Individual Lab Report 04

Team C: [R.A.M.S]

03.28.2019

Ganesh Iyer

Teammates: Rohan Tiwari, Lukas Merkle, Karun Warrior, Stanley Winata

Individual Progress

For the past week, my individual efforts have been focused on block pose estimation.

In the previous ILR, I demonstrated segmenting the XYZRGB (color) point cloud data from the realsense to focus on a tracked block for pickup. This week, I built up on that work to estimate the pose of the block.

Estimating the pose of the block is important from the standpoint of the challenge. Our current strategy to pick up a block is based on the “Position-based visual servoing” (PBVS) scheme. PBVS is a method of estimating the pose of the camera with respect to some reference coordinate frame in order to apply a control strategy, that retains the position of the hexrotor platform with respect to a target. The general strategy is to minimize the continuous control error e(t), which is the difference between the current pose and the desired pose with respect to said coordinate frame, based purely on the visual estimation of a target. On a higher level, if we can reliably achieve estimating the pose of the block, we can now have a feedback error term as a pose difference (in translation and axis angle), that we can reduce to servo towards a block.

My method to find the pose of the block is based on estimating the principal surface normals visible to the camera, originating from the centers of the surfaces of the cuboidal block. These normals, in turn, are orthogonal to the surfaces and we can apply a least squares plane fitting method to estimate the surfaces planes. We can fit a plane by finding the covariance matrix of the nearest neighbor set of points around a given radius from the current point, and then finding the eigenvectors and their orthogonal vectors. One important consideration is the scaling parameter, that defines the k-neighborhood around the point, and thereby the quality of the estimated normals. For our purposes, we are focusing more finding the general direction of all the normals, I went with setting a larger neighborhood (setting a higher scaling value).

In terms of implementation, I continued using the PCL Point Cloud Library, for its rich API and deployment speed (such as using Kd-Trees for fast nearest neighbor search). Since the point cloud is segmented, we already have planar surfaces on the cloud. I used the Integral image normal estimation method (pcl::IntegralImageNormalEstimation), which is a faster variant specifically for organized point clouds (RGBD data generally, as opposed to data from lidar scanners). These point clouds are rectified and registered to the image. We pre-process the raw normals to remove all NAN Normals, that exist due to anomalies in the raw point cloud (such as holes, low texture regions, or occluded areas

that cast shadows across the projected light from the realsense). Once the normals are extracted, I cluster all the normals into 3 principal clusters, using a fast k-means variant. To improve speed, I use every 100th normal in the set. I also calculate the centroid of all points where valid normals exist. The clustering operation provides the 3 principal surface normals, which can be attributed to the 3 planes that a camera can view from a viewpoint. One point to note, is that in the case that there are only 2 orthogonal vectors present (due to one plane not being visible), I reject the non-orthogonal vector and consider the third vector orthogonal. These normals are now the column vectors of rotation matrix which defines the coordinate axis. We can now define the pose of the camera (thereby, the hexrotor, since the camera is extrinsically calibrated) with respect to the block and servo towards the block for a pickup. Further, the distance to the centroid defines the direct translation to the block from the camera. Figure [1] demonstrates the progress in block pose estimation

