ShelveBot: Fast Pick-and-Place in Human Form Factors

Julianna Schneider
MIT EECS Dept.
Massachusetts Institute of Technology
Cambridge, MA
juliaes@mit.edu

Shayan Pardis
MIT EECS Dept.
Massachusetts Institute of Technology
Cambridge, MA
shayanp@mit.edu

Anna Yang
MIT EECS Dept.
Massachusetts Institute of Technology
Cambridge, MA
ajyang@mit.edu

***Abstract*—Recent years have brought a renewed research interest in dexterous robotic manipulation capable of navigating space within human form factors, such as kitchens [1] and playrooms [2]. Current industry-standard robotics applications fall short of exploiting these recent advances, often fitting production lines to large robotic form factors or omitting them entirely due to the associated overhead. Enhanced robotic control will enable the integration of mobile base manipulators into existing pick-and-place-based task spaces currently inhabited by humans, expanding their viable real-world applications. To this end, we present Shelve-bot: a bimanual, mobile base manipulator that exhibits efficient motion, blending sampling-based planning and optimization to achieve for fast reorganization of objects within the human form factor of a bookshelf-style storage structure. We verify our results in simulation across a variety of shelve heights and locations.**

I. Introduction

To successfully complete an object reorganization task in a bookshelf-style storage devices of human form factor, a robotic system must be able to navigate its environment, grasp the object of interest, and perceive its surroundings. Performant solutions to the latter two challenges exist; therefore, we focus our contributions in these areas to effective implementation, making use of the antipodal heuristics for grasping and privileged information from our simulator for perception. For navigation, we draw inspiration from the way that humans blend the use of arms and feet in completing similar tasks. In practice, navigation is often sectioned into navigating the mobile base and navigating the manipulator arm, each of which is planned for independently and executed sequentially. Though functional, this separation precludes full-body solutions that leverage the unique kinematic advantages of a bimanual manipulator. Instead, we treat our mobile base as an additional three joints and plan over the entirety of our manipulator.

II. Related Works

A. Motion Planning

Optimization Optimization-based approaches to motion planning formulate the problem of navigating from one location to another as a cost function to minimize while respecting a set of constraints. Extensive prior work exists in the area of applying optimization-based approaches to solving navigation tasks. "Demonstrating Mobile Manipulation in the Wild: A Metrics-Driven Approach" [3] provides a prime example. In this paper, Bajracharya et. al present a general-purpose mobile manipulator system evaluated on a grocery shopping task. An optimization-based inverse kinematics solver is applied to find optimal paths from a set of pre-computed offline paths to viable real-world solutions at runtime. Similarly, in [7] Stavridis et. al define a set of kinematic constraints then relate them to one another in a priority hierarchy to create a robust bimanual pick-and-place system.

Sampling-Based Planning Sampling-based planners explore a set of viable motion plans in search of one that reaches the goal state, offering an alternative to optimization that is less susceptible to local minima. Prior work in the application sampling-based planning through Rapidly-exploring Random Trees (RRTs) to bimanual tasks dates back to Kuffner et. al in 2001 [5]. Recent work, such as Wu et. al's RRT variant, R3T, which demonstrates asymptotic optimality among other advances, provides a testament continued interest and fruitful work in the extension of this approach.

Recent work [8] has pushed the frontier on how best to reap the benefits of both optimization and sampling-based approaches to planning. In pursuit of a similar advantage, we leverage both in a hierarchical approach to motion planning as detailed in Section III-B.

B. Grasping

Antipodal grasping is a prevalent heuristic used to find feasible grasps finding points with opposite normal vectors, maximizing the contact wrench [11]. In "Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics" [13], Mahler et. el explore approaches to obtaining robust grasps, making use of the antipodal heuristic to identify the candidate grasps that they evaluate. Their proposed grasp planner, which samples antipodal grasp candidates and evaluates them using a GQ-CNN, boasts a 99% accuracy rate. We leverage this state-of-the-art heuristic in our system as well.

C. Perception

Point Cloud Representation Point clouds are a widely used method for representing 3D geometry. For instance, in [9] the authors discuss their implementation of a robotic manipulator for use in human environments, such as peoples' homes. They emphasize the importance of accurate point clouds in enabling the robot to manipulate unfamiliar objects and navigate new environments without collisions. Similarly, the mobile manipulator presented in [3] also utilizes point clouds to achieve a similar end goal. Since we face a similar need to represent the 3D geometry of our grasp objects, we also opted to use the point-cloud representation, but we utilized privileged perception to obtain them in order to fit our project to the scope of the course.

