AutoCar

**Umetna Inteligenca – Projektna naloga**

**Tim Vehovar | Uroš Kos | Luka Lašič**

1. Začetek projekta

Sprva smo si skupaj namestili ter vzpostavili delovanje simulatorja Carla – verzija 0.9.13.

Nato pa smo razdelili naloge sledeče:

- Pomoč pri ohranjanju avta v pasu ( lane assist ) – Uroš Kos

- Prometni znaki ( hitrost ) – Tim Vehovar

- Razpoznava pešcev – Luka Lašič

Za razpoznavo smo uporabili realno-časovno razpoznavanje objektov YOLO – verizija 3 ter knjižnico TensorFlow za umetno inteligenco oziroma učenje naprave.

Nato smo inicializirali obe vključeni knjižnici s pomočjo convert\_weight.py, ki nam bo nekako pretvoril tensorflow model kjer bomo jo rabili ( Carla ).

1. Luka Lašič

Vzpostavil sem razpoznavo pešcev, kot je videno sledeče



s pomočjo carla-pedestrian.py

Slika, ki vsebuje besede besedilo

Opis je samodejno ustvarjen

kjer povežem program s serverjem simulatorja Carla ter nato s pomočjo zajemanja framov ter ostalih funkcij razpoznava pešce (tudi motoriste / kolesarje) glede na meje proporcionalnosti pešca zazna njegovo obliko/prisotnost ter s pomočjo cv2 knjižnice, ki služi za preurejanje slik oziroma razumevanje slik, kajti slike/frame lahko dojemamo kot numpy arraye ter jih s cv2 nato lahko uredimo.

Zaenkrat se pešci obnašajo le kot vzorni državljani, torej še morem dodati nepričakovano obnašanje pešcev kot je npr: skakanje čez cesto

Datum 11. 5. do 17. 5

Spremenil sem obnašanje pešcev tako, da jih polovica ustvarjenih teka ter da vsi grejo skozi vsak prehod, kateri jim je najbljižje, kar nam ustvari okolje kjer mora voznik oziroma sistem pomoči pri vožnji biti toliko bolj previden na cesti, kajti se lahko vsak čas pojavi pešec pred nami. Sicer nisem naredil, da stečejo skozi cesto kjerkoli kadarkoli, kajti nisem našel koristne pomoči na internetu glede spremembe pešcevega vedenja (da bi šel cez cesto) za našo trenutno verzijo 0.9.13.

V prihodnje pa mi je cilj, da povežemo vse tri funkcionalnosti v skupen program ( trenutno še ne moremo, kajti še nimamo vsi dokončane razpoznave v carli) ter nato preklopiti na drugi del projektne naloge torej manevriranje z vozilom glede na določene situacije ( ustaviti avto, če zazna pešca spredaj).

Datum 18. 5. do 24. 5

Glede skalabilnosti simulatorja sem testiral odzivnost ter natančnost zaznave pri različnem številu drugih udeležencev prometa, pri čemer sem ugotovil, da ni zaznavne spremembe, kajti vseeno prepozna vse objekte v bližini v enakem času. Preveril sem pri 10 udeležencih, 30ih ter 90ih. Dodal sem še prepoznavo avtov/tovornjakov poleg pešcev, motoristov ter kolesarjev. Nažalost nisem uspel narediti, da bi avto avtomatsko sledil vozilu spredaj, kajti še nimam narejene funkcije, ki bi potrdila zaznavo ter nato manipulirala s hitrostjo ter smerjo avta. Sem pa našel način, da avto začne zavirati tako, da še mi manjka potrditev zaznave pešca ter nato zaviranje avta. Seveda pa morem prilagoditi, da bo zavirati začel le če se pešec nahaja pred nami ali pa hodi proti sprednji strani avta ( da ne bo avto zaviral četudi bo pešec daleč ob strani avta ).

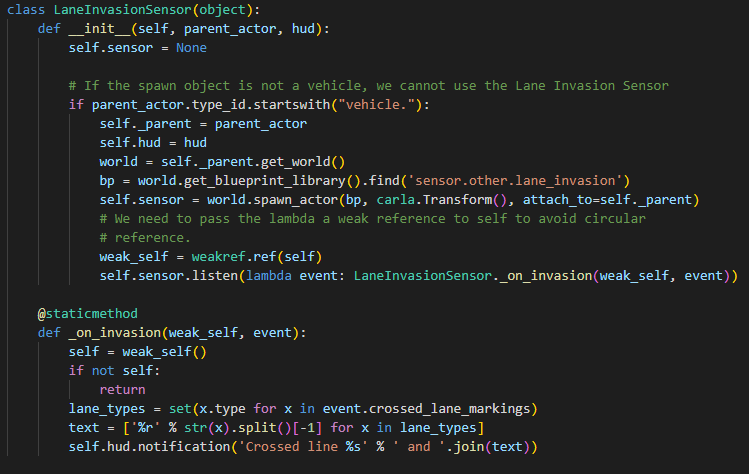
1. Uroš Kos

Ustvaril sem razpoznavo vseh vrst črt.





