



# Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης

Τμήμα Πληροφορικής

Πτυχιακή Εργασία

## Αυτόνομη Οδήγηση σε Γραφικό Περιβάλλον με Χρήση **YOLO**

Συγγραφέας:

Θωμάς Αλαβάνος  
ΑΕΜ: 3105

Επιβλέπων:

Αν. Καθ. Παναγιώτης Κατσαρός  
Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης

Σεπτέμβριος, 2022



## Περίληψη

Η έρευνα για την αυτόνομη οδήγηση και τον τομέα των αυτόνομων οχημάτων βρίσκεται σε διαρκή εξέλιξη. Καταβάλλονται επίμονες προσπάθειες από τους ανθρώπους, που διοικούν την εξέλιξη της αυτόνομης οδήγησης, να πραγματοποιηθεί ευαισθητοποίηση στην κοινότητα των οδηγών σχετικά με την ευθύνη που συνεπάγεται το να κατέχεις και να χυκλοφορείς στους δρόμους ένα όχημα. Στο πλευρό τους, οι ερευνητές εργάζονται σκληρά για να κάνουν τα οχήματα όσο το δυνατόν ασφαλέστερα προκειμένου να μειωθεί ο αριθμός των θανατηφόρων ατυχημάτων στους δρόμους μας. Σύμφωνα με το Εθνικό Συμβούλιο Ασφάλειας, το 2020 περίπου το 30% αυτών των ατυχημάτων είχαν ως βασικό παράγοντα την ταχύτητα [42]. Από μερικές δεκαετίες πριν μέχρι και στις μέρες μας, η ταχύτητα αποτελεί την κεντρικά ευρέως ερευνημένη θεματολογία από τις μεγάλες αυτοκινητοβιομηχανίες.

Η Ευρωπαϊκή Ένωση στις 6 Ιουλίου του 2022, ανακοίνωσε μέτρα που θα ληφθούν με στόχο να μειωθούν τα τροχαία ατυχήματα. Αξιοσημείωτο, είναι το μέτρο που αναγκάζει τα νέα οχήματα που θα κατασκευαστούν από εδώ και έπειτα να έχουν ενσωματωμένη μια έξυπνη συσκευή υποβοήθησης ταχύτητας να ενημερώνει τον οδηγό όταν υπερβαίνει το όριο ταχύτητας του δρόμου [43]. Ενδιαφέροντα μέτρα είναι τα χαρακτηριστικά όπως τα συστήματα διατήρησης λωρίδας κυκλοφορίας και η αυτόματη πέδηση. Επιπλέον, θα υπάρχουν τεχνολογίες για την καλύτερη αναγνώριση πιθανών τυφλών σημείων, προειδοποιήσεις για την αποφυγή συγκρούσεων με πεζούς ή ποδηλάτες και συστήματα παρακολούθησης για την ελαστική πίεση [43]. Δυστυχώς, τα παλαιότερα αυτοκίνητα θα παραμείνουν χωρίς αυτά τα συστήματα.

Η συγκεκριμένη πτυχιακή εργασία θέλει να συμβάλει σε αυτήν την έρευνα ενσωματώνοντας σε ένα ρεαλιστικό προσομοιωτή οδήγησης, ένα σύστημα ικανό να ανιχνεύει και να αναγνωρίζει τους σηματοδότες και τα χρώματα που έχουν την δεδομένη στιγμή στο δρόμο καθώς και αμάξια που βρίσκονται σε επικάνδυνα κοντινή απόσταση στο δρόμο. Με αυτές τις πληροφορίες, το σύστημα θα λαμβάνει αποφάσεις που βοηθούν τον χρήστη/οδηγό να διευκολύνει τη διαδικασία οδήγησης και να την καθιστά ασφαλέστερη.

Ο προσομοιωτής που χρησιμοποιήθηκε σε αυτή τη μελέτη είναι το CAR Learning to Act (CARLA) [14], ένας προσομοιωτής ανοιχτού κώδικα για αυτόνομη έρευνα που αποτελείται χυρίων από δύο ενότητες: τον προσομοιωτή CARLA και τη μονάδα CARLA Python API. Για να ανιχνευτούν στο δρόμο οι σηματοφόροι με τα χρώματα τους καθώς και τα αμάξια, χρησιμοποιείται ένας αλγόριθμος αιχμής για την ανίχνευση αντικειμένων: ο αλγόριθμος You Look Only Once (YOLO). Αντί να χρησιμοποιείται ένα συρόμενο παράθυρο σε πολλές τοποθεσίες μέσα σε μια εικόνα, ο YOLO βλέπει ολόκληρη την εικόνα κατά τη διάρκεια της φάσης εκπαίδευσης και δοκιμής, κωδικοποιώντας πληροφορίες σχετικά με τις κατηγορίες αντικειμένων καθώς και την συχνότητα των εμφανίσεων τους. Αυτό το χαρακτηριστικό του επιτρέπει να είναι εξαιρετικά γρήγορος αλγόριθμος κατά την αξιολόγηση και ταξινόμηση μιας εικόνας.

