Diagonal rod correction for Delta printing XY dimensional calibration

Salmerón Valdivieso, H

February-2017

Contents

1 Abstract

Domestic 3D Printing can be a very frustrating task because of the continuous and problematic calibration of your printer. This is increased if you work with a Delta geometry 3D Printer, which are much harder to build and calibrate.
This text develops the calculation of the diagonal rod value for each tower. It may solve the XY dimensional error of your printings, with the assumption that you have a calibrated Horizontal Radius, steps/mm and Nema motors.

2 Mathematical model

Let's model a translation from the centre of the print bed towards one of the three towers (Figure 1).

I'll call D_o to the diagonal rod value given to the controller and D to the "real" diagonal rod value that matches with the dimensional accuracy.

As seen, the controller is going to displace vertically the carriages a distance Dh function of the parameters actually given to match the distance L^* that you ordered. The software, supposing correct parameters, calculates h^*.
When the nozzle is centred we know that the projection of diagonal rod is known and equal to the Horizontal Radius R.Because of the Pythagoras theorem:

$$h^* = \sqrt{D_o^2 - R^2} \tag{1}$$

$$Dh = \sqrt{D_o^2 - (R - L^*)^2} - h^* \tag{2}$$

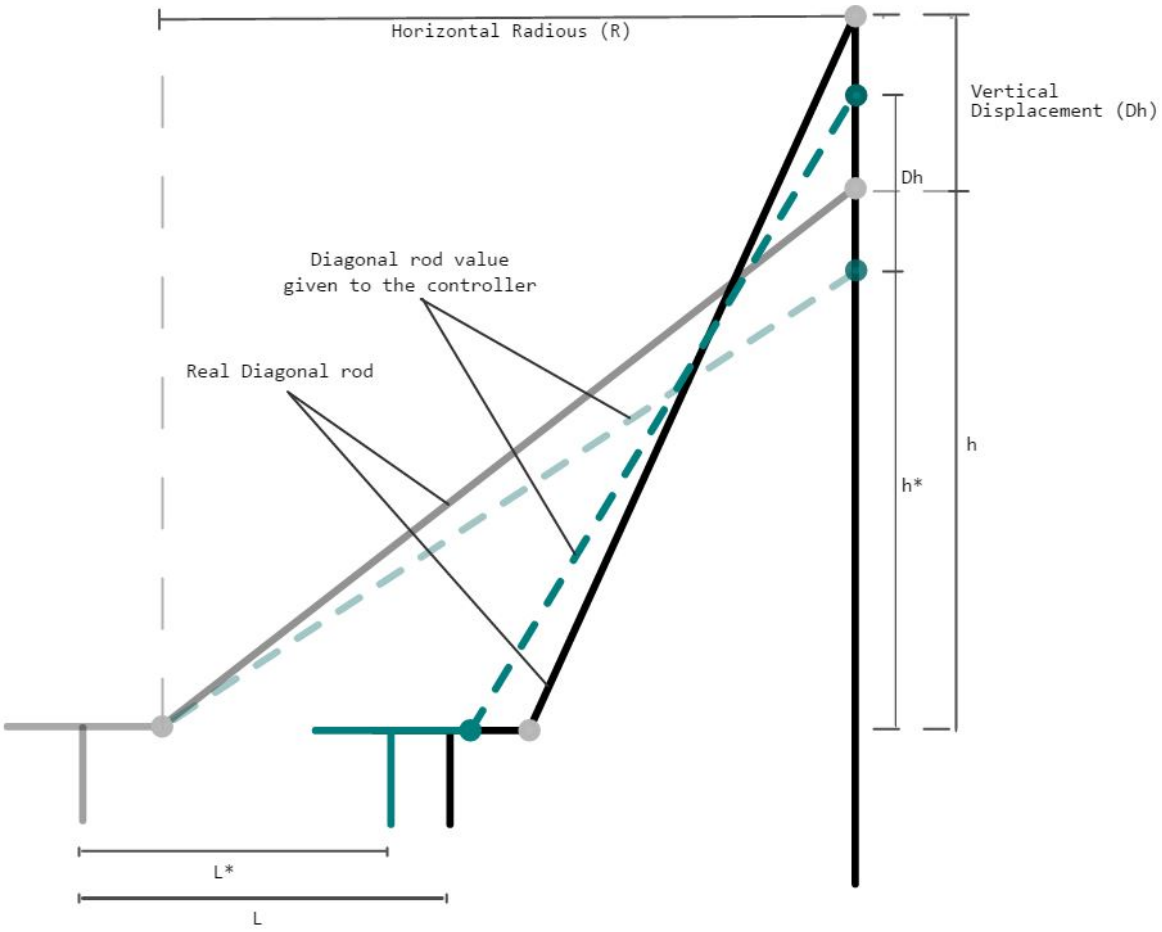


Figure 1: Schematic of a translation

The nozzle covers another distance L (which we can measure). This is due to calculating Dh as function of a different diagonal rod value than the real one. Let us calculate that "real" diagonal rod value D which produces a translation length L for a vertical displacement Dh of the carriages.

