Individual Lab Report #2

Akshit Gandhi

Team H (PhoeniX)

February 21, 2019

Team Mates:

Shubham Garg
Parv Parkhiya
Zhihao Zhu

Individual Progress

After the last progress review, the task was to test the position hold mode for the phoeniX – UAV. It is very important for us to have a drone with it is possible to hold the desired x-y-z position with centimeter level accuracy. So we decided to test the mode using AprilTags, specifically my task was to get the drone up and running with the AprilTag detection and using the vision data to guide the drone to maintain it's position in 3D space.

Specifically during the last 7 days I worked on the camera calibration, marker detection, finding the marker frame with respect to the camera, calibrate the IMU and camera frame to get a transform from the IMU frame to the marker frame and perform corrective measures to guide the drone in maintaining the position. I'll go into more details in the subsequent sections.

Camera Calibration:

We are using the ZED Stereo camera on our system. These cameras come with a calibration configuration/YAML file as a part of the SDK where the SDK automatically downloads the calibration file (obtained by pre-calibration from the manufacturer), but from the images which I saw, I felt a need to calibrate the camera again so I used the tool provided by ZED to calibrate the camera again and the parameters obtained by my calibration were a bit different from the one's given by the manufacturer and hence we used my calibration file.

Marker Detection:

AprilTags come with a Open-Source marker detection software and luckily there was a wrapper readily available for ROS. I integrated the wrapper with the camera data from the ZED sensor. Eventhough the wrapper was available readily there were a lot of params to be tweaked in order to get very good performance. The default performance also was acceptable but it was very important to get things right otherwise we can risk a crash which might cost us atleast a week to get back on track. For this, I read the AprilTag paper and understood the various thresholds needed to be tweaked and I also made some minor adjustments in the code so that I could use the data from the ZED sensor. By default the wrapper listens to the /image_rect and /camera_info topics to get the image and camera information, but despite remapping the data in the launch file, it did not work so I had to write a re-mapper ROS node to do this task ensuring the data being fed to the wrapper is synchronized in time. The results of this ROS node were the transform of the marker and the image with a bounding box around the detected marker.

Camera-IMU calibration:

In order to get the calibration right between the camera frame and IMU frame, we relied on a calibration tool developed at ETH Zurich named Kalibr. This was Shubhams idea and he did the calibration task and I helped him by getting the dataset for the tool. The tool uses the camera images and IMU data to fit a spline which would produce a transform matrix between the two frames. To get the tool working we need a grid of 6x6 AprilTag 36h11 markers of size 0.088 m and having a spacing ratio of 0.3. The grid was printed by Team RAMS on a A0 size paper and we used the same calibration grid.

To calibrate the data, we had to record the images of the grid with the camera facing the grid and move the drone along the roll, pitch and yaw axis so that IMU data and the images from camera could be used to find the transform matrix. The data was recorded in a rosbag which was then given to the Kalibr tool. I worked on creating the rosbag and conversion of images from RGB to grayscale for the tool.

