UNIVERSITY OF PENNSYLVANIA

Final Project: Pick and Place Challenge

MEAM 5200 - Introduction to Robotics

Authors:
ZHIXIN YIN

Professor:
RACHEL HOLLADAY

Last Updated: 18 Dec, 2025

Contents

1 Introduction **2**

2 Methods **2**

2.1 Framework and Kinematic Theory . . . 2

2.2 Static Block Manipulation . . . 3

2.2.1 Pick up procedure . . . 3

2.2.2 Stacking procedure . . . 4

2.3 Dynamic Block Manipulation . . . 5

2.3.1 Strategy and Rationale . . . 5

2.3.2 Grasp Position Calculation . . . 5

2.3.3 End Effector Pose Construction . . . 6

2.3.4 Multi-Waypoint Navigation for IK Convergence . . . 6

2.3.5 Periodic Grasping Mechanism . . . 7

2.3.6 Complete Dynamic Sequence . . . 7

2.4 Block Stacking . . . 8

3 Evaluation **9**

3.1 Static Block Manipulation . . . 9

3.2 Dynamic Block Manipulation . . . 9

4 Analysis **10**

4.1 Static Block Manipulation . . . 10

4.2 Dynamic Block Manipulation . . . 11

5 Challenges and Lesson Learned **12**

1 Introduction

The Pick and Place Challenge requires us to leverage all the concepts that we have learned throughout the course and help the FRANKA EMIKA Panda Robot Arm to pick up blocks autonomously in different environments. In particular, there are a shared turntable and a static platform where our robot can pick up and stack as many blocks as possible. Additionally, we have to combine our learned knowledge and the given technology such as the `ObjectDetector` and `ArmController` classes to develop a robust algorithm to achieve the best score. In the early phase, we can test our algorithms on Gazebo simulation environment and find out the most appropriate solution. After that, we can translate our work to the hardware with several testing sessions, where we can fine tune real-life parameters. This report will further discuss our solution and show how it worked in practice sessions, but not in the actual competition.

2 Methods

There are many options to develop a "Pick and Place" plan for the arm including Rapidly-Exploring Random Trees (RRT), Potential Fields, and Inverse Kinematics (IK). However, the first two choices are only better at complex environments, and can also cause computational overhead. Therefore, we selected IK with secondary task of joint centering to maintain the end effectors and joints at a relatively neutral position. In the following parts, we will describe how we integrated the given technology and what we have learned to handle both static and dynamic blocks.

2.1 Framework and Kinematic Theory

Before motion planning steps, we had to develop a reliable method to pick and drop the blocks. Thus, we are given `ObjectDetector` and `ArmController` classes to manage the arm's vision and movement, respectively.

First, the `ObjectDetector` class will process the vision images captured from the camera mounted at the front of the arm. There are methods that provide the transformation matrices between camera-block position, and camera-end effector position. We will also need to use forward kinematics (`calculateFK`) to track the end effector absolute position. As a result, we can get the precise position of each block in the world frame by applying this formula:

$$T_0^{\text{block}} = T_0^{\text{ee}} \cdot T_{\text{ee}}^{\text{camera}} \cdot T_{\text{camera}}^{\text{block}} \tag{1}$$

For fine control, we need the `ArmController` class. In particular, we used it to open, close, and check the gripper if there is any grabbed block. Furthermore, the `safe_move_to_position` method would check for collision and help us visualize a better path.

Regarding the primary task of finding joint velocities for a desired pose, we used the least-squares solution given in Lecture 10:

$$\dot{q_p} = J^+(q)\,\xi$$

Specifically, $\dot{q}$ (a 7×1 vector) is the desired joint velocity, J^+ is the generalized pseudoinverse of the Jacobian matrix, and ξ is a 6×1 vector containing the linear and angular velocity of the end-effector. In the Panda arm case, we have more joints than the dimensionality of the workspace, so we apply the "short and fat" (right) pseudoinverse formulation from Lecture 10:

$$J^+ = J^T(JJ^T)^{-1}$$

Additionally, we implement a gradient descent algorithm to solve the inverse kinematics problem as:

$$\Delta q = -\alpha J^+(q)\,e$$

Since the arm has more joints than the dimensionality of the workspace, we can implement a secondary task. According to Lecture 12, we compute the null space and apply the projection operator as:

$$N = I - J^+ J$$

$$z = N \cdot b$$

Here, b represents the joint velocity command for the secondary task of keeping each joint near the center of its joint limits. Projecting this command into the null space allows the robot to perform the secondary task without significantly affecting the primary end-effector motion. Thus, the final velocities for the joints are:

$$\dot{q} = \dot{q_p} + z$$

2.2 Static Block Manipulation

2.2.1 Pick up procedure

We kept the strategy of picking the static blocks as simple as possible. First, we predefined (as Table 1) the starting position, the initial block spotting position to observe and grab the static blocks, and the block placing position to drop them. After that, we used IK to autonomously control the arm to move between the positions. Note that we chose these configurations after referring to some work of previous cohort and numerous simulation experiments. For more detail, when the arm reached spotting position, we will use Equation

