SmartShopper: CS449 Final Report

Berkay Yalman
Department of Electrical and Electronics Engineering
Bilkent University
Ankara, Turkey
berkay.yalman@ug.bilkent.edu.tr

Hakan Kara
Department of Electrical and Electronics Engineering
Bilkent University
Ankara, Turkey
hakan.kara@ug.bilkent.edu.tr

Yiğit Türkmen
Department of Electrical and Electronics Engineering
Bilkent University
Ankara, Turkey
yigit.turkmen@ug.bilkent.edu.tr

***Abstract*—This project aims to develop a robotic system that is capable of collecting and bringing shopping items with variety of shapes. The projects integrates concepts such as robotic manipulation, path planning, and computer vision to tackle the objective.**

***Index Terms*—robotic manipulation, path planning, computer vision, autonomous robotics, adaptive grasping**

I. Introduction

Robotic pick-and-place systems are crucial for applications in homes, retail, and industry, but they often struggle with unstructured environments and diverse object shapes. Building on recent work like ShelveBot [1], we propose a mobile manipulator capable of identifying, grasping, and placing three differently shaped objects into a basket. Unlike ShelveBot, which organizes items on shelves, our system will tackle object diversity and confined placement in a basket, addressing challenges in perception, adaptive grasping, and precise motion planning. This project aims to showcase a versatile robotic solution for dynamic, human-centered spaces.

II. Methodology

Three main aspects of this project are motion planning, grasping objects with varying shapes and perception.

A. Simulation Environment

Simulation environment is based on the Homework 2 Question 2. However, books in the environment are replaced with two objects that represents shopping items with different shapes. One of the shapes is a water bottle, and the second one is a chips packet. Third shape is a plate. Also there are four different shelves for our environment. A green rectangle box is added to the cargo-bot that is fixed to its base (child of the base). Objects are placed in different shelves. The blue target area is created in the middle range, away from the original position of the robot. To simulate the process physically, we used BOTOP from the RAI. See figure 1 for the simulation environment. And figure 2, 3 and 4 shows that the object in the shelves.

B. Motion Planning

The motion planning task of the project is divided into two distinct phases: moving the robot to grasp an object from a shelf and placing it into a basket (Phase 1) and subsequently retrieving the object from the basket and placing it in a designated goal area (Phase 2). The entire process utilizes the KOMO framework to compute optimal paths and ensure smooth transitions.

In Phase 1, the robot first calculates a path to the location of the object using ”movement solver”, which sets objectives for positional alignment with a predefined way point while minimizing unnecessary movements. Once the robot reaches the object’s way point, the ”grasping solver” generates a collision-free trajectory to align the gripper with the object for grasping. The solver incorporates parameters such as approach direction, grasping axis, and pre-alignment distance to ensure the gripper’s proper positioning and orientation. After grasping, the robot reverses its trajectory to its initial state and transitions to an intermediate way point above the basket (basket enter point) using IKsolver. Bottles are placed at the center of the basket for stability, while plates and chip bags are placed near the edges to facilitate easier retrieval during Phase 2. This strategic placement reduces the risk of collision or obstruction when retrieving objects from the basket. You can see figure 5, 6 and 7).

Phase 2 focuses on retrieving the object from the basket and placing it at the goal location.(Figure 8,9 and 10) The robot uses ”grasping solver” to calculate a path to align the gripper with the object inside the basket, employing similar optimization techniques to those used in Phase 1. Once the object is grasped, the robot moves to the goal area and calculates a placing trajectory using ”placing solver” to ensure precise placement. Throughout both phases, the ”play path” and ”play path reverse” functions simulate the execution of calculated paths in the robotic environment. For real-time validation, the BotOp interface synchronizes the robot’s movements with the simulation, ensuring that the calculated paths are physically feasible and executed accurately.

C. Grasping Objects with Varying Shapes

Our updated grasp-planning pipeline identifies stable grasp points by focusing on opposing surface normals and suitable grasp widths.(See figure 13 and 14) Specifically, we employ the function `find_antipodal_grasping_points(...)` to filter, compare, and score point pairs in the following steps:

1) **Normal-Alignment Filtering:** We first select those points in the object's point cloud whose normals closely align with a chosen grasp axis (e.g., the robot's gripper axis). Mathematically, this requires

$$|\mathbf{n}_i \cdot \mathbf{g}| \geq 0.95,$$

where $\mathbf{n}_i$ is a point's normal and $\mathbf{g}$ is the grasping-axis unit vector. We then split these points into a "positive" side (normals in the same direction as $\mathbf{g}$) and a "negative" side (normals in the opposite direction).

