Final Report

MAE 263A: Kinematics of Robotic Systems

Instructor:
Prof. Dennis Hong

Prepared by:

Chen, Po-Chih, Chou, Wei-Yi, Shih, Shuo-Wu, Trejo, Joshua, Yang, Jimmy

December 8, 2025

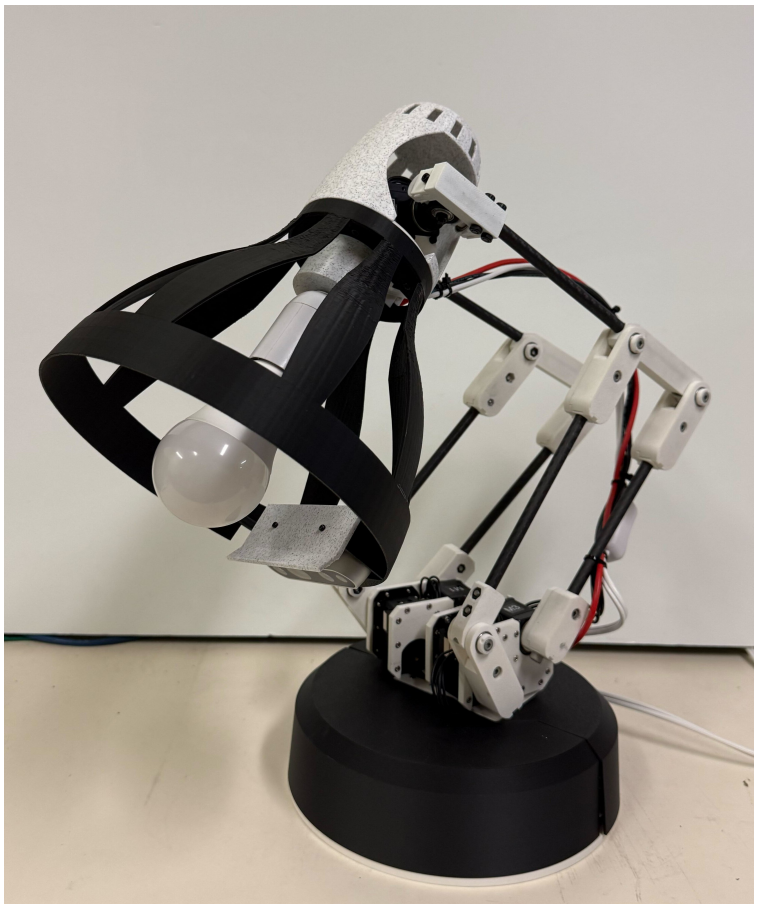


Figure 1: Luxo Jr.

Contents

1 Introduction and Design Overview

1.1 Background

Lighting adjustments in filmmaking are frequent and exacting, yet the standard process for making them remains inefficient. Typically, a technician must be stationed directly at the fixture, relying on verbal instructions to manually position the light source. This workflow is often prone to miscommunication and a long process of making adjustments. To resolve this, we propose a robotic lamp capable of tracking a user's hand position. By enabling direct, non-contact control, the system eliminates the need for an intermediary operator and allows for immediate, precise adjustments.

1.2 Design Overview

The lamp design is based on a robotic leg design created by Disney Robotics used in high-performance bipedal robots [1]. Sensor feedback from hand detection is used to determine the position and orientation of the lamp. The system 2 is as follows

- **Configuration:** The manipulator operates in 3D with **6 Degrees of Freedom (DoF)**.
- **Sensors and Actuators:** The system is powered by Dynamixel MX-28AR servo motors. The motors also provide position feedback. The system uses Intel Realsense 2 cameras for visual feedback for tracking.
- **Task:** We want to achieve real-time hand tracking using YOLO v4.

