A Geometric Approach to Recycling Cans via Vision-Based Manipulation

George Chen
Department of Mechanical Engineering
Massachusetts Institute of Technology
Cambridge, MA, USA
gcfchen@mit.edu

Susan Ni
Department of Electrical Engineering and Computer Science
Massachusetts Institute of Technology
Cambridge, MA, USA
sni@mit.edu

***Abstract*— In an effort to assist in cleaning polluted oceans and littered parks, this project presents a solution to use vision-based manipulation and optimization-based motion planning to recycle soda cans. With captured images from a camera above, we are able to perform multi-mask segmentation and normal estimation to calculate grasp poses for each can, both of which are our main novel innovations. A designed nominal trajectory starting with said grasp poses is accomplished by optimization-based inverse kinematics solved subject to constraints imposed by the environment, completing the pipeline for vision-based manipulation.**

***Index Terms*— segmentation, normal estimation, inverse kinematics, motion planning, trajectory optimization**

I. Introduction

Pollution has been a growing threat to many marine life species and environmental well-being. Large efforts have been devoted to clean up oceans, parks, and other areas polluted by plastic remains and human litter. This research project will attempt to use robot arms to assist the cleaning effort. With automation, we could potentially reduce the risk on human cleaners and achieve greater efficiency and speed, which accelerates the goal in creating a cleaner, healthier planet.

There are existing literature on using soft robots equipped with touch sensing to distinguish materials of recyclable objects in order to facilitate single-stream recycling. [1] They used manual feature selection, as opposed to k-nearest neighbors and support machine vectors which achieved lower accuracy in material detection. In their conclusion, they specifically cited that incorporating vision sensing could help improve their accuracy. Our task is simpler, as we are not trying to identify object materials, but we will solely be relying on vision sensing. By combining ideas from computer vision and trajectory optimization, we are interested in demonstrating a working manipulation pipeline for this task, and we present and discuss the details of our approach in this paper.

II. Approach

Our simulation environment is modeled after the example clutter clearing station. [2] We have an iiwa arm with a Schunk WSG gripper, a bin that acts as our ”recycling bin”, cans (cylinders) to be ”recycled”, a custom plate to hold all of our cans, and static cameras with RGB and depth sensors as shown in Figure 1. Our strategic placement of the plate was to minimize other objects being in view of the cameras while still having it close enough to the arm so cans even on the edge of the plate are in reach.

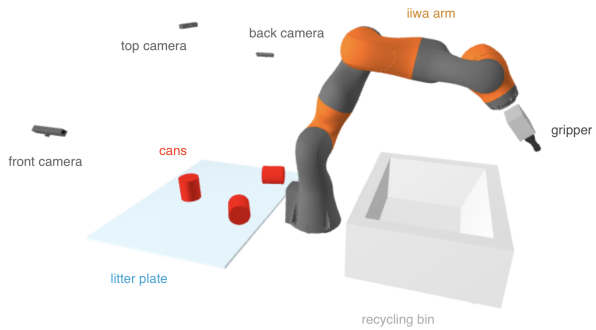

Fig. 1. Environment Setup.

The overall pipeline of our project is shown in Figure 2. The pipeline can be broken down into two main systems, perception and motion planning, were the initial input are images from the cameras in the scene and the final output are joint position commands to the iiwa.

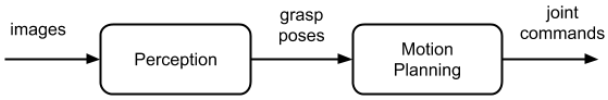

Fig. 2. Pipeline of entire project.

III. Perception

Our perception setup includes static cameras in the world frame that can view the full litter plate, so we are purely relying on vision and depth sensing. We also only want to leverage geometric approaches for segmentation, as opposed to learning such as Mask R-CNN. Figure 3 displays an overview of our perception system pipeline starting with images captured by our cameras to producing the grasp poses required for motion planning.

In order to make a geometric approach simpler, we have made some assumptions for our environment.

