SDC Fun – IndiConcept Team

Team Name	:	**IndiConcept**
Team Members	:	(a) Rajeev Kumar Sharma (rajeevsharma.phd@gmail.com , Delhi INDIA) (b) Malik Khan (mayk9999@gmail.com , Adelaide Australia)
Team Formation & working Approach	:	We (Rajeev & Malik) initiated our work independently on complete project scope and couldn't able to form team in early stage of project. Student hub and group formation interface of Udacity gave opportunity to connect and slight lately initiated to work collaboratively as a duo team. To overcome project show stoppers we plan to work in tandom and identify stable software component from individual work and kept them in final merged project repository. During debugging and testing took an agile approach to bridge functional and performance gaps. This project has three parts (1) Traffic Light detection (2) Control System(3) Waypoint Following who are supposed to be system integrated through underneath ROS framework. Setup software and system dependencies was most challenging part of project. So we took an approach of having multiple setup (1) Udacity AWS workspace (2) Local Ubuntu (18.04) GPU enabled tensorflow (2) Independent VM.
Repository Link (Capstone Project)	:	https://github.com/RajeevSharma2015/Capstone-SDC
Repository Link (Traffic Light Classifier)	:	https://github.com/RajeevSharma2015/SDC-TrafficLight-Classifier
Project Video Demo	:	https://drive.google.com/drive/my-drive?ogsrc=32

Traffic Light Classifier :

4 Different models we trained for traffic light's classification

1. SSD Udacity 20K Steps : https://drive.google.com/file/d/14O3IpuCKm1ZRdgeqk8RX3lSxCsCgsAN8/view?usp=sharing
2. SSD Simulator 20K Steps: https://drive.google.com/file/d/13ZexYyCWuM4XeIBsaa7bPN-G20a3wayv/view?usp=sharing
3. FRCN 20k Steps: https://drive.google.com/file/d/1hvMwgB4Im8UyybpdDchzNqPRru_4b3SJ/view?usp=sharing
4. FRCN 10K Steps: https://drive.google.com/file/d/1PLp_jA5yilzwK5il5tq-va7wo09HMWIT/view?usp=sharing

A) **Self Driving Car – System Integration Project Objectives**

The goals / steps of this project are the following:

a) Setup Creation – As per specified OS, SW requirement.txt & ROS pre-requisites [Refere – Project readme.md for details]

b) ***ROS Based System Integration*** of SW modules;
   1. Path Planning
   2. Traffic Light Classification
   3. Obstacle Identification
   4. Vehicle Motion control

c) SW arhitecture planned to implement - Image1

d) A simulator for DBW provided by Udacity & vehicle motion verification supposed to be a 2 step activities process:
   1. Through DBW Simulator
   2. Loading SW on ***Carla car***

e) A SW working framework and partially implemented code (learned during – nano degree program) share by Udacity Team

Image References:

Image-1 : Targeted System Architecture – An system overview
Image-2 : Tensorflow-Gpu enabled machine setup
Image-3 : Tensorflow-Gpu verification
Image-4 : SSD V2 inception model training
Image-5 : faster RCN V2 inception model training

➢ System Architecture – Overview in ros model, Node, Topic & Messages

ROS system integration SDC node architecture framework is provided by Udacity. Basic SW component, Simulator and module interfaces were defined for ease of understing and implementing ros system integration approach for SDC fun project.

Student are supposed to build up a vehicle system – perception, planning and control module. ROS undestanding and interface mechanism was a knowledge pre-requisite to complete this project. So every student supposed to go through complete ROS tutorials (practically). Simulator message specification and interface detail (node, topic/message) become a building block to communicate within system.