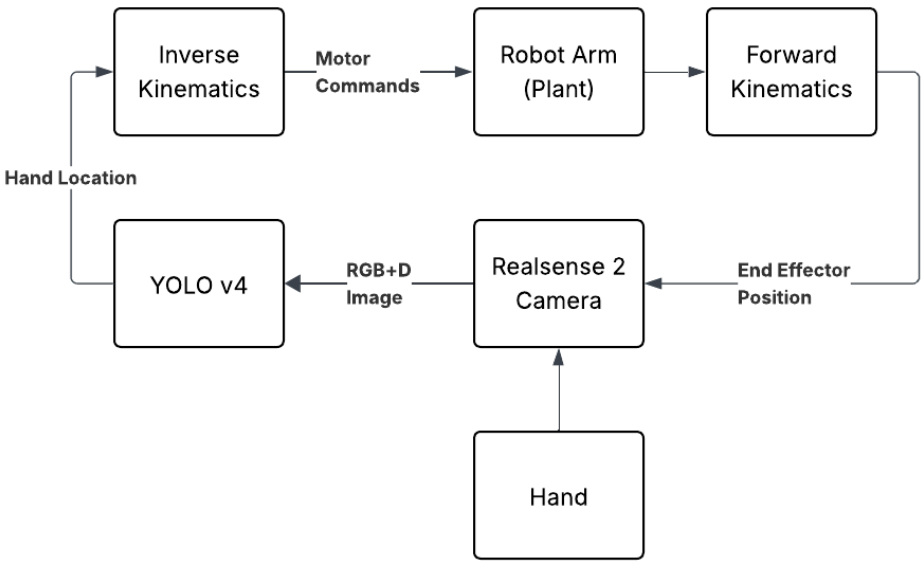


Figure 2: System overview; we assume the motor is a stable plant for our range of inputs and does not require external control to stabilize

2 Mechanical Design

2.1 Mechanism Selection: The Hybrid 5-Bar Linkage

We selected a serial-parallel hybrid mechanism rather than a standard serial arm. This design features a **5-bar linkage mechanism** for the primary pitch motions. In a standard serial arm, motors are placed at the joints, adding weight to the moving arm. In this hybrid design, the hip and knee pitch actuators are placed co-axially at the base. This significantly reduces the structural inertia of the moving parts. Lightweight materials such as PLA plastic and carbon fiber were also used to further reduce the inertia of the arm while maintaining rigidity. This ensures that the motors are able to drive the arm and allows for smoother, faster, and more stable movement of the end effector.

2.2 Electronics and Control Architecture

To achieve the intelligent tracking task, we integrated a high-level control stack:

- **Microcontroller:** A **Raspberry Pi** 5 was chosen as the primary microcontroller for interfacing with the sensors and actuators. **ROS2** was used as the underlying software framework for interfacing with the actuators, processing sensor data, and communication.
- **Sensing:** An **Intel RealSense 2** depth camera is used for 3D perception of the environment. It allows real time RGB and depth image feedback at a rate of 15 FPS.
- **Communication:** The Raspberry Pi communicates joint angle commands to the Dynamixel motors via the RS-485 protocol. The Raspberry Pi communicates with the Realsense 2 camera using UART serial communication via USB.

The `realsense2_camera` node obtains the RGB and depth images from the Realsense 2 Camera. The `predict_node` takes these images and performs hand location recognition using YOLO v4. The combination of the RGB image and depth image gives the 3D hand location. The 3D hand location is then sent to the `motor_node` where inverse kinematics is performed on a location close to the hand location to give the relevant motor commands. The motor commands are sent to `motor_read_write` node which interfaces with the motors.