III. Methodology

Three main components make up our system: Motion Planning, Grasping, and Perception. Designed in alignment with symbolic programming principles, these modules can be easily combined and re-ordered to form high-level plans, which are manually specified offline. Section III-A describes our simulation environment while Sections III-B through III-D provide details on Motion Planning, Grasping, and Perception, respectively.

A. Simulation Environment

Using Drake [10], we designed two simulation environments: one for task completion and one for gathering point cloud data. The former simulation environment is comprised of shelves, objects of interest, and a PR2 robot. The shelves are located in all four cardinal directions surrounding the origin, each at a distance of four meters. PR2 initializes at the origin along the $+x$ axis. Each shelf is has three floors on which objects can be located: each shelf floor is 0.5 meters tall and 0.4 meters deep. The latter simulation environment is comprised six cameras, one along each face of the box enclosing our object of interest, and the object itself. The cameras are made invisible to simplify the process of capturing point cloud data on the object of interest.

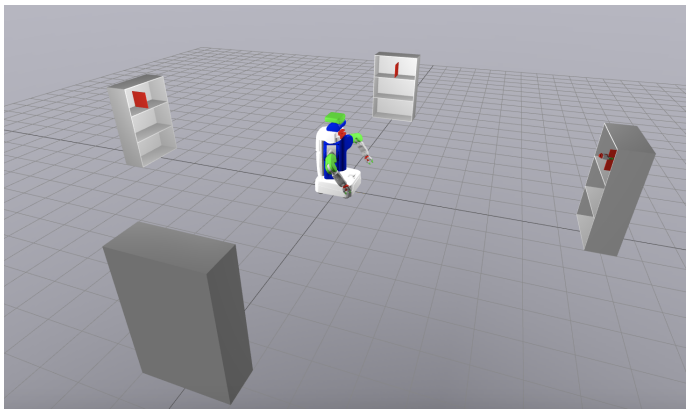

Fig. 1. Our simulation environment, consisting of the PR2, shelves, and books on shelves. PR2 is depicted in the nominal position we crafted for it to maximize its success rate.

B. Motion Planning

In order to navigate from the simulation origin to the object of interest, the motion planner employs a hierarchical planning system. First, we heuristically select which end effector to use by selecting the one with the smallest Euclidean distance to the target. We then formulate an inverse kinematics optimization problem to find the joint positions that will align the chosen end effector with the object of along the plane formed by the front of the shelf. Our inverse kinematic problem's cost is the quadratic error of the current pose against the nominal pose, seen in Figure 1. We found this choice of nominal pose and cost formulation increased our success rate by incentivizing a human-like stance where one arm does not preclude the other arm's access to any part of the shelves, nor does it limit mobile base adjustments during the heavily constrained task of reaching inside the shelves. We run inverse kinematics up to 10 times with different randomized joint initializations and use the first successful solution.

We then use the Rapidly-exploring Random Trees (RRT) algorithm [11] to chart out a global plan from the initialization location to the end effector goal in joint space. We account for the impact of our mobile base's rotational joint through the following formulation of distance for use in finding the nearest neighboring nodes in RRT.

$$min(q_{des} - q, q_{des} - q - 2\pi, q_{des} - q + 2\pi)$$

We then generate smooth local paths between each point in our global plan using Kinematic Trajectory Optimization (KTO). We place a cost of weight 10 on the trajectory duration and a quadratic error cost on the error between the robot's current position and the first and last control points in the trajectory's b-spline formulation, respectively. We constrain the trajectory to stay within the PR2's joints' position and velocity bounds, begin and end at the points specified by the pair of RRT points we're interpolating, have a duration $t \in [0.5, 50.0]$ seconds, and have zero velocity at the start and end of the entirety of the RRT path. In pursuit of fast object reorganization, we find the fastest feasible traversal of each of the trajectories from KTO using Time Optimal Path Parametrization based on Reachability Analysis (TOPPRA) [12]. In aggregate, this module enables fluid motion plans optimized over the entirety of the robot and executed as quickly as possible.

