Analysis of Space Robotic Manipulator and Chaser-Target Modelling for Spacecraft Rendezvous &Docking

June 16, 2021

Submitted to: Dr. Dipak Kumar Giri

Submitted by: Priyank Dubey

Overview

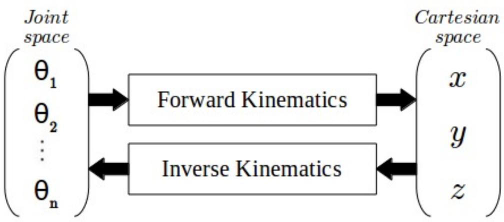
Robotic Manipulators are robots composed of an assembly of links connected by joints (a Kinematic Chain in a particular fashion), and the most advanced types of robots available.Robotic manipulators have become ubiquitous in space industry, they perform varieties of tasks such as assisting in the deployment of satellites, docking and berthing of spacecrafts to the (International Space Station) ISS, On-Orbit Servicing ,Assisting in the mobility of astronauts during Extra Vehicular Activity (EVA).

All of these jobs require the same core capability and complete control of the robotic arm's end-effector to reach specific 3D coordinates within its workspace so that it can interact with the environment at that particular location and perform specified task.

Kinematics Analysis

There are two types of Kinematics - Forward Kinematics and Inverse Kinematics. Forward Kinematics is the process of computing a manipulator's end-effector(EE) position in Cartesian coordinates from its given joint angles. This can be achieved by a composition of homogeneous transformations that map the base frame onto the end-effector's frame, taking as input the joint angles. The end-effector's coordinates can then be extracted from the resulting composite transform matrix while Inverse Kinematics is the reverse process where the EE position is known and a set of joint angles that would result in that position need to be determined. This is a more complicated process than Forward Kinematics as multiple solutions can exist for the same EE position. However, no joint angle solutions exist for any EE position outside the manipulator's workspace. There are two main approaches to solve the IK problem: numerical and analytical. The later approach is used in this project.

Consideration :All the six joints are considered to be Revolute with marginal zero error and EE(End effector) is thought to be gripper with plain geometry and the EE reference frame O_{EE} and the six reference frame of the each attached links are O_1,O_2,$O_3 O_4$,O_5,O_6 respectively.



Denavit–Hartenberg (DH) Method: It is the method used to solve the serial manipulators problems, It has four parameters describing the rotations and translations between adjacent links called D-H parameters. The definition of these parameters constitutes a convention for assigning coordinate reference frames to the links of a robotic manipulator. We will be using modified convention of DH parameters as defined by [Craig, JJ. (2005)].

The parameters are defined as follows:

- α_{i-1}: twist angle between the z-axes of links i-1 and i (measured about xi-1 in a right-hand sense)
- a_{i-1}: link distance between the z-axes of links i-1 and i (measured xi-1)
- d_i: link offset signed distance between the x-axes of links i-1 and i (measured along zi)
- θ_i: joint angle between the x-axes of links i-1 and i (measured about zi in a right-hand sense)

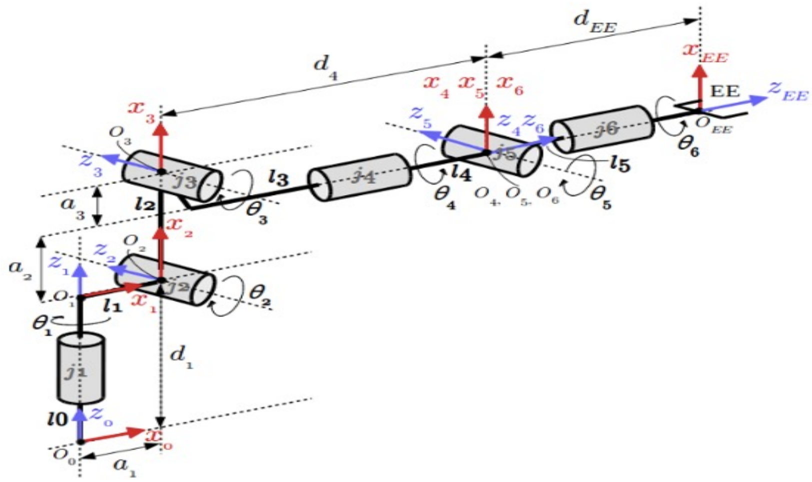
The parameter assignment process for open kinematic chains with n degrees of freedom (i.e., joints) is summarized as:

1. Label all joints from {1, 2, ... , n}.
2. Label all links from {0, 1, ..., n} starting with the fixed base link as 0.
3. Draw lines through all joints, defining the joint axes.
4. Assign the Z-axis of each frame to point along its joint axis.
5. Identify the common normal between each frame Z_{i-1} and Z_i
6. The endpoints of intermediate links (i.e., not the base link or the end effector) are associated with two joint axes, {i} and {i+1}.For i from 1 to n-1, assign the $X_i$to be ...
7. For skew axes, along the normal between Z_i and Z_{i+1} and pointing from {i} to {i+1}.
8. For intersecting axes, normal to the plane containing Z_i and Z_{i+1}. For parallel or coincident axes, the assignment is arbitrary; look for ways to make other DH parameters equal to zero.
9. For the base link, always choose frame {0} to be coincident with frame {1} when the first joint variable (ϑ_1 or $d_{1)i}$s equal to zero. This will guarantee that $\alpha_0 = a_0 = 0$, and, if joint 1 is a revolute, $d_1 = 0$. If joint 1 is prismatic, then $\vartheta_1 = 0$.
10. For the end effector frame, if joint n is revolute, choose X_n to be in the direction of X_{n-1} when $\vartheta_n = 0$ and the origin of frame {n} such that $d_n = 0$.

Special cases involving the Z_{i-1} and $Z_i$axes:

- collinear lines: alpha = 0 and a = 0
- parallel lines: alpha = 0 and a $\neq$ 0
- intersecting lines: alpha $\neq$ 0 and a = 0
- If the common normal intersects Z_i at the origin of frame i, then d_i is zero.

Assignment of frames to each joint according to DENAVIT HATENBERG method and indicated D-H parameters. In the zero configuration, all joint angles are assumed to be zero.



Once the frame assignments are made, the DH parameters are typically presented in tabular form (below). Each row in the table corresponds to the homogeneous transform from frame {i} to frame {i+1}.

D-H table prepared from the above figure:

Joints()	θ_i	α_{i-1}	a_{i-1} (along X axis)	d_i(along Z axis)
1	θ_1	0	0	d_1
2	$\theta_2 - 90$	-90	a_2	0
3	θ_3	0	a_3	0
4	θ_4	-90	a_4	d_4
5	θ_5	-90	0	0
6	θ_6	-90	0	0
7	0	0	0	d_7

Where $R, D, \theta_i, \alpha_{i-1}$ are the defined as modified DH parameters and ruled by DH convention.

