Self Driving Car Nanodegree - Capstone Project : System Integration

Team :

- Jun Zhang, team lead
- André Bonatto
- April O`Neil
- Fabian Hertwig

Project objectives

The objective of this project is to implement ROS-based core of an autonomous vehicle. The vehicle shall be able to complete a closed-circuit test-track, detecting the traffic lights and stopping whenever required. The code will be evaluated in a Unity simulator and a real-world Lincoln MKZ. More details on the project can be found here.

Specifications

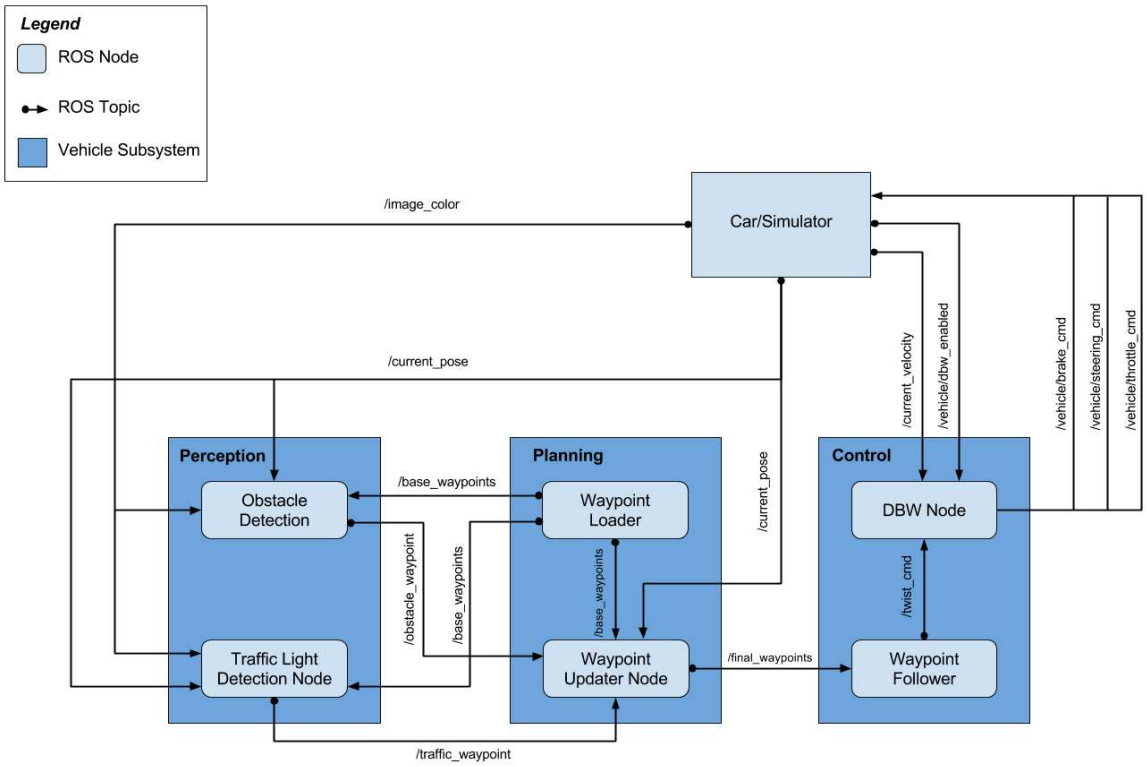
The car should :

- Smoothly follow waypoints in the simulator.
- Respect the target top speed set for the waypoints' twist.twist.linear.x in waypoint_loader.py.
- Stop at traffic lights when needed.
- Stop and restart PID controllers depending on the state of /vehicle/dbw_enabled.

- Publish throttle, steering, and brake commands at 50hz.

ROS Architecture

The autonomous driving system is composed of perception, planning and control. The modules communicate according to the following ROS structure of nodes and topics :



Build Instructions

Please use **one** of the two installation options, either native **or** docker installation.

Native Installation

- Be sure that your workstation is running Ubuntu 16.04 Xenial Xerus or Ubuntu 14.04 Trusty Tahir. [Ubuntu downloads can be found here](#).
- If using a Virtual Machine to install Ubuntu, use the following configuration as minimum:
  - 2 CPU
  - 2 GB system memory
  - 25 GB of free hard drive space

  The Udacity provided virtual machine has ROS and Dataspeed DBW already installed, so you can skip the next two steps if you are using this.

- Follow these instructions to install ROS
  - ROS Kinetic if you have Ubuntu 16.04.
  - ROS Indigo if you have Ubuntu 14.04.
- Dataspeed DBW
  - Use this option to install the SDK on a workstation that already has ROS installed: One Line SDK Install (binary)
- Download the Udacity Simulator.