Η αναγνώριση των σηματοφόρων με τα χρώματα τους εντός δρόμου καθώς και των κοντινών αμαξιών, μπορεί να έχει ένα ευρύ φάσμα βιοηθητικών υπηρεσιών. Σε αυτή την πτυχιακή, τέσσερις εφαρμογές εκτίθενται: μια προειδοποιητική εφαρμογή, για να ειδοποιεί τον χρήστη ότι υπάρχει σηματοδότης και τι χρώμα έχει καθώς και για την ύπαρξη κοντινού αμαξιού, μια εφαρμογή ελέγχου, που θα ελέγχει την ορθή διαδικασία του φρεναρίσματος αν αντικριστεί κόκκινος και κίτρινος σηματοδότης ή υπερβολικά κοντινό αμάξι. Η τρίτη εφαρμογή αφορά την εμφάνιση του πλαισίου του αντικειμένου που εντοπίζει ο YOLO. Τέλος, η τελευταία εφαρμογή αφορά την προσπέλαση των δεδομένων που καταμετρά αριθμητικά τις συχνότητες εμφανίσεων τιμών και εξάγει στατιστικά δεδομένα. Τα αποτελέσματα δείχνουν ότι η διαδικασία ανίχνευσης επιτυγχάνεται ικανοποιητικά με ακρίβεια 95.84%.

## *Summary*

Research into autonomous driving and the field of autonomous vehicles is constantly growing. Efforts are being made by the people who manage the development of autonomous driving to create awareness among the driving community about the responsibility that comes with owning and driving a vehicle. On their side, researchers are working hard to make vehicles as safe as possible in order to reduce the number of fatal accidents in our roads. According to the National Safety Council, in 2020 about 30% of these crashes had speed as a primary factor [42]. From a few decades ago to the present day, speed has been the central widely researched topic by the major car manufacturers.

The European Union on July 6, 2022, announced measures taken to reduce road accidents. Worthy of comment, it is the measure which forces new vehicles from now on to have a smart speed assist device built in, to inform the driver when the road speed limit is exceeded [43]. Interesting measures are the features such as lane keeping systems and automated braking. Furthermore, there will be technologies for better recognising possible blind spots, warnings to prevent collisions with pedestrians or cyclists and tyre pressure monitoring systems [43]. Unfortunately, older cars will remain without these systems.

This thesis aims to contribute to this research by integrating into a realistic driving simulator, a system capable of detecting and recognizing the signals and their colors on the road as well as cars that are dangerously close. With this information, the system will make decisions helping the user/driver to make the driving process easier and safer.

The simulator used in this study is CAR Learning to Act (CARLA) [14], an open source simulator for autonomous research that mainly consists of two modules: the CARLA simulator and module CARLA Python API. In order to detect traffic lights with their colors as well as cars on the road, a state-of-the-art object detection algorithm is used: the You Look Only Once (YOLO) algorithm. Instead of using a sliding window at multiple locations within an image, YOLO looks at the entire image during the training and testing phase, encoding information about object classes as well as their frequency of occurrence. This feature makes YOLO to be an extremely fast algorithm when evaluating and classifying an image.

The recognition of traffic lights with their colors on the road and nearby cars can have a wide range of auxiliary services. In this thesis, four applications are exposed: a warning application, to notify the user that there is a traffic light and what color it is, as well as the presence of a nearby car, a control application, which will check the correct braking process if red and yellow are encountered in a traffic light or if a car is too close. The third application concerns the display of the context of the object detected by YOLO. Finally, the last application concerns the data access which numerically counts

the frequencies of values occurrences and extracts statistical data. The results show that the detection process is achieved satisfactorily with 95.84% accuracy.

