UNIVERSITÉ DE TOULON

Modeling of a Simple Pendulum and a Hexabot Leg using MATLAB/Simulink

Farooq OLANREWAJU
Nimra JABEEN

Contents

1 Modeling of a Simple Pendulum

1.1 Introduction

A simple pendulum is modelled as a simple kinematic chain with one link and one revolute joint as shown in Figure 1. The pendulum is required to fall under the influence of gravity. Hence, it is actuated by the effect of its own weight. The links and the frames are modelled using the Simulink Multibody toolbox, the results are extracted and discussed at the end of the chapter.

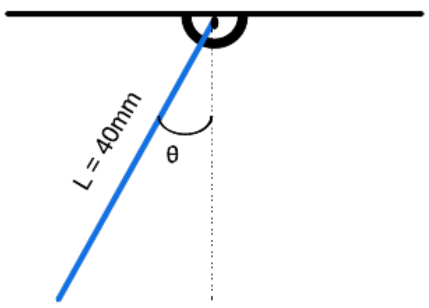

Figure 1: Free Body Diagram of the Simple Pendulum.

1.2 Modelling

To model the pendulum, the frames are placed on the link as shown in Figure 2. The base frame is (X_0, Y_0, Z_0). The revolute joint is placed at the same location but on the frame (X_1, Y_1, Z_1). The revolute joint is placed such that the axis of rotation is the z-axis and the x-axis is along the link. Hence, the axis of the z-axis of the follower is aligned to the y-axis of the base. On the Multibody toolbox, solid objects are drawn from the centre of gravity. Hence, the frame is moved along the x-axis to the expected centre of the solid body to obtain the frame (X_2, Y_2, Z_2).

The model is implemented on Simulink Multibody toolbox as shown in Figure 3. The visual representation of the frame on the MATLAB Mechanics Explorers is shown in Figure 4. The video of the motion obtained from the simulation is on Youtube. This model was executed without altering the internal mechanics and the state targets of the revolute joint.

As required for the modelling, the internal mechanics of the revolute joint is modelled with Damping Co-efficient of 0.01 Nm/s^2. Likewise, a state target of 30^0 is also included in the model. This has a significant effect on the mechanics of the simple pendulum. The simulation output is also available on Youtube and is discussed in the Results.

1.3 Result

The initial simulation without damping shows a sinusoidal motion of the pendulum with constant amplitude and frequency (as shown in Figure 5). When the damping co-efficient was added to the

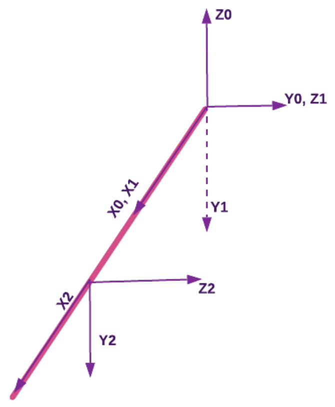

Figure 2: Simulink Multibody Model of a Simple Pundulum

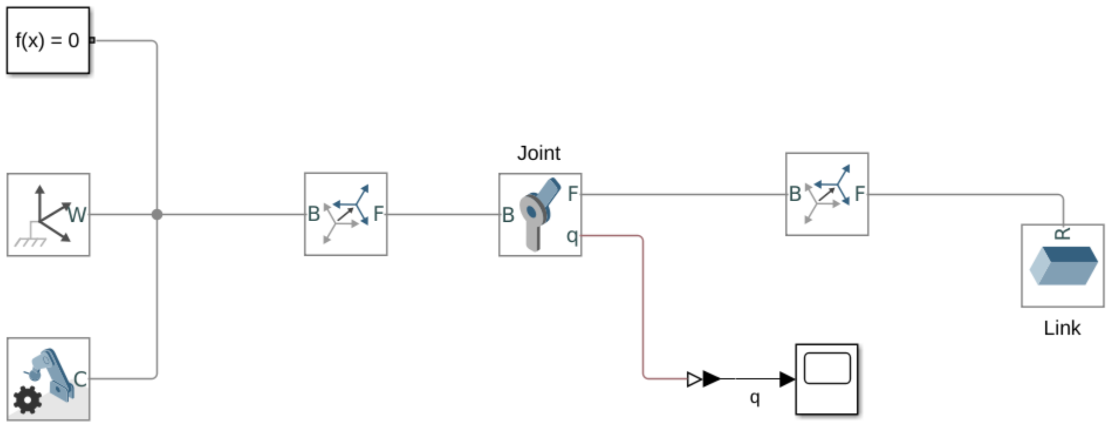

Figure 3: Simulink Multibody Model of a Simple Pundulum

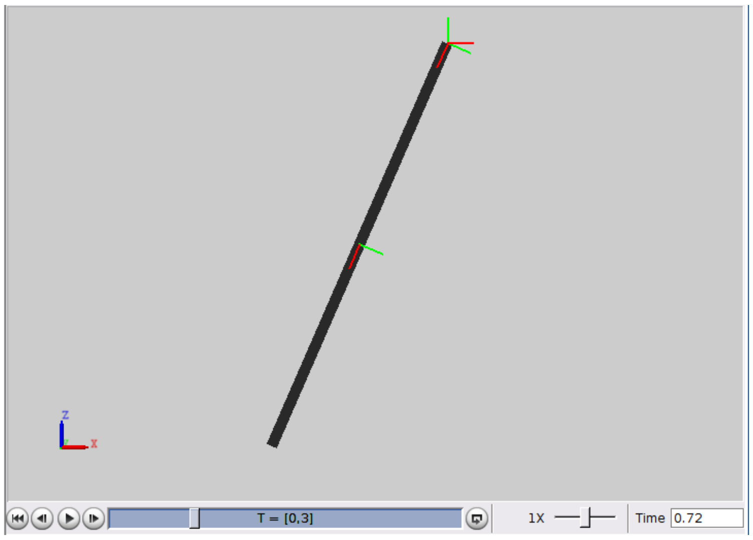

Figure 4: The frames on the pendulum link.

internal mechanics, the motion was greatly damped and the link came to rest more quickly within few seconds (as shown in Figure 7). The second simulation is a better representation of the physics of the simple pendulum.