Docker Installation

Install Docker

Build the docker container

```
docker build . -t capstone
```

Run the docker file

```
docker run -p 4567:4567 -v $PWD:/capstone -v /tmp/log:/root/.ros/ --rm -it
```

Port Forwarding

To set up port forwarding, please refer to the instructions from term 2

Usage

1. Clone the project repository

```
git clone https://github.com/udacity/CarND-Capstone.git
```

2. Install python dependencies

```
cd CarND-Capstone
pip install -r requirements.txt
```

3. Make and run styx

```
cd ros
catkin_make
```

```
source devel/setup.sh
roslaunch launch/styx.launch
```

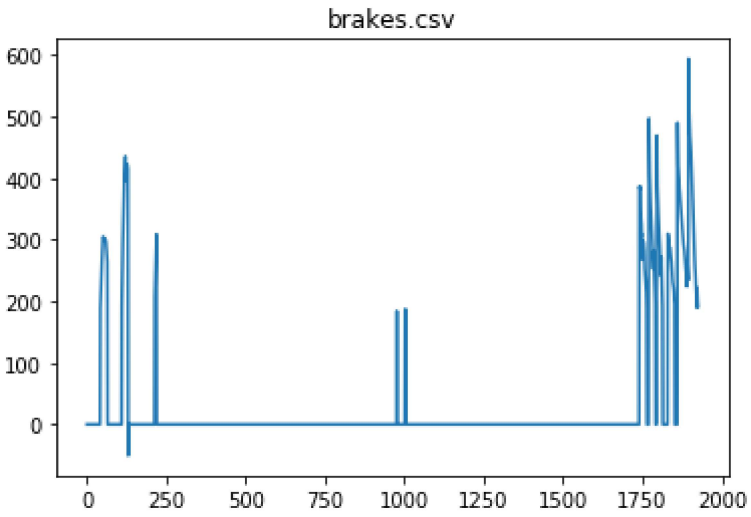
4. Run the simulator

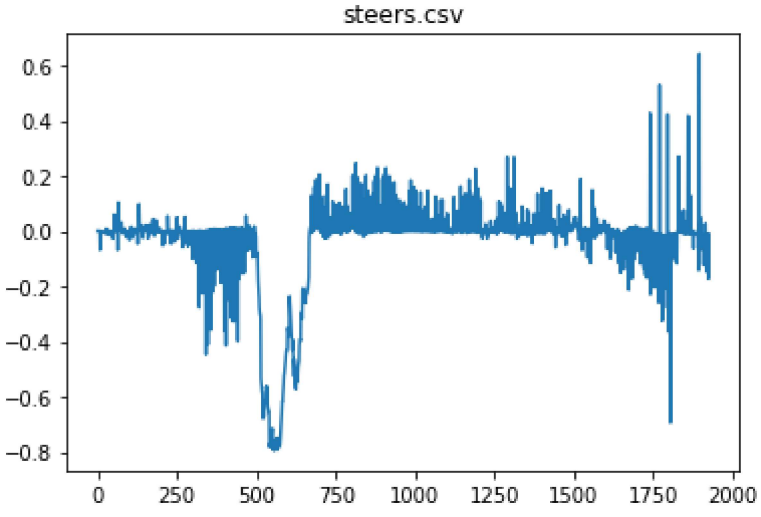
Testing the implementation

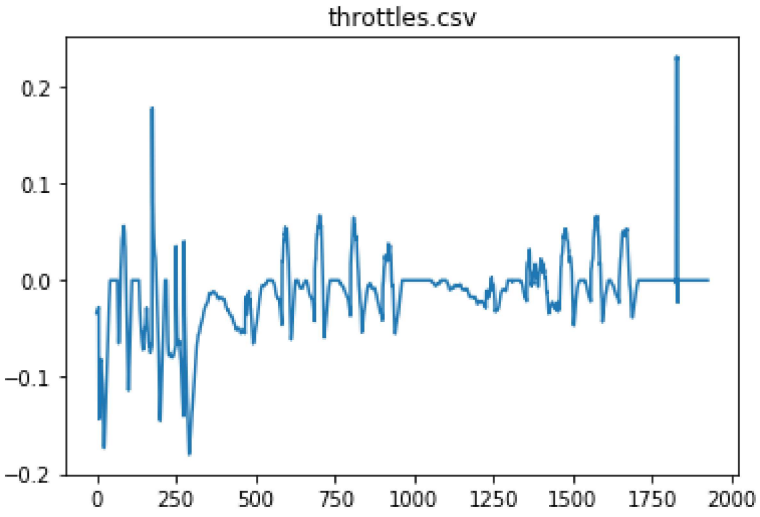
Drive-by-wire testing

1. Download the dbw bag
2. Unzip the file to CarND-Capstone/ros and rename it to dbw_test.rosbag.bag
3. source ros/devel/setup.sh
4. roslaunch ros/src/twist_controller/launch/dbw_test.launch

This will produce the files brakes.csv, steers.csv and throttles.csv, comparing the reference command with the current implementation.







Traffic light detection with real world images

1. Download training bag that was recorded on the Udacity self-driving car.
2. Unzip the file

```
unzip traffic_light_bag_file.zip
```

3. Play the bag file

```
rosbag play -l traffic_light_bag_file/traffic_light_training.bag
```

4. Launch your project in site mode

```
cd CarND-Capstone/ros
roslaunch launch/site.launch
```

5. Confirm that traffic light detection works on real life images