Gough Stewart platform inverse kinematics modeling

Kinematics, Dynamics and Control for Robots, Homework 1

Ivan Senilov

2018/02/03

1 Introduction

In this report, we review 6-DoF Gough Stewart platform (or just Stewart platform) [1] robotic manipulator and calculate its inverse kinematics. Furthermore, we model the inverse kinematics in Matlab and make an animation of circular movement of the platform.

The Stewart Platform consists of 2 rigid frames connected by 6 variable length legs. One of the common configurations of the Stewart platform with universal joints at the bottom plate is shown in Figure 1.

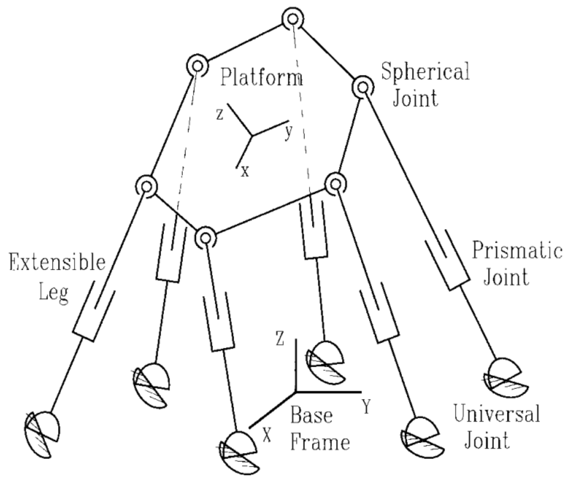

Figure 1: The 6-UPS Stewart platform (reprinted from [1]).

2 Robot kinematics and transformation matrices

For description of the manipulator kinematics, we will use transformation matrices. Simplified kinematics scheme of the Stewart platform is shown in Figure 2. There we may see two reference frames (x, y, z) and (x', y', z') for base plate and platform respectively. As mentioned in the Introduction, the manipulator has 6 degrees of freedom: 3 translational displacements (along x, y and z axes) and 3 angular displacements:

- Rotate an angle α (yaw) around the z-axis
- Rotate an angle β (pitch) around the y-axis
- Rotate an angle γ (roll) around the x-axis

Translation is fairly simple whereas the rotation matrices of the Platform relative to the Base are:

$$R_z(\alpha) = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{1}$$

$$R_y(\beta) = \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix}. \tag{2}$$

$$R_x(\gamma) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{pmatrix}. \tag{3}$$

And full rotation matrix:

$$\begin{aligned} &R(\alpha,\beta,\gamma) = R_z(\alpha)\,R_y(\beta)\,R_x(\gamma) = \\ &\begin{pmatrix} \cos\alpha\cos\beta & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \end{pmatrix}. \end{aligned} \tag{4}$$

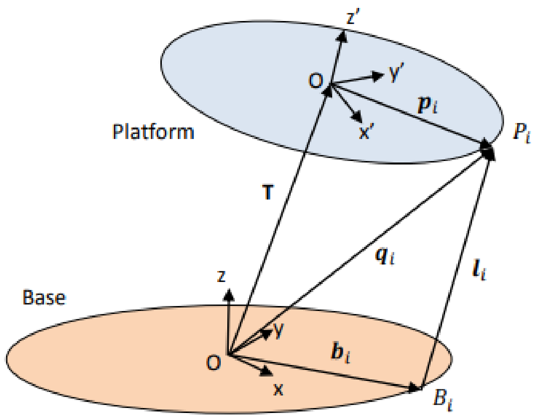

Figure 2: Stewart platform simplified kinematics

The coordinates q_i of the point P_i with respect to the Base reference frame are given by the equation:

$$\mathbf{q_i} = \mathbf{T} + {}^{\mathbf{P}}\mathbf{R_B} \cdot \mathbf{p_i} \tag{5}$$

Where $\mathbf{T}$ is the translation vector, giving the positional linear displacement of the origin of the Platform frame with respect to the Base reference framework, and $\mathbf{p_i}$ is the vector defining the coordinates of the anchor point $\mathbf{P_i}$ with respect to the Platform framework.

The length of the i^{th} leg is given by:

$$\mathbf{l_i} = \mathbf{T} + {}^{\mathbf{P}}\mathbf{R_B} \cdot \mathbf{p_i} - \mathbf{b_i} \tag{6}$$

where $\mathbf{b_i}$ is the vector defining the coordinates of the lower anchor point $\mathbf{B_i}$.

On the other hand, considering the configuration of each leg, from bottom to top:

- Revolute joint around z-axis
- Link with length $d_1 = 0.02m$
- Revolute joint around y-axis
- Link with length $d_2 = 0.2m$
- Prismatic joint with variable length $d_3 = 0 \dots 0.3m$

In this case, the length of the leg may be expresses like this (in homogeneous coordinates):

$$\mathbf{l_i} = R_z(\lambda_1)T_z(d_1)R_y(\lambda_2)T_z(d_2 + d_3) \tag{7}$$

For simplicity, we assume that after first link there is a spherical passive joint instead of two revolute joints so we don't care about rotation angles in the leg.