Name	Configuration
Starting Position (0)	$[-0.0178, -0.7601, 0.0198, -2.3421, 0.0298, 1.5412 + \pi/2, 0.7534]$
Static Block Spotting	**Blue:** $[0.1345, -0.0507, 0.1806, -1.6082, 0.0091, 1.5583, 1.1004]$ **Red:** $[-0.1645, -0.0505, -0.1516, -1.6082, -0.0076, 1.5583, 0.4693]$
Block Placing Position	**Blue:** $[-0.1354, 0.1106, -0.1694, -2.0057, 0.0218, 2.1146, 0.4704]$ **Red:** $[0.2050, 0.1096, 0.0977, -2.0057, -0.0125, 2.1147, 1.0939]$

Table 1: Configuration details for the blue and red teams

(1) to get the absolute position and orientation of a block. The matrix T_{camera}^{block} is returned from the method `ObjectDetector.get_detections`, while T_{ee}^{camera} is taken from the method `ObjectDetector.get_H_ee_camera`. Furthermore, the end effector position in world frame is computed by applying forward kinematics.

The manipulation process begins with the robot arm moving to a predefined observation pose $\mathbf{q}_{\text{spot}}$, which provides a clear view of the workspace. At this pose, the vision detector is used to identify all visible blocks, and only blocks labeled as *static* are considered for grasping.

For a detected block with pose $\mathbf{T}^c_{\text{block}}$ expressed in the camera frame, a coarse world-frame estimate of the block pose is computed using forward kinematics and the camera–end-effector calibration:

$$\mathbf{T}^0_{\text{block, coarse}} = \mathbf{T}^0_e(\mathbf{q}_{\text{spot}})\, \mathbf{T}^e_c\, \mathbf{T}^c_{\text{block}}. \tag{2}$$

From this transform, only the planar position is extracted:

$$x_{\text{coarse}} = \mathbf{T}^0_{\text{block, coarse}}(0,3), \qquad y_{\text{coarse}} = \mathbf{T}^0_{\text{block, coarse}}(1,3). \tag{3}$$

At this stage, the estimate is intentionally coarse and is used only to guide the arm above the block, rather than to perform a direct grasp. This design choice is motivated by empirical observations: when a block is placed at a sharp angle or near the corner of the platform, the estimated block orientation can become unreliable due to camera perspective distortion and limited viewing angles. Using such inaccurate orientation estimates directly for grasping often results in misalignment between the gripper and the block, leading to diagonal or unstable grasps.

In contrast, the estimated planar position (x, y) of the block remains relatively accurate even under these challenging conditions. Therefore, during the first observation stage, only the block's planar position is used. The robot arm moves to a hover pose located approximately 15 cm above the block center, preparing for a second observation step in which the block pose, especially its orientation, can be refined from a closer and more favorable viewpoint.

After moving to the hover pose, a second detection is performed for the same static block in order to refine its pose estimate. Using the updated joint configuration, the block pose in the world frame is recomputed as

$$\mathbf{T}^0_{\text{block}} = \mathbf{T}^0_e(\mathbf{q}_{\text{hover}})\, \mathbf{T}^e_c\, \mathbf{T}^c_{\text{block, refined}}. \tag{4}$$

From the refined rotation matrix $\mathbf{R}^0_{\text{block}} \in R^{3\times3}$, the block's local x- and y-axes expressed in the world frame are extracted as

$$\mathbf{a}_x = \mathbf{R}^0_{\text{block}}(:,1), \qquad \mathbf{a}_y = \mathbf{R}^0_{\text{block}}(:,2). \tag{5}$$

Since the grasp is performed from above, only the orientation projected onto the horizontal $(x{-}y)$ plane is relevant. Therefore, the planar projections of the two axes are evaluated:

$$p_x = \|\mathbf{a}_x^{xy}\|, \qquad p_y = \left\|\mathbf{a}_y^{xy}\right\|. \tag{6}$$

The axis with the larger planar projection is selected as the block's principal edge direction, as it is less affected by perspective distortion and provides a more reliable reference for gripper alignment. The corresponding yaw angle is then computed as

$$\theta = \text{atan2}(a_y, a_x). \tag{7}$$

To ensure that the gripper aligns with the block edges while avoiding unnecessary 90° rotations, the angle is wrapped into the minimal equivalent range:

$$\theta_{\text{aligned}} = \left(\theta + \frac{\pi}{4}\right) \bmod \frac{\pi}{2} - \frac{\pi}{4}. \tag{8}$$

Finally, the desired grasp orientation is constructed by rotating a fixed downward-facing end-effector orientation $\mathbf{R}_{\text{down}}$ about the world z-axis:

$$\mathbf{R}_{\text{grab}} = \mathbf{R}_z(\theta_{\text{aligned}})\,\mathbf{R}_{\text{down}}. \tag{9}$$

