Dynamic Modelling of a 6DOF Manipulator
Final Project Report

Rishabh Biyani
UID - 114827845

December 17, 2016

Contents

List of Figures

1 Introduction

This project focused on coming up with a dynamic model of an industrial manipulator. A manipulator from the ABB Robotics was taken as a reference to obtain practical dimensions and specifications. The manipulator chosen for this project is an elbow manipulator with spherical wrist which goes by the model number **IRB- 140**. By dynamic modelling the intention is to estimate the dynamic parameters like the mass, centre of gravity and inertias of the links and to perform the Inverse Dynamics for this manipulator. Recursive computation of Newton-Euler Formulation - also know as the Luh-Paul-Walker Algorithm was used to perform the Inverse Dynamics and also compute the equations of the motion.

2 Scope & Motivation

A dynamic model that closely resembles the actual manipulator is crucial in developing effective control strategies and also for performing simulations of the manipulator. Independent joint control like PD control guarantees asymptotic stability but are too slow to be used for executing fast motions. Strategies like Inverse Dynamics and Robust Adaptive control that achieve smooth tracking of arbitrary trajectories leverage a good dynamic model. Furthermore, there is a need to be efficient in real-time computing to achieve rapid motions. Lagrangian Equations in its original form has a complexity of $O(n^4)$. A recursive computation approach like that taken with Newton-Euler formulation brings down the computation to $O(n)$ [1]. Hence, the study of this approach is important.

3 Objectives

- Derive the complete dynamic model of a Robotic Manipulator by the Newton-Euler formulation. Based on the parameters obtained in this step construct a SimMechanics simulation model of the manipulator.
- Validate the integrity of the model by matching the value of Torques as seen in the SimMechanics Model with the one obtained by solving analytically. In other words, the SimMechanics results will be used as a benchmark
- Perform the payload analysis and determine the permissible value of torques that can be applied at the joints

4 Overview of the Approach

- **Model Simplification:** The model will be simplified using simple geometries for links like a cylinder and a rectangular brick. It is assumed that links are made up of uniform density material.
- **Interpretation of the Manipulator Design:** The given IRB-140 model in the data sheet is investigated thoroughly for the link dimensions. Specific emphasis was given

to the design of spherical wrist w.r.t the positioning of the origins. Infact, the final wrist design was no longer spherical as the origins were not coincident.

- **Skeleton Model:** The 6DOF Manipulator model was created describing various frame assignments and DH convention table was formulated.
- **Forward Kinematics:** As part of the Newton-Euler Formulation it is required that we translate different coupling forces back and forth between frames. Hence, forward kinematics are performed using the DH convention.
- **Dynamic Parameter Estimation:** One of the crucial tasks was to estimate the dynamic parameters, since these are not directly available in the data sheet. The data known is the length of each link and the total mass of the manipulator (98 kg). Thus, a constrained task was adopted to obtain the dimensions of each link. The density, breadth and thickness were iteratively varied to compute the feasible dimensions using a trial and error process. The script for this part is written in MATLAB. Now, it was possible to estimate the centre of gravity of each link and the corresponding masses and the Inertia Matrix. The beauty of Newton-Euler formulation is that these remain constant irrespective of manipulator configuration.
- **Newton-Euler Formulation:** The formulation was studied properly to understand the forward and backward recursions. This algorithm is also know as the Luh-Paul-Walker algorithm. The algorithm in its original form is better presented in [1]. The algorithm was written in Mathematica.
- **Creation of Simulation Model**: The model with its simplified geometry is created directly in SimMechanics. This had to be done after dynamic parameter estimation step too to just confirm that manipulator looks reasonable with a aesthetically nice geometry. Then, to perform the payload analysis the simmechanics model was modified to add a point mass at the end of the manipulator
- **Verification of the Kinematic Model:** This step was carried out after DH parameter extraction. Verification was done using the SimMechanics model by looking at various frames and also commanding it to a specified position. This result was also verified using the on-paper calculations.
- **Verification of the Dynamic Model:** This is accomplished by computing the joint torques for a given joint position say q and setting angular velocities and angular acceleration at the joints to be zero. This will give the value of torques required to maintain the manipulator at a particular configuration under the influence of the gravity load. The above values are cross-checked with what is computed through the sensor values at the revolute joint in SimMechanics simulation. It is found that these results are closely matching and hence, the correctness of the model is verified
- **Determination of Permissible Torques:** For the specified max payload in the datasheet, maximum permissible torques at the joint were determined. These were also verified with the corresponding torques in the simmechanics model. The results were in very good agreement.

This entire approach will be elaborated in the upcoming sections.

5 Newton-Euler Method

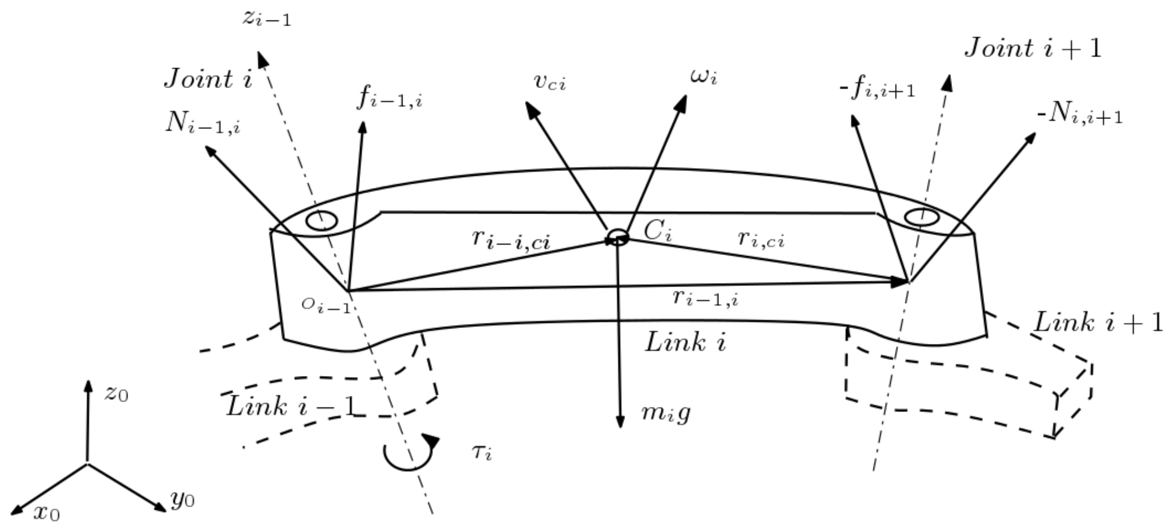


Figure 1: Free Body Diagram of Link i

Newton-Euler method forms the basis of the recursive computation algorithm that will be discussed. In this method we draw the free body diagram of each link as shown in the figure 1 and write down the equations of motion [1]. The equations described described in this section are based on the textbook by H.Asada et. al [1]. The dynamic equations of a rigid body can be represented by two equations: one describing the translational motion of the centre of mass and the other describes rotational motion about the centre of mass. The former is Newton's equation of motion for a mass particle and the latter is called the Euler's equation of motion. These equations are always written in an inertial frame of reference. Hence, they always have a inertial force/torque term in the equation. We can write the equations from the diagram as follows:

$$f_{i-1,i} - f_{i,i+1} + m_i g - m_i \dot{v_{ci}} = 0 \tag{1}$$

$$N_{i-1,i} - N_{i,i+1} + r_{i,ci} \times f_{i,i+i} - r_{i-1,ci} \times f_{i-1,i} - I_i \dot{\omega_i} - \omega_i \times (I_i \omega_i) = 0 \tag{2}$$

$$for \quad i = 1.....n$$

Where, in equation 1, f is the coupling force acting between the links - one at the back and one at the front. $\dot{v_{ci}}$ is the acceleration of the centre of mass of the link as seen from

[1]A good way to remember the axis and link naming nomenclature is to say that the z_{i-1} axis of the link $i-1$ holds the actuator of the link i and is responsible for the motion of joint i. In a similar way we can comment about the relation between link i and link $i+1$

the inertial frame. Whereas, in equation 2 N is the rotational coupling force between the links. I_i is the Rotational Inertia of the link as seen from the Inertial frame. $I\omega_i$ is the angular momentum of the Link and because the Inertia Tensor varies with the orientation of the Link, we also have a Gyroscopic term in the form of $\omega \times (I_i\omega_i)$. An important point to note here is that τ_i - the actuator torque as shown in the figure 1, only bears the component of the normal coupling forces acting along the rotational axis z_{i-1}. The rest of the components are beard by the structure of the manipulator. Thus,

$$\tau_i = b_i^T N_{i-1,i} \tag{3}$$

Where, $b_i = z_{i-1}$. Generally, this is the last step in the recursive computation process of Inverse Dynamics. When we will use the equation 3 in the recursive formulation, the vectors b_i and $N_{i-1,i}$ are going to be expressed in the link's own frame of reference instead of the ground frame. This will be more clear in the next section.

5.1 Recursive Computation

The key concept is formulating dynamic equations in a *recursive* form so that the computation can be accomplished from one link of the Manipulator arm to another. The first phase of the recursive Newton-Euler formulation is to determine all the kinematic variables like the linear and angular velocities and accelerations of each link that are needed to evaluate the Newton-Euler equations. The algorithm starts with the first link. Given the joint displacements q_1 and the joint velocity and acceleration $\dot{q_1}$ and $\ddot{q_1}$, the linear and angular velocities and accelerations of the Centroid C_1 is determined. Then using the velocities and accelerations obtained for the first link, we compute the velocity and acceleration of the second link with the data specified by q_1, $\dot{q_1}$ and $\ddot{q_1}$. This procedure is repeated until all the centroidal velocities and accelerations, as well as the angular velocities and accelerations are determined for all the links involved.

The second phase involves the computation of the Newton-Euler equations with the computed kinematic variables to determine the joint torques. This has to be performed starting from the last link all the way back to the first link. Like, if we rewrite the equation 1 for the last link. We have,

$$f_{n-1,n} = f_{n,n+1} - m_n g + m_n a_{cn} \tag{4}$$

where, $f_{n,n+1}$ is the linear endpoint force. This is a known entity [2] and when plugged in we can find out $f_{n-1,n}$. Similar, is the case with the Euler equation 2. Thus, we move recursively backwards to obtain the values of all the coupling forces. Once, the coupling force and moment of each joint is determined, the joint torque is computed using the equation 3.

5.1.1 Moving Coordinates

The velocity and acceleration of link i depends on the motion of the link $i-1$. It also depends on the relative motion of this link as a result of the actuator on the link $i-1$. This is essentially the situation in the first phase of the computation. What we are dealing with here is the relative motions defined in a moving co-ordinate frame. Thus, we will need some results about the moving co-ordinate frames. These are enlisted in this section.