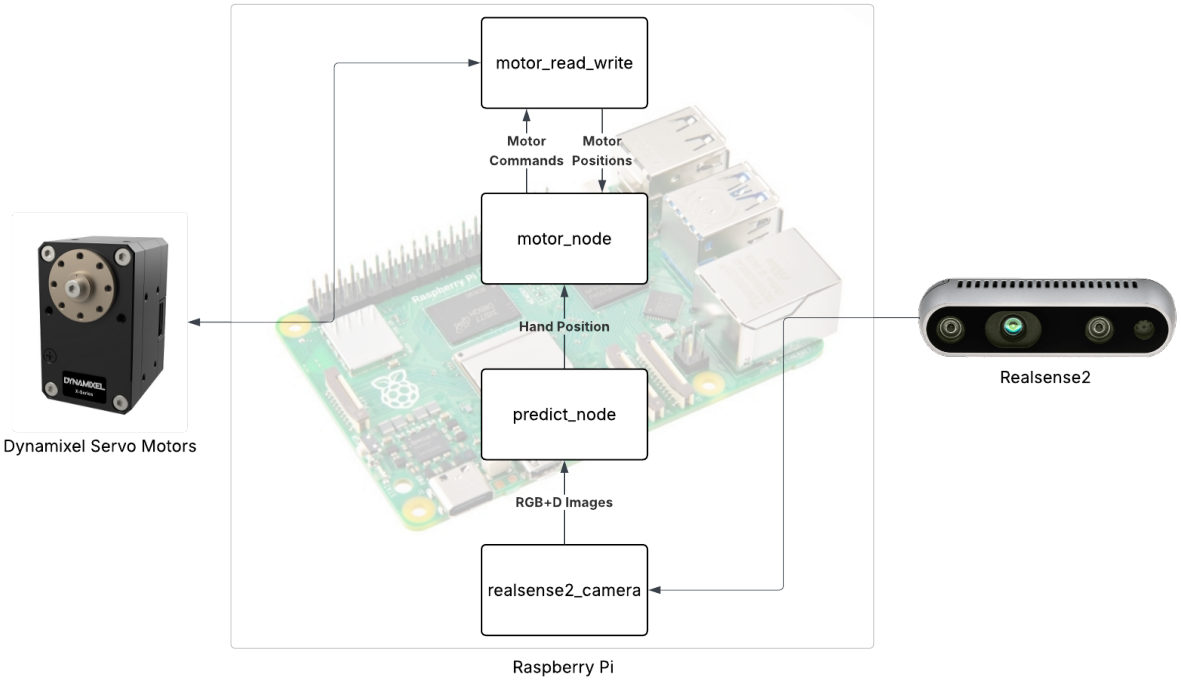


Figure 3: ROS 2 Structure

3 Kinematics

3.1 Forward Kinematics (FK)

Due to the complex structure of this closed-chain robot, the DH parameters required for forward kinematics cannot be directly given by motor angles. Instead, motor angles are converted to joint angles by evaluating geometry relations within the structure. The length of each link is defined in Table 4, and the derived DH parameters are in Table 5.

To solve forward kinematics, our strategy is to find the joint angles required for the DH parameters by solving the joint angles of closed-chain linkages. We first solve the the angles of the mechanism before rotation. Points before rotation carry subscript 0. The legged-mechanism can be separated into two same planar four-bar linkages. θ_1 and θ_2 can be told directly from the motors and solve θ_3 by finding the coordinates of each point.

$$P_{10} = \begin{bmatrix} l_1 c\theta_1 \\ l_1 s\theta_1 \\ l_{11} \end{bmatrix}, P_{20} = \begin{bmatrix} -l_4 + l_2 c\theta_2 \\ -l_3 + l_2 s\theta_2 \\ l_{11} \end{bmatrix}, \quad P_{30} = \begin{bmatrix} l_1 c\theta_1 + l_6 c\theta_3 \\ l_1 s\theta_1 + l_6 s\theta_3 \\ l_{11} \end{bmatrix}$$

$\overline{P_{20}P_{30}}$ should have constant length l_5: $\overline{P_{20}P_{30}} = l_5^2$. After some derivation, we can have this expression

$$K_1 c\theta_3 + K_2 s\theta_3 = K_3$$

l_1	200	l_2	50	l_3	40
l_4	25	l_5	217	l_6	80
l_7	245	l_8	50	l_9	73
l_{11}	31.5	l_{12}	20	l_{14}	35

Figure 4: Link lengths

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	l_4	θ_1
2	$-90°$	0	l_{11}	θ_2
3	0	l_1	0	θ_3
4	$90°$	l_7	0	$-90°$
5	$90°$	l_{12}	0	θ_5
6	$-90°$	0	0	θ_6
7	$-90°$	0	0	θ_7
8	$-90°$	0	l_{13}	θ_7
9	$-90°$	l_{14}	0	θ_7