After that we only need to maneuver the gripper to specific heights as defined in Figure 1. The grasp execution is formulated directly in terms of a desired end-effector pose expressed as a homogeneous transformation matrix. Given the refined block pose and the computed grasp orientation $\mathbf{R}_{\text{grab}}$, a target hover pose is defined as

$$\mathbf{T}^0_{\text{hover}} = \begin{bmatrix} \mathbf{R}_{\text{grab}} & \mathbf{p}_{\text{block}} + \begin{bmatrix} 0 \\ 0 \\ z_{\text{hover}} \end{bmatrix} \\ \mathbf{0}^\top & 1 \end{bmatrix}, \tag{10}$$

where z_{hover} denotes a fixed vertical offset above the block center, ensuring sufficient clearance for safe approach.

The corresponding joint configuration is obtained by solving inverse kinematics using `IK_position_null`, which incorporates a secondary joint-centering objective to avoid joint limits. The robot arm is then commanded to move to the hover pose above the block.

From the hover configuration, the end-effector is translated vertically downward to a predefined grasp height while maintaining the same orientation. The gripper is then closed to grasp the block, after which the end-effector is lifted vertically by a fixed offset to reach a safe transport height.

Grasp success is evaluated based on the distance between the two gripper fingers after closure. If the measured gripper opening lies within a tolerance band of ±1 cm around the known block width, the grasp is considered successful. Otherwise, the grasp is deemed to have failed. In the event of a failed grasp, the robot returns to the initial observation pose and repeats the entire perception and grasping procedure. Upon a successful grasp, the controller transitions to the stacking phase, where the block is transported to the predefined stacking location.

2.2.2 Stacking procedure

Similarly to the picking task, we divide the dropping procedure into smaller checkpoints where the arm can safely adjust its height. First, we define z_{top} as the sum of the height of the platform and the number of stacked blocks (In Figure 2, assume there is only 1 stacked block). Then, we will define z_{stack} half a block higher than z_{top} as the dropping height for the gripper to open. After successfully stacking block number 2, we compute the new z_{top} and move the arm $10cm$ up to the new z_{hover} height to safely maneuver back to the other static blocks.

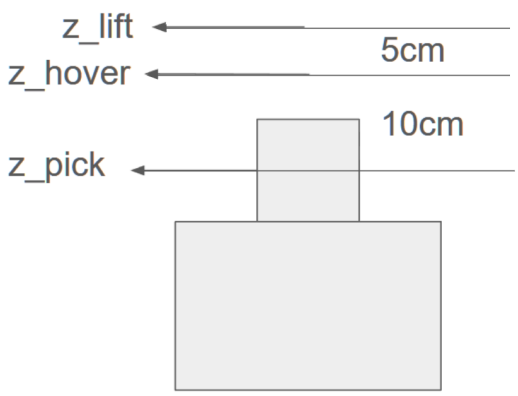

Figure 1: Visualization for z_{pick}, z_{lift}, and z_{hover} when picking blocks

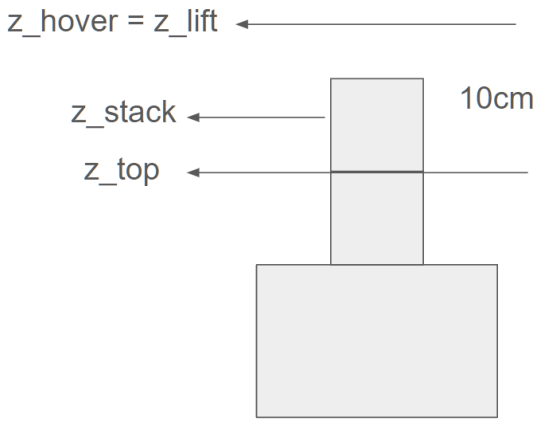

Figure 2: Visualization for z_{top}, z_{stack}, and z_{hover} when dropping blocks

2.3 Dynamic Block Manipulation

2.3.1 Strategy and Rationale

In the dynamic scenario, blocks are placed on a rotating turntable with radius 0.305 m centered at the world frame origin, with its surface 0.200 m above the world's x-y plane. Traditional approaches typically involve vision-based tracking and trajectory prediction, where the robot detects blocks using `ObjectDetector` class, estimates their angular velocity, and computes future positions to intercept the moving targets. However, through extensive simulation testing, we observed that such prediction-based methods achieved only $20 - 30\%$ success rates due to cumulative errors in angular velocity estimation, timing synchronization issues, and the computational overhead of continuous detection.

After analyzing these failure modes, we adopted an alternative strategy that we term "Fixed-Position Periodic Grabbing." Rather than actively tracking and predicting block trajectories, we position the end-effector at a fixed point on the turntable's perimeter and execute periodic gripper open-close cycles. When a block passes through this fixed grasping point, the gripper captures it. This approach eliminates the need for real-time vision processing, coordinate transformations between turntable and world frames, and angular velocity prediction, resulting in a significantly more robust and deterministic system.

