UNIVERSITÀ DEGLI STUDI DI SALERNO

Dipartimento di Ingegneria dell'Informazione ed Elettrica e Matematica applicata

Autonomous Vehicle Driving 2021/2022: Final Project

Gruppo 19:

Antonio Iannaccone matr. 0622701557

Carmine Fruncillo matr. 0622701615

Nicolò Grieco matr. 0622701578

Leonardo Galiano matr. 0622701608

ANNO ACCADEMICO 2021/2022

Executive Summary

Il progetto finale assegnatoci ha come obiettivo estendere e completare il mid term project. Partendo da quest'ultimo bisogna utilizzare nel progetto finale tutti i seguenti sensori: almeno una telecamera, almeno un sensore di profondità (depth camera o lidar), semantic segmentation. Bisogna sostituire uno dei sensori perfetti con uno reale e gestire almeno le seguenti situazioni:

- Non effettuare collisioni con altri veicoli o pedoni (escludendo pedoni che colpiscono il veicolo che stava già passando e veicoli che non dovessero rispettare il codice della strada)
- Stop e ripartenza al semaforo
- Ignorare il pedone fermo sul marciapiede, evitando inutili decelerazioni o frenate brusche
- Fermarsi se il pedone si accinge ad attraversare, valutando la sua traiettoria.

Assunzioni fatte:

- La simulazione è stata effettuata di giorno, in condizioni meteorologiche non avverse, velocità di default.
- Frame rate 30 fps
- Condizioni meteo variabili
- Presenza di ostacoli in movimento come veicoli o pedoni
- Sorpassi non permessi
- Grafica: Low Quality

Mid term upgrade

I dati dell'analisi quantitativa della prova mid term evidenziavano come maggiori problemi quelli relativi ad alcune fermate in prossimità dei semafori. In particolare, quando la simulazione iniziava con il veicolo posizionato troppo vicino al semaforo quest'ultimo non veniva rilevato in tempo e quindi il veicolo lo ignorava. Per ovviare a ciò è stato implementato un detector specifico per i semafori. Quest'ultimo è in grado di tracciare le bounding boxes dei semafori presenti in un frame. Abbiamo deciso di utilizzare due camere, una frontale e una specifica per individuare meglio i semafori in modo da agevolare il detector. Una volta localizzato un semaforo il detector fa una predizione dello stato in cui si trova (due stati possibili: *STOP* & *GO*).

I dati utilizzati per effettuare la predizione sono quelli di entrambe le camere: in primo luogo la camera specifica per i semafori e poi la camera frontale. Se l'accuracy della predizione effettuata sulla prima camera è maggiore di un certo treshold (0,35) l'output di quest'ultimo viene considerato affidabile. In caso contrario, la stessa procedura viene effettuata sulla camera frontale ma con un treshold più basso (0,20). Nel caso in cui nessuna delle due predizioni vada a buon fine, lo stato del semaforo viene settato a *NO_TL* che sta ad indicare l'assenza di semafori nel frame.

Nel caso in cui lo stato del semaforo sia in *STOP*, viene calcolata la sua distanza dal veicolo, indicata dalla variabile *dist_stop*, e viene confrontata con un valore di treshold (*DIST_TO_TL*) pari a 15m. Se *dist_stop* < *DIST_TO_TL* allora il semaforo viene preso in considerazione. Successivamente viene calcolato il waypoint più vicino ad esso al quale fermarsi. Il veicolo passerà dallo stato di FOLLOW_LANE allo stato di DECELERATE_TO_STOP e ci resterà fin quando lo stato del semaforo sarà *GO* o *NO_TL*.

Nel caso in cui lo stato del semaforo sia GO il veicolo proseguirà la sua corsa rimanendo nello stato FOLLOW_LANE.

Altro problema emerso dall'analisi quantitativa della prova mid term riguardava la gestione delle collisioni con i pedoni, nello specifico vi erano alcuni casi particolari in cui il pedone o attraversava improvvisamente o con una traiettoria particolare che non dava modo al veicolo di fermarsi in tempo. Per risolvere questo problema si è ricorso alla semantic segmentation perfetta di CARLA per migliorare il codice precedentemente sviluppato. La semantic segmentation è stata implementata attraverso una camera dedicata con cui siamo in grado di filtrare soltanto gli elementi a cui siamo interessati nello scenario (pedoni e strada).

