MRSD Project Course

Team I – AIce

Autonomous Zamboni Convoy

Individual Lab Report 7

Team

Rathin Shah

Nick Carcione

Yilin Cai

Jiayi Qiu

Kelvin Shen

Author

Kelvin Shen

Sep 29, 2022

Table of Contents

Individual Progress

During the past two weeks, I have been working on two tasks mainly. One is obstacle detection, and the other is integration of the previous perception system onto ATV.

Obstacle Detection

Obstacle detection is one last subsystem in our stack that is not implemented. We originally planned to pass the detection result into the planner such that the follower will avoid the obstacle accordingly. However, after discussion about the project scope with one of the stakeholders, Zamboni, we were informed that avoiding obstacles by contouring is not eligible because it might cause defective ice resurfacing. Therefore, we decide to stop both the leader and follower whenever there's an obstacle blocking the way. To this end, all we need for our system is the relative location of the obstacle to the follower's ego. Once we output the relative location, we check whether it intersects the trajectory that the follower is following, and if so, we stop the follower while the driver of the leader vehicle will be notified and stopped as well.

In order to calculate the relative position of obstacles, we have two options. The first is to leverage the depth stream on RealSense D435i that we are already using, fuse it with outputs of any object detector such as YOLO, and unproject into 3D to get obstacles' relative positions. However, the depth accuracy of D435i will drift noticeably after 3 to 4 meters away from the camera, known as RMS error, which impacts the D435 model more than the D415 model due to D435's hardware design. In our use case, the longitudinal offset between the leader and the follower will be 6 meters, and obstacles can appear anywhere between, which makes D435 an unreliable choice of depth estimation. The other option is to calibrate camera and LiDAR, filter out points that are associated with outputs of any object detector, and directly get obstacles' relative positions. This option will be much more robust than purely relying on D435i because the range of VLP-16, the LiDAR we will use, can go up to 100 meters. In addition, taking the second approach expands our knowledge as well as hands-on experience of sensor fusion.

Since the extrinsics of the LiDAR and the camera are unknown on the ATV, we use the *lidar_camera_calibration* package to calibrate VLP-16 and D435i. In particular, it finds a rotation and translation that transform all the points in the LiDAR frame to the monocular camera frame. It finds the rigid body transformation between the camera frame and the LiDAR frame by corresponding manually annotated checkerboard's edges in point clouds with the detected checkerboard in images. As shown in Figure 1, we use an online available dataset coming with the package that contains the point cloud associated with the image of two ArUco markers in order to experiment and get familiar with the package. Given the point clouds of two boards, we manually annotate each edge of the board in clock-wise order starting from the top-left, which shows up in the top left window in the screenshot on the right. Then the package uses 3D-3D point correspondences and gives a closed form solution of the rigid body transform between LiDAR and camera frame.

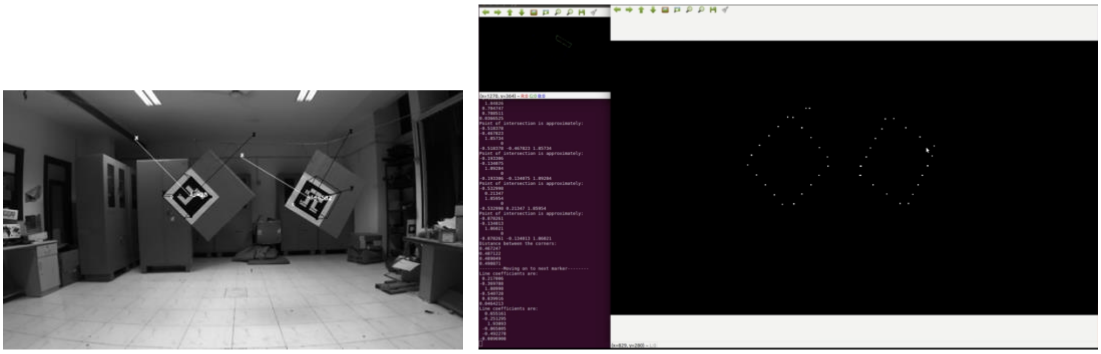

Figure 1. LiDAR-Camera Calibration

The next step will be calibrate our own sensor set after we properly mount D435i onto the ATV. And then we plan to use YOLOv4 for object detection in image frame, and filter the point clouds associated with detected objects. Finally we will remove the points associated with the leader vehicle based on the ArUco detection and output the centroids of the remaining points to the downstreaming planner.