2.3.2 Grasp Position Calculation

The fixed grasping position is computed based on the geometric relationship between the robot base frame and the turntable as it is shown in the lab manual.

The turntable center is located at the world frame origin $(0, 0)$, while the robot bases are positioned at $y = +0.990$m (red team) and $y = -0.990$m (blue team) in world coordinates. To transform positions between the world frame and the robot base frame, we apply:

$$y_{\text{base}} = y_{\text{world}} - y_{\text{robot}}$$

where $y_{\text{robot}} = -0.990$m for the blue team and $y_{\text{robot}} = +0.990$m for the red team. Thus, the turntable center in the base frame is located at $(0, -0.99)$ for the blue team and $(0, +0.99)$ for the red team.

The grasping position is selected at a fixed radius r_{block} from the turntable center, along the line connecting the turntable center to the robot base. This ensures the end-effector intercepts blocks at the point closest to the robot, maximizing reachability. Since blocks are randomly dispersed on the turntable (which has a radius of 0.305 m), the parameter r_{block} must be tuned based on the observed block placement in each match. For the blue team, the grasp position in the base frame is:

$$\mathbf{p}_{\text{grasp}} = \begin{bmatrix} 0 \\ -0.99 + r_{\text{block}} \\ z_{\text{platform}} + \dfrac{h_{\text{block}}}{2} \end{bmatrix}$$

where r_{block} is the tunable block radius on the turntable (typically between 0.18 m and 0.25 m), $z_{\text{platform}} = 0.20$m is the turntable surface height, and $h_{\text{block}} = 0.05$m is the block height. For the red team, the y-component becomes $0.99 - r_{\text{block}}$.

2.3.3 End Effector Pose Construction

Once the grasping position is determined, we construct the complete end-effector poses for the grasping sequence. The end-effector orientation is set with the gripper pointing downward, perpendicular to the turntable surface:

$$R_{\text{grab}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

Three key poses are constructed as homogeneous transformation matrices. The hover pose positions the end-effector above the grasping point:

$$T^0_{\text{hover}} = \begin{bmatrix} R_{\text{grab}} & \mathbf{p}_{\text{grasp}} + \Delta z_{\text{hover}} \cdot \hat{\mathbf{z}} \\ \mathbf{0}^T & 1 \end{bmatrix}$$

where $\Delta z_{\text{hover}} = 0.10$m. The pick pose places the gripper at the block center height:

$$T^0_{\text{pick}} = \begin{bmatrix} R_{\text{grab}} & \mathbf{p}_{\text{grasp}} \\ \mathbf{0}^T & 1 \end{bmatrix}$$

The lift pose raises the block after successful grasping:

$$T^0_{\text{lift}} = \begin{bmatrix} R_{\text{grab}} & \mathbf{p}_{\text{grasp}} + \Delta z_{\text{lift}} \cdot \hat{\mathbf{z}} \\ \mathbf{0}^T & 1 \end{bmatrix}$$

where $\Delta z_{\text{lift}} = 0.15$m.

2.3.4 Multi-Waypoint Navigation for IK Convergence

A significant challenge we encountered was that the inverse kinematics solver frequently failed to find valid joint configurations when transitioning between the static block stacking position and the dynamic grasping position. These two workspace regions require substantially different arm configurations: static block manipulation occurs on one side of the robot (toward the static platform at $y = \pm 1.159$m), while dynamic block grasping requires extending the arm in the opposite direction (toward the turntable at the world origin). When the IK solver uses the current joint configuration as its initial seed, it may converge to solutions that violate joint limits or fail to converge entirely due to the large configuration space distance between these regions.

To address this challenge, we implemented a multi-waypoint navigation strategy. Instead of directly commanding the end-effector to the target pose, we define a sequence of intermediate joint configurations (waypoints) that guide the arm through the configuration space. These waypoints are predefined based on the team assignment and progressively orient the arm toward the turntable while gradually extending the reach.

First, we define a ready configuration q_{ready} that serves as a neutral starting point facing the turntable:

$$q^{\text{blue}}_{\text{ready}} = \left[-0.6, 0.0, 0.0, -\frac{\pi}{2}, 0.0, \frac{\pi}{2}, \frac{\pi}{4}\right]^T$$

$$q_{\text{ready}}^{\text{red}} = \left[0.6, 0.0, 0.0, -\frac{\pi}{2}, 0.0, \frac{\pi}{2}, \frac{\pi}{4}\right]^T$$

Two additional waypoints further extend the arm configuration toward the turntable. For the blue team:

$$q_{\text{wp1}}^{\text{blue}} = \left[-0.9, 0.2, 0.0, -1.8, 0.0, 1.6, \frac{\pi}{4}\right]^T$$

$$q_{\text{wp2}}^{\text{blue}} = \left[-1.2, 0.3, 0.0, -1.5, 0.0, 1.8, \frac{\pi}{4}\right]^T$$

