MEAM 5200 Pick and Place Challenge
Final Report

Yunpeng Wang, Jianning Cui, Yuqi Xiang, Vaibhav Wanere

May 9, 2023

1 Scope

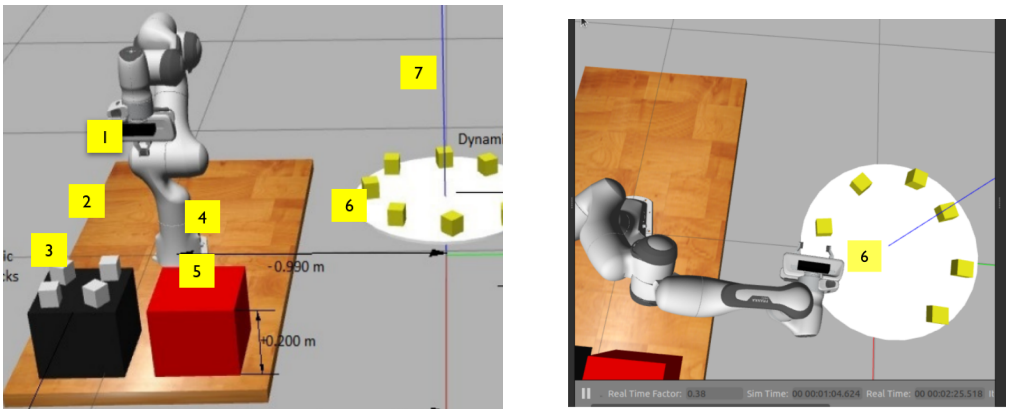
We have been assigned with a task of picking up blocks from one platform and placing them on another platform in a stacked way. The platforms to pick the block from are both static and moving. The platform to place the block is static. The blocks are provided with April Tags and thus we can use the vision system to detect and grab the block. For placing the block we have to do offline planning. The objective is to maximize the number of blocks stacked within the given time.

2 Method

We broadly divide our method to achieve the task outlined into 'grabbing the static blocks' and 'grabbing the dynamic blocks'. This is because the complexity to grab the dynamic block is more than that of grabbing the static block. Once we are successful in grabbing the block, we can use our offline planning algorithm to stack the blocks one over other on the static platform.

2.1 Path Planning

We used a few pre-defined configurations to reduce the time required of solving the IK. The corresponding positions to reach to the target are as shown in the Figure 1.



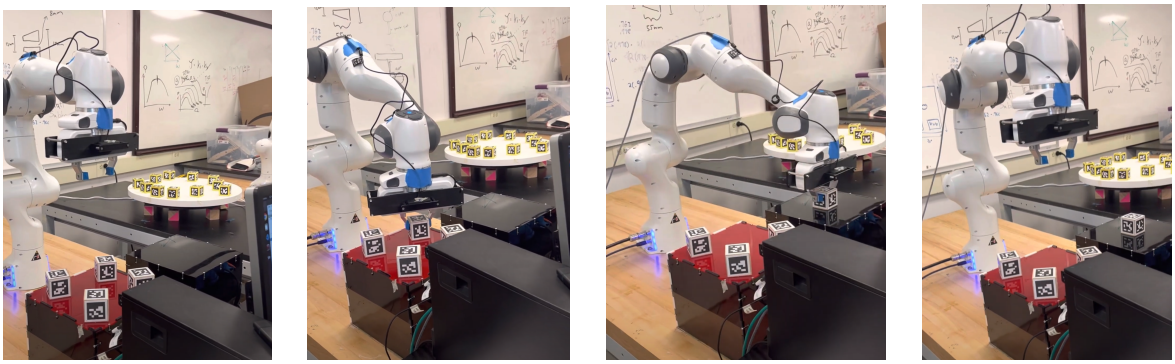
(a) Pre-defined poses: Static blocks

(b) Pre-defined poses: Dynamic blocks

Figure 1: Pre-defined poses for path planning

- First Position: neutral or starting position.
- Second Position: scanning position to scan the poses and orientations of blocks.
- Third Position: used to grab each static block; unique for grabbing each static blocks.
- Fourth Position: scanning position used to scan the height of stacked blocks at the destination platform.
- Fifth Position: stacking position to stack blocks and its height increases per block stacked
- Sixth Position: waiting position to wait for the dynamic blocks before grabbing
- Seventh Position: designed to scan the positions of dynamic blocks but is impossible to reach. So the position six is both wait and grab position.