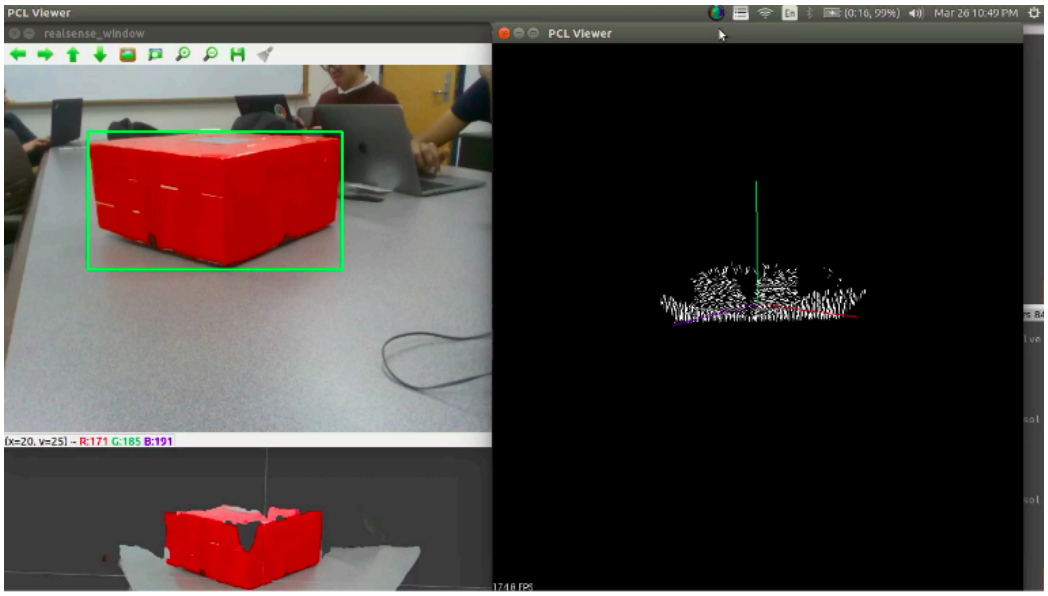

Figure 1: Block pose estimation. Detected block is shown on the left, the normals and axis vectors on the right.

Challenges

I faced one challenge that I discovered after the method was implemented. I noticed that the code has a bottleneck in the filtering operation (segmentation step), that causes some part of the cloud to be filtered out. This doesn't affect the block pose in the case of small areas, but the normals cannot be estimated especially in highly dynamic environments, i.e when the camera is moving at a slightly faster speed. The cloud readjusts, and for the

slight delay till the camera re-stabilizes, the normals are not estimated. One method that helped in mitigating this (credits to Stanley), was to directly filter and cluster the cloud in HSV space before estimating the normals. This retains the cloud as opposed to the spatial filter in the previous case and maintains the pose estimate. In the future, we will also be exploring “Image based visual servoing” (IBVS) for cases where we are directly above the block.

Teamwork

The last week, the team’s effort was heavily focused on outdoor flight tests. We showcased the stability of the platform when outdoors (with GPS lock) in offboard (autonomous) and position hold (manual) modes. For autonomous operation, we showed waypoint takeoff landing, such that the platform reaches a height of 3m, maintains its position with low drift for a few seconds, and then safely lands. We saw a slight yaw drift issue, which we hope to fix soon. In manual, we also conducted a successful experiment of traversing with the block, using the manipulator arm. The pre-attached block was made to take-off to a height of 3m, and made to place the block by switch control a few metres ahead. Figure [2] shows the same.

Figure 2: Traversing with the block, and placing it.

In terms of each person’s individual efforts, apart from the tests, Rohan was key in writing all the test deployment codes to conduct safe experiments, and I assisted him in the same. Stanley was very helpful in finding the HSV cloud filtering method for mitigating issues with the block pose estimation. Karun worked on detecting and classifying the blocks using the selective search based region proposals and logistic regression method. Lukas worked on creating a URDF file for the manipulator arm, that we will be using to simulate the challenge in Gazebo, while maintaining the dynamics as close to the real manipulator as possible.

Future Plans

In terms of individual goals for the next week, I hope to make the block pose estimation more robust. I will also work towards deploying the whole pipeline on the Jetson and check at the real time computation mid-flight.

In terms of team goals, we will be focusing on the following points in the coming week:

1. We will be exploring faster methods for detection and classification (convolutional network, or block tracking during pickup)
2. We will try to perform waypoint following for multiple waypoints and longer durations outdoors.
3. We will attempt to successfully pick up a block after takeoff.
4. We hope to tune the gains for additional stability and quicker recification. This will also be helpful for maintaining additional stability when traversing with the block.