

Building a Self-Driving Car

Aadesh Neupane
Najma Mathema



Problem Statement

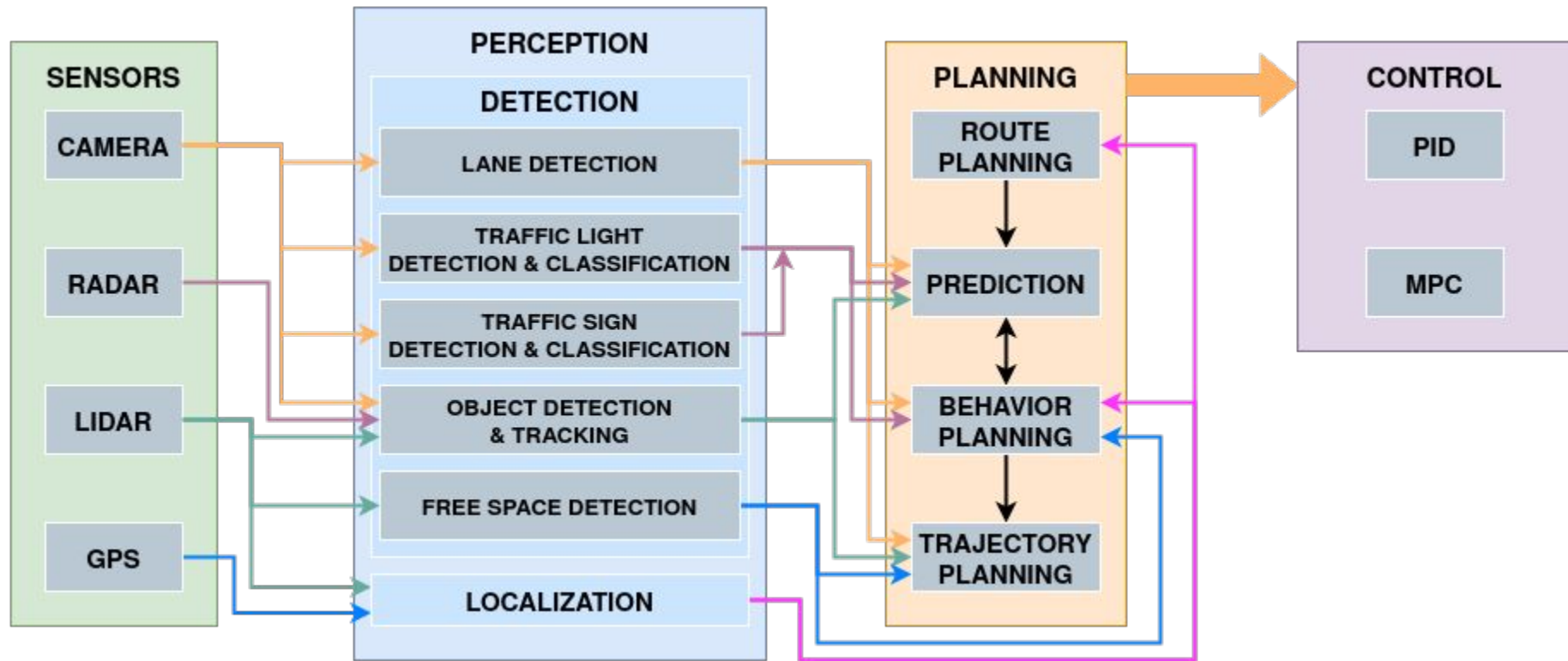
- More than 35k people die in auto accidents in the US each year*
- We need autonomous driving
- Apply computer vision, deep learning in practice
- Integrate related components to build a self driving car
- Test using Udacity Simulator

**National Highway Traffic Safety Administration, 2018*

Self driving cars

- A car that can guide itself without human conduction
- SAE defined levels of autonomy, level 0 to level 5
- Use various sensors, radars, lidars, cameras, machine learning for operation
- Leading companies Google, Tesla
- Two major components
 - ◆ Perception system
 - car localization, traffic light detection, obstacles detection, vehicle detection, road mapping, tracking, recognition
 - ◆ Decision making system
 - route planning, behavior selection, motion planning, obstacle avoidance and control

General Architecture



Project Description

- Implement the learned concepts to build a self driving car
- Sub systems built
 1. Lane Detection
 2. Traffic Light Classifier
 3. Model predictive Control
 4. Visual Odometry
 5. ROS pipeline
- Test on Udacity simulator