For the red team, the waypoints are mirrored with respect to the first joint angle (base rotation):

$$q_{\text{wp1}}^{\text{red}} = \left[0.9, 0.2, 0.0, -1.8, 0.0, 1.6, \frac{\pi}{4}\right]^T$$

$$q_{\text{wp2}}^{\text{red}} = \left[1.2, 0.3, 0.0, -1.5, 0.0, 1.8, \frac{\pi}{4}\right]^T$$

The robot sequentially moves through $q_{\text{ready}} \rightarrow q_{\text{wp1}} \rightarrow q_{\text{wp2}}$ before the IK solver computes the final configuration for T_{hover}^0. By using q_{wp2} as the IK seed, the solver starts from a configuration that is much closer to the target in joint space, dramatically improving convergence reliability.

When returning from the turntable to the stacking position after a successful grasp, we traverse the waypoints in reverse order: $q_{\text{wp2}} \rightarrow q_{\text{wp1}} \rightarrow q_{\text{ready}} \rightarrow q_{\text{stack}}$. This symmetric approach ensures that the IK solver always receives an appropriate seed configuration, regardless of the direction of motion.

2.3.5 Periodic Grasping Mechanism

Once the end-effector reaches the pick position, we execute periodic gripper cycles to capture blocks as they pass. The gripper alternates between closed and open states with configurable timing parameters:

1. Close the gripper and hold for $t_{\text{close}} = 0.5$s
2. Query the gripper state to determine if a block was captured
3. If no block is detected, open the gripper and wait for $t_{\text{cycle}} = 1.5$s before the next attempt
4. Repeat for up to 8 cycles per descent

Block detection is performed by reading the gripper finger positions from the `ArmController` using the `get_gripper_state` method. When a block is successfully grasped, the total gripper width $w = w_{\text{left}} + w_{\text{right}}$ falls within a characteristic range:

$$w_{\min} \leq w \leq w_{\max}$$

where $w_{\min} = 0.040$m and $w_{\max} = 0.060$m correspond to the 50 mm block dimensions with tolerance for gripper compliance. If w falls within this range, we conclude that a block has been captured.

2.3.6 Complete Dynamic Sequence

The complete dynamic block manipulation procedure consists of the following steps:

1. **Initialization:** Move to the ready configuration q_{ready} facing the turntable.
2. **Approach:** Navigate through waypoints $q_{\text{wp1}} \rightarrow q_{\text{wp2}}$ to extend the arm toward the turntable.
3. **Positioning:** Use IK to move to T_{hover}^0, then descend to T_{pick}^0.
4. **Periodic Grasping:** Execute gripper open-close cycles until a block is detected or the maximum cycle count is reached.

5. **Lifting:** Upon successful detection, move to T^0_{lift} and verify the block remains in the gripper.
6. **Return Transit:** Traverse waypoints in reverse order ($q_{\text{wp2}} \rightarrow q_{\text{wp1}} \rightarrow q_{\text{ready}}$) to return to a safe configuration.
7. **Stacking Transition:** Move to the stacking configuration q_{stack}.
8. **Stacking:** Compute the stacking pose based on the current stack height and place the block.
9. **Repeat:** Return to q_{ready} and repeat from Step 2 for the next block.

This approach achieved near-perfect success rates in simulation. For hardware deployment, only two parameters require tuning: the block radius r_{block} (to match the actual block placement on the turntable) and the platform height z_{platform} (to account for physical turntable height variations). The elimination of vision-based tracking and trajectory prediction makes the system inherently more robust to sensor noise and timing uncertainties that are prevalent in real-world conditions.

2.4 Block Stacking

After a block is successfully picked up (either static or dynamic), it must be placed on the team's goal platform to score points. According to the competition rules, points are awarded based on the formula Points $=$ Value $\times$ Altitude, where Altitude is the distance from the block center to the goal platform surface in millimeters, and Value is 10 for static blocks and 20 for dynamic blocks. Therefore, stacking blocks vertically maximizes the score.

The goal platform for each team is a 0.250 m $\times$ 0.250 m surface located at $(0.562, \pm 0.731)$m in the world frame, with its surface 0.200 m above the ground. We use a predefined stacking configuration q_{stack} that positions the end-effector above the goal platform center. The stacking position coordinates $(x_{\text{stack}}, y_{\text{stack}})$ are extracted from this configuration using forward kinematics:

$$T^0_{\text{stack}} = \text{FK}(q_{\text{stack}})$$

$$x_{\text{stack}} = T^0_{\text{stack}}[0,3], \quad y_{\text{stack}} = T^0_{\text{stack}}[1,3]$$

For each block placement, the stacking height is computed based on the current number of blocks already on the platform. Let n denote the number of blocks currently stacked. The target placement height is:

$$z_{\text{stack}} = z_{\text{platform}} + n \cdot h_{\text{block}} + \frac{h_{\text{block}}}{2}$$

where $z_{\text{platform}} = 0.20$m and $h_{\text{block}} = 0.05$m. The hover height above the stack is set to $z_{\text{hover}} = z_{\text{platform}} + n \cdot h_{\text{block}} + 0.10$m to provide clearance. The stacking pose is then constructed as:

$$T^0_{\text{stack}} = \begin{bmatrix} R_{\text{approach}} & [x_{\text{stack}}\ y_{\text{stack}}\ z_{\text{stack}}]^T \\ \mathbf{0}^T & 1 \end{bmatrix}$$

where R_{approach} is the downward-facing orientation matrix identical to R_{grab}.