Figure 5: DH Table

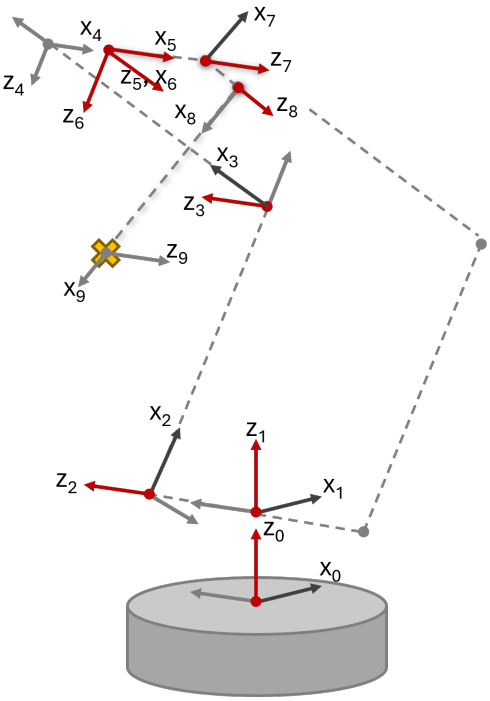
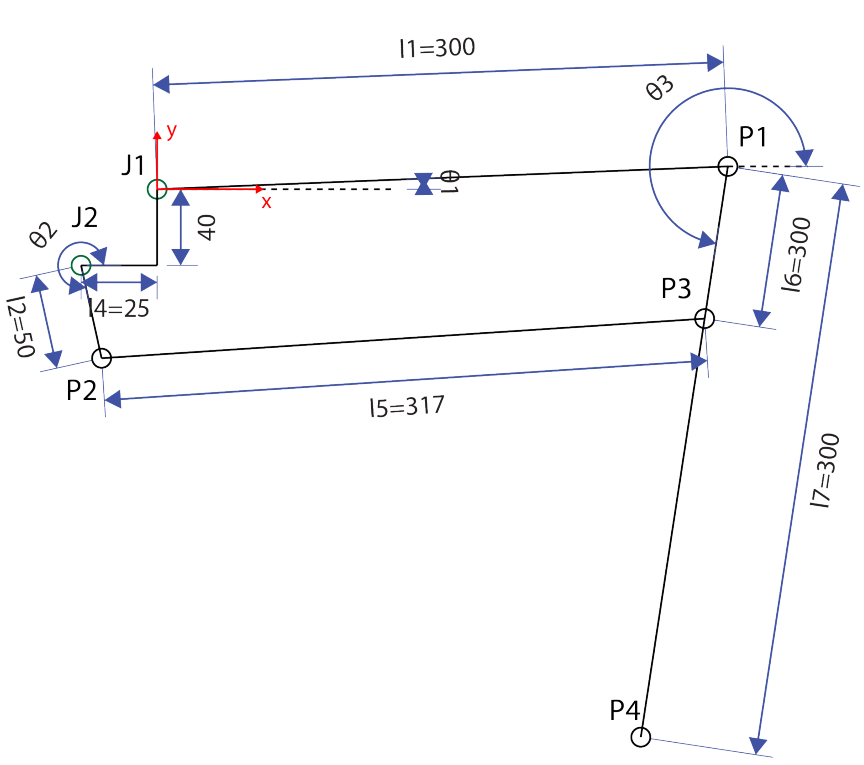
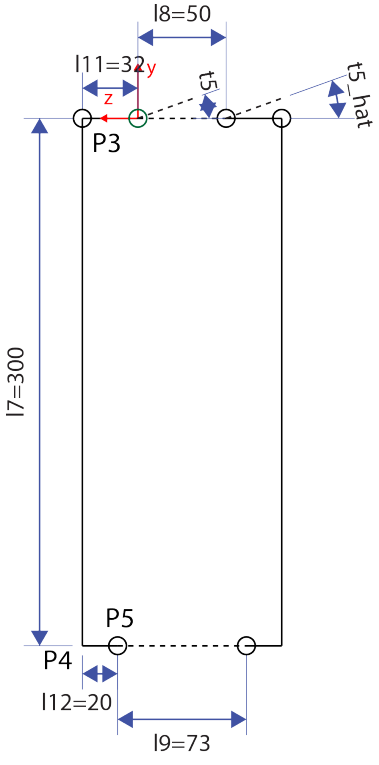


Figure 6: Skeleton diagram



(a) Planar mechanism skeleton (unit: mm)