In quest'ultimo andiamo a controllare se ogni pixel appartiene alla classe 4 (specifica del pedone nella semantic segmentation di CARLA) e se la sua distanza, ottenuta mediante una camera di profondità, è inferiore a 7m. In questo caso il punto viene passato alla funzione *pointOnRoad* che controlla se esso è adiacente a uno dei punti di classe 7 (specifica della strada nella semantic segmentation di CARLA). Questi punti vengono ottenuti analogamente a quelli del pedone. La funzione restituisce TRUE se il punto è adiacente, il che sta ad indicare che il pedone è su strada, o FALSE in caso contrario.

Nel caso in cui il pedone si trovi su strada viene settato il flag *pedestrian_found* a TRUE. Inoltre, viene effettuato un ulteriore controllo per verificare se la distanza tra il pedone e il veicolo è critica, ovvero minore o uguale a 3m. In tal caso, viene settata a TRUE una nuova flag *emergency_stop.* A questo punto se la traiettoria del pedone interseca quella del veicolo e contemporaneamente se il pedone è su strada (*pedestrian_found* = TRUE) viene settata la variabile *obstacle_on_lane* del behavioral planner a TRUE e viene gestito come già descritto nel summary della prova mid term.

Lo scopo della flag *emergency_stop* è quello di effettuare una frenata di emergenza quando il pedone si trova pericolosamente vicino al veicolo. Questo viene ottenuto attraverso una modifica effettuata nella funzione *compute_velocity_profile* del modulo velocity planner. Se il flag è TRUE viene settata la variabile *stop* a infinito

che viene passata alla funzione *decelerate_profile* dove viene effettuato il controllo sulla distanza di frenata. A questo controllo viene aggiunta la variabile *stop* che se equivale a infinito consente di passare il controllo decelerando maggiormente. Se invece il pedone non si trova nel raggio di 3 metri *stop* sarà di default a 0 (la macchina decelera normalmente).

La maschera di pixel, riportata in figura a sinistra, mostra i pixel della strada e quelli dei pedoni in un raggio di 7 metri. Questi sono ottenuti con la **semantic segmentation**.

