Advanced Robotics Manipulation

HomeWork #1

Team: Vladislav Ostankovich, Nikita Dayanov, Mikhail Bortnikov

**Report**

Codes are here: [GitHub](https://github.com/isaakbormental/Delta3DPrinterModeling)

1. Description of the considered robot:

A 'delta' robot or printer is a robot where a platform is maintained by three pair of arms set in a triangle. The pairs of parallel arms maintain the horizontality of the platform and the movement of these arms displace the platform in the three dimensions. Arm pairs are attached to carriage sliding along parallel rails. This geometry is called linear delta and is the most frequent type used in 3D printer world, the machines originating the movement being the Rostock and the Kossel. The term 'parallel delta' shall not be used as all robots with parallel arms are called parallel robots.

1. Kinematic scheme:

3P parallel robot:

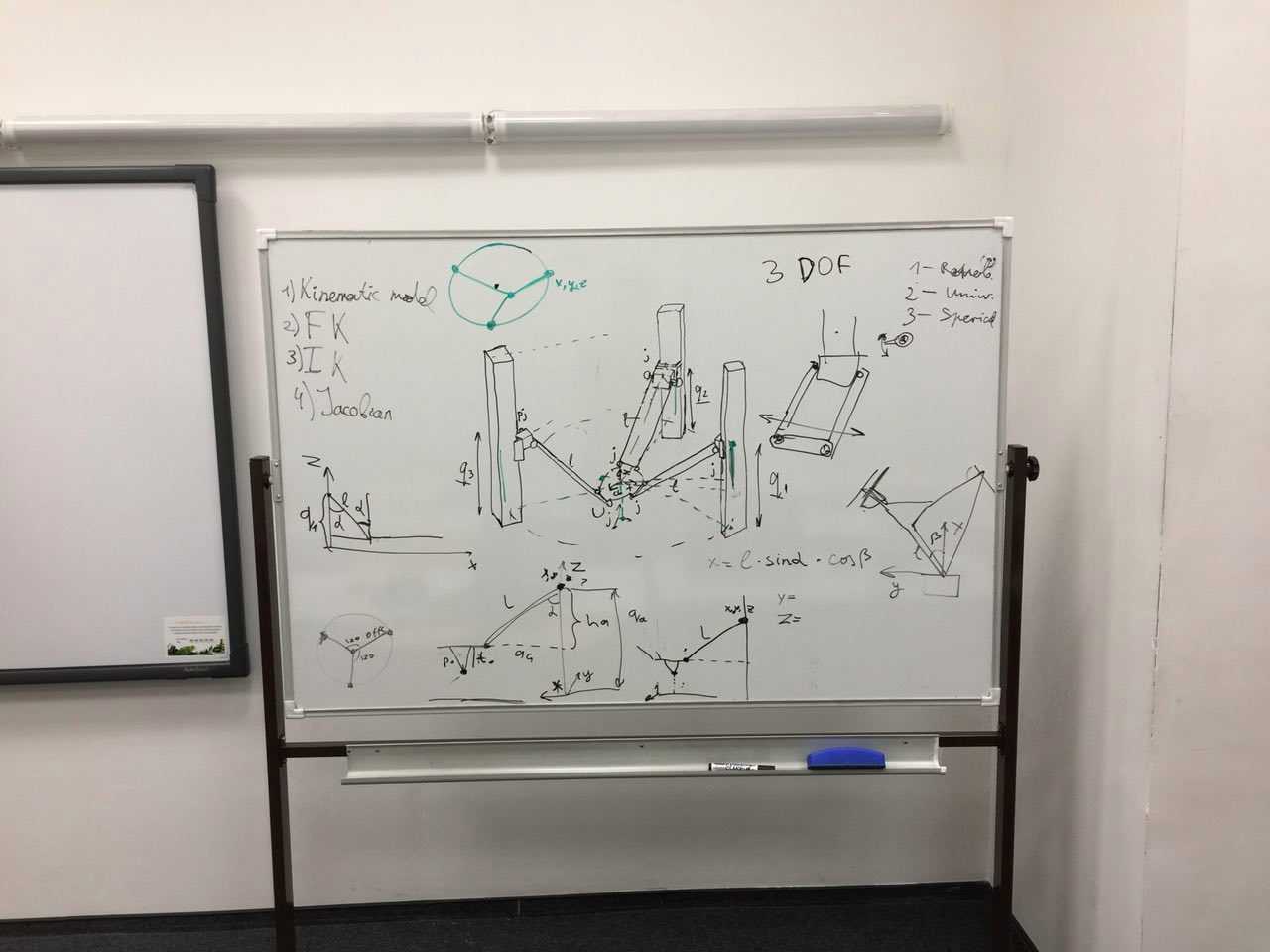


Figure 1. Schematics

1. Step by step explanation of the inverse kinematics solution:

A2 = (a2x,a2y,a2z)  
B2 = (b2x,b2y,b2z)  
C2 = (c2x,c2y,c2z)