C. Grasping

Utilizing pre-computed point clouds with normal vector estimates, we determine antipodal grasp candidates to which the PR2 can move its end-effector. We follow a similar approach to the antipodal grasping method presented in the course textbook. Each candidate's pose is determined as follows:

First, we randomly sample a point in the point cloud. With this point and the normal vector estimation at that point, we determine how and where to orient the gripper with respect to the sampled point such that we might obtain a viable antipodal grasp pose. Then, we score the grasp candidate with

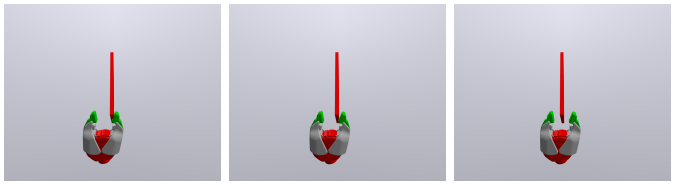

Fig. 2. Our grasp candidate pipeline has three stages. First, we put the finger on a random point in the point cloud such that the gripper's Y-axis is aligned with the normal of the point. Second, we solve an optimization problem to move the gripper out of collision. Third, we heuristically centralize the gripper in the Y-axis with respect to the points near the fingers.

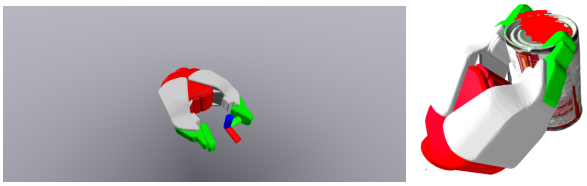

Fig. 3. left: In the gripper frame, the x-axis is toward the approaching direction and the y-axis is toward the fingers.
right: the point cloud that is bounded by the fingers is colored in red

a cost function that checks for collisions as well as deviations of the gripper from desirable orientations. Our cost function helps ensure that the grasps found are antipodal as well. After sampling potential antipodal grasp poses and scoring them, they are sorted such that those lowest costs are our best candidates. The cost function we evaluate grasps on is the following, based on the concepts presented in the course textbook [11]:

$$Cost_{grasp} = Cost_{direction} + Cost_{cover}$$

$$Cost_{direction} = -c_1 n_{approach}^T . x_{gripper}$$

$$Cost_{cover} = -c_2 \sum_{i \in S} |n_i . y_{gripper}|^2$$

Where $x_{gripper}, y_{gripper}$ are the axis of the gripper (figure 3), n_i is normal of ith point in the point cloud, S is the set of labels of points bounded by the fingers, and $n_{approach}$ is the direction that the robot approaches the object.

To bridge the gap between determining a grasp pose and bringing the PR2 robot's end-effector to that position, we make use of the aforementioned full-body inverse kinematics (IK) to determine the appropriate joint parameters, then apply trajectory optimization to move the PR2 between its previous motion planning position and the new grasping position. In the IK formulation we constraint the position and the orientation of the gripper except for the roll. We observed that without providing slack on roll value, IK will most likely become infeasible for the full robot (due to the collision constraints).

D. Perception

In order to properly interact with the objects and shelves within the environment, the robot must obtain information about the its surroundings. We make use of privileged information, allowing our PR2 robot to operate on perfect perception data.

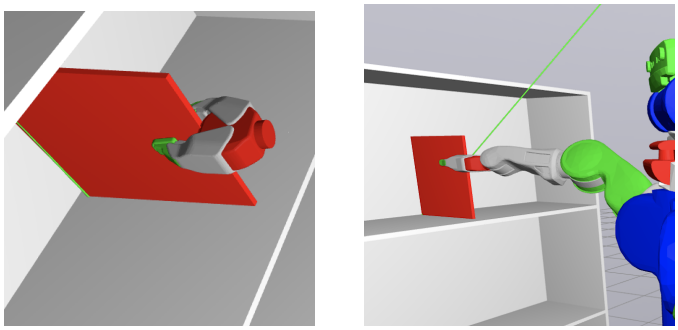

Fig. 4. Visualizing grasp poses with PR2 gripper.

One of the ways in which we obtain our privileged perception data is through pre-computed point clouds. By creating separate scenes including multiple cameras pointing the object of interest, we obtain robust point clouds with estimated normals for each object that the robot would grasp, as seen in Figure 5. Pre-computing and storing this data enables our PR2 robot to easily work with reliable data when determining grasp candidates, and it eliminates the risk of incomplete point cloud data due to occlusion or partial views.

Fig. 5. Pre-computed point cloud of a soup can.

Images obtained from the depth cameras must be merged together and mapped to construct a useful point cloud. This mapping of pixels to poses is specified by the pinhole camera model [14], where (X_C, Y_C, Z_C) are the points in the camera frame, f_x, f_y are the focal lengths in the x and y directions, respectively, and c_x, c_y specify pixels:

$$X_c = (u - c_x)\frac{Z_c}{f_x}$$

$$Y_c = (v - c_y)\frac{Z_c}{f_y}$$

Rather than directly performing this computation, however, we were able to further utilize privileged information in Drake by making use of each camera's point_cloud output port in the system.