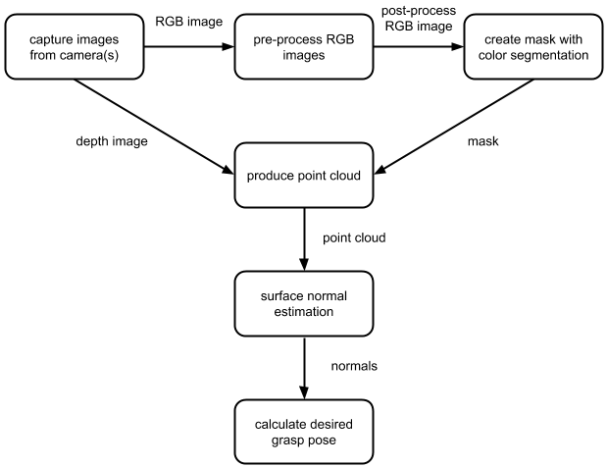

Fig. 3. Workflow of the perception system.

1) The litter plate will be a distinct color from the objects on it.
2) The objects are far enough apart on the plate such that grasping one will not interfere with the other objects on the scene. A consequence of the this assumption is that no objects are occluded from a camera above the litter plate and have distinct gaps between them in an image captured by the camera.
3) We have prior knowledge of geometry of the cans, i.e. height, radius, color, etc.
4) There is allowed to be a calibration period before segmentation is performed. The cameras will be adjusted to view the entire litter plate and we will create methods to filter out pixels that are beyond the plate.

These assumptions do not completely limit our approach. To relax the first assumption, if we wanted a more robust approach that does not depend on color, we could have used RANSAC for plane removal. [2] Additionally, the second assumption was made so we could capture images from the cameras only once and determine all of the poses of the objects in the scene at one time. This serves as a proof of concept, but if objects are occluded or grasping one object moves another, we could have re-run our perception pipeline to re-calculate grasp poses after each can is removed.

A. Image Pre-processing

From some camera angles, in order to view the entire litter plate, other objects, such as the iiwa arm, are also in view. However, in order to preserve our simple color segmentation problem, objects outside of the litter plate can be removed and treated as empty space in the RGB image. This can easily be done by determining the equations of the lines representing the edges of the litter plate for both the front and back camera images (Figure 4) and setting all pixels outside of these lines equal to the color of empty space. Example post-process images are shown in Figure 5.

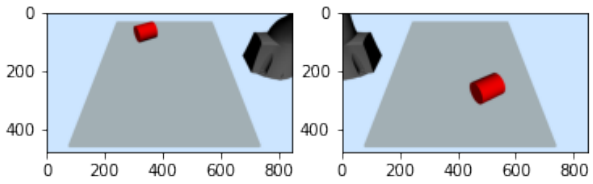

Fig. 4. Front camera RGB image (left) and back camera RGB image (right) before processing.

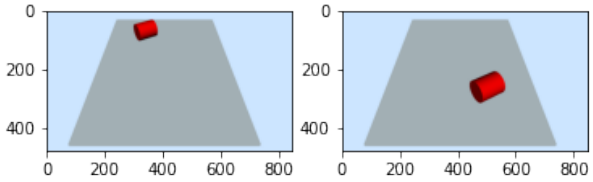

Fig. 5. Front camera RGB image (left) and back camera RGB image (right) after arm is cropped out.

B. Segmentation

With image pre-processing, the color segmentation process becomes very simple. Even from different camera angles, the litter plate is always a gray-ish color such that all the values of R, G, and B are all around the same and greater than 100. On the other hand, our cans are a bright red so the plate can easily be filtered out to create a mask for the cans via color segmentation. Figure 6 is an example of an RGB image and Figure 7 is the mask produced.

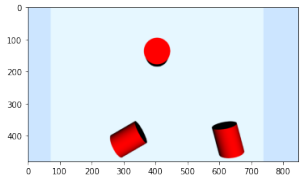

Fig. 6. RGB image captured by top camera.

