Progress Review 12

Nikhil Baheti

Team F: ADD_IN

Teammates: Ihsane Debbache, Dan Berman and Astha Prasad

ILR11

April 13th, 2016

1 Individual Progress

For this progress review, I had to work modifying the printer’s firmware to ensure that the print does not drift. I also completed the R-axis homing command and increased the R-axis stepper motor holding torque. I worked with the team on printing parts to narrow down the drift error and finally print around COTS. I also went through the path planning code in the software to help with any minor fixes that the code needed when other team members were busy.

1.1 Drift Error Reduction

The following steps were performed to narrow down the drift in the print.

- Firstly, we tested prints without feed rates. It was observed that increasing the feed rate would increase the drift in the print.
- Printing the part using the original firmware did not produce any drift. This implied that the drift existed due to some computations in inverse kinematics block of code.
- This might be due to the limited resolution of the sine and cosine look up tables and thus we accumulated the stepping error across G-codes. This did not solve the drift issue.
- Then we tried printing the same part without R-axis commands in the G-code. This uses the inverse kinematics code block but does not change dx and dy with time. The drift error persisted which confirmed that the resolution of the look up table was not an issue.
- Comparing the changes with the original firmware we found out that at the beginning of every G-code the stepping variables dx, dy, dr and dz are set to “step_event_count/2.” We were setting it to 0. Making this change reduced the drift error and now the feed rate could be increased without affecting the drift.
- Minor drifts still exists which we are still working on. One way of eliminating the drift for a simple cylinder was to cumulate the R-axis commands till the relative movement is a minimum of 5 degrees. However, this only solved the drift issues with the cylinder and not all parts in general.
- Also, cumulating the R-axis resulted in G-codes which did not have R-axis commands and thus would process quicker than the other G-codes because they did not have to solve inverse kinematics. This resulted in the non-uniform extrusion rates between the G-codes resulting in thick lines in one case and thin lines in the other.
- To solve this we have introduced the same calculations even when the R-axis commands are not present, and then we reject these calculations. So this gives similar extrusion rates for G-codes with and without R-axis commands.

1.2 R-axis Homing

The R-axis homing command was not implemented. Thus the firmware was modified to ensure homing all axes also homes the R-axis and extended the R-axis homing to G28 command. Another, feature is provided to set an offset rotation in the firmware to account for non-alignment between the nozzle tip and the slot on the encoder wheel. Also, since the R-axis motor current was low it would result in mechanical shifts in nozzle orientation which would result in errors. This was controlled by increasing the motor current till the stepper motor does not heat up. It is currently set to 0.9A.

1.3 Printing around COTS:

Dan made a MATLAB script that could generate G-codes for hollow cylinders. We used this G-code to print around COTS item. Since, the heat blocks and nozzles were machined, the nozzle offset changed and the look up tables had to be modified for the firmware. After this we printed around the screw insert which is shown in Figure 1.

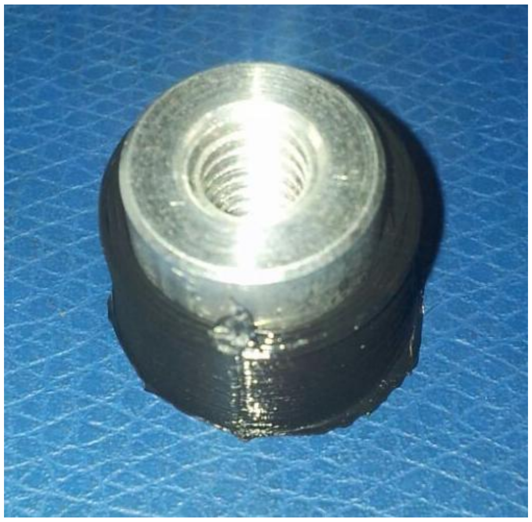
Figure 1: Printing around a screw insert

1.4 Software Changes

While printing a square I noticed that the R-axis would rotate by 45 degrees around the corners. This would occur whenever slicer generated small G-codes that would shift between perimeters. So I went ahead and modified the “ComputeRaxis” function in MATLAB to not rotate the R-axis for small G-code executions. This also resulted in the software not generating any R-axis commands for the cylinder. Then Dan suggested a better solution which was to generate the R-axis codes based on where the R-axis would point at the end of the G-code instead of the centre of the G-code. This was the appropriate fix and this ensured that even for squares the G-code did not generate large R-axis changes for G-codes with small lengths.

2 Challenges

The challenges faced during the term of this progress review which is quite consistent with the previous Arduino problems and other challenges are explained as follows:

1. Printing parts takes a long time and since the drift is small checking for corrected solutions takes a long time.
2. Sometimes the print does not stick to the print bed and thus monitoring the print constantly is another time consuming task.

3 Teamwork

- **Astha:** She worked on solving minor bugs in the software Gcode generation. Since working with the firmware bugs has now saturated my thinking, Astha helps me out while I am working with the firmware.
- **Dan:** Dan worked on solving all the mechanical issues with the printers. He made the heat block for both the printers. He also helped me with pointers to narrow down bugs in the firmware. He also, made a quick MATLAB code to generate cylinders. This was used to test if cumulating the R-axis would actually solve the problem. This, also solved helped us print parts to test for quality while the drift issues were being tackled.
- **Ihsane:** He was continuously testing print and solved the nozzle jamming issues whenever they occurred.
- **Team collaboration:** This progress review we integrated all subsystems and thus everyone was working in proximity with each other. Most of the collaboration was to get printing around COTS and thus we kept meeting to solve issues in the individual subsystems.

4 Plan

For the next progress review, I plan to work on solving the drift in the printing due to the firmware. I will also keep testing the prints to ensure we have tested and validated the system before SVE.