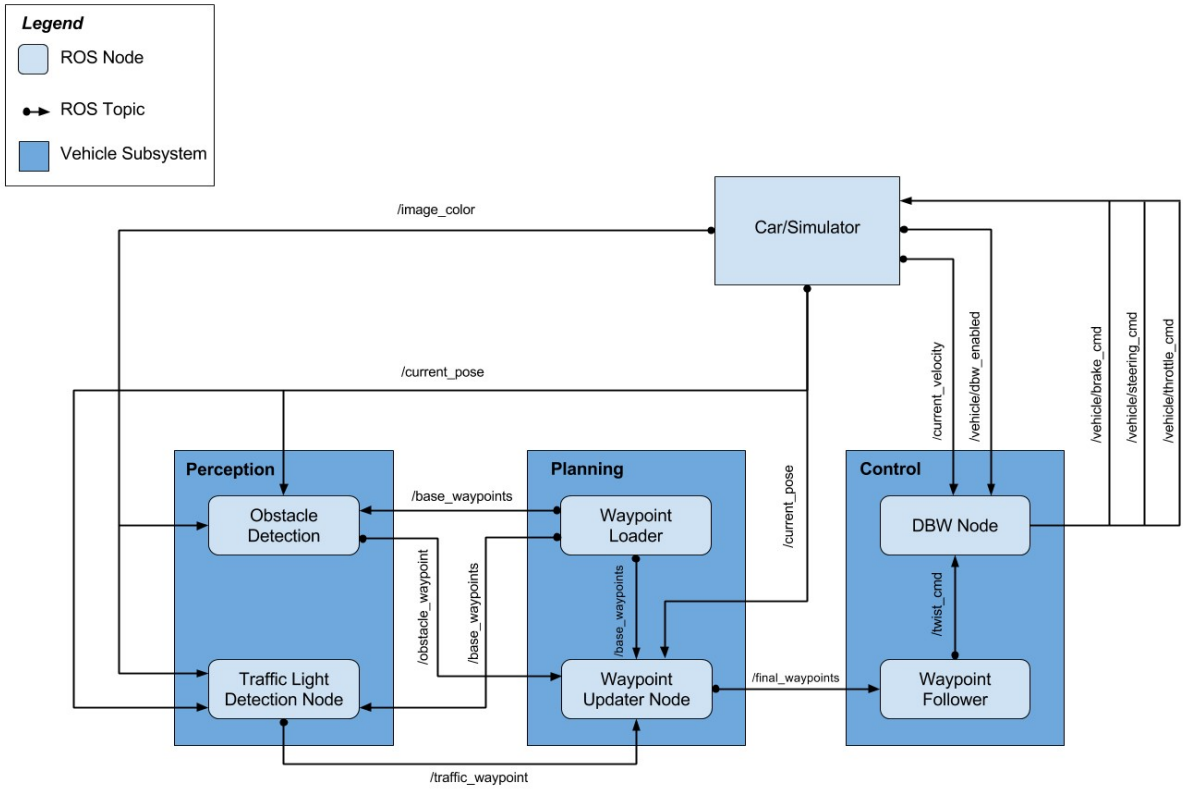

Image-1 Targeted system architecture

- **IndiConcept Repository (Files submitted & code - overview)**

  Submission includes all required files and can be used to run the simulator in autonomous mode, my project includes the following files:

  1. Project Structure :

     IndiConcept repository structure has 4 folders - ros, data, data_science & IMG_DOC
     - ros : ros packages maintains project source (src), compilation, linking, build and run time environment along with executabled. This folder has sub folders - **build**, **devel, launch** and **src**
     - data : simulator *.csv and other data files for different modes of testing (sim & site)
     - data_science : learned models (*.h5) for "traffic light classification" and learn/test environment
     - IMG_DOC : This folder maintains reference images and technical reference documents
  2. Video's Link : Project execution recorded in a video and reference given.
  3. IndiConceptTeam-writeup.pdf describe project, summarizing result, objectives achieved and challenges faced.

- **Submission includes functional code & other files**

  IndiConcept team project contains a fully functional code in simulator environment but latency issues are observed frequently on Udacity workspace. Simulations video's recorded for different scenarios and samples images also maintained for reference in repository. The final testing of code can be analyzed and concluded in Udacity simulator/Carla test environment.