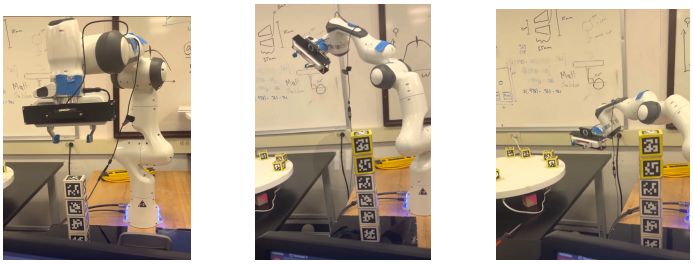
1. **Path planning for static blocks:**
   While grabbing static blocks, the end effector will follow and repeat the path: $2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 4 \rightarrow 2$.



(a) Poses 1 (b) Pose 2 (c) Pose 3 (d) Pose 4

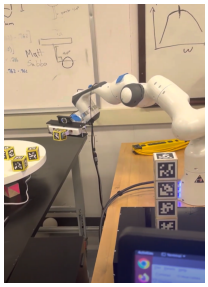
Figure 2: Poses for static block

2. **Path planning for dynamic blocks:**
   While grabbing dynamic blocks, the end effector will follow and repeat the path: $4 \rightarrow 6 \rightarrow 4 \rightarrow 5 \rightarrow 4$.
   Figure 3 and 4

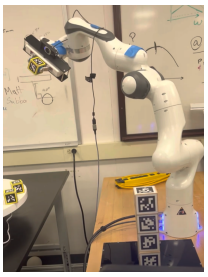


(a) Poses 1 (b) Pose 2 (c) Pose 3

Figure 3: Poses for Dynamic blocks



(a) Pose 4



(b) Pose 5

Figure 4: Poses for Dynamic blocks

2.2 Vision System

We use vision system only for the static blocks. The robot end effector is positioned at position 2, which captures all the block present on the static platform. We have been given the function to get the transformation matrix for the block with respect to the end effector camera frame i.e. H_{block}^{camera}. We need the transformation in the robot coordinate frame i.e. H_{block}^{arm}. But we know the transformation from camera frame to the end effector frame: H_{camera}^{ee}. We also know the transformation from end effector frame to the robot frame: H_{ee}^{arm} using forward kinematics for a given joint configuration. Thus, we can calculate H_{block}^{arm} as:

$$H_{block}^{arm} = H_{ee}^{arm}.H_{camera}^{ee}.H_{block}^{camera} \tag{1}$$

```
def transform_from_camera_frame_to_robot_frame(p, q, detector: ObjectDetector):
    H_ee_camera = detector.get_H_ee_camera()
    jointPositions, T0e = FK().forward(q)
    pose = T0e @ H_ee_camera @ p
    return pose
```

Figure 5: Block pose in robot frame

Thus, the robot end effector goes to position 2, scan all the blocks present on the platform. Calculate the H_{block}^{arm} matrix for each block and we save this matrices in a list. Then we iteratively go to each static block position i.e position 5, which is unique for each block as calculated from the above equation and grab the block. One thing to be noted is that we grab the leftmost and rightmost blocks first for the red and blue team respectively.

2.3 Grabbing static blocks

To grab static blocks, there are two challenges we need to deal with. The first one is aligning the orientation of the end effector with the blocks, which will be discussed in section 2.2. The second one is avoiding collision with blocks and sweeping them out of the platform.

