Oregon State University

CS 461: Senior Capstone

Tech Review

Khuong Luu

Fall 2018

Contents

1 Introduction

1.1 Terminologies

- **Lidar** A visual sensor that fundamentally works like a radar but uses lasers instead of radio waves
- **GPS** Global Positioning System
- **IMU** Inertial Measurement Unit
- **ROS** Robot Operating System
- **SSD** Single Shot Multibox Detector
- **mAP** mean Average Percision
- **SIFT** Scale Invariant Feature Transform
- **SLAM** Simutaneously Localization And Mapping
- **EKF** Extended Kalman Filter
- **GFR** Global Formula Racing

1.2 Problem Introduction

Our team is responsible for constructing the vision, mapping, and localization components of a driverless formula style racecar for the Global Formula Racing (GFR) team - a collaborative team between Oregon State University and DHWB university in Germany. This racecar is expected to compete in the Formula SAE (FSAE) competition in August 2019. In this competition, the driverless racing car must complete 4 types of track with as fast speed as possible while not violoating competition rule such as going out of track. The four types of tracks are skidpad, straight-drive, autocross, and trackdrive. In skidpad, the vehicle must complete several laps around a figure-eight style track. In straight-drive, the car must accelerate in a straight line as fast as possible. In autocross, the vehicle must complete one lap around a custom track as quickly as possible. And in trackdrive, the vehicle must complete 10 laps around a custom track as fast as possible. Therefore, our team's responsible is to implement a perception system that consists of lidar vision, camera vision, sensor fusion (or sensor statistical analysis), and simulaneous localization and mapping.

1.3 Scope, context, and requirement

First, the vision consists of two sub-components: laser vision and visual vision. The laser vision get the scanning data from the lidar attached in front of the car. The lidar spins 360 degree to scan the environemtn around the car. Similarly, the visual vision component will get the visual data from the camera attached on the high top of the car. These two sub-components, which form the vision component, help the car locate the relative locations of the cones on two sides of the course which guide the car on the track. The cone locations and colors will be used to construct a dynamic map of the world, updating the map as the vehicle moves and new cones are seen. As the vehicle moves, our system will need to update the position of the vehicle within the map. The location of the vehicle and cones will be used to compute the optimal path and velocity to successfully maneuver around the track.

1.4 My role

My role in the team is develop ROS nodes for cone detection and localization with lidar, and then send those cones locations to the SLAM component, for which three of my other teamates are responsible.

2 Cone Detection with Lidar

2.1 Tranditional Template Matching methodology

This approach collect and use some common pattern from lidar point cloud as template, and then for each incoming potential pattern data for a cone, this approach uses these templates to determine if this is a cone or not, and what kind of color it is, based totally on the existing template and how closely they match with them. This approach is not good since there can be a lot of variation of cone pattern in lidar point cloud data since we may not be able to obtain the exact cone; second, lidar data is very inconsistence.

2.2 Tranditional Machine Learning

There has been work on this topics based on this approach. This approach is using traidition machine learning method, in which the most prominent method is Support Vector Machine (SVM), to build a model about yellow cone and blue cone. However, this approach archive low accuracy on poor condition of raining weather in which the laser did not refect off the cone fully and it missed a lot of data.

2.3 Modern Machine Learning - Deep Learning

This approach build a deep learning model on cones using convolutional neural network (CNN).

3 Platform Infrastructure and Toolings

3.1 Platform

There are several choice for robotics platform. Generally, we can either build our own platform or choose an existing one. For the later options, there are some options such as Mobile Robot Programming Toolkit (www.mrpt,org), Microsoft Robotics Developer Studio(MRDS), CARMEN, and ROS. In these options, ROS is the most mature and has the biggest community and existing development, and most importantly, our client already decided to use ROS and all of the existing packages are written on top of ROS, we decided to use ROS, too.

3.2 Programing Language and Toolings

There are two major options: Python and C++ Because the scanning data from the lidar (hardware) must be, in real time, computationally processed by a lidar processing algorithm, then gives out the cone detection results, send this result to the Sensor Fusion component of the system. All of these must happened successfully in real time. ROS can work with both Python and C++. Python will help us get the prototype up and running quickly, and also most of machine learning model implementations and deployment pipelines are written in Python. In the other hand, we can archive better performance with C++, and also most of existing implementations in the ROS ecosystem are written in C++. Therefore, to get the benefits of both words, choosing Python for early developement state, especially those that involve machine learning and algorithm implementation, and then rewrite the performance critical part in our codebase to C++ in perfecting and completing state (in the later half of Winter and the first half of Spring, hopefully)