- **Submission code is usable and readable**

  The IndiConcept team contains the code for functionalities mentioned in goal/objectives. The file containes definition of various function, their execution outcome, visualisation for specific objectives.

  - This report describe each function how algorithm work and brief of code syntax.
  - Code has modularity, independent functions for specific purposes
  - No duplicacy of code syntax and mostly system defined parameters (to avoid hardcodings).

(B) Solution Development

Motion control depends on knowledge of the waypoints. In this case, the waypoints are sufficiently close together to not require interpolation. We first changed waypoint_loader.py to publish a single, latched read of the waypoints, as it had initially been set to publish remarkably frequently and wasted a lot of valuable resources.

The waypoint_updater.py perform 2 critical tasks:

1. It takes current pose, runtime property, static waypoints and produces Lane topic that consist of subset of waypoints starting with first waypoint in front of car.
2. Traffic light control is performed

The logic implementation is done through KDTree and waypoint indexes selection, nearest to car and in front position. Next depends on control mechanisms from traffic light. Preparing a velocity profiling is desirable. This profile calulate velocity for forward waypoints based on accelaration and decelaration. Objective is to provide smooth, controlled acceleration and smooth breaking. The calculated list integrate with Lane topic and publish.

Waypoint_follower.py implement pure pursuit algorithm, which return TwistStamped topic that compute best move of car based on algorithm. dbw_node.py subscribes to the TwistStamped topic produced by the waypoint follower. Here we seek to control throttle, brake and steering of car using desired linear, twist velocities and current velocity. Main loop in **dbw_node.py** create simple data structure passed to an instance of controller class to get desired control.

PID controller manage output of throttle and brake. It conditionally allow scaling of throttle in range [0,1] while allow scale to maximum torque as defined by car property for braking. PID controller tunned manually, and scaling function for both acceleration and breaking is a variant of soft step – reason smooth transition, bounding due to asymptotic nature and ability to change degree of control. Control is also responsive for dbw_enable topic and reset of integral accumalator of PID controller when this toggle.

Traffic light detection required several steps to accomplish, first step was to collect the training data. Full images were classified after loading ROSBAG and output as h5 files. The traffic lights were extracted from collection of frames by color slection and clustering. The available data was split into – training, test and validation set. A pre-trained VGG16, Faster RCN and SDD were tried. Post initial analysis we finally used a SSD V2 model in our project.

Traffic light classification source code & data managed in a seperate repository. Link for repository provides details of project execution. This section has 2 core activities (1) Training Data Preparation (2) Tensorflow Setup creation (3) Training of a suitable model. We have used old labeled data for training.

Tensorflow-gpu setup:

