Indian Institute of Technology, Gandhinagar

ME 691-XI

Advanced Robotics

Semester–I, Academic Year 2023-24

Class Project- 1
FRANKA EMIKA PANDA 7-DoF ROBOT

By

Team: *AK 47*

Anavart Pandya 20110016

Kush Patel 20110131

1. MOTIVATION :

The selection of the Franka Emika Panda Robot for our class project on the analysis and simulation of a robotic manipulator is a result of careful consideration, as it is imperative that the choice of the robot aligns with the goals and objectives of the project. In this section, we will delve into the detailed motivation behind selecting the Franka Emika Panda Robot, emphasizing its critical aspects and relevance, including its unique approach to singularity.

Figure 1: [Franka Emika Panda 7-Dof Robot]

- **Degrees of Freedom (DOF):** The primary reason for choosing the Franka Emika Panda Robot is its impressive 7 degrees of freedom (7-DOF). Having a high number of DOF is crucial for accomplishing complex manipulation tasks with flexibility and precision. It allows for a wide range of motion, making it suitable for a variety of applications, from pick-and-place tasks to more intricate operations.

- **End-Effector Dexterity**: The end-effector of the Panda Robot is equipped with a soft and adaptable gripper. This feature enhances its dexterity, making it capable of handling objects with different shapes, sizes, and materials. This dexterity is invaluable for real-world applications where the robot needs to manipulate objects in a human-like manner.

- **Singularity Mitigation:** The Franka Emika Panda Robot is known for its innovative approach to mitigating singularities. The joint offsets in the Panda's design make it comparatively less prone to singularities than other robotic manipulators. Singularities can cause erratic behavior and complicate control, so the Panda's reduced susceptibility to them is a significant advantage, even though it may introduce some complexity in inverse kinematics.

- **Research Community Recognition:** The Franka Emika Panda Robot has gained recognition and popularity within the research community. It has been adopted in numerous academic and industrial projects, resulting in a wealth of research papers, documentation, and resources. This extensive body of work provides valuable insights and references for our project, ensuring that we have access to a robust knowledge base for analysis and simulation.

- **Real-World Applicability:** The Panda Robot is designed to be used in real-world scenarios, particularly in collaborative and industrial settings. Its advanced safety features, including torque sensors in each joint, enable it to work safely alongside humans. This makes it a relevant choice for projects that aim to bridge the gap between simulation and practical application.

- **Availability of Simulation Environment:** We have chosen CoppeliaSim as our simulation software, and the Franka Emika Panda Robot is well-supported within this environment. The availability of an accurate simulation model and a user-friendly interface greatly facilitates our simulation efforts, enabling us to explore and validate various manipulation tasks efficiently.

- **Educational Value**: The Franka Emika Panda Robot is widely used in educational institutions and robotics courses. Its adoption in the academic world ensures that we have access to comprehensive learning resources, including tutorials, software libraries, and teaching materials. This enhances our ability to understand, model, and simulate the robot effectively.

- **Relevance to Modern Robotics:** The selection of the Franka Emika Panda Robot reflects our commitment to staying at the forefront of modern robotics. As robotics technology continues to evolve, the Panda Robot is a prime example of state-of-the-art robotic systems, making it an ideal subject for analysis and simulation in an academic context.

2. TASK REPRESENTATION :

- Configuration Space : $S^1 \times S^1 \times S^1 \times S^1 \times S^1 \times S^1 \times S^1 = T^7$
- Task Space : $R^3 \times S^2 \times S^1$ [3 $Translation$ & 3 $Rotational$]
- Work Space :
    - Using the given joint limits, work space can be calculated although it requires a significant amount of computational power.
    - The Panda robot's unique kinematic design, featuring seven revolute joints, includes a spherical shoulder, two-offset elbow, and a non-spherical wrist. This configuration expands the robot's workspace by eliminating inaccessible areas near its base.

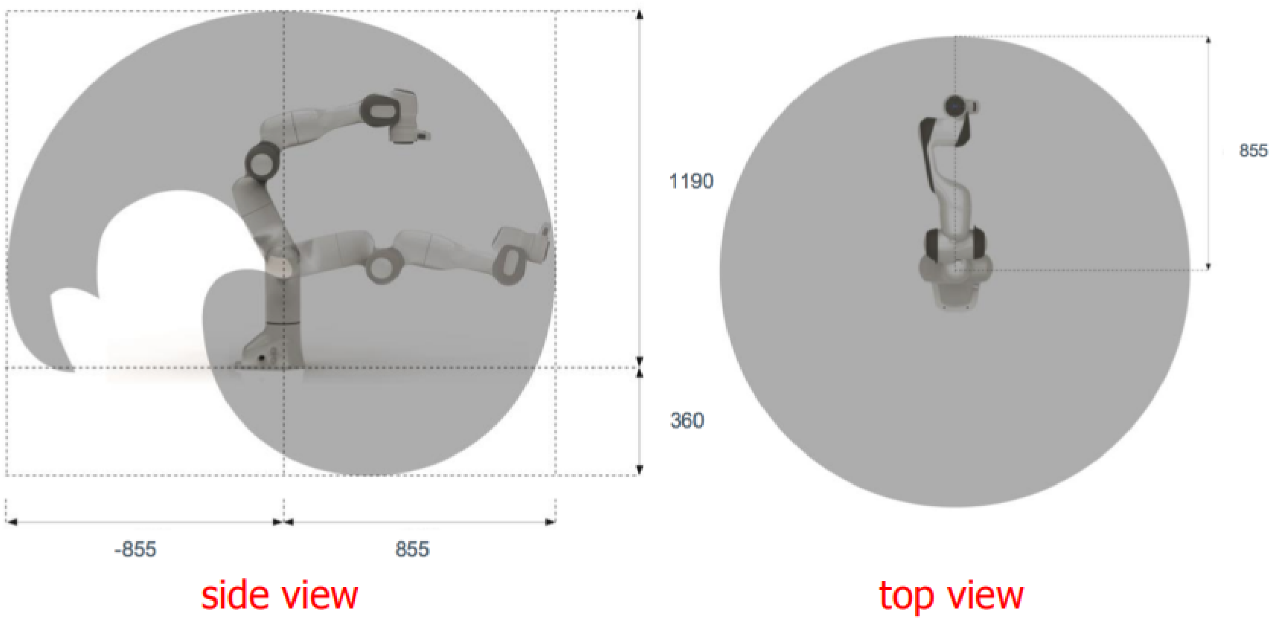

Figure 2: Workspace of Panda Robot

Name	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6	Joint 7	Unit
q_{max}	2.8973	1.7628	2.8973	-0.0698	2.8973	3.7525	2.8973	rad
q_{min}	-2.8973	-1.7628	-2.8973	-3.0718	-2.8973	-0.0175	-2.8973	rad

Figure 3: Joint space limits of Panda Robot

3. POSITION ANALYSIS:

Position analysis is a fundamental aspect of understanding the configuration and movement of robotic manipulators. In the case of the Franka Emika Panda Robot, we employ a thorough position analysis followed by forward kinematics and inverse kinematics. These analyses are crucial for mapping the robot's joint space to its workspace.

3.1 Forward Kinematics:

Forward kinematics allows us to determine the end-effector's pose (position and orientation) given the joint angles of the robot. The Franka Emika Panda Robot features seven degrees of freedom (7-DOF), which makes forward kinematics essential for mapping the complex relationships between its joints and its end-effector.