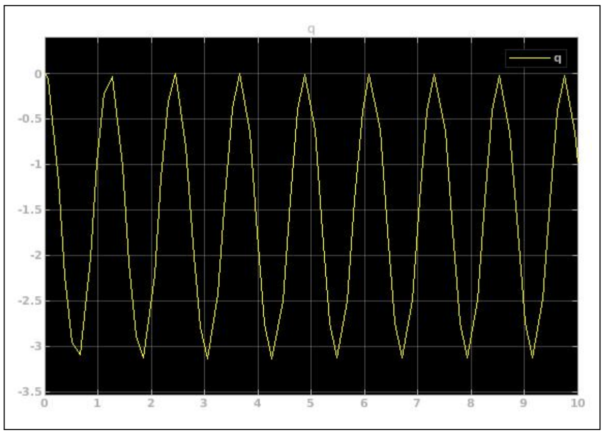

Figure 5: The graphical output of the rotation of the link without damping (in rad).

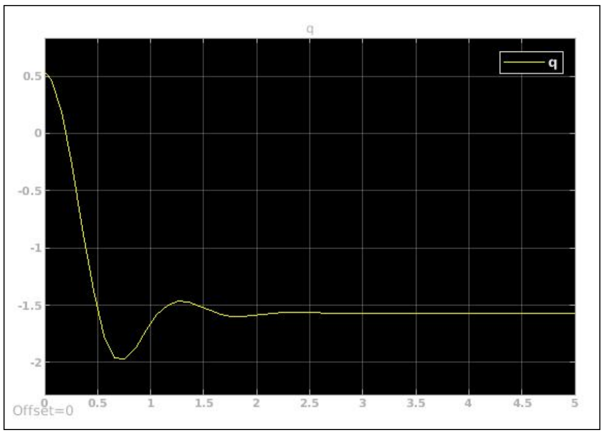

Figure 6: The graphical output of the rotation of the link with damping (in rad).

Furthermore, a brief consideration of the angular velocity of the revolute joints (as shown in Figure 6 and Figure 8) reveals the relationship of the velocities and the position of the pendulum. In the no-damping case, the velocities are maximum where the positions are zero and the velocities are zeros and the maximum position. In the damping case, the velocities are such as to obtain the targeted state.

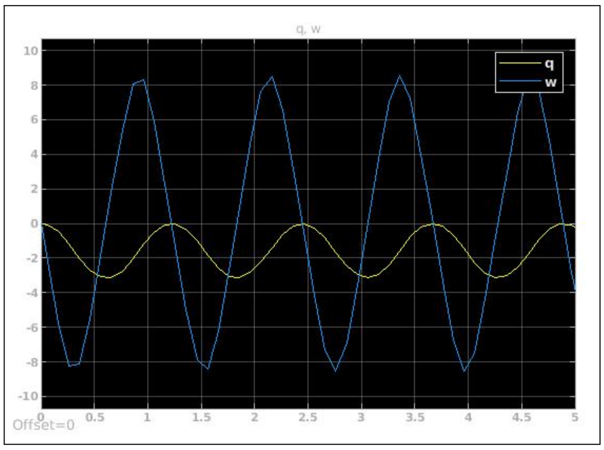

Figure 7: The graphical output of the rotation and the angular velocity of the link without damping (in rad and rad/s respectively).

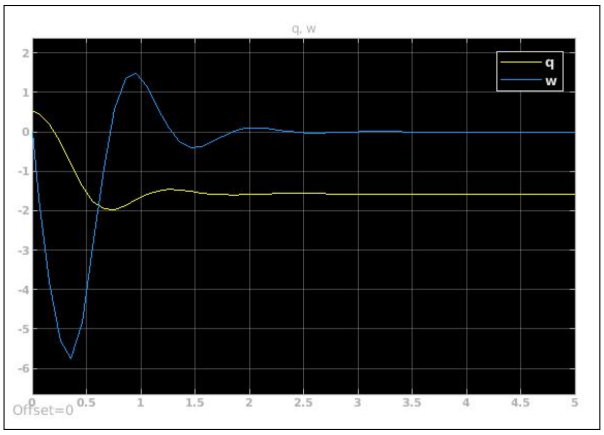

Figure 8: The graphical output of the rotation and the angular velocity of the link with damping (in rad and rad/s respectively).

2 Modelling of Hexabot Leg

A Hexabot is a robot that walks using six legs essentially imitating insect locomotion. The robot is capable of multi-direction walking and self-balancing. It is widely used to study engineering problems due to the flexibility and stability provided by the use of six legs.

2.1 Desired DOF

For the limb to be capable of motion along an elliptic path relative to the ground frame, it should have a minimum of 3 DOF.

2.2 Homogeneous Transformation Matrices

2.2.1 Frame Positioning using the DH-Formalism

Before discussing about the frame, it is pertinent to mention some convention used in the nomenclatures. O_n is the origin of a frame n. (X_n, Y_n, Z_n) are the three mutually orthogonal axes of the frame. For the DH-formalism, it is required that the X_i points to the next link and the Z_i should be axis of rotation of the revolute joint.

