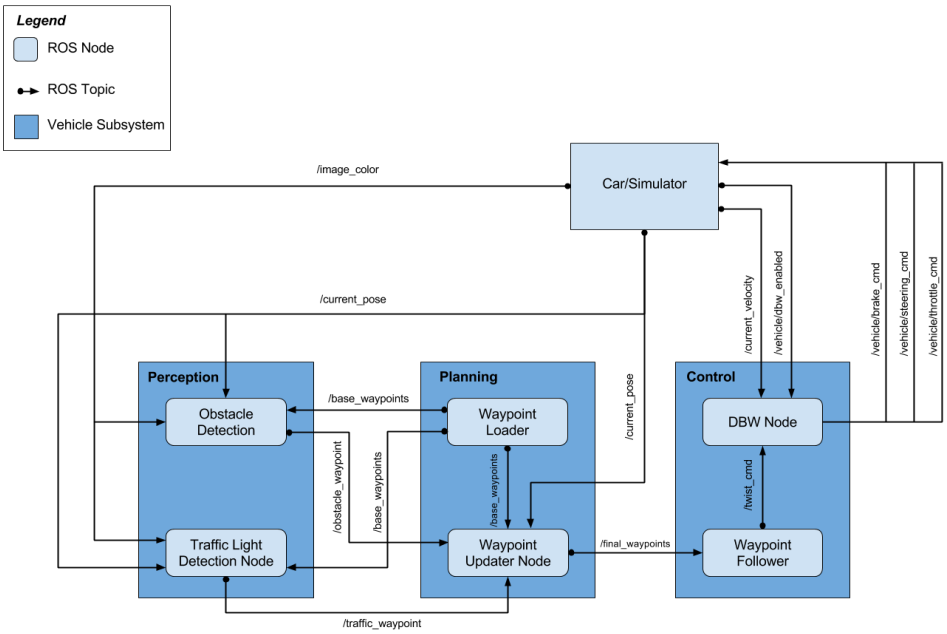
Team

Kerem Par : kerempar@gmail.com (Team lead)
Ali Ufuk Peker : aufukpeker@gmail.com
Ruixuan Li : liruixuan.xidian@gmail.com
Lakshmi Narayana : j.lakshmi.narayana@gmail.com

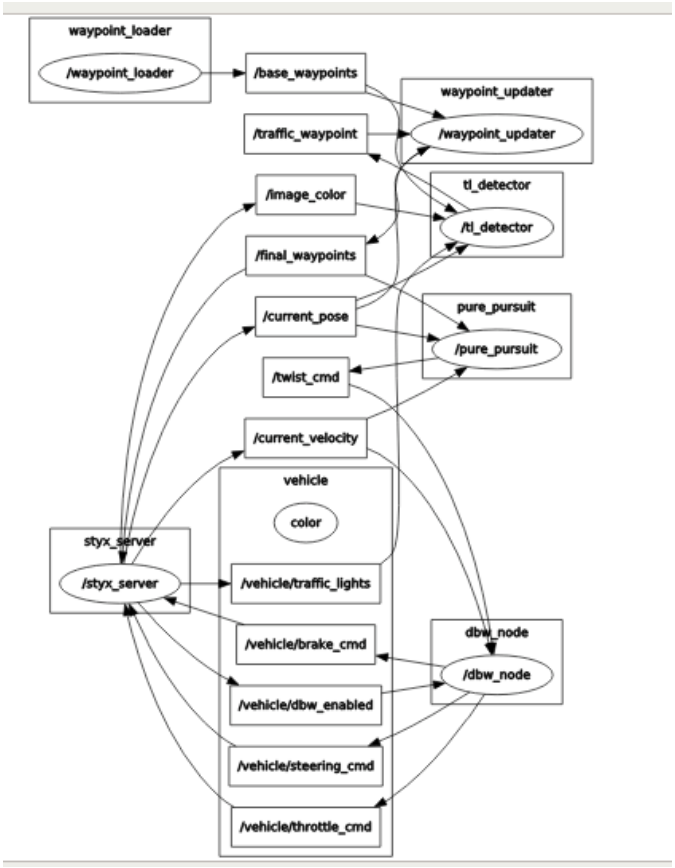
Software Architecture

Following architecture is used.



Obstacle detection was not included in the project target.

Following was the resulting ROS module/topic configuration



Implementation

Following 5 modules were implemented.

Traffic Light Detection

TL_detector

The implementation in the project walkthrough is used. First the traffic light status is not detected but directly retrieved from the way points data.

TL_classifier

After other modules are tested with the initial implementation. A simple detection using a search for red color in the image sent from the simulator is used. This is implemented in the tl_classifier. It simply searches the colors in the image. Normally a fully convolutional network can be used for the light detection.

Planning

Waypoint_updater

The code in the walkthrough is used.

Control

DBW_node

The code in the walkthrough is used

Twist_controller

The code in the walkthrough is used

Tests

- Vehice was tested at 20 km/h, 30 km/h, 40 km/h, 50 km/h and 60 km/h in the simulator. It is observed that the vehicle smoothly followed the waypoints in the simulator and respected the target top speeds set for the waypoints (It kept the target top speed as 40 km/h for the speeds set higher than 40 km/h).

- Stopped at traffic lights when needed.

- Stopped and restarted PID controllers depending on the state of dbw_enabled.

- Published throttle, steering and brake commands at 50Hz.

We experienced system latency when the camera is used.