The stacking sequence proceeds as follows: the robot first moves to the hover position above the current stack top, then descends to the placement height, opens the gripper to release the block, and finally retreats upward. After each successful placement, the stack counter n is incremented, ensuring that the subsequent blocks are placed at progressively higher altitudes. Our approach will focus on stacking all four static blocks first and then proceed to handle the dynamic ones to achieve the best score possible.

3 Evaluation

3.1 Static Block Manipulation

To evaluate our method of stacking static blocks, we set up the simulation and ran it for ten times. At first, we had many failures as Figure 4 where the robot arm cannot avoid the platform. However, we only needed some slight adjustments to fix that: using `safe_move_to_position()` function more constantly and defining the desired location more carefully. After that, the system always reached successful state as in Figure 3. Therefore, we had high expectations when setting up for the hardware experiments.

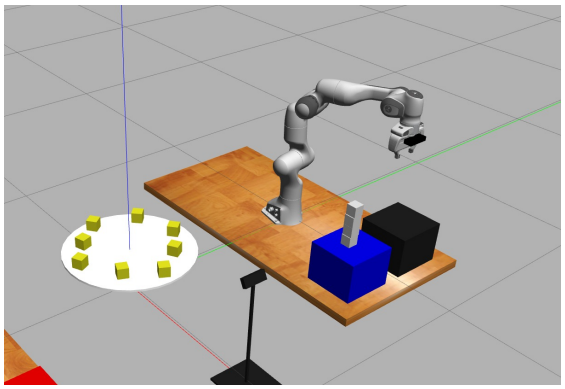

Figure 3: Success static block stacking simulation attempt

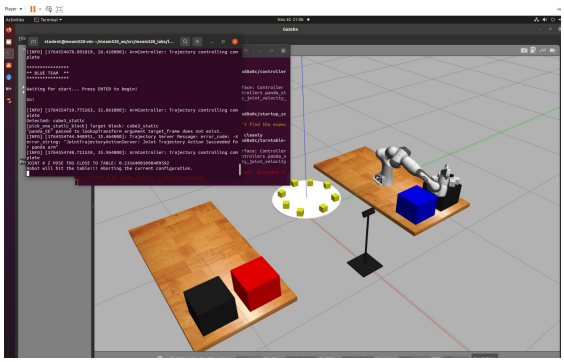

Figure 4: Failed static block stacking simulation attempt

As expected, the hardware experiments finished well and the process is demonstrated in Figure 7. The first photo showed that the gripper is properly oriented according to the orientation of the target block. Then, the arm proceeded to grab it, move to the desired height and location, then drop it on the previous block. Note that we successfully stacked all 4 static blocks within 2 minutes, which is quite good for the competition. However, we could also reduce that by letting $z_{lift} = z_{hover}$ at some points of the process. This successful trial also proved that our algorithm worked and only need calibrations for the final competition.

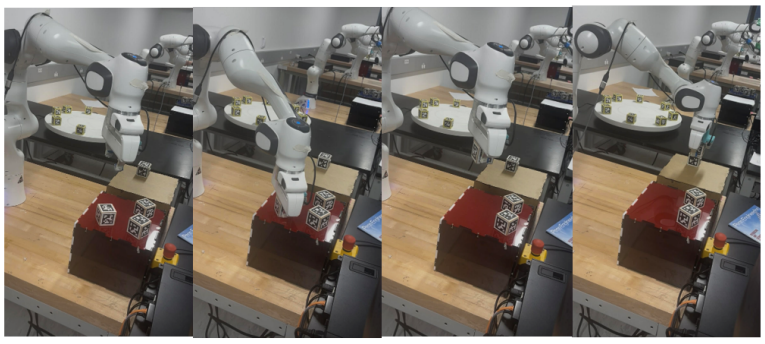

Figure 5: From left to right, process of capturing static blocks on hardware

3.2 Dynamic Block Manipulation

We evaluated the dynamic module as a complete end to end capability: after completing the static block stage, the Panda arm transitions into a fixed position periodic capture policy, executes a waypoint guided inverse kinematics approach into a single capture pose above the turntable, and then reuses the same placement and stacking routine

used in the static pipeline. To report performance in a way that is both objective and diagnostic, we tracked three metrics: dynamic grasp to lift success, defined as a block being securely acquired and lifted with clear separation from the turntable surface; dynamic grasp to stack success, defined as a grasped block being placed onto the stack without destabilizing previously placed blocks; and a failure mode breakdown that distinguishes inverse kinematics infeasibility from timing and contact related failures.