[2]For a known payload in this case

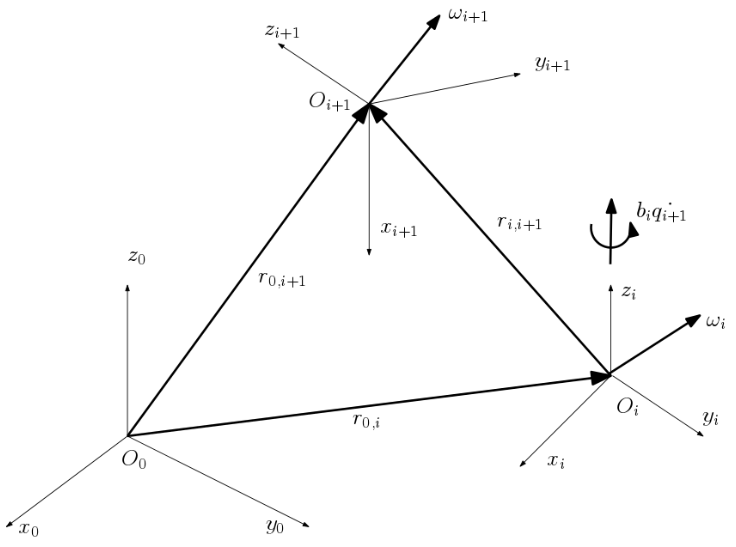


Figure 2: Motion relative to a moving reference frame

Figure 2 shows a general situation where we are describing the motion of the origin of the frame $i+1$ with reference to the frame i which has some angular velocity w.r.t to the inertial frame with Origin O_0. The frames are essentially the frames attached to the links $i+1$ and link i. The vector $r_{i,i+1}$ is a vector that is expressed in a moving reference frame. The velocity of the origin $i+1$ as seen from the inertial frame is written using the equation:

$$v_{i+1} = v_i + \frac{dr_{i,i+1}}{dt}|_{rel.} + \omega_i \times r_{i,i+1} \tag{5}$$

where,

$$\begin{aligned} v_{i+1} &= \text{Velocity of the origin of the reference frame } i+1 \\ v_i &= \text{Velocity of the origin of the frame } i \\ \frac{dr_{i,i+1}}{dt}|_{rel.} &= \text{This is the velocity contribution solely due to the actuator on the link } i \\ \omega_i &= \text{Angular velocity of the frame } i \text{ itself.} \end{aligned}$$

In general, the time rate of change of any arbitrary vector that moves relative to a rotating coordinate frame is computed using differential operator symbolically denoted by

$$\frac{d}{dt}|_{fixed} = \frac{d}{dt}|_{rel.} + \omega\times \tag{6}$$

Now, the expression for acceleration can be obtained by differentiating the equation 5 w.r.t to the inertial frame. Keeping in mind that:

$$\frac{d}{dt}|_{fixed}\left(\frac{dr_{i,i+1}}{dt}|_{rel.}\right) = \frac{d^2 r_{i,i+1}}{dt^2} + \omega_i \times \frac{dr_{i,i+1}}{dt}$$

Where, the suffix *fixed* means that the time rate of change is viewed from a fixed reference frame. Thus, what we have is:

$$a_{i+1} = a_i + \frac{d^2 r_{i,i+1}}{dt^2}|_{rel.} + \dot{\omega_i} \times r_{i,i+1} + 2\omega_i \times \frac{dr_{i,i+1}}{dt}|_{rel.} + \omega_i \times (\omega_i \times r_{i,i+1}) \tag{7}$$

The equation 5, 6 and 7 comes in handy to derive the equations for the forward recursion.

5.1.2 Luh-Walker-Paul's Algorithm

The recursive computation algorithm of the Newton-Euler Dynamic equation was original developed by Luh, Walker and Paul, 1980. We will write the equations for a manipulator with all joints as revolute.

- The frame $i+1$ is rotated at an angular velocity $\dot{q_{i+1}} b_i$ and with an angular acceleration $\ddot{q_{i+1}} b_i$ about the moving co-ordinate frame attached to the Link i. The angular velocity of link $i+1$ referred to the base frame is given by

  $$\omega_{i+1} = \omega_i + \dot{q}_{i+1} b_i \tag{8}$$

  By taking the time derivative we can obtain an expression for the angular acceleration, keeping in mind that the second term is defined as a vector relative to the moving coordinate frame. Thus, using the equation 6. We have[3]

  $$\dot{\omega_{i+1}} = \dot{\omega_i} + \ddot{q}_{i+1} b_i + \omega_i \times \dot{q}_{i+1} b_i \tag{9}$$

- The recursive computations for linear velocities are derived from equations 5 and 7. In this case we have

  $$\frac{dr_{i,i+1}}{dt}|_{rel.} = \dot{q}_{i+1} b_i \times r_{i,i+1} \tag{10}$$

  $$\frac{d^2 r_{i,i+1}}{dt^2}|_{rel.} = \ddot{q}_{i+1} b_i \times r_{i,i+1} + \dot{q}_{i+1} b_i \times (\dot{q}_{i+1} b_i \times r_{i,i+1}) \tag{11}$$

  Substituting, equation 8 and 10 in equation 5, we get

  $$v_{i+1} = v_i + \omega_{i+1} \times r_{i,i+1} \tag{12}$$

  Further, substituting equation 9 and 11 in equation 7 and using the identity of vector triple product, we obtain

  $$a_{i+1} = a_i + \dot{\omega}_{i+1} \times r_{i,i+1} + \omega_{i+1} \times (\omega_{i+1} \times r_{i,i+1}) \tag{13}$$

[3]The equation 9 for angular acceleration of link $i+1$ in Spong et. al [3] is incorrect. In the term $\omega_i \times \dot{q}_{i+1} b_i$, ω_{i+1} is used instead of ω_i

- The Newton-Euler Equations are expressed in terms of the centroidal accelerations and at this point of time we have the formulation expressed with respect to the origin of the coordinate frame of each link. Therefore, we have to transform these acceleration values. The centroidal velocity is given by

$$v_{ci} = v_i + \omega_i \times r_{i,ci} \tag{14}$$

  Differentiating the above equation using the differential operator defined in the equation 6, we get the centroidal acceleration:

$$\dot{v}_{ci} = \dot{v}_i + \dot{\omega}_i \times r_{i,ci} + \omega_i \times (\omega_i \times r_{i,ci}) \tag{15}$$

- Till now, we have derived all the equations with reference to the base frame. The problem with this is that the Inertia Matrix is a configuration dependent Matrix and thus I_i has to be obtained for each arm configuration. This requires extra computation time. To surmount this problem, in the Luh-Paul-Walker Algorithm we express all the variables and the parameters in the link coordinates. This, makes the Inertia Matrix invariable to the configuration and also computationally efficient. To express the equation in link co-ordinates, we simply replace v_i, ω_i and the other variables with the ones referred to that link coordinate frame. Further, when a variable referred to frame i is involved in an equation referred to frame $i+1$, it is first premultiplied by $\mathbf{R_i^{i+1}}$, so that all the variables are expressed with reference to frame $i+1$. In link co-ordinates r_i, ci and $r_i, i+1$ are constant[4] since they are fixed to the link body. These steps will be performed explicitly when we apply this algorithm to the current problem in the section 8

6 Modelling Setup

The following subsections explain the modelling of the 6DOF manipulator

6.1 Link Parameters

A very rough estimate of the dimensions were extracted from the manipulator. Basically an elbow manipulator with spherical wrist is required for the current problem. The spherical wrist alone is built up by three single degree-of-freedom revolute joints, where the rotation axes intersect in the wrist center point. Thus the two links in between(l_4 and l_5) will have zero length. This means that they carry no mass or weight. Clearly, this is not true for a manipulator in general and this links must have same weight. Also, from the weight distribution point of view, this is not very practical. Because, then all the weight of the manipulator will get concentrated at the the three links(and the base). To get a more feasible model, the spherical wrist consideration is relaxed and rather l_3, l_5 are made equal to zero. This gives a more uniform weight distribution across the manipulator. Further, the link offset distance of $70mm$ is neglected and out of $352mm$ it is assumed that $152mm$ belongs to the base of the manipulator which is stationary. Thus, From the diagram above, and the stated assumptions we can extract following dimensions for the links

[4]For the case of revolute joints

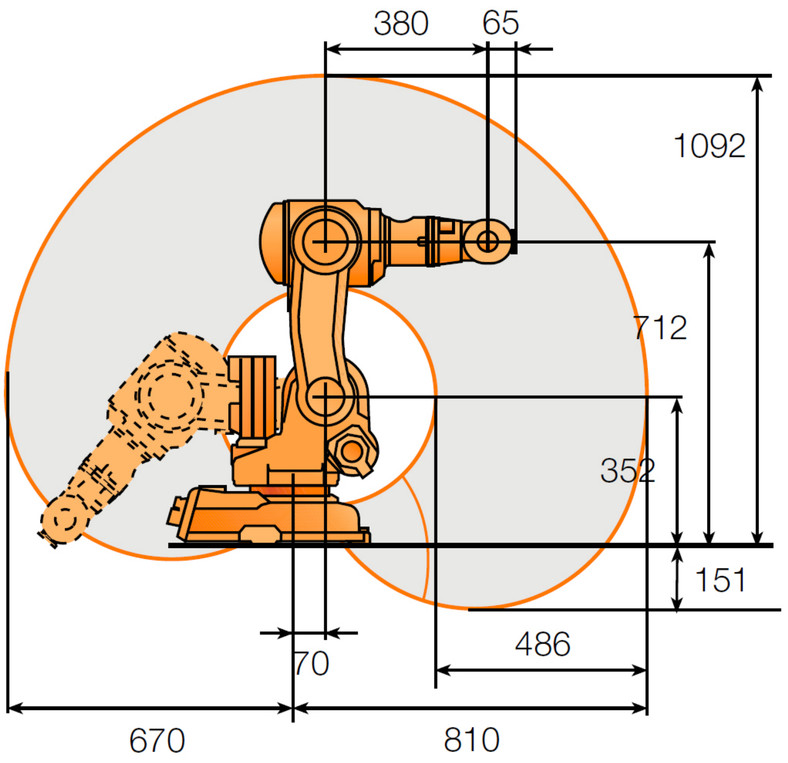


Figure 3: Diagram showing Link Dimensions of the Industrial Manipulator where all dimensions are in mm [2]

Link	**Length** (m)
l_1	0.2
l_2	(0.712 - 0.352) = 0.36
l_3	0
l_4	0.38
l_5	0
l_6	0.065

6.2 Symbolic Representation and DH Table

As per the frame assignment in the figure 4 and our convention to represent various link dimensions as per the figure 1, we can write following vector representations for the link

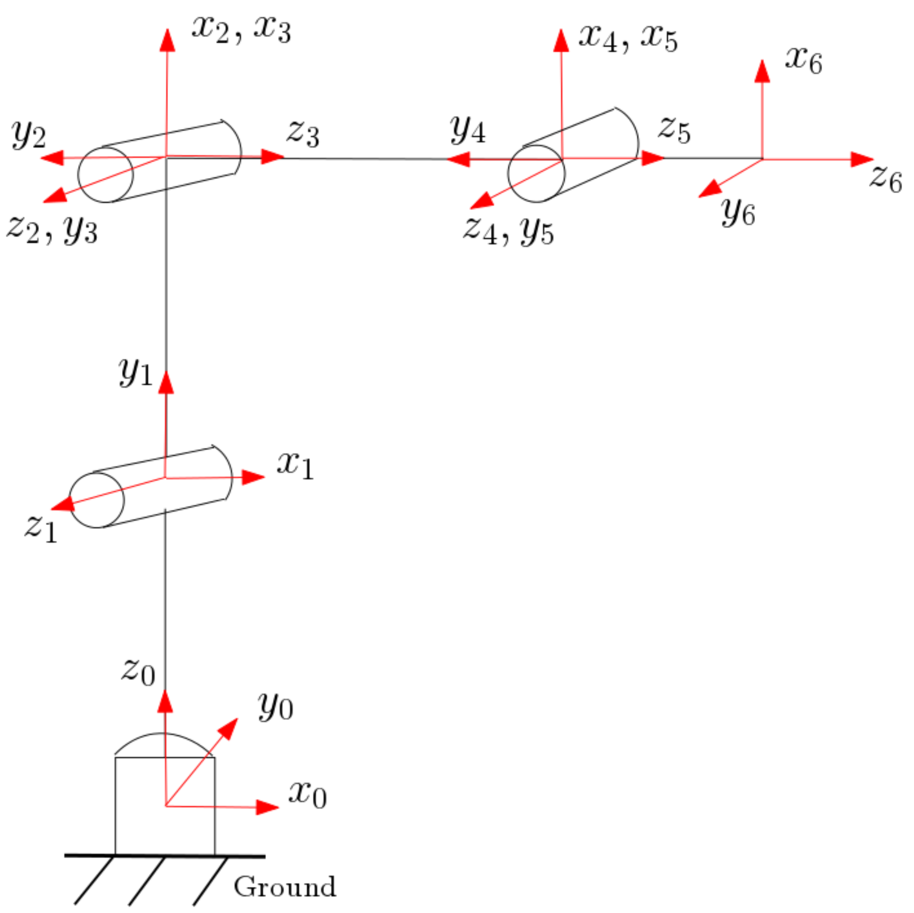


Figure 4: Frame Assignment for the DH table

lengths and its Centre of Mass position.

$$r_{01} = [0 \quad 0.2 \quad 0]^T \tag{16}$$
$$r_{12} = [0.36 \quad 0 \quad 0]^T \tag{17}$$
$$r_{23} = [0 \quad 0 \quad 0]^T \tag{18}$$
$$r_{34} = [0 \quad -0.38 \quad 0]^T \tag{19}$$
$$r_{45} = [0 \quad 0 \quad 0]^T \tag{20}$$
$$r_{56} = [0 \quad 0 \quad 0.065]^T \tag{21}$$
$$r_{1c1} = [0 \quad -0.1 \quad 0]^T \tag{22}$$
$$r_{2c2} = [-0.18 \quad 0 \quad 0]^T \tag{23}$$
$$r_{3c3} = [0 \quad 0 \quad 0]^T \tag{24}$$
$$r_{4c4} = [0 \quad 0.19 \quad 0]^T \tag{25}$$
$$r_{5c5} = [0 \quad 0 \quad 0]^T \tag{26}$$
$$r_{6c6} = [0 \quad 0 \quad -0.0325] \tag{27}$$
$$\tag{28}$$

Also, we can write the equations for $r_{i-1,ci}$ using the rule of vector addition as follows:-

$$\begin{cases} r_{0c1} = r_{01} + r_{1c1} \\ r_{1c2} = r_{12} + r_{2c2} \\ r_{2c3} = r_{23} + r_{3c3} \\ r_{3c4} = r_{34} + r_{4c4} \\ r_{4c5} = r_{45} + r_{5c5} \\ r_{5c6} = r_{56} + r_{6c6} \end{cases} \tag{29}$$

Based on the figure 4, the DH table is written down as shown in Table 1. A point with the DH frame assignment is to realize that the frames assigned are the body fixed frames to be used in conjunction with the Newton-Euler formulation as will be seen in the later sections. These frames are going to form the basis for writing the co-ordinates for the link lengths and the their respective centre of masses. Furthermore, the way the Moment of Inertia Matrix is defined across different axes also depends on the co-ordinate axes we assign for DH parameter extraction.