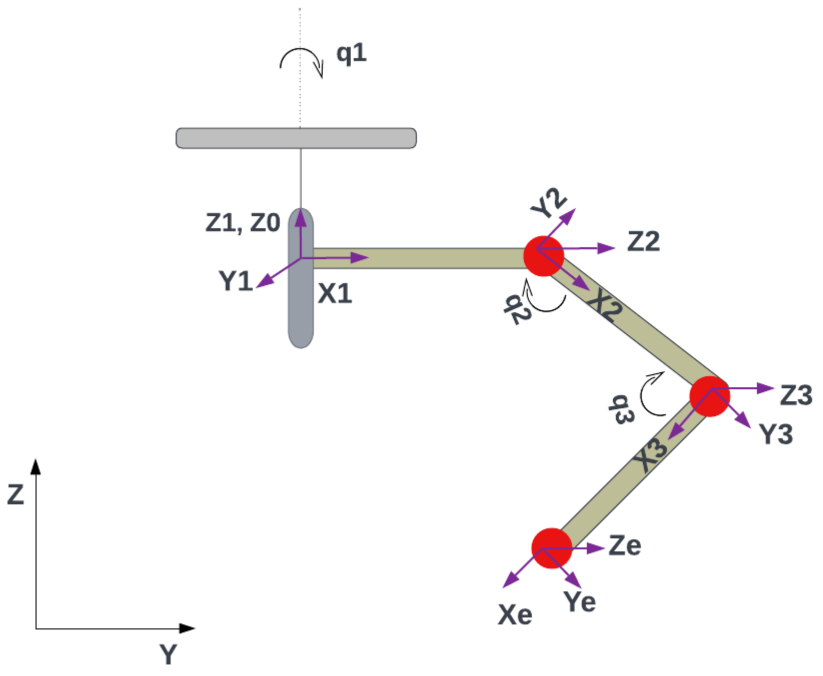

Figure 9: Free Body Diagram of the Hexabot Leg

For the modelling of the Hexabot leg, we assume that the ground is an inertia frame and we also place the base frame on the ground. The base frame is (X_0, Y_0, Z_0). The first revolute joint has a frame (X_1, Y_1, Z_1) which is placed at the same origin as the base frame but the X_1 axis is placed along the link L_1 by rotating around the Z_0 axis through angle q_1. Hence, there is only a rotation of q_1 around the Z_0

axis between frame 0 and frame 1.

The next frame (X_2, Y_2, Z_2) was translated through the length of the link L_1 along the X_1 axis so that it is on the second revolute joint. Then, the frame has to be placed such that the X_2 axis points to the link L_2 and the Z_2 is the axis of rotation of the joint. Hence, the frame O_1 is rotated through an angle of 90^0 in the clockwise direction around the X_1 axis (thereby aligning the z-axis to the axis of rotation of the joint). Then, the joint is rotated through an angle q_2 to put the X_2 axis along the next frame. Summarily, there is a translation of L_1 along X_1 and a rotation of 90^0 around it; then, there is a rotation of q_2 around the Z_1 axis to obtain the frame 2

Thereafter, the frame (X_3, Y_3, Z_3) is obtain simply by translating through the link L_2 along the X_2 axis. In the position, the new Z-axis is already inline with the axis of rotation of the revolute joint. Thereafter, the frame is rotated through an angle q_3 to the make the x-axis point to the next link. Hence, in the transformation from frame 2 to frame 2, there is a translation of L_2 along the X_2 axis and a rotation of q_3 around the Z_2 axis.

Finally, the frame on the end effector is obtained solely by translating along the L_3.

Frame, i	Joint, σ	Dist X_{i-1}; d	Angle X_{i-1}; α	Dist Z_{i-1}; r	Angle Z_{i-1}; θ
1	0	0	0	0	q_1
2	0	L_1	$-\pi/2$	0	q_2
3	0	L_2	0	0	q_3
4	$\sim$	L_3	0	0	0

Table 1: DH Parameters for Hexabot Robot

Table 1 gives a summary of the DH parameters for the Hexabot model.

2.2.2 Transformation Matrices to Intermediate Frame

The Homogeneous Transformation matrices for the intermediate frames are generated using Equation 1. Simply by substituting the values of d, α, r and θ from the DH table in Table 1, we will obtain the Equations 2 to 5.

$$T = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & d \\ \sin\theta cos\alpha & \cos\theta cos\alpha & -\sin\alpha & -r\sin\alpha \\ \sin\theta sin\alpha & \cos\theta sin\alpha & \cos\alpha & r\cos\alpha \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

$$T_1^0 = \begin{bmatrix} \cos q_1 & -\sin q_1 & 0 & d \\ \sin q_1 cos0 & \cos q_1 cos0 & -\sin 0 & -r \times sin0 \\ \sin 0 \sin q_1 & \cos 0 \sin q_1 & \cos 0 & r \times cos0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

$$T_2^1 = \begin{bmatrix} \cos \mathrm{q}_2 & -\sin \mathrm{q}_2 & 0 & \mathrm{d} \\ \sin \mathrm{q}_2 cos(-\pi/2) & \cos \mathrm{q}_2 cos(-\pi/2) & -\sin (\text{-}\pi/2) & -\mathrm{r} \times sin(-\pi/2) \\ \sin (\text{-}\pi/2) sinq_2 & \cos (\text{-}\pi/2) sinq_2 & \cos (\text{-}\pi/2) & \mathrm{r} \times cos(-\pi/2) \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c_2 & -s_2 & 0 & l_1 \\ 0 & 0 & 1 & 0 \\ -s_2 & -c_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

$$T_3^2 = \begin{bmatrix} \cos \mathrm{q}_1 & -\sin \mathrm{q}_1 & 0 & \mathrm{d} \\ \sin \mathrm{q}_1 cos0 & \cos \mathrm{q}_1 cos0 & -\sin 0 & -\mathrm{r} \times sin0 \\ \sin 0 \sin \mathrm{q}_1 & \cos 0 \sin \mathrm{q}_1 & \cos 0 & \mathrm{r} \times cos0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c_3 & -s_3 & 0 & l_2 \\ s_3 & c_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4}$$

$$T_4^3 = \begin{bmatrix} \cos \mathrm{q}_1 & -\sin \mathrm{q}_1 & 0 & \mathrm{d} \\ \sin \mathrm{q}_1 cos0 & \cos \mathrm{q}_1 cos0 & -\sin 0 & -\mathrm{r} \times sin0 \\ \sin 0 \sin \mathrm{q}_1 & \cos 0 \sin \mathrm{q}_1 & \cos 0 & \mathrm{r} \times cos0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & l_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

