Control of 3DoF Robotic Manipulator for Lego Block Assembly (Team #5)

Agrawal, Ayush
Mechanical Engineering
University of California Los Angeles
Los Angeles, CA, USA
ayushagrawal26@ucla.edu

Chakrabarti, Debjyoti
Mechanical Engineering
University of California Los Angeles
Los Angeles, CA, USA
debjyoti@ucla.edu

Sivakumar, Premkumar
Mechanical Engineering
University of California Los Angeles
Los Angeles, CA, USA
premkumar22@ucla.edu

***Abstract*—In this work, we have designed and developed a 3-DOF robotic manipulator for the task of Lego block assembly. We have developed an inverse dynamics controller that ensures the smooth operation of picking up the Lego blocks from an initial position, bringing them to a final position, and subsequently press-fitting the block on the Lego bed, using a magnet-fitted rack-and-pinion mechanism. We show the performance of our designed controller in smoothly following a reference trajectory. We also compare the performance of the proposed controller for various sets of gains for the same task. The present study only focuses on joint-space inverse dynamics control, which considers the interaction forces from the end-effector as a disturbance. The same application with force control is left for the future.**

Keywords—LEGO assembly, inverse dynamics control, magnetic actuation, centralized control

I. Overview

Whenever a robotic manipulator interacts with its environment, the contact force at the end-effector becomes a critical factor in determining the outcome of that interaction. For instance, the difference between safely placing a grape in a basket and crushing it lies in how precisely the applied force is controlled.

In our case, to limit complexity, the manipulator is designed to interact with rigid Lego blocks — objects that are not easily deformed or crushed. However, the challenge arises during the press-fit operation, where one block must be firmly inserted onto another. This interaction requires precise force control at the end-effector, providing the key motivation for this project.

Picking and placing objects using robotic manipulators is predominantly a position control problem. However, tasks like Lego stacking introduce an additional layer of complexity: **force-sensitive insertion**. To accomplish this without access to an end-effector force sensor, we are motivated to explore **sensor-less impedance control** — a method that uses internal signals to infer interaction forces. This makes the problem both technically rich and practically constrained, offering a realistic scenario.

With the rise of **modular construction techniques**, rapid and repeatable assembly and disassembly of structures has become increasingly important. A large-scale version of this 3-DOF robotic system could be adapted for such applications, automating the assembly process and significantly reducing construction time. This project serves as a scaled-down prototype for investigating such future possibilities.

While there are multiple relevant works in academia and industry on implementing force and position control, the works of Tang et al. [1] and D.H.Lee et al. [2] are some works relevant to our problem statement and have been our source of inspiration. [1] has implemented a 4 DOF (4R) robot with an eye-in-hand setup for the Lego assembly. This robot uses a dexterous hand as an end-effector for grasping and uses high-precision force feedback to press the Lego blocks through vision-based tactile sensing. On the other hand, [2] implemented a robotic peg-in-hole which is essentially similar to the problem statement at hand due to the requirement of precise positioning of the peg and pressing it into the hole. The implementation uses a dual dexterous arm for manipulation and grasping with an eye-to-hand setup. They have demonstrated a feed-forward task space force control scheme and also demonstrated a hybrid force-position control scheme.

II. Objectives of the Control Design

The primary objective of this project is to develop a control scheme for a robotic arm to autonomously assemble a Lego structure. The system will operate without any external force sensors, relying solely on internal measurements, such as joint positions obtained from motor encoders, to estimate interaction forces (if needed) and apply them to achieve the desired performance. This project aims to demonstrate both the potential and limitations of motion control in a purely contact-driven task like Lego assembly.
The objectives of this project can be summarized as,

- Designing and tuning of the motion controller to assemble Lego blocks using a 3DoF manipulator
- Achieving proper alignment and insertion of LEGO bricks without excessive force.
- Evaluating task performance through metrics such as RMS error in states.

III. METHODS

A. Visualization of robot design

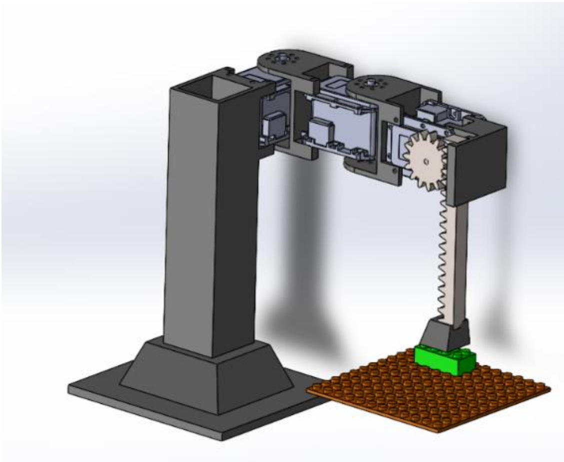


Figure 1: CAD Model of 2R-1P Manipulator for Lego block fitting

In our problem statement, we focus on press-fitting a Lego block vertically on a Lego plate. Clearly, we need mobility along all three Cartesian axes, $x-y-z$, indicating requirements for at least 3 actuators. Since we are dealing with picking and placing full-dimensional rigid bodies, the spatial orientation of the object becomes crucial while performing the press-fitting operation. However, we simplify both hardware and control design by,

- Strategically placing the Lego block so that its yaw (z) orientation aligns with end-effector orientation during the picking maneuver
- Fixing the Lego block in the robot's reachable workspace so that the orientation of the end-effector and Lego block aligns with the Lego bed's orientation at the drop location.

