PROGRESS REVIEW 10

Astha Prasad

Team F: ADD_IN

ILR #9

March 17th, 2016

Teammates:

Daniel Berman

Nikhil Baheti,

Ihsane Debbache

Introduction

The primary goal for this progress review was to print around COTS items. For this purpose, the software implementation required certain changes and bug fixes in order to generate accurate G-Code for the printer to run. These included:

- Modifying the writeToGCode function
- Allowing the GCode to include negative coordinates (for certain implementations)

Modifications to 'writeToGCode' function

The 'writeToGCode' function is responsible for converting the stored parameter values to GCode. The parameter values (G-Command, XYZR coordinates, extrusion 'e', speed 'f', bed temperature, nozzle temperature, etc.) for each line of GCode are stored in objects of the class 'printerState'. Once the R angles have been computed, the printerStates are complete and ready to be written to a .gcode file.

Motivation

In the previous implementation, the 'writeToGCode' function would directly access the 'printerState.Gcommand' variable to determine the type of GCode that was being written. However, this execution would result in the function blindly repeating unnecessary parameter values that were not required by the printer for a particular GCode. In order to correct this issue, Dan and I decided that a better implementation would be to for the function to observe the variables and the changes occurring between two consecutive printerStates to govern which GCode command is being executed and hence, intelligently add the GCommand (eg, G1, M190, G92 etc.). This change was implemented in the following fashion:

Preamble

The bulk of every GCode file is preceded with general settings for the print to be executed. Some of these are exhibited in Figure 1.

```
G21 ; set units to millimeters
G90 ; use absolute coordinates
M82 ; use absolute distances for extrusion
```

Figure 1: Preamble (initial settings)

In order to distinguish commands such as these, a variable called 'Preamble' was added to the printerState class which would be set to '1' during the 'readGCodeFile' function (i.e. when the .gcode file is read into MATLAB in the form of printerStates). Hence, if a printerState is found with the printerState.Preamble set to 1, the GCommand is directly written into the final .gcode file.

Axis commands

The bulk of the GCode comprises of mostly the following:

- G1: Contains updated values for XYZR axes, extrusion ‘e’ and speed ‘f’
- M190: Sets the temperature of the bed
- M104: Set the temperature of the nozzle
- G92: Used to reset the extrusion value ‘e’ to 0

In order to recognize these commands, the function monitors 2 consecutive printerStates and checks for changes in the relevant variables. If changes are found in the variables mentioned above, the relevant GCode is appended and only the necessary ones are written into the GCode file.

Negative Coordinates

Although, ideally, the XYZ coordinates should all be positive, they can turn out to be negative due to the settings in the configuration file. For example, in case the end stop for the Z-axis is not accurately calibrated and is at an error of ‘c’, the Z layer can be set to start from ‘-c’ using the configuration file. In such cases, the Z coordinates for the first couple of layers turns out to be negative. This case of negative coordinates was not taken care of by the read function of the MATLAB implementation and resulted in errors in the final GCode. This was rectified by making a minor change in the ‘GetVal’ function that returned the values following alphabetical (XYZ etc.) commands. It now searches for negative numbers too.

Challenges

With increasing amount of testing and running our software on various types of STL files, different software bugs are slowly coming to light. This may cause the software team to rethink some of the algorithms used to implement certain functions, such as the ‘writeGCodeFile’ function mentioned above. As the team strives to improve the print quality of the modified printer, there will undoubtedly be many more such changes required in the implementation.

Teamwork

Remaining goals for the Progress Review were distributed as follows:

Nikhil Baheti

Nikhil worked on porting his Inverse Kinematics implementation from MATLAB to the firmware.

Dan Berman

Dan worked on altering the ComputeRAxis function in order to correct some irregularities seen during test prints. Along with this, he generated basic GCode to print around an old heat block. He also helped Nikhil with some firmware fixes.

Ihsane Debbache

Ihsane tinkered around with the printer for a while but was mostly busy with other assignments. He was unable to contribute to this PR much due to a conference he was attending and will hopefully be putting in the lost hours for the coming PR.

Future Goals

After preliminary printing around a COTS item, several issues have come to light. Accordingly, the team will be working towards accomplishing the following goals:

- Fixing the extrusion rate during the dwell time around corners
- Improving the nozzle mount to make it more rigid
- Conducting adhesion testing
- Conducting COTS item feature location repeatability testing
- Implementing the MATLAB GUI on a web interface.

In addition, there will no doubt be more software issues that will come up over the next two weeks. The team's main focus is on improving the print quality around COTS items and all 4 of us will be invested in this task.