To perform forward kinematics, we have adopted the Denavit-Hartenberg (DH) representation. However, rather than utilizing the classical standard DH parameters, we have chosen to employ the modified DH representation based on a specific paper titled "Analytical Inverse Kinematics for Franka Emika Panda – a Geometrical Solver for 7-DOF Manipulators with Unconventional Design." This decision is grounded in the practical advantages that modified DH parameters offer. Their reordering of the parameters, specifically employing "a, d, α, θ" instead of the standard "a, α, d, θ" results in a more intuitive and simplified mathematical framework for computing the transformation matrices, ultimately streamlining the entire forward kinematics process. The modified DH representation employs a more intuitive approach by specifying the joint axes to intersect at the joint centers. This choice simplifies the physical interpretation of the joint parameters and their geometric arrangement. The modified DH parameters for each joint are as follows: *[We assume Flange(F) as our end effector and hence neglect the last DH parameter row (End effector (EE)) for further analysis in this report.]*

Frame	a (m)	d (m)	α (rad)	θ (rad)
Joint 1	0	0.333	0	q_1
Joint 2	0	0	$-\pi/2$	q_2
Joint 3	0	0.316	$\pi/2$	q_3
Joint 4	0.0825	0	$\pi/2$	q_4
Joint 5	-0.0825	0.384	$-\pi/2$	q_5
Joint 6	0	0	$\pi/2$	q_6
Joint 7	0.088	0	$\pi/2$	q_7
Flange (F)	0	0.107	0	0
End effector (EE)	0	0.1034	0	$\pi/4$

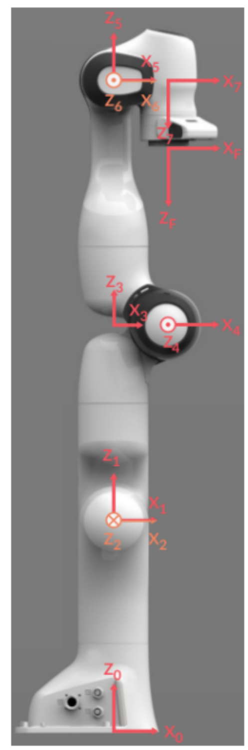

Figure 4: Modified DH parameters of Panda Robot

These DH parameters allow us to construct a specific transformation matrix for each joint's transformation with respect to the previous joint. The transformation matrix for the modified DH parameters takes the following form:

$$\begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & a \\ \sin(\theta)\cos(\alpha) & cos(\theta)\cos(\alpha) & -\sin(\alpha) & -\sin(\alpha)*d \\ \sin(\theta)sin(\alpha) & cos(\theta)sin(\alpha) & \cos(\alpha) & cos(\alpha)*d \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Where , θ $-$ *Joint angle*

α $-$ *Link twist angle*

a $-$ *Link length*

d $-$ *Link offset*

The final homogeneous transformation matrix for the Panda Robot, constructed using the DH parameters, provides us with the end-effector's pose concerning the robot's base frame. This matrix is an essential component for inverse kinematics. Our FK code [*Panda_FK_IK_Analysis.py*] leverages two main functions:

- *def final_transform*(*dh_params*) : This function computes the final homogeneous matrix using the provided DH parameters.
- *def dh_transform*(*a, d, alpha, theta*) : It calculates a single transformation matrix based on the DH parameters which is shown above.

3.2 Inverse Kinematics:

Inverse kinematics is the process of finding the joint angles necessary to achieve a desired end-effector pose. In our analysis of the Panda Robot, we've implemented an inverse kinematics code that employs an analytical geometric approach rather than a numerical one, ensuring precision and reliability.

3.2.1 Analytical Geometrical Approach for IK :

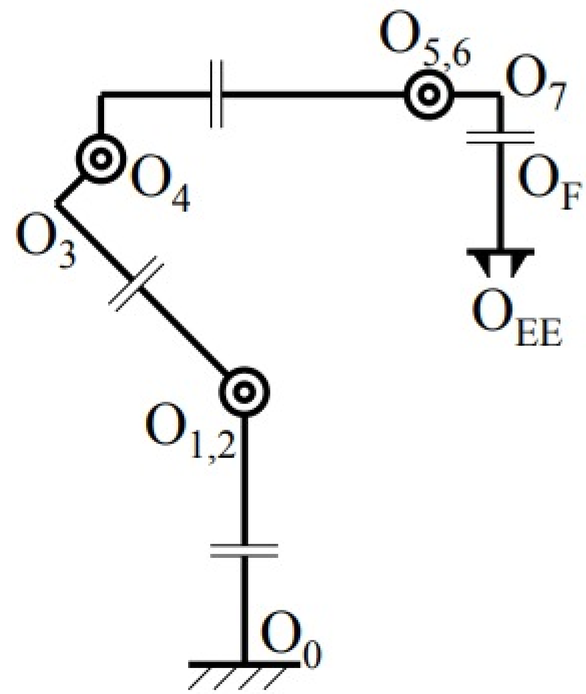

Figure 5: Kinematic chain of Franka Emika Panda

The analytical approach used for Franka Emika Panda's kinematics can be extended to other 7-DOF anthropomorphic manipulators with the following general procedure:

- Choose q7 as the redundancy parameter.
- Analyze the triangle $\triangle O_2O_4O_6$ to determine two potential solutions for q4.
- For each q4 case, solve for two possible values of q6.
- For each q4-q6 combination, determine the two possibilities for the (q1, q2) pair.
- For each q4-q6-(q1, q2) combination, calculate the unique values of q3 and q5.

Fixing q7 as redundancy parameter:

To ensure a unique solution for our 7-DOF manipulator, we addressed redundancy by fixing the parameter q7. This choice became necessary as the offset on Link 6 prevented independent motion of the last joint without introducing undesired translational effects on Frame 6. Therefore, we selected q7 as the redundancy parameter and considered its value as known for the subsequent calculations. This approach allowed us to resolve redundancy effectively.

Calculating q4:

Theoretically, there are two equivalent scenarios, denoted as A1 and A2 in our study, for solving q4, as illustrated in Fig.6. The triangular configurations $\Delta O_2O_4O_6$ in Fig. 6.a and Fig. 6.b exhibit symmetry about the line O_2O_6, resulting in identical positions for Frame 6 and, consequently, the same end effector pose. This phenomenon, known as the "elbow-up/down" bifurcation, is observed in various robotic systems. However, for the Franka Emika Panda robot, the shape of its elbow and mechanical joint limits severely restrict joint motion in Case A1, where q4 is confined to the range of [-26.76◦, -4◦]. As a result, A2 is the practical choice explored in this approach. Should the need arise, A1 can be similarly addressed.

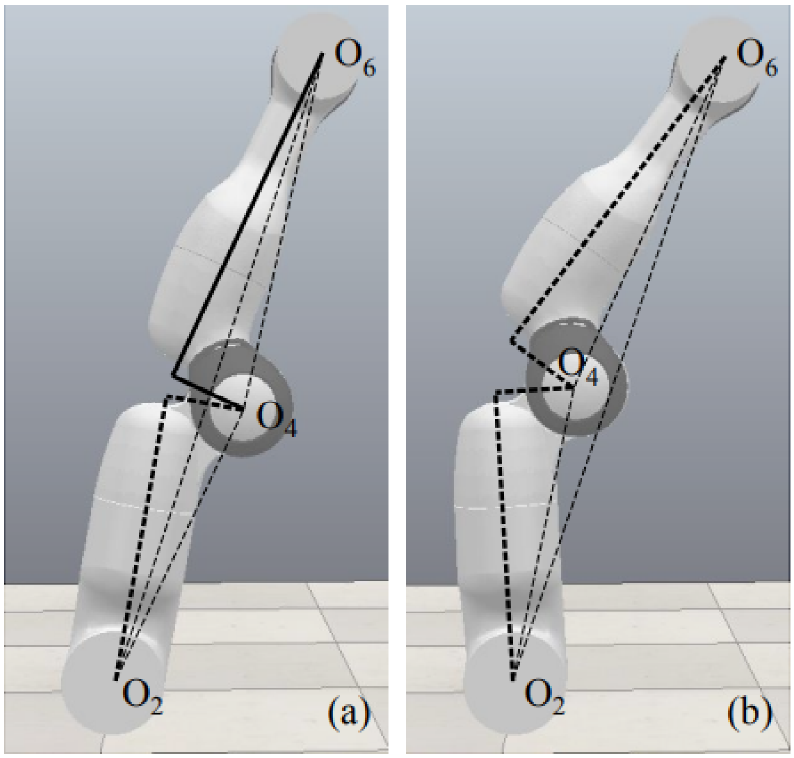

Figure 6: Two equivalent cases when solving Joint 4 angle, with Case A1 in (a) and A2 in (b)