By doing so, we omit the need for 4^{th} joint, which may have controlled the yaw (z) orientation of the end effector. As a result, our controller only commands three joints to reach the desired $x-y-z$ rotation. Omission of 4^{th} joint limits the overall inertia of the robot, ultimately keeping the required torque magnitudes well within the limits. The above arguments helped us converge on the robot design shown in Figure 1.

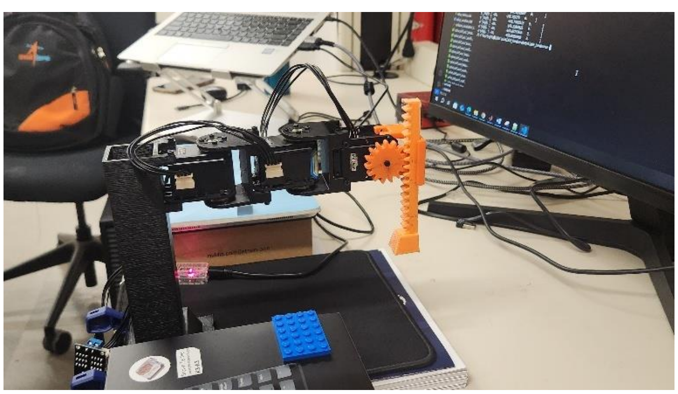


Figure 2: Final hardware design of the robot and layout of the planned workspace

Note: The above design was achieved in the second iteration. Details of changes made in the robot design are provided in the appendix.

B. *DH Parameters and Kinematics*

For our 3-DOF robot, the following is the modified-DH parameter Table 1,

Table 1: Modified DH parameters

$i-1$	a_{i-1}	α_{i-1}	d_i	θ_i
0	0	0	d_1	θ_1
1	a_1	0	0	$\theta_{]2}$
2	a_2	0	$-d_2$	0

Here, a_1, a_2 are distances between motor1-motor 2 and motor2-motor3 pair respectively. The height of the links from the base is d_1. Parameters d_2, θ_1, θ_2 are variables for prismatic and revolute joints.

The corresponding forward kinematics is given by,

$$T = \begin{bmatrix} \cos(\theta_1+\theta_2) & -\sin(\theta_1+\theta_2) & 0 & a_2\cos(\theta_1+\theta_2)+a_1\cos(\theta_1) \\ \sin(\theta_1+\theta_2) & \cos(\theta_1+\theta_2) & 0 & a_2\sin(\theta_1+\theta_2)+a_1\sin(\theta_1) \\ 0 & 0 & 1 & d_1-d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

To solve the joint variables, the inverse kinematics equations are as follows, say, $R = T_{1:3,1:3}$ (rotation matrix) and $P = T_{1:3,4}$ be the position part of the transformation matrix 'T'. Then it follows,

$$\begin{aligned} \theta_{12} &= 2(R_{21}, R_{22}) \\ d_2 &= P_3 - d_1 \\ \\ s_{12} &= R_{21} \\ c_{12} &= R_{22} \\ \\ \cos(\theta_1) &= \frac{P_1 - a_2 c_{12}}{a_1} \\ \sin(\theta_1) &= \frac{P_2 - a_2 s_{12}}{a_1} \\ \\ \theta_1 &= 2(\sin(\theta_1), \cos(\theta_1)) \\ \theta_2 &= \theta_{12} - \theta_1 \end{aligned} \tag{2}$$

We use inverse kinematics to convert the via-point descriptions to joint space for trajectory planning. The details of the trajectory planning are described in the appendix.

Table 2: Objective metrics for control design

Design requirements	Objective metric
Free-space pose accuracy	• RMS joint position error $\leq 2°$ • RMS joint velocity error $\leq 5°/sec$
Damping ratio	Tuned gains must lead to an underdamped response, $\zeta_i < 1$, ensuring low rise time.
Hardware limits	Control torques must not exceed motor torque limits, i.e., $u_i < 0.5\ Nm\ \forall\ i \in \{0,1,2\}$

C. *Control Design Requirements*

For assembly/stacking of Lego blocks, the robot must reach the specified location where the block needs to be placed. Hence, a highly accurate trajectory tracking control must be designed for the robot. To this end, we choose Joint Space Inverse Dynamics Control to accomplish the task. Besides a stable design, the controller must be tuned so that joints track the reference trajectory (desired position (θ_d), desired velocity ($\dot{\theta}_d$)) to meet the accuracy metrics while obeying the hardware limits as listed in the Table 2.