To avoid collision with the platform when grabbing static blocks, for the red platform we grab the leftmost (in robot frame) blocks first and for the blue platform grab the rightmost blocks first (in robot frame)

1. **The geometric trick:** Even if we get the pose of the block from equation 1 in in robot coordinate frame, we cannot use of directly to solve the IK, because the xyz axes of the block coordinate frame

are not known. In other words the z-axis is not always vertically upward. The coordinate frames of the block vary with respect to the orientation of the block with respect to the robot frame. What we only know certainly is the position of block with respect to the robot coordinate frame. Which we express as the translation vector from robot frame origin to the block frame origin. Which can be obtained by extracting the last column's three elements of the matrix H_{block}^{arm}.

```python
def get_angle_along_z_axis_from_block_pose(block_pose: np.ndarray) -> np.ndarray:
    block_rotation = block_pose[0:3, 0:3]  # robot frame
    assert block_rotation.shape == (3, 3)
    v1, v2, v3 = block_rotation[:, 0], block_rotation[:, 1], block_rotation[:, 2]
    u1, u2 = filter_z_axis(v1, v2, v3)
    w = filter_odd(u1, u2)
    w = w / np.linalg.norm(w)
    angle = np.arctan2(w[1], w[0])

    assert angle >= 0 and angle <= pi / 2
    if angle>pi/4:#Jianning Cui make this change
        angle=angle-pi/2#to make this angle between -45 degrees to 45 degrees
    return angle
```

Figure 6: The Geometric Trick

To tackle this challenge, we first eliminate the axis of the block which aligns with the robot frame z-axis and use the remaining two axes two calculate the angle to align the gipper jaws with block side faces. We feed this angle to the FK solver to to align the gripper with block. Thus we eliminate the need to solve IK to reorient the gripper with the block thus saving computation time and increasing the speed of grabbing and placing the block. We call this as the geometric trick. Figure 6

2.4 Grabbing dynamic blocks

For dynamic blocks grabbing, the most important part is to calculate the correct joint position to catch the block like Figure 1b without collision with the rotating plate. The other details like intermediate joint positions (covered in Section 2.1) and the waiting and picking strategy are not covered in this part because they are relatively easy to understand and implement.

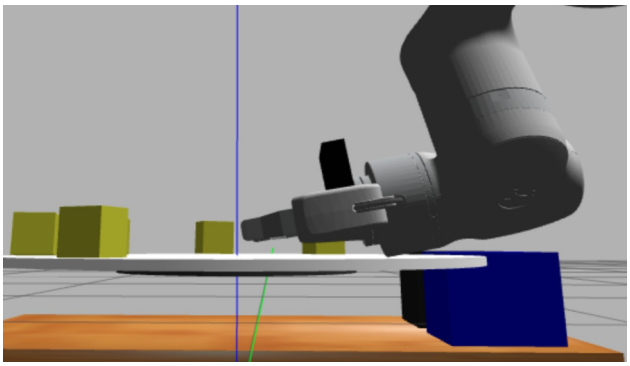


Figure 7: Waiting for Dynamic Blocks On Side View

We used an offline IK method to calculate the joint position of Panda Arm, given the desired the end effector pose. As is shown in Figure 7, we designed an 10 degrees tilting for the end effector to avoid collision when catching the dynamic block.

The main challenge of the dynamic part is the pose of the robot in Figure 1b and 7 is really close to singularity.

Therefore, the pseudoinverse method (implemented in lab3) will not work because it is hard to converge in such singular position. Instead, we introduces a Damped least squares method, which finds the value of $\Delta\boldsymbol{\theta}$ that minimizes the quantity

$$\|J\Delta\boldsymbol{\theta} - \overrightarrow{\mathbf{e}}\|^2 + \lambda^2\|\Delta\boldsymbol{\theta}\|^2$$

where J is the jacobian matrix, $\Delta\boldsymbol{\theta}$ is the offset of the joint position and $\overrightarrow{\mathbf{e}}$ is the end effector transformation vector (in 6D) and λ is the damped coefficient.

