# Edward Moor 20415081

# Writing Robot Software Project Plan

# Outline of the Problem to be Solved

The ultimate goal of the project is to develop a piece of software to enable a writing robot to “draw out” text from a file. The software will read in font data from a file named ‘SingleStrokeFont.txt’, read in the text to be drawn from another file named ‘TextData.txt, and define the hight of the output text via user input. The software will generate and send G-Code commands to the Arduino via a virtual RS232 Comms serial port. This provides instructions to the robotic arm how to raise, lower, and move the arm to specified X,Y locations in order to construct the text.

The software must be capable of processing text of any length, therefor utilisation of dynamic allocation of memory will be critical. It must write between a maximum writing width of a 100mm, without any breaks in the words “drawn”. The height of the text must be defined by user input but remain between 4-10mm. The software must scale the output text by a factor equal to the desired height/18 due to the 18 units in the font file.

The software should be developed with git for version control, there should be an initial commit consisting of the skeleton code. Further files generated throughout the development process, both code and documentation, must be committed to the repository.

The software will read in font data from a file named ‘SingleStrokeFront.txt’. Receive a user input for the text height and validate the input, ensuring it is within the specified range. The text to be drawn will be read from the text file. Each word in the text file must be read, processed and output to the writing robot prior to the reading of the next.

The program will run on a standard computer and will interface with an Arduino to communicate with an X,Y plotter drawing robot.

# Key Data Items

|  |  |  |
| --- | --- | --- |
| Name | Data type | Rationale |
| TextFile.txt | str | Text File contains a string of charcaters |
| ASCII Value | int | ASCII Values are integers and range from 0 to 127, int suffices |
| Stroke Data X,Y | float | Ensures fractional values are accurately represented, important because X,Y undergo scaling from scale factor |
| Stroke Data, P | int | Either 1 or 0 |
| Scale Factor | float | Needs to be float to ensure decimal |
| Text Height | Float | Should allow for the any value between 4 and 10mm |
| WordStructure | Struct with int | Stores ASCII values and word length for each word |
| FontData | Struct with int | Flexible storage for character stroke data |
| Line Limit | Int | Defines maximum line width for layout |

# Function Declarations

*int ExtractWord(FILE \*file, WordStructure \*wordStruct)*

*Parameters:*

*FILE\*file - Pointer to text file for reading*

*WordStructure \* wordStruct – Pointer to a structure where the extracted word will be stored*

*int LoadFontData(const char \*textfile, FontData \*fontArray)*

*Parameters:*

*Const char\*textfile – Path to the font data file*

*FontData \* fontArray – Array to store font data for ascii charcaters*

# Testing Information

|  |  |  |  |
| --- | --- | --- | --- |
| Function | Test Case | Test Data | Expected Output |
| Height Prompt | User inputs value for text height | 11 | “Invalid” |
| Scale Factor | Calculate S.F based on input height | 8 | 0.444 |
| Main | Ensuring full program function | Text File : “Hello World” | 1st Word: “Hello”  2nd Word: “World” |
| Process Word | Testing line handling loop | Text File exceeds 100mm on the 3rd word | “Word 1 Word 2  Word 3” |
| Process  Word | ASCII Value | H | 72 |
| Process Word | Coordinate fetch | H | Figure 1, Appendix |
| Process Word | Letter Offset | H H | Figure 2, Appendix |
| Process Word | G Code Process | H | Figure 3, Appendix |

*Extend table as required. Note that ‘Function’ includes main()*

# Flowchart(s)

# Appendix

Figure 1

A table with numbers and numbers

Description automatically generated

Figure 2

A white paper with black text

Description automatically generated

Figure 3

A white paper with black text

Description automatically generated