Laavanye Bahl

Team D: CuBi

Teammates:
Jorge Anton, Paulo Camasmie, Changshen Shen, Nithin Subbiah Meganathan

ILR10

November 7, 2019

Individual Progress

The main focus of this review was robust object detection to adapt to the new environment and testing out the validation pipeline with test programs and improving it further.

Grasp Validation
For grasp validation, the developed pipeline was tested against our test cases and it was observed that shadows and lighting changes were creating a lot of problems for the rgb based validation pipeline. This has further been described in the challenges section.

The previous pipeline with rgb images was discarded and a new pipeline with depth images was developed which was robust in the new environment where there were a lot of lighting variations because of the presence of just a single source of light as opposed to multiple present at the previous location of NSH 4th floor.

Following describes the pipeline:

- Start a publisher to get an aligned depth image (with the rgb image) from the Realsense camera.
- Command manipulator to go to the fixed validation position on boot up.
- Collect the depth image of the empty tray as shown in Fig 1a and store it.
- Get the pre-defined corners of the region to crop.
- Warp this cropped portion with inverse perspective mapping and rectify it for better comparison as shown in Fig 2a and 2b.
- Obtain the current depth image when the comparison is required as shown in Fig 1b.
- Follow the same procedure as the empty tray.
- Now for comparison, the difference of the depth images of the cropped regions in both can be used against a threshold to validate the presence of an object in the tray.
- But, the depth image contains NaN points (points that are not reconstructed in the depth image). This challenge and its solution is discussed further in the challenges section.

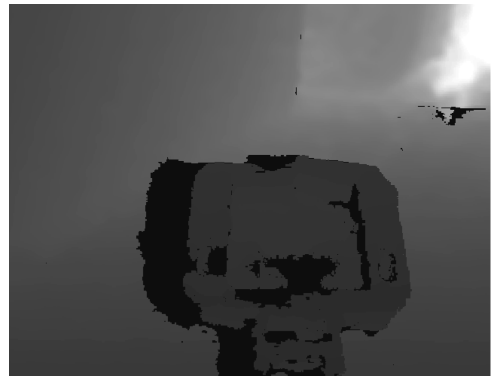

Fig 1a. Depth image of empty tray

Fig 1b. Depth image of the tray with object

Fig 2a. Cropped and warped image

Fig 2b. Cropped and warped image

Robust Object Detection:

The new environment created problems of different lighting conditions, walls, ground surface among many more. Hence the object detection pipeline had to be improved to accommodate all these differences.

There were a lot of false positives and these were removed by setting up the right thresholds for validating an object from a segmented point cloud.

Hyper-parameters were improved such as the thresholds, voxel filtering size, clustering size, etc.

The changes were tested at different positions in the new environment.

The results can be seen in Fig 3.

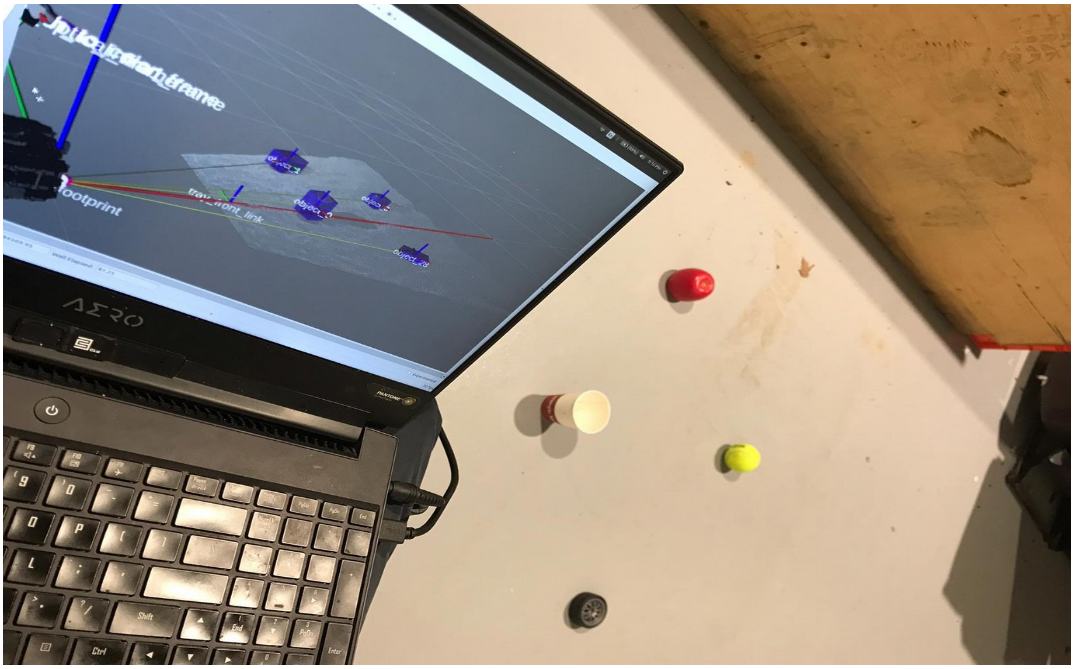

Fig 3. Real-time object detection pipeline in the new environment

Obstacle Detection:

Nithin worked on some approaches for obstacle detection.