This method avoids many of the pseudoinverse method's problems with singularities because it introduces the $\lambda^2\|\Delta\boldsymbol{\theta}\|^2$ penalty to ensure that $\Delta\boldsymbol{\theta}$ will not be too large due to numerical problems. The closed-form solution for this problem is

$$\Delta\boldsymbol{\theta} = J^T\left(JJ^T + \lambda^2 I\right)^{-1}\overrightarrow{\mathbf{e}}$$

2.5 Stacking blocks

We stack blocks at the center of the target platform and use the vision system to determine the height of the positions the robot put the blocks.

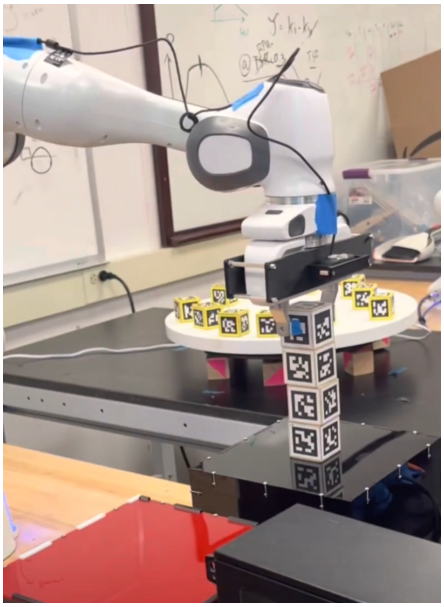


Figure 8: Stacking Static Blocks

While stacking the block one over the other, we bring the end effector to a position which ensures that the gap between the topmost block and the block being stacked is approximately 5 mm. This ensures that the block being stacked does not disturb the already stacked blocks and thus ensures safety of stacking the blocks. Refer to Figure 8.

3 Evaluation

We verify our method in both simulation and in-person labs. For dynamic blocks, only the in-person labs work because the friction in simulation doesn't match the one in the real world and makes it hard for the robot to grab a dynamic block. We've uploaded our in-person lab recording for static and dynamic block stacking and it can be found at here. The method was successful in stacking four static blocks and three dynamic blocks.

3.1 Effective Path Planning

The path $1 \rightarrow 2 \rightarrow 4 \rightarrow 6$ is followed by our robot in simulation and the recording can be found here for red platform, and here for blue platform. This tests our IK solution.

Our method is stable and fast in the in-person labs. It takes 90 seconds to grab and stack 4 static blocks and at most 4 extra dynamic blocks in the left 210 seconds. Although the wait-and-pick strategy for dynamic blocks sounds slow, it can work without the vision system and hence is not affected by the camera noise.

3.2 Axial alignment of static blocks

The blocks stacking movement can be viewed here. It can be observed that there is about 5 mm axial misalignment between the consecutively stacked blocks. But in simulation the alignment is perfectly axial and thus we assign the cause of this stacking error to two things, inherent difficulty of replicating everything from software to hardware and the vision system noise.

3.3 Use of vision system to stack the blocks

The simulation result for end effector rotation is uploaded here. The vision system helps the robot detect the position and orientation of the static blocks. If we only utilize the position of the static blocks, then the stacking cannot put one block perfectly on another block.