(b) Skeleton diagram from top

Figure 7: Robot dimensions

where

$$K_1 = l_1 c\theta_1 + l_4 - l_2 c\theta_2 + l_6 c\theta_3$$
$$K_2 = l_1 s\theta_1 + l_3 - l_2 s\theta_2 + l_6 s\theta_3$$
$$K_3 = \frac{l_9^2 - K_1^2 - K_2^2 - l_6^2}{2l_6}.$$

Then, we can get $\theta_3 = atan2\left(-\sqrt{K_1^2 + K_2^2 - K_3^2}, K_3\right) + atan2\left(K_2, K_1\right)$. The second mechanism can be solved similarly. Once we have θ_3, we can get the coordinate of P_4 as

$$P_{40} = \begin{bmatrix} l_1 c\theta_1 + l_7 c\theta_3 \\ l_1 s\theta_1 + l_7 s\theta_3 \\ l_{11} \end{bmatrix}, \quad P_{50} = \begin{bmatrix} l_1 c\theta_1 + l_7 c\theta_3 \\ l_1 s\theta_1 + l_7 s\theta_3 \\ l_{11} - l_{12} \end{bmatrix} = \begin{bmatrix} x_{50} \\ y_{50} \\ z_{50} \end{bmatrix}, \quad \widehat{P_{50}} = \begin{bmatrix} l_1 c\widehat{\theta_1} + l_7 c\widehat{\theta_3} \\ l_1 s\widehat{\theta_1} + l_7 s\widehat{\theta_3} \\ -l_{11} + l_{12} \end{bmatrix} = \begin{bmatrix} \widehat{x_{50}} \\ \widehat{y_{50}} \\ \widehat{z_{50}} \end{bmatrix}$$

The other planar mechanism has the same design and the corresponding points are label with hat ($\widehat{\ }$). Both planar mechanisms are connected to a motor or a revolute joint. For ease of analysis, local coordinate rotation for each planar mechanism is considered separately. After rotation , the suffix 0 is removed, and P_{50} and $\widehat{P_{50}}$ become P_5 and $\widehat{P_5}$

$$P_5 = \begin{bmatrix} x_{50} \\ y_{50} c\theta_6 - z_{50} s\theta_6 \\ y_{50} s\theta_6 + z_{50} c\theta_6 \end{bmatrix} = \begin{bmatrix} x_5 \\ y_5 \\ z_5 \end{bmatrix}, \quad \widehat{P_5} = \begin{bmatrix} x_{50} \\ y_{50} c\theta_6 - z_{50} s\theta_6 \\ y_{50} s\theta_6 + z_{50} c\theta_6 \end{bmatrix}$$

As $\overline{P_5\widehat{P_5}}$ is a constant and $\overline{P_5\widehat{P_5}}^2 = l_9^2$. After simplification, we can get the following coefficient

$$T_1 = 2y_5\widehat{y_{50}} + 2(z_5 + l_8)\widehat{z_{50}}$$
$$T_2 = 2y_5\widehat{z_{50}} + 2(z_5 + l_8)\widehat{y_{50}}$$
$$T_3 = (x_5 - \widehat{x_{50}})^2 + y_5^2 + (z_5 + l_8)^2 + \widehat{y_{50}}^2 + \widehat{z_{50}}^2 - l_9^2.$$

Hence, $\widehat{\theta_5} = atan2\left(-\sqrt{T_1^2 + T_2^2 - T_3^2}, T_3\right) + atan2\left(T_2, T_1\right)$, and thus we can know the coordinates of both P_5 and $\widehat{P_5}$. As P_3, P_4, P_5, and $\widehat{P_5}$ are available, $\overrightarrow{P_4P_3}$, $\overrightarrow{P_4P_5}$, and $\overrightarrow{P_5\widehat{P_5}}$ can be used to find θ_6 and θ_7. Define normalized $\overrightarrow{P_4P_3}$, $\overrightarrow{P_4P_5}$, and $\overrightarrow{P_5\widehat{P_5}}$ as $\overrightarrow{e_x}$, $\overrightarrow{e_y}$, $\overrightarrow{v}$, and derive $\overrightarrow{e_z} = \overrightarrow{e_x} \times \overrightarrow{e_y}$. Form the rotation matrix as $R = \begin{bmatrix} \overrightarrow{e_x} & \overrightarrow{e_y} & \overrightarrow{e_z} \end{bmatrix}$. In this coordinate, rotate $\overrightarrow{e_x}$ around $\overrightarrow{e_y}$ then $\overrightarrow{e_z}$ θ_6 and θ_7 gets $\overrightarrow{v}$. Therefore,