Sub System 1- Lane Detection

- a. Basic lane detection
- b. Advanced lane detection
- c. SCNN lane detection, Deep learning based model



a) Basic Lane detection

b) Advance Lane detection

c) SCNN Lane detection

Sub System 2- Traffic Light Classification

- Trained Faster RCNN on Bosch Small Traffic lights dataset
 - ◆ Merged 15 class labels to just 4 labels, Green, Red, Yellow and Off



Sub System 3- MPC

ANGLE -0.43° Mode: Autonomous

46.71 MPH

```

adashnpr@hcmi-r2d2 ~/Documents/BYU/semesters/Winter_2019/ELEN773/MPC/CarN...
File Edit View Search Terminal Help
882991, "psi": 3.97099, "x": -112.2261, "y": 45.62823, "steering_angle": -0.00966997, "th
rottle": 0.5769812, "speed": 45.42565]}
42["telemetry", {"ptxs": [-107.7717, -123.3917, -134.97, -145.1165, -158.3417, -164.316
4], "ptsy": [50.57938, 33.37102, 18.404, 4.339378, -17.42898, -30.18062], "psi_unity": 3.
875824, "psi": 3.970158, "x": -113.8697, "y": 43.82759, "steering_angle": -0.006982893, "t
hrottle": 0.5741427, "speed": 45.69762]}
42["telemetry", {"ptxs": [-107.7717, -123.3917, -134.97, -145.1165, -158.3417, -164.316
4], "ptsy": [50.57938, 33.37102, 18.404, 4.339378, -17.42898, -30.18062], "psi_unity": 3.
869134, "psi": 3.984848, "x": -115.511, "y": 41.99546, "steering_angle": -0.008036495, "t
hrottle": 0.5688163, "speed": 45.9626]}
42["telemetry", {"ptxs": [-107.7717, -123.3917, -134.97, -145.1165, -158.3417, -164.316
4], "ptsy": [50.57938, 33.37102, 18.404, 4.339378, -17.42898, -30.18062], "psi_unity": 3.
863564, "psi": 3.990417, "x": -116.8765, "y": 40.45242, "steering_angle": -0.008036495, "t
hrottle": 0.5688163, "speed": 46.17678]}
42["telemetry", {"ptxs": [-107.7717, -123.3917, -134.97, -145.1165, -158.3417, -164.316
4], "ptsy": [50.57938, 33.37102, 18.404, 4.339378, -17.42898, -30.18062], "psi_unity": 3.
857015, "psi": 3.996966, "x": -118.5121, "y": 38.5816, "steering_angle": -0.006634559, "t
hrottle": 0.5661777, "speed": 46.42763]}
42["telemetry", {"ptxs": [-107.7717, -123.3917, -134.97, -145.1165, -158.3417, -164.316
4], "ptsy": [50.57938, 33.37102, 18.404, 4.339378, -17.42898, -30.18062], "psi_unity": 3.
851783, "psi": 4.002198, "x": -119.8728, "y": 37.00709, "steering_angle": -0.006324488, "t
hrottle": 0.5639928, "speed": 46.63201]}
    
```

