Github link: <https://github.com/Jord159/IMDCGP206-Jordan_Carman>

12/10/18

Concept:

The concept I’m working on for this project is movement of a character in a game world using the Emotiv Epoc +. I plan on using mental commands for forwards and backwards movement. Once that works, I plan on using either the accelerometer or more mental commands to rotate the character.

19/10/18

Plan:

|  |  |
| --- | --- |
| **Week** | **Target** |
| 1 | Refamiliarise myself with API. Figure out how to use Cortex alongside Unity and set neutral state and mental commands |
| 2 | Apply mental commands to simple objects, ensure reliability in detection and execution of commands |
| 3 | Apply mental commands to forwards and backwards movement of a character in Unity, experiment with using accelerometer to rotate objects |
| 4 | If accelerometer viable: ensure accuracy in rotations, apply to character in Unity  If accelerometer not viable: apply more mental commands to character in Unity for rotations |
| 5 | Presentation of progress |
| Onwards | If work is delayed, complete incomplete work. After that, experiment with more ways to use Emotiv headset to control a character |

After reading through the Cortex documentation, the best way to get the Cortex API to work alongside Unity is by loading the required data into Unity via JSON files. I still need to find out exactly how often the data is updated but my hope is that it is frequent enough that I could load the data from the JSON files in a FixedUpdate call.

26/10/18

I’ve spent several hours today reading through the documentation and the example project on the Emotiv Github repository and am yet to figure out how the JSON requests are passed to the API. I’ll spend some more time throughout the week to try to figure this out but if I can’t find out how I can even pass requests to the API then I will be unable to do anything with the Emotiv headset and will need to change the project I’m working on.

02/11/18

I’ve finally found how to access the data via Unity and it appears to work fine, though I will need to double check this once I have access to the headset. It seems that the CortexExamples solution has methods created which automatically do a lot of the requests. Using the CortexAccess namespace within the Unity project, I should hopefully be able to access the methods I’ll need to access the data.

Once that was done, I looked for the format for the data I’m given. It appears the data is output as a block of bytes in a file stream. I’m not entirely sure where the data gets output to, it seems to just go into the build location for the project, but I won’t be able to check this until I have access to the headset. My plan was to prepare an object to receive and use the data from the headset in Unity so I could quickly check how everything works but I don’t think I’ll be able to until I can actually see what’s output and where.

In the meantime, now that I have a better understanding of how the example project works, I’ll read through the code to see how I can train the neutral state and add mental commands. This is something I’m going to need to implement very soon and so I need to understand how to do it as soon as I can.