Link	a	α	d	θ
1	0	$\frac{\pi}{2}$	l_1	θ_1
2	l_2	0	0	$\frac{\pi}{2} + \theta_2$
3	0	$\frac{\pi}{2}$	0	θ_3
4	0	$\frac{-\pi}{2}$	l_4	θ_4
5	0	$\frac{\pi}{2}$	0	θ_5
6	0	0	l_6	θ_6

Table 1: DH Parameters

6.3 Link Mass, Moment of Inertia & Other Dimensions

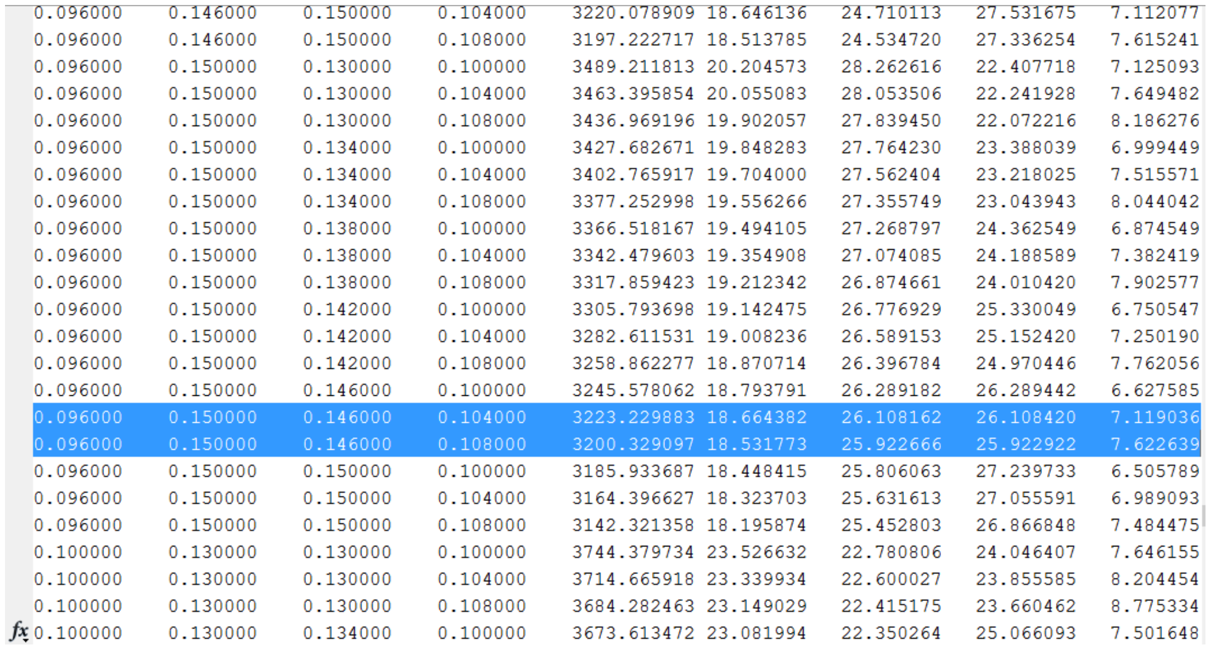
As stated before the link mass extraction process is done with the constraint of overall weight of the Manipulator being $98Kg$. A data obtained from IRB-140 guide[2]. This section explains the iterative procedure used.

The candidate for geometry of links had to be chosen between the cylinder and the rectangular brick. These two geometries were considered due to couple of reasons. First, these are symmetric geometries, making their calculation for moment of inertias simpler and the resulting Inertia Matrix diagonal. Second, these geometries can be obtained directly in SimMechanics, eliminating the need to do any CAD modelling. At the time of performing the iterative process the approach adopted was to go back and forth between a MATLAB script and the simmechanics model of the manipulator, until a aesthetically pleasing manipulator is obtained. It was initially thought that the same geometry will be used for all the links, where more inclination was towards the rectangular brick because of its greater flexibilty in modelling in SimMechanics. But one important observation was that the cylinder links pack more mass in a lesser length for a given density. So, for the first link(l_1), if a rectangular brick was chosen, the breadth and height of the geometry had to be very close to its actual length of $0.2m$ in order to satisfy the value of the mass that has to be carried by this link. This didn't look like it resembled the manipulator and hence a cylindrical geometry was chosen for the first link. Similar, is the case for the last link l_6. Cylindrical geometry was chosen for this too. Thus, a mixture of both these geometries was finally employed.

To obtain the dimensions and the masses, there were a few simplifying assumptions :-

- The density of all the links was assumed to be equal. This assumption could very well be a fair assumption
- For rectangular geometry of the Link 2 and Link 4 the breadth and height of the brick was assumed to be the same.
- Mass of the immobile foundation or the base is taken to be $\approx 20Kg$.

The density, the radius of the two cylindrical links and the breadth and height of the two rectangular links are the chosen variables which are going to be varied such that the total weight sums to $98kg$. The MATLAB script to do this is located in A.1. Out of the different combinations that the script laid out, a snap showing some of the outputs is shown in figure 5. The two highlighted combinations were considered due to uniformity of the masses of the links and the density. Finally, the one with density of $\approx$ 3200 was chosen as the final configuration. The density of $3200\frac{kg}{m^3}$ looks practical considering the fact that the density of steel is $\approx$ 2.5 times this density.



```
0.096000   0.146000   0.150000   0.104000   3220.078909 18.646136   24.710113   27.531675   7.112077
0.096000   0.146000   0.150000   0.108000   3197.222717 18.513785   24.534720   27.336254   7.615241
0.096000   0.150000   0.130000   0.100000   3489.211813 20.204573   28.262616   22.407718   7.125093
0.096000   0.150000   0.130000   0.104000   3463.395854 20.055083   28.053506   22.241928   7.649482
0.096000   0.150000   0.130000   0.108000   3436.969196 19.902057   27.839450   22.072216   8.186276
0.096000   0.150000   0.134000   0.100000   3427.682671 19.848283   27.764230   23.388039   6.999449
0.096000   0.150000   0.134000   0.104000   3402.765917 19.704000   27.562404   23.218025   7.515571
0.096000   0.150000   0.134000   0.108000   3377.252998 19.556266   27.355749   23.043943   8.044042
0.096000   0.150000   0.138000   0.100000   3366.518167 19.494105   27.268797   24.362549   6.874549
0.096000   0.150000   0.138000   0.104000   3342.479603 19.354908   27.074085   24.188589   7.382419
0.096000   0.150000   0.138000   0.108000   3317.859423 19.212342   26.874661   24.010420   7.902577
0.096000   0.150000   0.142000   0.100000   3305.793698 19.142475   26.776929   25.330049   6.750547
0.096000   0.150000   0.142000   0.104000   3282.611531 19.008236   26.589153   25.152420   7.250190
0.096000   0.150000   0.142000   0.108000   3258.862277 18.870714   26.396784   24.970446   7.762056
0.096000   0.150000   0.146000   0.100000   3245.578062 18.793791   26.289182   26.289442   6.627585
0.096000   0.150000   0.146000   0.104000   3223.229883 18.664382   26.108162   26.108420   7.119036
0.096000   0.150000   0.146000   0.108000   3200.329097 18.531773   25.922666   25.922922   7.622639
0.096000   0.150000   0.150000   0.100000   3185.933687 18.448415   25.806063   27.239733   6.505789
0.096000   0.150000   0.150000   0.104000   3164.396627 18.323703   25.631613   27.055591   6.989093
0.096000   0.150000   0.150000   0.108000   3142.321358 18.195874   25.452803   26.866848   7.484475
0.100000   0.130000   0.130000   0.100000   3744.379734 23.526632   22.780806   24.046407   7.646155
0.100000   0.130000   0.130000   0.104000   3714.665918 23.339934   22.600027   23.855585   8.204454
0.100000   0.130000   0.130000   0.108000   3684.282463 23.149029   22.415175   23.660462   8.775334
0.100000   0.130000   0.134000   0.100000   3673.613472 23.081994   22.350264   25.066093   7.501648
```

Figure 5: A snap showing output of the MATLAB script

Geometry	Link i	Dimensions $(l_i \times b_i \times h_i)$	Dimensions $(r_i \times h_i)$	Mass (kg)
Cylindrical	1	-	0.098×0.2	19.31
Rectangular	2	$0.36 \times 0.15 \times 0.15$	-	25.92
Rectangular	4	$0.38 \times 0.146 \times 0.146$	-	25.92
Cylindrical	6	-	0.108×0.065	7.62

Table 2: Link Parameters

Thus, the chosen link parameters are rewritten in Table 2.

Again, by looking at how the axes are attached to link from figure 4, we can write the Mass Moment of Inertia of each link as follows:

$$I_1 = \begin{bmatrix} \frac{1}{12}m_1\left({h_1}^2 + 3{r_1}^2\right) & 0 & 0 \\ 0 & \frac{m_1{r_1}^2}{2} & 0 \\ 0 & 0 & \frac{1}{12}m_1\left({h_1}^2 + 3{r_1}^2\right) \end{bmatrix} \tag{30}$$

$$I_2 = \begin{bmatrix} \frac{1}{12}m_2\left({b_2}^2 + {h_2}^2\right) & 0 & 0 \\ 0 & \frac{1}{12}m_2\left({h_2}^2 + {l_2}^2\right) & 0 \\ 0 & 0 & \frac{1}{12}m_2\left(b2^2 + l2^2\right) \end{bmatrix} \tag{31}$$

$$I_3 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{32}$$

$$I_4 = \begin{bmatrix} \frac{1}{12}m_4\left({b_4}^2 + {l_4}^2\right) & 0 & 0 \\ 0 & \frac{1}{12}m_4\left({b_4}^2 + {h_4}^2\right) & 0 \\ 0 & 0 & \frac{1}{12}m_4\left({h_4}^2 + {l_4}^2\right) \end{bmatrix} \tag{33}$$

$$I_5 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{34}$$

$$I_6 = \begin{bmatrix} \frac{1}{12}m_6\left({h_6}^2 + 3{r_6}^2\right) & 0 & 0 \\ 0 & \frac{1}{12}m_6\left({h_6}^2 + 3{r_6}^2\right) & 0 \\ 0 & 0 & \frac{m_6{r_6}^2}{2} \end{bmatrix} \tag{35}$$

6.4 Verification of the Kinematic Model

The DH and the pertaining frame assignments are crucial components in the process of validating the dynamic model. It is imperative that while constructing the Simscape model the frame assignment should match with what was constructed in figure 4 for any chance of validating the dynamic model with Simscape as the reference. This should be obvious by the fact that in Newton-Euler formulation various configuration independent parameters are defined in the body-fixed frame which as stated before is nothing but the link attached frame used in the DH convention. This was learnt the hard way when even after few revisions of the