# Περιεχόμενα

Περίληψη	ii
<b>Summary</b>	iv
1 Εισαγωγή	1
1.1 Επιστημονική Περιοχή . . . . .	5
1.2 Κοινωνικός Αντίκτυπος . . . . .	7
1.2.1 Θετικός Αντίκτυπος . . . . .	7
1.2.2 Αρνητικός Αντίκτυπος . . . . .	7
1.3 Αρχιτεκτονική Τλοποίησης Αυτόνομου Οχήματος . . . . .	8
1.3.1 Input . . . . .	9
1.3.2 Processing . . . . .	10
1.3.3 Output . . . . .	10
1.4 Σκοπός και Συνεισφορά της Πτυχιακής Εργασίας . . . . .	11
2 Μηχανή Προσομοίωσης <b>Carla</b>	13
2.1 Γενικές Πληροφορίες . . . . .	13
2.2 Επικοινωνία Carla με τον Υπολογιστή . . . . .	14
2.3 Δυνατότητες . . . . .	16
2.4 Συνθήκες Περιβάλλοντος . . . . .	17
2.4.1 Καιρικές Συνθήκες . . . . .	18
2.4.2 Τηλικές Υποδομές . . . . .	18
2.4.3 Αλληλεπίδραση με Actors . . . . .	18
3 Τλοποίηση Μοντέλου Αντίληψης	20
3.1 Darknet και YOLO . . . . .	21
3.2 Τροποποιήσεις Δικτύου και Υπερπαραμέτρων . . . . .	23
3.3 Διαδικασία Εκπαίδευσης Δικτύου . . . . .	25
3.3.1 Συλλογή Δεδομένων . . . . .	25
3.3.2 Χρήση labelImg . . . . .	27
3.3.3 Εκπαίδευση YOLO . . . . .	28
3.4 Εξαγωγή Αποτελεσμάτων μέσω Darknet . . . . .	30
4 Εφαρμογές στο <b>Carla</b>	32
4.1 Εφαρμογή προειδοποίησης . . . . .	32
4.1.1 Εντοπισμός Σηματοδότη . . . . .	33
4.1.2 Εντοπισμός οχήματος . . . . .	34
4.1.3 Λανθασμένη Προσπέραση Σηματοδότη . . . . .	36
4.2 Εφαρμογή Ελέγχου . . . . .	38
4.3 Εφαρμογή Πλαισίου Αντικειμένων . . . . .	39

4.4 Εφαρμογή Προσπέλασης Δεδομένων . . . . .	41
4.4.1 Προσπέλαση Καταμέτρησης του Dataset . . . . .	41
4.4.2 Προσπέλαση Αποδοτικότητας σε Test Dataset . . . . .	42
5 Πειράματα και Αποτελέσματα . . . . .	44
5.1 Πειράματα . . . . .	44
5.1.1 Εκπαίδευση με διαφορετικό Dataset . . . . .	44
5.1.2 Άλλαγές στις Υπερπαραμέτρους . . . . .	45
5.1.3 Άλλαγές στις Συνθήκες Περιβάλλοντος . . . . .	46
5.1.4 Δοκιμές του YOLO υπό διάφορες περιπτώσεις . . . . .	47
5.2 Αποτελέσματα . . . . .	48
5.2.1 Στατιστικά Αποτελέσματα και Συγχρίσεις . . . . .	48
5.2.2 Αποτελέσματα μέσω του Carla σε real-time . . . . .	51
5.2.3 Σχολιασμός και Σημασία Αποτελεσμάτων . . . . .	56
6 Σύνοψη και Μελλοντική Εργασία . . . . .	57
A' Εγκατάσταση λογισμικού . . . . .	59
A'.1 Εγκατάσταση Darknet . . . . .	59
A'.2 Εγκατάσταση Carla . . . . .	60
A'.3 Εγκατάσταση labelImg . . . . .	60
Βιβλιογραφία . . . . .	62