$$q_4 = \angle O_2O_4O_3 + \angle HO_4O_6 + \angle O_2O_4O_6 - 2\pi$$

with

$$\angle O_2O_4O_3 = \operatorname{atan}(d_3/a_4)$$
$$\angle HO_4O_6 = \operatorname{atan}(d_5/|a_5|)$$
$$\angle O_2O_4O_6 = \operatorname{acos}\frac{\overline{O_2O_4}^2 + \overline{O_4O_6}^2 - \overline{O_2O_6}^2}{2 \cdot \overline{O_2O_4} \cdot \overline{O_4O_6}}$$

(NOTE: Frame sign convention is as follows:- in any A_i^j, base frame is ***'j'*** and current frame ***'i'*** , in other words, from 'i'th joint to 'j'th joint)

We are taking Flange(F) as end effector so that the position of Frame 7 can be first calculated as

$$p_7 = p_{EE} - (d_F)z_{EE}$$

x_6 can be represented in the end effector frame as

$$x_6^{EE} = [\ cos(-\ q_7)\ \ sin(-\ q_7)\ \ 0]^T$$

And it's represents in the world frame is

$$x_6 = R_{EE} \cdot x_6^{EE}$$

The position of Frame 6 and its distance to Frame 2 can then be obtained

$$\boldsymbol{p}_6 = \boldsymbol{p}_7 + \overrightarrow{\mathrm{O}_7\mathrm{O}_6} = \boldsymbol{p}_7 - a_7\boldsymbol{x}_6$$
$$\overline{\mathrm{O}_2\mathrm{O}_6} = ||\overrightarrow{\mathrm{O}_2\mathrm{O}_6}|| = ||\boldsymbol{p}_6 - \boldsymbol{p}_2||$$

where

$$\boldsymbol{p}_2 = [\ 0\ \ 0\ \ d_1\ ]^{\mathrm{T}}$$

Calculating q6:

In the context of solving for q6, there are two equivalent scenarios, as depicted in Fig. 7. It's important to note that, due to joint limitations, the pose shown in Fig. 7.b is not physically achievable, but it is presented in the illustration to illustrate the underlying concept. The triangular configurations $\Delta O_2 O_4 O_6$ in these two cases still exhibit symmetry about the line $O_2 O_6$. However, the key distinction between this bifurcation and the previous one concerning q4 is that in this case, q4 remains constant, while q3 and q5 change by 180 degrees.

By analyzing the robot's structure, we can deduce the following two conditions:

- Points O_5, O_6, O_7, O_{EE} and H lie in the same plane.
- The angle between the z_5 and the vector from O2 to O6 is equal to the angle $\angle O_2 O_6 H$.

These conditions enable us to derive a representation of the z5 axis within Frame 6.

$$\begin{aligned} {}^6\boldsymbol{z}_5 &= [\ \cos(-q_6 + \pi/2)\ \ \ \sin(-q_6 + \pi/2)\ \ \ 0\ ]^{\mathrm{T}} \\ &= [\ \sin(q_6)\ \ \ \cos(q_6)\ \ \ 0\ ]^{\mathrm{T}} \end{aligned}$$

where the $\frac{\pi}{2}$ offset is a result of the manufacturer definition of the zero pose, and the following condition holds

$${}^6\boldsymbol{z}_5 \cdot {}^6\overrightarrow{\mathrm{O}_2\mathrm{O}_6} = \overline{\mathrm{O}_2\mathrm{O}_6} \cdot \cos\angle\mathrm{O}_2\mathrm{O}_6\mathrm{H}$$

$${}^6\overrightarrow{\mathrm{O}_2\mathrm{O}_6} = [\ {}^6x_{26}\ \ {}^6y_{26}\ \ {}^6z_{26}\ ]^{\mathrm{T}} = \boldsymbol{R}_6^{\mathrm{T}} \cdot \overrightarrow{\mathrm{O}_2\mathrm{O}_6}$$

$$\sqrt{{}^6x_{26}^2 + {}^6y_{26}^2}\sin(q_6 + \phi_6) = \overline{\mathrm{O}_2\mathrm{O}_6} \cdot \cos\angle\mathrm{O}_2\mathrm{O}_6\mathrm{H}$$

with

$$\phi_6 = \mathrm{atan2}\left({}^6y_{26}, {}^6x_{26}\right)$$

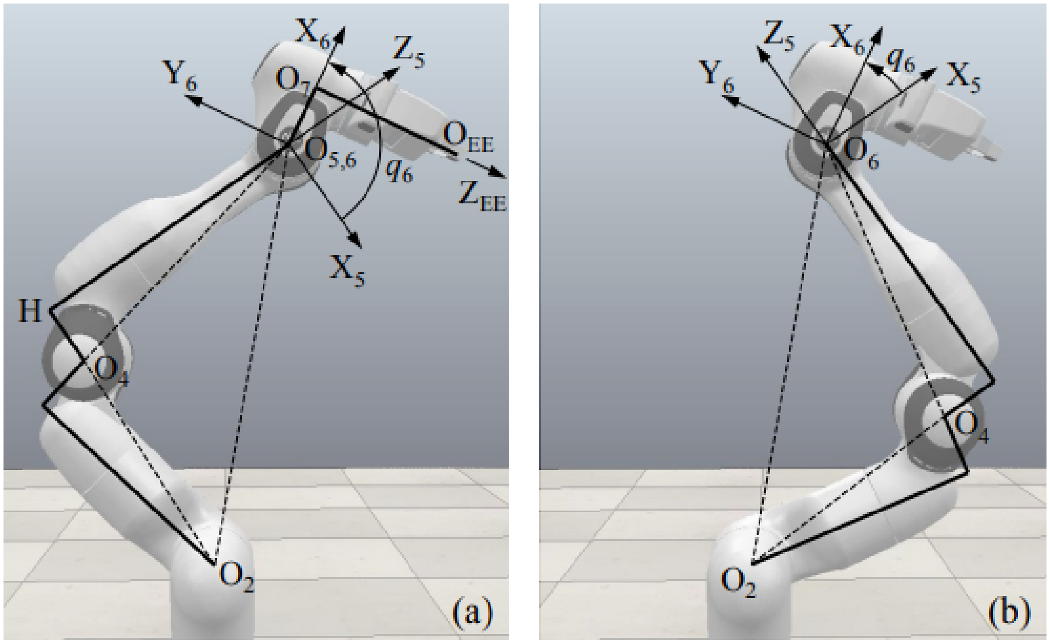

Figure 7: Two cases for calculating Joint 6 angle (B1 and B2)

Due to robot design, ϕ_6 is never singular. Let

$$\psi_6 = \operatorname{asin}\left(\frac{\overline{\mathrm{O_2O_6}} \cdot \cos\angle \mathrm{O_2O_6H}}{\sqrt{{}^6x_{26}^2 + {}^6y_{26}^2}}\right)$$

Finally, two possibilities of q8 exist as

$$q_6 = \begin{cases} \pi - \psi_6 - \phi_6 + 2k\pi \\ \psi_6 - \phi_6 + 2k\pi \end{cases}$$

The integer k should be selected in such a way that q6 is within its joint limit range. These two q6 solutions above will be referred to as Case B1 and B2, respectively.