Newton-Euler Formulation the torque values didn't match. Therefore, it is recommended to construct a simscape model to verify the frames as the next step in the process. Figure 6 shows the simscape model constructed using the parameters mentioned in Table 1. This Manipulator has six revolute joints. The frame in the centre of of each link is the default body-fixed frame in the simscape and this is a sort of handle for its manipulation. The simscape block diagram for this manipulator is located in Appendix A.2

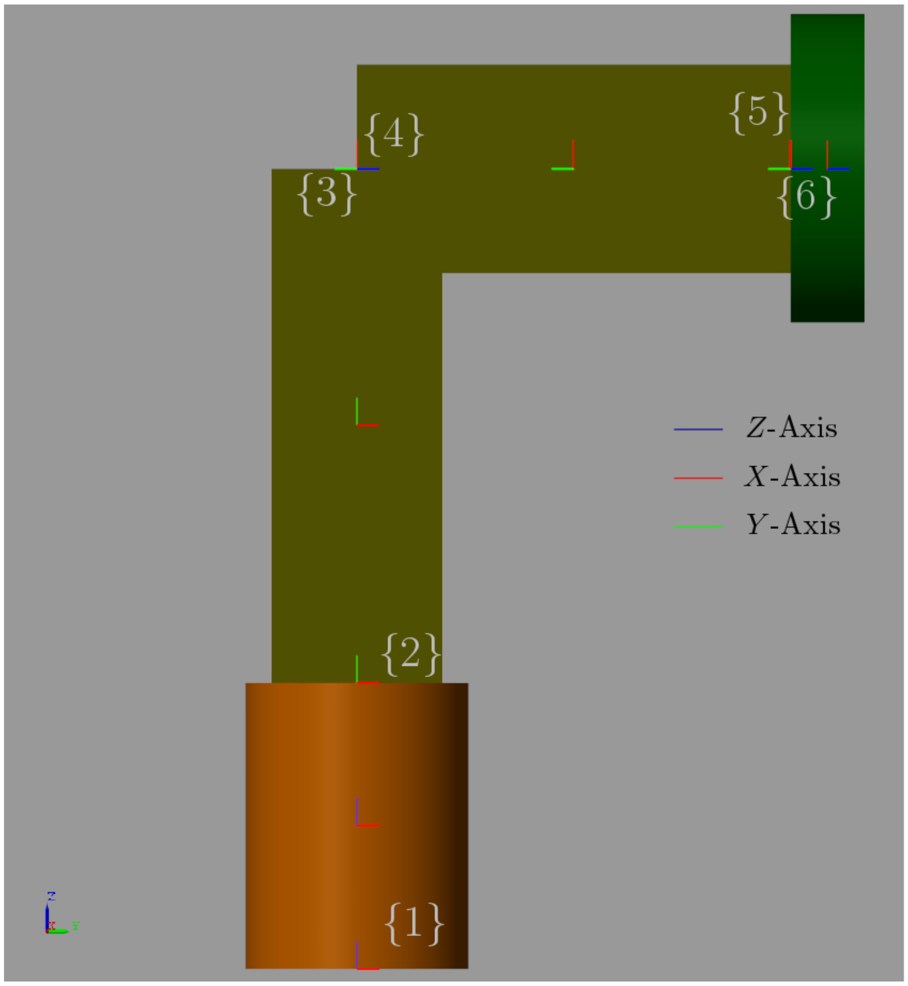


Figure 6: Frames assigned in the simscape model are in agreement with that assigned in figure 1

Where, as stated before Link 3 and Link 5 are modelled as zero length links.

7 Forward kinematics

The equation for forward kinematics using DH convention is written as [[3]]

$$\begin{cases} A_i = Rot_{z,\theta_i} Trans_{z,d_i} Trans_{x,a_i} Rot_{x,\alpha_i} \\ \Rightarrow \begin{pmatrix} \cos(\theta) & -\sin(\theta)\cos(\alpha) & \sin(\theta)\sin(\alpha) & a\cos(\theta) \\ \sin(\theta) & \cos(\theta)\cos(\alpha) & -\cos(\theta)\sin(\alpha) & a\sin(\theta) \\ 0 & \sin(\alpha) & \cos(\alpha) & d \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{cases} \tag{36}$$

Thus, we can compute the above matrix for each link as per the parameters in 1. What we essentially need is the rotation Matrix from these A matrices, which is nothing but the first three rows and columns. Thus, evaluating the above equation for each DH param we have,

$$R_{01} = \begin{pmatrix} \cos(q_1(t)) & 0 & \sin(q_1(t)) \\ \sin(q_1(t)) & 0 & -\cos(q_1(t)) \\ 0 & 1 & 0 \end{pmatrix} \tag{37}$$

$$R_{12} = \begin{pmatrix} -\sin(q_2(t)) & -\cos(q_2(t)) & 0 \\ \cos(q_2(t)) & -\sin(q_2(t)) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{38}$$

$$R_{23} = \begin{pmatrix} \cos(q_3(t)) & 0 & \sin(q_3(t)) \\ \sin(q_3(t)) & 0 & -\cos(q_3(t)) \\ 0 & 1 & 0 \end{pmatrix} \tag{39}$$

$$R_{34} = \begin{pmatrix} \cos(q_4(t)) & 0 & -\sin(q_4(t)) \\ \sin(q_4(t)) & 0 & \cos(q_4(t)) \\ 0 & -1 & 0 \end{pmatrix} \tag{40}$$

$$R_{45} = \begin{pmatrix} \cos(q_5(t)) & 0 & \sin(q_5(t)) \\ \sin(q_5(t)) & 0 & -\cos(q_5(t)) \\ 0 & 1 & 0 \end{pmatrix} \tag{41}$$

$$R_{56} = \begin{pmatrix} \cos(q_6(t)) & -\sin(q_6(t)) & 0 \\ \sin(q_6(t)) & \cos(q_6(t)) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{42}$$

Where, $q_1(t), q_2(t), q_3(t), q_4(t), q_5(t), q_6(t)$ are the selected generalized coordinates for each revolute joint. Now, the Rotation Matrices can be solved for a direct relation between the each link and the base frame as follows:

$$R_{02} = R_{01}R_{12} \tag{43}$$
$$R_{03} = R_{02}R_{23} \tag{44}$$
$$R_{04} = R_{03}R_{34} \tag{45}$$
$$R_{05} = R_{04}R_{45} \tag{46}$$
$$R_{06} = R_{05}R_{56} \tag{47}$$

Also, in the computations of the forward recursion in the Newton-Euler Method we need a vector b_i. This component expresses the rotation axis z_{i-1} in the coordinate frame of link i. The z_{i-1} axis is simply the z-axis of the body-attached frame on the $i-1$ link. The link i rotates about the z_{i-1} axis as can be seen in the figure 1. In every $i-1$ frame the z_{i-1} axis will have components as $z_0 = [0 \quad 0 \quad 1]^T$. Thus, we can use this aspect and obtain b_i as follows:-

$$b_1 = (R_{01})^T z_0 = [0,1,0]^T \tag{48}$$
$$b_2 = (R_{02})^T R_{01} z_0 = [0,0,1]^T \tag{49}$$
$$b_3 = (R_{03})^T R_{02} z_0 = [0,1,0]^T \tag{50}$$
$$b_4 = (R_{04})^T R_{03} z_0 = [0,-1,0]^T \tag{51}$$
$$b_5 = (R_{05})^T R_{04} z_0 = [0,1,0]^T \tag{52}$$
$$b_6 = (R_{06})^T R_{05} z_0 = [0,0,1]^T \tag{53}$$
$$\tag{54}$$

Clearly, the b_i components are constant and independent of the manipulator configurations. One could argue that, this is going to depend on how DH frames are assigned for the Manipulator. If b_i were configuration dependent we will have to compute this at every step of the forward recursion and it will make the computation slightly expensive. This will be more clear in the section 5.1.2. Thus, DH frame assignment also play a role in deciding the efficiency of the recursive algorithm. A constant value of the b_i vectors is always a better choice.

8 Dynamics of the 6DOF Manipulator

We apply the recursive algorithm described in the last section to the Manipulator under consideration. The algorithm is applied for the two cases. First is when there is no load acting on the last link and second is when there is a Payload attached at the end of the last link. The only difference in the computation chain is the value we attach to the end of the last link i.e. $f_{n,n+1}$ and $N_{n,n+1}$. In the former case both these are zero and in the later case $f_{n,n+1} = m_p g_6$ and $N_{n,n+1} = 0$. Where, m_p is the mass of the payload = 6 kg as obtained from the max permissible load in the datasheet [2]. g_6 is the gravity vector expressed in the frame of Link 6. Equations for the case when there is no payload are presented here. The values to be used for all the parameters used here are defined in the section 7.

Forward Recursion

Link 1

The initial conditions are:

$$\omega_0 = [0,0,0]^T$$
$$\alpha_0 = [0,0,0]^T$$
$$a_0 = [0,0,0]^T$$

The angular velocity and acceleration of the first link can be written as per the equation 8 and 9

$$\begin{aligned}\omega_1 &= b_1 q_1'(t)\\ \alpha_1 &= b_1 q_1''(t)\end{aligned}$$

Similarly, the acceleration for the end-point of the link and its centre of mass can be written from equation 13 and 15 as -

$$\begin{aligned}a_1 &= \alpha_1 \times r_1 + \omega_1 \times (\omega_1 \times r_1)\\ a_{\mathrm{c}1} &= a_1 + \alpha_1 \times r_{1\mathrm{c}1} + \omega_1 \times (\omega_1 \times r_{1\mathrm{c}1})\end{aligned}$$

Link 2

Substituting similar equations as for Link 1 the angular velocity, angular acceleration, end-point acceleration and centroidal acceleration for the Link 2 can be written as -

$$\begin{aligned}\omega_2 &= b_2 q_2'(t) + (R_{12})^T \omega_1\\ \alpha_2 &= \left((R_{12})^T \omega_1\right) \times (b_2 q_2'(t)) + b_2 q_2''(t) + (R_{12})^T \alpha_1\\ a_2 &= (R_{12})^T a_1 + \alpha_2 \times r_{12} + \omega_2 \times (\omega_2 \times r_{12})\\ a_{c2} &= a_2 + \alpha_2 \times r_{2c2} + \omega_2 \times (\omega_2 \times r_{2c2})\end{aligned}$$

Link 3

Writing equations for Link 3

$$\begin{aligned}\omega_3 &= b_3 q_3'(t) + (R_{23})^T \omega_2\\ \alpha_3 &= \left((R_{23})^T \omega_2\right) \times (b_3 q_3'(t)) + b_3 q_3''(t) + (R_{23})^T \alpha_2\\ a_3 &= (R_{23})^T a_2\\ a_{c3} &= a_3\end{aligned}$$

Link 4

Writing equations for Link 4

$$\begin{aligned}\omega_4 &= b_4 q_4'(t) + (R_{34})^T \omega_3\\ \alpha_4 &= \left((R_{34})^T \omega_3\right) \times (b_4 q_4'(t)) + b_4 q_4''(t) + (R_{34})^T \alpha_3\\ a_3 &= (R_{23})^T a_2 + \alpha_3 \times r_{23} + \omega_3 \times (\omega_3 \times r_{23})\\ a_{c3} &= a_3 + \alpha_3 \times r_{3c3} + \omega_3 \times (\omega_3 \times r_{3c3})\end{aligned}$$

Link 5

Writing equations for Link 5

$$\begin{aligned}\omega_5 &= b_5 q_5'(t) + (R_{45})^T \omega_3\\ \alpha_5 &= \left((R_{45})^T \omega_4\right) \times (b_5 q_5'(t)) + b_5 q_5''(t) + (R_{45})^T \alpha_4\\ a_5 &= (R_{45})^T a_4\\ a_{\mathrm{c}5} &= a_5\end{aligned}$$

Link 6
Writing equations for Link 6

$$
\begin{aligned}
\omega_6 &= b_6 q_6'(t) + (R_{56})^T \omega_5 \\
\alpha_6 &= \left((R_{56})^T \omega_5\right) \times (b_6 q_6'(t)) + b_6 q_6''(t) + (R_{56})^T \alpha_5 \\
a_6 &= (R_{56})^T a_5 + \alpha_6 \times r_{56} + \omega_6 \times (\omega_6 \times r_{56}) \\
a_{\mathrm{c}6} &= a_6 + \alpha_6 \times r_{6c6} + \omega_6 \times (\omega_6 \times r_{6c6})
\end{aligned}
$$

Notice that, even though the Link 3 and Link 5 are modelled as zero length links they cannot be excluded from the forward recursion. They also possess some angular and linear accelerations.

