Progress Review 11

Nikhil Baheti

Team F: ADD_IN

Teammates: Ihsane Debbache, Dan Berman and Astha Prasad

ILR10

March 31th, 2016

1 Individual Progress

For this progress review, I had to work on making the inverse kinematics map time efficient. Then modify the new printer's firmware to ensure that testing may be started on that printer too. I also completed the R-axis homing command. During the integration phase we had problems with bed heating and a fan burn out which were repaired too.

1.1 Inverse Kinematics Map

The inverse kinematics and uniform velocity profile for the nozzle tip was solved the previous PR but it had issues with time of execution. If the ISR takes a long time to execute, then the G-code parsing gets affected as it is a parallel processing block with the ISR. This results in the firmware missing a few lines of G-codes or resulting in checksum error while parsing the G-codes. The following steps were performed to mitigate these errors:

- The first way to reduce the execution time of the ISR is to use integer computations instead of float computations. The following is the modification in the velocity calculations for the x and y axes:

$$dx = dxe - round(offset * \text{x\_steps\_per\_mm} * sign(dr) * sign(dye) * maxstep * (\cos(r + dr) - \cos(\text{r})));$$

$$dy = dye - round(offset * \text{y\_steps\_per\_mm} * sign(dr) * sign(dye) * maxstep * (\sin(r + dr) - \sin(r)));$$

$where,$

$dxe \rightarrow steps\ the\ nozzle\ tip\ moves\ along\ X - axis\ for\ a\ given\ G - code$

$dx \rightarrow steps\ the\ X - axis\ stepper\ motor\ must\ move\ for\ a\ given\ G - code$

$dye \rightarrow steps\ the\ nozzle\ tip\ moves\ along\ Y - axis\ for\ a\ given\ G - code$

$dy \rightarrow steps\ the\ Y - axis\ stepper\ motor\ must\ move\ for\ a\ given\ G - code$

$dr \rightarrow steps\ the\ R - axis\ stepper\ motor\ must\ move\ per\ timer\ interrupt$

$r \rightarrow current\ R - axis\ angle$

$\text{x\_steps\_per\_mm} \rightarrow number\ of\ steps\ X - axis\ stepper\ motor\ must\ move\ to\ cover\ 1\ mm$

$\text{y\_steps\_per\_mm} \rightarrow number\ of\ steps\ Y - axis\ stepper\ motor\ must\ move\ to\ cover\ 1\ mm$

$maxstep \rightarrow The\ maximum\ steps\ any\ stepper\ motor\ moves\ for\ a\ given\ G - code$

$offset \rightarrow offset\ in\ mm\ of\ the\ tip\ of\ the\ extruder\ from\ the\ printer\ interface$

- To futher reduce the time a look table was used instead of computing the sine and cosine values. The lookup table stores the value of the difference of two consecutive sine differences in steps of the r-axis stepper motor resolution. Since the minimum difference of the two sine values is very small the look up table stores values to a magnified resolution of 2^20. This storage saves memory and also saves time to look for the table values in the ISR.

- The above computations need a storage type of “long long” to maintain the resolution. By now the frequency of G-codes being skipped reduced and the check sum error still persisted.
- To further reduce the time to compute the IK we had to make sure computations were in the resolution of “long” instead of “long long.” Since, the nozzle offset and steps_per_mm are constants for a given printer the look up table was now computed by multiplying these values and now the resolution was reduced to 2^11. Thus the computations could be performed in “long” without overflowing. This stopped the skipping of G-codes error.
- To stop the check sum error a lot of methods were used to narrow the error. To ensure that the computations aren’t causing the error I initialized the variables to “int” and only performed an increment or decrement operation on the variables but the error still persisted.
- Commenting the IK block would stop the error. The only other part of IK was stepping and therefore the error must exist here. We were using signed typecast to check if the direction of stepping must be changed. Changing the usage of signed typecast stopped this error.
- After all these changes there is still some drift in the nozzle tip while it prints and the possible reason for this is the errors that accumulate due to the truncation in the accuracy of the look up table. Figure 1.1 shows the drift while printing a cylinder.