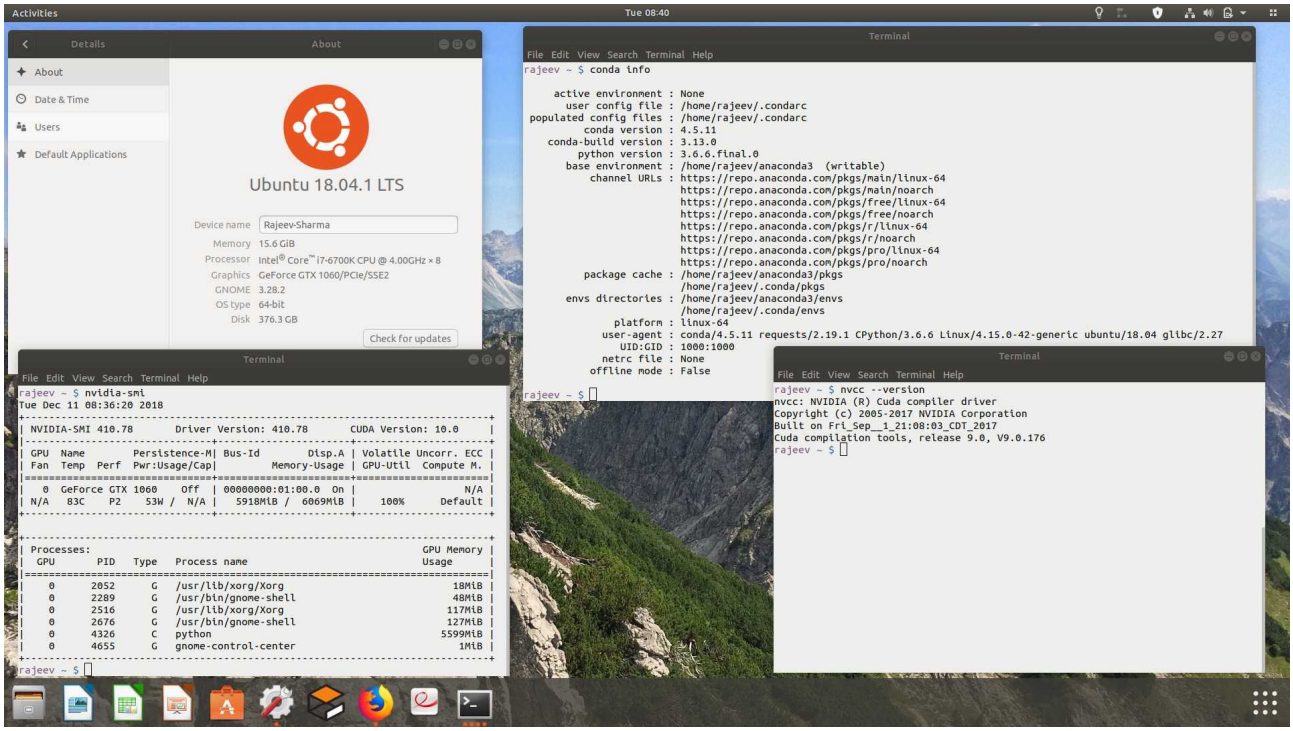

Image-2 Tensorflow-Gpu enabled machine setup

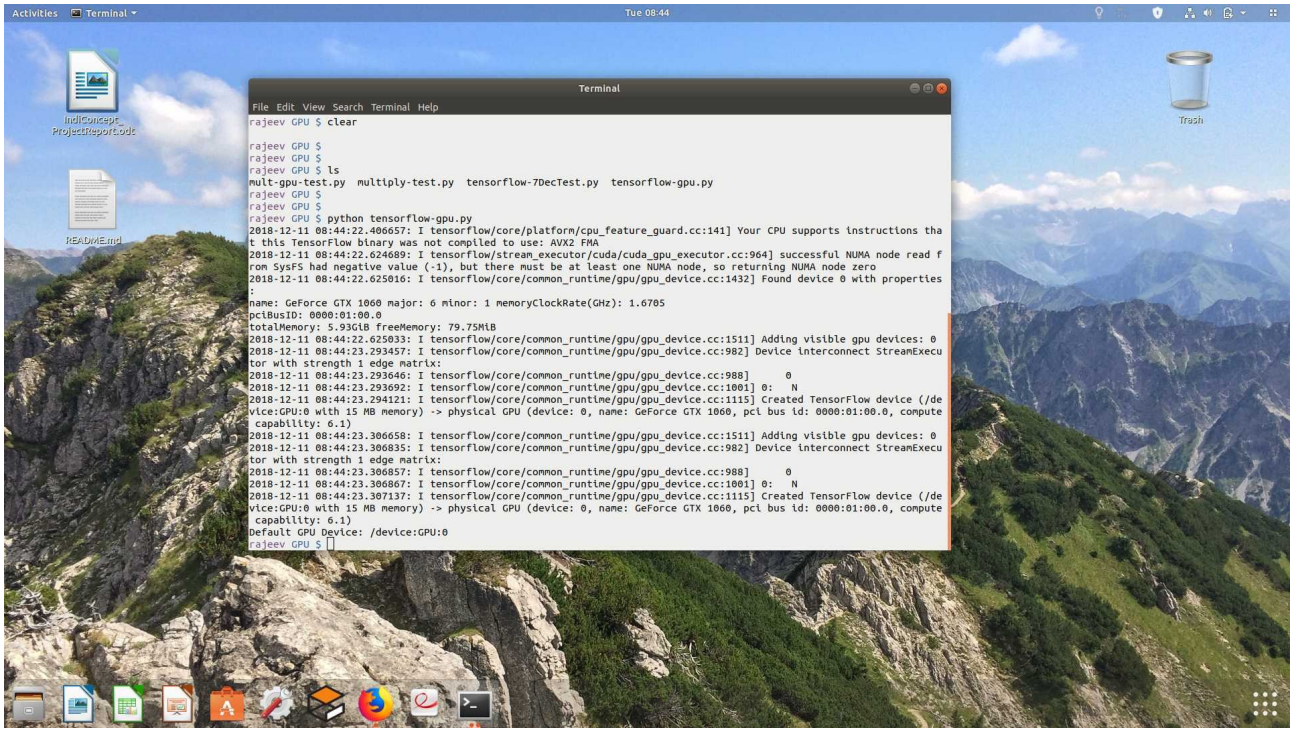

Image-3 Tensorflow-Gpu verification