D. *Assumptions*

We take the following assumption to simplify the mathematical model of robot dynamics, which further simplifies the control design process for executing the above task.

1. Low speed/acceleration trajectory assumption:
   The reference trajectory is designed to ensure that the joint accelerations remain small. Consequently, the Coriolis and Centrifugal terms, denoted $C(q, \dot{q})$, are considered negligible and are omitted for the control law.

2. Simplification of gravity compensation:
   The robotic structure is mechanically designed to support its own gravitational load, with the exception of the rack-and-pinion mechanism at the final joint.
   a. Due to this design, only the last joint is responsible for compensating for the gravitational effects of the rack-and-pinion assembly.
   b. The corresponding gravity compensation vector is therefore given by:
   $$g_{comp} = [0, 0, m_e g r_{pinion}]$$

where, m_3 is the mass of the rack and pinion gear.

E. *Joint space inverse dynamics (JSID) controller*

The maneuvers for the robots are planned in joint space. JSID controller very well solves the problem of tracking trajectory in joint space. We begin the control design with the dynamic model of an n-joint manipulator.

$$\tau = M(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q) + F(q)\dot{q} \tag{3}$$

where, $M(q)$ is the manipulator inertia matrix, $C(q, \dot{q})$ is the matrix of Christoffel symbols representing the Coriolis and centrifugal terms, $g(q)$ represents gravity load, and $F(q)$ represents the viscous friction in motor. With assumption C.1, C.2, the dynamics simplify to

$$\tau = M(q)\ddot{q} + \begin{bmatrix} 0 \\ 0 \\ m_3 g r_{pinion} \end{bmatrix} \tag{4}$$

Taking the control τ as a function of the manipulator state in the form

$$\tau = M(q)y + \begin{bmatrix} 0 \\ 0 \\ m_3 g r_{pinion} \end{bmatrix} \tag{5}$$

leads to the overall system described by

$$\ddot{q} = y \tag{6}$$

where, y is the outer loop control vector. Choosing y as the following stabilizing control vector

$$y = \ddot{q}_d + K_d(\dot{q}_d - \dot{q}) + K_p(q_d - q) \tag{7}$$

where, K_d, K_p are appropriately chose positive definite matrices. With this, our final control law becomes

$$\tau = M(q)\left[\ddot{q}_d + K_d(\dot{q}_d - \ddot{q}) + K_p(q_d - q)\right] + \begin{bmatrix} 0 \\ 0 \\ m_3 g r_{pinion} \end{bmatrix} \tag{8}$$

We implement the same in our code for the JSID control, see [4].

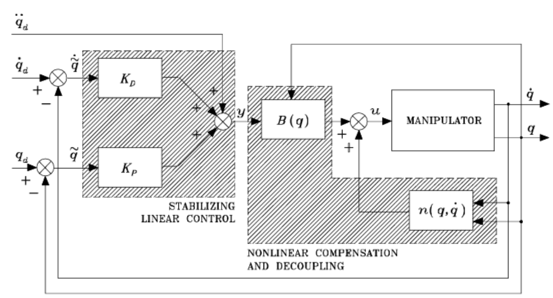


Figure 3: Block Diagram of Joint Space Inverse Dynamics Control [3]

IV. RESULTS

A. *Snapshots of hardware demonstration*

The outcome of above development can be visualized in the following six snapshots. The table below describes the state of motion at each time stamp.

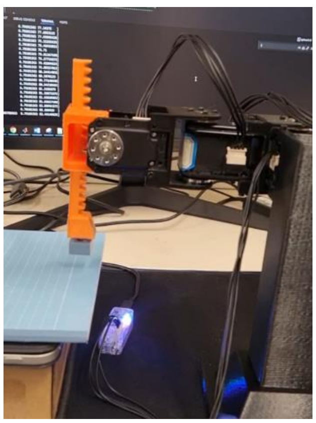
Table 3: Description of motion states at different time stamps

Time Stamp	State of Motion
T = 0 sec	Robot at homing position
T = 10sec	Joint 1 turns by +90° to reach the (x, y) coordinate of Lego Block
T = 20 sec	Joint 3 extends downwards to pick the Lego

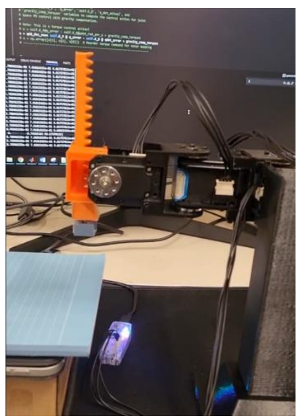
T = 30 sec	Joint 3 retracts upwards
T = 40 sec	Joint 1 & 2 rotate by $-90°, -180°$ to reach (x, y) coordinate of drop position
T = 50 sec	Joint 3 presses the Lego block in the Lego bed successfully