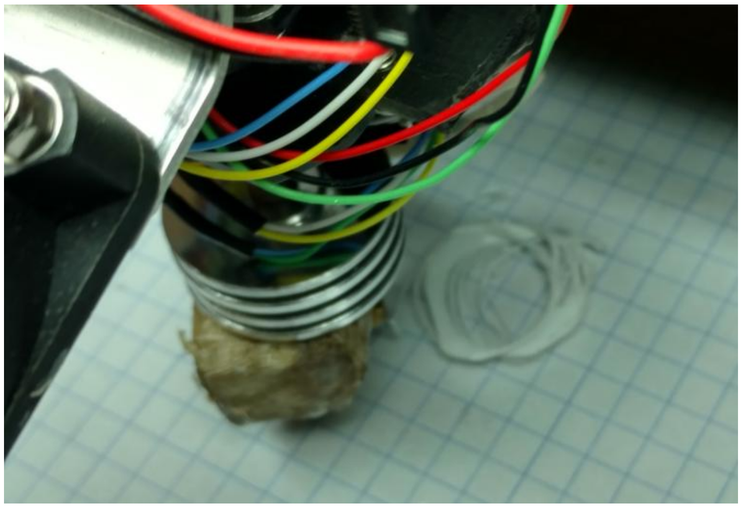
Figure 1.1: Drift in the print while printing a cylinder

1.2 R-axis Homing

Homing the x, y and z axes results in the movement of the r-axis too. This is because I had not modified the planner function call to pass the r-axis steps while homing. Thus, the variable for r-steps did not get updated in the ISR. This resulted in r-axis stepper motor making random rotations. After making the change in the planner function call this error was solved.

1.3 New Printer's Firmware:

I had to modify the new printer's firmware to ensure that ADD_IN firmware is capable of running on both printers. The new printer was compatible with the original Marlin firmware but not with the ADD_IN firmware. The reason for this is that the ADD_IN firmware uses the LCD screen to resume printing after COTS item insertion. The LCD screen uses I2C communication protocol and no signal on the start and data lines results in a blocking of the USB communication lines with the PC. Commenting the LCD screen definition solved this problem too.

2 Challenges

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

1. While performing the IK I was using long type cast and the Arduino compiler treats long as unsigned by default. This took a lot of time to be narrowed down. But the clue was that the print statements always were a constant value to 2048 for dx and dy. This value is equal to 2^12 and which led to the conclusion that must be the value when unsigned long is right shifted by 20. Now a signed long type cast is used instead.
2. Since, the dynamic memory of ATMega2560 is only 8kb, the lookup tables cannot be stored as global variables. Thus they were stored in the program memory using PROGMEM. But the access of these arrays must be done by a function defined in the Arduino library and not like accessing general array. It took me a while to understand this concept and then making the change solved the problem of dx and dy assuming random values.
3. While testing the heater bed wasn't heating up which was due to a burnt MOSFET on the board. So we switched to the spare board but that had the y-axis stepper driver burnt. So we had to desolder a MOSFET from latter board and solder it to the former board. The connector that supplies power to the heater bed had also melted and we had to change the connectors too.

3 Teamwork

- **Astha:** She worked on solving minor bugs in the software Gcode generation. She also helped me fix the MOSFET and connector soldering issue. Since working with the firmware bugs has now saturated my thinking, Astha helps me out while I am working with the firmware. She is going through the firmware to understand it better so that the issues with firmware may be solved quicker.
- **Dan:** Dan worked on setting up the new printer. He made a sturdier mount for the new printer and an adapter for his nozzle. He also helped me with pointers to narrow down bugs in the firmware.

- **Ihsane:** He printed the new mount and helped us with nozzle clogging issues. He also performed the COTS locating feature test on the Makerbot printer.
- **Team collaboration:** This progress review we integrated all subsystems and thus everyone was working in proximity with each other. The entire team kept meeting alternate days to share problems and help each other out.

4 Plan

For the next progress review, I plan to work on the making solving the drift in the printing due to the firmware. Since the next PR is the dress rehearsal I will take up any work that helps us achieve the objectives of the SVE.