2.2.3 Transformation Matrices to Base Frame

The Homogeneous Transformation matrices from each intermediate frame to the base frame are obtained by multiplying the successive transformation matrices till we get to the base frame. The equation for T_1^0 remains the same.

$$T_2^0 = T_1^0 \times T_2^1 = \begin{bmatrix} \mathrm{c}_1 c_2 & -\mathrm{c}_1 s_2 & -\mathrm{s}_1 & \mathrm{c}_1 l_1 \\ \mathrm{s}_1 c_2 & -\mathrm{s}_1 s_2 & \mathrm{c}_1 & \mathrm{s}_1 l_1 \\ -\mathrm{s}_2 & -\mathrm{c}_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

$$T_3^0 = T_2^0 \times T_3^2 = \begin{bmatrix} c_1 c_{23} & -c_1 s_{23} & -s_1 & c_1 l_1 + c_1 c_2 l_2 \\ s_1 c_{23} & -s_1 s_{23} & c_1 & s_1 l_1 + s_1 c_2 l_2 \\ -s_{23} & -c_{23} & 0 & -s_2 l_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7}$$

$$T_4^0 = T_3^0 \times T_4^3 = \begin{bmatrix} c_1 c_{23} & -c_1 s_{23} & -s_1 & c_1 l_1 + c_1 c_2 l_2 + c_1 c_{23} l_3 \\ s_1 c_{23} & -s_1 s_{23} & c_1 & s_1 l_1 + s_1 c_2 l_2 + s_1 c_{23} l_3 \\ -s_{23} & -c_{23} & 0 & -s_2 l_2 - s_{23} l_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{8}$$

2.3 Jacobian Matrix

The Jacobian is computed using the direct method by considering the contribution of each joint to the end-effector motion.

For the Revolute Joint:

$$\overrightarrow{V_i} = \dot{q_j}.\overrightarrow{z_j} \times \overrightarrow{O_j O_{n+1}} \tag{9}$$

Hence:

$$\begin{aligned}\overrightarrow{V_1} = \dot{q_1}.\overrightarrow{z_1} \times \overrightarrow{O_1 O_E} &= \begin{bmatrix} c_1 & -s_1 & 0 \\ s_1 & c_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \times \left[\begin{bmatrix} c_1 l_1 + c_1 c_2 l_2 + c_1 c_{23} l_3 \\ s_1 l_1 + s_1 c_2 l_2 + s_1 c_{23} l_3 \\ -s_2 l_2 - s_{23} l_3 z \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \right] \dot{q_1} \\ &= \begin{bmatrix} -s_1(l_1 + c_2 l_2 + c_{23} l_3) \\ c_1(l_1 + c_2 l_2 + c_{23} l_3) \\ 0 \end{bmatrix} \dot{q_1}\end{aligned} \tag{10}$$

$$\begin{aligned}\overrightarrow{V_2} = \dot{q_2}.\overrightarrow{z_2} \times \overrightarrow{O_2 O_E} &= \begin{bmatrix} \mathrm{c}_1 c_2 & -\mathrm{c}_1 s_2 & -\mathrm{s}_1 \\ \mathrm{s}_1 c_2 & -\mathrm{s}_1 s_2 & \mathrm{c}_1 \\ -\mathrm{s}_2 & -\mathrm{c}_2 & 0 \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \times \left[\begin{bmatrix} c_1 l_1 + c_1 c_2 l_2 + c_1 c_{23} l_3 \\ s_1 l_1 + s_1 c_2 l_2 + s_1 c_{23} l_3 \\ -s_2 l_2 - s_{23} l_3 z \end{bmatrix} - \begin{bmatrix} \mathrm{c}_1 L_1 \\ \mathrm{s}_1 L_1 \\ 0 \end{bmatrix} \right] \dot{q_2} \\ &= \begin{bmatrix} -c_1(s_2 l_2 + s_{23} l_3) \\ -s_1(s_2 l_2 + s_{23} l_3) \\ -c_2 l_2 - c_{23} l_3 \end{bmatrix} \dot{q_2}\end{aligned} \tag{11}$$

$$\begin{aligned}\overrightarrow{V_3} = \dot{q_3}.\overrightarrow{z_3} \times \overrightarrow{O_3 O_E} &= \begin{bmatrix} c_1 c_{23} & -c_1 s_{23} & -s_1 \\ s_1 c_{23} & -s_1 s_{23} & c_1 \\ -s_{23} & -c_{23} & 0 \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \times \left[\begin{bmatrix} c_1 l_1 + c_1 c_2 l_2 + c_1 c_{23} l_3 \\ s_1 l_1 + s_1 c_2 l_2 + s_1 c_{23} l_3 \\ -s_2 l_2 - s_{23} l_3 z \end{bmatrix} - \begin{bmatrix} c_1 l_1 + c_1 c_2 l_2 \\ s_1 l_1 + s_1 c_2 l_2 \\ -s_2 l_2 \end{bmatrix} \right] \dot{q_3} \\ &= \begin{bmatrix} -c_1 s_{23} l_3 \\ -s_1 s_{23} l_3 \\ -c_{23} l_3 \end{bmatrix} \dot{q_3}\end{aligned} \tag{12}$$