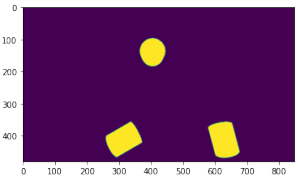

Fig. 7. Mask created for top camera RGB image via color segmentation.

C. Multi-object Mask Segmentation

If there are multiple objects on the litter plate, such as in Figure 6, then capturing an image will lose information distinguishing separate objects. There is a lot of literature on Mask R-CNN approaches for multi-object segmentation, but we are solely implementing geometric algorithms. Pseudocode for our geometric approach to mask separation is displayed in Algorithm 1.

D. Point Cloud Projection

The pinhole camera model was used to map pixel locations to 3D points in the world frame (Figure 8).

There was a slight complication likely caused by the fact that pixel indices are discrete values while 3D points can take on continuous values. Some of the pixels in the mask

Algorithm 1: Geometric Mask Separation.

Data: M = mask containing all objects. ($True$ = object pixel, $False$ otherwise)
Result: out = a list containing a mask for each object.
while M *contains* $True$ *pixel* **do**
  $i \longleftarrow$ index of any $True$ pixel in M
  $indices \longleftarrow$ iteratively find all indices of $True$ pixels that are neighbors, neighbors of neighbors, etc., of pixel at i
  $mask \longleftarrow$ image the size of M where pixel = $True$ at $indices$, $False$ otherwise add $mask$ to out change all pixels at $indices$ in M to $False$ to remove detected object
end

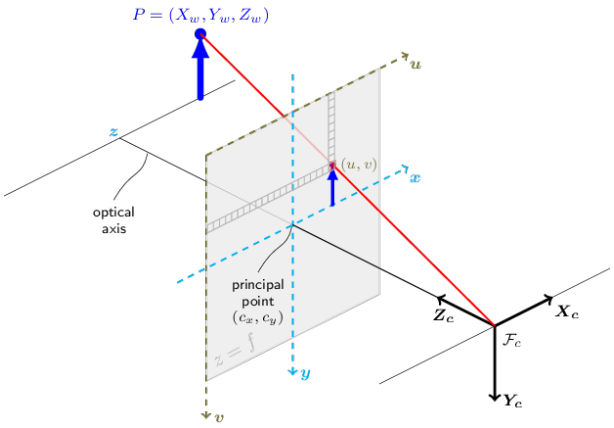

Fig. 8. Pinhole camera model. Adapted from [3].

that corresponded to the edge of the can projected onto the plate, instead of the can. In Figure 9, there is a ring of green points on the plate to the left of the can that should have been the edge of the can. This single ring of edge points is not entirely necessary, so we eroded the mask to get remove the outer pixels of object in the mask to avoid having incorrectly projected pixels. [4]

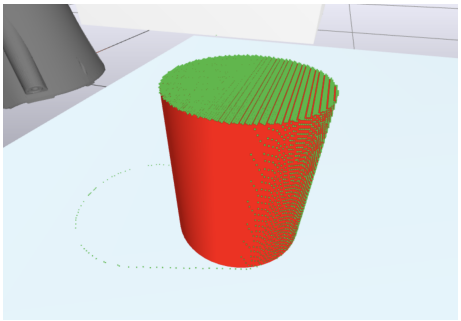

Fig. 9. Points incorrectly projected at edge of can.

E. Normal Estimation

We used the method for surface normal estimation described in Section 5.2.2 of the 6.881 course notes and flipped all normals to be facing the camera that generated the point cloud. [2]

F. Calculating Grasp Poses

The final output of our perception system is grasp poses for each can in the scene. Given the assumptions of our environment, a can is either going to be horizontal or vertical. In the vertical case, an ideal grasp pose is above the center of the can with a rotation in which the y-axis is pointing downward, due to the specification of the gripper frame (Figure 10). In the horizontal case, an ideal grasp pose is also above the center of the can in the x-y plane with a rotation where the y-axis is pointing vertically downward, but additionally the z-axis has to be parallel to the long axis of the can. This is because the gripper is not wide enough to grasp the can from the ends.