Another key way in which we utilize privileged information is by means of Drake's query_object. Obtaining the

query_object from the plant and inspecting it enables us to directly glean information about the objects in the scene and determine the presence of collisions.

While we could have integrated sensors, such as cameras, directly into our simulation to collect this data, perception was not the main priority of our implementation, so we opted to make this simplification in favor of developing strong motion planning and grasping modules.

IV. Evaluation and Discussion

To quantitatively characterize our system's performance we designed environments and initializations in an attempt to find edge cases or shortcomings.

A. RRT Experiment

We verified that the functionality of our distance formulation for RRT by checking that the number of nodes necessary to reach an object that differed in location only in it's angle relative to PR2's nominal was relatively constant, as expected in our symmetric simulation environment (see Figure 6).

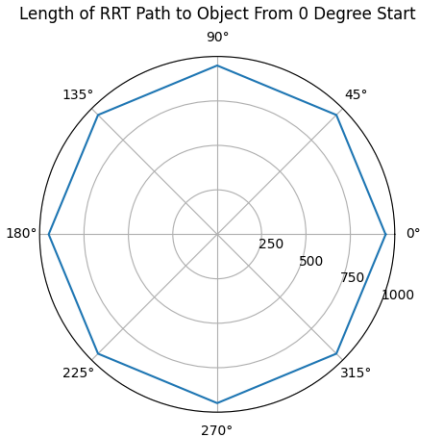

Fig. 6. The number of points in the RRT path generated for the same target object at different angles away from PR2's nominal of 0 degrees. As displayed above, the number of nodes in this experiment $\in [948, 949]$

B. Vertical Object Reachability

We verified that our motion planner is capable of navigating to successful pre-grasp poses for objects located all three floors of our shelves, which spans a height of 1.5 meters (see Figure 7).

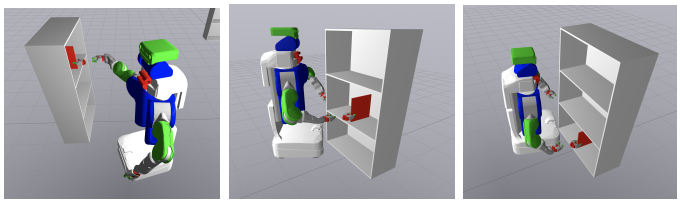

Fig. 7. Our motion planning pipeline navigates PR2 to the pre-grasp pose for successful object grasping on all three shelf floors.

```
OpenGripper(hand).then(
    MoveToPlace(hand, object)
).then(
    Grasp(hand, object)
).then(
    CloseGripper(hand)
).then(
    EscapeCollision(hand, object)
).then(
    GoBack()
).then(
    MoveToPlace(hand, object)
).then(
    Place(hand, object)
).then(
    OpenGripper(hand)
).then(
    EscapeCollision()
).then(
    GoBack()
)
```

Fig. 8. shows our symbolic plan for each object. Since grasping and placing actions are sensitive to distance and collisions, we utilize EscapeCollision to start in a position in which the robot is not in collision. MoveToPlace and MoveToGrasp use our motion planning component. Grasp and Place use Kinematic Trajectory optimization to move the gripper to the desired position.

C. Horizontal Object Reachability

Through repeated trials beginning at the outermost edges of the shelf and working inwards, we found that the first object location that we can reliably navigate to and grasp is 15cm in from the edge of the shelf, see Figure 9.

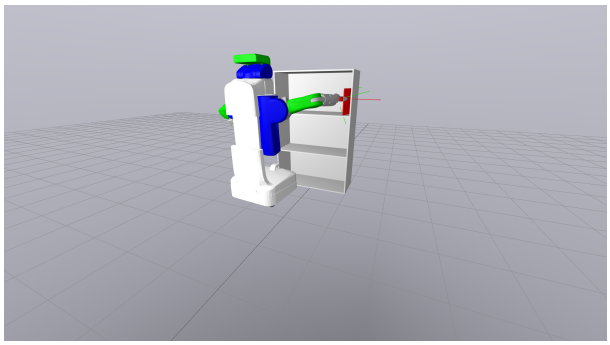

Fig. 9. Most extreme rechable object location, in terms of horizontal placement.

D. Limitations and Next Steps

Obstacle Avoidance While our implementation successfully avoids obstacles in terms of respecting minimum distance constraints surrounding the shelves and objects, it does not account for other obstacles in the system. The next step in increasing the applicability and robustness of this work would

be to improve its ability to navigate around both large static obstacles, enhancing its ability to operate in messy environments, and dynamic obstacles, making progress towards close-contact human-computer interaction spaces.

