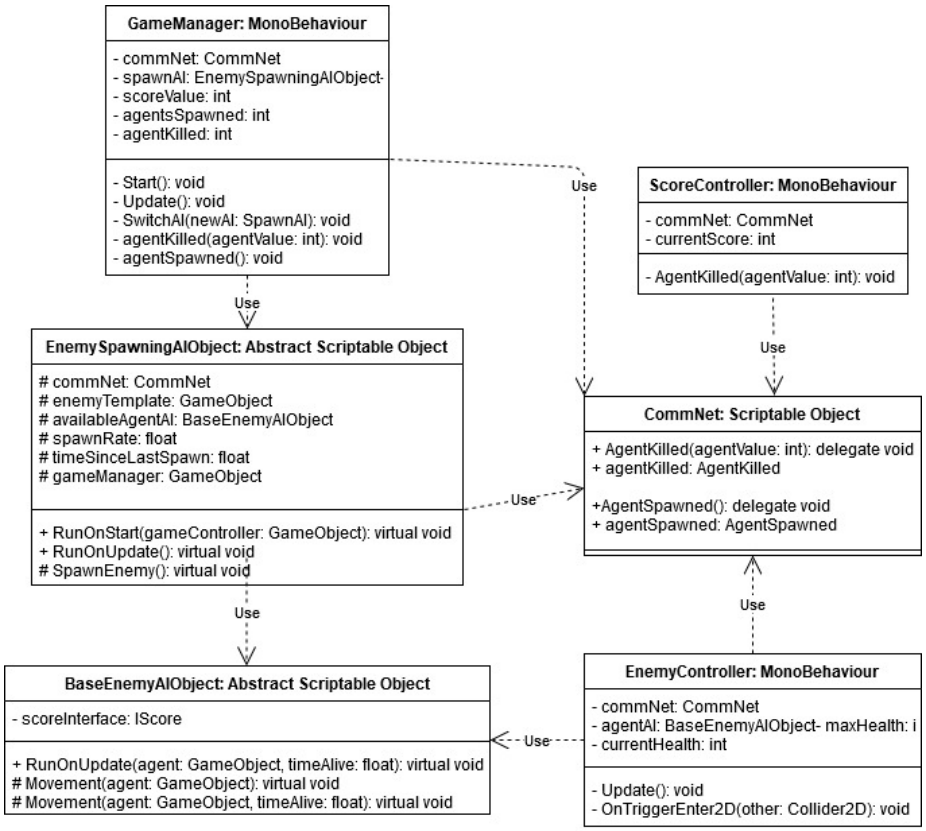

Figure 5: Class Diagram

4 Reflection

4.1 Concept

My initial concept idea, a choose your own adventure style game played using the same controller, with the text displayed on the vacuum fluorescent display, unfortunately had to be scrapped due to how the display was mounted to the PCB. This new concept was intentionally scaled back in order to allow more time to work on the GAM130 project, and has mostly been acting as a test bed for creating hot-swappable AI modules and using scriptable objects and delegates to make decoupled event driven systems.
I have previously done electronics at GCSE, as such I've already been exposed to this style of project and have limited interest in the physical aspect. It has, however, provided a nice break and change of pace to our usual projects, and working on a physical piece can certainly be satisfying.

4.2 UML Diagrams

This was my first time really using UML diagrams, and I have actually found them very useful. While the user diagram was mostly unnecessary given the simplicity of the interactions, the class diagram proved very useful for planning out classes and seeing dependencies. This made it easy to see that there were a lot of interdependencies, and directly led to me implementing the CommNet scriptable object to fix this problem. I can see the use of these event systems leading to much cleaner code, with the planning that comes from creating a UML diagram being a big help.

4.3 Software

As mentioned above I have used this project to experiment with delegates and AI modules, neither of which I had done before. This project did lead to a 'eureka' moment when I finally understood what delegates actually were, whereupon I begun to realise just how useful they could be.
I am also quite proud of the AI modules, while the actual AI is very simple, the framework I've built could easily be used in other projects. This could potentially form the basis for a finite state machine style AI and should make the assignment and management of sub and super states easier.