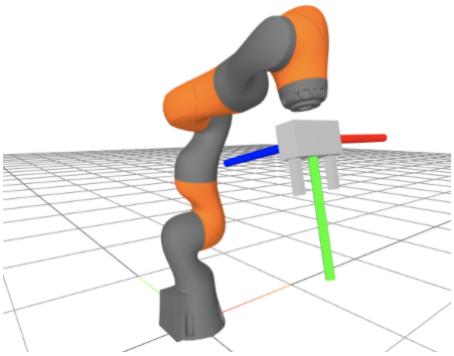

Fig. 10. Gripper frame. RGB corresponds with XYZ, respectively. Adapted from [2].

In 6.881, we learned about using the antipodal pairs heuristic to generate a grasp candidate (Section 5.3.4). [2] We tried to use this method with a single can in the scene, which required two cameras (the front and back cameras in Figure 1) because point clouds of opposite sides of the can have to be visible in order to even have an anitpodal pair of surface normals. We had a working solution, but we had to leverage the geometry of the can because a pair of normals from the ends of the can were not a viable antipodal pair since the distance between the pair would be greater than the gripper width. Additionally, we wanted to eventually extend our grasp pose calculations for multiple cans in the scene, but one can might occlude another can from the perspective of one camera, which goes against the purpose of why we had two cameras in the scene in the first place.

Instead of using antipodal pairs, we developed another method for generating grasp candidates that still depended on the geometry of a can, but only required one camera from above the center of the plate. In this method, for each can, we filtered the surface normals to only keep the ones that were sufficiently vertical and far from the plate. The vertical

normals of multiple cans can be seen in Figure 11. In the case of a vertical can, these normals were all on the top/end of the can. For a horizontal cylinder, the vertical normals should all be in a line parallel to the long-axis of the can at the highest point on the can's side. However, since we are using sliding windows for surface normal estimation, we might not have calculated a normal centering exactly on the highest point of the cylinder, but a linear regression of the positions in the x-y plane of the sufficiently vertical normals results in a good estimation. Since this regression line is parallel to the long-axis of the can, then it will also be parallel to the normals on the ends of the can. As discussed earlier, the z-axis of rotation of the grasp pose should be parallel to the normals on the ends of the can. The position of the grasp position can simply be found from the average position of the vertical normals in the x-y plane due to the symmetry of cylinders. The z-axis position value will be equal to the highest z-axis position of the vertical normals plus some offset to give the gripper some clearance. Sample grasp poses calculated via this method are shown in Figure 12. Pseudocode summarizing this method is in Algorithm 2.

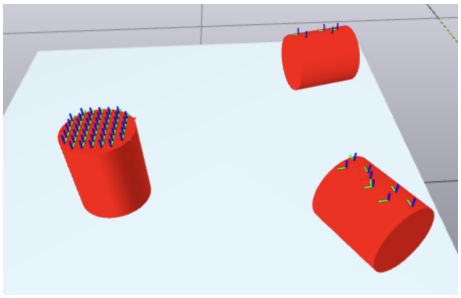

Fig. 11. Vertical normals on multiple cans.

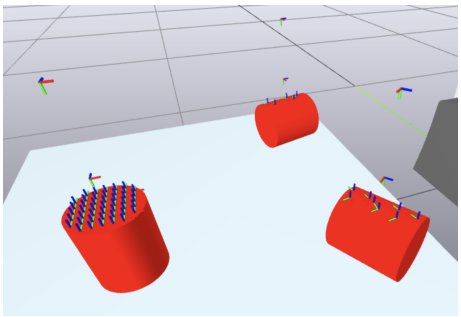

Fig. 12. Grasp poses with a pre-grasp pose some offset higher.

This method is leveraging the properties of a cylinder, but unlike the antipodal pairs method, it does not necessarily require the specifications of the can, such as its height, as long as it is given that the gripper is wider than the diameter of the can. The height is needed to explicitly differentiate between vertical and horizontal cans, which is what we do in our code, but if the height is unknown, then using the same linear regression calculation on the normals of a vertical can will still produce a viable grasp rotation.

Algorithm 2: Calculating Grasp Poses.