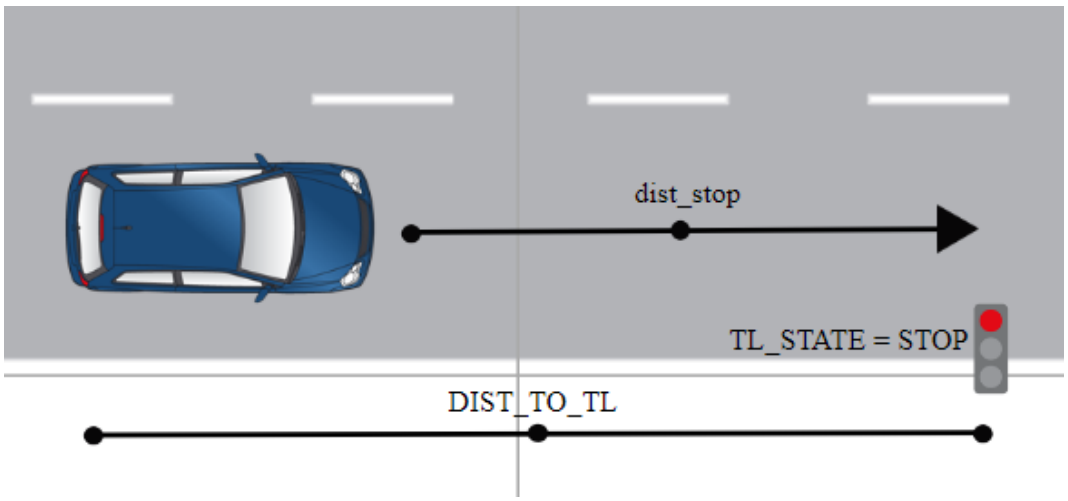


Figure 1 : FOLLOW_LANE -> DECELERATE_TO_STOP

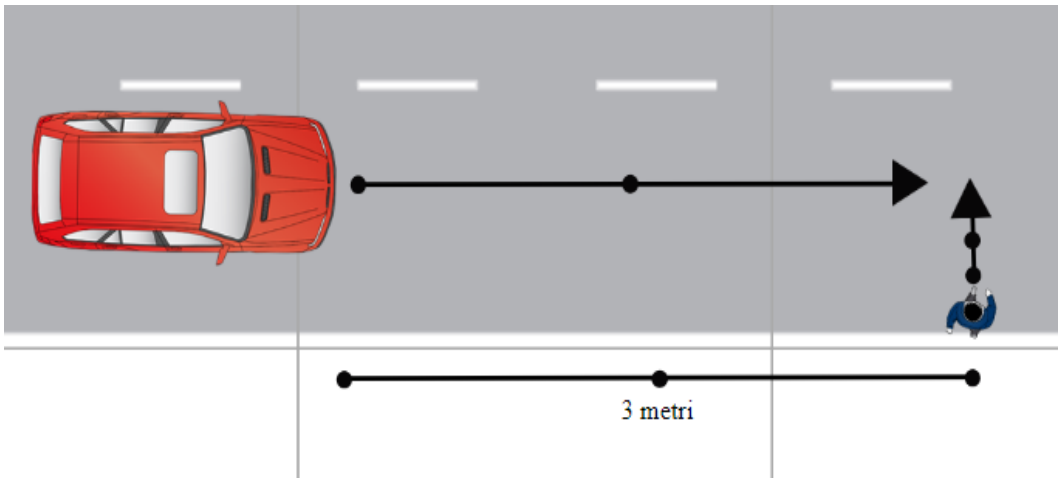


Figure 2 : EMERGENCY_STOP

Design and Implementation

L’automa a stati finite dopo le modifiche apportate alla mid term è quello sopra riportato e presenta i tre stati:

- FOLLOW LANE: il veicolo procede a velocità costante seguendo la corsia
- DECELERATE_TO_STOP: il veicolo inizia a rallentare per fermarsi a un semaforo
- STOP FOR OBSTACLES: il veicolo decelera ed eventualmente si ferma in presenza di un ostacolo (per ostacolo si intende pedone o veicolo)

Nel behavioral planner vengono gestiti i seguenti casi:

- Se il semaforo si trova nello stato FOLLOW_LANE: rimane nel seguente stato se non ci sono ostacoli nel campo visivo del veicolo e se il semaforo individuato nel campo visivo è verde. Passa nello stato DECELERATE_TO_STOP se nel campo visivo è presente un semaforo rosso. Passa nello stato STOP_FOR_OBASTACLES se è presente un ostacolo nel campo visivo del veicolo.
- Se il veicolo si trova nello stato DECELERATE_TO_STOP: rimane nel seguente stato se è presente un semaforo rosso nel campo visivo del veicolo. Passa nello stato FOLLOW_LANE se è presente un semaforo verde nel campo visivo del veicolo o se non viene rilevato nessun semaforo. Passa nello stato STOP_FOR_OBSTACLES se è presente un ostacolo nel campo visivo del veicolo.
- Se il veicolo si trova nello stato STOP_FOR_OBSTACLES: rimane nel seguente stato se l’ ostacolo continua ad essere nel campo visivo del veicolo. Passa nello stato FOLLOW_LANE se non ci sono ostacoli nel campo visivo del veicolo.