In the case where a reference frame is both simultaneously rotated and translated (transformed) with respect to some other reference frame, a homogeneous transform matrix describes the transformation.Transformation Matrix can be represented by (H_i^{i-1})

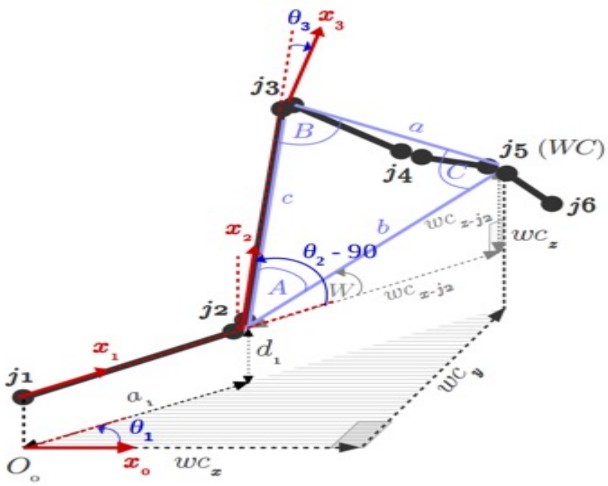
$$\begin{bmatrix} cos\theta_i & -sin\theta_i & 0 & a_{i-1} \\ cos\alpha_{i-1}sin\theta_i & cos\alpha_{i-1}cos\theta_i & -sin\alpha_{i-1} & -sin\alpha_{i-1}d_i \\ cos\alpha_{i-1}sin\theta_i & cos\alpha_{i-1}cos\theta_i & cos\alpha_{i-1} & cos\alpha_{i-1}d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

and in simpler form, it can be expressed as -

$$\begin{bmatrix} & R_i^{i-1} & d_i^{i-1} & \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Where R_i^{i-1} and ${d_i}^{i-1}$ represents the Rotation Matrix, which implies a means of expressing a vector in one coordinate frame in terms of some other coordinate frame. and Translation vector respectively.

Calculations for Inverse Kinematics



From Top view,we can see that ϑ_1 is the joint-1 angle between x_0 and x_1 measured about z_1.It is calculated using the x and y coordinates of Wrist center (WC) relative to the base frame we can easily compute-

$$\theta_1 = \arctan \frac{y_{0,1}}{x_{0,1}}$$

Where $y_{0,1}$ & $x_{0,1}$ are X and Y co-ordinate positions of End Effector with respect to O frame (Ground Frame). Also,

$$\theta_2 - 90 = -(A + \psi)$$

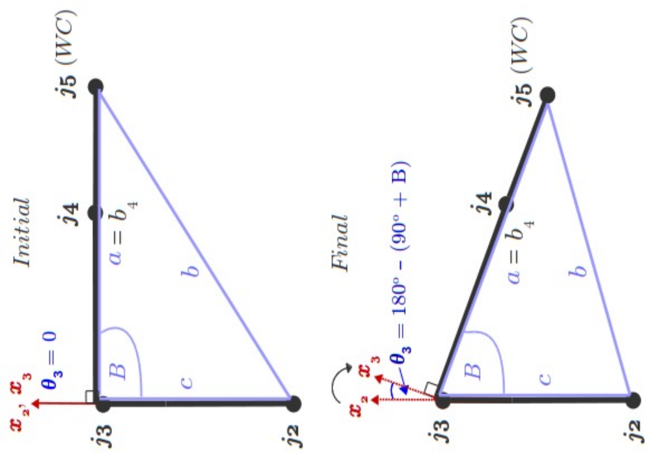
$$\theta_2 = 90 + (A + \psi)$$

Where

$$\psi = \arctan \frac{z_{j,2}}{x_{j,2}}$$

$z_{j,2}$ and $x_{j,2}$ are positions of end effector with respect to Joint 2

$$\theta_2 = 90 + (A + \arctan \frac{z_{j,2}}{x_{j,2}})$$



$$\theta_3 = 180 - (90 + B)$$

$$\theta_3 = (90 - B)$$

B can be calculated by using law of cosines (LOC) $B = \arccos \frac{a^2+c^2-b^2}{2ac}$ Hence,

$$\theta_3 = (90 - \arccos \frac{a^2 + c^2 - b^2}{2ac})$$

Rotation Matrix with respect to joints can be taken from standard D-H matrix and can be written as :

$$R_i^{i-1} = \begin{bmatrix} cos\theta_i & -sin\theta_i & 0 \\ cos\alpha_{i-1}sin\theta_i & cos\alpha_{i-1}cos\theta_i & -sin\alpha_{i-1} \\ cos\alpha_{i-1}sin\theta_i & cos\alpha_{i-1}cos\theta_i & cos\alpha_{i-1} \end{bmatrix}$$

$R_1^0, R_2^1, R_3^2, R_4^3, R_5^4, R_6^5$ can be written as -

$$R_1^0 = \begin{bmatrix} cos\theta_1 & -sin\theta_1 & 0 \\ sin\theta_1 & cos\theta_1 & 0 \\ sin\theta_1 & cos\theta_1 & 1 \end{bmatrix} \qquad R_2^1 = \begin{bmatrix} sin\theta_2 & -cos\theta_2 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

$$R_3^2 = \begin{bmatrix} cos\theta_3 & -sin\theta_3 & 0 \\ sin\theta_3 & cos\theta_3 & 0 \\ sin\theta_3 & cos\theta_3 & 1 \end{bmatrix} \qquad R_4^3 = \begin{bmatrix} sin\theta_4 & -cos\theta_4 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

$$R_5^4 = \begin{bmatrix} cos\theta_4 & -sin\theta_4 & 0 \\ 0 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix} \qquad R_6^5 = \begin{bmatrix} cos\theta_6 & sin\theta_6 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

We know that By Transformation Rule

$$R_6^0 = R_1^0.R_2^1.R_3^2.R_4^3.R_5^4.R_6^5$$

$$R_6^0 = R_3^0.R_6^3$$

$$\left(R_3^0\right)^{-1} R_6^0 = R_6^3$$

Where $R_3^0 = R_1^0.R_2^1.R_3^2$

$$\begin{bmatrix} cos\theta_1 & -sin\theta_1 & 0 \\ sin\theta_1 & cos\theta_1 & 0 \\ sin\theta_1 & cos\theta_1 & 1 \end{bmatrix} * \begin{bmatrix} sin\theta_2 & -cos\theta_2 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} cos\theta_3 & -sin\theta_3 & 0 \\ sin\theta_3 & cos\theta_3 & 0 \\ sin\theta_3 & cos\theta_3 & 1 \end{bmatrix}$$

$$\begin{bmatrix} c\theta_1 s\theta_2 c\theta_3 - c\theta_1 c\theta_2 s\theta_3 - s\theta_1 s\theta_3 & -c\theta_1 s\theta_2 s\theta_3 - c\theta_1 c\theta_2 c\theta_3 - s\theta_1 s\theta_3 & -s\theta_1 \\ s\theta_1 s\theta_2 c\theta_3 - s\theta_1 c\theta_2 s\theta_3 + c\theta_1 s\theta_3 & -s\theta_1 s\theta_2 s\theta_3 - s\theta_1 c\theta_2 c\theta_3 + c\theta_1 c\theta_3 & c\theta_1 \\ s\theta_1 s\theta_2 c\theta_3 - s\theta_1 c\theta_2 s\theta_3 + c\theta_1 s\theta_3 & -s\theta_1 s\theta_2 s\theta_3 - s\theta_1 c\theta_2 c\theta_3 + c\theta_1 c\theta_3 & c\theta_1 \end{bmatrix}$$

