Sawyer Vision Grasp

Jacob Gottesman
ME Undergrad
jacobgottesman@berkeley.edu

Kevin Wang
EECS Undergrad
kevwang@berkeley.edu

John Vicino
EECS Undergrad
johncvicino@berkeley.edu

***Abstract*—We present in this paper a method for robotic arms to grasp objects using depth sensor data that the arm gathers about an object. We do this by applying various techniques to accurately reconstruct an object mesh via multiple depth images, and combine that with classical grasp evaluation metrics to determine where to best position our grasp. We apply these techniques to a Sawyer robot arm with an Intel RealSense camera mounted onto the arm, and show that it is highly effective at grasping objects in the real world.**

I. Introduction and Motivation

In order for robots to operate safely and effectively in the real world, they must be aware of the changing environment around it and be able to interact with and manipulate objects accordingly. One such interaction that is applicable to many robotic tasks is grasping, or the ability for a robotic limb to stably pick up and hold onto an object.

In Project 4 of EECS C106B at UC Berkeley, we learned to use various metrics to compute the ability of a robotic arm to complete a certain grasp on a mesh, and used that to sample a high quality grasp for the Sawyer to do on a real object.

However, this method is significantly limited and brittle in real-world applications. First, it requires the system to have an existing 3D model of the object that it desires to grasp, which is a requirement that is rarely ever satisfied in practical situations.

Moreover, another major cause of failed grasps in the project was the difference between the modelled and actual positions and orientations of the objects. If an object was offset by some amount, any quality grasp is still not guaranteed to be successful

Thus, in this final project, we aimed to build a system to resolve these limitations by using sensory data to accurately grasp unknown objects placed in unknown positions.

II. Related Work

A. 3D Reconstruction from Point Clouds

Various techniques have been proposed for reconstructing meshes from 3D point clouds, including alpha shapes [1], IPD [2], Bayesian reconstruction [3], Tomographic reconstruction [4], and more recently, Point2Mesh [5]. We explore the use of several of these techniques in this paper to use for computing grasping metrics.

Among the array of techniques we particularly focused on alpha shapes [1], which allowed us to control the level of detail in our mesh by setting the alpha parameter.

B. 3D Reconstruction from Color

Moreover, in recent years there have also been methods proposed to create meshes from purely visual data, without the use of depth sensors. These include NeuralRecon [8] and LivePose [9]. In order to isolate the desired object from the environment, one possible approach would be to use the Segment Anything Model [10] or, for video, Segment-and-Track Anything [11] to filter for the object and only apply 3D reconstruction to the pixels identified by the segmentation model.

This is an approach we are highly interested in exploring, but unfortunately we did not get the opportunity to do so in the short time frame of this project.

C. Grasp Metrics

Classical grasp metric techniques have been studied extensively, with some of the foundational methods including force closure [6] and Ferrari-Canny [7]. We use both of these methods to evaluate possible grasps on our reconstructed mesh, along with robust force closure and also using convex optimization to compute if there are contact forces possible to counteract the force of gravity with each grasp.

III. Methods

A. Setup

This project utilizes a Sawyer 7-degrees-of-freedom robotic arm, as well as an Intel RealSense D435i camera mounted to the arm for depth data. The Sawyer moves to pre-programmed joint angles/poses around a capture area and uses the RealSense to record a series of images.

B. Point Cloud Generation

We then to use these images to generate an object point cloud by doing the following:

1) To eliminate noise and outliers, we take multiple (11) images at each capture location and use the median depth at

each pixel out of all of the images.

2) Using camera intrinsics and the transformation matrix from the camera to the robot base frame, we convert the de-noised depth image from each capture location to a point cloud in the base frame.

3) To isolate the object we want to grasp, we crop out points in the point cloud that are outside of a set bounding box we expect the object to be at, which we define as a 20cm x 20cm x 20cm box right above the surface of the table that the object is put on.

4) Finally, we combine all of the point clouds from each angle into a complete point cloud of the object.

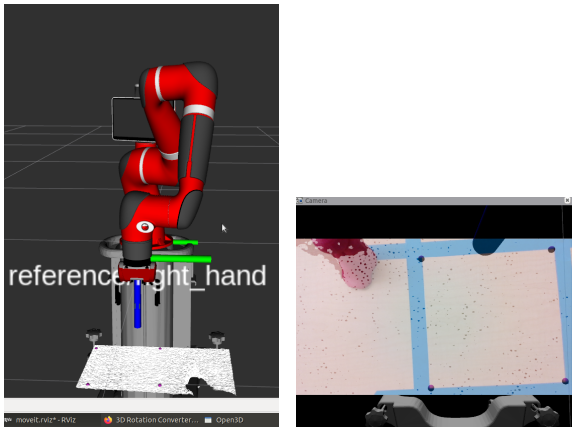
C. Camera Transformation Calibration

In order to merge multiple capture angles into a single point cloud by just adding them together, a precise calibration of the transformation of the RealSense to the Saywer arm is needed. In this section, we will discuss how we approached this.