$$x_{t+1} = x_t + v_t * \cos(\psi_t) * dt$$

$$y_{t+1} = y_t + v_t * \sin(\psi_t) * dt$$

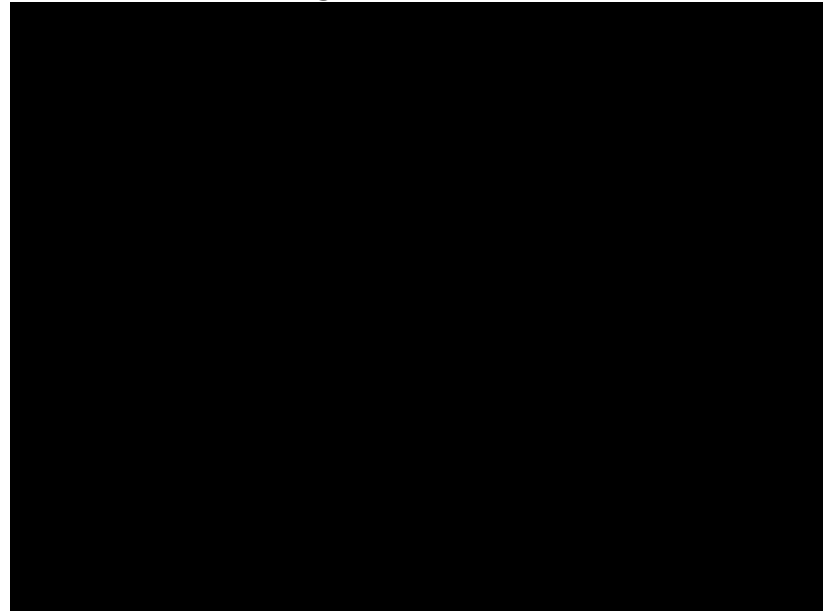
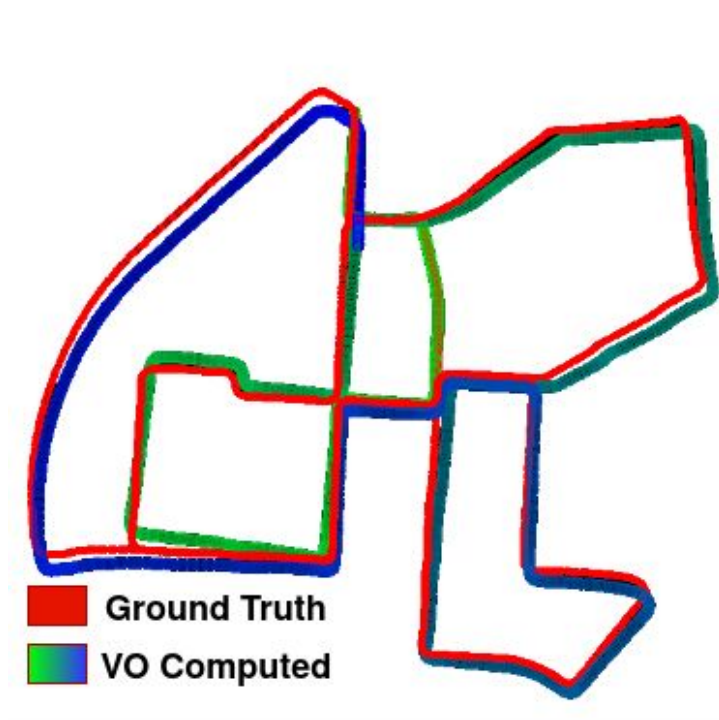
$$\psi_{t+1} = \psi_t + \frac{v_t}{L_f} * \delta_t * dt$$