Data: N = A list of normals for each object in the scene.
Result: G = a grasp pose for each object.
for *each list of normals in* N **do**
  filter out all normals that are not sufficiently vertical or too close to the plate
  $position_{x,y} \longleftarrow$ average position of all vertical normals in x-y plane
  $position_z \longleftarrow \max$(z-position of all vertical normals) + offset
  $roll \longleftarrow -\frac{\pi}{2}$
  $pitch \longleftarrow 0$
  if *can is horizontal* **then**
    $axis \longleftarrow$ linear regression of positions of all vertical normals in x-y plane
    $yaw \longleftarrow$ angle between $axis$ and y-axis in x-y plane
  else
    $yaw \longleftarrow 0$ `// can be anything`
  end
  grasp pose $\longleftarrow$ $(position_{x,y,z}, rotation_{roll,pitch,yaw})$
end

IV. Motion Planning

The method for motion planning in our manipulation pipeline is inspired by an exercise from Prof. Tedrake's textbook. [2] Our approach solves a series of inverse kinematics problems along the trajectory, and the use of optimization-based motion planning ensures that the iiwa's end-effector poses are as close as possible to the nominal trajectory subject to constraints.

A. Defining Nominal Trajectory

In the designed nominal trajectory, several waypoints are put in place in order to avoid collision with the bin, other cans, and even the iiwa itself. They are strategically placed above the bin, above the litter plate, and in the middle between the bin and litter plate. Refer to Figure 13 to visualize the locations of these waypoints.

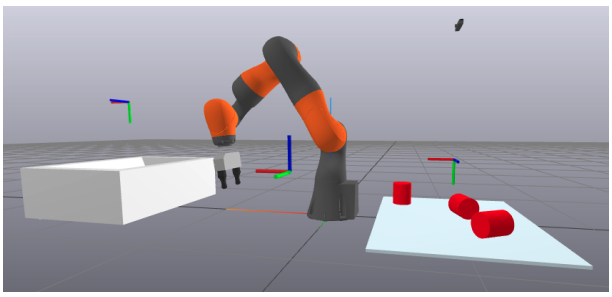

Fig. 13. Waypoints above the bin (left), in between (middle), and above the plate (right).

These waypoints, along with the pre-grasp and grasp poses output by the perception system, divide the nominal trajectory of grasping each can into the following segments. The pick and place cycle for each can lasts 45 time units.

- $t \in [0, 5)$: From above the plate to pre-grasp pose, gripper is open
- $t \in [5, 10)$: From pre-grasp pose to grasp pose
- $t \in [10, 13)$: Stays still at grasp pose, closes gripper
- $t \in [13, 18)$: From grasp pose to above the plate
- $t \in [18, 24)$: From above the plate to the middle frame
- $t \in [24, 30)$: From the middle frame to above the bin
- $t \in [30, 33)$: Stays still at grasp pose, opens gripper
- $t \in [33, 39)$: From above the bin back to the middle frame
- $t \in [39, 45)$: From the middle frame back to above the plate, resets $t \leftarrow 0$

In addition, the simulation also allows for an initialization period of 15 time units, in which the iiwa gets ready by moving from its initial pose to above the plate. The end-effector linearly interpolates its translation and rotation separately when traveling between poses. One cycle of the pick and place trajectory is illustrated in Figure 14.

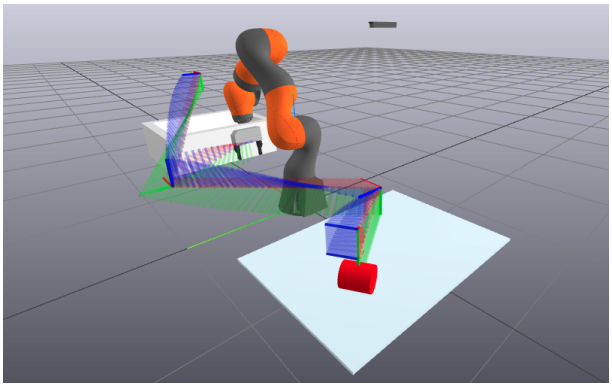

Fig. 14. Trajectory visualization for picking and placing one soda can.