Similarly R_6^0 can be calculated and Symbolically R_6^3 can be written as -

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

By actual calculation , we will get R_6^3 as

$$\begin{bmatrix} -s\theta_4 s\theta_6 + c\theta_4 c\theta_5 c\theta_6 & -s\theta_4 c\theta_6 - c\theta_4 c\theta_5 s\theta_6 & -c\theta_4 s\theta_5 \\ c\theta_6 s\theta_5 & -s\theta_6 s\theta_5 & c\theta_5 \\ -s\theta_4 c\theta_6 c\theta_5 + c\theta_4 s\theta_6 & s\theta_4 s\theta_6 c\theta_5 + c\theta_4 c\theta_6 & s\theta_4 s\theta_5 \end{bmatrix}$$

Angles $\theta_4, \theta_5, \theta_6$ can be calculated by comparing the sysmbolic and actual representation-

$$\frac{-r_{33}}{r_{13}} = \frac{s\theta_4 s\theta_5}{c\theta_4 s\theta_5}$$

$$= tan\theta_4$$

$$\theta_4 = \arctan\frac{-r_{33}}{r_{13}}$$

$$\frac{\sqrt{(r_{13})^2 + (r_{33})^2}}{r_{23}} = tan\theta_5$$

$$\theta_5 = \arctan\frac{\sqrt{(r_{13})^2 + (r_{33})^2}}{r_{23}}$$

$$\frac{-r_{22}}{r_{21}} = \frac{-s\theta_6 s\theta_5}{c\theta_6 s\theta_5}$$

$$= tan\theta_6$$

$$\theta_6 = \arctan\frac{-r_{22}}{r_{21}}$$

Colllectively , we can write

$\theta_1 = \arctan\frac{y_{0,1}}{x_{0,1}}$

$\theta_2 = 90 + (A + \arctan\frac{z_{j,2}}{x_{j,2}})$

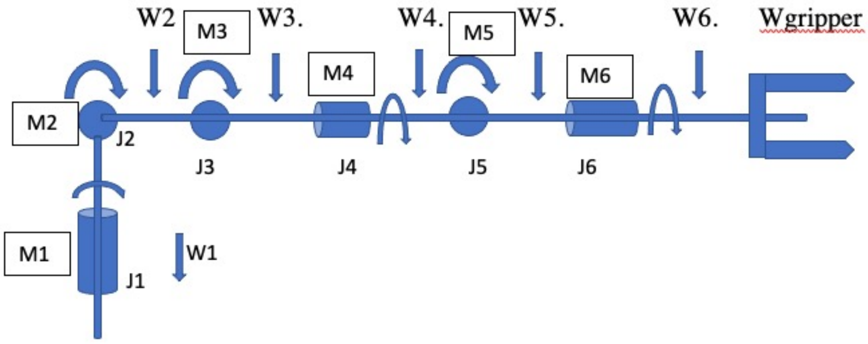
$\theta_3 = (90 - \arccos\frac{a^2+c^2-b^2}{2ac})$

$\theta_4 = \arctan\frac{-r_{33}}{r_{13}}$

$\theta_5 = \arctan\frac{\sqrt{(r_{13})^2+(r_{33})^2}}{r_{23}}$

$\theta_6 = \arctan\frac{-r_{22}}{r_{21}}$

Dynamics Analysis



Free Body Diagram of Robotic manipulator in stretched out pose

Dynamics analysis is done in worst case possibility i.e. the torque on particular motors will be maximum.

Where J1,J2,J3,J4,J5,J6 represents the respective joints of robotic manipulator.

W_1 = Weight of link L1
W_2 = Weight of link L2
W_3= Weight of link L3
W_4= Weight of link L4
W_5 = Weight of link L5
W_6 = Weight of link L6
Wgripper = Weight of gripper
Wpayload = Weight of payload

L_1 = Length of link L1
L_2 = Length of link L2
L_3 = Length of link L3
L_4 = Length of link L4
L_5 = Length of link L5

W_{j1} = Weight of Joint1
W_{j2} = Weight of Joint2
W_{j3} = Weight of Joint3
W_{j4} = Weight of Joint4

W_{j5} = Weight of Joint5
W_{j6} = Weight of Joint6

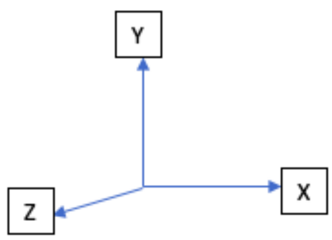
Each joint will experience two types of Torque - 1. Due to Gravitational force 2. Due to Inertial effect

Let resistive torque at joint 1 due to gravity = T_{1g}
Let resistive torque at joint 2 due to gravity = T_{2g}
Let resistive torque at joint 3 due to gravity = T_{3g}
Let resistive torque at joint 4 due to gravity = T_{4g}
Let resistive torque at joint 5 due to gravity = T_{5g}
Let resistive torque at joint 6 due to gravity = T_{6g}

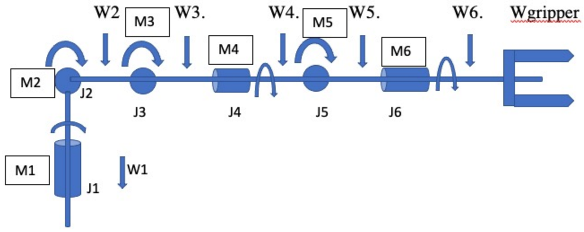
We have T_i = Iα, where i is for every joint.
Where , is considered to be the safe angular acceleration of each motor without wear and tear.
We will be utilizing Parallel Axis Theorem in order to obtain Torque due to inertia effect.

$$I = I_{cm} + ml^2$$

$M_1 \ldots\ldots M_6$ represents masses of motors respectively.
$I_{m1} \ldots\ldots$ I_{m6} represents respective moments of inertia of motors.
$I_{j1} \ldots\ldots$ I_{j6} represents respective moments of inertia of joints.
I_{L1}........ I_{L6}represents respective moments of inertia of Links respectively.
As per requirement , I_{xx}, I_{yy}, I_{zz}obtained from CAD model with following co-ordinate assumption



Hence for each joint-
Total Torque = Torque due to gravity + Torque due to inertial effect

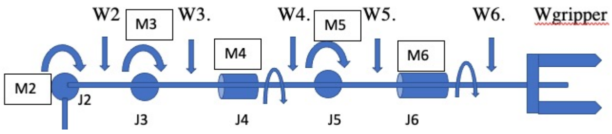


T_{1g} = 0 Nm since waist rotation does not cause motion of any link in the vertical plane (i.e. against gravity).