$$L = R - \sqrt{D^2 - (\sqrt{D^2 - R^2} + Dh)^2} \tag{3}$$

As reader can see the dimensional error on the perpendicular direction of the tower is only introduced by the diagonal rods of that tower, so this method allows us to distinguish the error introduced by each tower's diagonal rods.

3 Implementation

An easy tool to use this method is published on my "Thingiverse" profile (*http://www.thingiverse.com/thing:1274733*).

To implement those calculations we must solve the equations (1), (2) and (3). After having a mathematical expression to calculate the corrected value of the diagonal rod D as function of the given displacement L^*, L, R and D_o, we can print a piece and introduce the measured values on our function.

A good idea is to print an hexagon, because it has two faces confronted in front of each tower, so you can measure the distance between each two faces, divide by two and use that distance (with double measurement accuracy).

3.1 Use of the "Thingiverse" file.

The main idea of this implementation is to print the Hexagon piece attached to the file and then use some measurements on an also attached spreadsheet.

The printing must be made without any XY dimensional correction or any modification of the particular diagonal rod of each tower.

The notation for those measurement is the one used on Figure 2.

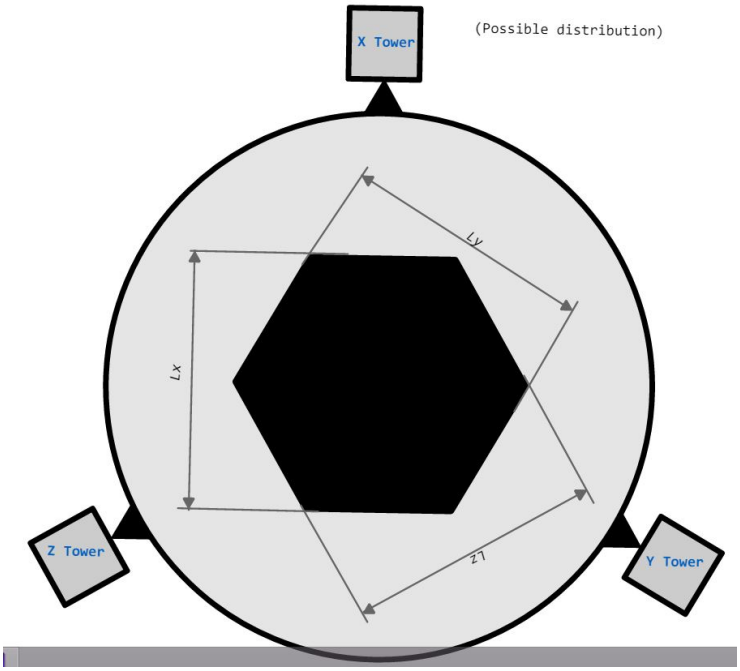


Figure 2: Measurements of the hexagon printed piece

So L is going to be the distance between faces that is supposed to be, and L_x, L_y and L_z the one we measure in the printed piece.

To use L_x, L_y and L_z in our mathematical expression we must divide them by two, because our model supposed a translation from the bed centre, but the spreadsheet is already prepeared to work with the measured distance, so no operation is needed to work with it.
So we name

$$LX = L_x$$
$$LY = L_y$$
$$LZ = L_z$$

and introduce them on our Excel file.
After introducing our Horizontal Radius, current used value of Diagonal Rod length, the exact length of the digital model and our measurements on the Excel file, we must get something like as shown in the Figure 3.