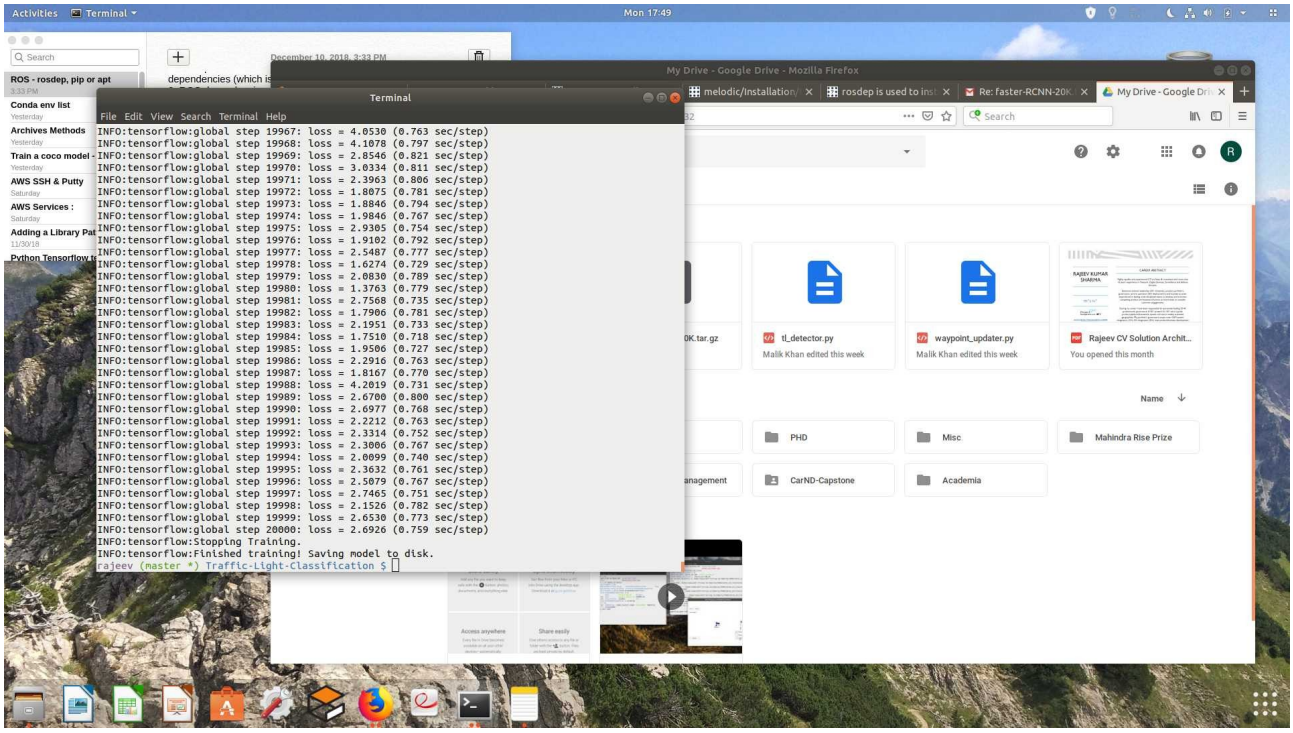

Image-4 : SSD V2 inception model training

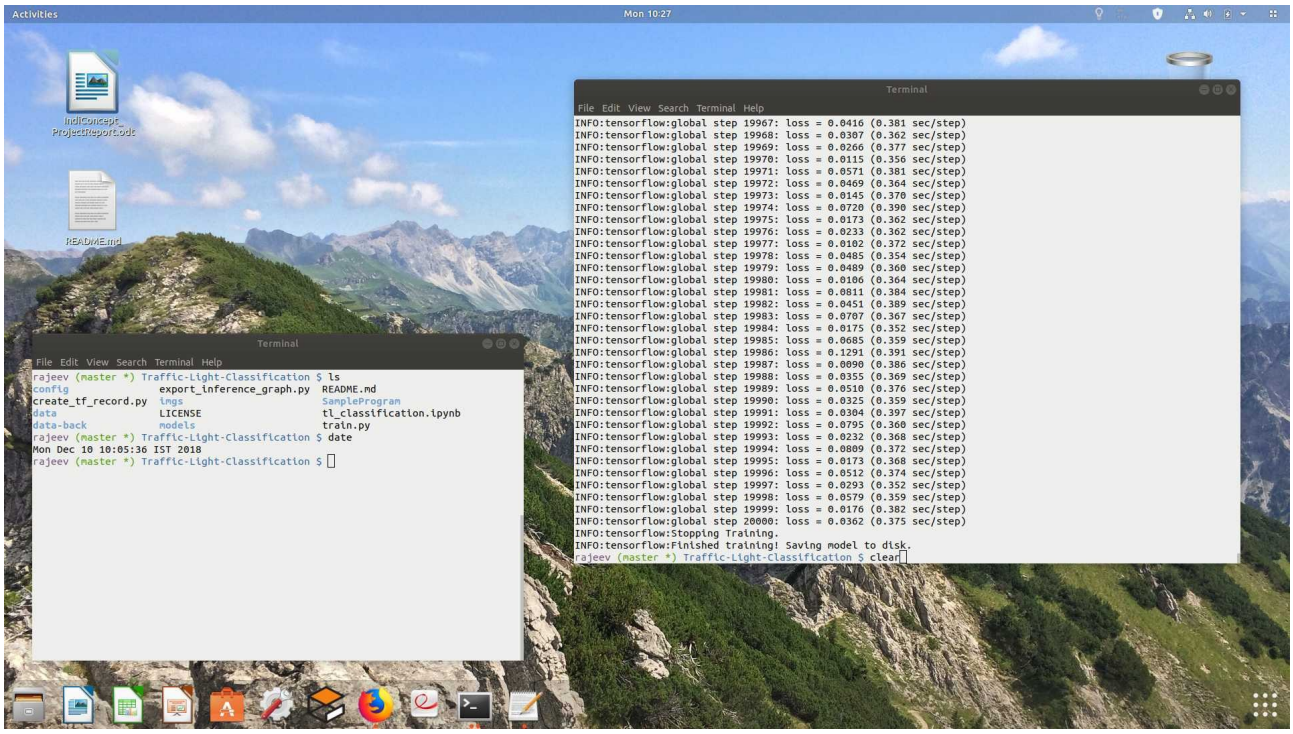

Image-5 : faster RCN V2 inception model training

(C) Simulation & Test Results:

Initial system development & Test observation:

a) Project simulation test (video recording) - while CAMERA (Untick) & Manual (Untick).