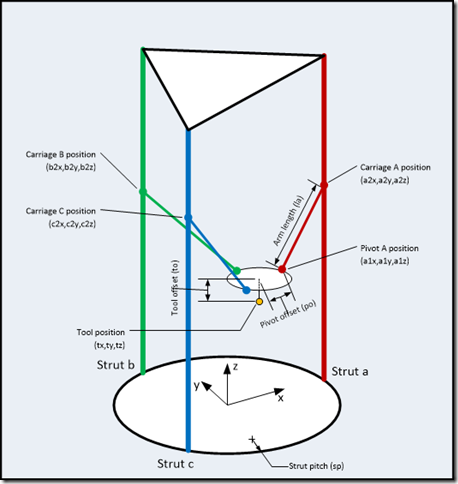
[](http://robinsonia.com/wp/wp-content/uploads/2013/02/image.png)

Figure 2. One more schematics

A1 = (a1x,a1y,a1z)  
B1 = (b1x,b1y,b1z)  
C1 = (c1x,c1y,c1z)

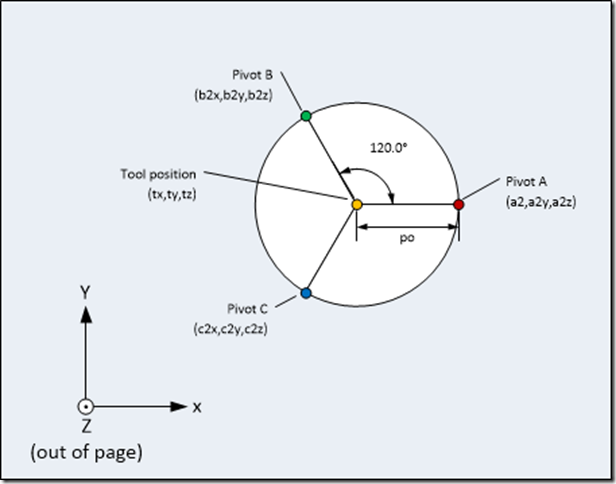
[](http://robinsonia.com/wp/wp-content/uploads/2013/02/image1.png)

Figure 3. Top vies of moving platform

|  |  |  |
| --- | --- | --- |
| **Pivot A** | **Pivot B** | **Pivot C** |
| a1x = tx + po a1y = ty a1z = tz + to | b1x = tx + po \* cos(120) b1y = ty + po \* sin(120) b1z = tz + to | c1x = tx + po \* cos(240) c1y = ty + po \* sin(240) c1z = tz + to |

Where [tx, ty, tz] = T

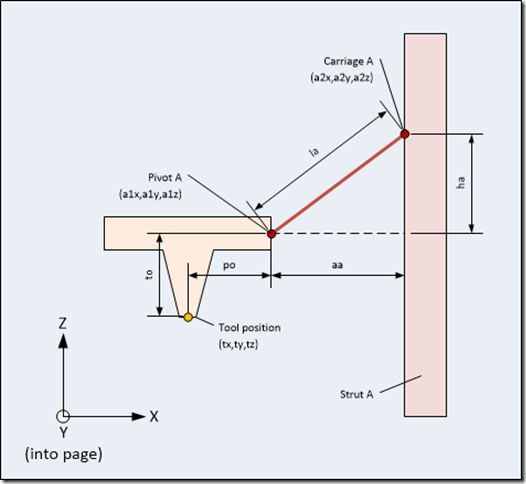
[](http://robinsonia.com/wp/wp-content/uploads/2013/02/image2.png)

Figure 4. Side view of one chain of manipulator

|  |  |  |
| --- | --- | --- |
| **Carriage A** | **Carriage B** | **Carriage C** |
| a2x = sp a2y = 0 | b2x = sp \* cos(120) b1y = sp \* sin(120) | c2x = sp \* cos(240) c2y = sp \* sin(240) |

aa = sqrt((a2x-a1x)\*(a2x-a1x) + (a2y-a1y)\*(a2y-a1y))  
ab = sqrt((b2x-b1x)\*(b2x-b1x) + (b2y-b1y)\*(b2y-b1y))  
ac = sqrt((c2x-c1x)\*(c2x-c1x) + (c2y-c1y)\*(c2y-c1y))

ha = sqrt((la\*la) – (aa\*aa))  
hb = sqrt((la\*la) – (ab\*ab))  
hc = sqrt((la\*la) – (ac\*ac))

a2z = tz + to + ha  
b2z = tz + to + hb  
c2z = tz + to + hc

Where q = [a2z, b2z, c2z]