INPUT					
Horiz. Radius	D0	L	LX	LY	LZ
84,000	209,500	75,000	75,620	74,440	74,800

Auxiliar			
D (mean)	Estimated Dx	Estimated Dy	Estimated Dz
209,418	210,566	208,532	209,155

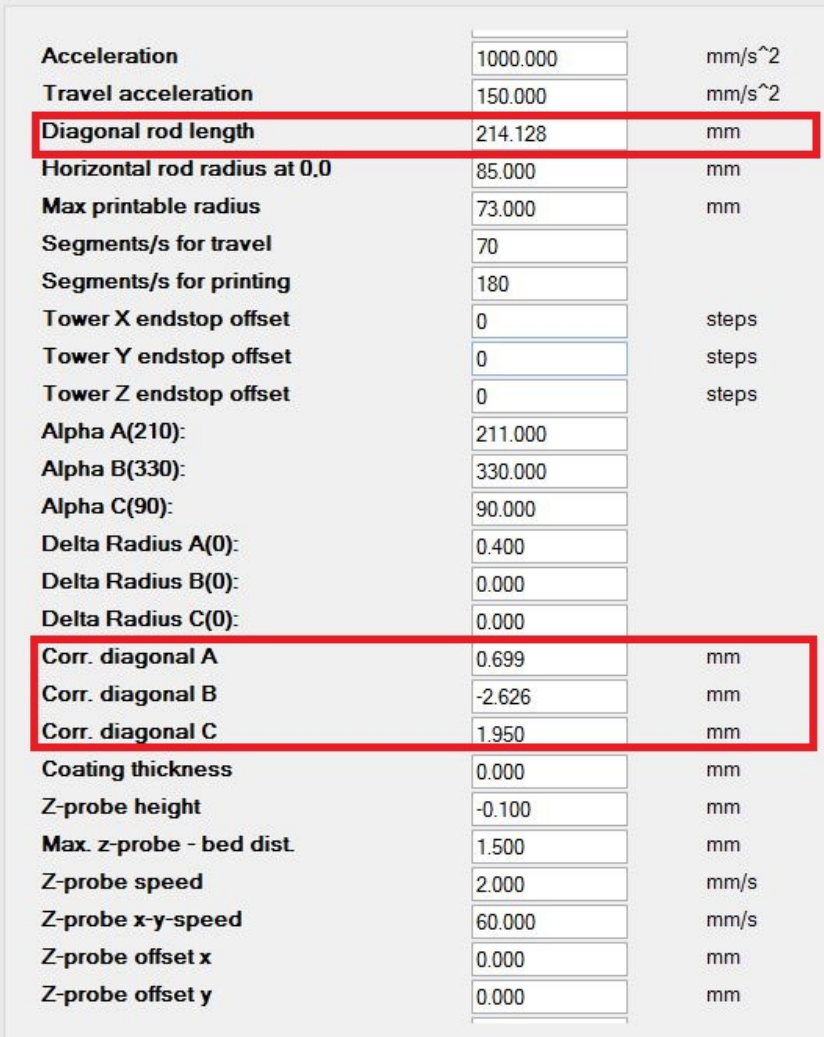
OUTPUT For Repetier			
Diagonal rod lenght	Corr.diagonal A	Corr.diagonal B	Corr. diagonal C
209,418	1,148	-0,886	-0,263

OUTPUT For Marlin			
D (mean)	Diag_rod_x	Diag_rod_x	Diag_rod_x
209,418	210,566	208,532	209,155

Figure 3: Example of the spreadsheet after introducing data.

3.1.1 Modifying values on Repetier Firmware

You can introduce the new values just editing the Eprom from Repetier software (Figure 4).



Acceleration	1000.000	mm/s^2
Travel acceleration	150.000	mm/s^2
Diagonal rod length	214.128	mm
Horizontal rod radius at 0.0	85.000	mm
Max printable radius	73.000	mm
Segments/s for travel	70	
Segments/s for printing	180	
Tower X endstop offset	0	steps
Tower Y endstop offset	0	steps
Tower Z endstop offset	0	steps
Alpha A(210):	211.000	
Alpha B(330):	330.000	
Alpha C(90):	90.000	
Delta Radius A(0):	0.400	
Delta Radius B(0):	0.000	
Delta Radius C(0):	0.000	
Corr. diagonal A	0.699	mm
Corr. diagonal B	-2.626	mm
Corr. diagonal C	1.950	mm
Coating thickness	0.000	mm
Z-probe height	-0.100	mm
Max. z-probe - bed dist.	1.500	mm
Z-probe speed	2.000	mm/s
Z-probe x-y-speed	60.000	mm/s
Z-probe offset x	0.000	mm
Z-probe offset y	0.000	mm

Figure 4: Eprom memory editor window

3.1.2 Modifying values on Marlin Firmware

Setting different values for each diagonal rod in Marlin firmware is a little more tricky, but can be done in a few minutes. This is an example of the main idea, but your version of marlin may diverge from this one and those lines may doesn't exist on your code.