Combing the Equations 10, 11 and 12 using the formula in Equation 13, we will obtain the Jacobian matrix in Equation 15. This Jacobian is verified on MATLAB using both Direct and Indirect Methods (see Appendix**??** and the attached matrixcomputation.mlx MATLAB LiveScript)

$$\overrightarrow{V} = \begin{bmatrix} \overrightarrow{V_1} & \overrightarrow{V_2} & \overrightarrow{V_3} \end{bmatrix} \tag{13}$$

$$\overrightarrow{V} = \begin{bmatrix} -s_1(l_1+c_2l_2+c_{23}l_3) & -c_1(s_2l_2+s_{23}l_3) & -c_1s_{23}l_3 \\ c_1(l_1+c_2l_2+c_{23}l_3) & -s_1(s_2l_2+s_{23}l_3) & -s_1s_{23}l_3 \\ 0 & -c_2l_2-c_{23}l_3 & -c_{23}l_3 \end{bmatrix} \dot{q} \tag{14}$$

$$J = \begin{bmatrix} -s_1(l_1+c_2l_2+c_{23}l_3) & -c_1(s_2l_2+s_{23}l_3) & -c_1s_{23}l_3 \\ c_1(l_1+c_2l_2+c_{23}l_3) & -s_1(s_2l_2+s_{23}l_3) & -s_1s_{23}l_3 \\ 0 & -c_2l_2-c_{23}l_3 & -c_{23}l_3 \end{bmatrix} \tag{15}$$

The Equation 14 shows the relationship between the jacobians, the end effector velocities and the angular speed of the revolute joints. And Equation 15 is the Jacobian matrix. This will be used in the Multibody toolbox for the modelling of the Hexabot leg.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} -s_1(l_1+c_2l_2+c_{23}l_3) & -c_1(s_2l_2+s_{23}l_3) & -c_1s_{23}l_3 \\ c_1(l_1+c_2l_2+c_{23}l_3) & -s_1(s_2l_2+s_{23}l_3) & -s_1s_{23}l_3 \\ 0 & -c_2l_2-c_{23}l_3 & -c_{23}l_3 \end{bmatrix} \times \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{16}$$

2.4 Generating a Forward Step

For the robot to obtain a motion in any given direction, we need to observe the Jacobian matrix.

$$\begin{aligned} det(J) = l_2l_3(l_3s_2c_3^2 + l_3c_2s_3c_3 + l_1s_3 - l_3s_2 + l_2c_2s_3) = 0 \\ q_3 = k\pi \quad (for\, k \in Z) \end{aligned} \tag{17}$$

The determinant equation of the Jacobian matrix which is given in Equation 17 can be utilized to know the singularity point of the robot. From the equation, singularity occurs when q_3 = 0^0 or 180^0. For these values of q_3, the motion would be impossible.

Hence, if we set the velocity $\dot{y}$ and $\dot{z}$ to zero at angles of the Revolute Joint where we don't have singularities, we can actuate the robot along the X_0 axis. This is verified from the Simulink model in the next session.

2.5 Implementing Hexabot on Simscape and Multibody Toolbox of MATLAB

The Hexabot model was simulated in Simulink with the help of Simscape and Multibody Toolbox as shown in 10. The hexabot joints and the links are modelled using the Multibody toolbox and the state of each joint is observed (for computing in the Jacobian matrix). Similarly, a sine wave and a constant block sources are used to provide the desired velocities of the end-effector. These values are used in the Jacobian to obtain the $\dot{q}$ matrix according to the Equation 19. This is implemented in the *inverseJacobian* MATLAB function.

Recall from Equation 16:

$$\overrightarrow{v} = J\dot{q} \tag{18}$$

Then:

$$\dot{q} = J^{-1}\overrightarrow{v} \tag{19}$$

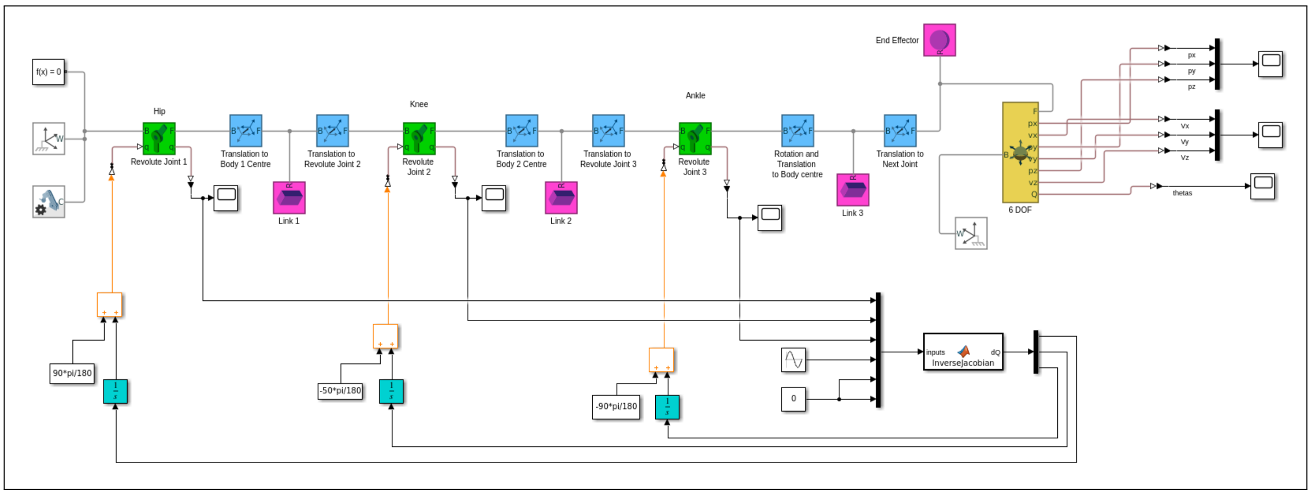
End Effector
f(x) = 0
Hip
Revolute Joint 1
Translation to Body 1 Centre
Translation to Revolute Joint 2
Knee
Revolute Joint 2
Translation to Body 2 Centre
Translation to Revolute Joint 3
Ankle
Revolute Joint 3
Rotation and Translation to Body centre
Translation to Next Joint
Link 1
Link 2
Link 3
6 DOF
px
py
pz
vx
vy
vz
thetas
90*pi/180
-50*pi/180
-90*pi/180
0
InverseJacobian

Figure 10: Simulink Model of Hexabot Leg

Thereafter, the max output is demuxed and passed through an integrator to obtain the new q values of each revolute joint.