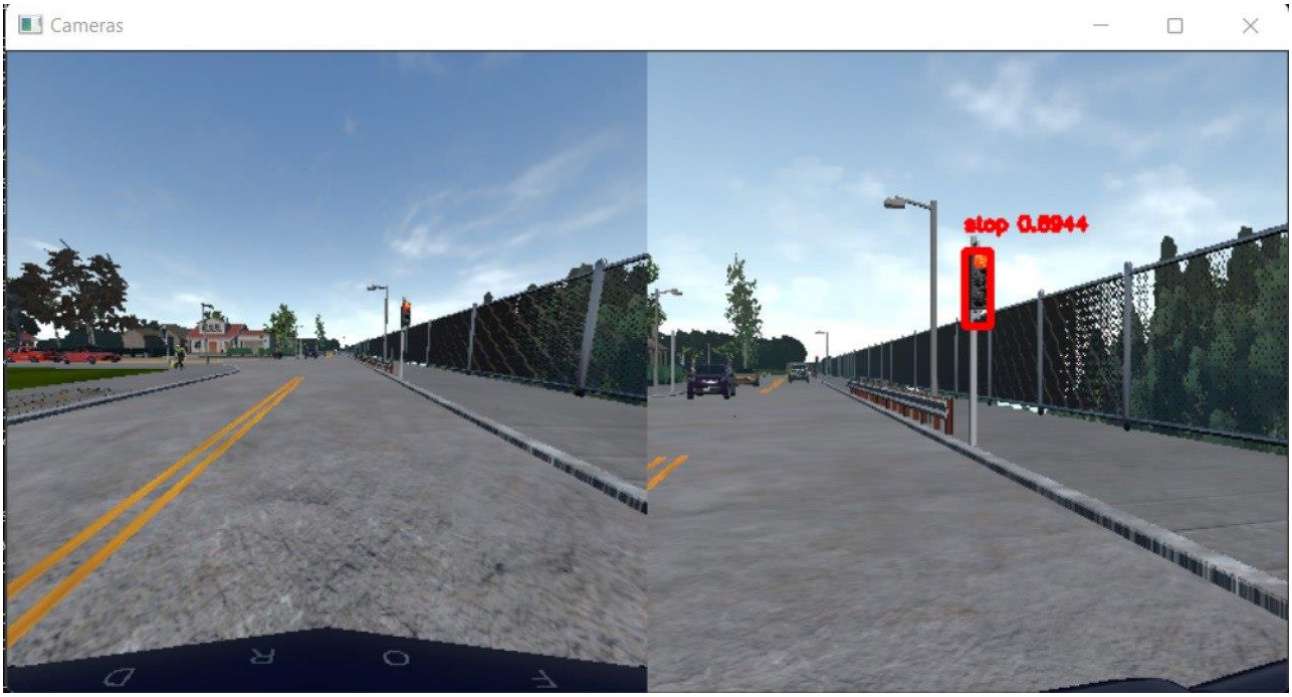
Camera Details

Sono presenti nel progetto finale le seguenti camere:

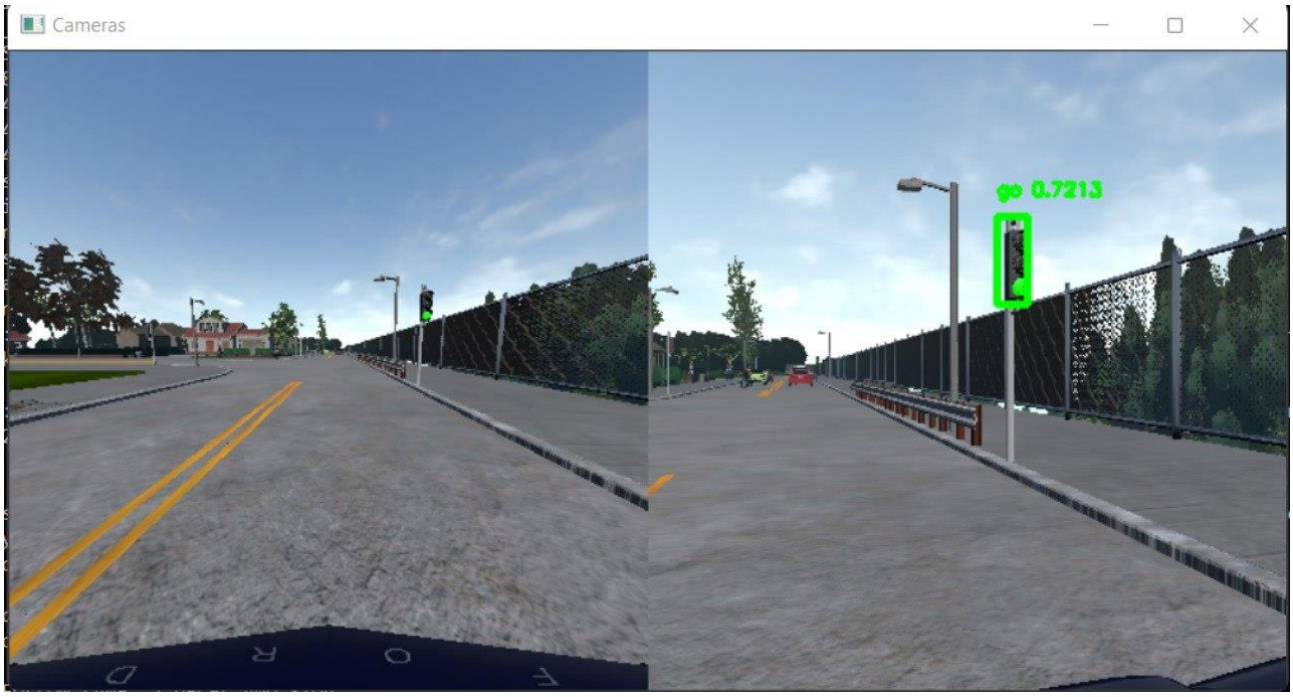
- Una camera frontale RGB con FoV di 120°, coordinate xyz [2.0, 0, 1.3], una size di 400x400 e un pitch di -6,5. (*CameraRGB_Front*)
- Una camera frontale RGB per i semafori con FoV di 60°, coordinate xyz [1.5, 0, 1.3], size di 400x400 e uno yaw di 20°. *(CameraRGB_TL*)
- Una camera frontale per la segmentazione semantica con FoV di 120°, coordinate xyz [2.0, 0, 1.3], una size di 200x200 (*Segmentation*)