2) **Antipodal Scoring:** Next, we form all pairwise combinations of points between the positive and negative sets. For each candidate pair, the code computes:
   - A *distance* measure to ensure the two contact points are neither too close nor too far, given user-defined `grasping_distance_min` and `grasping_distance_max`.
   - An *antipodality* term based on how opposite the normals are $\left(|\mathbf{n}_+ \cdot \mathbf{n}_-| \approx 1\right)$.
   - A *parallelity* score indicating how well the vector between the two points aligns with the chosen grasp axis.
3) **Combined Criterion and Ranking:** We multiply these metrics—distance feasibility, antipodal match, and parallelity—into a single `total_criteria` value. The function then extracts either the top-k pairs with the highest scores or (optionally) an "average" pair if `return_average = True`. This yields robust grasp points best suited for pinching or encompassing the object from opposing sides.
4) **Integration with Planning:** Finally, the selected contact points (or the averaged pair) inform the end-effector pose and gripper width. We feed this into our path-optimization module (via KOMO and NLP solver) to generate a collision-free trajectory for the actual grasp. This approach fosters stable, adaptive grasping across diverse shapes—such as bottles, plates, and flexible chip bags—by emphasizing physically meaningful contact geometry.

D. Perception

In the perception module of our robotics project, we utilize a dual-camera system for every object to capture synchronized RGB and depth images from "camera1" and "camera2." A pre-trained Mask R-CNN model accurately detects and segments objects within the RGB images, generating binary masks based on a confidence threshold of 0.5 [2]. These masks are applied to the corresponding depth images to extract precise point clouds [3], which are then transformed from camera coordinates to world space using the cameras' positional and rotational data. To enhance the quality of the perceived data, we compute surface normals from the point clouds, providing essential geometric information for better object recognition and environmental understanding. Additionally, voxel down-sampling with a voxel size of 0.006 meters is performed using Open3D to reduce the point count while preserving structural integrity. A filtering process further refines the data by discarding outlier points beyond a 0.3-meter threshold from the mean point, ensuring the reliability and accuracy of the perceived information. This streamlined perception pipeline enables the robot to effectively recognize, localize, and understand objects within its environment, facilitating robust navigation and interaction. See figures 15 to 21 for this part.

III. Results

For the all Results, refer to the demo video.

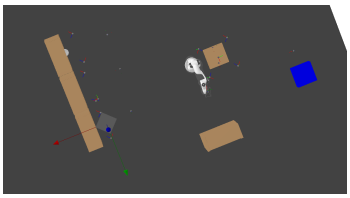

Fig. 1. Environment

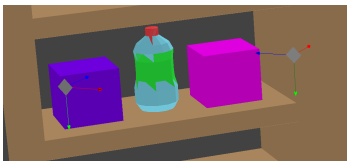

Fig. 2. Bottle in the shelve

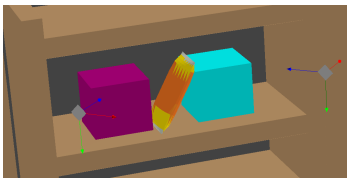

Fig. 3. Chips in the shelve

References

[1] J. Schneider, S. Pardis, and A. Yang, "ShelveBot: Fast Pick-and-Place in Human Form Factors," Massachusetts Institute of Technology, Dept. of Electrical Engineering and Computer Science, 2023. https://github.com/jschneider03/ShelveBot

[2] K. Shankar, M. Tjersland, J. Ma, K. Stone, and M. Bajracharya, "A Learned Stereo Depth System for Robotic Manipulation in Homes," IEEE Robotics and Automation Letters, vol. 7, no. 2, pp. 2305–2312, 2022. doi:10.1109/lra.2022.3143895

[3] M. Bajracharya et al., "Demonstrating Mobile Manipulation in the Wild: A Metrics-Driven Approach," in Robotics: Science and Systems XIX, Robotics: Science and Systems Foundation, Jul. 2023. doi: 10.15607/RSS.2023.XIX.055.

