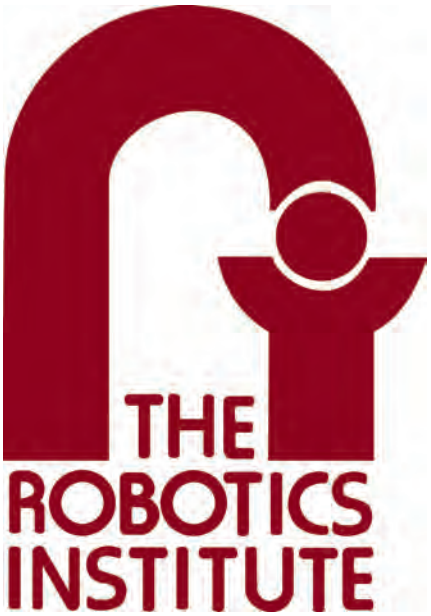

MRSD Project Course

Team I – AIce

Autonomous Zamboni Convoy

Individual Lab Report 4

Team

Rathin Shah

Nick Carcione

Yilin Cai

Jiayi Qiu

Kelvin Shen

Author

Jiayi Qiu

March 24, 2022

Contents

1 Individual Progress

In the previous progress review, I developed the relative leader velocity estimation algorithm. In this progress review, I should be able to estimate leader's velocity in follower's odometry frame based on the estimated leader's global pose. However, we realized that it's not correct for the follower to use the leader's velocity profile directly. As shown in Figure 1, if the follower use the leader's velocity profile from the beginning, the follower would not reach the turning point when the leader turns. The velocity of the leader during the turn cannot be applied to the follower, since it still moves along a straight line. If the follower starts using leader's velocity profile after reaching the position with a lateral offset from the leader's initial position, they should still have different velocities during the turn. That's because the the follower and the leader have different radii when making turns. Follower in this example needs to move faster than the leader during the turn, so that it can keep up with the leader.

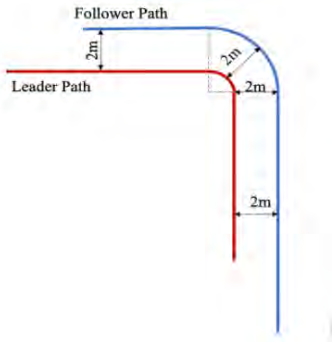

Figure 1: Example of leader's path and follower's path

Therefore, I implemented an algorithm to estimate follower desired velocity using an Extended Kalman Filter based on the follower desired poses generated by Yilin. The desired pose is generated by adding an lateral offset on leader's pose. The state equation is

$$X_{k+1} = \begin{bmatrix} x_{k+1} \\ y_{k+1} \\ v_{k+1} \\ \theta_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & dt\cos\theta_k & 0 \\ 0 & 1 & dt\sin\theta_k & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_k \\ y_k \\ v_k \\ \theta_k \end{bmatrix}$$

where x, y is position, v is velocity, θ is heading angle, and dt is the time step. One sample result is shown in Figure 2, the red profile represents the velocity calculated by dividing the distance between two adjacent waypoints with the time step. The blue profile is the estimated follower desired velocity using EKF. The green profile is the leader's ground truth velocity.

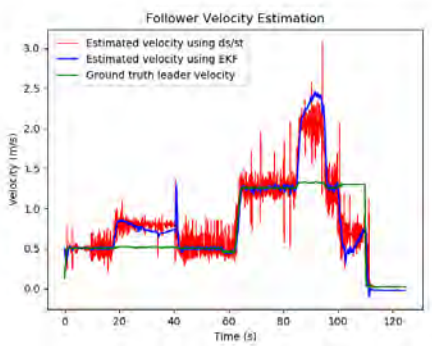

Figure 2: Follower velocity estimation

I also implemented a ROS node code which subscribed the follower desired pose, estimated follower desired velocity using EKF, generated waypoints as follower desired trajectory. For the initial gap between the leader and the follower shown in Figure 3, I designed some waypoints for the follower to move at a constant velocity of 0.5 m/s. In this figure, the green path is the leader's path. Red path is the follower desired path obtained from leader's path. The blue path is the path I designed for the initial gap.

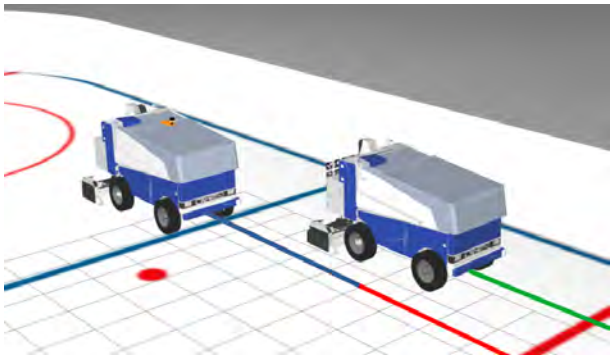

Figure 3: Path for initial gap between leader and follower

In another ROS node, I subscribed the topic of follower desired trajectory published in previous node. Then, a KDT Tree was generated in each loop to find the waypoint closest to the current position of the follower. Starting from the closest waypoint, a constant number of waypoints were published as the local path for the follower to track.