Backward Recursion

Note that, the equations contain the gravity term. This has to expressed in the Link frame as well. Therefore,

$$g_0 = [0 \quad 0 \quad -9.81]$$

To express this in the the Link Frame i, we simply premultiply by R_{0i}^T. Also, The coupling force(f) and coupling moments(N) exerted on the link are calculated using the equation 1 and 2. Thus, we can calculate these forces for each of the link as follows.

Link 6
The end-point conditions are $f_{6,7} = 0$ and $N_{6,7} = 0$.

$$
\begin{aligned}
g_6 &= (R_6)^T g_0 \\
f_{56} &= m_6 a_{\mathrm{c}6} - g_6 m_6 \\
N_{56} &= I_6 \alpha_6 + r_{5c6} \times f_{56} + \omega_6 \times (I_6 \omega_6)
\end{aligned}
$$

Link 5
Writing equations for Link 5

$$
\begin{aligned}
g_5 &= (R_{05})^T g_0 \\
f_{45} &= R_{56} f_{56} \\
N_{45} &= R_{56}(N_{56}) + \omega_5 \times (I_5 \omega_5)
\end{aligned}
$$

Link 4
Writing equation for Link 4

$$
\begin{aligned}
g_4 &= (R_4)^T g_0 \\
f_{34} &= m_4 a_{\mathrm{c}4} + R_{45} f_{45} - g_4 m_4 \\
N_{34} &= I_4 \alpha_4 - r_{4c4} \times (R_{45} f_{45}) + r_{3c4} \times f_{34} + R_{45} N_{45} + \omega_4 \times (I_4 \omega_4)
\end{aligned}
$$

Link 3
Writing equations for Link 3

$$
\begin{aligned}
g_3 &= (R_3)^T g_0 \\
f_{23} &= R_{34} f_{34} \\
N_{23} &= R_{34} N_{34}
\end{aligned}
$$

Link 2
Writing equations for Link 2

$$
\begin{aligned}
g_2 &= (R_2)^T g_0 \\
f_{12} &= m_2 a_{c2} + R_{23} f_{23} - g_2 m_2 \\
N_{12} &= I_2.\alpha_2 - r_{2c2} \times (R_{23}.f_{23}) + r_{1c2} \times f_{12} + R_{23} N_{23} + \omega_2 \times (I_2 \omega_2)
\end{aligned}
$$

Link 1
Writing equations for Link 1

$$
\begin{aligned}
g_1 &= (R_1)^T g_0 \\
f_{01} &= m_1 a_{c1} + R_{12} f_{12} - g_1 m_1 \\
N_{01} &= I_1 \alpha_1 - r_{1c1} \times (R_{12} f_{12}) + r_{0c1} \times f_1 + R_{12} N_{12} + \omega_1 \times (I_1 \omega_1)
\end{aligned}
$$

Finally, We can obtain the value of the torques required at each joint using the equation 3

$$
\begin{aligned}
\tau_1 &= b_1^T (N_{01}) \\
\tau_2 &= b_2^T (N_{12}) \\
\tau_3 &= b_3^T (N_{23}) \\
\tau_4 &= b_4^T (N_{34}) \\
\tau_5 &= b_5^T (N_{45}) \\
\tau_6 &= b_6^T (N_{56})
\end{aligned}
$$

where, b_i are as defined in the section 7.

8.1 Comments

The results in this section were interesting and verifies why the Newton-Euler formulation often is the preferred choice for manipulators with many degrees of freedom. The recursive algorithm is easy to implement and once we get the frame assignments right, there are small chances of doing any mistakes in the derivation. It is noticed that even though link 3 and 5 have zero length and mass, they still have to be considered in the recursions.

Although utilizing the Newton-Euler formulation appears to be quite easy, the complexity of the resulting model should be emphasized. The basic idea behind recursion is that the solution to a problem depends on solutions to smaller instances on the same problem. The backward recursion of link 1 depends on the backward recursion of link 2, which depends of the backwards recursion of link 3, and so on. All in all the backward recursion of link 1 is directly dependent on all 11 steps back to the forward recursion of link 1. Thus, it should not be a surprise that calculating τ_1 results in a huge vector.

9 Verification of the Dynamic Model

Mathematica was used to solve these equations. The entire code to do this is located in the Appendix A.3. We can solve for the equations in the general sense by specifying $q = [q_1(t) \quad q_2(t) \quad q_3(t) \quad q_4(t) \quad q_5(t) \quad q_6(t)]$ and their derivatives in that order or we can explicitly specify some values for the joint positions and look at the Torque values required at the each of the joints to keep the manipulator from falling against the gravity load. We use the latter approach and compare the resulting Torque values with that of the SimMechanics Model. Let, the joint values be

$$\begin{aligned} q &= [0 \quad -\frac{\pi}{2} \quad \frac{\pi}{2} \quad 0 \quad 0 \quad 0] \\ \dot{q} &= [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0] \\ \ddot{q} &= [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0] \end{aligned}$$

These values correspond to the position of the Manipulator when it is completely stretched. Solving, this using the Mathematica code we get following values of the Torque. This is highlighted on the page 8 of the Appendix A.3.

$$\tau = [0 \quad 243.382 \quad 79.1556 \quad 0 \quad 2.43004 \quad 0]^T \tag{55}$$

The Simscape model is also simulated for the same position by using the 'Motion provided by the Input' option in the Revolute joint block. The resulting Torque is set to be 'computed automatically' and it is sensed using the Sensing option in the Revolute Joint Block. Some plots as observed in the model are shown in figure 7. The exact values of the torques are obtained using the 'toWorkspace' block. The exact values were equal to

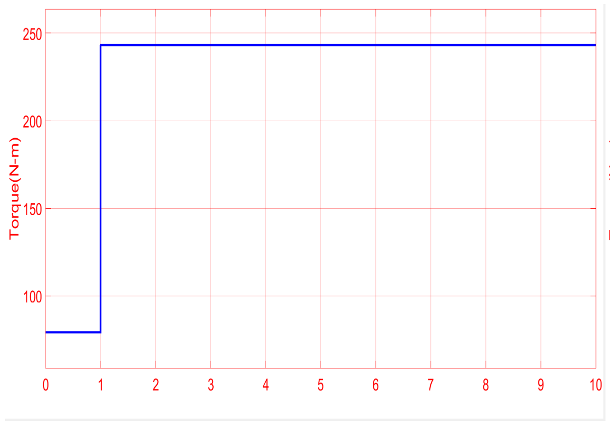
$$\tau = [10^{-14} \quad 243.299 \quad 79.1285 \quad 0 \quad 2.4292 \quad 0]^T \tag{56}$$

Comparing equation 55 and 56 we can see that the values of the torque are in very good agreement. Similar, approach was repeated for some other values of the joint positions and the result was in a good agreement in all the cases. Thus, the validity of the model is verified. For a further cross-verification we can compare the torques experienced in the presence of the payload as performed in the next section.

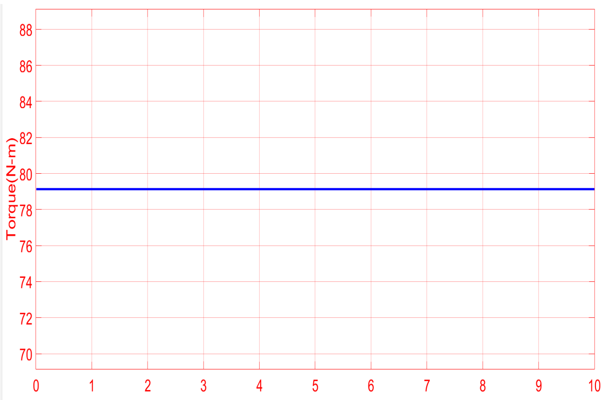
10 Payload Analysis

For performing the Payload Analysis the entire calculation in the Appendix A.3 is repeated with the force on the tip of the last Link in the first step of the backward recursion is modified to $f_{67} = -m_p \times g_6$. Where $m_p = 6kg$ and g_6 is the gravity expressed in the frame of Link 6. The values of the required joint positions are kept same as the last section. This will give us the value of the **max permissible torques** when the Manipulator arm is completely stretched. Thus, the Torques obtained from Mathematica are as follows-

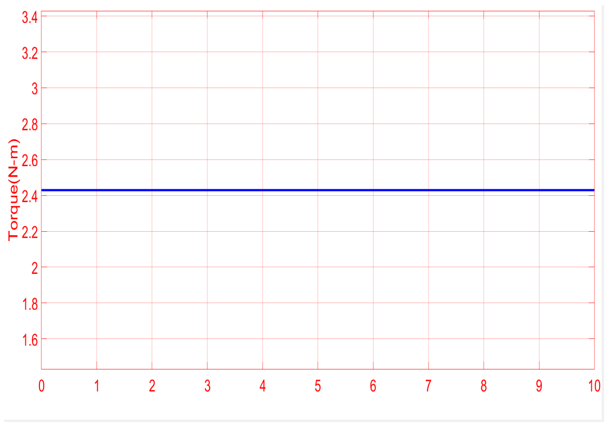
$$\tau = [0 \quad 290.765 \quad 105.348 \quad 0 \quad 6.25594 \quad 0] \tag{57}$$



(a) Torque at Joint 2



(b) Torque at Joint 3



(c) Torque at Joint 5

Figure 7: Torque values as observed at the Revolute Joints 2,3 and 5 of the SimMechanics Model.

To verify this we can augment the simscape model in the Appendix A.2 to add a additional solid sphere which is modelled as a point mass. The 3D model as a result of this addition is as shown in figure 8. The red coloured sphere in the figure is the Payload modelled as the point mass. Thus, the resulting model is simulated again and the torque values obtained using the ’toWorkspace’ block are as follows

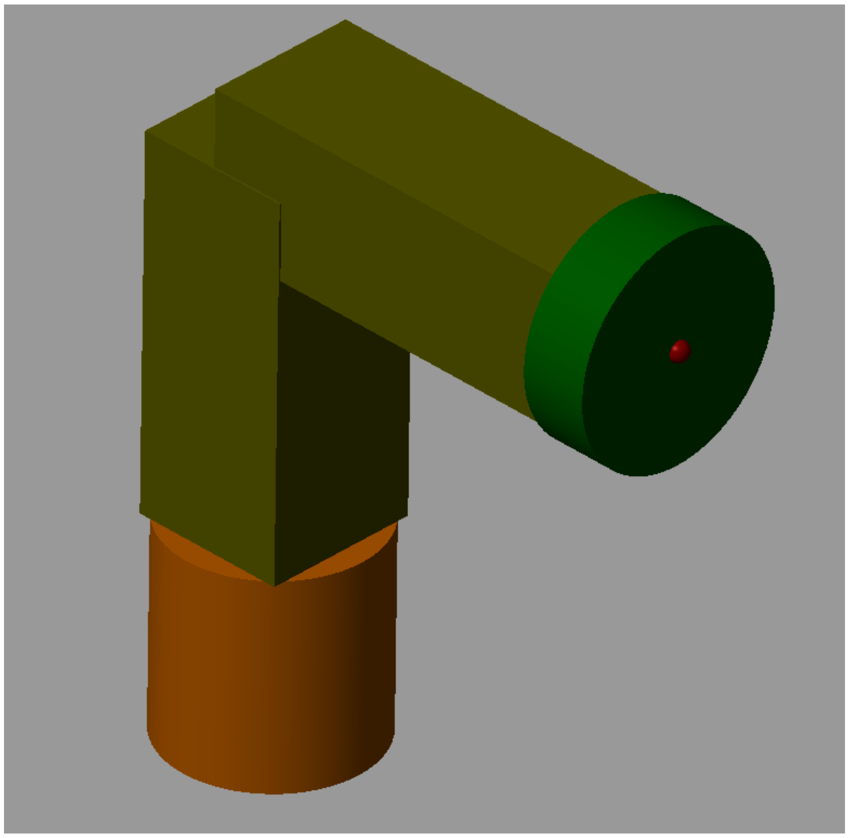


Figure 8: A snap showing the manipulator with a payload in red attached at the end. This payload is modelled as a point mass

$$\tau = [0 \quad 290.665 \quad 105.3123 \quad 0 \quad 6.2538 \quad 0] \tag{58}$$

Thus, these results too are in close agreement.