1. Computation of Jacobian

Given in a MATLAB script in symbolic form

1. Singularity maps for robot workspace

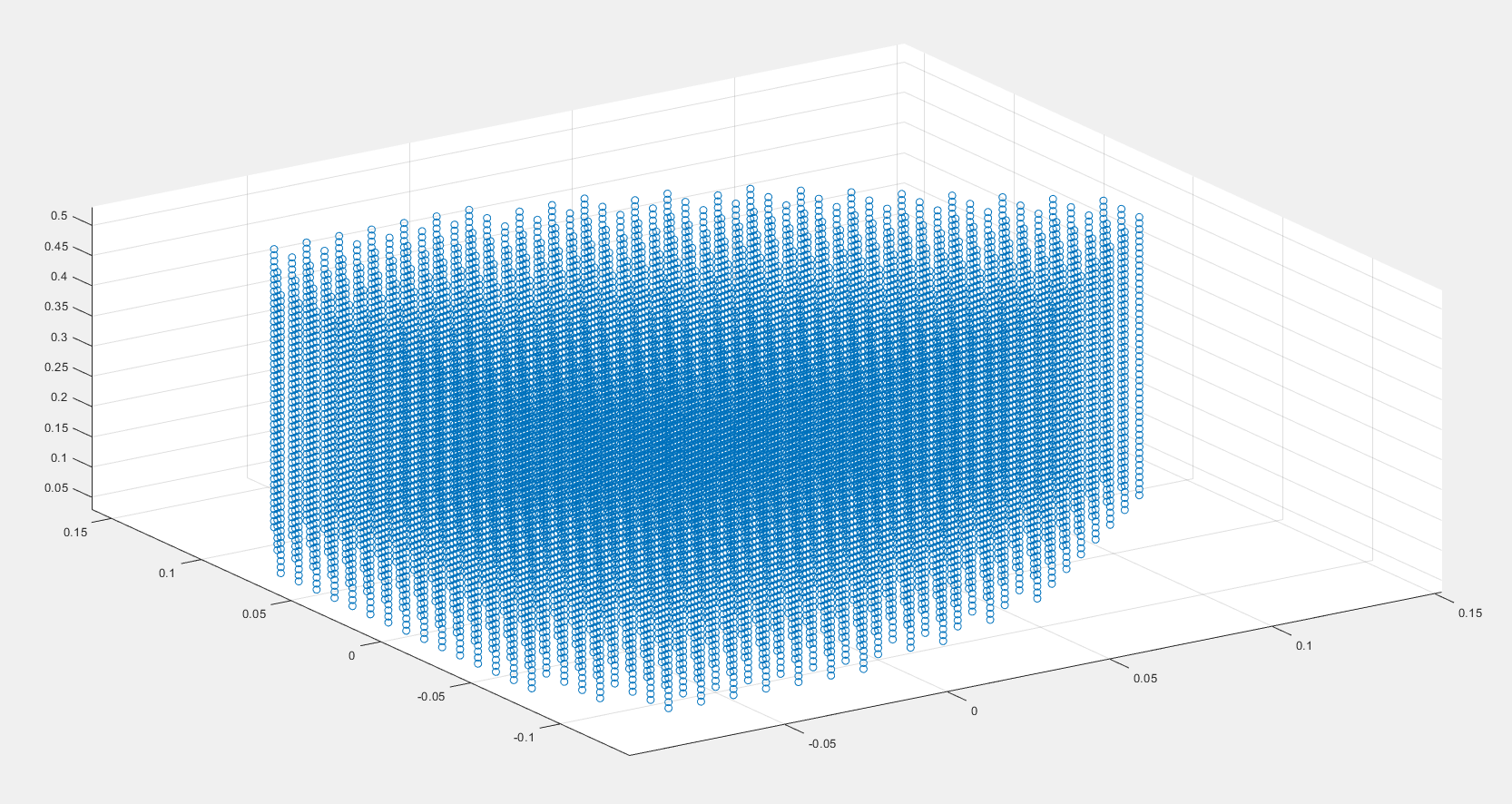


Figure 5. Reachability map

1. Analysis of obtained results

Obtained results are good.

1. Summary

To conclude, everything is ok.