Considering scheme of mounting points of legs on Base and Platform (see Figure 3):

$$\mathbf{b_1} = \begin{pmatrix} R_b\cos(330° + \Delta\theta) \\ R_b\sin(330° + \Delta\theta) \\ 0 \end{pmatrix}; \mathbf{b_2} = \begin{pmatrix} R_b\cos(90° - \Delta\theta) \\ R_b\sin(90° - \Delta\theta) \\ 0 \end{pmatrix}; \mathbf{b_3} = \begin{pmatrix} R_b\cos(90° + \Delta\theta) \\ R_b\sin(90° + \Delta\theta) \\ 0 \end{pmatrix};$$

$$\mathbf{b_4} = \begin{pmatrix} R_b\cos(210° - \Delta\theta) \\ R_b\sin(210° - \Delta\theta) \\ 0 \end{pmatrix}; \mathbf{b_5} = \begin{pmatrix} R_b\cos(210° + \Delta\theta) \\ R_b\sin(210° + \Delta\theta) \\ 0 \end{pmatrix}; \mathbf{b_6} = \begin{pmatrix} R_b\cos(330° - \Delta\theta) \\ R_b\sin(330° - \Delta\theta) \\ 0 \end{pmatrix};$$

$$\mathbf{p_1} = \begin{pmatrix} R_t\cos(30° - \Delta\theta) \\ R_t\sin(30° - \Delta\theta) \\ 0 \end{pmatrix}; \mathbf{p_2} = \begin{pmatrix} R_t\cos(30° + \Delta\theta) \\ R_t\sin(30° + \Delta\theta) \\ 0 \end{pmatrix}; \mathbf{p_3} = \begin{pmatrix} R_t\cos(150° - \Delta\theta) \\ R_t\sin(150° - \Delta\theta) \\ 0 \end{pmatrix};$$

$$\mathbf{p_4} = \begin{pmatrix} R_t\cos(150° + \Delta\theta) \\ R_t\sin(150° + \Delta\theta) \\ 0 \end{pmatrix}; \mathbf{p_5} = \begin{pmatrix} R_t\cos(270° - \Delta\theta) \\ R_t\sin(270° - \Delta\theta) \\ 0 \end{pmatrix}; \mathbf{p_6} = \begin{pmatrix} R_t\cos(270° + \Delta\theta) \\ R_t\sin(270° + \Delta\theta) \\ 0 \end{pmatrix};$$