11 Equations of Motion

The Matrix form of equations of Manipulator is written as shown in [3]

$$D(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q) = \tau \tag{59}$$

It can be shown that, the Mass Matrix $D(q)$, the Coriolis and Centrifugal terms $C(q, \dot{q})$ and the gravitational terms $g(q)$ can be extracted from the existing formulation of the recursive computation in Appendix A.3. To do this, we simply make two changes. First, we modify the the q_{des} vector from a known constant value to a general symbolic expression $q = [q_1(t) \quad q_2(t) \quad q_3(t) \quad q_4(t) \quad q_5(t) \quad q_6(t)]$. Similarly, we make changes to its derivatives and specify it in a generic form. This aspect is commented in the general parameters section of code A.3. Second, we modify the 'takeg' variable in the code from -9.81 to a generic 'g'. Now, we evaluate the notebook again and obtain the value of Torques(τ). This τ is basically a function of q_i, $\dot{q}_i$ and $\ddot{q}_i$. We can construct the Mass Matrix by setting different values of q_i and its derivative equal to zero except one parameter which is set to 1. For example, for calculating the value of first element of the mass matrix m_{11} we simply evaluate the first component of the Torque vector by setting all q_i and its derivative equal to zero except for $\ddot{q_1(t)}$ which we set equal to one. This observation can be verified from the equation 59. Thus, by changing the combinations of the parameters we can obtain the complete Mass Matrix. Similarly, we can evaluate each component of the Torque vector by setting all q_i and its derivatives equal to zero. This will give us the Gravitational terms. Finally, we can get the C Matrix by $C(q, \dot{q}) = \tau - D(q)\ddot{q} - g(q)$.

Complete computation to obtain these matrices can be seen from page 8 onwards of the Appendix A.3. Once, the equations of the motion are known we can start thinking about formulating more sophisticated controllers like the Inverse Dynamics controller.

12 Conclusion and Future Work

Dynamic Model of a 6DOF Manipulator was successfully modelled and validated using the SimMechanics Workbench. Because of the linear nature of calculations involved in the recursive computation, it could very well be the case that the SimMechanics also uses the Luh-Walker-Paul algorithm.

The next logical step after obtaining a dynamic model is to study the open-loop characteristics and think about formulating controllers to achieve a desired performance. The equations of motion developed in the section 11 can very well be used to develop a inverse dynamics control which is a type of control where we give a non-linear input to linearize the dynamics. Other techniques like showed in the 'Dynamics' chapter of [[3]] can be investigated to identify the dynamic parameters which is scalable to the real world manipulator. This requires an access to the actual manipulator. But then, it will make more sense in simulating and formulating controllers. Finally, the framework provided in the Appendix A.3 can be easily reformulated to provide a general automated framework for any given number of freedom manipulator. It can be reframed such that all values can simply be inputs and the relevant sections can be added or subtracted as per the requirement.

A Appendix

A.1 MATLAB script for Manipulator dimensions

```matlab
clc
% Script to get the dimensions with the constraint of overall weight of the
% manipulator = 98KG. Density, the radii, breadth and heights are the
% variables
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Chosen Ones are :
% first link: r = 0.098, h = l1,
% Second link: l = l2, b = h = 0.15
% Fourth Link: l = l4, b = h = 0.146
% Sixth Link: r = 0.108, h = l6
%Density is 3200 kg/m^3
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

l1 = 0.200;
l2 = 0.360;
l4 = 0.380;
l6 =0.065;
fprintf('\n\tr\t\t\ta1\t\t\ta2\t\t\ta3\t\t\trho\t\t\tW1\t\t\tW2\t\t\tW3\t\t\tW4\n');

% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%r is the radius of the cylinder of the first link,
%a1 is the height and breadth of the second link
%a2 is the height and breadth of the fourth link
%a3 is the radius of the sixth link
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for r = 0.08:0.004:0.1
    for a1 = 0.13:0.004:0.15
        for a2 = 0.13:0.004:0.15
            for a3 = 0.1:0.001:0.11
```

```matlab
                rho = 78 / (pi * r^2 * l1 + l2 * a1 * a1 + l4 * a2 * a2 + pi
                    * l6 * a3 * a3);
                W1 = pi * r^2 * l1 * rho;
                W2 = l2 * a1 * a1 * rho;
                W3 = l4 * a2 * a2 * rho;
                W4 = pi * l6 * a3 * a3 * rho;
                fprintf('%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\n',r,a1,a2,a3,rho
                    ,W1,W2,W3,W4);
            end
        end
    end
end
```

A.2 Simscape Model of the Manipulator

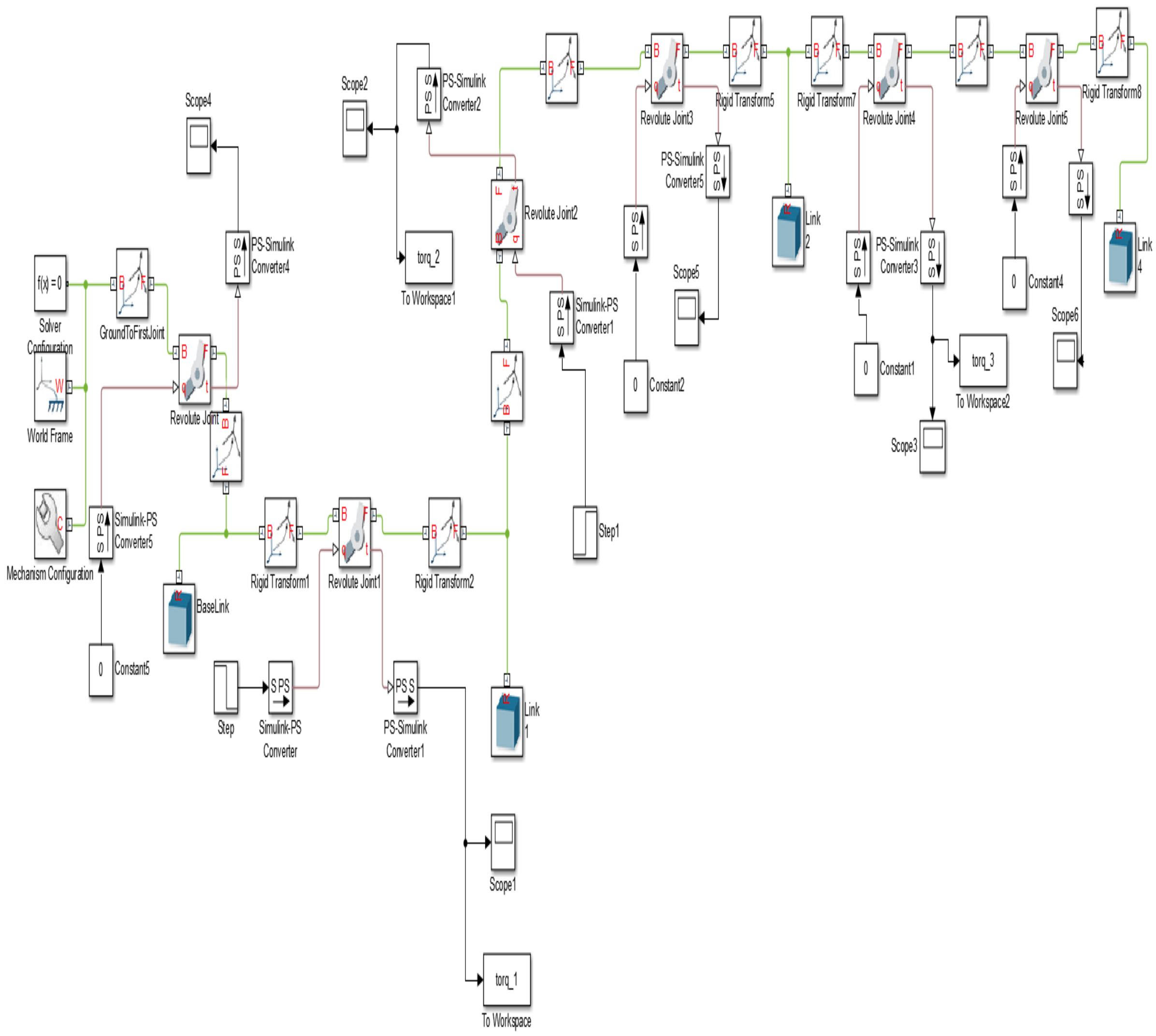


Figure 9: Simscape model of the 6DOF Elbow Manipulator with Spherical wrist

A.3 Mathematica code for the 6DOF Manipulator

Get the Transformation Matrices from DH

```
A = {{Cos[θ], -Sin[θ] * Cos[α], Sin[θ] * Sin[α], a * Cos[θ]}, {Sin[θ], Cos[θ] * Cos[α],
    -Cos[θ] * Sin[α], a * Sin[θ]}, {0, Sin[α], Cos[α], d}, {0, 0, 0, 1}};
A1 = A /. {a → 0, α → Pi/2, d → 0.2, θ → q1[t]};
A2 = A /. {a → 0.36, α → 0, d → 0, θ → (Pi/2 + q2[t])};
A3 = A /. {a → 0, α → Pi/2, d → 0, θ → q3[t]};
A4 = A /. {a → 0, α → -Pi/2, d → 0.38, θ → q4[t]};
A5 = A /. {a → 0, α → Pi/2, d → 0, θ → q5[t]};
A6 = A /. {a → 0, α → 0, d → 0.065, θ → q6[t]};
R12 = A2[[1 ;; 3, 1 ;; 3]];
R23 = A3[[1 ;; 3, 1 ;; 3]];
R34 = A4[[1 ;; 3, 1 ;; 3]];
R45 = A5[[1 ;; 3, 1 ;; 3]];
R56 = A6[[1 ;; 3, 1 ;; 3]];
```

```
T01 = A1;
T02 = Simplify[A1.A2];
T03 = Simplify[T02.A3];
T04 = Simplify[T03.A4];
T05 = Simplify[T04.A5];
T06 = Simplify[T05.A6];
R01 = T01[[1 ;; 3, 1 ;; 3]];
R02 = T02[[1 ;; 3, 1 ;; 3]];
R03 = T03[[1 ;; 3, 1 ;; 3]];
R04 = T04[[1 ;; 3, 1 ;; 3]];
R05 = T05[[1 ;; 3, 1 ;; 3]];
R06 = T06[[1 ;; 3, 1 ;; 3]];
```

```
zvec = {0, 0, 1};
b1 = Transpose[R01].zvec;
b2 = Simplify[Transpose[R02].R01.zvec];
b3 = Simplify[Transpose[R03].R02.zvec];
b4 = Simplify[Transpose[R04].R03.zvec];
b5 = Simplify[Transpose[R05].R04.zvec];
b6 = Simplify[Transpose[R06].R05.zvec];
```

General Parameters

```
rho = 3200;
g0 = {0, 0, -g};
(*qdes = {q1[t],q2[t],q3[t],q4[t],q5[t],q6[t]};
qdotdes = {q1'[t],q2'[t],q3'[t],q4'[t],q5'[t],q6'[t]};
qdotdotdes = {q1''[t],q2''[t],q3''[t],q4''[t],q5''[t],q6''[t]};*)
qdes = {0, -Pi/2, Pi/2, 0, 0, 0};
qdotdes = {0, 0, 0, 0, 0, 0};
qdotdotdes = {0, 0, 0, 0, 0, 0};
```

Link Parameters

Link 1

```
r1 = 0.098;
h1 = 0.2;
r01 = {0, 0.2, 0};
r1c1 = {0, -0.1, 0};
r0c1 = r01 + r1c1;
m1 = rho * Pi * r1^2 * h1;
```

$$I_1 = \begin{pmatrix} \frac{m_1}{12} * (3 * r1^2 + h1^2) & 0 & 0 \\ 0 & \frac{m_1}{2} * r1^2 & 0 \\ 0 & 0 & \frac{m_1}{12} * (3 * r1^2 + h1^2) \end{pmatrix};$$