Beyond binary outcomes, we logged the tuned control and geometry parameters that materially determine success on the real Panda arm. The tuned grasp radius relative to the turntable center was approximately 0.24 m, and the grasp height was set to the platform height plus half the block height, which yields a pick height of 0.225 m given a 0.20 m platform and 0.05 m blocks. The end effector executed a periodic capture window with a closed hold of 0.5 s followed by an open wait of 1.5 s, producing a 2.0 s sampling period that repeats up to eight cycles per descent, which bounds the per descent capture time budget and provides a concrete basis for comparing different tuning choices. The gripper based success detector used the measured finger separation as an internal proxy for contact, declaring a grasp when the combined opening lies between 0.040 m and 0.060 m, a range that aligns with the 50 mm cube size plus compliance and tolerances.

To expose both strengths and limits, we performed parameter stress tests that intentionally push the system toward its dominant sensitivities. When the grasp radius was moved inward toward the turntable center, within the lower end of the practical tuning range around 0.18 m, inverse kinematics frequently failed before any motion was executed, collapsing the dynamic success rate even though gripper timing and the capture logic were unchanged, which shows that the method is bounded first by kinematic feasibility of the prescribed capture pose rather than by timing alignment alone. On hardware, two calibration parameters proved load bearing. The effective lateral alignment of the turntable relative to the robot base, which operationally is the grasp radius and its associated offset, directly controls whether blocks actually pass through the capture locus and whether the inverse kinematics solution remains comfortably inside joint limits. The grasp height offset is equally critical because it must simultaneously allow full finger closure on the block while maintaining clearance from the table surface; a competition relevant failure occurred when this height was not tuned conservatively, where the fingers contacted the table during closure and caused missed grasps or disturbances despite correct lateral alignment.

Finally, because the policy does not explicitly estimate turntable phase, we validated the sensitivity to angular speed mismatch between simulation and hardware by retuning the gripper cycle timing. When the open close period was adjusted to match the realized turntable angular velocity, dynamic success improved noticeably, consistent with the policy's reliance on repeated phase coincidence rather than on prediction.

4 Analysis

4.1 Static Block Manipulation

In the simulation environment, our static grasping success rate reached 100%, with perfectly aligned blocks and no noticeable positional or angular deviation. In the laboratory validation, we had sufficient time to individually measure the hardware offsets and compensate for them through software adjustments. As a result, the experimental performance was also excellent, achieving a 100% success rate.

Although we have shown in the previous part that our method successfully stacked all four blocks in the hardware experiment, we did not perform well in the final competition. The reason was that we did not have enough time to measure all of the offset parameters for both Red and Blue sides. These offsets came from the different environment set up between the laboratory and the final stage. As a result, our arm constantly had collisions with the blocks instead of grabbing the center of them. It was unfortunate that we only needed to change the original offsets by a few centimeters to strongly compete in the competition. However, it was also a great lesson for us so that we have to be more well-prepared in future projects. As the robot failed in the static challenge, it stopped moving and we did not have a chance to test the dynamic one. However, we are also not very confident for the dynamic challenge for the same reasons.

4.2 Dynamic Block Manipulation

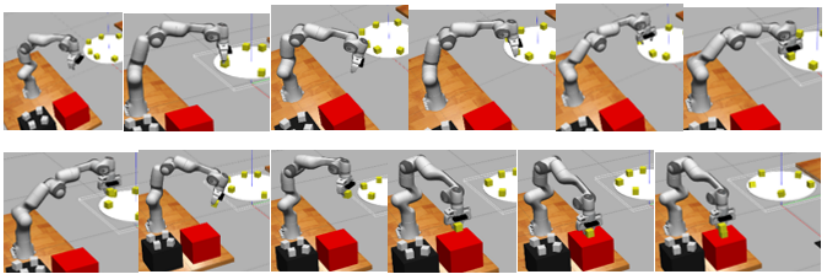

Figure 6: From left to right, from top to bottom, process of capturing dynamic blocks in simulation

Figure 7: From left to right, from top to bottom, process of capturing dynamic blocks on hardware

Figure 6 and Figure 7 illustrate the simulation and hardware execution of the dynamic pipeline, respectively, including the transition from the stacking workspace to a predefined turntable approach configuration, the descent into a fixed capture pose, the gripper closure during periodic sampling, and the subsequent retreat and placement back onto the stack. Both the grasping stage and the stacking stage rely on intermediate waypoints rather than a single direct inverse kinematics solve. This choice reduces throughput because each waypoint adds motion time and dwell time, but it substantially improves reliability by keeping the arm in well-conditioned configurations and by providing consistent seeds for the solver. In practice, waypoint-guided motion prevents rare but disruptive events where an ill-conditioned inverse kinematics solution causes large joint excursions, which can appear as unpredictable arm swings and can jeopardize nearby contacts, the turntable clearance, and the integrity of the existing stack.

