Automated Driving Using External Perception

Individual Lab Report - ILR07

September 28, 2023

Team E - Outersense

Author:

Ronit Hire

Team Members:

Atharv Pulapaka
Dhanesh Pamnani
Jash Shah
Shreyas Jha

Carnegie Mellon University

Contents

1 Individual Progress **1**
1.1 Perception 1
1.1.1 Multi-Object Tracking 1
1.1.2 AruCo based tracking 1

2 Challenges **3**
2.1 Slow Detections 3
2.2 Online muSSP 3

3 Team Work **3**

4 Plans **4**

1 Individual Progress

The goal of this PR was to make incremental progress towards scaling our system. We plan to proceed by first scaling the software components (perception and controls) to handle multiple RC cars and then move on to hardware scaling which involves lengthening our track and adding a third infrastructure sensor. My efforts for this PR were focused on reviewing existing methods on Multi-Object tracking and putting together an AruCo based tracking algorithm.

1.1 Perception

For either independent or centralized control of each of our RC cars, we need to be able to distinguish them on the perception end as well as associate and maintain unique ids. This is a well known problem in the computer vision community and the solutions to tackle it range from deep learning methods to graph algorithms. I reviewed some of these methods to find a suitable approach for our use case.

1.1.1 Multi-Object Tracking

Current state of the art algorithms in Multi-Object Tracking (MOT) are primarily based on graph neural networks, the most recent of them being "Unifying Short and Long-Term Tracking with Graph Hierarchies". It opens the avenue for long term tracking in videos through "clip-splitting" and proposes a new model SUSHI for long-term and short-term data association. While recent and generalized, the approach is too computationally heavy to be tractable for our case. There is also the problem of pre-trained weights of their model being available only for certain types of datasets and those do not work in our scaled down toy environment.

Most of the classical approaches model data-association in tracking as a min-cost flow problem from source to sink where each node represents an independent detection and an edge represents the cost of association. The algorithms typically assume non-overlapping trajectories and minimize the cost of adding edges from source to sink. "MuSSP: Efficient Min-cost Flow Algorithm for Multi-object Tracking" offers a fast and reliable implementation and looks promising for our setup. However, there are several other problems with this approach which are highlighted in the challenges section.

1.1.2 AruCo based tracking

The aforementioned approaches offer a generalized solution to the data association problem but we can sidestep the problem in our project by simply pasting AruCo markers as unique identifiers on each of our cars. We then just have to track these markers for pose estimation. This was our backup solution in case none of the other approaches work. However, since this activity blocks testing of the controls subsystem, I prioritized the implementation of this method over a generalized solution. I updated our old Aruco detection function to return centers of specific ids and started Lucas-Kanade optical flow on each of them independently. The result of Aruco based tracking can be seen in the figure below. I also handled the parallel processing of these functions to avoid computation delay.

The old interface between perception and control could only handle information about a single object. I modified the ROS message structure to handle multiple as well as variable number of objects. The current implementation of information exchange through publishers and subscribers

is inefficient as an a new publisher is spawned for each tracked object. I will try to fix this in the next PR.

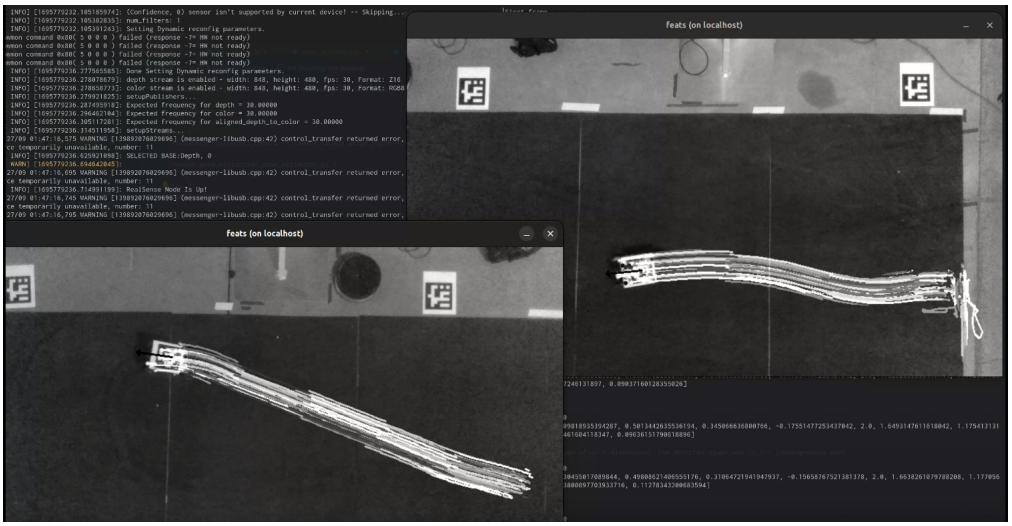

Figure 1: Multi-object tracker tracking 2 different cars in different cameras

2 Challenges

2.1 Slow Detections

The current tracking algorithm depends on being able to detect AruCo's with specified IDs at a constant rate of 10 Hz. However due to strange lighting conditions in the NSH B-level, AruCo detection is a bit flaky and unreliable. The problem is compounded by the fact that the OpenCV Aruco Detection module attempts some unecessary image optimization if it senses lighting changes. This slows down the detection module which in turn affects tracking. Since AruCo based tracking is our backup approach and since we are not fully committed to it, we will keep this on the back-burner for now and focus completing other methods which will automatically eliminate this problem.

2.2 Online muSSP

The open-source implementation of muSSP (section 1.1.1) can only handle offline tracking i.e. it takes in detection over the span of the entire video once at the end and computes tracking ids as an aftermath. We will be needing an online version of the algorithm which can give us tracks in real-time. Also there is no existing ROS package for the algorithm. Since the approach looks promising, we will be attempting to understand the algorithm and re-implement it with added functionality. Creating a ROS package should be straightforward once we have our own custom working code.

3 Team Work

Along with individual perception tasks, we as a team also made progress on Control, Simulation and Planning fronts.

- **Jash Shah**: Jash started building a minimal Gazebo simulator which replicates our test environment, ackermann RC car and ROS message data flow. This will help in testing the planning subsystem as it can sometimes get difficult to test on hardware.
- **Shreyas Jha**: Shreyas worked on understanding the odometry information published by our low-level motor controller (VESC). He had to port VESC ROS2 packages to ROS1 for compatability with our system.
- **Dhanesh Pamnani**: Apart from handling project management and logistics, Dhanesh has started an extensive review of the Planning subsystem. He is working with Jash on a Voronoi diagram based path planner.
- **Atharv Pulapaka**: Atharv scaled the controls block to handle multiple cars. He can now independently generate control commands for each RC car by running separate instances of MPC. He also designed the planner to controls interface which will help us in the near future.

4 Plans

Moving ahead, my individual goals are:

- Optimizing long term object-tracking with AruCo's to reduce system load.
- Implementing global data association with MuSSP for consistent long term tracking.
- Generating an occupancy grid with poses of all the detected objects in the scene.

As a team our goals are:

- Setup new track with three infrastructure sensors and work towards testing things end-to-end quickly.
- Improve the pose estimates coming from the perception system.
- Implement a simplistic planner to handle collision avoidance between controlled vehicles.