Calculating q1 and q2:

In Figure 7.a, we denote the intersection point of the lines O_6H and O_2O_3 as P. The direction of the vector from O2 to P, represented as O2P, determines the joint angles q1 and q2. This determination is based on the fact that

$$\angle O_2O_6P = \angle O_2O_6H$$
$$\angle PO_2O_6 = \angle O_3O_2O_4 + \angle O_4O_2O_6$$

Due to the mechanical limit of Joint 4, $\angle O_2PO_6$ is never zero or π, therefore according to the law of sines

$$\overline{\mathrm{PO_6}} = \overline{\mathrm{O_2O_6}} \cdot \frac{\sin \angle \mathrm{PO_2O_6}}{\sin \angle \mathrm{O_2PO_6}}$$

Vector O_2P is then calculated by

$$\overrightarrow{\mathrm{O_2P}} = \overrightarrow{\mathrm{O_2O_6}} + \overrightarrow{\mathrm{O_6P}} = \overrightarrow{\mathrm{O_2O_6}} - \overline{\mathrm{PO_6}} \cdot \boldsymbol{R}_6 \cdot {}^6\boldsymbol{z}_5$$

By denoting the elements of vector O_2P as $[x_{2P} \ \ y_{2P} \ \ z_{2P}]^T$, there are the possible 2 pair of the solution for q1 and q2

$$q_1 = \mathrm{atan2}\left(y_{2\mathrm{P}}, x_{2\mathrm{P}}\right) \qquad q_1 = \mathrm{atan2}\left(-y_{2\mathrm{P}}, -x_{2\mathrm{P}}\right)$$
$$q_2 = \mathrm{acos}\left(z_{2\mathrm{P}}/\overline{\mathrm{O_2P}}\right) \qquad q_2 = -\mathrm{acos}\left(z_{2\mathrm{P}}/\overline{\mathrm{O_2P}}\right)$$

The occurrence of a singularity problem is a possibility in this context. It arises when both x_{2P} and y_{2P} become zero simultaneously, causing potential issues with the *atan2* function. This particular situation materializes when the vector O_2P points directly upward, essentially setting q2 to 0. As illustrated in Fig. 8, the links between Joint 1 and 3 have the freedom to rotate about z_0 independently, without affecting the rest of the robot body. Consequently, this scenario generates an infinite number of solutions for q1 and q3. To resolve this, a predetermined value must be assigned to q1 to ensure a unique and well-defined result.

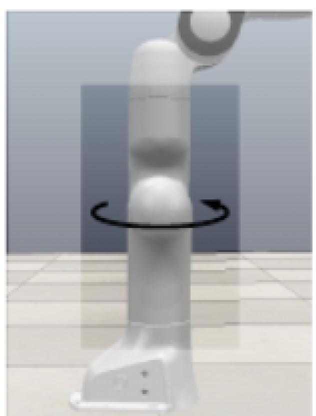

Figure 8: IK Singularity caused by q2=0

Calculating q3:

As depicted in Fig.9, the joint angle q3 corresponds to the rotation angle of the vector from O2 to M (represented as O_2M) with respect to the x2 axis. This vector O_2M is a projection of the x3 axis onto the $Z_2 - X_2$ plane. Frame 3's Y axis is orthogonal to the triangle ΔO_2PO_6, and as a result, y3 is the normalization of the cross product between vectors O_2P and O_2O_6. Since O_2P and O_2O_6 are never collinear due to Joint 4's limitations, y3 remains non-singular. The z3 axis is merely the normalization of vector O_2P. This assignment follows the right-hand rule. $x_3 = y_3 \times z_3$

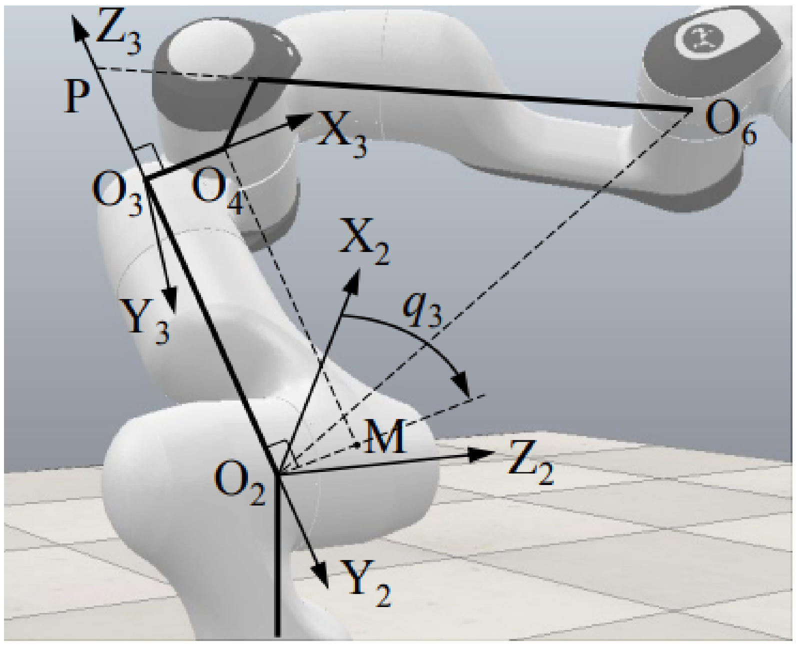

Figure 9: Calculating Joint 3 angle

With q1 and q2 solved in the previous section, the orientation of Frame 2 can be obtained by forward kinematics

$$\boldsymbol{R}_2 = \boldsymbol{R}_1(q_1) \cdot {}^1\boldsymbol{R}_2(q_2)$$

After transforming x3 into Frame 2 and extracting its internal elements

$${}^2\boldsymbol{x}_3 = [\, {}^2x_{x3} \quad 0 \quad {}^2z_{x3} \,]^{\mathrm{T}} = \boldsymbol{R}_2^{\mathrm{T}} \cdot \boldsymbol{x_3}$$

the value of q3 can be calculated with

$$q_3 = \operatorname{atan2}\left({}^2z_{x3}, {}^2x_{x3}\right)$$

Calculating q5:

Defining S as the projection of O_4 onto the $X_5 - Y_5$ plane, the joint angle q5 can be interpreted as the rotational angle of the x5 axis from the vector O_5S, as illustrated in Fig.10. The vector HO_4 is equal to

$$\overrightarrow{\mathrm{HO}_4} = \boldsymbol{p}_4 - \boldsymbol{p}_{\mathrm{H}} = \boldsymbol{p}_4 - (\boldsymbol{p}_6 - d_5\boldsymbol{z}_5)$$

With q6 is already solved, O_5S can be computed by

$$\overrightarrow{\mathrm{O}_5\mathrm{S}} = \left[\ {}^5x_{5\mathrm{S}} \quad {}^5y_{5\mathrm{S}} \quad 0\ \right]^{\mathrm{T}}$$
$$= \boldsymbol{R}_5^{\mathrm{T}} \cdot \overrightarrow{\mathrm{HO}_4} = {}^5\boldsymbol{R}_6\left(q_6\right) \cdot \boldsymbol{R}_6^{\mathrm{T}} \cdot \overrightarrow{\mathrm{HO}_4}$$