Link 2

```
l2 = 0.36;
b2 = 0.15;
h2 = 0.15;
r12 = {0.36, 0, 0};
r2c2 = {-0.18, 0, 0};
r1c2 = r12 + r2c2;
m2 = rho * l2 * b2 * h2;
```

$$I_2 = \begin{pmatrix} \frac{m_2}{12} * (b2^2 + h2^2) & 0 & 0 \\ 0 & \frac{m_2}{12} * (l2^2 + h2^2) & 0 \\ 0 & 0 & \frac{m_2}{12} * (b2^2 + l2^2) \end{pmatrix};$$

Link 3

```
r₂₃ = {0, 0, 0};
r3c3 = {0, 0, 0};
r2c3 = r₂₃ + r3c3;
m₃ = 0;
```

$$I_3 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix};$$

Link 4

```
l4 = 0.38;
b4 = 0.146;
h4 = 0.146;
r₃₄ = {0, -0.38, 0};
r4c4 = {0, 0.19, 0};
r3c4 = r₃₄ + r4c4;
m₄ = rho * l4 * b4 * h4;
```

$$I_4 = \begin{pmatrix} \frac{m_4}{12} * \left(b4^2 + l4^2\right) & 0 & 0 \\ 0 & \frac{m_4}{12} * \left(b4^2 + h4^2\right) & 0 \\ 0 & 0 & \frac{m_4}{12} * \left(h4^2 + l4^2\right) \end{pmatrix};$$

Link 5

```
r₄₅ = {0, 0, 0};
r5c5 = {0, 0, 0};
r4c5 = r₄₅ + r5c5;
m₅ = 0;
```

$$I_5 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix};$$

Link 6

```
r6 = 0.108;
h6 = 0.065;
r₅₆ = {0, 0, 0.065};
r6c6 = {0, 0, -0.0325};
r5c6 = r₅₆ + r6c6;
m₆ = rho * Pi * r6² * h6;
```

$$I_6 = \begin{pmatrix} \frac{m_6}{12} * (3 * r6^2 + h6^2) & 0 & 0 \\ 0 & \frac{m_6}{12} * (3 * r6^2 + h6^2) & 0 \\ 0 & 0 & \frac{m_6}{2} * r6^2 \end{pmatrix};$$

Forward Recursion

Link 1

ω_0 = {0, 0, 0};
α_0 = {0, 0, 0};
a_0 = {0, 0, 0};
ω_1 = Transpose[R_{01}].ω_0 + q_1'[t] * b_1;
α_1 = Simplify[Transpose[R_{01}].α_0 + q_1''[t] * b_1 + (Transpose[R_{01}].ω_0) × (q_1'[t] * b_1)];
a_1 = Transpose[R_{01}].a_0 + α_1 × r_{01} + ω_1 × (ω_1 × r_{01});
a_{c1} = a_1 + α_1 × r1c1 + ω_1 × (ω_1 × r1c1);

Link 2

ω_2 = Transpose[R_{12}].ω_1 + q_2'[t] * b_2;
α_2 = Simplify[Transpose[R_{12}].α_1 + q_2''[t] * b_2 + (Transpose[R_{12}].ω_1) × (q_2'[t] * b_2)];
a_2 = Transpose[R_{12}].a_1 + α_2 × r_{12} + ω_2 × (ω_2 × r_{12});
a_{c2} = a_2 + α_2 × r2c2 + ω_2 × (ω_2 × r2c2);

Link 3

ω_3 = Transpose[R_{23}].ω_2 + q_3'[t] * b_3;
α_3 = Simplify[Transpose[R_{23}].α_2 + q_3''[t] * b_3 + (Transpose[R_{23}].ω_2) × (q_3'[t] * b_3)];
a_3 = Simplify[Transpose[R_{23}].a_2 + α_3 × r_{23} + ω_3 × (ω_3 × r_{23})];
a_{c3} = Simplify[a_3 + α_3 × r3c3 + ω_3 × (ω_3 × r3c3)];

Link 4

```
ω₄ = Transpose[R₃₄].ω₃ + q₄'[t] * b₄;
α₄ = Simplify[Transpose[R₃₄].α₃ + q₄''[t] * b₄ + (Transpose[R₃₄].ω₃) × (q₄'[t] * b₄)];
a₄ = Transpose[R₃₄].a₃ + α₄ × r₃₄ + ω₄ × (ω₄ × r₃₄);
ac4 = a₄ + α₄ × r4c4 + ω₄ × (ω₄ × r4c4);
```

Link 5

```
ω₅ = Transpose[R₄₅].ω₄ + q₅'[t] * b₅;
α₅ = Simplify[Transpose[R₄₅].α₄ + q₅''[t] * b₅ + (Transpose[R₄₅].ω₄) × (q₅'[t] * b₅)];
a₅ = Transpose[R₄₅].a₄ + α₅ × r₄₅ + ω₅ × (ω₅ × r₄₅);
ac5 = a₅ + α₅ × r5c5 + ω₅ × (ω₅ × r5c5);
```

Link 6

```
ω₆ = Transpose[R₅₆].ω₅ + q₆'[t] * b₆;
α₆ = Transpose[R₅₆].α₅ + q₆''[t] * b₆ + (Transpose[R₅₆].ω₅) × (q₆'[t] * b₆);
a₆ = Transpose[R₅₆].a₅ + α₆ × r₅₆ + ω₆ × (ω₆ × r₅₆);
ac6 = a₆ + α₆ × r6c6 + ω₆ × (ω₆ × r6c6);
```

Backward Recursion

Link 6

```
g₆ = Transpose[R₀₆].g₀;
f₆₇ = {0, 0, 0};
f₅₆ = f₆₇ - m₆ * g₆ + m₆ * ac6;
τ₅₆ = -r6c6 × f₆₇ + r5c6 × f₅₆ + I₆.α₆ + ω₆ × (I₆.ω₆);
```

Link 5

```
g₅ = Transpose[R₀₅].g₀;
f₄₅ = R₅₆.f₅₆ - m₅ * g₅ + m₅ * ac5;
τ₄₅ = (R₅₆.τ₅₆) - r5c5 × (R₅₆.f₅₆) + r4c5 × f₄₅ + I₅.α₅ + ω₅ × (I₅.ω₅);
```

Link 4

```
g₄ = Transpose[R₀₄].g₀;
f₃₄ = R₄₅.f₄₅ - m₄ * g₄ + m₄ * ac4;
τ₃₄ = R₄₅.τ₄₅ - r4c4 × (R₄₅.f₄₅) + r3c4 × f₃₄ + I₄.α₄ + ω₄ × (I₄.ω₄);
```

Link 3

g_3 = Transpose[R_{03}].g_0;
f_{23} = R_{34}.f_{34} − m_3 * g_3 + m_3 * a_{c3};
τ_{23} = R_{34}.τ_{34} − r3c3 × (R_{34}.f_{34}) + r2c3 × f_{23} + I_3.α_3 + ω_3 × (I_3.ω_3);

Link 2

g_2 = Transpose[R_{02}].g_0;
f_{12} = R_{23}.f_{23} − m_2 * g_2 + m_2 * a_{c2};
τ_{12} = R_{23}.τ_{23} − r2c2 × (R_{23}.f_{23}) + r1c2 × f_{12} + I_2.α_2 + ω_2 × (I_2.ω_2);

Link 1

g_1 = Transpose[R_{01}].g_0;
f_{01} = R_{12}.f_{12} − m_1 * g_1 + m_1 * a_{c1};
τ_{01} = R_{12}.τ_{12} − r1c1 × (R_{12}.f_{12}) + r0c1 × f_{01} + I_1.α_1 + ω_1 × (I_1.ω_1);

Computation

```
takeg = 9.81;
t1 = τ01 /. {q1[t] → qdes[[1]], q2[t] → qdes[[2]], q3[t] → qdes[[3]],
     q4[t] → qdes[[4]], q5[t] → qdes[[5]], q6[t] → qdes[[6]], q1'[t] → qdotdes[[1]],
     q2'[t] → qdotdes[[2]], q3'[t] → qdotdes[[3]], q4'[t] → qdotdes[[4]],
     q5'[t] → qdotdes[[5]], q6'[t] → qdotdes[[6]], q1''[t] → qdotdotdes[[1]],
     q2''[t] → qdotdotdes[[2]], q3''[t] → qdotdotdes[[3]], q4''[t] → qdotdotdes[[4]],
     q5''[t] → qdotdotdes[[5]], q6''[t] → qdotdotdes[[6]], g → takeg};
t2 = τ12 /. {q1[t] → qdes[[1]], q2[t] → qdes[[2]], q3[t] → qdes[[3]],
     q4[t] → qdes[[4]], q5[t] → qdes[[5]], q6[t] → qdes[[6]], q1'[t] → qdotdes[[1]],
     q2'[t] → qdotdes[[2]], q3'[t] → qdotdes[[3]], q4'[t] → qdotdes[[4]],
     q5'[t] → qdotdes[[5]], q6'[t] → qdotdes[[6]], q1''[t] → qdotdotdes[[1]],
     q2''[t] → qdotdotdes[[2]], q3''[t] → qdotdotdes[[3]], q4''[t] → qdotdotdes[[4]],
     q5''[t] → qdotdotdes[[5]], q6''[t] → qdotdotdes[[6]], g → takeg};
t3 = τ23 /. {q1[t] → qdes[[1]], q2[t] → qdes[[2]], q3[t] → qdes[[3]],
     q4[t] → qdes[[4]], q5[t] → qdes[[5]], q6[t] → qdes[[6]], q1'[t] → qdotdes[[1]],
     q2'[t] → qdotdes[[2]], q3'[t] → qdotdes[[3]], q4'[t] → qdotdes[[4]],
     q5'[t] → qdotdes[[5]], q6'[t] → qdotdes[[6]], q1''[t] → qdotdotdes[[1]],
     q2''[t] → qdotdotdes[[2]], q3''[t] → qdotdotdes[[3]], q4''[t] → qdotdotdes[[4]],
     q5''[t] → qdotdotdes[[5]], q6''[t] → qdotdotdes[[6]], g → takeg};
t4 = τ34 /. {q1[t] → qdes[[1]], q2[t] → qdes[[2]], q3[t] → qdes[[3]],
     q4[t] → qdes[[4]], q5[t] → qdes[[5]], q6[t] → qdes[[6]], q1'[t] → qdotdes[[1]],
     q2'[t] → qdotdes[[2]], q3'[t] → qdotdes[[3]], q4'[t] → qdotdes[[4]],
     q5'[t] → qdotdes[[5]], q6'[t] → qdotdes[[6]], q1''[t] → qdotdotdes[[1]],
     q2''[t] → qdotdotdes[[2]], q3''[t] → qdotdotdes[[3]], q4''[t] → qdotdotdes[[4]],
     q5''[t] → qdotdotdes[[5]], q6''[t] → qdotdotdes[[6]], g → takeg};
t5 = τ45 /. {q1[t] → qdes[[1]], q2[t] → qdes[[2]], q3[t] → qdes[[3]],
     q4[t] → qdes[[4]], q5[t] → qdes[[5]], q6[t] → qdes[[6]], q1'[t] → qdotdes[[1]],
     q2'[t] → qdotdes[[2]], q3'[t] → qdotdes[[3]], q4'[t] → qdotdes[[4]],
     q5'[t] → qdotdes[[5]], q6'[t] → qdotdes[[6]], q1''[t] → qdotdotdes[[1]],
     q2''[t] → qdotdotdes[[2]], q3''[t] → qdotdotdes[[3]], q4''[t] → qdotdotdes[[4]],
     q5''[t] → qdotdotdes[[5]], q6''[t] → qdotdotdes[[6]], g → takeg};
t6 = τ56 /. {q1[t] → qdes[[1]], q2[t] → qdes[[2]], q3[t] → qdes[[3]],
     q4[t] → qdes[[4]], q5[t] → qdes[[5]], q6[t] → qdes[[6]], q1'[t] → qdotdes[[1]],
     q2'[t] → qdotdes[[2]], q3'[t] → qdotdes[[3]], q4'[t] → qdotdes[[4]],
     q5'[t] → qdotdes[[5]], q6'[t] → qdotdes[[6]], q1''[t] → qdotdotdes[[1]],
     q2''[t] → qdotdotdes[[2]], q3''[t] → qdotdotdes[[3]], q4''[t] → qdotdotdes[[4]],
     q5''[t] → qdotdotdes[[5]], q6''[t] → qdotdotdes[[6]], g → takeg};
```