Pose at T = 0

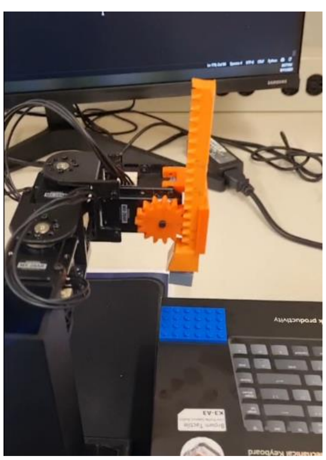
Pose at T = 10 sec



Pose at T = 20 sec



Pose at T = 30 sec



Pose at T = 40 sec

Pose at T = 50 sec

B. Error / Performance assessment

We assess the performance of our controller based on the criteria defined in Table 2, row 1. For our initial choice of gains in Table 4 below, we computed the RMS error values as reported below. The reported metrics do not fall within the defined error bounds.

Potential explanation behind violation of metrics:

Under the assumption of low-speed trajectory, we dropped complex terms in the equation of motion of manipulator. Besides $C(q, \dot{q})$, we also dropped the viscous friction terms, i.e., $F(q)$. This term in coalition with K_d increase the sluggishness of the system response, hence increasing undershoot. In the first 10 seconds, joint 1 moves from 0° to 90°. With friction providing additional damping, the motor struggles to reach the angle, hence we the error signal build up to reach 10°, see figure 4. Potential solution to this issue is increased K_p.