To sem storil s pomočjo LaneInvasionSensor funkcije, kjer sem pregledoval, če je vozilo dejansko prevozilo bilokakšno črto ki jo nato izpišemo glede na tip.



Poleg te funkcije pa se program prav tako poveze preko tcp povezave in obnašanje programa, v prihodnje moram dodati še obnašanje avtomobila oziroma funkcije v bližini križišč, definitivno pa bi na dvopasovnicah moral narediti še prilagoditev na smernike. V prihodnje pa bom tudi probal skreirati lastno mapo.

Datum 11. 5. do 17. 5

Slika, ki vsebuje besede besedilo, nebo, zunanje, prizor

Opis je samodejno ustvarjenSlika, ki vsebuje besede besedilo, zunanje, prizor, smer

Opis je samodejno ustvarjenTa teden sem se posvetil prepoznavi pasu oziroma cestnega pasu pred avtomobilom in izrisovanje le tega, sprva sem to storil po sliki nato se po videu. V prihodnje bom to realiziral se v carli, kar nebi smelo biti preveč težavno glede na to da tam tudi pridobivamo slike iz barvnih kamer in podobno.

18.5. do 25.5

Ta teden semi z slik in videa prestavo program za obdelavo crt v carlo, s tem sem imel kar nekaj tezav, saj mi je Carla zelo velik crashala, prav tako sem program moral popolno prirediti velikosti novih framov kamere in za enkrat to dela.

26.5. do 1.6

Ta teden sem moral malo boljse nacrtovati lane detection ker nisem gledal ce avto precka cesto zato sem ustvaril/nasel lane detection ki nam dejansko sporoci v kero smer moramo zapeljati, sedaj se samo dam noter da mi ob sporocilo avto dejansko zamakno za nek procent v levo ali desno smer.

<https://youtu.be/FXd1eWmrLHA>

Tim Vehovar

Vzpostavil sem YOLOv3 in TensorFlow da program zajema slike in na njih označuje objekte ter jih posreduje naprej. Najprej še je pa treba dodati v Carlo znake da jih bom lahko razpoznaval.



Program v katerem vozimo avto in ta zajema slike, ter program za prepoznavanje oblik znakov sta zaenkrat ločena in jih morem združiti da bosta delovala skupaj medtem ko se vozimo.

Datum 11. 5. do 17. 5

Algoritem iz prejšnjega tedna sem dodal v Carlo, za katero smo prav tako našli mapo, ki ima znake, da jih lahko prepoznavamo, ki jih na prejšnji mapi ni bilo.

A screenshot of a video game

Description automatically generated

A screenshot of a video game

Description automatically generated

A screenshot of a video game

Description automatically generated

Tako zdaj sproti v živo prepoznava in označuje znake, vendar je pa problem da še ne zna prepoznati kolikšna je hitrost napisana na znaku.

18.5. do 25.5

Na spletu sem našel bazo slik znakov in v obliki CSV označeni kateri znaki se kje na sliki nahajajo. Ti znaki so bili oblike ppm in sem jih najprej pretvoril v jpg. Nato sem se hotel lotiti učenja teh znakov v YOLO obliki, da bi dobil ustrezne uteži in datoteke, ki bi jih zamenjal z obstoječimi v našem programu. Vendar sem ugotovil, da to ne gre v obliki zapisa CSV in sem se lotil označevanja znakov v YOLO obliki s pomočjo labelImg. Označil sem dobrih 200 slik in jih s pomočjo roboflowa pretvoril v YOLO keras in YOLO darknet obliki, da bi lahko učil prepoznavanje znakov. Nato sem se lotil raziskovanja, kako sploh začnem z učenjem iz podanih primerov in sem kloniral nekaj githubov, vendar noben ni bil uspešen saj so manjkale datoteke do katerih nisem imel dostopa in sem jih potreboval za učenja. Poskusil sem tudi s pomočjo Google Colaboratory-ja, vendar je podan notebook namenjen učenju iskanja samo enega določenega predmeta, tako da še morem malce raziskati kako morem nadgraditi program da bo delal za moj primer.

A screenshot of a video game

Description automatically generated with low confidence

S pomočjo labelImg je za vsako sliko nastal posebej .txt file v katerem je napisan id znaka in pozicija v sliki.

Text

Description automatically generated

Nato sem pa te podatke spremenil v YOLO keras obliko.

Pri uporabi TrainYourOwnYOLO gita nisem imel dostopa do h5 datotek in nisem moral strenirati teh podatkov.

26.5. do 1.6

Zdaj se mi pri znaku prebere hitrost omejitve in se zadnja prebrana shrani, da vemo katero hitrost trenutno uporabljamo.   
Ko pridemo do znaka avto upočasni do hitrosti, ki smo jo prebrali in nam ne pusti pospešiti preko te hitrosti.

Speed limit:

<https://youtu.be/QIUy9W0iJ_4>

<https://youtu.be/KPLanlp9AMw>