Actual Torques Required

```
τz1 = TrigReduce[b1.t1];
τz2 = TrigReduce[b2.t2];
τz3 = TrigReduce[b3.t3];
τz4 = TrigReduce[b4.t4];
τz5 = TrigReduce[b5.t5];
τz6 = TrigReduce[b6.t6];
Torq = {τz1, τz2, τz3, τz4, τz5, τz6}
```

```
{0, 243.3824141750203, 79.15555672137201, 0, 2.430038123143249, 0}
```

Following sections should only be used to extract the Manipulator equations

Configuration Dependent Mass Matrix

```
m₁₁ = τz1 /.
   {q₁'[t] → 0, q₂'[t] → 0, q₃'[t] → 0, q₄'[t] → 0, q₅'[t] → 0, q₆'[t] → 0, q₁''[t] → 1,
    q₂''[t] → 0, q₃''[t] → 0, q₄''[t] → 0, q₅''[t] → 0, q₆''[t] → 0, g → 0};
m₂₁ = τz2 /. {q₁'[t] → 0, q₂'[t] → 0, q₃'[t] → 0, q₄'[t] → 0,
    q₅'[t] → 0, q₆'[t] → 0, q₁''[t] → 1, q₂''[t] → 0,
    q₃''[t] → 0, q₄''[t] → 0, q₅''[t] → 0, q₆''[t] → 0, g → 0};
m₃₁ = τz3 /. {q₁'[t] → 0, q₂'[t] → 0, q₃'[t] → 0, q₄'[t] → 0,
    q₅'[t] → 0, q₆'[t] → 0, q₁''[t] → 1, q₂''[t] → 0,
    q₃''[t] → 0, q₄''[t] → 0, q₅''[t] → 0, q₆''[t] → 0, g → 0};
m₄₁ = τz4 /. {q₁'[t] → 0, q₂'[t] → 0, q₃'[t] → 0, q₄'[t] → 0,
    q₅'[t] → 0, q₆'[t] → 0, q₁''[t] → 1, q₂''[t] → 0,
    q₃''[t] → 0, q₄''[t] → 0, q₅''[t] → 0, q₆''[t] → 0, g → 0};
m₅₁ = τz5 /. {q₁'[t] → 0, q₂'[t] → 0, q₃'[t] → 0, q₄'[t] → 0,
    q₅'[t] → 0, q₆'[t] → 0, q₁''[t] → 1, q₂''[t] → 0,
    q₃''[t] → 0, q₄''[t] → 0, q₅''[t] → 0, q₆''[t] → 0, g → 0};
m₆₁ = τz6 /. {q₁'[t] → 0, q₂'[t] → 0, q₃'[t] → 0, q₄'[t] → 0,
    q₅'[t] → 0, q₆'[t] → 0, q₁''[t] → 1, q₂''[t] → 0,
    q₃''[t] → 0, q₄''[t] → 0, q₅''[t] → 0, q₆''[t] → 0, g → 0};
m₁₂ = τz1 /. {q₁'[t] → 0, q₂'[t] → 0, q₃'[t] → 0, q₄'[t] → 0,
    q₅'[t] → 0, q₆'[t] → 0, q₁''[t] → 0, q₂''[t] → 1,
    q₃''[t] → 0, q₄''[t] → 0, q₅''[t] → 0, q₆''[t] → 0, g → 0};
m₂₂ = τz2 /. {q₁'[t] → 0, q₂'[t] → 0, q₃'[t] → 0, q₄'[t] → 0,
    q₅'[t] → 0, q₆'[t] → 0, q₁''[t] → 0, q₂''[t] → 1,
    q₃''[t] → 0, q₄''[t] → 0, q₅''[t] → 0, q₆''[t] → 0, g → 0};
m₃₂ = τz3 /. {q₁'[t] → 0, q₂'[t] → 0, q₃'[t] → 0, q₄'[t] → 0,
    q₅'[t] → 0, q₆'[t] → 0, q₁''[t] → 0, q₂''[t] → 1,
    q₃''[t] → 0, q₄''[t] → 0, q₅''[t] → 0, q₆''[t] → 0, g → 0};
m₄₂ = τz4 /. {q₁'[t] → 0, q₂'[t] → 0, q₃'[t] → 0, q₄'[t] → 0,
    q₅'[t] → 0, q₆'[t] → 0, q₁''[t] → 0, q₂''[t] → 1,
    q₃''[t] → 0, q₄''[t] → 0, q₅''[t] → 0, q₆''[t] → 0, g → 0};
m₅₂ = τz5 /. {q₁'[t] → 0, q₂'[t] → 0, q₃'[t] → 0, q₄'[t] → 0,
    q₅'[t] → 0, q₆'[t] → 0, q₁''[t] → 0, q₂''[t] → 1,
    q₃''[t] → 0, q₄''[t] → 0, q₅''[t] → 0, q₆''[t] → 0, g → 0};
m₆₂ = τz6 /. {q₁'[t] → 0, q₂'[t] → 0, q₃'[t] → 0, q₄'[t] → 0,
    q₅'[t] → 0, q₆'[t] → 0, q₁''[t] → 0, q₂''[t] → 1,
    q₃''[t] → 0, q₄''[t] → 0, q₅''[t] → 0, q₆''[t] → 0, g → 0};
m₁₃ = τz1 /. {q₁'[t] → 0, q₂'[t] → 0, q₃'[t] → 0, q₄'[t] → 0,
    q₅'[t] → 0, q₆'[t] → 0, q₁''[t] → 0, q₂''[t] → 0,
    q₃''[t] → 1, q₄''[t] → 0, q₅''[t] → 0, q₆''[t] → 0, g → 0};
m₂₃ = τz2 /. {q₁'[t] → 0, q₂'[t] → 0, q₃'[t] → 0, q₄'[t] → 0,
    q₅'[t] → 0, q₆'[t] → 0, q₁''[t] → 0, q₂''[t] → 0,
    q₃''[t] → 1, q₄''[t] → 0, q₅''[t] → 0, q₆''[t] → 0, g → 0};
m₃₃ = τz3 /. {q₁'[t] → 0, q₂'[t] → 0, q₃'[t] → 0, q₄'[t] → 0,
    q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
    q3''[t] → 1, q4''[t] → 0, q5''[t] → 0, q6''[t] → 0, g → 0};
m43 = τz4 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
    q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
    q3''[t] → 1, q4''[t] → 0, q5''[t] → 0, q6''[t] → 0, g → 0};
m53 = τz5 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
    q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
    q3''[t] → 1, q4''[t] → 0, q5''[t] → 0, q6''[t] → 0, g → 0};
m63 = τz6 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
    q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
    q3''[t] → 1, q4''[t] → 0, q5''[t] → 0, q6''[t] → 0, g → 0};
m14 = τz1 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
    q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
    q3''[t] → 0, q4''[t] → 1, q5''[t] → 0, q6''[t] → 0, g → 0};
m24 = τz2 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
    q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
    q3''[t] → 0, q4''[t] → 1, q5''[t] → 0, q6''[t] → 0, g → 0};
m34 = τz3 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
    q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
    q3''[t] → 0, q4''[t] → 1, q5''[t] → 0, q6''[t] → 0, g → 0};
m44 = τz4 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
    q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
    q3''[t] → 0, q4''[t] → 1, q5''[t] → 0, q6''[t] → 0, g → 0};
m54 = τz5 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
    q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
    q3''[t] → 0, q4''[t] → 1, q5''[t] → 0, q6''[t] → 0, g → 0};
m64 = τz6 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
    q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
    q3''[t] → 0, q4''[t] → 1, q5''[t] → 0, q6''[t] → 0, g → 0};
m15 = τz1 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
    q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
    q3''[t] → 0, q4''[t] → 0, q5''[t] → 1, q6''[t] → 0, g → 0};
m25 = τz2 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
    q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
    q3''[t] → 0, q4''[t] → 0, q5''[t] → 1, q6''[t] → 0, g → 0};
m35 = τz3 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
    q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
    q3''[t] → 0, q4''[t] → 0, q5''[t] → 1, q6''[t] → 0, g → 0};
m45 = τz4 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
    q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
    q3''[t] → 0, q4''[t] → 0, q5''[t] → 1, q6''[t] → 0, g → 0};
m55 = τz5 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
    q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
    q3''[t] → 0, q4''[t] → 0, q5''[t] → 1, q6''[t] → 0, g → 0};
m65 = τz6 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
    q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
    q3''[t] → 0, q4''[t] → 0, q5''[t] → 1, q6''[t] → 0, g → 0};
m16 = τz1 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
    q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
    q3''[t] → 0, q4''[t] → 0, q5''[t] → 0, q6''[t] → 1, g → 0};
m26 = τz2 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
     q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
     q3''[t] → 0, q4''[t] → 0, q5''[t] → 0, q6''[t] → 1, g → 0};
m36 = τz3 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
     q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
     q3''[t] → 0, q4''[t] → 0, q5''[t] → 0, q6''[t] → 1, g → 0};
m46 = τz4 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
     q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
     q3''[t] → 0, q4''[t] → 0, q5''[t] → 0, q6''[t] → 1, g → 0};
m56 = τz5 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
     q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
     q3''[t] → 0, q4''[t] → 0, q5''[t] → 0, q6''[t] → 1, g → 0};
m66 = τz6 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0,
     q5'[t] → 0, q6'[t] → 0, q1''[t] → 0, q2''[t] → 0,
     q3''[t] → 0, q4''[t] → 0, q5''[t] → 0, q6''[t] → 1, g → 0};
```

$$M = \begin{pmatrix} m_{11} & m_{12} & m_{13} & m_{14} & m_{15} & m_{16} \\ m_{21} & m_{22} & m_{23} & m_{24} & m_{25} & m_{26} \\ m_{31} & m_{32} & m_{33} & m_{34} & m_{35} & m_{36} \\ m_{41} & m_{42} & m_{43} & m_{44} & m_{45} & m_{46} \\ m_{51} & m_{52} & m_{53} & m_{54} & m_{55} & m_{56} \\ m_{61} & m_{62} & m_{63} & m_{64} & m_{65} & m_{66} \end{pmatrix}$$

Gravitational Terms

```
g6 = τz6 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0, q5'[t] → 0, q6'[t] → 0,
     q1''[t] → 0, q2''[t] → 0, q3''[t] → 0, q4''[t] → 0, q5''[t] → 0, q6''[t] → 0};
g5 = τz5 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0, q5'[t] → 0, q6'[t] → 0,
     q1''[t] → 0, q2''[t] → 0, q3''[t] → 0, q4''[t] → 0, q5''[t] → 0, q6''[t] → 0};
g4 = τz4 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0, q5'[t] → 0, q6'[t] → 0,
     q1''[t] → 0, q2''[t] → 0, q3''[t] → 0, q4''[t] → 0, q5''[t] → 0, q6''[t] → 0};
g3 = τz3 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0, q5'[t] → 0, q6'[t] → 0,
     q1''[t] → 0, q2''[t] → 0, q3''[t] → 0, q4''[t] → 0, q5''[t] → 0, q6''[t] → 0};
g2 = τz2 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0, q5'[t] → 0, q6'[t] → 0,
     q1''[t] → 0, q2''[t] → 0, q3''[t] → 0, q4''[t] → 0, q5''[t] → 0, q6''[t] → 0};
g1 = τz1 /. {q1'[t] → 0, q2'[t] → 0, q3'[t] → 0, q4'[t] → 0, q5'[t] → 0, q6'[t] → 0,
     q1''[t] → 0, q2''[t] → 0, q3''[t] → 0, q4''[t] → 0, q5''[t] → 0, q6''[t] → 0};
G = {g1, g2, g3, g4, g5, g6}
```

Coriolis Terms

```
CDq = Torq - M.qdotdotdes - G
```

References

[1] *H. Asada and J.—J.E. Slotine. Robot Analysis and Control. John Wiley and Sons, USA, 1986*

[2] *ABB Robotics. ABB Product Guide for IRB 140.* `http://new.abb.com/products/robotics/industrial-robots/irb-140`

[3] *Spong, Mark W., Seth. Hutchinson, and M. Vidyasagar. Robot Modeling and Control.Hoboken, NJ:JohnWiley,2006. Print*