Table 4: RMS position error for initial set of gains

Joint #	#1	#2	#3
K_p	120	100	75
K_d	0.2	0.2	3
RMS Error	3.2047°	2.9423°	0.0035 m

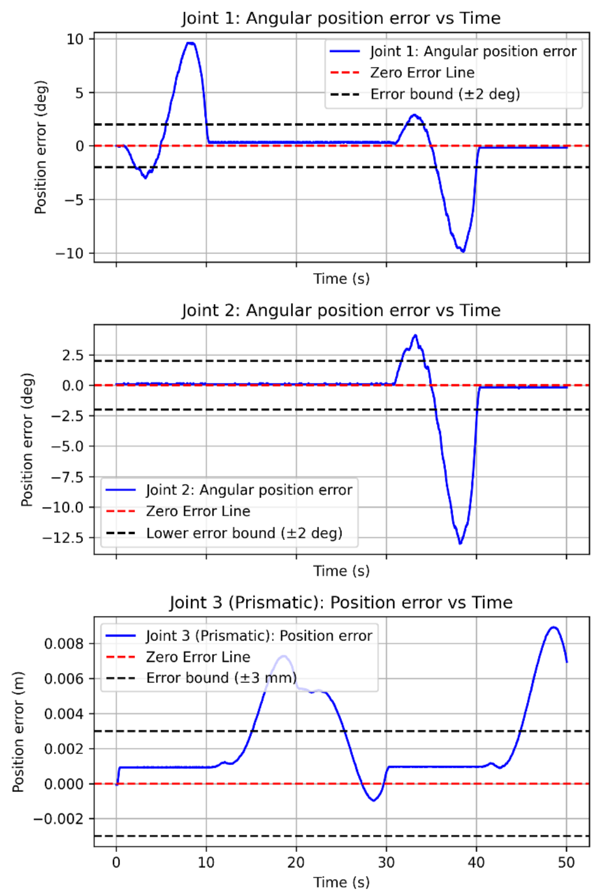


Figure 4: Position error in joints while trajectory tracking

C. *Time history of system variables*

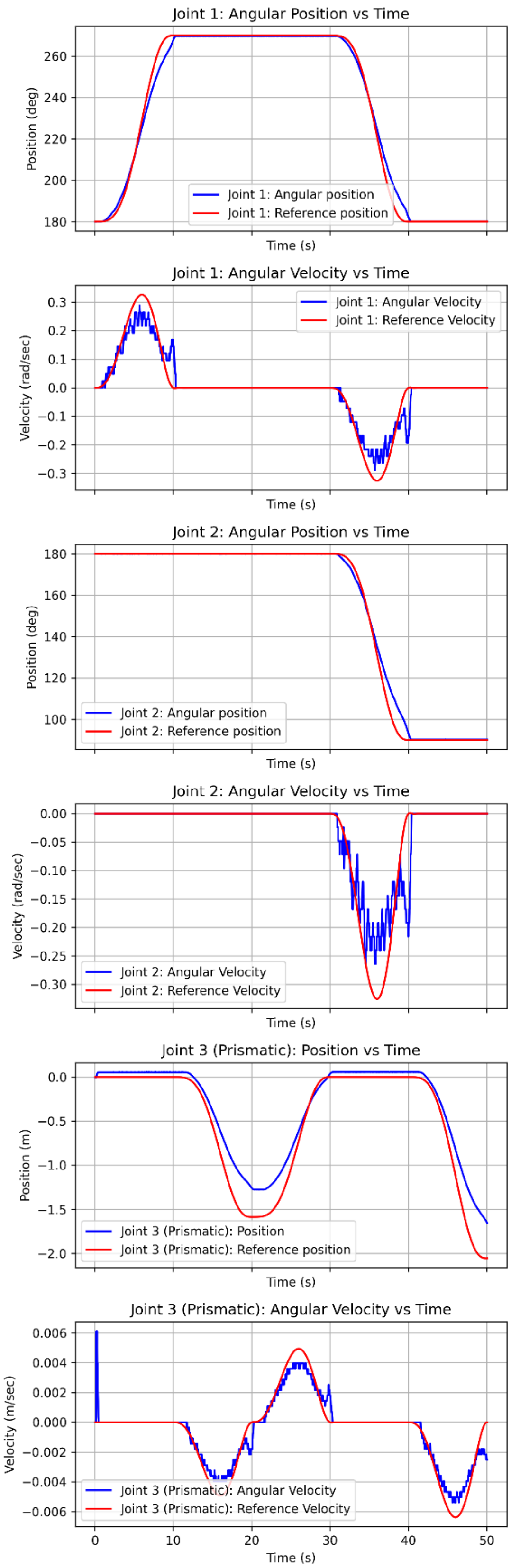


Figure 5: Time history of manipulator state variables

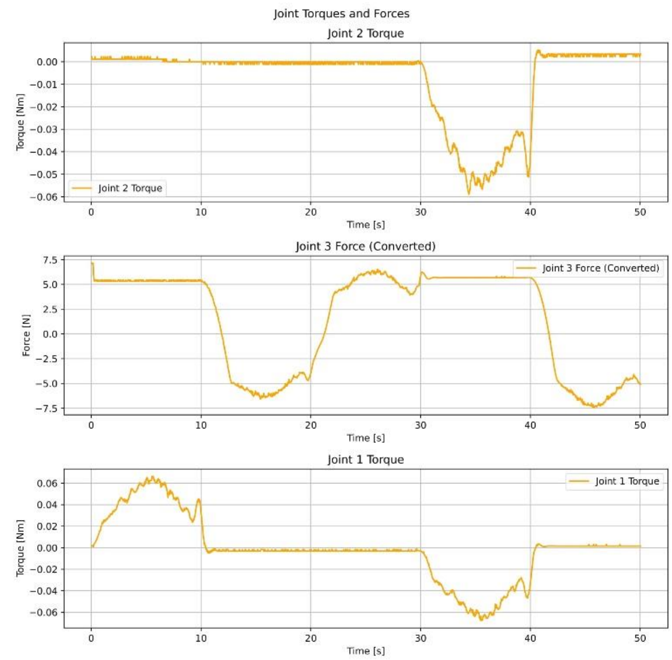


Figure 6: Time history of computed control variables

Figure 5 shows the evolution of system states with time. Visible differences between the actual and reference trajectory clearly depicts the error caused by the simplifying assumption. Besides this, figure 6 shows the control torque profile for each joint. It is apparent that the torque values never exceed the motor torque limits, i.e., 0.5 Nm. To justify the same for joint 3, we multiply the max force by gear radius, i.e., 0.015m. With max torque = $7.5 \times 0.015 = 0.1125\ Nm$. Hence, our implemented strategy meets the controls design requirement on hardware limits.

V. DISCUSSION

A. *Gain tuning*

In this section, we present some insights gathered while tuning the controller to meet the performance metrics. With the reasoning presented in section IV.B, we increased the K_p values to 250 in next set of gains. With this modification, the performance metrics were met, RMS position error = 0.9129. However, this also caused the motor to oscillate about the reference point before settling. On further increasing either K_p, K_d gains, the system became unstable. But later on, we slightly reduced the K_p to 150 while keeping the $K_d = 3$. With this, we recorded a response that satisfied the metrics without jitters or oscillations in the motion.

Table 5: Chosen set of gains in the gain tuning process

Gain #	K_p	K_d	ζ	RMS q_{error}
0	75	3	0.173	2.9423°
1	250	6	0.19	0.9129°
2	250	10	0.316	3.3079°
3	350	3	0.08	4.5128°
4	150	3	0.1225	1.5165°