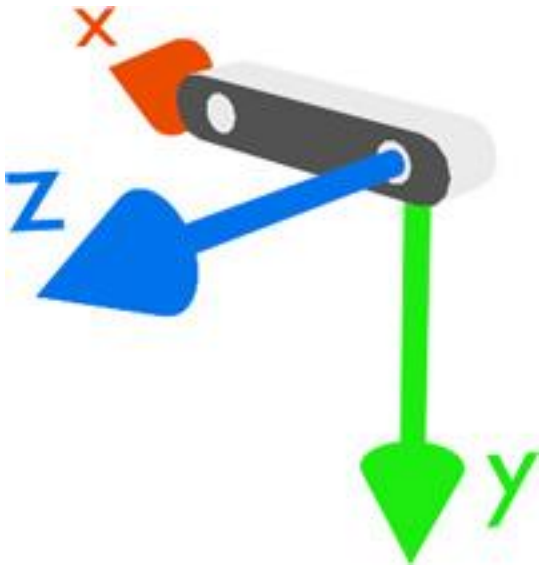

Figure 1: ZED Camera frame

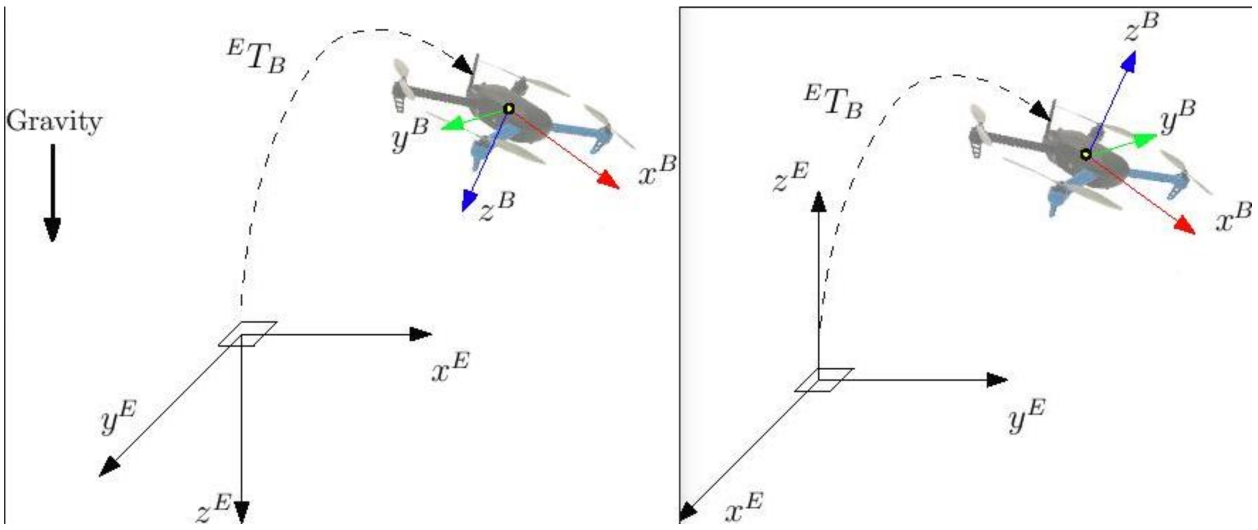

Figure 2: PX4 NED frame on left and ENU frame on right

The rosbag can also be used to get the camera intrinsics.

UAV Position Hold Code:

The homogenous transform obtained using the calibration and the marker detection are the fused together to get the world frame co-ordinates of the drone with respect to the marker. The values of X and Y translations are then taken and fed to the drone to perform counter measures in order to be exactly above the marker. The messages from the algorithm are sent to /mavros/vision_pose/pose topic in the form of PoseStamped() messages which include the translations and the quaternions. The code was tested with drone on ground but test flight is pending.

Challenges:

Since I had never worked with camera calibration and AprilTags, it was a challenge to understand how the system works. Specifically how to calibrate the camera with a checkerboard using OpenCV was something which I was using but later I found that ROS has some packages which make it much simple and ZED's calibration tool was the simplest of all. The determination of threshold for the AprilTag markers was tough as it required me to modify some of the values in the code of the wrapper. This was inherently difficult because the images obtained had some shadow of the drone as the marker was directly above the marker, although by default the AprilTag library takes care of light shadows but I saw a significant improvement by playing around with the contrast on the images being fed to the wrapper.

The default remap in the launch file did not work because the images being subscribed by the AprilTag were supposed to be published at a different rate and the images being published by the camera ROS wrapper were at a different rate along with the /camera_info not being published at the same rate as the image. It took me and Shubham several hours to debug the problem and finally I wrote a ROS node which subscribed to this data and published it back to the AprilTag wrapper at the prescribed rate.

The camera-IMU calibration tool took atleast 8-10 hours of work in order to calibrate the frames but eventually we couldn't get it working some of the challenges we faced are:

1. The tool requires a rosbag with more than 1000 images, which leads to a ROS bag of size in a few GBs.
2. The tool does not work on jetson tx2 and it needs to be run on the development machine, which requires transfer of the the ROS bag from jetson to a laptop, which took atleast 10-15 minutes for each datatset that we captured so it was time intensive task.
3. Every time we feed the dataset (ROS bag) to the tool, it would say that it couldn't detect any corners in the image despite us verifying the images showing the whole calibration grid.
4. After surfing the web for the issues, we found that it required Grayscale images of the calibration grid, we converted the ROS bag images from color to grayscale
5. The grayscale images also did not work so we tried the whole process atleast 15 times and eventually we had to move on and calculate the transformation matrix manually.
6. The images obtained with the ZED camera were seen to have dropped frames and we took some time to figure out the loss of data and since the camera is in warranty, we have initiated the warranty process. To get more insight on the video obtained please see the video attached with the link.

The ROS node developed on my laptop for the position hold did not compile on the target jetson as it couldn't find a node on which it was dependent. The dependency was already present in the source code. Later I debugged and realized that the python file was not given executable rights and changing the rights with chmod +x made the node work. Verifying the results on ground led me debug more issues in the code and with this code I ported it back to the ROS Action server and ROS safety bond architecture which we are implementing as a part of the project. This produced more issues as I integrated the code with the existing take-off and land code more specifically with the TaskServer not spinning my object of marker tracking which despite being scheduled used to get cancelled as the drone couldnot see the marker so now I have solved that issue with marker initially not being in sight.

Teamwork:

The team is split into two smaller teams where Parv and Zhihao (Steve) are working on the husky where they were trying to get the husky running and move around using the IMU data. They had to connect the Zotac PC to the monitor every time to work over SSH as they had to get the IP, but now I helped them setup a static IP on CMU-DEVICE wireless network. Specifically, Parv’s responsibility was to interface the IMU and calibrate it along with MAG. Zhihao was responsible for writing subscriber for the IMU data and write code along with Parv in order to perform some basic tasks on the Husky like moving forward and backward along with rotating 360 degrees as a part of the presentation on progress review.

The other two members i.e me and Shubham were working on the UAV where Shubham helped me out with the marker tracking and understanding of the camera calibration process. Shubham was also responsible to create the tf tree which was important for me to get the world co-ordinates of the drone and feed it into the position-hold code.

Future Plans:

The future plans for the next presentation is to simultaneously work on the SLAM and creation of dataset and model training for the fire detection pipeline. Parv and Shubham will be working on ORB-SLAM and my and Zhihao’s task is to find/create a dataset for the fire detection pipeline and use classical or deep learning methods to detect the regions having potential fire.