# MECE E4602 Final Project Report: “Grab-and-Go” Robot

Mai Cai[1], Shiksha Sabharwal[1], Yuan Yang[1], Zhuoqun Wu[1]

[1] Department of Mechanical Engineering, School of Engineering and Applied Sciences, Columbia University, NY, USA

***Abstract*—Grocery shopping is common for everybody in their daily lives, it could be satisfying or redundant for some depending on their shopping experiences. Oftentimes, items are placed on higher shelves that may be too heavy or unreachable for customers and require human assistance. Human assistance is necessary but can be minimized by using a robotic arm that fetches items to customers’ carts.**

## I. INTRODUCTION

The main purpose of this study is to build a retail store robot arm that fetches items and drops them in customers’ shopping carts. With the implementation of this robotics application, it can be extremely convenient and helpful for old and disabled people or those who don’t want to wait for help from an assistant.

Past research has confirmed the use of transfer robots in large environments like warehouses as they are efficient in moving or arranging goods for storage management.[1] However, such high demand for space is not suitable for grabbing small items with different packaging shapes.[2] Due to the limitation of specificity, warehouse robots are not suitable enough to deal with groceries transportation in common retail stores.

Unlike some prototypes of smart shopping carts that are expensive and unreliable, the “Grab-and-Go” robot arm slides along the aisle which increases mobility while saving space and cost. The dynamic structure of the arm involving 2 revolute and 3 prismatic joints enables it to move in a 5-DOF workspace enabling better orientation and reach.

## II. SYSTEM DESIGN

Our solution is a flexible robot arm with 3-DOFs for translational motion, controlled by the cable of one main frame and two prismatic joints, and 2-DOFs for rotational motion, controlled by motors at the connection of two revolute joints. (Fig. 1).

Unlike vending machines pushing a water bottle or a bag of snacks and letting it fall from shelves, the “Grab-and-Go” robot can grab any grocery product, including crushable items. With our robot arm programmed, only by pressing the pad of selected sections on the shelf, the end-effector fetches things from shelves and places them in your shopping cart directly in a swift and gentle manner.

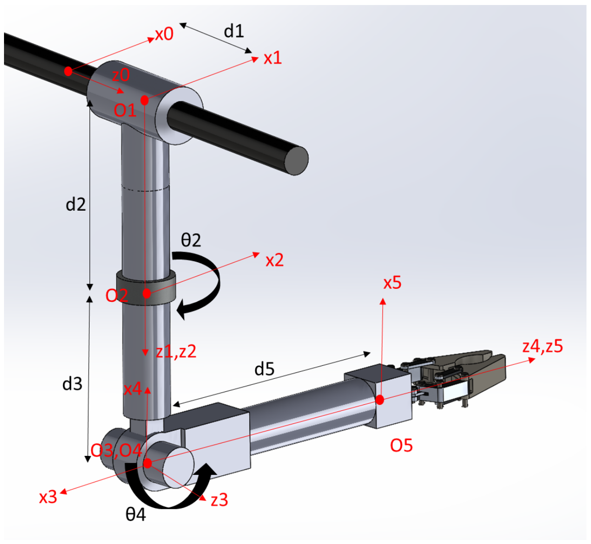

Fig. 1. 3D model of “Grab-and-Go” Robot with labels

Table 1. D-H Table for the five joints

| | $a_i$ | $\alpha_i$ | $d_i$ | θi |
|---|---|---|---|---|
| 1 | 0 | 90 | $d_1$* | 0 |
| 2 | 0 | 0 | $d_2$ | $\theta_2$* |
| 3 | 0 | 90 | $d_3$* | 180 |
| 4 | 0 | 90 | 0 | $\theta_4$* |
| 5 | 0 | 0 | $d_5$* | 0 |

## III. SYSTEM ANALYSIS

### A. Jacobian Analysis

The Jacobian matrix (J) can be calculated with forward kinematics:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} 0 & s2^* s4^* d5^* & 0 & -c2^* c4^* d5^* & -c2^* s4^* \\ 0 & 0 & -1 & -s4^* d5^* (c2^*)^2 - s4^* d5^* (s2^*)^2 & c4^* \\ 1 & -c2^* s4^* d5^* & 0 & -c4^* s2^* d5^* & -s2^* s4^* \\ 0 & 0 & 0 & -s2^* & 0 \\ 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & c2^* & 0 \end{bmatrix}$$