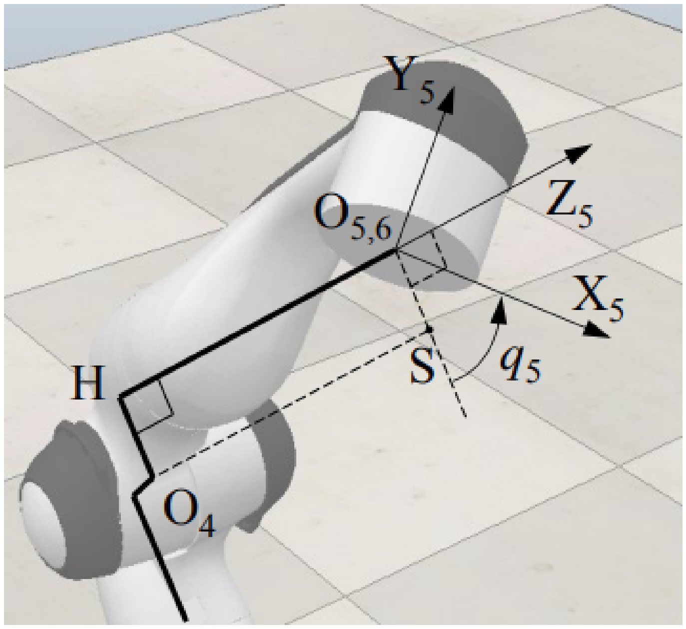

Figure 10: Calculating Joint 5 angle

And the solution to q5 can be obtained as

$$q_5 = -\operatorname{atan2}\left({}^5y_{5\mathrm{S}}, {}^5x_{5\mathrm{S}}\right)$$

IK Code:
We implemented the geometric approach for obtaining the inverse kinematics of the Panda robot and translated it into Python code (*Panda_IK.py*). This code allows us to determine the joint angle configuration based on a given homogeneous matrix obtained from the forward kinematics procedure.

3.2.2 Validation:

Joint Trajectory Generation:

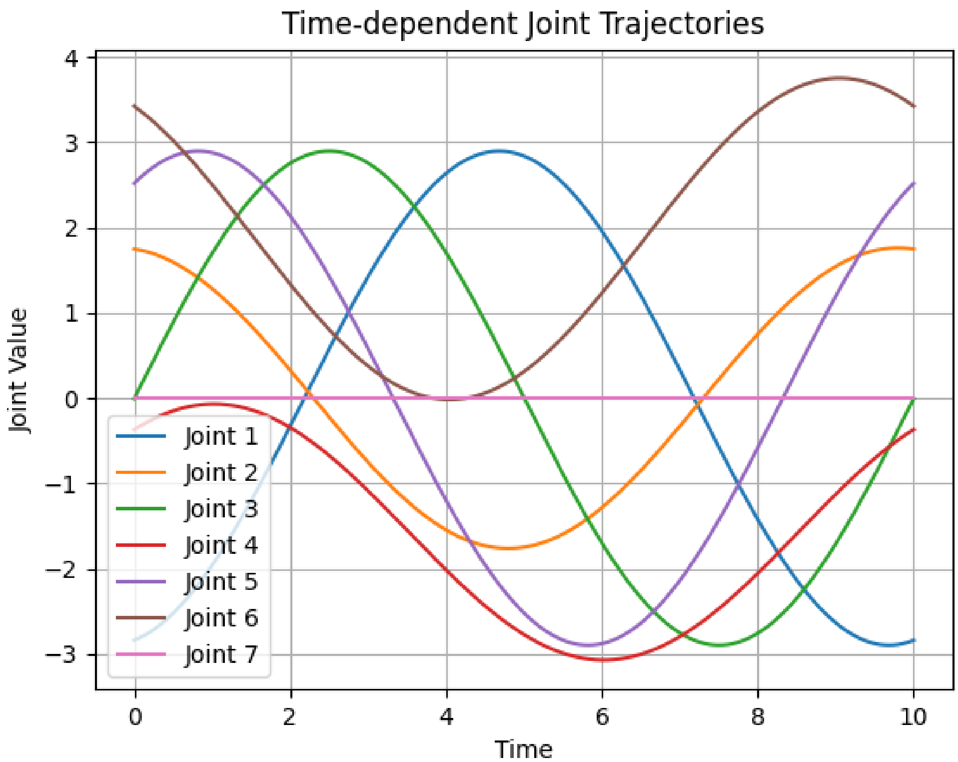

Figure 11: Joint Trajectory vs Time Graph

For validation perspective, we created a joint angle trajectory within the joint angle limits. We generated a sinusoidal trajectory for each joint within these limits over a specified time range. To add variety, we introduced a random phase shift (phi) to each joint's trajectory while ensuring that it remains within the prescribed limits. The resulting trajectory smoothly varies the joint angles for the robot as Fig.11, taking into account both the joint limits and the randomized phase shift. The reason behind taking the joint limits is that the trajectory lies in the configuration space to avoid uncertain circumstances in later calculations.

Cartesian Trajectory Comparison:

After obtaining joint angle trajectories, we utilize them in the forward kinematics process to calculate a homogeneous transformation matrix. From this matrix, we extract the actual 3D position and orientation parameters (Euler angles) representing the robot's end-effector coordinates. Subsequently, we apply this transformation matrix to the inverse kinematics method to derive four potential configurations (solutions) for the robot's joint angles, while considering joint limits. We then select a valid configuration among these solutions.