T_{1i}=[I_{L1}+I_{j2}+I_{L2}+$W_2H_1{}^2$+ I_{J3}+I_{L3}+$W_3H_2{}^2$+I_{J4}+I_{L4}+$W_4H_3{}^2$+I_{J5}+I_{L5}+$W_5H_4{}^2$

+I_{j6}+I_{L6}+$W_6H_5{}^2$+(Wgripper+payload)$H_6{}^2$]α

All inertia is calculated along Y-axis ,Where H_n is the perpendicular distance from Joint 1

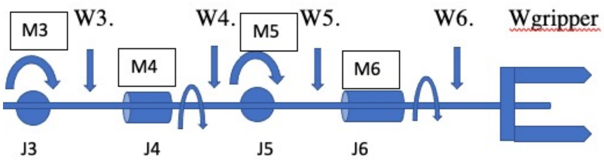


T_{2g} = W_2*L_2/2 + L_2*W_{j3} +(L_2+L_3/2)*W_3 + (L_2+L_3)*W_{j4}+
(L_2+L_3+L_4/2)*W_4+ (L_2+L_3+L_4)*W_{j5}+ (L_2+L_3+L_4+L_5/2)*W_5+
(L_2+L_3+L_4+L_5)W_{j6}+(L_2+L_3+L_4+L_5+L_6/2)*W_6+(L_2+L_3+L_4+L_5+L_6)(Wgripper+Wpayload)

T_{2i}=[I_{j2}+I_{L2}+$W_2H_1{}^2$+ I_{J3}+I_{L3}+$W_3H_2{}^2$+I_{J4}+I_{L4}+$W_4H_3{}^2$+I_{J5}+I_{L5}+$W_5H_4{}^2$

+I_{j6}+I_{L6}+$W_6H_5{}^2$+(Wgripper+payload)$H_6{}^2$]α

All inertia is calculated along Z-axis ,Where H_n is the perpendicular distance from Joint 2

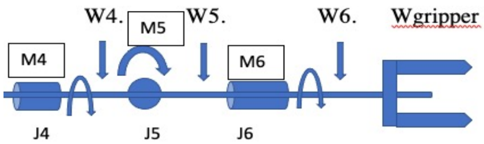


T_{3g} = L_3/2 *W_3 + (L_3)*W_{j4}+
(L_3+L_4/2)*W_4+ (L_3+L_4)*W_{j5}+ (L_3+L_4+L_5/2)*W_5+

$(L_3+L_4+L_5)W_{j6}+(L_3+L_4+L_5+L_6/2)*W_6+(L_3+L_4+L_5+L_6)(Wgripper+Wpayload)$
T_{3i}=[$I_{J3}+I_{L3}+W_3H_2{}^2+I_{J4}+I_{L4}+W_4H_3{}^2+I_{J5}+I_{L5}+W_5H_4{}^2$

$+I_{j6}+I_{L6}+W_6H_5{}^2+(Wgripper+payload)H_6{}^2]\alpha$

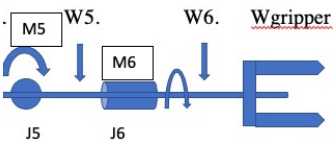
All inertia is calculated along Z-axis ,Where H_n is the perpendicular distance from Joint 3



$T_{4g} = 0$
T_{4i}=[$I_{J4}+I_{L4}+W_4H_3{}^2+I_{J5}+I_{L5}+W_5H_4{}^2$

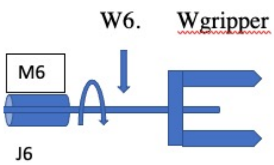
$+I_{j6}+I_{L6}+W_6H_5{}^2+(Wgripper+payload)H_6{}^2]\alpha$

All inertia is calculated along X-axis ,Where H_n is the perpendicular distance from Joint 4 and all zero



$T_{5g} = L_5/2*W_5+$
$(L_5)W_{j6}+(L_5+L_6/2)*W_6+(L_5+L_6)(Wgripper+Wpayload)$
$T_{5i}=[I_{J5}+I_{L5}+W_5H_4{}^2+I_{j6}+I_{L6}+W_6H_5{}^2+(Wgripper+payload)H_6{}^2]\alpha$

All inertia is calculated along Z-axis ,Where H_n is the perpendicular distance from Joint 5



$T_{6g} = 0$
$T_{6i}=[I_{j6}+I_{L6}+W_6H_5{}^2+(Wgripper+payload)H_6{}^2]\alpha$

All inertia is calculated along X-axis ,Where H_n is the perpendicular distance from Joint 6 and all zero

Effectively for each type of Joint , summation of Maximum torque from gravitational force and inertial effect is preferred .Ideally ,the one with maximum torque is considered for motor selection. Let T_{Ti} represents the total torque on any joint i then-

$$T_{T1} = T_{1g} + T_{1i}$$

$$T_{T2} = T_{2g} + T_{2i}$$

$$T_{T3} = T_{3g} + T_{3i}$$

$$T_{T4} = T_{4g} + T_{4i}$$

$$T_{T5} = T_{5g} + T_{5i}$$

$$T_{T6} = T_{6g} + T_{6i}$$

Ideally our motor should be of that joint torque, that has maximum torque value.

Path Planning & Trajectory Generation

While going from frame velocities to Actuator Velocities, we use Jacobian.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}_{6\times 1} = J_{6\times n} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dots \\ \dot{q}_n \end{bmatrix}_{n\times 1}$$

Where n is number of joints. Jacobian can be divided into two parts, where the first three rows are used for linear velocities, and the bottom three rows are used for angular velocities.

$$J = \begin{bmatrix} J_v \\ J_\omega \end{bmatrix}_{6\times n}$$

$$\dot{e} = \begin{bmatrix} \dot{e}_p \\ \dot{e}_o \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{r}_x \\ \dot{r}_y \\ \dot{r}_z \end{bmatrix} = \begin{bmatrix} \vartheta \\ \omega \end{bmatrix} = \begin{bmatrix} \vartheta_x \\ \vartheta_y \\ \vartheta_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = J \cdot \dot{q} = \begin{bmatrix} J_p \\ J_o \end{bmatrix} \cdot \dot{q} = \begin{bmatrix} \frac{\partial e_x}{\partial q0} & \frac{\partial e_x}{\partial q1} & \frac{\partial e_x}{\partial q2} & \frac{\partial e_x}{\partial q3} & \frac{\partial e_x}{\partial q4} & \frac{\partial e_x}{\partial q5} \\ \frac{\partial e_y}{\partial q0} & \frac{\partial e_y}{\partial q1} & \frac{\partial e_y}{\partial q2} & \frac{\partial e_y}{\partial q3} & \frac{\partial e_y}{\partial q4} & \frac{\partial e_y}{\partial q5} \\ \frac{\partial e_z}{\partial q0} & \frac{\partial e_z}{\partial q1} & \frac{\partial e_z}{\partial q2} & \frac{\partial e_z}{\partial q3} & \frac{\partial e_z}{\partial q4} & \frac{\partial e_z}{\partial q5} \\ \frac{\partial e_{r_x}}{\partial q0} & \frac{\partial e_{r_x}}{\partial q1} & \frac{\partial e_{r_x}}{\partial q2} & \frac{\partial e_{r_x}}{\partial q3} & \frac{\partial e_{r_x}}{\partial q4} & \frac{\partial e_{r_x}}{\partial q5} \\ \frac{\partial e_{r_y}}{\partial q0} & \frac{\partial e_{r_y}}{\partial q1} & \frac{\partial e_{r_y}}{\partial q2} & \frac{\partial e_{r_y}}{\partial q3} & \frac{\partial e_{r_y}}{\partial q4} & \frac{\partial e_{r_y}}{\partial q5} \\ \frac{\partial e_{r_z}}{\partial q0} & \frac{\partial e_{r_z}}{\partial q1} & \frac{\partial e_{r_z}}{\partial q2} & \frac{\partial e_{r_z}}{\partial q3} & \frac{\partial e_{r_z}}{\partial q4} & \frac{\partial e_{r_z}}{\partial q5} \end{bmatrix} * \dot{q}$$