### B. Singularity Analysis

Since the Jacobian matrix is non-square, the singularities can't be found via Det (J). Instead, the existence of singularities can also be confirmed if the Jacobian matrix can lose its rank. By plugging in specific values of joint variables ($\theta_2$ = -90° and $\theta_4$ = 90°), column 1 was identical to column 5, and the matrix now has a rank of 4. Therefore, there is at least one singularity in this configuration. The code for the calculation of Jacobian and singularities can be found here.

*C. Inverse Kinematics*

To find the joint variable positions, inverse kinematics come into the picture. It is defined as finding joint variables in terms of end-effector position and orientation.[3] This explains all the positions where the robotic arm can reach and fetch the retail store's items to the customers' shopping cart. However, the revolute joints may not be able to have a complete 360° rotation due to workspace constraints and object avoidance in the path which can restrict the scope of the arm. The inverse kinematics can be calculated as:

$$d_1 = \frac{d_2}{tan\Theta_2} - Z_0^5\ ;\ d_3 = d_5 sin\Theta_4 - d_2 - X_0^5;\ d_5 = \frac{Z_0^5 + d_1}{cos\Theta_4} \qquad (1)\sim(3)$$

$$\theta_2 = tan^{-1}\left(\frac{d_2}{Z_0^5 + d_1}\right) - Z_0^5\ ;\ \theta_4 = cos^{-1}\left(\frac{Z_0^5 + d_1}{d_5}\right) \qquad (4)\sim(5)$$

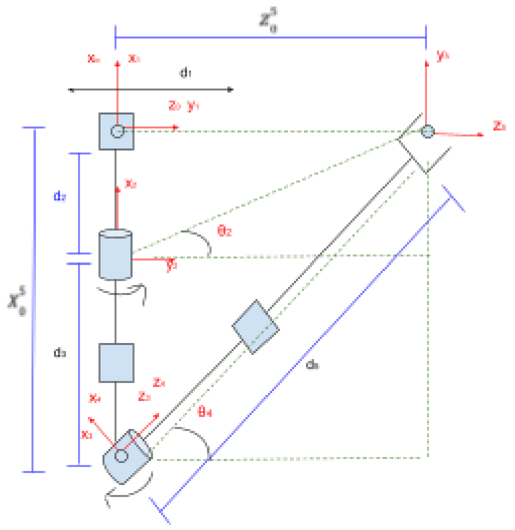

Fig. 2. Inverse kinematics of 5-DOF manipulator

*D. Quintic Polynomial Trajectories*

Planning trajectories with the cubic polynomial method can get continuous positions and velocities but discontinuities in the acceleration. Therefore, the quintic polynomial method is used instead to avoid acceleration discontinuities.

According to the equations in [3], parameters $a_0$~$a_5$ can be determined:

$$a_0 = q_0;\ a_1 = 0;\ a_2 = 0$$

$$a_3 = 10(q_f - q_0);\ a_4 = -15(q_f - q_0);\ a_5 = 6(q_f - q_0)$$

Finally, the trajectory equations can be obtained:

$$q_i(t) = q_0 + 10(q_f - q_0)t^3 - 15(q_f - q_0)t^4 + 6(q_f - q_0)t^5 \quad (6)$$

$$\dot{q}_i(t) = 30(q_f - q_0)t^2 - 60(q_f - q_0)t^3 + 30(q_f - q_0)t^4 \quad (7)$$

$$\ddot{q}_i(t) = 60(q_f - q_0)t - 180(q_f - q_0)t^2 + 120(q_f - q_0)t^3 \quad (8)$$

Where:

$q_0, q_f$= Initial & Final position;
$\dot{q}_0, \dot{q}_f$= Initial & Final position;
$\ddot{q}_0, \ddot{q}_f$= Initial & Final acceleration

*E. Path Planning Analysis*

Some assumptions are made for Matlab Trajectory Simulation. First, the first prismatic joint was assumed to be moved to the target position already by placing an order through the pad. Hence, the things that need to be determined are analyzed motions from the 2nd to the 5th joints (revolute, prismatic, revolute, and prismatic joints) of the robot.