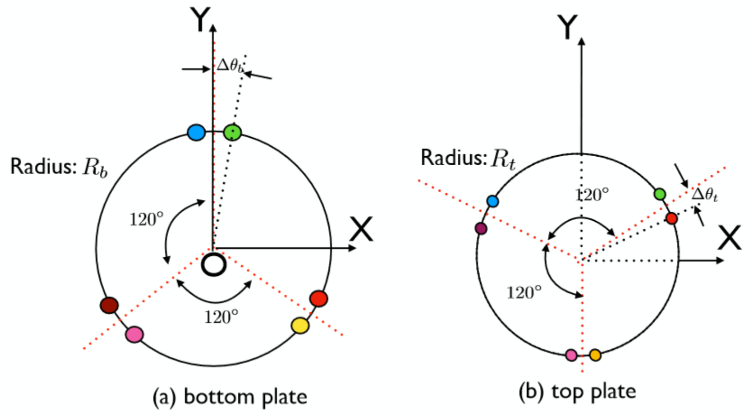

Figure 3: Mounting points on the Base and the Platform

The following code was used to calculate the inverse kinematics (calculation of prismatic joints lengths based on (x, y, z) coordinates of the platform and (α, β, γ) Euler angles) of the Stewart platform and make an animation (please see comments in the code for clarification):

```
clear variables
syms x y z al be ga real

% translation of the top plate w.r.t. base frame
```

```
T = [1 0 0 x; 0 1 0 y; 0 0 1 z; 0 0 0 1];

% rotation of the top plate w.r.t. base frame
R = Rz(degtorad(al))*Ry(degtorad(be))*Rx(degtorad(ga))

rb = 0.5; % radius of bottom plate
rt = 0.2; % radius of top plate
dt = 10; % half of angle between two legs mount point
l = [0; 0; 0.02; 0]; % static link at the bottom of the leg

% calculation of mounting points on base (b) and platform (p)
b = zeros(4,6); p = zeros(4,6);
b(:,1) = [rb*cos(degtorad(330+dt)); rb*sin(degtorad(330+dt)); 0; 1];
b(:,2) = [rb*cos(degtorad(90-dt)); rb*sin(degtorad(90-dt)); 0; 1];
b(:,3) = [rb*cos(degtorad(90+dt)); rb*sin(degtorad(90+dt)); 0; 1];
b(:,4) = [rb*cos(degtorad(210-dt)); rb*sin(degtorad(210-dt)); 0; 1];
b(:,5) = [rb*cos(degtorad(210+dt)); rb*sin(degtorad(210+dt)); 0; 1];
b(:,6) = [rb*cos(degtorad(330-dt)); rb*sin(degtorad(330-dt)); 0; 1];

p(:,1) = [rt*cos(degtorad(30-dt)); rt*sin(degtorad(30-dt)); 0; 1];
p(:,2) = [rt*cos(degtorad(30+dt)); rt*sin(degtorad(30+dt)); 0; 1];
p(:,3) = [rt*cos(degtorad(150-dt)); rt*sin(degtorad(150-dt)); 0; 1];
p(:,4) = [rt*cos(degtorad(150+dt)); rt*sin(degtorad(150+dt)); 0; 1];
p(:,5) = [rt*cos(degtorad(270-dt)); rt*sin(degtorad(270-dt)); 0; 1];
p(:,6) = [rt*cos(degtorad(270+dt)); rt*sin(degtorad(270+dt)); 0; 1];

% calculation of legs' vectors (L)
L = sym('L',[4 6]); d = zeros(4,6);

% initial position
x=0.2; y=0; z=0.25; al=0; be=0; ga=0;

for i=1:6
    L(:,i) = T*R*p(:,i) - b(:,i) - l;
    d(:,i) = subs(L(:,i));
end

lengths = zeros(6,1);
for n=1:6
    lengths(n) = norm((d(1:3,n))) - 0.2;
end

lengths

% animation code
iter=1000;
len = zeros(6,iter);
angle = 0;
delta = 0.04;
for n=1:iter
    for i=1:6
        L(:,i) = T*R*p(:,i) - b(:,i) - l;
        d(:,i) = subs(L(:,i));
    end
    for m=1:6
        quiver3(b(1,m),b(2,m),0.02,d(1,m),d(2,m),d(3,m));
```

```
        hold on;
    end
    for k=1:6
        len(k,n) = norm(d(1:3,k)) - 0.2;
    end
    angle = angle + delta;
    x = cos(angle)*0.2;
    y = sin(angle)*0.2;
    saveas(gcf,'pic/chart' + string(n) + '.png')
    pause(0.1);
    clf;
end

figure;
for a=1:6
    subplot(3,2,a);
    plot(1:iter,len(a,:))
end
```

The animation and code themselves may be found in the folder with this report.

In Figure 4, we may see plots of length of each of 6 joints for circular movement with radius of 0.2m on height of 0.25m.

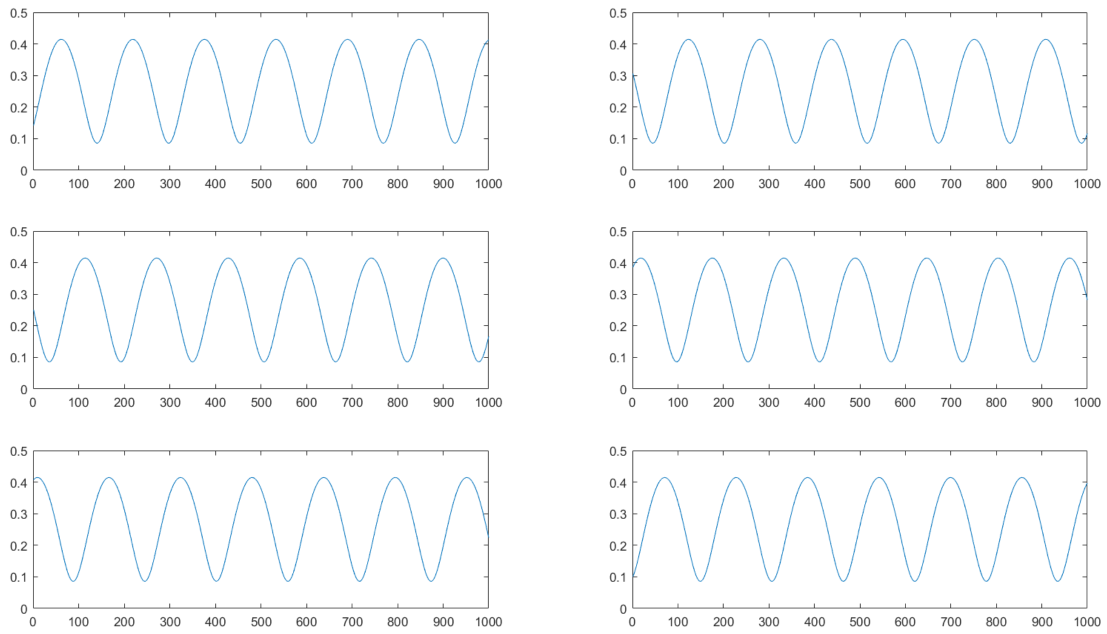

Figure 4: Prismatic joints lengths in circular motion of the platform

As it may be seen, the length of joints exceeds maximum specified length. The reason is that author of the report didn't have enough time to finalize the report.

3 Conclusion

Even though the work has not been finished due to very high educational load of the author and his need to study Matlab from zero as he never used it before, author has calculated the inverse kinematics and wrote the corresponding code. Further steps would include imposing constraints on the length of prismatic joints. It is obvious that higher values of z coordinate of the platform are also not possible.

References

[1] Bhaskar Dasgupta and TS Mruthyunjaya. The stewart platform manipulator: a review. *Mechanism and machine theory*, 35(1):15–40, 2000.