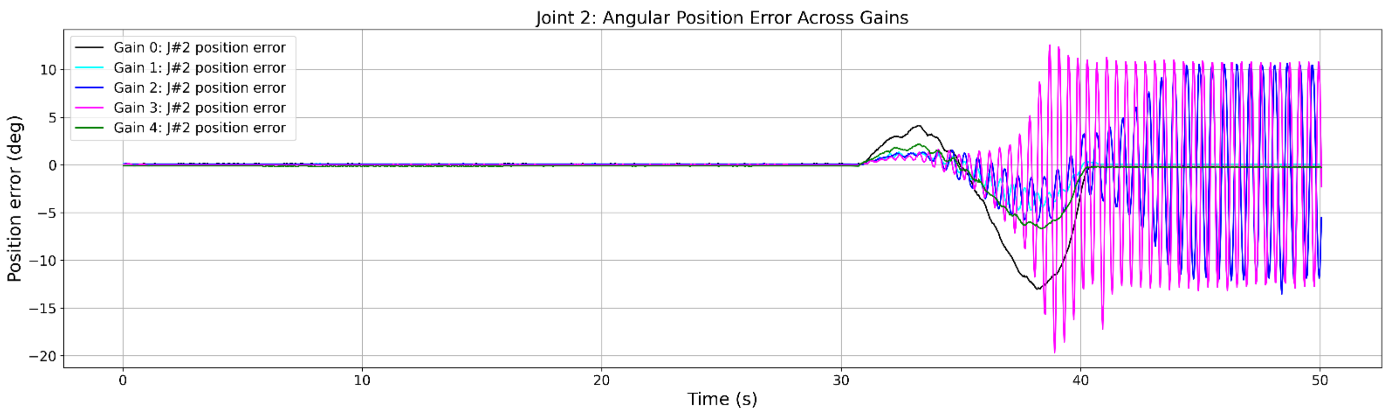


Figure 7: Angular position error in joint 2 for different sets of gains as shown in Table 5

Insights: Slightly increased values of K_p helps in overcoming the frictional resistance. But at the same time, it may induce unwanted oscillation in the response. Further tuning by dropping K_p value or increasing artifical damping (K_d) in the system can reduce the error values while tracking the trajectory. Finally, we recommend the controller with Gain #4.

B. *Challenges*

1. **Jacobian Singularity:**
   The robot occasionally encountered kinematic singularities, leading to a loss of Jacobian invertibility. As a result, the computed joint torques became infeasible, undermining the effectiveness of the control law. Specifically, the control laws (like Impedance control) involving J_a^{-1} in the torque expression, becomes ill-defined in these configurations, destabilizing the commanded motion.

2. **Lack of force sensing:**
   In the absence of dedicated force sensors, we estimated the environmental stiffness experimentally (approximately $5000\ N/m$) to inform gain tuning. This empirical approach enabled us to achieve adequate contact forces for press-fitting tasks while ensuring that the joint torques, particularly at the final joint, remained within safe operational limits (below 0.5 Nm).

3. **Hardware constraints:**
   Practical limitations such as joint friction, low control loop frequency, and communication latency contributed to system instability. These factors hindered the performance of otherwise well-designed control algorithms, underscoring the challenges inherent in real-time robotic implementation.

VI. FUTURE WORK

In the future, we plan to omit the simplifying assumption to expand the robot's capabilities as follows:

1) **Expanding fitting capabilities on vertical and overhead surfaces:**

In addition to vertical press-fitting and stacking operations, the robot will be enhanced to perform assembly tasks on **walls and ceilings**. This will be achieved by:

- Integrating additional degrees of freedom
- Accurate representation of Christoffel symbols and gravity compensation terms

2) **Environment-Agnostic Assembly via Force Sensing**

To enable reliable operation across diverse environments, the system will be equipped with force/torque sensors at the end effector. This will allow the robot to:

- Adapt to varying contact conditions during assembly.
- With parallel motion and force control, perform compliant motion and ensure safe and precise interaction.

ACKNOWLEDGEMENT

We would like to thank our Professor, Dr. Veronica Santos, and our Teaching Assistant, Cole Ten, without whose support this project would not have been possible. We would also like to take this opportunity to express our gratitude to our classmates who kept us motivated throughout this project. Finally, special thanks to the **MakersSpace** team for supporting in 3D-printing of several parts of our robot.

REFERENCES

[1] Tang, Zhenran, Ruixuan Liu, and Changliu Liu. "Eye-in-Finger: Smart Fingers for Delicate Assembly and Disassembly of LEGO." *arXiv preprint arXiv:2503.06848* (2025).

[2] D. -H. Lee, M. -S. Choi, H. Park, G. -R. Jang, J. -H. Park and J. -H. Bae, "Peg-in-Hole Assembly With Dual-Arm Robot and Dexterous Robot Hands," in *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 8566-8573, Oct. 2022, doi: 10.1109/LRA.2022.3187497.

[3] (2010). Motion Control. In: Robotics. Advanced Textbooks in Control and Signal Processing. Springer, London. https://doi.org/10.1007/978-1-84628-642-1_8

[4] Link to codebase:
https://github.com/ayushagrawal149/mae263c_finalPrj_codeBase.git

the final design for our application. Figure A3. Shows the corresponding workspace of our robot, which is an arc-like geometry suitable to serve our problem requirements.