I implemented a pure pursuit control algorithm for simulation. I collaborated with Yilin to simulate the leader follower convoy. In our simulation, the lateral offset was 2 meters and the initial longitudinal offset was 6 meters. The initial positions of the two Zamboni vehicles were set near the ice rink boundary instead of the center. The leader path, follower desired path, follower local path, and follower actual path were visualized in RViz. A screenshot for the simulation is shown in Figure 4.

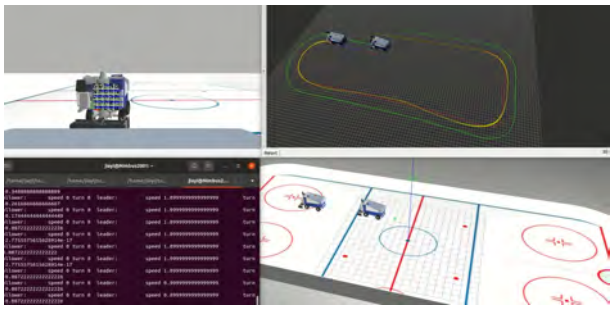

Figure 4: Leader follower convoy in simulation

2 Challenges

As described in Individual Progress part, the leader's velocity profile cannot to applied to the follower directly. It took me some time to figure out that I could directly estimate the follower desired velocity based on its desired pose.

Another challenge is that there is an initial section of path without any waypoints. That is because the follower's desired path is generated by adding an lateral offset to the leader's path. After running the pure pursuit control algorithm, the follower would move at a constant velocity until it reaches first target point.This constant velocity is the same as the velocity of the

first leader waypoint stored, which is usually very small. As a result, the longitudinal distance between the two vehicle would become too large. TO solve this, I designed a constant number of waypoints for this initial gap. The follower is able to move at a reasonable constant velocity until it reaches the position with a lateral offset from the leader's initial position. In this way, the longitudinal distance will not become too large.

The challenges I faced during the integration of subsystems for leader follower convoy simulation are from perception. The relative pose between the Aruco board frame and the camera frame was not accurate enough. This led to obvious errors in leader pose estimation. It turned out to be an misunderstanding of the OpenCV document. The returned pose of perception is the board pose in the camera frame, which we interpreted as the camera pose in the board frame. Although the problem solved at the last minute, we used the ground truth of perception to estimate the leader's pose in the simulation this time. Another challenge from perception is the Aruco marker may dissapers from the filed of view of the camera when taking turns. An example screenshot is shown in Figure 5. The situation was improved in simulation by making large curved turns and slowing down. To thoroughly solve this, sensor fusions and multiple Aruco boards are needed.

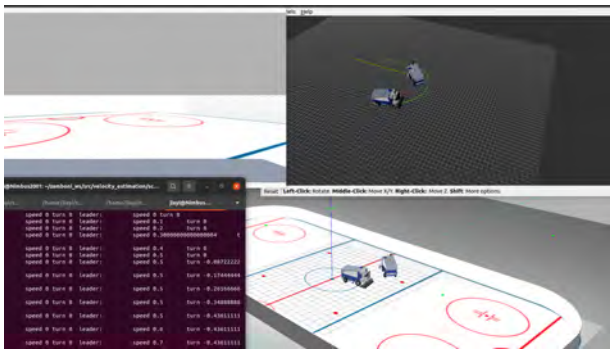

Figure 5: Detection failed situation

3 Teamwork

Each team member's distributions are shown below:

Rathin Shah:

- Gear design for encoder
- PDS design for rc car power distribution
- Waypoint follower testing and stanley controller validation

Nick Carcione:

- Designed mount and gear for encoder
- Mounted encoder onto RC car
- Figured out how to get RC car to move in (sort of) a straight line
- Did waypoint testing with Rathin

Yilin Cai:

- Setup simulation and visualization for autonomous convoy demo

- Generate the leader's path by integrating the localization, perception and tf broadcast. Tested with Kelvin the perception accuracy in path generation by comparing with ground truth.
- Planned motion for follower Zamboni based on the leader's path
- Integrated autonomy subsystems to conduct the convoy demo.

Jiayi Qiu:

- Implemented real-time follower desired velocity estimation using EKF
- Implemented ROS node to publish the follower's desired trajectory
- Generated follower's local trajectory for low-level control
- Simulated leader follower convoy using pure pursuit control with Yilin

Kelvin Shen:

- Set up Velodyne's Puck in simulation. Converted the published PointCloud2 message into PCL format. Downsampled, filtered, and clustered the point cloud to get only two clusters, one for the leader and the other for the follower.
- Fixed the incorrect transform from the board frame to the camera frame with Yilin
- Printed a 42in-by-42in marker board with the help of SCS Poster Printing
- Calibrated IMU on RealSense with Rathin

4 Plan

I plan to simulate leader-follower autonomous convoy with correct estimation from perception. The motion planning and the controller of the follower still need further improvement. I will improve them to maintain the distance between the leader and follower. One possible solution is to apply a PID controller for the longitudinal control. I will also work with my teammates to integrate ROS, Jetson and Arduino on the RC car. What's more, I plan to test the follower velocity estimation on the RC car.