Abbiamo inoltre 3 depth camera, associate ad ognuna delle camere RGB e di conseguenze con gli stessi parametri delle camere relative. Esse sono *CameraDEPTH_FRONT, CameraDEPTH_TL, SegmentationDEPTH*.

Sono riportate di seguito le immagini del veicolo fermo al semaforo dopo averlo rilevato. In basso, a sinistra *CameraRGB_Front,* in basso a destra *CameraRGB_TL.*



Inoltre, viene mostrata la ripartenza del veicolo al semaforo, nel momento in cui diventa verde. Le camere mostrate in basso sono le precedenti già descritte.



Traffic Light Detection with YOLOv2

Per effettuare la Traffic Light detection abbiamo scelto di utilizzare un modello per object detection basato su TinyYOLOv2. Dato che i dispositivi a nostra disposizione per effettuare i test avevano prestazioni limitate abbiamo scelto la versione Tiny in grado di girare anche su dispositivi limitati. Il nostro modello è stato preaddestrato sul dataset COCO e successivamente addestrato sui dataset LISA(5800 immagini) e CARLA(1800 immagini) specifici per la Traffic Light Detection.

Experimental Analysis of the System

Analisi quantitativa

	TEST 1	TEST 2	TEST 3	TEST 4	TEST 5	TEST 6
SCENARIO	7 – 15	45 - 53	55-70	13-15	51-20	20-40
VEHICLE	250	200	100	250	200	333
PEDESTRIAN	250	450	750	250	200	330
SEED PEDESTRIAN	0	0	0	0	0	0
SEED VEHICLES	5	5	5	5	5	5

I percorsi sono stati scelti per verificare le collisioni con veicoli e pedoni.

I percorsi dei test sono i seguenti:

- Test 1: il veicolo percorre un rettilineo, si approccia a un semaforo, poi svolta a sinistra e percorre un altro rettilineo.
- Test 2: il veicolo percorre un rettilineo e deve gestire le collisioni con i pedoni e l'approccio a un semaforo
- Test 3: il veicolo si approccia immediatamente ad un semaforo, successivamente gira a sinistra, si approccia ad un altro semaforo e poi gira a sinistra.
- Test 4: il veicolo si approccia ad un semaforo, poi svolta a sinistra e percorre un rettilineo.
- Test 5: il veicolo percorre un rettilineo, successivamente si approccia ad un semaforo e gira a destra e poi dopo un breve rettilineo gira di nuovo a destra
- Test 6: il veicolo percorre un rettilineo approcciandosi a due semafori.

N.B: Le simulazioni vengono svolte ogni 2 frame per semplificarle a livello computazionale.

TEST 1:

VEHICLE COLLISION	0
PEDESTRIAN COLLISION	1
DESCRIPTION	Nello scenario del Test 1 il veicolo segue le norme della strada e il veicolo davanti eseguendo il percorso correttamente. Da segnalare un pedone che colpisce il veicolo dopo che quest'ultimo l'aveva già superato.
NOTE	Simulazione completata correttamente

TEST 2:

VEHICLE COLLISION	0
PEDESTRIAN COLLISION	0
DESCRIPTION	Nello scenario del Test 2 il veicolo si ferma davanti a un pedone che attraversa in maniera defilata decelerando di colpo, ripartendo, si ferma poi a un semaforo rosso. Mentre è fermo al rosso passano tre pedoni e riconoscendoli rimane fermo. Riparte poi col verde
NOTE	Simulazione non completata