I tried another approach to extend our current object detection pipeline to identify potential small obstacles as well based on the size of the 3D bounding box around the segmented point cloud. It could be an obstacle if its size and position is higher than the thresholds and criteria set for classifying it as a pick-able object.

Size thresholding was performed and colored markers were used to visualize the output on Rviz. This is not robust to each kind of obstacle as there can be many. Hence, it would involve future work.

Challenges

Individual challenges:

The validation pipeline with rgb images failed because of the lighting variations, mainly because of the shadows formed by the single source of light as opposed to the multiple ones in our previous testing location.

As shown in the Fig 4a. and Fig 4b. we can see the shadow effects when the gripper in rotated by 180 degrees. The shadows make edges that makes it very hard to get a consistent base/ default image of empty tray to validate with.

Lots of enhancement methods on rgb were also tried such as:

- Adaptively enhancing the contrast of the images
- Structural Similarity Index (SSIM)
- Comparing image hashes
- Canny edges comparison.

All the above approaches were not robust to all the prevalent testing scenarios.

Hence, the depth image comparison was implemented as a solution that has already been described in detail in the individual progress section.

Fig 4a. Histogram of colors of empty tray

Fig 4b. Histogram of colors of the object

The next challenge faced was that of the NaN points in the depth image formation.

They are the points that are often found around the background regions of the images because the background is prone to occlusions by the foreground objects. These are shown in Fig 5a and Fig. 5b.

These points were forced to have values of 0 in the corresponding images of depth chosen for validation. While comparing the images if a pixel with a depth value is compared with the corresponding pixel in the other image, it will add to the error difference, since anything subtracted from 0 is positive. This can prove to be bad as we just want the difference coming from the pixels occupied by an object.

Hence, to solve this problem, the NaN points in image 1 were replicated in image 2 and vice-versa.

Hence the difference of these pixels will be zero and we will get a robust pipeline which is good even in changing lighting conditions and shadows because of depth not getting affected by the rgb values.

Fig 5a. NaN points in empty tray image

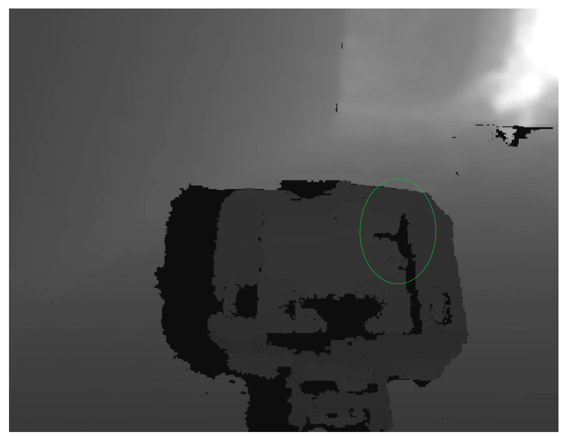

Fig 5b. NaN points in tray image with an object

Team challenges:

Integration and updating state machine:

When there are a lot of changes made by different members of the team, it requires a lot of time and effort to add new functionality to the state machine.

An abstract state machine was developed by Jorge to make it easier to modify the state machine easily to run different individual experiments without hurting someone else's code.

Project management

Due to the end of the semester and different course schedules, it is difficult to find the same time to work on problems together and integration. Better project management and a shared free-time schedule was adopted to tackle this problem.

Time with the robot

Multiple people require access to the robot at the same time to do individual tasks. It requires controlling the robot.

Bag files were efficiently used to do some work on personal laptops.

Teamwork

Following describes the work done by the team members and how I interacted with them:

Paulo:

Paulo worked on point to point path traversal with obstacle avoidance trying out the right turn following strategy. I interacted with him to discuss his implementation and explain my work for this progress review.

Bobby:

Bobby worked with Jorge on exploration and localization. I interacted with him to debug a power failure case in the robot and fixing it.

Jorge:

Jorge worked along with Bobby for exploration and localization. He also did a good job in project management. I interacted with him for making the state machine better, project management and integration work.

Nithin:

I worked with Nithin on obstacle detection. He tried a couple of ros packages for obstacle detection. I discussed various implementations with him.

Future Work

Individual plans:

1. Lots of integration work.
2. Making object detection even more robust to uneven surfaces and challenging lighting conditions.
3. Optimize the vision module to save on computing.
4. Plan and perform tests for fall validation demos.

Team plans:

1. Integration of the several new packages for exploration, planning, validation, etc.
2. End to end automation.
3. Improve individual subsystem modules independently.
4. Increase collaboration and interactions.
5. FVD demonstration preparation.