Step 1

You must open the source code and go to archive ‘‘`Marlin_main.cpp`’’. Look for the code shown on Figure 5 and sustitute it by the code in Figure 6 using your results.
`Diag_rod_x`, `Diag_rod_y` and `Diag_rod_z` are the new diagonal rod values.

```
DELTA_DIAGONAL_ROD_2 = pow(delta_diagonal_rod,2);
```

Figure 5: Original code lines of Marlin.

```
DELTA_DIAGONAL_X_ROD_2 = pow(Diag_rod_x,2);
DELTA_DIAGONAL_Y_ROD_2 = pow(Diag_rod_y,2);
DELTA_DIAGONAL_Z_ROD_2 = pow(Diag_rod_z,2);
```

Figure 6: The new code.

Step 2

Then look for the code shown in Figure 7 and replace it by the one shown in Figure 8.

Step 3

After this you must set `DEFAULT_DELTA_DIAGONAL_ROD` in ‘‘`Configuration.h`’’ with the mean value given by the spreadsheet.

```
void calculate_delta(float cartesian[3])
{
  delta[X_AXIS] = sqrt(DELTA_DIAGONAL_ROD_2
                       - sq(delta_tower1_x-cartesian[X_AXIS])
                       - sq(delta_tower1_y-cartesian[Y_AXIS])
                       ) + cartesian[Z_AXIS];
  delta[Y_AXIS] = sqrt(DELTA_DIAGONAL_ROD_2
                       - sq(delta_tower2_x-cartesian[X_AXIS])
                       - sq(delta_tower2_y-cartesian[Y_AXIS])
                       ) + cartesian[Z_AXIS];
  delta[Z_AXIS] = sqrt(DELTA_DIAGONAL_ROD_2
                       - sq(delta_tower3_x-cartesian[X_AXIS])
                       - sq(delta_tower3_y-cartesian[Y_AXIS])
                       ) + cartesian[Z_AXIS];
```

Figure 7: Original code lines of Marlin

```
void calculate_delta(float cartesian[3])
{
  delta[X_AXIS] = sqrt(DELTA_DIAGONAL_X_ROD_2
                       - sq(delta_tower1_x-cartesian[X_AXIS])
                       - sq(delta_tower1_y-cartesian[Y_AXIS])
                       ) + cartesian[Z_AXIS];
  delta[Y_AXIS] = sqrt(DELTA_DIAGONAL_Y_ROD_2
                       - sq(delta_tower2_x-cartesian[X_AXIS])
                       - sq(delta_tower2_y-cartesian[Y_AXIS])
                       ) + cartesian[Z_AXIS];
  delta[Z_AXIS] = sqrt(DELTA_DIAGONAL_Z_ROD_2
                       - sq(delta_tower3_x-cartesian[X_AXIS])
                       - sq(delta_tower3_y-cartesian[Y_AXIS])
                       ) + cartesian[Z_AXIS];
```

Figure 8: The new code

3.1.3 Example.

I want to calibrate my delta printer using this method. For the calibration I use a common *diagonal rod* value of 209.5 mm on the three towers, and an *Horizontal Radius* value of 84 mm. and I use no dimensional correction on X or Y.

In my case I'll print an hexagon which size 75 mm face to face, so the parameter $L = 75mm$. After printing I measure the distance between the faces in front of each tower as shown in the figure 2, and I get this measurements:

$L_x = 75.62mm$
$L_y = 74.44mm$
$L_z = 74.80mm$

After measurement I introduce all the data on the spreadsheet and I obtain something as shown in the Figure 3.

I use Repetier Firmware so I copy the output for Repetier and paste it where the Figure 4 points.

4 FAQ

1. The method doesn't fix a little error on some direction. Am I doing it wrong? -You may have another kind of error source such tower distance or tower angle, and probably the first one. Try check those parameters with little variations.
Been able to reach a uniform bed calibration indicates that angle and tower radius are calibrated. When both are well calibrated, the shape of the height error on the bed is a 3d parabola centred on the bed centre.
If this is not your situation you must check:

- Your height sensor is accurate and precise enough: is usual to have angular error on your probe which fake results.
- Calibrate your endstops height until you can't reach more homogeneity.
- Check tower angle and radius. Those errors can't be compensated with anything.

2. I can't see the code lines of the example on my marlin code. -Marlin versions may diverge one from each other. The idea is always the same: assign different rod length values for each diagonal rod.

3. I can't reach exact length after calibrating. -That's common, delta printers are not very accurate, and depending on the quality of the filament and extrusor, thermal dilation become relevant. Don't worry if you can't reach precision under 0.2-0.3 mm. If you want more accuracy, check possible loose joints, inertia, printing temperature and accuracy of your nema motors. Usually 0.4mm nozzle with 1.8º Nema can't reach stable precision under 0.3mm