- Here SDC running in autonomous mode, camera OFF and TL detection is disabled.
  Youtube link : https://youtu.be/ZHrT34p2jiw.

- ***Conculsion - Partial Feature Testing***
  Base_waypoint updator & follower is able to publish point on lane, and controller able to drive SDC smoothly on track. This simulation is tried on Udacity workspace. In this mode no latency observed.

- Project simulation test (video recording) - while Manual (Untick), CAMERA (Untick) & TL classifier loaded
  - Here SDC running in autonomous mode, Camera ON & Traffic light detection active
  - Setup - Udacity GPU workspace
  - VGG16 - TL trained classifier used
    Youtube link : https://youtu.be/2t5HuuquY-8

  - ***Conclusion - Fully Loaded Feature Testing***
    - SDC start normally and initiate following waypoints
    - RED light detection happens, SDC usages braking to stop car
    - Traffic light change state GREEN, SDC able to detect and manage throttle to drive vehicle
    - Meanwhile setup latency seems plays significant role and final_waypoint update and following seems get disturbs
    - Submission and a final trial on Carla seems to be a way forward to test final result.

Final solution & Test observation:

In final project we itroduced KDTree & used “SSD v2 inception model” for traffic light classification.

- Simulation test (video recording) - while CAMERA (Untick) & Manual (Untick).
- Setup – VM image
- Trained classifier – 20k steps SSD-V2-Inception, model download link;
  https://drive.google.com/file/d/13ZexYyCWuM4XeIBsaa7bPN-G20a3wayv/view?usp=sharing
- Demo video link - https://drive.google.com/drive/my-drive?ogsrc=32
- ***Conclusion*** – traffic lights identification, vehicle control and waypoint trajectory follow up happen well. Latency concern is also mitigated.

Summary:

SDC FUN - IndiConcept team, ROS system integration project repository contains working code, functions definition and outcome visualisation. All sections give desired result. Code has modularity, no hardcoding and readability. Code successfully compiled, build, development setup and roslaunch executed successfully. Simulator establishes connection with developed vehicle sub systems and system executes desired functionalities in synchronization end to end as a result vehicle smoothly run on test track and stops at traffic lights.

Waypoints update, path following on waypoint, lane identification and vehicle system control (steering, throttle and brake) happens smoothly. While camera ON, traffic light detection identify lights status and manage control system to stop and run vehicle.

References :

1. *Optimum setting of automated controller reference paper by J.G. ZIEGLER and N. B. NICHOLS • ROCHESTER, N. Y , https://github.com/RajeevSharma2015/Capstone-SDC/blob/master/IMG_DOC/Technical_Papers/AutomotedController.pdf*
2. *Unified, Small, Low Power Fully Convolutional Neural Networks for Real-Time Object Detection for Autonomous Driving by Bichen Wu, Forrest Iandola, Peter H. Jin, Kurt Keutzer UC Berkeley, DeepScale https://github.com/RajeevSharma2015/Capstone-SDC/blob/master/IMG_DOC/Technical_Papers/DNN_Classifiers.pdf*
3. *Design of PI and PID Controllers With Transient Performance Specification by J. C. Basilio and S. R. Matos https://github.com/RajeevSharma2015/Capstone-SDC/blob/master/IMG_DOC/Technical_Papers/Path_Controller.pdf*

4. *Implementation of the Pure Pursuit Path 'hcking Algorithm by R. Craig Conlter*
https://github.com/RajeevSharma2015/Capstone-SDC/blob/master/IMG_DOC/Technical_Papers/PathTracking.pdf
5. *Traffic Light Mapping and Detection by Nathaniel Fairfield Chris Urmson*
https://github.com/RajeevSharma2015/Capstone-SDC/blob/master/IMG_DOC/Technical_Papers/TrafficLight_Detection.pdf