The Parameter is determined as well. The robot arm starts and ends at rest, meaning the initial and final velocities are 0. The initial time is considered as t0, which is 0. There are 4 phases in total, and the time for each phase to process is 2 seconds. The coordination of the end effector and move for each phase is worked out and shown in Fig. 3. To confirm the feasibility of the robot movement, the paths were simulated with the Robotics Toolbox in MATLAB.[4]

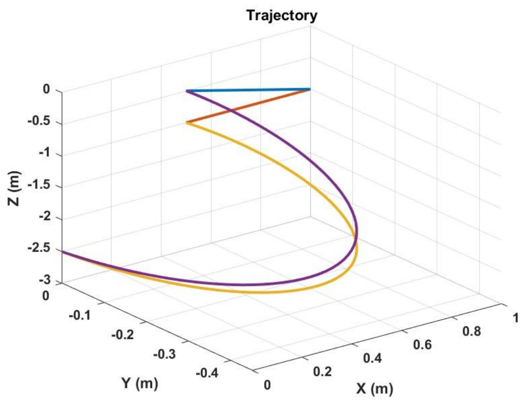

Fig. 3. Screenshot of path simulation in MATLAB

Phase 1 (the blue path) describes how the robot reaches the product from the start point. Then, to avoid obstacles, phase 2 (the red path) is set up to describe how the robot grabs and backs out from the shelf, in case of hitting other products. After that, phase 3 (the purple path) shows how the robot reaches the shopping cart and puts the products into the cart. Finally, the robot goes back to the start point and ends the process in phase 4 (the yellow path). According to this process, the coordinates of 4 designated points are determined as (0.5, 0, -0.5), (0.5, 0, -1), (0, 0, -2.5), and (0.5, 0, -0.5).

*F. Joint Trajectory Analysis*

Based on the planned paths, the motions of joints 2 to 5 are analyzed, and a cycle of their trajectory plots through 4 phases was made (Fig. 4). Note that the x-axis represents the time, and the robot starts and pauses at rest, meaning that the initial and final velocities are 0.

Compared with cubic polynomial trajectories, quintic polynomial trajectories avoid an "impulsive Jerk" and vibrational modes. In other words, this method allows more smooth and continuous motion, precise and accurate control, and flexibility in defining the trajectory. Fig. 4 shows the result straightforwardly.

It is also noticeable that the revolute joints (joint 2 and joint 4), stay still till the beginning of phase 3, since phases 1 and 2 only take place in the 2D X-O-Z plane, as shown in Fig. 3. As a comparison, the prismatic joints (joint 3 and joint 5) pretty much keep moving during the whole cycle.

The code for the quintic polynomial trajectories and joint trajectory plotting can be found here.

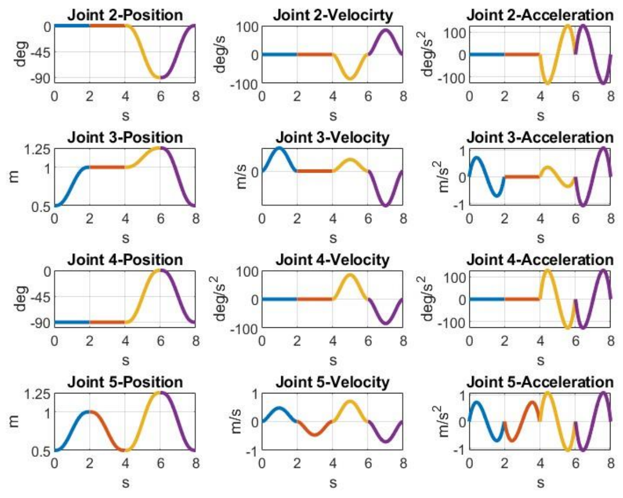

Fig. 4. Position, velocity, and acceleration plots of joints 2-5

## G. *Workspace Analysis*

As mentioned before, joint 1 was assumed to be moved to the target position already. For joint 2 to joint 5, their range of activities is limited to the following:

Prismatic joints (joint 2 and joint 4): [0m ~ 2m]

Revolute joints (joint 3 and joint 5): [-90° ~ 90°]

With the joint restrictions defined above, a workspace simulation was generated in MATLAB. (Fig. 5). The blue dots represent the workspace that the robot can reach in 3D space.