Initially, our calibration process involved establishing a fixed reference point that was easily identifiable on the end effector of the Sawyer arm. We selected four corners of a physical square located on a table directly in front of the Sawyer (on a Z-plane) and in the base frame. To calibrate our camera pose, we manually moved the Sawyer to each point.

Following the setup of these reference points, the next step involved positioning the RealSense camera to capture these points effectively. We adjusted the camera to a viewing angle that was perpendicular to the table or Z-plane. This orientation was crucial to avoid any angular distortion that could affect the accuracy of the point cloud. The placement was also calculated so that the camera could comfortably view all four calibration points.

Displaying both the depth image and calibration points in Rviz, we adjust the RealSense transformation in regards to the right hand frame (Z translation, Y-axis rotation) and physically rotate the RealSense about it's mounting bolt (X-axis) so that the calibration points are flat on depth image created table plane. We then adjust the transformation further until the calibration points as being viewed through the RealSense camera image align with the actual physical location of the points (X translation, Y translation, Z rotation).



(a) Z translation, XY Rotation (b) XY translation, Z Rotation

Fig. 1: Calibrated Transformation

D. Cloud Cleaning and Mesh Generation

After we have a complete point cloud for the scanned object, we apply a series of post processing steps. First we clean the point cloud by applying a statistical outlier remover (STD of 2, neighbor count of 20). This helps remove any sensor noise that was embedded in the depth image.

Following the outlier removal, we employ Voxel downsampling to lower the point cloud's density. Specifically, we uniformly down sample the point cloud to 5% of the original number of points.

After cleansing, the point cloud undergoes surface reconstruction using the alpha shape technique. This method serves as a generalization of convex hulls, but introduces a parameter that allows for a customizable degree of concavity. This allows us to capture more of the detail in the object's geometry. We converted the data into a triangular mesh using an alpha shape parameter of 0.005. Simple smoothing is used to remove any irregularities and helps bring out the key features of the object, leading to more effective grasps.

After completing the mesh, we proceed to calculate the surface normals. This crucial step involves determining the orientation of each segment of the mesh surface, which plays a pivotal role in the grasping stage and in optimizing the gripper's orientation. Accurately calculated normals ensure that the robotic gripper aligns perfectly with the object's surface, enhancing the stability and success of the grasp. These normals are not only essential for the mechanical interaction between the gripper and the object but also for ensuring that the robot can handle the object in the most efficient and secure manner possible.

E. Grasping

Once the mesh is created, we proceeded by sampling points from the bounding box around it. We sample a collection of

random points and project them inward, towards the mesh. We then check for mesh intersection allowing us to only pick points on the outside facing surfaces of the mesh. In addition to the mesh intersection check, we also limited the points to a set Z height. This helped avoid collisions with the table.

After the points have been selected they are analyzed for viability through a robust force closure check. We experimented with a variety of quality thresholds and eventually landed on a 0.8 threshold for our use case. This metric seemed to return enough grasps, while still ensuring the objects would lift from the table. Lowering this value occasionally caused some objects to fail their grasps. Additionally, raising it too high significantly diminished the number of feasible grasps identified

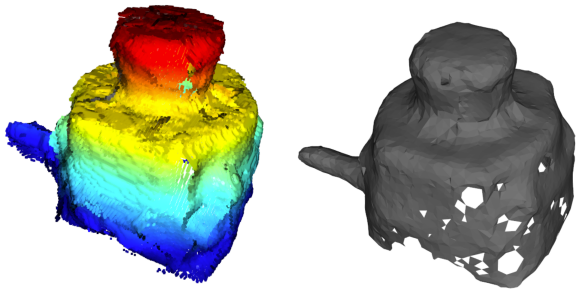
Our grasping algorithm also checks the gripper angle. We force the gripper to approach at an angle relative to the object normal. This improved side grasps as it helped avoid situations where the gripper would push or knock the object.

Once we have calculated a suitable grasp we run it. We move the Sawyer to the start position, the object grasp position, and the raise object position using IK.

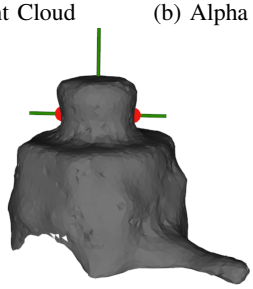
IV. Results

Overall our project was successful in its goals. We were able to collect point clouds, generate a mesh, and search for then execute grasp points that satisfy our grasping criteria.