$$\overrightarrow{v_0} = R^T \cdot \overrightarrow{v} = R_z(\theta_7) R_y(\theta_7) \overrightarrow{e_x}$$
$$\theta_6 = atan2\left(-v_{0x}, \sqrt{1 - v_{0x}^2}\right)$$
$$\theta_7 = -atan2\left(\frac{v_{0y}}{\cos\theta_6}, \frac{v_{0x}}{\cos\theta_6}\right)$$

and thus we can construct the DH table as Table 5

3.2 Inverse Kinematics (IK)

Due to the complex, coupled nature of the serial-parallel hybrid mechanism, a closed-form analytic solution for the inverse kinematics does not exist. Consequently, we implemented a numerical optimization approach to resolve the joint angles required to reach a specific target end effector coordinate (x_d, y_d, z_d).

We utilized the **Levenberg-Marquardt (LM)** algorithm, a standard iterative method used for solving non-linear least squares problems. This algorithm interpolates between the Gauss-Newton algorithm and the method of gradient descent, making it robust for finding local minima. The solver iteratively adjusts the joint angles θ to minimize the Euclidean distance error between the target position P_{target} and the current end-effector position calculated via Forward Kinematics $FK(\theta)$:

$$\min_{\theta} ||P_{target} - FK(\theta)||^2 \tag{1}$$

Algorithm 1 Levenberg-Marquardt Inverse Kinematics Solver

Require:
Target Position $\mathbf{p}_{target}$
Previous Joint Angles $\boldsymbol{\theta}_{t-1}$
Ensure: Optimal Joint Angles $\boldsymbol{\theta}_t$

$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}_{t-1}$ ▷ Initialization
$\lambda \leftarrow 0.01$ ▷ Initial damping factor
$k \leftarrow 0$
while $k <$ MAX_ITER **and** Error $> \epsilon$ **do**
  $\mathbf{p}_{current} \leftarrow FK(\boldsymbol{\theta})$ ▷ Forward Kinematics
  $\mathbf{e} \leftarrow \mathbf{p}_{target} - \mathbf{p}_{current}$ ▷ Compute error vector
  if $||\mathbf{e}|| < \epsilon$ **then**
    break ▷ Convergence reached
  end if
  $\mathbf{J} \leftarrow$ ComputeJacobian($\boldsymbol{\theta}$)
  $\Delta\boldsymbol{\theta} \leftarrow (\mathbf{J}^T\mathbf{J} + \lambda\mathbf{I})^{-1}\mathbf{J}^T\mathbf{e}$ ▷ Damped Least Squares
  $\boldsymbol{\theta}_{new} \leftarrow \boldsymbol{\theta} + \Delta\boldsymbol{\theta}$
  $\mathbf{e}_{new} \leftarrow \mathbf{p}_{target} - FK(\boldsymbol{\theta}_{new})$
  if $||\mathbf{e}_{new}|| < ||\mathbf{e}||$ **then**
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}_{new}$ ▷ Accept new angles
    $\lambda \leftarrow \lambda/10$ ▷ Reduce damping (closer to Gauss-Newton)
  else
    $\lambda \leftarrow \lambda \times 10$ ▷ Increase damping (closer to Gradient Descent)
  end if
  $k \leftarrow k + 1$
end while
return $\boldsymbol{\theta}$

Initialization and Convergence: Numerical solvers depend heavily on the quality of the initial guess to converge quickly and avoid local minima (such as mechanically invalid singularities). To optimize for real-time tracking, we exploited the temporal coherence of the movement. The joint angles solution from the previous time step (θ_{t-1}) is used as the initial guess for the optimization at the current time step (θ_t). This strategy significantly reduces the computational load and ensures the manipulator motion remains smooth and continuous without erratic jumps between kinematic solutions.

4 Simulation and Implementation

4.1 Task Simulation

We validated our kinematics using **MATLAB**. The simulation plotted the effective workspace and verified that the maximum calculated velocities (e.g., lateral velocity $V_y \approx 816$ mm/s) were within the safe operating limits of the MX-28AR motors [1]. The simulation is shown in the figure below.