The end effector motion is observed by attaching a 6 DOF joint to the sphere. The joint is attached such that the base is the World Frame and the follower is the sphere. This ensures that the observation of the end-effector is relative to the World Frame. The observations of the dynamics are discussed in the next section.

2.6 Observations

The observations from the Simulink Model in Figure 10 are given in Figures 11 and 12 below. .

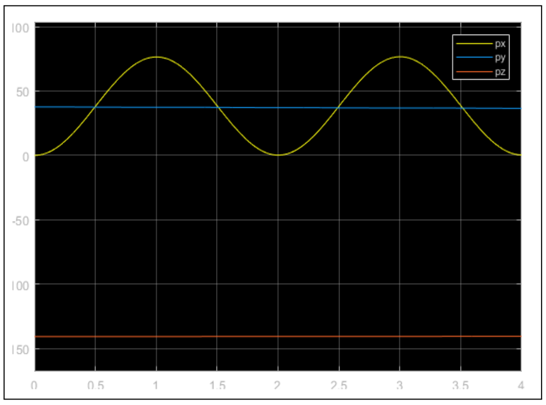

Figure 11: Position Profile for the foot of Hexabot Leg

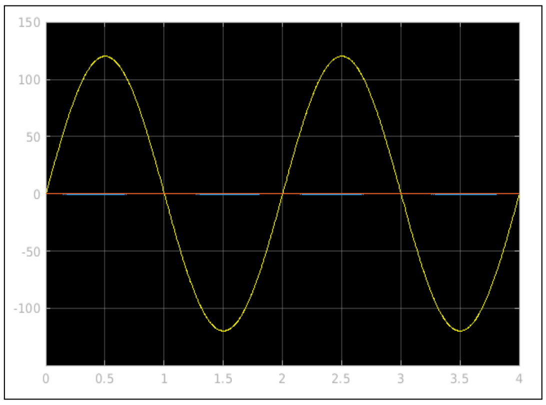

Figure 12: Velocity Profile for the foot of Hexabot Leg

In Fig. 11, we see that motion in x-axis exist without any motion in the other two axes. Further, from the simulation output in Fig. 12, we observe that it is possible to control the Hexabot leg along the base X-axis only because there is only sinusoidal change in velocity for the V_x and the other velocities are zero. The velocity profile given shows that the foot is following the desired motion. This can be further verified from our YouTube video.

The Figures 13, 14 and 15 gives the plot of the rotation of each joints to obtain the desired motion of the end-effector along the *X* axis on the ground. Similarly, the Figure 16 shows that the end-effector is rotating about the axes. Hence, it is translating along the X axis and also rotating about its own axes.

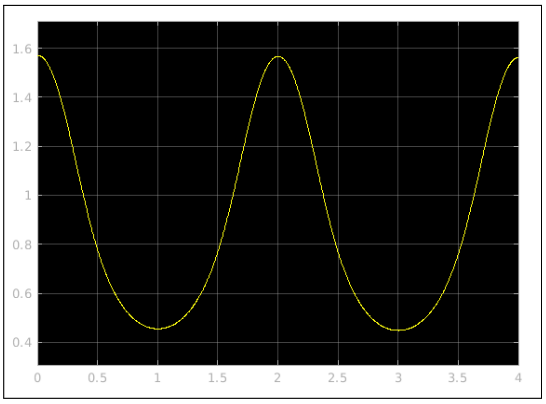
Figure 13: Rotation of the Revolute Joint Q_1

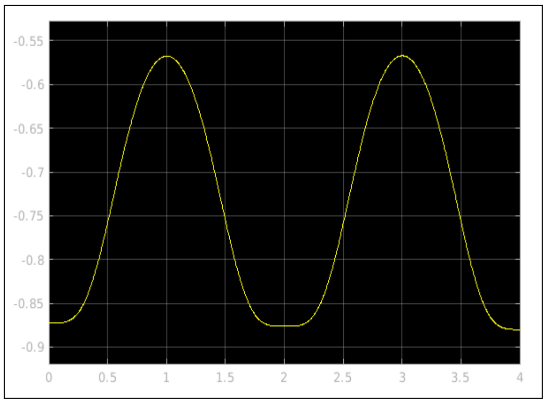
Figure 14: Rotation of the Revolute Joint Q_2

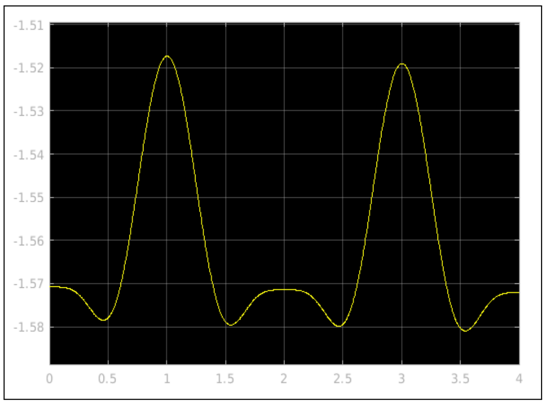
Figure 15: Rotation of the Revolute Joint Q_3

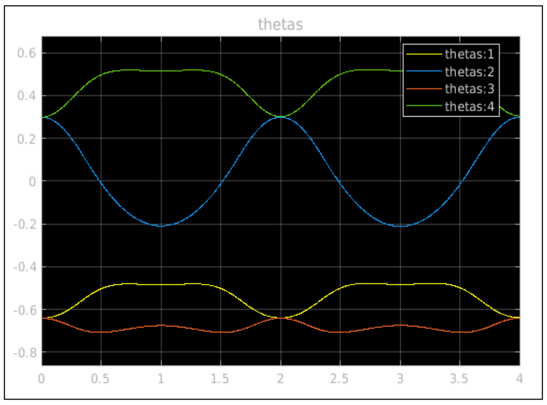

Figure 16: Rotation of the End Effector about its axes.