Perception Test

Last semester we found the leader-following is very unstable because the perception subsystem is not robust against noise. Therefore, starting this semester, we start stress-testing RealSense so we can understand its limits better and complement them with any post-processing accordingly.

First we placed the ArUco marker board 6 meters away from the camera per the performance requirement. Then we tested the marker detection performance by moving the board or moving the camera respectively. When moving the camera, we tried rotating the camera as well as translating longitudinally. We found the detection was lost only when the camera was rotating aggressively, i.e., when there was a large angular acceleration (shown in Figure 2), which leads to significant motion blur in the image. This is crucial to our system because the pose estimation completely relies on a robust detection of the marker; the marker won’t be detected even if it gets a little blurred. Any fiducial marker library is based on the correspondences obtained from the clearly detected corners. This explained why the RC car in SVD was not able to follow the leader smoothly or converge back to the steady state once it started wobbling.

There are three ways that we can combine together to resolve this. First, we will increase the marker size relative to the board size so that even under motion blur, the corner correpsondences are still legible. Secondly, we find that we can increase the frame rate of the camera by switching from the RGB stream to the near-infrared stream. In SVD we used the RGB stream of D435i for detection, which consumed a lot of bandwidth, supported only 60 FPS as its highest frame rate and easily caused jello effect that distorted markers. On the other hand, near-infrared stream that publishes monochrome images requires much less resources than RGB and it supports as high as 90 FPS in normal resolution (840x480). Since RGB info is useless to either pose estimation or obstacle detection in our use case, switching to the near-infrared stream is a matter of course. Lastly, we plan to implement EKF for pose estimation to account for any loss of detection even when a higher FPS is used. In particular, during the prediction step, we will extrapolate the leader pose based on the history of estimated poses and the timestamp, while in the update step, we will correct the prediction using the successful marker detection.

Figure 2. Left: Successful marker detection. Right: Losing detection under motion blur

The other alternatives we have discussed to resolve loss of detection are: (1) use learning-based recognition methods to detect the marker even when the image is blurred. Deep learning can indeed increase the robustness of detection, but it is not usable in our case because even if the position of the marker is detected inside the image, the fiducial marker library is still unable to estimate the pose based on its pre-known correspondences. (2) use ATV’s onboard camera which is designed for long-range detection. However, it is a legacy product and hence not scalable to Zamboni. Plus, the highest FPS of the onboard camera, Multisense S21, can only go up to 60 FPS, which is less promising than the IR stream on D435i.

Challenges

The major challenges I have encountered are:

- Increasing robustness of marker detection is not a trivial task. As explained in the previous section, our downstreaming subsystems largely depend on a good estimation of leader's relative pose, without which the planner will easily lose track, start wobbling and diverge from the leader's trajectory. To resolve it, I have brainstormed six options that are covered above and come up with the plan that involves little reworking and computation cost.
- In order to decide which set of sensors we should use as well as what available algorithms we can find to detect obstacles, I have done a simple trade study between purely camera-based approach versus LiDAR-camera fusion approach.

Teamwork

- Nick actively designed mounts necessary to mount camera onto ATV as well as both camera and LiDAR onto Zamboni. He also took charge of the project management where he updated the WBS and JIRA issues frequently.
- Rathin coordinated with Isuzu and Zamboni to ship the vehicle. He tested the localization stack of the ATV using ros bag files provided by the original ATV project members.
- Yilin reviewed papers to come up with potential methods for leader-following controller that maintains a constant offset. He worked with Jiayi to develop the PID longitudinal controller.
- Jiayi implemented the PID controller to maintain constant longitudinal distance. She also wrote scripts that publish steering and velocity commands to ATV. She continued to coordinate with Isuzu to get latest updates on their DBW conversion.

Plans

Before the next progress review, I plan to calibrate our VLP-16 and RealSense D435i after we mount the camera onto ATV. I also plan to finish the fusion of LiDAR-camera by using YOLOv4 for object detection. Simultaneously, I will work on the EKF to improve robustness of the leader pose estimation.