$$v_{t+1} = v_t + a_t * dt$$

$$cte_{t+1} = f(x_t) - y_t + (v_t * \sin(e\psi_t) * dt)$$

$$e\psi_{t+1} = \psi_t - \psi_{des_t} + \frac{v_t}{L_f} * \delta_t * dt$$

Sub System 3- Visual Odometry



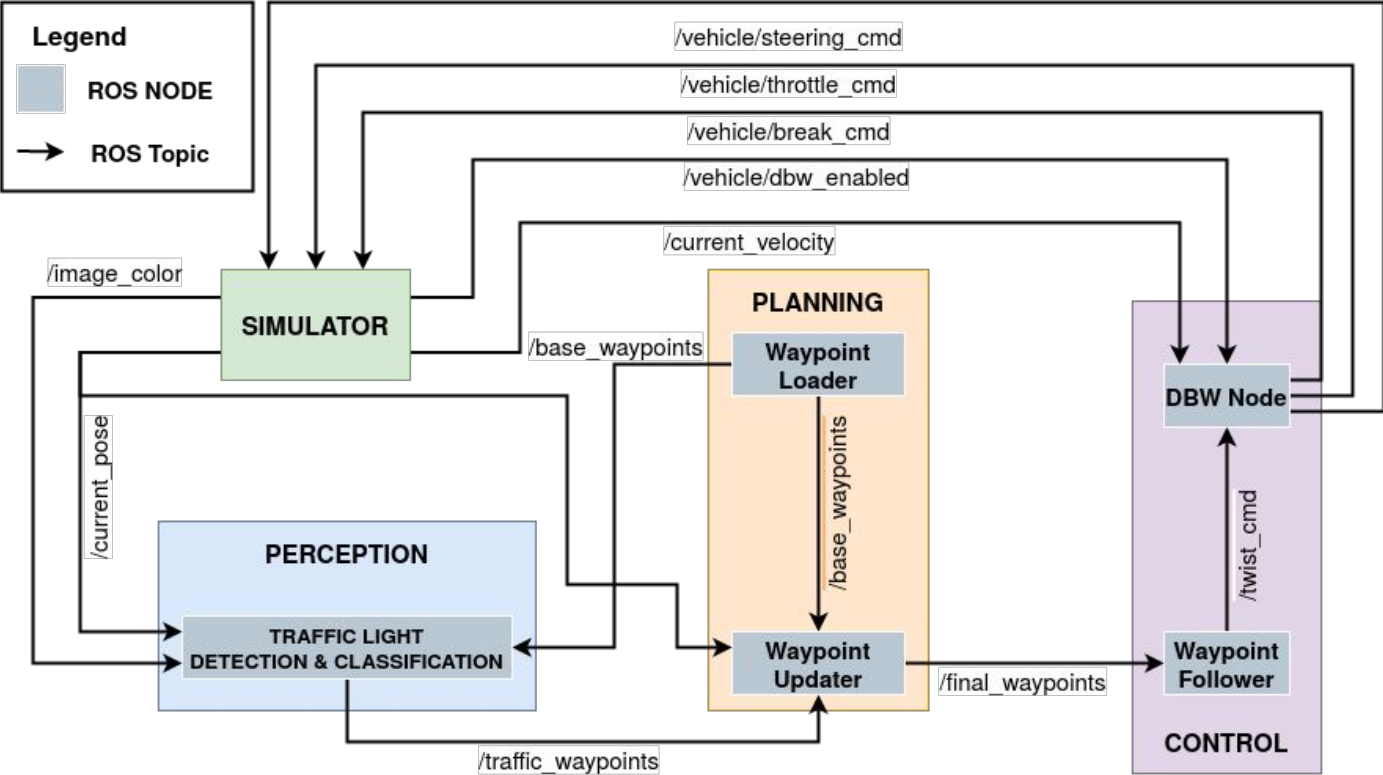
VO Pipeline

- Detect features using `cv2.FastFeatureDetector`
- Track features using `cv2.calcOpticalFlowPyrLK`
- Find essential matrix using `cv2.findEssentialMat`
- Recover pose using `cv2.recoverPose`
- Update the current pose based on previous pose

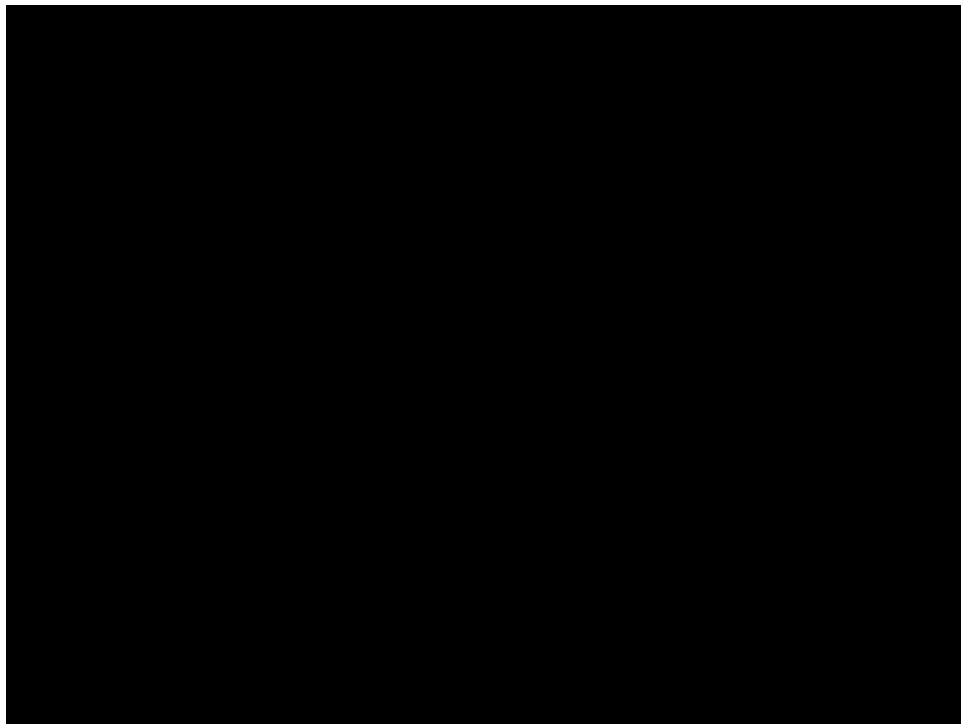
Udacity Simulator



ROS Pipeline



Demo Video



Challenges

- Project setup with ROS Noetic and Python3
- Lagging of Traffic light detection module

Conclusion and Future work

Conclusion

- Explored and implemented core components of self-driving car
- A minimal working autonomous vehicle in simulation

Future Work

- Semantic segmentation to detect static and dynamic objects
- Remove lagging in the traffic detection module
- Use vSLAM instead of simulated provided waypoints for planning

Questions

Thank You!