Where $\dot{q}$ is actuators velocities and it can be written as

$$\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \\ \dot{q}_5 \\ \dot{q}_6 \end{bmatrix}$$

$$\boxed{\begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{\omega} \end{bmatrix} = \begin{bmatrix} J_p \\ J_o \end{bmatrix} \dot{\boldsymbol{q}}}$$

$$\begin{bmatrix} \boldsymbol{v} \\ \boldsymbol{\omega} \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = J\dot{q} = \begin{bmatrix} J_p \\ J_o \end{bmatrix} \dot{q} = \begin{bmatrix} \overbrace{\begin{matrix} \frac{\partial p_x}{\partial q_1} & \frac{\partial p_x}{\partial q_2} & \frac{\partial p_x}{\partial q_3} & \frac{\partial p_x}{\partial q_4} & \frac{\partial p_x}{\partial q_5} & \frac{\partial p_x}{\partial q_6} \\ \frac{\partial p_y}{\partial q_1} & \frac{\partial p_y}{\partial q_2} & \frac{\partial p_y}{\partial q_3} & \frac{\partial p_y}{\partial q_4} & \frac{\partial p_y}{\partial q_5} & \frac{\partial p_y}{\partial q_6} \\ \frac{\partial p_z}{\partial q_1} & \frac{\partial p_z}{\partial q_2} & \frac{\partial p_z}{\partial q_3} & \frac{\partial p_z}{\partial q_4} & \frac{\partial p_z}{\partial q_5} & \frac{\partial p_z}{\partial q_6} \end{matrix}}^{J_p \text{ 3 x 6 Position Jacobian}} \\ \underbrace{\begin{matrix} a_{x_0} & a_{x_1} & a_{x_2} & a_{x_3} & a_{x_4} & a_{x_5} \\ a_{y_0} & a_{y_1} & a_{y_2} & a_{y_3} & a_{y_4} & a_{y_5} \\ a_{z_0} & a_{z_1} & a_{z_2} & a_{z_3} & a_{z_4} & a_{z_5} \end{matrix}}_{J_o \text{ 3 x 6 Orientation Jacobian}} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \\ \dot{q}_5 \\ \dot{q}_6 \end{bmatrix}$$

$$J^{-1} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}_{6\times 1} = J^{-1} J \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dots \\ \dot{q}_n \end{bmatrix}_{n\times 1}$$

Now for Finding Jacobian , D-H parameters are used , for

Revolute Joints

$$J = \begin{bmatrix} J_v \\ J_w \end{bmatrix} = \begin{bmatrix} R^0_{i-1} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \times (d^0_n - d^0_{i-1}) \\ R^0_{i-1} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \end{bmatrix}$$

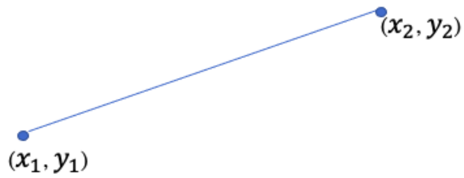
Where i = numbers of Joints in robotic manipulators R = Rotation Matrix from. D-H table and D-H matrix. d = vector factor from D-H table (We will not be using Modified D-H method for Jacobian matrix)

For our case Jacobian will be of form

$$\begin{pmatrix} R^0_0 \begin{matrix} 0 \\ 0 \\ 1 \end{matrix} (d^0_6 - d^0_0) \\ R^0_0 \begin{matrix} 0 \\ 0 \\ 1 \end{matrix} \end{pmatrix} \begin{pmatrix} R^0_1 \begin{matrix} 0 \\ 0 \\ 1 \end{matrix} (d^0_6 - d^0_1) \\ R^0_1 \begin{matrix} 0 \\ 0 \\ 1 \end{matrix} \end{pmatrix} \begin{pmatrix} R^0_2 \begin{matrix} 0 \\ 0 \\ 1 \end{matrix} (d^0_6 - d^0_2) \\ R^0_2 \begin{matrix} 0 \\ 0 \\ 1 \end{matrix} \end{pmatrix} \begin{pmatrix} R^0_3 \begin{matrix} 0 \\ 0 \\ 1 \end{matrix} (d^0_6 - d^0_3) \\ R^0_3 \begin{matrix} 0 \\ 0 \\ 1 \end{matrix} \end{pmatrix} \begin{pmatrix} R^0_4 \begin{matrix} 0 \\ 0 \\ 1 \end{matrix} (d^0_6 - d^0_4) \\ R^0_4 \begin{matrix} 0 \\ 0 \\ 1 \end{matrix} \end{pmatrix} \begin{pmatrix} R^0_5 \begin{matrix} 0 \\ 0 \\ 1 \end{matrix} (d^0_6 - d^0_5) \\ R^0_5 \begin{matrix} 0 \\ 0 \\ 1 \end{matrix} \end{pmatrix}$$

Path Planning and Trajectory generation

Lets say we plan a path to be straight Line , means our End Effector will go by the same path while travelling between two fixed points, lets say (x_1,y_1) and (x_2,y_2)



In robotics path equations are described by Parametric Form:

$$x(t) = \frac{v(x_2 - x_1)}{\sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2}} t + x_1$$

$$y(t) = \frac{v(y_2 - y_1)}{\sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2}} t + y_1$$

Where v represents the velocity of end effector EE , and these equation gives the position of EE while moving at time t.

Now the trajectory generation , differentiating these equation :

$$\dot{x}(t) = \frac{v(x_2 - x_1)}{\sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2}}$$

$$\dot{y}(t) = \frac{v(y_2 - y_1)}{\sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2}}$$

Plugging these in above equations we can calculate end effector EE velocities

$$\begin{bmatrix} \vartheta \\ \omega \end{bmatrix} = J\,\dot{q} \Leftrightarrow \dot{q} = J^{-1} \begin{bmatrix} \vartheta \\ \omega \end{bmatrix}$$

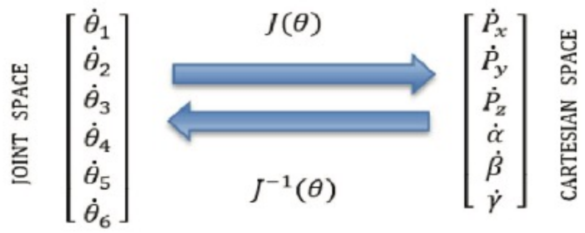
If we want to have path parallel to Y axis and move the end effector in a straight line along X- axis

For that we have to put $y_2 - y_1 = 0$ in above equations

$$x(t) = vt + x_1$$

$$y(t) = y_1$$

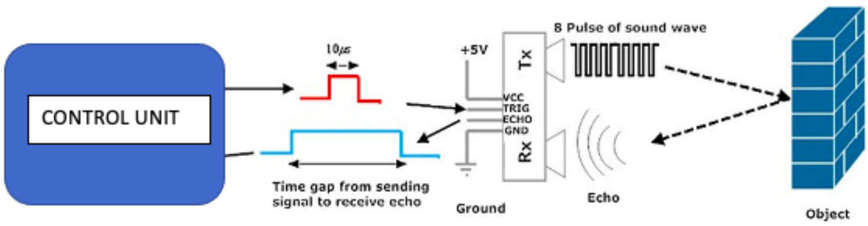
Similarly , If we want to have path parallel to X axis and move the end effector in a straight line along Y- axis, we can put $x_2 - x_1 = 0$