Appendix

The Matrix Computations of the Transformation Matrices and the Jacobian using MATLAB

Definition of the Variables used in the Computation

```
syms L1 L2 L3 q1 q2 q3 pi_val
```

Obtaining the Intermediate Transformation Matrices using DH-formalism

```
T01 = DHtransffxn(0, 0, 0, q1)
```

```
T01 =
```

$$\begin{pmatrix} \cos(q_1) & -\sin(q_1) & 0 & 0 \\ \sin(q_1) & \cos(q_1) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
T12 = subs(DHtransffxn(L1, -pi_val/2, 0, q2), [sin(pi_val/2), cos(pi_val/2)], [1, 0])
```

```
T12 =
```

$$\begin{pmatrix} \cos(q_2) & -\sin(q_2) & 0 & L_1 \\ 0 & 0 & 1 & 0 \\ -\sin(q_2) & -\cos(q_2) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
T23 = DHtransffxn(L2, 0, 0, q3)
```

```
T23 =
```

$$\begin{pmatrix} \cos(q_3) & -\sin(q_3) & 0 & L_2 \\ \sin(q_3) & \cos(q_3) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
T34 = DHtransffxn(L3, 0, 0, 0)
```

```
T34 =
```

$$\begin{pmatrix} 1 & 0 & 0 & L_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Obtaining the Transformation Matrices to the Base

```
T02 = T01 * T12
```

```
T02 =
```

$$\begin{pmatrix} \cos(q_1)\cos(q_2) & -\cos(q_1)\sin(q_2) & -\sin(q_1) & L_1\cos(q_1) \\ \cos(q_2)\sin(q_1) & -\sin(q_1)\sin(q_2) & \cos(q_1) & L_1\sin(q_1) \\ -\sin(q_2) & -\cos(q_2) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

```
T03 = simplify(T01 * T12 * T23)
```

```
T03 =
```

$$\begin{pmatrix} \cos(q_2+q_3)\cos(q_1) & -\sin(q_2+q_3)\cos(q_1) & -\sin(q_1) & \cos(q_1)\,\sigma_1 \\ \cos(q_2+q_3)\sin(q_1) & -\sin(q_2+q_3)\sin(q_1) & \cos(q_1) & \sin(q_1)\,\sigma_1 \\ -\sin(q_2+q_3) & -\cos(q_2+q_3) & 0 & -L_2\sin(q_2) \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

where

$$\sigma_1 = L_1 + L_2\cos(q_2)$$

```
T04 = simplify(T01 * T12 * T23 * T34)
```

```
T04 =
```

$$\begin{pmatrix} \cos(q_2+q_3)\cos(q_1) & -\sin(q_2+q_3)\cos(q_1) & -\sin(q_1) & \cos(q_1)\,\sigma_1 \\ \cos(q_2+q_3)\sin(q_1) & -\sin(q_2+q_3)\sin(q_1) & \cos(q_1) & \sin(q_1)\,\sigma_1 \\ -\sin(q_2+q_3) & -\cos(q_2+q_3) & 0 & -L_3\sin(q_2+q_3) - L_2\sin(q_2) \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

where

$$\sigma_1 = L_1 + L_3\cos(q_2+q_3) + L_2\cos(q_2)$$

Computing the Jacobian Using Indirect Method

The Jacobian was first calculated using Indirect Method because it appears easier and less prone to error.

```
P = T04(1:3, 4)
```

```
P =
```

$$\begin{pmatrix} \cos(q_1)\,(L_1 + L_3\cos(q_2+q_3) + L_2\cos(q_2)) \\ \sin(q_1)\,(L_1 + L_3\cos(q_2+q_3) + L_2\cos(q_2)) \\ -L_3\sin(q_2+q_3) - L_2\sin(q_2) \end{pmatrix}$$

```
J_indirect = Jacobian(P, [q1; q2; q3])
```

```
J_indirect =
```

$$\begin{pmatrix} -\sin(q_1)\,\sigma_2 & -\cos(q_1)\,\sigma_1 & -L_3\sin(q_2+q_3)\cos(q_1) \\ \cos(q_1)\,\sigma_2 & -\sin(q_1)\,\sigma_1 & -L_3\sin(q_2+q_3)\sin(q_1) \\ 0 & -\sigma_3 - L_2\cos(q_2) & -\sigma_3 \end{pmatrix}$$

where

$$\sigma_1 = L_3\sin(q_2+q_3) + L_2\sin(q_2)$$

$$\sigma_2 = L_1 + \sigma_3 + L_2\cos(q_2)$$

$$\sigma_3 = L_3\cos(q_2+q_3)$$

Obtaining Determinant of the Jacobian

```
JacDet = simplify(det(J_indirect))
```

JacDet = $L_2\,L_3\,\left(L_3\sin(q_2)\cos(q_3)^2 + L_3\cos(q_2)\sin(q_3)\cos(q_3) + L_1\sin(q_3) - L_3\sin(q_2) + L_2\cos(q_2)\sin(q_3)\right)$

```
result = solve(JacDet == 0, [q3, q2], "ReturnConditions",true, 'Real',true)
```

```
result = struct with fields:
              q3: [3×1 sym]
              q2: [3×1 sym]
      parameters: [1×2 sym]
      conditions: [3×1 sym]
```

```
simplify(result.q2)
```

ans =

$$\begin{pmatrix} x \\ x \\ x \end{pmatrix}$$