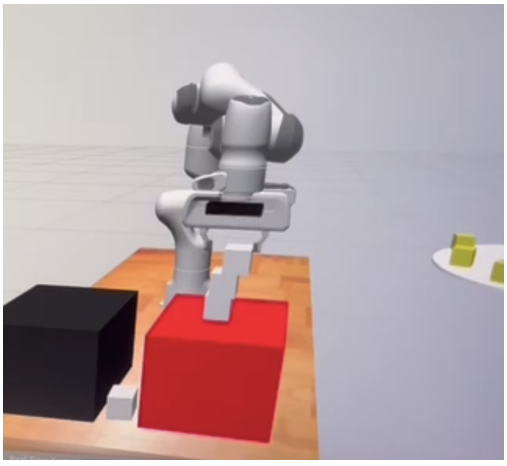


Figure 9: Leaning Stacking

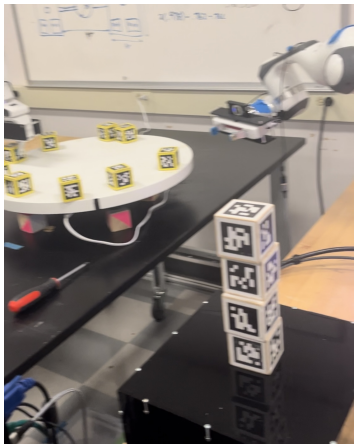


Figure 10: Straight Stacking

The effect of stacking such a leaning tower is shown in Figure 9. But if we take into consideration the orientation of static blocks, then a relatively straight tower could be achieved and is shown in Figure 10.

3.4 Grabbing dynamic blocks

We have uploaded the video of the one shot success of the wait-and-pick strategy for the dynamic blocks. here. To prevent the camera noise from affecting the grabbing dynamic blocks, we use the wait-and-pick strategy, which proved to work pretty well in the final competition. We've seen some teams cannot locate the dynamic blocks with the vision systems accurately so their robot cannot grab the dynamic blocks and even collide with the turntable.

We had anticipated such challenges and since we we successful in using the wait and grab strategy and given the limited time to optimize the strategy, we went ahead with the same.

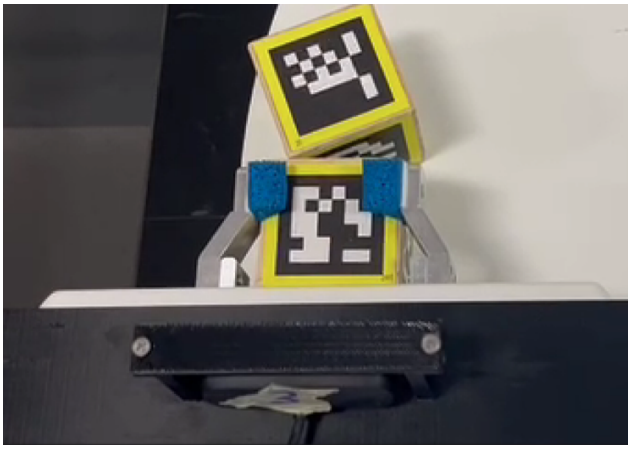


Figure 11: Grabbing Dynamic Blocks

We do both the simulation and in-person labs to verify our methods and find that the waiting position and orientation are important. The position and orientation from the top and side view are shown in Figure 1b and 7. We tilt the end effector 10 degrees in the waiting position to avoid collision with the turntable.

Wait until grabbed strategy:

The gripper will use the open and close loop to grab dynamic blocks until it senses the blocks. Sensing the blocks could be done in two ways. The first one is using the force sensor on the gripper to decide if it grabs something, while the second one is using the distance between the jaws of the grippers, to detect if there is anything between the grippers after closing. We finally decided to use the second one because the force feedback may be unstable and may not be accurate. Figure 11 shows how the robot grip a dynamic block. Using this strategy we have ensured that the end effector will not move until and unless it has grabbed the dynamic block. The wait time was set to 3 seconds and we found that this approximation proved practically helpful.

```
while dis < 0.01:
    gripper_state = arm.get_gripper_state()['position']
    print('gripper_state', gripper_state)

    arm.exec_gripper_cmd(0.12)
    time.sleep(5)
    arm.exec_gripper_cmd(0.049, force=100)
    dis = np.abs(arm.get_gripper_state()['position']).mean()

arm.safe_move_to_position(q_source)
```

Figure 12: While loop for wait until grab strategy

We use a while loop Figure 12 that ensures that if the closed gripper position is less than 10 mm it

will open again until the closed gripper position becomes 49 mm, which indicates that the block has been grabbed.

$$Block_{not\_grabbed} = 10\ mm \tag{2}$$

$$Block_{grabbed} = 49\ mm \tag{3}$$

3.5 Stacking blocks

The stacking process is contained in the whole process but we still have a bit different video. We were able to successfully stack four static blocks and three dynamic blocks. Figure 13