Computer Vision

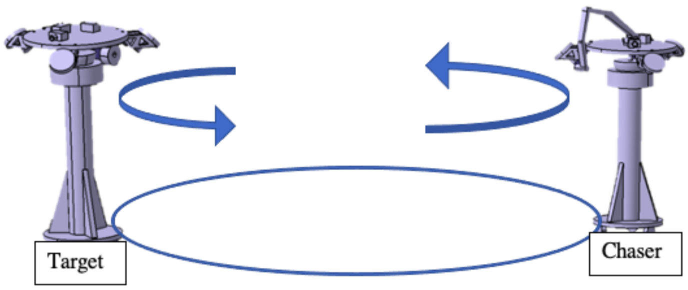
Once we have designed a control structure, it becomes important to design a system of putting the data into the system.This can be acieved in two ways - Camera Input & Sensor Input . For our project we will be using Ultrasonic sensor for inputing the data into the system.

Working principle of ultrasonic sensor: Ultrasonic Sensors emit short, high frequency (around 100 KHz to 50MHz) sound pulses at regular intervals by employing the transreciever and transducer. These propagate in the air at the velocity of sound. If they strike an object, then they reflected back as an echo signal to the sensor, which itself computes the distance to the target based on the time-span between emitting the signal and receiving the echo.



For our setup, we will require this to mimic rendezvous and docking. It is should continuously give the position of target, so that the control unit can process it and take decision accordingly.

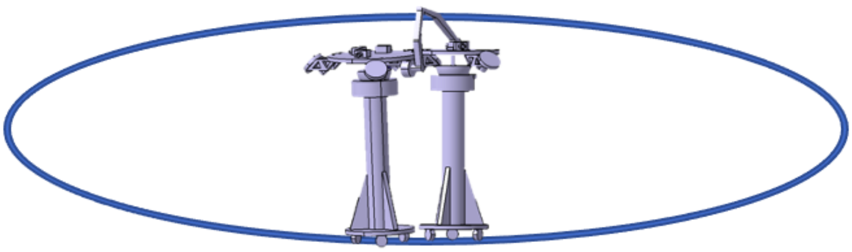
Rendezvous and Docking



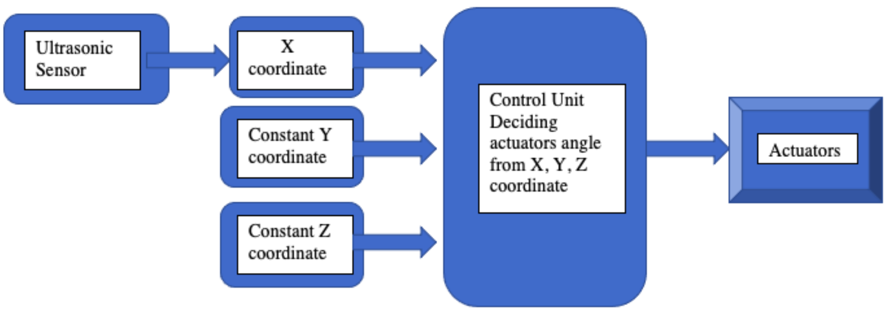
Target and Chaser are both in motion

Ultrasonic sensors embeded in target and chaser in order to calculate the orbital distance during rendezvous.This distance will eventually start sinking and finally becomes zero during docking. If the target and chaser are in range then ultrasonic sensor continue to give the distance between them and accordingly the control unit will decide the after trajectories. Also they can sense the orbital descend speed . there can also be the case , where target remains stationary ,while chaser continuosly decrese their orbital radius and finally dock with the target. Similar princile can be applied in that case also

Docking or berthing of spacecraft is the joining of two space vehicles . this connection can be temporary or permanent,or partially permanent such as for space station modules.It specially refers to joining of two separate free-flying space vehicles. In our case For the docking two scenerios arises - first when the target is stationary and chaser is in motion and second when they both are in motion.



Case 1 - When the target is stationary and the chaser is in motion,then the chaser will measure the distance of target(x cordinate) also the y and z cordinate wil be constant .Hence x,y,z are determined ans accordingly it can calculate the joint angles utilizing the calculations and theory made in above parts.Case 2 - When the target and chaser are both in motion ,when the relative velocity between them becomes zero they will dock and they should simultaneously decrease their orbital distances .

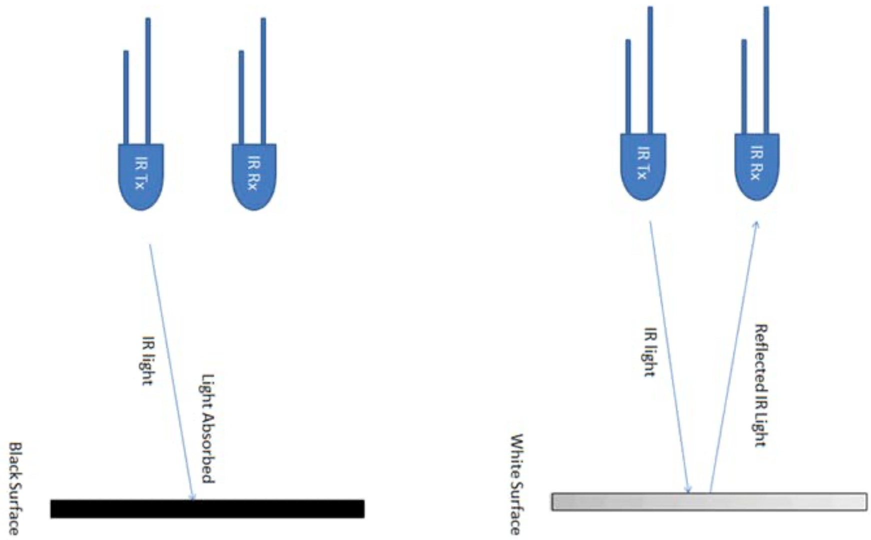


We can use this control structure in each case, while attempting for docking

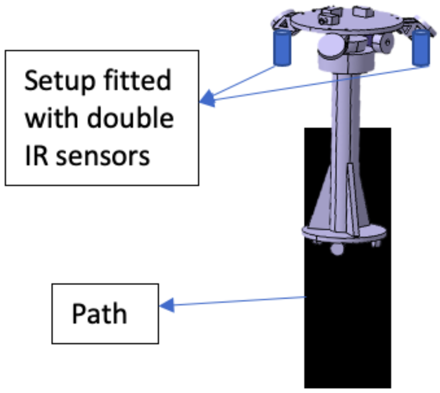
Trajectory Following

For our work, we require chaser and target to be in the same trajectory and they should follow each other and then dock as the time progresses. For achieving our goal, we will be using the path detection algorithm or path following algorithm. We will be using proximity device or sensor .A proximity sensor is a sensor able to detect the presence of nearby objects without any physical contact. A proximity sensor often emits an electromagnetic field or a beam of electromagnetic radiation (infrared, for instance), and looks for changes in the field or return signal.