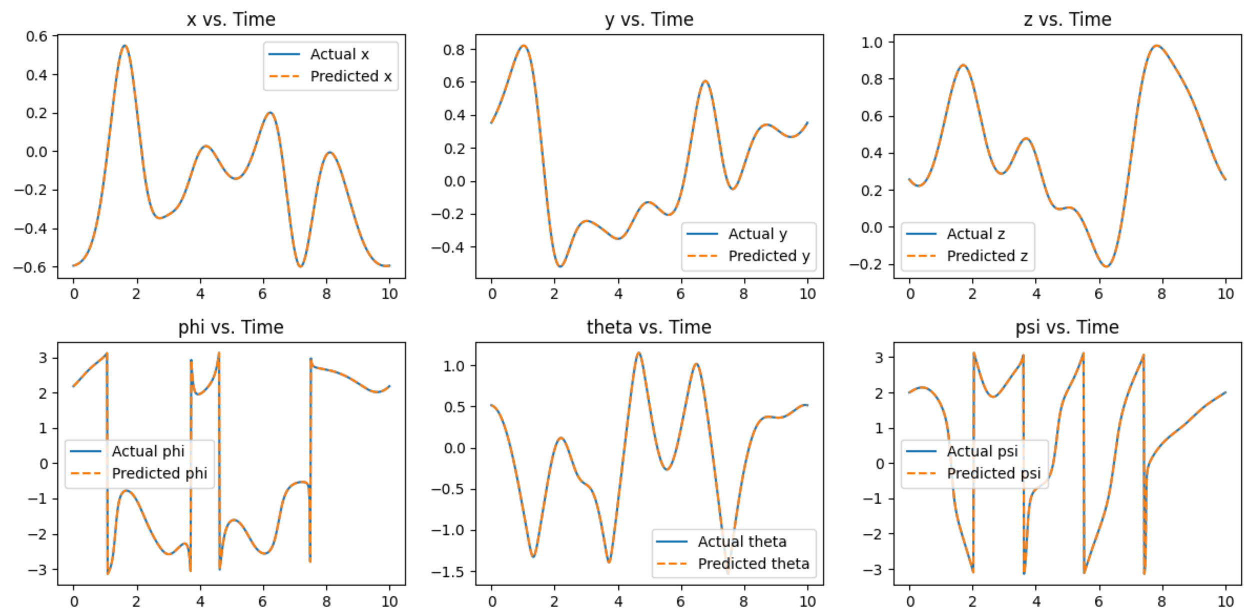

Figure 12: Comparison for Trajectory - 1

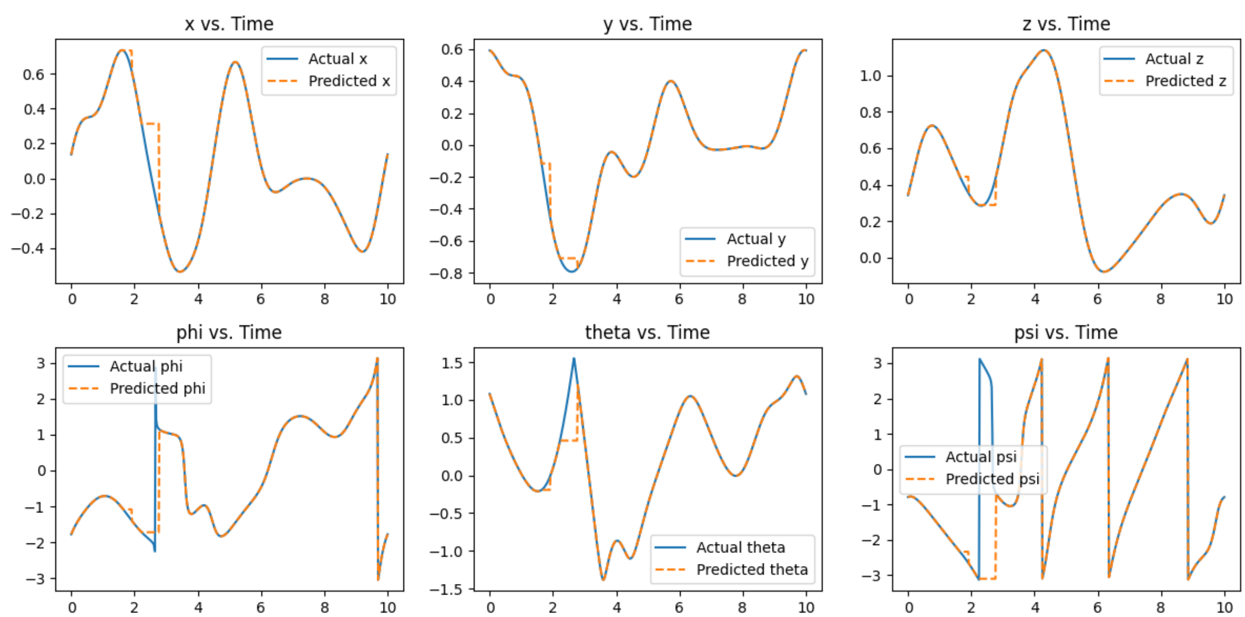

Figure 13: Comparison for Trajectory - 2

By feeding this configuration back into the forward kinematics, we obtain predicted 3D position and orientation parameters. The final step involves comparing the actual and predicted coordinates, as depicted in Fig.12, across three distinct trajectories. Remarkably, the congruence between these coordinates provides strong validation for the accuracy and reliability of our forward kinematics and analytical inverse kinematics approaches.

If we encounter a situation where a valid configuration cannot be obtained for a particular instance, we revert to using the joint angles from the previous iteration. This observation is visually represented in Fig.13, highlighting our adaptive approach to ensure continued progress in cases where valid configurations might be momentarily unattainable.

4. VELOCITY ANALYSIS & REDUNDANCY RESOLUTION:

4.1 Cartesian Velocity Comparison:

For velocity analysis, we derive the actual Cartesian velocity using the following equation.

$$V_{actual} = J_{actual} \cdot q(dot)_{actual}$$

To obtain the actual Jacobian matrix, we utilize the modified DH parameters specific to the Panda robot and employ the Python function '*compute_jacobian*' (*Panda_Jacobian.py*). This function yields the actual Jacobian matrix. We then calculate the derivative of the joint angles from the generated joint angle trajectory and subsequently compute the actual Cartesian velocity.

For comparative purposes, we calculate the predicted Cartesian velocity using the following equation.

$$V_{predicted} = J_{predicted} \cdot q(dot)_{predicted}$$

$$q(dot)_{predicted} = J^{-1}_{predicted} \cdot V_{actual}$$

To compute the predicted Jacobian, we use the joint angle configurations obtained after applying the inverse kinematics. We predict the DH parameters based on these configurations and calculate the predicted Jacobian. To compute $q(dot)_{predicted}$, we utilize the above equation. The anticipated Jacobian is expected to have a shape of 6x7. However, when calculating its inverse, we only take into account the initial 6 columns, altering its dimensions to 6x6. We do this to simplify the inverse computation by ensuring $q(dot)_7$ equals 0. This approach assists in addressing the redundancy issue in inverse velocity kinematics.

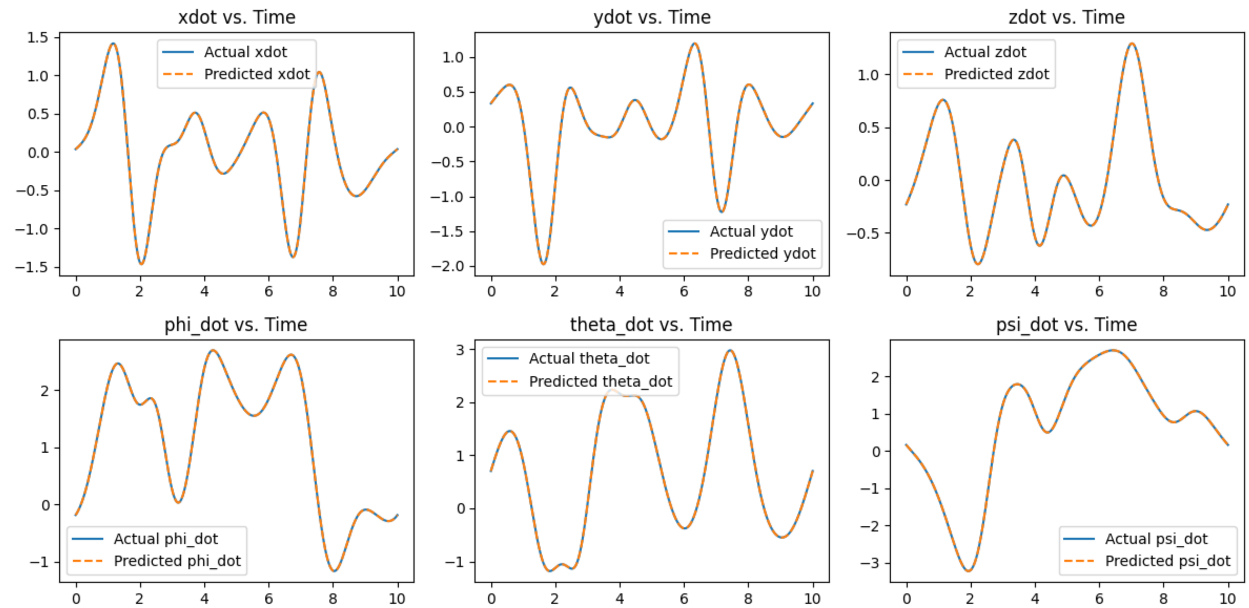

Figure 14: Comparison of Cartesian Velocity

In the final step, we compare the actual and predicted Cartesian velocities, as illustrated in Fig.14, focusing on trajectory 1. This remarkable congruence between the velocities provides robust validation for the precision and reliability of our Jacobian computation approach.