These defined end-effector trajectories are converted to joint-space trajectories using optimization-based inverse kinematics, discussed below.

B. Inverse Kinematics as Constrained Optimization

With the nominal trajectory defined, we construct an inverse kinematics problem for the MultibodyPlant in simulation. Simultaneously, the inverse kinematics problem solves for joint states

$$q = f_{kin}^{-1}\left(X^G\right)$$

where f_{kin} is the forward kinematics function and X^G is the gripper frame. The joint states must also minimize the cost function

$$\min \|q - q_{desired}\|_2^2$$

subject to constraints such as maximum joint velocities and maximum position and rotation deviations from the nominal trajectory. In our implementation, the deviation allowances are narrowed when the gripper is near the cans, and loosened when the gripper is traveling between the litter plate and the bin.

V. Future Work

Our simulation admittedly has an environment too simplified for our approach to be applied to the real-world environment. To start, the cans model is adopted from the mug SDF model in the drake examples with the handle removed. However, we can construct a more realistic models of soda cans or import betters ones from, for instance, Gazebo. [5] In the real world, cans will also likely have dents and we can no longer rely on surface normals to reliably calculate grasp poses. We can envision some sort of smoothing algorithm if one wants to continue with this geometric approach, but we likely would have to incorporate some sort of learning. Realistically, there are also more just than cans in an environment that requires recycling. If we were to include arbitrary objects to our scene, then a top camera is no longer sufficient because some shapes look the same from a birds-eye view. For example, a can and an hour glass could look the same from a top camera, but only the can be recycled. We also likely can no longer utilize color segmentation and would need to use deep learning approaches.

We hope to improve our motion planning aspect by comparing optimization-based trajectory planning algorithms against sampling-based ones such as Rapidly Exploring Random Tree (RRT), and determine which produces more optimal paths and more natural poses for the robot arm. In tuning the inverse kinematics solver, we may also reduce the tolerance from the nominal path (while obeying joint limits, collision avoidance, and other physical constraints) so that in cluttered scenarios, the gripper is less likely to collide into other objects. To make this study more applicable, we could potentially make a mobile base for the iiwa and/or the trash bin to simulate a moving robot and a moving person holding trash bag, our intended real-life scenario.

VI. Conclusion

Our project demonstrates the full manipulation pipeline covered during the semester. The perception system can successfully detect multiple objects in a scene and our motion planning system can construct trajectories to recycle them to a bin. We present success in integrating multi-mask segmentation, grasp pose generation, and optimization-based motion planning to identify, pick and place objects. Key takeaways from our work include the natural reliance on the geometric properties of a scene when solely utilizing geometric approach. When these properties are not necessarily a given, such as in the real world, the limitations of our perception scheme are exposed and alternate methods have to be further developed to account for these variances.

References

[1] L. Chin, J. Lipton, M. C. Yuen, R. Kramer-Bottiglio, and D. Rus, "Automated recycling separation enabled by soft robotic material classification," in *2019 2nd IEEE International Conference on Soft Robotics (RoboSoft)*, 2019, pp. 102–107.

[2] R. Tedrake, "Robot Manipulation: Perception, Planning, and Control (Course Notes for MIT 6.881)." downloaded on Dec. 6, 2020 from http://manipulation.csail.mit.edu/.

[3] OpenCV, “Pinhole camera model,” 2020, Accessed Dec. 6, 2020. [Online]. Available: https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html
[4] A. Mordvintsev and A. K., “Morphological Transformations,” 2013, Accessed Dec. 6, 2020. [Online]. Available: https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_morphological_ops/py_morphological_ops.html
[5] Osrf, “Population of models,” Apr 2020, the world explained. [Online]. Available: http://gazebosim.org/tutorials?tut=model_population&cat=build_world#Theworldexplained

VII. Appendix

The source code of this project is contained in this Google Colaboratory notebook.

A demonstration of the working simulation is uploaded to YouTube here.