The code for path planning analysis and workspace simulation can be found here.

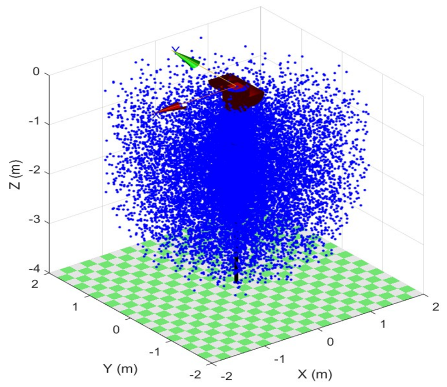

Fig. 5. Screenshot of workspace simulation in MATLAB

## IV. Discussion

Our "Grab-and-Go" robot arm should be capable of moving groceries from retail store shelves directly to the shopping cart. The robot arm can fetch things in high or unreachable places for customers swiftly and efficiently. With DH-tables and robotic architecture shown, the hardware specifications of our design are illustrated. With forward and inverse kinematics calculated as well as the jacobian matrix, we found how the angle and length variables of each five joints affect the position of our end-effector. With path planning, trajectory, and workspace analysis determined on MATLAB, the results show us the "Grab-and-Go" robot is suitable for a retail shelves environment, compared to other large and space-consuming warehouse robots. However, there are a few aspects of limitations so far:

1) The current end-effector may not be able to handle different types of groceries or packaging based on size and weight and may fail to pick up or place in one piece in the shopping cart.

2) There are a few workspace constraints that explain why our robotic arm may not be able to rotate a full 360° due to object avoidance in the path.

3) The robot can only fetch an item in certain areas divided by the control pad system. The current robot only addresses one customer at a time.

## V. Conclusion

Through this study, we learned how concepts like path planning and joint trajectory can help solve problems in the real world. In the process of determining the workspace and trajectories of our robot to fetch items, we reviewed what we learned in class and put them into practice.

For future extensions:

1) Adding a switchable tool for the end-effector based on force analysis, a bar code scanner for every item, and a sensor camera to recognize item types from the shelves to increase efficiency and accuracy is our main optimization.

2) Furthermore, the function of picking up groceries automatically through an app before the customer arrives is also considered to improve our retail store robot.

3) Adding the feature of putting the not-required item back in place. There is a possibility that the customer who ordered an item decides not to go ahead with it, should be able to drop it into a section available on the shelf from which the robot shall pick it up and put it back in its position by recognizing its position from the bar code present on it.

4) Additionally, this concept can be applied to all retail stores where the end-effector's shape can be determined as per the application, and having a well-tailored design of our first prismatic joint which is attached to existing shelves will help automate the stores in no time.

## VI. Citations

[1] Liang, C., et al. "Automated robot picking system for e-commerce fulfillment warehouse application." *The 14th IFToMM World Congress*. 2015.DOI: 10.6567/IFToMM.14TH.WC.OS13.077

[2] Laber, Joshua, Ravindra Thamma, and E. Daniel Kirby. "The impact of warehouse automation in amazon's success." *Int. J. Innov. Sci. Eng. Technol* 7 (2020): 63-70.

[3] Mark W. Spong, Seth Hutchinson, and M. Vidyasagar. "*Robot Modeling and Control"; John Wiley & Sons, Inc.,* Third Edition.

[4] "Robotics toolbox," Peter Corke, 27-Apr-2020. [Online]. Available: https://petercorke.com/toolboxes/robotics-toolbox/. [Accessed: 13-Dec-2022].