Perception Our implementation currently makes use of privileged information for perception. A potential avenue for future work is to add a perception module where sensors interface with the robot to provide information about the environment. We could then explore the challenges of imperfect or incomplete perception information and investigate the effects of noisy sensor data on our motion planner.

Task Planning The high-level planner employed by our project currently makes use of manually specified plans (figure 8). Our symbolic programming framework, however, has the potential to easily be extended through various other means. For instance, the symbolic programming design could be integrated with a higher-level task planner through program synthesis to achieve some end. Large Language Models (LLMs) also offer a promising avenue with which to integrate our motion planner. This is particularly relevant to our use case, as we aim to develop an efficient mobile manipulator for bookshelves, a space where humans frequently navigate and perform organization tasks—such an extension of our planner could enable symbiotic human-computer interaction in line with our focus on human form factors. With TidyBot, Wu et. al [4] show the strong potential of LLMs integrated with robotics and how they can be used to offer directives given a user's preferences.

V. Conclusion

Our pipeline successfully identifies, navigates to, grasps, and relocates objects of interest. As demonstrated in our results, our system provides object reorganization coverage across a large portion of the shelves and its performance is robust to target object location across a 64 meter environment. Furthermore, these capabilities are verified within an environment designed in a human form factor, providing a strong proof of concept for robots that can operate in existing industrial and everyday applications, rather than requiring new custom-made environments like their modern-day counterparts. We hope this work provides a valuable stepping stone in the pursuit of increasingly versatile and robust autonomous mobile manipulator systems.

Acknowledgment

We would like to thank Professor Tedrake, the technical staff, and communication instructors for their support, as well as their wonderful and informative lectures and recitations this semester.

VI. Contributions

Julianna S. led the system design, implementation, and integration and developed the motion planning pipeline. Shayan P. led the development of the simulation, grasping pipeline, and symbolic programming setup. Anna Y. led point cloud generation and data collection and contributed to the development of the grasping pipeline. All members contributed to the communication of results.

References

[1] "TRI Taking on the Hard Problems in Manipulation Research Toward Making Human-Assist Robots Reliable and Robust — Toyota Research Institute." Accessed: Nov. 02, 2023. [Online]. Available: https://www.tri.global/news/tri-taking-hard-problems-manipulation-research-toward-making-human-assist-robots-reliable

[2] S. University, "Robot provides personalized room cleanup," Stanford News. Accessed: Nov. 02, 2023. [Online]. Available: https://news.stanford.edu/2023/10/03/robot-provides-personalized-room-cleanup/

[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.

[4] J. Wu et al., "TidyBot: Personalized Robot Assistance with Large Language Models".

[5] J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, and H. Inoue, "Motion Planning for Humanoid Robots Under Obstacle and Dynamic Balance Constraints," Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation, 2001. doi:10.1109/robot.2001.932631

[6] A. Wu, S. Sadraddini and R. Tedrake, "R3T: Rapidly-exploring Random Reachable Set Tree for Optimal Kinodynamic Planning of Nonlinear Hybrid Systems," 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 2020, pp. 4245-4251, doi: 10.1109/ICRA40945.2020.9196802.

[7] S. Stavridis, P. Falco and Z. Doulgeri, "Pick-and-place in dynamic environments with a mobile dual-arm robot equipped with distributed distance sensors," 2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids), Munich, Germany, 2021, pp. 76-82, doi: 10.1109/HUMANOIDS47582.2021.9555672.

[8] T. Cohn, M. Petersen, M. Simchowitz, and R. Tedrake, "Non-Euclidean Motion Planning with Graphs of Geodesically-Convex Sets," Robotics: Science and Systems XIX, May 2023. doi:10.15607/rss.2023.xix.057

[9] 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

[10] R. Tedrake and the Drake Development Team, Drake: Model-based design and verification for robotics, 2019.

[11] R. Tedrake, Robotic Manipulation. 2023.

[12] H. Pham and Q.-C. Pham, "A New Approach to Time-Optimal Path Parameterization based on Reachability Analysis," IEEE Transactions on Robotics, vol. 34, no. 3, pp. 645–659, 2018. doi:10.1109/tro.2018.2819195

[13] J. Mahler et al., "Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics," Robotics: Science and Systems XIII, 2017. doi:10.15607/rss.2017.xiii.058

[14] "Camera Calibration and 3D Reconstruction," OpenCV, https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html