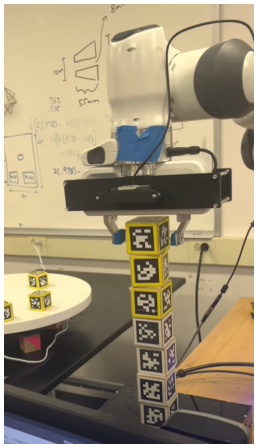


Figure 13: Stacking seven blocks in 5 min

4 Analysis

1. **Simulation results:**

   For static blocks we obtained perfect results in simulations. All the static blocks were perfectly stacked over each other vertically. This has two reasons, first the simulation environment itself and the absence of noise from the vision system. When we compared the simulation results with the robot lab hardware test, we noticed that the blocks were not being stacked as seen in the simulation. The blocks were being stacked offset with respect to each other. Also, there was not a fixed pattern in stacking the block and there was a randomness in the process. We thought of quantifying the uncertainty in the stacking process, but given the limited time we couldn't complete our idea. Thus we relied on uncertainty for our success. After some trials we gained the confidence that even with the uncertainty we can stack almost seven blocks.

2. **Results on Hardware:**

   One thing which deteriorated the performance was the error in estimating the position of stacked blocks. This posed challenges is stacking the blocks, we had kept a safety margin of 5 mm while stacking the block. To our surprise the robot arm was estimating the block pose incorrectly and hence the safety margin was violated and the block was not stacked from a height of 5 mm. Instead it used to touch the block on which it was being stacked. This posed threat to already stacked blocks. Sometimes, the block was stacked from a height much larger that 5 mm, say 20-30 mm, which created the disturbance in the already stacked blocks.

3. **During Competition:**

   Though we won the competition, we observed a strange thing that the performance of our controller was not as good as we observed during the lab sessions. When we thought about this, we realise that the lighting conditions in the robot lab and that of the competition were very much different from each other. This might have caused the noise in the vision system and might have lead to the error in detecting the pose of the blocks. If we would have got more time, we could have verified our hypothesis.

4. **What could we have done better?**

   We think that, instead of offline planning for the dynamic blocks, we could have used the vision system. After fine tuning, we would have been able to perfectly grab the blocks. Also, instead of hard-coding the dimensions of the block platform, we can just use the April tags for the platforms so that even if the platform moves by some distance, we won't have to change the code. This way the code can become more modular. I think this is important because as we saw during competition, we had to take care of the height of the platform as well as the distance of the platform from the robot frame. If there is slight change in the height or the distance, out controller cannot reflect those changes and hence the error can propagate to a considerable magnitude. This may lead to improper stacking of blocks and even collision of the end effector with the table or surrounding.

5 Lessons Learned

1. **Correct inputs:** We have realised that many of the times our code didn't work was because we were not reflecting the mathematical equations perfectly into the code. Sometimes we made mistakes in the matrix multiplication sequence and sometimes we provided the wrong input dimensions to the code because of which the output was surprisingly different and forced us to debug our work.

2. **Correct understanding of the April Tags:** We didn't think about the pose of the blocks with respect to robot frame and with our super simple assumptions we went ahead and implemented the code. To our surprise the robot arm behave weirdly in the simulations. After carefully observing the behaviour in the simulation, we understood the underlying pattern and realised that our assumption was wrong. We didn't have the understanding of how April Tags work and hence we faced such challenge.

3. **Labs so far:** The lessons learnt in previous labs were very helpful for the final project competition and we realised how the things from lab0 to lab3 are built so that we can successfully tackle the pick and place challenge.

4. **Future Scope:** If we have had the opportunity to spend more time on the task, we could have tried to use the vision system for the picking the dynamic block. Also, we could have tried to quantify the noise in the vision system to make more accurate predictions of the block pose in the robot frame.

Acknowledgements We thank Prof Nadia for such a wonderful opportunity and course and all the TAs for their support throughout the semester, without which this project would have been impossible. We thank you for giving us the opportunity for competition and we are grateful to win the competition.