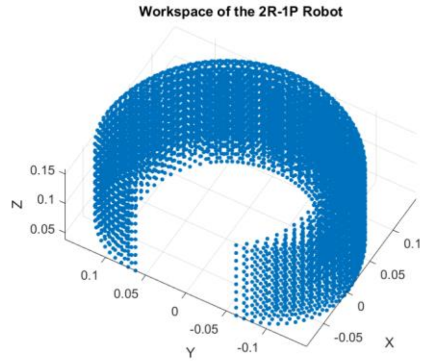


Figure A3: Workspace of 3-DOF robotic manipulator

Moreover, for the pick-and-place task, we thought of designing a two-jaw gripper during the first iteration. Moreover, we conceived of closing the griper jaws to provide force during press-fit. However, due to the brevity of time, we later switched to magnet-based operation, where the magnet at the tip of the prismatic joint would pick up the Lego block (which also contains a magnet). This simplified our block collection and placement task to a great extent.

APPENDIX

A. *Design of our Robotic Manipulator:*

In this section, we briefly describe the iterations in the design of our robotic manipulator for the Lego-assembly task. We started with the design of a 4-DOF robotic manipulator, whose first two DOFs are used to reach a desired x-y position. The remaining two DOFs are used to pick or place the block. Figure A1 shows CAD model of our robot during first iteration.

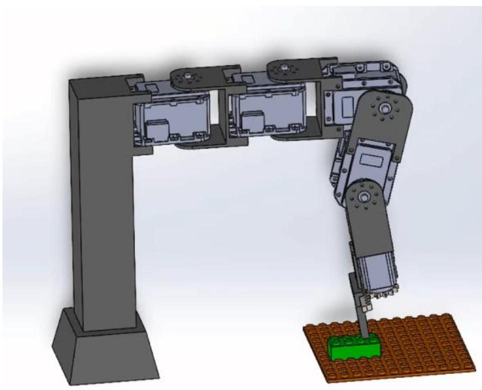


Figure A1: 4-DOF robotic manipulator for Lego assembly building

The corresponding workspace is shown in Figure A2. It can be clearly seen that the spherical nature of the workspace and coupling between the third and fourth revolute joints make it extremely difficult to exactly lower the gripper for pickup/ placing.

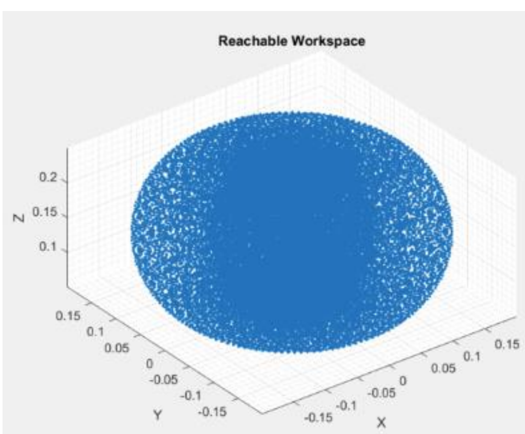


Figure A2: Workspace of 4-DOF manipulator

This gave us the motivation to decouple the horizontal and vertical motion by adding a prismatic joint instead of the last two DOFs. Moreover, the dynamics became clean because of the decoupling, resulting in a unique solution for the inverse kinematics. This resulted in the 3-DOF manipulator (2R-1P) as

B. *Trajectory Planning for Lego assembly*

We worked out the trajectory planning for the pick and place of a single Lego block, which helps us to demonstrate our design performance in hardware. This can be easily extended to multiple Lego blocks for building entire assemblies. Figure B1 shows to initial pick-up point for the Lego block wrt. Our robot.

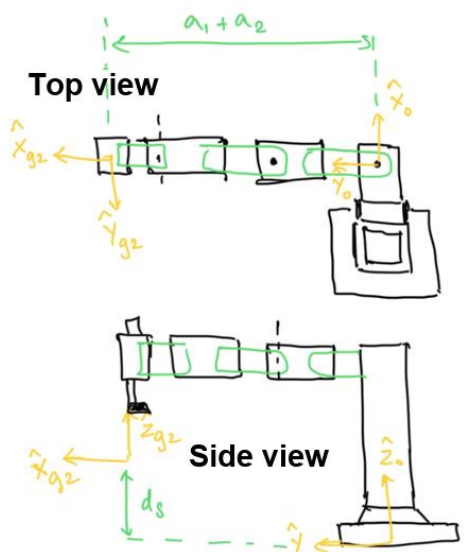


Fig B1. Initial pose for the manipulator operation

The final pose for the operation, along with the placement of the Lego bed, is shown in Figure B2.