Below is one such example of the process. Fig. 2a shows the point cloud of an E-stop after being scanned from multiple capture angles. A video of the scan taking place can be found at the end of the report. Fig. 2b shows the triangular mesh created after cleaning the point cloud, applying an alpha shape, and smoothing the resultant mesh. Lastly, Fig. 2c shows the pair of grasps points and pose generated by our grasping algorithm. An execution of these grasp points can be found in the video linked at the end of the report.



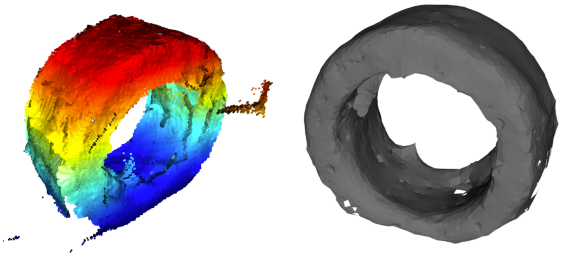
(a) Merged Point Cloud (b) Alpha Shape Mesh



(c) RFC Grasp

Fig. 2: E-Stop Scan & Mesh

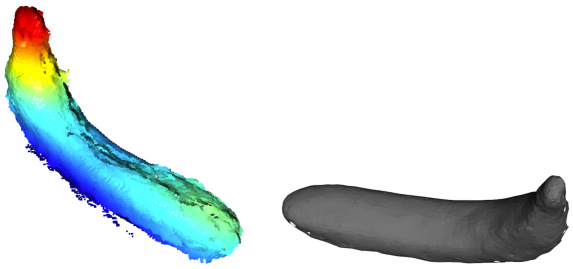
Another object we scanned was a role of duct tape as seen below in Fig. 3. As can be seen in Fig. 3a, some noise was recorded in the point cloud for the tape likely as a result of it's reflective surface. However, as we can see in Fig, 3b, this noise was cleaned and removed from the mesh creation process.



(a) Merged Point Cloud (b) Alpha Shape Mesh

Fig. 3: Tape Scan & Mesh

Lastly, below is yet another scanned and meshed object.



(a) Merged Point Cloud (b) Alpha Shape Mesh

Fig. 4: Banana Scan & Mesh

A final proof of the efficacy of this method was our live demonstration during the EECS C106B final project presentations, where the Sawyer was able to successfully grasp a tape measure, an object that was provided by an audience member and which the system had never been tested on before.

V. Conclusion and Discussion

A. Recap

This project featured the successful 3D reconstruction of an object from multi-angle scanned point cloud images. Initially, the point cloud underwent a series of prepossessing steps. These included outlier removal and downsampling.

Following the preprocessing steps the refined point cloud was transformed into a mesh. The mesh was subjected to smoothing and was later used to determine grasp locations. The robot engaged the object at these points and successfully picked it up.

Additional trials with a variety of objects were performed to further validate our approach.

B. Difficulties

As previously noted, achieving accurate merged point clouds necessitates a highly precise transformation between the RealSense camera and the right hand of the Sawyer robotic arm. This level of accuracy, within millimeters, is crucial for ensuring that the spatial data collected from multiple viewpoints are correctly aligned and integrated. The calibration process to achieve such precision is intricate and time-consuming, involving multiple steps of adjustment and verification. It requires careful manipulation of the robotic arm and camera settings to fine-tune their relative positions. Each calibration session involves repeatedly positioning the arm and adjusting the camera, followed by capturing test data to analyze the alignment accuracy. This took a lot of time and we had to calibrate at the start of a new session or when moving between different robots.

Additionally, optimizing the settings for point cloud cleaning and mesh creation involved extensive trial and error, which was crucial to our workflow as each parameter significantly influenced the quality and usability of the final mesh for grasping applications. We needed to carefully balance the level of detail we retained while addressing issues with sensor noise, which occasionally introduced outliers into our point cloud. The depth sensor's inherent noise presented a challenge; too aggressive in removing these outliers could result in losing important surface details in our mesh. Consequently, finding the right balance was key to ensuring that the essential features were preserved for accurate grasp execution while eliminating unnecessary noise that could affect the model.

On the grasping side, some paths computed by the Inverse Kinematics (IK) led the robot along trajectories that risked collisions or were not optimal for object handling. In certain instances, the robot inadvertently flipped the object upside down during the post-grasp movement. Although we were able to disengage the robot along these paths testing the robot in a more optimal environment would have been beneficial.

Moreover, the surfaces of some objects posed additional challenges. Our current algorithm does not consider the texture or precise weight of an object, resulting in situations where, despite technically correct grasp positioning, the actual lifting capability was inadequate.

