INDIVIDUAL LAB REPORT #4

AWADHUT THUBE

Team G - The Pit Crew

Team Members : Alex Withers, Justin Morris

Individual Progress

Since the last progress review, my work has been focused on improving the 3D reconstruction from the realsense stereo pair and running the brinkmanship code with the generated pointcloud. I was able to solve the previous problems of point cloud format. I found that the method used to generate the pointcloud resulted in a sparse cloud and hence started figuring out ways to obtain dense reconstructions. I found that, instead of using feature based methods for image alignment and 3d reconstruction, a disparity map can be used to obtain a dense reconstruction.

A disparity map contains information about pixel movement between stereo images. Consider a stereo pair having two images (left image, right image). A disparity map would be an image of the same size as the individual images, and containing information about differences in pixel positions between the images. Each pixel from the left image is mapped to a pixel in the right image and the difference in position is calculated.

For calculating the disparity map, OpenCV's api for stereo vision was used. This uses the following functions.

1. StereoSGBM or StereoBM Semi-Global Block Matching or Block Matching
2. StereoCompute

Opencv's SGBM matches blocks instead of matching individual pixels. The function takes in the following parameters:

1. numDisparities - Expected max disparity - min disparity
2. blockSize - Size of the block used for matching. Increasing the value causes the matching to be reliable.

So the high level idea is that given a pair of rectified images, the algorithm would pick one pixel from the first image and search for the corresponding (most similar) pixel along a horizontal line in the second image. This eventually leads to correspondences between most pixels after some smoothing during the post processing stage. The disparity map was obtained by passing the stereo images to the compute function which could be invoked using the object created by StereoSGBM_create()

Visualizing the disparity map provides an indication of the relative depths of the objects in the image. The disparity map was then used to reconstruct points in 3D. Almost all the pixels from the image are projected in the 3D space resulting in a dense reconstruction. The reconstruction was performed using the StereoCompute function which takes in the pair of images and the camera parameters. The results are shown in Figure 1.

Fig 1: (Left) Disparity Map, (Right) Generated Point Cloud

After obtaining the position of the points in space the pointcloud2 package from ROS was used to publish the pointcloud. The data was formed as a list of lists, where a single element would be a list of length 3 (containing the x, y and z coordinates). The brinkmanship code was also cleaned by removing the predefined transforms which previously caused the whole point cloud to rotate.

Apart from this, I also started migration from the NUC to the Jetson TX2. I was able to flash the TX2 with Ubuntu 16.04 and also installed ROS on the computer.

Challenges

Firstly, having to work remotely on the project has caused a slight loss in momentum. We haven’t been able to meet in person and work on building the robot which led to the task falling mainly on one person in the team. Having access to the lab and CMU campus was critical from the point of view of testing our work on hardware. We as a team have planned to continue with other tasks and look forward to testing our systems (in the real world) in the coming weeks.

Secondly, I was working on installing the realsense SDK on the TX2. I started by looking for instructions for installation of the SDK on an ARM architecture and found that it is poorly documented. I tried installing the SDK by looking at a couple of resources which infected the file system causing me to start fresh by flashing the OS again on the computer. The installation of the realsense SDK on the ARM architecture is a very challenging task and yet to be completed.

Team Work

Alex Withers - He has been working on improving the simulation visuals in the webots simulator. Alex has been successful in loading a pit model in the simulator and also added rocks on the terrain. He has been able to run the robot navigation code on a computer in the MRSD lab and has also started looking at planning algorithms which could be used in the project. Alex also designed and printed a camera mount which would support our camera on the pan-tilt mount.

Justin Morris - Justin has been working to complete the assembly of the rover. He has assembled the pan-tilt mount to be used on the rover. He is working on setting up the controller for the AX-12 motors on the mount.

Future Plan

As the Covid-19 situation evolves, we continue to alter our plans. For now, we have scheduled a field test on April 3. We may decide not to perform the field test depending on the situation. If we go ahead and perform the test, we aim to capture data which can be used to define behaviours near the pit edge. Alternatively, if we do not perform the field test, setting up the brinkmanship pipeline in simulation would be a crucial task.

Next, we would like to have completed migration to the TX2 and test all our codes on the same. We also plan on controlling the pan-tilt mount using the rover computer (TX2) after setting up the dynamixel SDK.