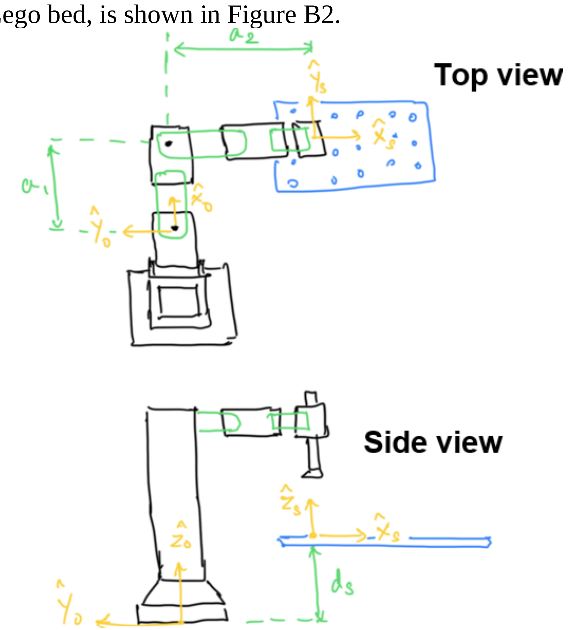


Figure B2: Reachable workspace of 4-DOF manipulator

With this, we defined the via-points of our problem and provided sufficient time between each via-point to have a smooth trajectory. Using the inverse dynamics, we computed the joint angles at those via-points. Moreover, we define zero initial and final joint velocities and accelerations for each pair of via-points. Finally, for the set of six boundary conditions, we fitted a quintic polynomial. The designed trajectories (position, velocity, and acceleration) for various joints are shown in Figures B3-B5

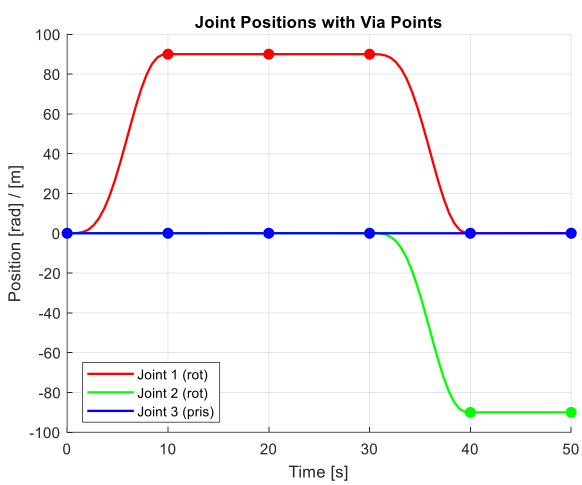


Figure B3: Position trajectory of various joints

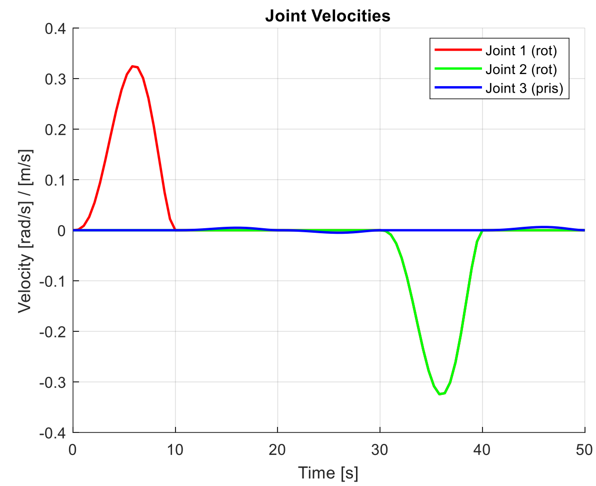


Figure B4: Velocity trajectory of various joints

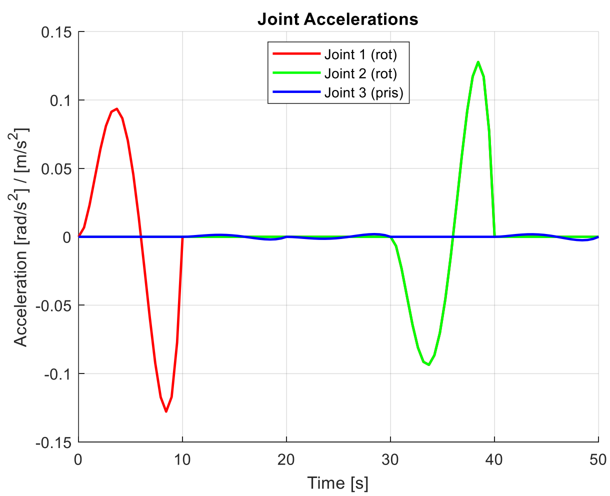


Figure B5: Acceleration trajectory of various joints

C. *Team Members and Contributions*

1. ***Ayush Agrawal***: literature review, Joint space inverse dynamics control (tuning and design), Impedance control, Trajectory data post processing for report writing
2. ***Debajyoti Chakrabarti***: literature review, forward and inverse kinematics, trajectory planning, Joint space inverse dynamics control (tuning and design)
3. ***Premkumar Sivakumar***: literature review, robot CAD design and 3D-printing, trajectory planning, Joint space inverse dynamics control (tuning and design)