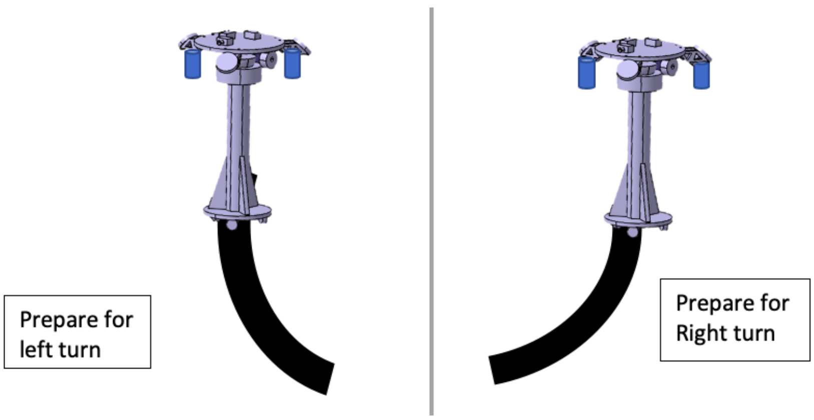
Concept of Line Following: Concept of working of line follower is related to light. We use here the behavior of light at black and white surface. When light fall on a white surface it is almost full reflected and in case of black surface light is completely absorbed. This behavior of light is used in building a line follower robot.

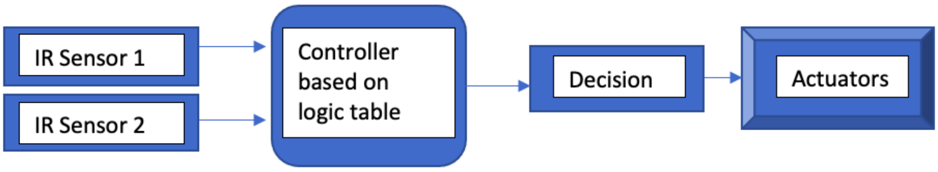


We will be using IR Transmitters and IR receivers also called photo diodes. They are used for sending and receiving light. IR transmits infrared lights. When infrared rays falls on white surface, it's reflected back and catched by photodiodes which generates some voltage changes. When IR light falls on a black surface, light is absorb by the black surface and no rays are reflected back, thus photo diode does not receive any light or rays.When sensor senses white surface then it gives 1 as input and when senses black line gives 0 as input, for a logic based controller.



When the setup moves forward, both the sensors wait for the line to be detected. If left sensor comes on black line then setup turn left side.If right sensor sense black line then setup turn right side until both sensor comes at white surface.When white surface comes setup starts moving on forward again.Also if both sensor detects black line, it stops.Below diagram explain the bove given method.





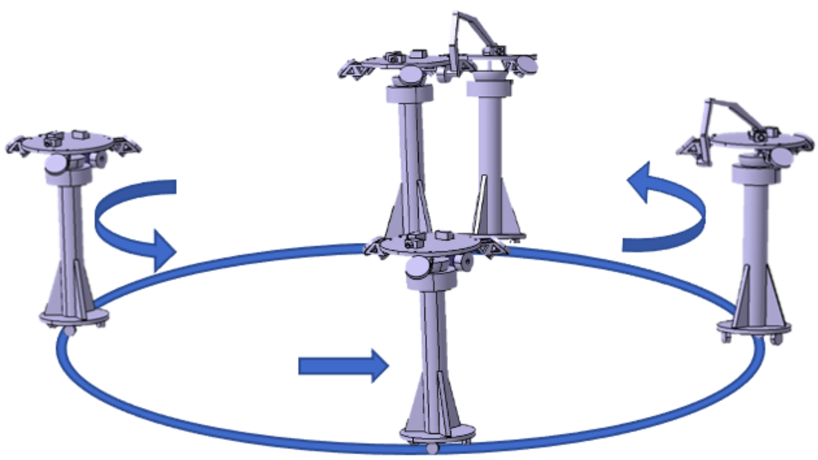
Control Structure

IR sensor 1 input(Left)	IR sensor 2 input(Right)	Result
0	0	Stop
1	0	Turn Right
0	1	Turn Left
1	1	Move

Logic table

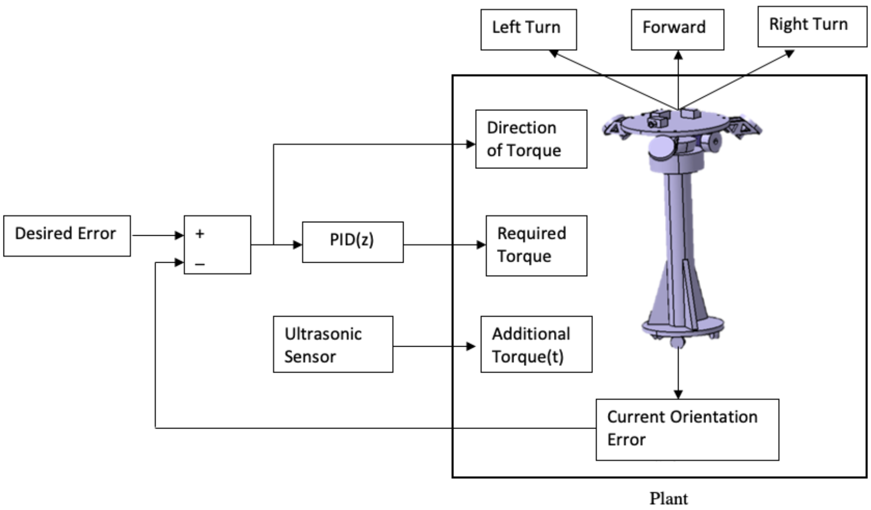
Relative Approach

In previous section, we have achieved the chaser and target to be in same orbit with almost similar velocities across orbit. In order to dock the chaser must increase its speed relative to target and when they are in range for docking, it should decrease its velocity and have to move in same velocities or we can say that the relative velocity between target and chaser to be zero. The point where the target should decrease the velocities can be sensed by Ultrasonic sensor fitted in chaser.

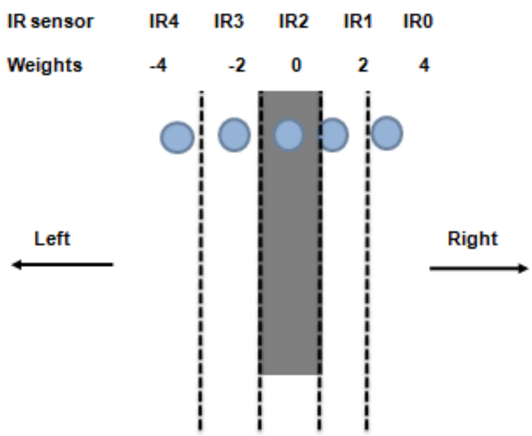


Modelling for Chaser-Target System

Based on previous theories, we will dessign a chaser-target model in simulink. If we go with the two sensor setup and do not use a controller then the system will be sudden, will give jerk on each turn. Hence the setup above may not perform well due to sudden jerks and also not safe. So we will design a closed-loop control system with PID controller based Chaser-Target model in order to achieve our goal.