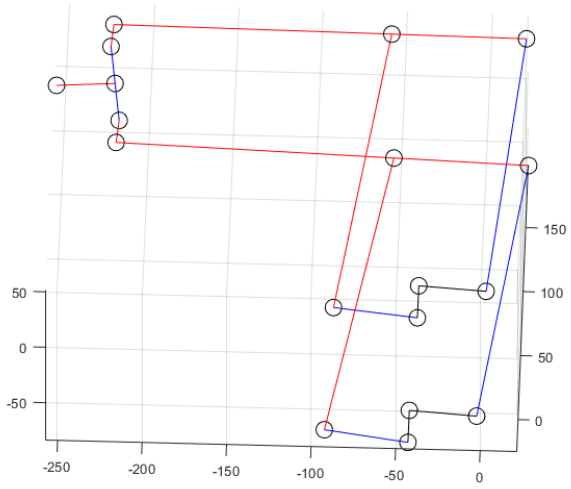


Figure 8: Motion Simulation with Matlab

4.2 Sensing and Feedback Control

We attempted to implement a tracking control loop using computer vision:

1. **Input:** The RealSense 2 camera captures RGB-D (Color + Depth) video.
2. **Processing:** A pretrained **YOLOv4** neural network runs on the Raspberry Pi to detect the bounding box of the user's hand.
3. **Targeting:** The center of the bounding box is mapped to a 3D coordinate $(x_{target}, y_{target}, z_{target})$ using the depth stream. Coordinate transformations are used to change the coordinates from pixel coordinates to end effector frame coordinates.
4. **Control:** The ROS kinematics node continuously updates the Goal Position of the IK solver to match $(x_{target} - 10, y_{target}, z_{target})$, ensuring the lamp hovers near the hand.

4.3 Hand Detection

To find the position of the detected object (the hand) in the camera frame, we used a YOLO v4 model to detect the hand and find the pixel coordination $((p_x, p_y))$ in the RGB image. The pixel coordination was

then used in the depth image to extract the depth (d). The position can be found with the equation below in the robot frame

$$x = d$$
$$y = -p_x \times d/f_x$$
$$z = -p_y \times d/f_y$$

where f_x and f_y represent focal length in x and y direction.

(a) RGB image (b) Depth image

Figure 9: RealSense camera images

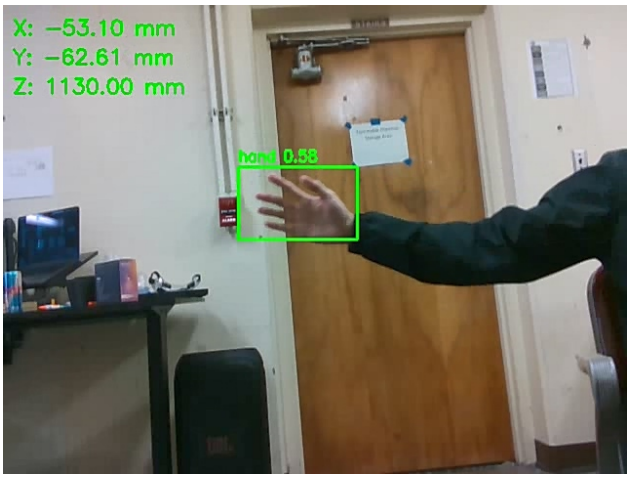


Figure 10: Hand detection in bounding box

5 Conclusion

We successfully designed, simulated, and built a 6-DoF robotic manipulator based on a serial-parallel hybrid mechanism. The design leverages the low inertia of the 5-bar linkage to create a stable and responsive robotic

lamp. The integration of depth sensing with the RealSense 2 camera and hand detection with YOLO v4 opens up the possibility of tracking control of the robotic arm from external sensor measurements.

6 Future Work

Currently due to motor torque limitations the lamp head cannot be held in the upright position as intended and can only face downwards. Therefore it is not possible to perform hand tracking. In the future we would replace the base motors with higher torque motors to stand the lamp upright so that it can face the hand it is meant to track.

References

[1] Kevin Genehyub Gim, Joohyung Kim, and Katsu Yamane. Design and fabrication of a bipedal robot using serial-parallel hybrid leg mechanism. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5095–5100. IEEE, 2018.