C. Future Work

1) Our system requires a reliable Lidar camera in order to generate the point cloud data. Looking forward, we are interested in exploring the potential of incorporating color images to perform the 3D reconstructions without the need for depth data. This approach could leverage advanced computer vision techniques, like Segment Anything Model (SAM), to infer the depth from color imagery. This could help bring down the cost of hardware as it could eliminate the need for expensive depth sensors and instead replace them with regular cameras.

2) Given the variations in our scanned point clouds across different sessions and setups, it would be highly beneficial to implement robust point cloud comparison metrics. Specifically, we could use a point cloud comparison metric such as the Hausdorff metric. This would allow us to see how much our scanned point cloud differentiates from scan to scan. It would also allow us to test our scans against benchmarks, such as known clouds or 3D printed items. Having this information could help us pinpoint specific causes of deviations, such as hardware or environmental factors, and assess the quality of our scans.

3) Occasionally, the meshes generated from our scanning process contained holes, which, although they did not hinder the functionality of our grasping algorithms, highlighted potential areas for improvement. These imperfections in the mesh can arise from various factors, such as inadequate coverage of the object's surface during scanning or limitations in the mesh reconstruction algorithms themselves. Some objects may need a more complete object reconstruction, since they may contain more complex features outside of our current scanning range. We are interested in investigating the ideal number and location of images needed to achieve this.

4) Another avenue of potential direction for future development is the integration of object recognition with our point cloud creation. By incorporating object detection, we could enable the robotic arm to identify various items within the point cloud data automatically. This capability would allow the robot to adjust its approach based on the type of object it encounters, changing the grasp metrics

accordingly. Object recognition could be implemented using machine learning techniques that classify objects based on their geometric and textural features extracted from the point clouds. This would enable the robotic to handle a diverse range of items, including fragile or strangely shaped, with appropriate grasping strategies tailored to each item's specific characteristics and requirements.

REFERENCES

[1] Edelsbrunner and D. G. Kirkpatrick and R. Seidel: On the shape of a set of points in the plane, IEEE Transactions on Information Theory, 29 (4): 551–559, 1983

[2] Hong-Wei Lin, Chiew-Lan Tai, Guo-Jin Wang. A mesh reconstruction algorithm driven by an intrinsic property of a point cloud, Computer-Aided Design, Volume 36, Issue 1, 2004, Pages 1-9, ISSN 0010-4485, https://doi.org/10.1016/S0010-4485(03)00064-2.

[3] P. Jenke, M. Wand, M. Bokeloh, A. Schilling, W. Straßer. Bayesian Point Cloud Reconstruction, Computer Graphics Forum, 25: 379-388. https://doi.org/10.1111/j.1467-8659.2006.00957.x

[4] Sitek, Arkadiusz, Ronald H. Huesman, and Grant T. Gullberg. "Tomographic reconstruction using an adaptive tetrahedral mesh defined by a point cloud." IEEE Transactions on medical imaging 25.9 (2006): 1172-1179.

[5] Hanocka, Rana, et al. "Point2mesh: A self-prior for deformable meshes." arXiv preprint arXiv:2005.11084 (2020).

[6] Nguyen, Van-Duc. "Constructing force-closure grasps." The International Journal of Robotics Research 7.3 (1988): 3-16.

[7] Ferrari, Carlo, and John F. Canny. "Planning optimal grasps." ICRA. Vol. 3. No. 4. 1992.

[8] Jiaming Sun, Yiming Xie, Linghao Chen, Xiaowei Zhou, and Hujun Bao. NeuralRecon: Real-Time Coherent 3D Reconstruction from Monocular Video. In CVPR 2021.

[9] Noah Stier, Baptiste Angles, Liang Yang, Yajie Yan, Alex Colburn, and Ming Chuang. LivePose: Online 3D Reconstruction from Monocular Video with Dynamic Camera Poses. In ICCV 2023.

[10] Kirillov, Alexander, et al. "Segment anything." Proceedings of the IEEE/CVF International Conference on Computer Vision. 2023.

[11] Yangming Cheng, Liulei Li, Yuanyou Xu, Xiaodi Li, Zongxin Yang, Wenguan Wang, and Yi Yang. Segment and Track Anything.

VI. APPENDIX

A. Code Repository

Github repository:
https://github.com/kevinzwang/sawyer-vision-grasp

B. Video Link

E-Stop Scan Video: https://drive.google.com/file/d/1KKCaPrU70NkIG7OGQR6fjtV2sFfDNEK0/view?resourcekey

E-Stop Grasp Execution Video: https://drive.google.com/file/d/1KQMqF4ezOHYhPMb0OXB6s8G2b49Jst40/view?resourcekey