The strong dependence on grasp radius is best explained by constrained inverse kinematics geometry on the Panda arm. The capture pose constrains both position on a circular locus and a near-fixed orientation with the tool pointing downward, which is beneficial for stable top-down acquisition but restricts the wrist and elbow to a narrow set of configurations; as the grasp radius shrinks, the end effector is commanded inward while the orientation constraint remains unchanged, and the solver is forced toward elbow folding and wrist alignments that approach joint limits, self-collision constraints, or near-singular configurations. So inverse kinematics fails not because the target is close in Euclidean distance, but because it is unreachable under the required approach direction and orientation.

The hardware parameters identified during testing are analytically central because they define whether the commanded motion is physically realizable as a contact task. The grasp height offset creates a narrow feasible band: if it is too low, the fingers collide with the table during closure, while if it is too high, the fingers close with insufficient normal force on the block side faces, increasing slip risk during lift. This band is systematically narrower in hardware than in simulation due to unmodeled compliance and small calibration bias. The lateral alignment of the capture locus plays a dual role, since it affects inverse kinematics margin and also determines the true relative position between the closing fingers and the block trajectory; even small systematic errors in this offset appear as

persistent lateral misalignment at the capture point, which reduces the probability that a periodic closure overlaps the block footprint.

The timing behavior of periodic grasping can be treated as a phase locking problem, which clarifies why angular velocity tuning is so influential. If the blocks are approximately uniformly distributed with angular spacing $\Delta\theta$ approximately equal to 2π divided by N, and the turntable angular velocity is ω, then the nominal time between successive blocks crossing the capture ray is T_b equal to $\Delta\theta$ divided by ω. The gripper executes capture attempts with period T_g and an effective closure window T_c, and overlap probability is maximized when T_g is commensurate with T_b and the closure phase is shifted so the center of T_c aligns with the expected crossing time. In hardware, even a small ω mismatch introduces phase drift that accumulates over cycles and reduces overlap frequency, which is why retuning the gripper period and the open-close dwell ratio can materially increase success without changing the geometric capture pose. The implementation reflects this structure directly by using a 0.5 s closed hold and a 1.5 s open wait, yielding a 2.0 s sampling period that can be adjusted to better match the realized crossing interval.

Even without model-based prediction, real-time delays behave like an effective phase offset. Camera to controller latency, command buffering, and the physical finger closure time shift the actual closure event later than the software trigger, so the correct compensation is a phase advance, triggering closure earlier by an amount approximately equal to the measured delay τ so that the fingers are closed when the block arrives.

A final phenomenon that explains the observed stack skew is the interaction between non-uniform block spacing and fixed-frequency sampling. Because the blocks on the turntable are not uniformly distributed while the capture cycle is fixed, many successful grasps occur when the block is not centered at closure, producing a small lateral grasp offset that propagates into placement. Repeated placements can therefore accumulate a visible lean in the stack even when each individual placement satisfies the success criterion of being stably released on the platform.

5 Challenges and Lesson Learned

Periodic fixed position dynamic grasping can be surprisingly effective after careful tuning, but its failure boundaries are dictated first by kinematics and contact geometry and only second by timing. The grasp radius relative to the turntable center is not merely a design convenience for choosing where to grab, but a feasibility parameter that determines whether the required downward capture pose is solvable with comfortable joint margin on the Panda arm; validating feasibility over a neighborhood of radii, rather than selecting a single value that appears reasonable in the simulator, is essential for robust hardware behavior.

We also learned that waypointing must be justified by both kinematics and throughput. Intermediate joint space waypoints improve inverse kinematics convergence by providing consistent seeds and preventing long range configuration jumps between the stacking workspace and the turntable workspace, but each additional waypoint consumes time and reduces the number of capture opportunities per minute, which directly reduces the expected yield of a non predictive sampling policy. The practical implication is that waypoint sets should be minimal and purpose driven, ideally one ready configuration and the smallest set of transitional postures that reliably keep the solver away from joint limits and self collision while preserving cycle time.

Finally, hardware success was dominated by a small set of hidden parameters that must be tuned on the real system: the effective lateral alignment of the turntable capture locus relative to the robot base, the grasp height offset that guarantees full closure without table contact, and the matching between turntable angular velocity and gripper cycle period. When these are tuned well, the system can achieve repeated dynamic grasps and stacks without explicit prediction; when they are not, failures occur even if the perceived alignment appears correct. This is the major reason that our group did not perform well in the final competition as we had not had enough time to measure all of the offset parameters. We also gained the practical insight that non uniform block spacing makes fixed frequency policies prone to off center grasps, which can bias the stack geometry over time; a simple improvement that preserves the overall approach would be to add a brief visual or proprioceptive recenter action after acquisition, or to allow a small corrective adjustment before placement, which would improve stack verticality without requiring full trajectory prediction.