5. MANIPULABILITY ANALYSIS & STATIC ANALYSIS:

5.1 Manipulability Index:

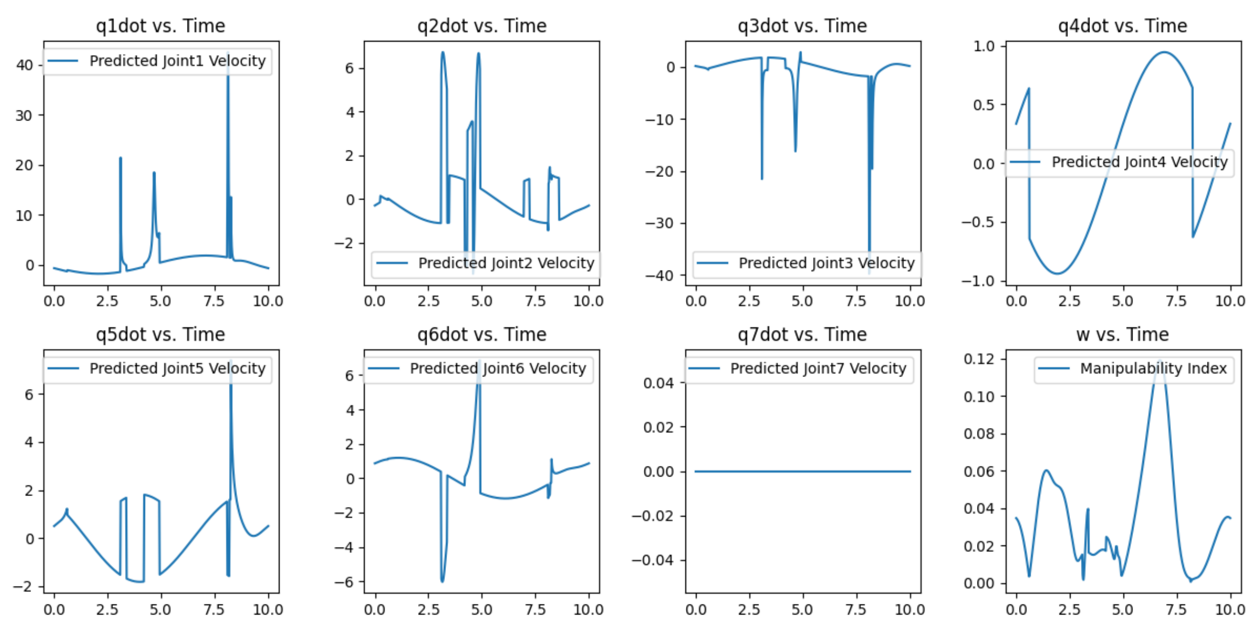

Figure 15: Joint velocity and manipulability index

To determine the manipulability index, we applied the following equation:

$$\omega = \sqrt{det(\mathfrak{J})} = \sqrt{J_{predicted} \cdot J^{T}_{predicted}}$$

In Fig.15, we have depicted the predicted joint velocity over time, alongside the Manipulability index graph. Notably, when there are peaks in the joint velocity graphs, signifying instances of singularity, the Manipulability index sharply drops to zero, which precisely confirms the expected behavior. These plots serve as a robust validation of our analysis.

5.2 Manipulability Ellipsoid & Force Ellipsoid:

As part of our singularity analysis, we calculated the singular values for both the Manipulability Ellipsoid and the Force Ellipsoid, followed by Singular Value Decomposition (SVD) analysis. Fig.16 and 17 display the patterns of singular values for both ellipsoids. These visualizations allow us to discern the directions of velocity and force where these quantities are maximized at specific points in time. This insight can be particularly valuable in the context of task-based trajectory planning and execution.

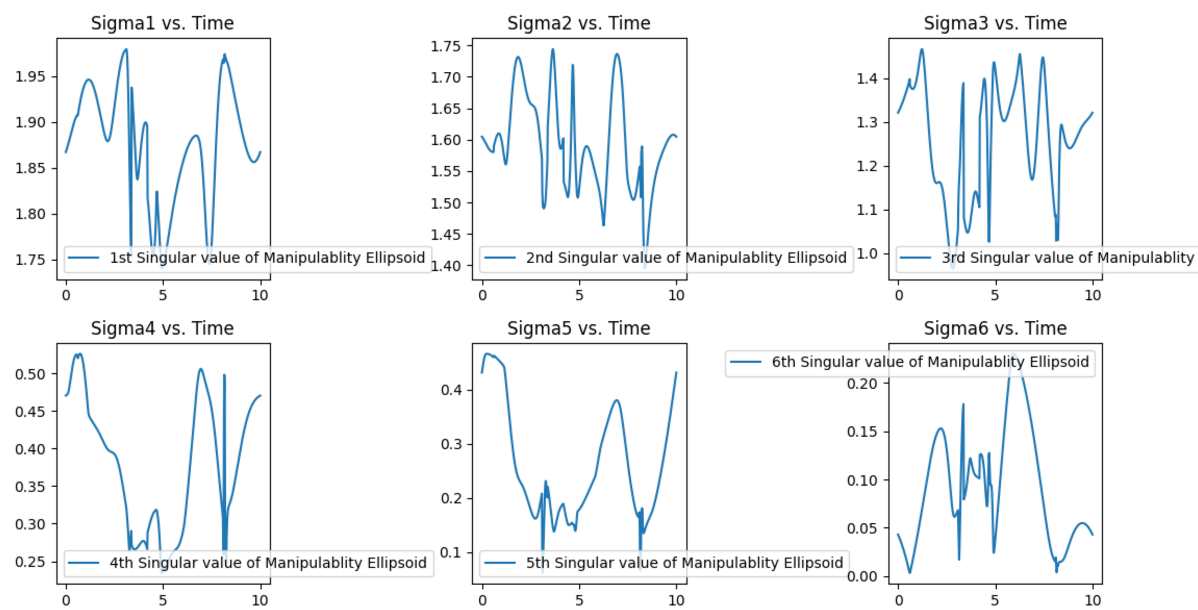

Figure 16: Singular values of manipulability Ellipsoid

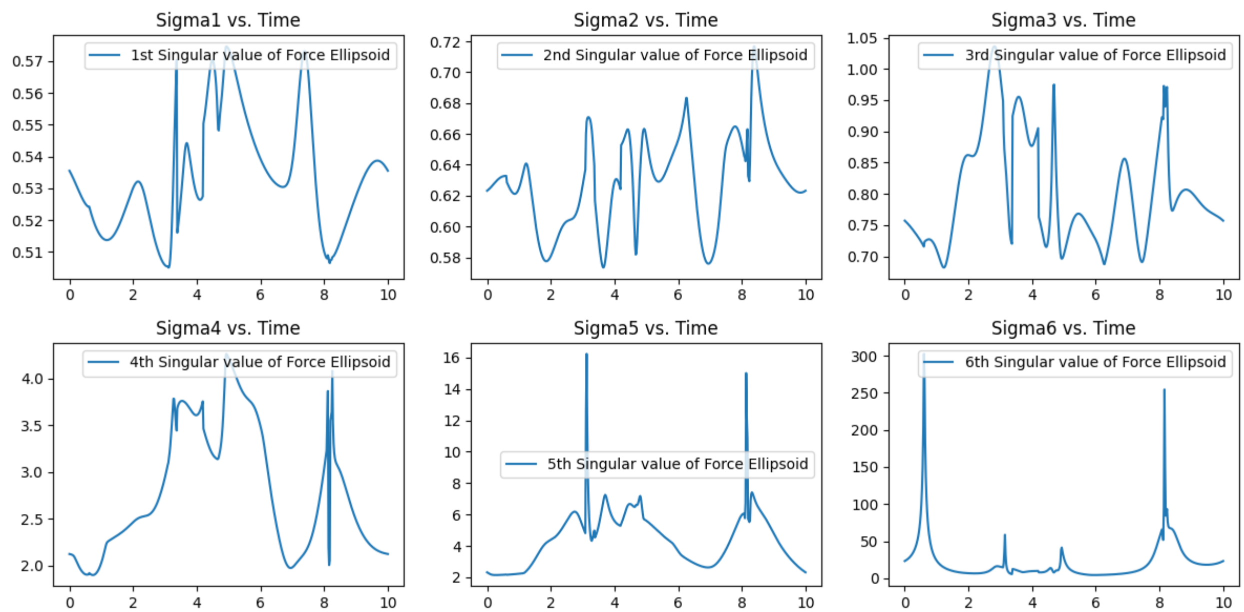

Figure 17: Singular values of force Ellipsoid

6. SIMULATION

For our simulation tasks, we chose to utilize CoppeliaSim due to its precise simulation models and user-friendly interface. We provided two CoppeliaSim simulation files:

- *CoppeliaSim_Panda_1.ttt:* In this simulation file, we subjected the Panda robot to a unique spiral + helical trajectory in which all the 6 degrees of freedom of the task space are used, allowing it to explore various configurations rather than staying within a single plane. For this simulation, we harnessed the built-in numerical solver provided by CoppeliaSim for inverse kinematics which was coded up in the lua script.