```
simplify(result.q3)
```

ans =

$$\begin{pmatrix} \pi\,k \\ 2\,\mathrm{atan}\left(\dfrac{\sigma_1 + L_3\sin(x)}{L_1 + L_2\cos(x) - L_3\cos(x)}\right) + 2\,\pi\,k \\ 2\,\pi\,k - 2\,\mathrm{atan}\left(\dfrac{\sigma_1 - L_3\sin(x)}{L_1 + L_2\cos(x) - L_3\cos(x)}\right) \end{pmatrix}$$

where

$$\sigma_1 = \sqrt{-{L_1}^2 - 2\,L_1\,L_2\cos(x) - {L_2}^2\cos(x)^2 + {L_3}^2}$$

```
result.parameters
```

```
ans = 
```
$$\begin{pmatrix} k & x \end{pmatrix}$$

```
result.conditions
```

```
ans = 
```
$$\begin{pmatrix} (k \in \mathbb{Z} \wedge x \in \mathbb{R} \wedge L_2 \neq 0 \wedge L_3 \neq 0 \wedge \sigma_1 < \sigma_2) \vee (k \in \mathbb{Z} \wedge x \in \mathbb{R} \wedge L_2 \neq 0 \wedge L_3 \neq 0 \wedge \sigma_2 \leq \sigma_1) \\ k \in \mathbb{Z} \wedge x \in \mathbb{R} \wedge L_2 \neq 0 \wedge L_3 \neq 0 \wedge \sigma_2 \leq \sigma_1 \\ k \in \mathbb{Z} \wedge x \in \mathbb{R} \wedge L_2 \neq 0 \wedge L_3 \neq 0 \wedge \sigma_2 \leq \sigma_1 \end{pmatrix}$$

where

$$\sigma_1 = L_3{}^2\ (\cos(x)^2 + \sin(x)^2)$$

$$\sigma_2 = (L_1 + L_2 \cos(x))^2$$

The above solution shows that singularity occurs at q3 = $\pi\, k$ where $k \in \mathbb{Z}$.

Computing the Jacobian Using Direct Method

```
O1 = T01(1:3, 4); R01 = T01(1:3, 1:3);
O2 = T02(1:3, 4); R02 = T02(1:3, 1:3);
O3 = T03(1:3, 4); R03 = T03(1:3, 1:3);
O4 = T04(1:3, 4); R04 = T04(1:3, 1:3);

v1 = SkewSymMat(R01 * [0; 0; 1]) * (O4-O1)
```

```
v1 = 
```
$$\begin{pmatrix} -\sin(q_1)\ (L_1 + L_3 \cos(q_2 + q_3) + L_2 \cos(q_2)) \\ \cos(q_1)\ (L_1 + L_3 \cos(q_2 + q_3) + L_2 \cos(q_2)) \\ 0 \end{pmatrix}$$

```
v2 = SkewSymMat(R02 * [0; 0; 1]) * (O4-O2)
```

```
v2 = 
```
$$\begin{pmatrix} -\cos(q_1)\, \sigma_2 \\ -\sin(q_1)\, \sigma_2 \\ \cos(q_1)\ (L_1 \cos(q_1) - \cos(q_1)\, \sigma_1) + \sin(q_1)\ (L_1 \sin(q_1) - \sin(q_1)\, \sigma_1) \end{pmatrix}$$

where

$$\sigma_1 = L_1 + L_3 \cos(q_2 + q_3) + L_2 \cos(q_2)$$

$$\sigma_2 = L_3 \sin(q_2 + q_3) + L_2 \sin(q_2)$$

```
v3 = SkewSymMat(R03 * [0; 0; 1]) * (O4-O3)
```

```
v3 = 
```

$$\begin{pmatrix} -L_3 \sin(q_2+q_3)\cos(q_1) \\ -L_3 \sin(q_2+q_3)\sin(q_1) \\ \cos(q_1)\ (\cos(q_1)\ (L_1+L_2\cos(q_2)) - \cos(q_1)\,\sigma_1) + \sin(q_1)\ (\sin(q_1)\ (L_1+L_2\cos(q_2)) - \sin(q_1)\,\sigma_1) \end{pmatrix}$$

where

$$\sigma_1 = L_1 + L_3\cos(q_2+q_3) + L_2\cos(q_2)$$

```
J_direct = simplify([v1, v2, v3])
```

```
J_direct =
```

$$\begin{pmatrix} -\sin(q_1)\,\sigma_2 & -\cos(q_1)\,\sigma_1 & -L_3\sin(q_2+q_3)\cos(q_1) \\ \cos(q_1)\,\sigma_2 & -\sin(q_1)\,\sigma_1 & -L_3\sin(q_2+q_3)\sin(q_1) \\ 0 & -\sigma_3 - L_2\cos(q_2) & -\sigma_3 \end{pmatrix}$$

where

$$\sigma_1 = L_3\sin(q_2+q_3) + L_2\sin(q_2)$$

$$\sigma_2 = L_1 + \sigma_3 + L_2\cos(q_2)$$

$$\sigma_3 = L_3\cos(q_2+q_3)$$

Verify that both DIrect and Indirect Approach are the same

```
J_indirect - J_direct
```

```
ans =
```

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

```
function T = DHtransffxn(d, alpha, r, theta) % Function for DH-Formalism
R = [cos(theta), -sin(theta), 0;
    sin(theta)*cos(alpha), cos(theta)*cos(alpha), -sin(alpha);
    sin(theta)*sin(alpha), cos(theta)*sin(alpha), cos(alpha)];
P = [d;
    -r*sin(alpha);
    r*cos(alpha)];

T = [R, P;
    0 0 0 1];
end

function S = SkewSymMat(vec) % Function for Obtaining the Skew Symmetric Matrix of a Vector
x = vec(1); y = vec(2); z = vec(3);
```

```
S = [0, -z, y;
    z, 0, -x;
    -y, x, 0];
end

function Jac = Jacobian(f, x) % Function for computing Jacobian

for f_i=1:size(f, 1)
    for x_i=1:size(x, 1)
        Jac(f_i, x_i) = diff(f(f_i), x(x_i));
    end
end

end
```