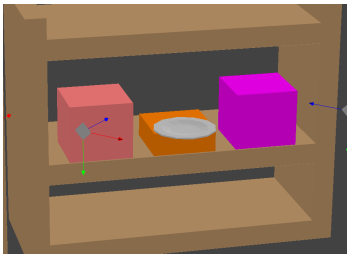

Fig. 4. Bowl in the shelve

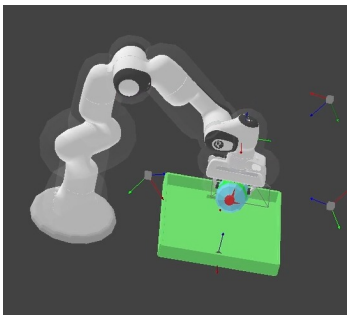

Fig. 5. Water bottle at the middle of basket

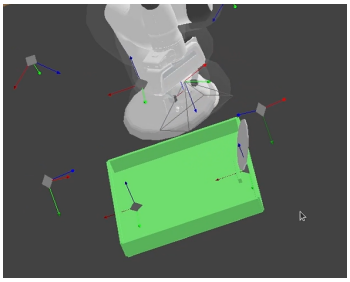

Fig. 6. Plate at the edge of basket

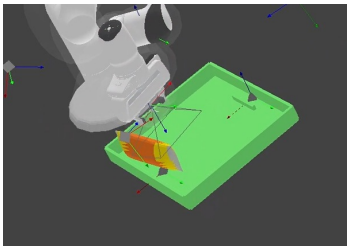

Fig. 7. Chip box at the edge of basket

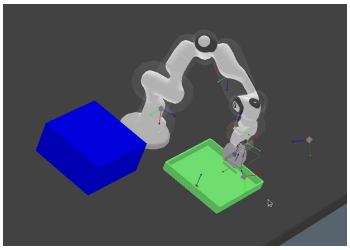

Fig. 8. Phase 2: Picking plate from basket to target area

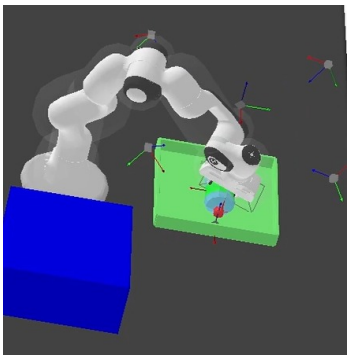

Fig. 9. Phase 2: Picking water bottle from basket to target area

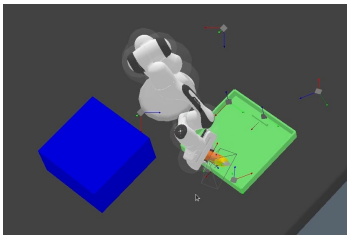

Fig. 10. Phase 2: Picking chips box from basket to target area

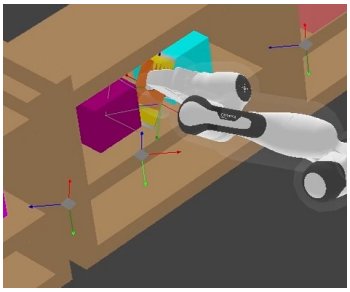

Fig. 11. Phase 1: Picking from shelves

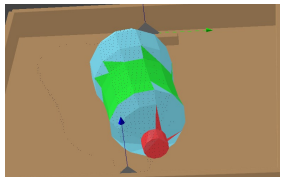

Fig. 12. Point cloud for the bottle in the basket

Fig. 13. Best Pair

Fig. 14. Best grasping point in the basket

Fig. 15. Camera 1 Vision With Masking

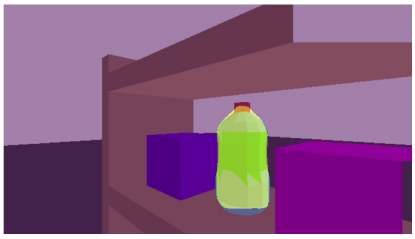

Fig. 16. Camera 2 Vision With Masking

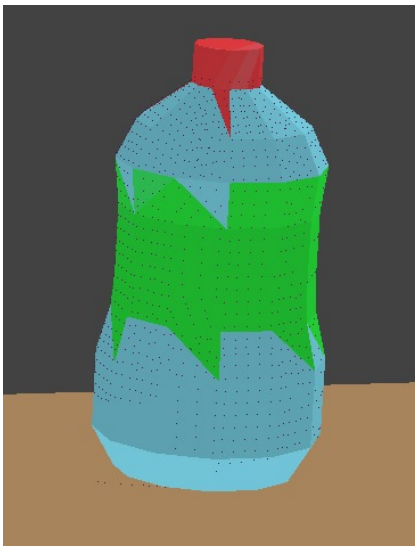

Fig. 17. Point Cloud for the bottle

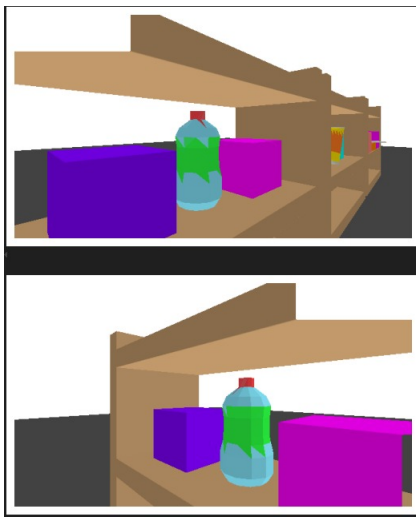

Fig. 18. Cam Results Without Masking

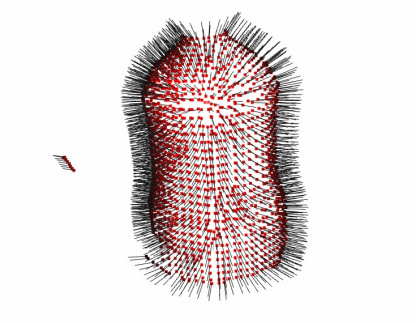

Fig. 19. Surface Normals

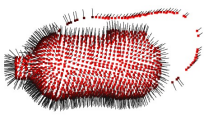

Fig. 20. Surface Normals in the basket

Fig. 21. Masking in the basket