TEST 3:

VEHICLE COLLISION	0
PEDESTRIAN COLLISION	0
DESCRIPTION	Nello scenario del Test 3 il veicolo procede rispettando le norme della strada ma si ferma dopo il secondo semaforo al centro della carreggiata senza un apparente motivo.
NOTE	Simulazione non completata

TEST 4:

VEHICLE COLLISION	0
PEDESTRIAN COLLISION	1
DESCRIPTION	Nello scenario del Test 4 il veicolo esegue correttamente il percorso. Un solo pedone colpisce la macchina lateralmente. Successivamente si ferma al semaforo, poi riparte e si ferma dietro al veicolo che lo precede anche se a distanza più elevata
NOTE	Simulazione completata correttamente

TEST 5:

VEHICLE COLLISION	0
PEDESTRIAN COLLISION	0
DESCRIPTION	Nello scenario del Test 5 il veicolo parte su un rettilineo e incontra un semaforo, passa con il giallo per poi svoltare a destra. Nonostante effettui una curva molto larga invadendo l'altra corsia, non collide con gli altri veicoli presenti sulla carreggiata in quanto riconoscendoli come ostacoli si arresta. Proseguendo la corsa si ferma in prossimità di un semaforo rosso per poi ripartire al verde e svoltare a destra dove al centro della strada incontra una cassetta della posta colpita da un altro veicolo. Questa viene ignorata ed il veicolo si ferma in coda ad un altro veicolo fermo ad un semaforo
NOTE	Simulazione non completata

TEST 6:

VEHICLE COLLISION	0
PEDESTRIAN COLLISION	0
DESCRIPTION	Nello scenario del Test 6 il veicolo si ferma in coda ad un altro veicolo fermo al semaforo, per poi ripartire al verde. In prossimità del secondo semaforo il veicolo si comporta analogamente a prima. Da segnalare che nel secondo caso il nostro veicolo si arresta a distanza molto elevata da quello che lo precede
NOTE	Simulazione non completata

Analisi qualititativa

Per l'analisi qualitativa dei risultati vengono di seguito riportati alcuni casi critici e migliorie:

- Problema riscontrato di cui non si riesce a capire la causa è che in alcuni rettilinei anche in assenza di pedoni e veicoli, l'auto rallenta per poi fermarsi.
- Per brevi istanti di tempo il detector rileva semafori anche dove non ci sono, ad esempio in alcuni scenari il detector confonde il sole, avendo esso forma circolare, con la luce verde del semaforo. Tuttavia, questo problema è

riscontrato solo nella camera frontale in quanto nella camera dedicata al riconoscimento del semaforo il sole non viene inquadrato.

- In alcuni scenari, per brevi istanti di tempo, anche alcune teste dei pedoni vengono riconosciute come luci del semaforo. Questo comporta che il veicolo rallenti per questi brevi lassi di tempo. Avvicinandosi ai pedoni questo problema viene risolto e quindi il veicolo accelera riprendendo la sua corsa.
- Spesso, il veicolo, dovendosi fermare per non collidere con la vettura che lo precede si arresta ad una distanza considerevole da questa.
- L'utilizzo della semantic segmentation, unito al calcolo della traiettoria dei pedoni, permette di ridurre attese inutili (ad esempio quando un pedone sta per attraversare ma è ancora sul marciapiede)
- L'utilizzo della semantic segmentation ci ha consentito di implementare anche una frenata di emergenza (*emergency_stop*) qualora un pedone si trovasse in una posizione molto ravvicinata al veicolo e quest'ultimo non avesse il tempo di fermarsi. Questo ci ha consentito di gestire anche alcuni casi in cui i pedoni attraversano in maniera sconsiderata.
- Mediante l'implementazione del detector per le traffic light, alcuni semafori che prima venivano ignorati perché troppo vicini al veicolo adesso vengono rilevati.

Link Video Youtube:

Di seguito vengono riportati due link che mostrano il funzionamento della semantic segmentation e del detector per i semafori:

- Semantic Segmentation:
  https://youtu.be/rlFUPuGgf6g

- Traffic Light Detection:
  https://youtu.be/PCMG3YjX9wM