In order to fine tune our model and achieve more efficiency, an array of five IR based sensors are used, a particular weight is assigned to each sensor according to their position on our system. Usually the weighted sum of the output of sensor will account for the desired torque and will provide more efficiency.



In the above figure, blue dots represent the positions of five IR sensors(IR0 - IR4). Each sensor output value 1 when below the threshold (bright or white background), 0 when above the threshold (dark or black background).

A weight is given to each sensor, with higher values to the extreme ones providing higher control signals when higher deviation is detected. The weighted current deviation is calculated from the below equation:

$$CurrentDeviation = \frac{\sum_0^4 sensors\,reading * corresponding\,weight}{number\,of\,sensors\,reading\,0}$$

When the system is centered on a thin black line, IR2 reads 0 while IR0, IR1, IR3, IR4 read 1.

Substituting the values in the equation, the current deviation is calculated as below:

$$CurrentDeviation = \frac{(1*-4)+(1*-2)+(0*0)+(1*2)+(1*4)}{1} = 0$$

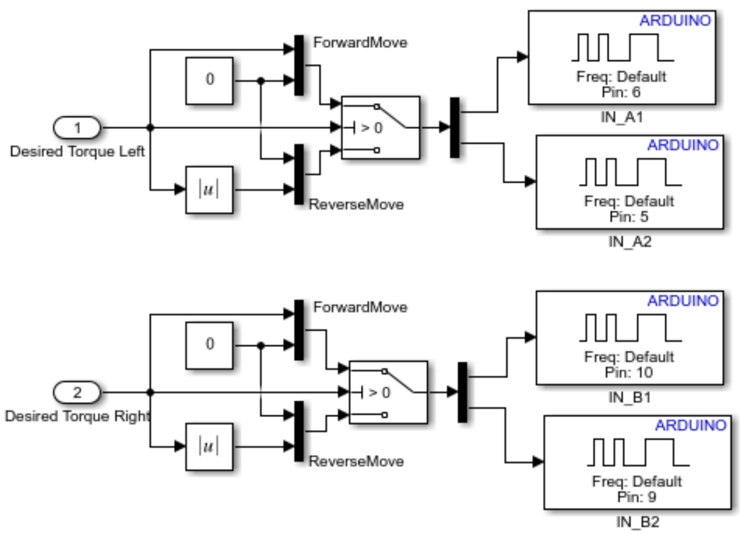
The Desired deviation when the robot is centered on a thin black line is set to a constant value of 0. The error is calculated as $Error = Desired\,error - Current\,orientation\,error$. In this case when the current deviation is 0, the error is 0 and the robot can move straight.positive error means right turn for the Arduino Robot > Left Motor torque increases, Right Motor torque decreases. A negative error for the Arduino Robot means left turn > Left Motor torque decreases, Right Motor torque increases.A summarized list of all possible errors is shown in the below table:

IR sensor reading	Deviation calculation	Current deviation	Error	Move
10001	(-4+4)/3	0	0	Straight
11011	(-4-2+2+4)/1	0	0	Straight
11111	0/0*	0	0	Straight
10000	-4/4	-1	1	Right
11000	(-4-2)/3	-2	2	Right
11100	(-4-2+0)/2	-3	3	Right
11110	(-4-2+0+2)/1	-4	4	Right
00001	4/4	1	-1	Left
00011	(2+4)/3	2	-2	Left
00111	(0+2+4)/2	3	-3	Left
01111	(-2+0+2+4)/1	4	-4	Left

Subsystem for Motor Direction and Control

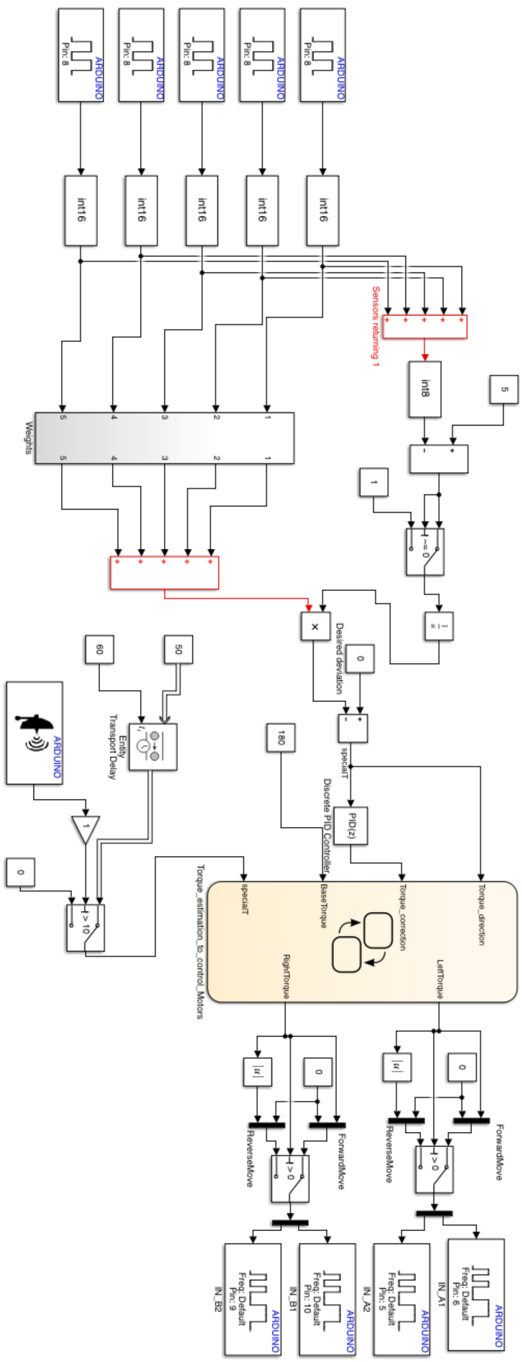
The motor driver controls the speed and the direction of both left and right motors by taking inputs from the microcontroller. Notice that the digital pins 6, 5, 10, 9 of the microcontroller are connected to IN_A1, IN_A2, IN_B1, IN_B2 of the motor driver. The inputs IN_A1, IN_B1 of the motor driver corresponds to the forward movement of the left and right motor respectively. IN_A2, IN_B2 corresponds to the reverse movement of the left and right motor respectively.

A pulse width modulated (PWM) signal can be used to control the motors. A positive input to the motor driver from the microcontroller corresponds to forward movement of the motors, whereas a negative input corresponds to reverse movement of the motors. For forward movement of both the motors, you have to apply the PWM signals to IN_A1 and IN_B1 and zero values to IN_A2 and IN_B2. Similarly for reverse movement, you have to apply the PWM signals only to IN_A2 and IN_B2 and zero values to IN_A1 and IN_B1.The PWM block accepts values from 0-255 (uint8). For the reverse movement, used an Abs block with Output data type -> int16, Integer rounding mode -> Round to convert the negative input value to positive and multiplex it with the Constant block used with value 0. Then used a Switch block with condition u2 > Threshold where u2 -> input and Threshold -> 0. If the condition is met then the forward movement multiplexed values are selected else reverse movement multiplexed values are selected.Then used a DEMUX block and connect the outputs to the 2 PWM blocks corresponding to IN_A1, IN_A2.



Subsystem for Motor Control

Simulink Model of Chaser



Simulink Model of Target