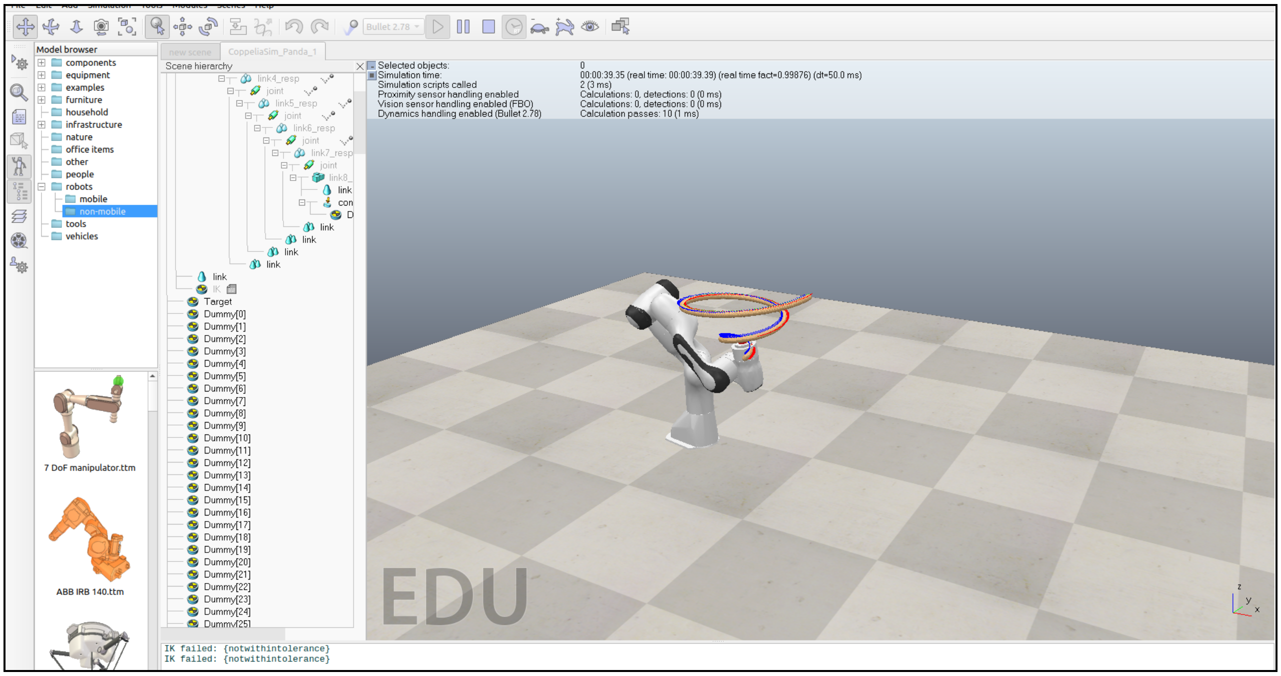

Figure 18: CoppeliaSim_Panda_1.ttt

- *CoppeliaSim_Panda_2.ttt:* Within this simulation file, we introduced two Panda robots. The robot on the left follows a trajectory determined by the joint angles computed using our Python inverse kinematics code, while the robot on the right adheres to a same trajectory generated by actual joint angle calculations. Interestingly, the trajectories of the two robots do not align. This discrepancy arises from the distinct DH parameters used in the CoppeliaSim model for the Panda robot, resulting in different final joint angles between the actual joint angle and the predicted joint angles from our Python code.

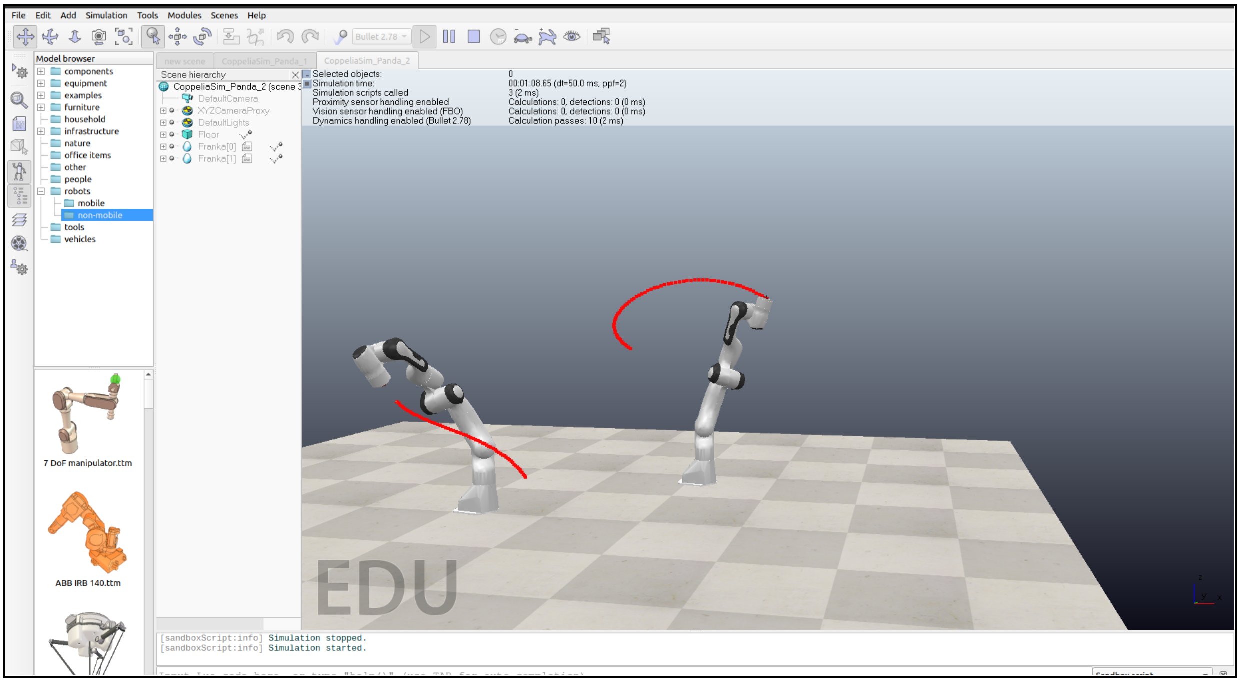

Figure 19: CoppeliaSim_Panda_2.ttt

REFERENCE

[1]https://www.researchgate.net/publication/357238256_Analytical_Inverse_Kinematics_for_Franka_Emika_Panda_-_a_Geometrical_Solver_for_7-DOF_Manipulators_with_Unconventional_Design

[2] https://wevolver-project-images.s3.amazonaws.com/0.xsiph4enopteaser.jpg

[3]https://www.diag.uniroma1.it/~deluca/rob1_en/WrittenExamsRob1/Robotics1_18.01.11.pdf

[4] https://frankaemika.github.io/docs/control_parameters.html

[5] https://github.com/Pradn1l/Rospy-FK-7Axis-Robot

[6] https://github.com/ffall007/franka_analytical_ik

[7] https://ieeexplore.ieee.org/abstract/document/10103352/references#references