# Κατάλογος Σχημάτων

1.1 Δρόμος πόλης στο Carla [14] . . . . .	5
1.2 Αρχιτεκτονική του YOLO [48] . . . . .	6
1.3 Συνδεδεμένη Αρχιτεκτονική Συστήματος Αυτόνομου Οχήματος [11] . . . . .	9
2.1 Επικοινωνία Προσομοιωτή με APIs [63] . . . . .	16
3.1 Οπτική λειτουργία του συνολικού έργου . . . . .	20
3.2 Συνολική δομή του YOLOv4 [32] . . . . .	21
3.3 Παράδειγμα εικόνας εκπαίδευσης . . . . .	26
3.4 Προσθήκη ετικέτας σε πράσινο σηματοδότη μέσω labelImg. . . . .	28
3.5 Διάγραμμα μέσου σφάλματος εκπαίδευσης . . . . .	29
3.6 Εντοπισμός οχήματος με 98% βεβαιότητα . . . . .	31
4.1 Μήνυμα προειδοποίησης κόκκινου σηματοδότη μέσα από παράθυρο pygame. . . . .	34
4.2 Μήνυμα προειδοποίησης κοντινού μπροστινού οχήματος μέσα από παράθυρο pygame. . . . .	35
4.3 Απόσπασμα κώδικα της περίπτωσης που δεν εντοπίστηκε κανένα αντικείμενο ενώ πρίν η ένδειξη έδειχνε κόκκινο ή κιτρινο σηματοδότη. . . . .	37
4.4 Απόσπασμα κώδικα ελέγχου του οχήματος. . . . .	38
4.5 Απόσπασμα κώδικα δημιουργίας πλαισίων αντικειμένων. . . . .	40
4.6 Απόσπασμα κώδικα προσπέλασης δεδομένων από access.py . . . . .	42
5.1 Διάγραμμα μέσου σφάλματος εκπαίδευσης για Δίκτυο 2 . . . . .	50
5.2 Διάγραμμα μέσου σφάλματος εκπαίδευσης για Δίκτυο 3 . . . . .	51
5.3 Διάγραμμα σχέσης penalty-actors. . . . .	52
5.4 Πείραμα απόδοσης εντοπισμού σηματοδότη από YOLO υπό φυσιολογική ταχύτητα αυτόματου πιλότου. . . . .	53
5.5 Πείραμα απόδοσης εντοπισμού σηματοδότη από YOLO υπό αργή ταχύτητα. . . . .	54
5.6 Πείραμα απόδοσης εντοπισμού σηματοδότη από YOLO υπό γρήγορη ταχύτητα. . . . .	54
5.7 Πείραμα απόδοσης εντοπισμού σηματοδότη από YOLO υπό την αναμονή πεζού να διασχίσει τον δρόμο. . . . .	55
5.8 Πείραμα απόδοσης ταυτόχρονου εντοπισμού σηματοδότη και οχήματος από YOLO. . . . .	55
A'1 Τροποποιήσεις στο CMake. . . . .	59
A'2 Δημιουργία εκτελέσιμου αρχείου μέσω Visual Studio. . . . .	60
A'3 . . . . .	61

# Κατάλογος Πινάκων

3.1	Τιμές διαμόρφωσης του YOLO από αρχείο yolov4-objs.cfg . . . . .	24
3.2	Ποσοτική ανάλυση του συνόλου εκπαίδευσης . . . . .	26
5.1	Ποσοτική ανάλυση του Dataset 2 . . . . .	45
5.2	Τιμές διαμόρφωσης του YOLO από τροποποιημένο αρχείο υπερπαραμέτρων. . . . .	46
5.3	Γενική ακρίβεια των 3 δικτύων. . . . .	49
5.4	Ακρίβεια πρόβλεψης κάθε κλάσης από τα 3 δίκτυα. . . . .	49
5.5	Πίνακας μέσου σφάλματος για τα 3 δίκτυα . . . . .	52
5.6	Αποτελέσματα του πειράματος της υποενότητας 5.1.3. . . . .	52

## Κεφάλαιο 1

### Εισαγωγή

Τα τελευταία χρόνια, τα αυτοοδηγούμενα αμάξια έχουν γίνει ένα από τα πιο πολυσυζητημένα και ενεργά θέματα όσον αφορά τις έρευνες περί αυτών. Από κάθε οπτική γωνία, αυτά τα ανεπτυγμένα συστήματα, τα οποία φέρουν την τρίτη ρομποτική επανάσταση, ανήκουν στον τομέα της ρομποτικής. Αυτό σαν δήλωση παραμένει αληθής παρόλο που συνηθίζεται από αρκετά μεγάλη μάζα ανθρώπων να τα κατατάσσουν στον τομέα της αυτοκινητοβιομηχανίας [13]. Η προσπάθεια επίτευξης του σύνθετου έργου της ανθρώπινης οδήγησης να πραγματοποιείται από ένα εξελιγμένο αυτόνομο σύστημα, έχει ως επακόλουθο να παρουσιάζονται αμέτρητες δημιουργικές προκλήσεις, οι οποίες περιλαμβάνουν το ευρύτερο πεδίο της ρομποτικής, συμπεριλαμβανομένης της αντίληψης του περιβάλλοντος, της λήψης αποφάσεων και του ελέγχου.

Ενώ τα περισσότερα από τα σημερινά πρωτότυπα είναι σε θέση να θριαμβεύσουν λειτουργικά υπό διάφορες συνθήκες, τα αυτοκινούμενα οχήματα ως καταναλωτικά προϊόντα απέχουν ακόμη αρκετά χρόνια από τη μαζική παραγωγή, χυρίως λόγω των αυστηρών και συνεχώς εξελισσόμενων κανονισμών και πρωτοκόλλων δοκιμών που απαιτούνται από τις αυτοκινητιστικές εταιρείες. Μέχρι τότε, νέες λειτουργίες θα συνεχίσουν να εισάγονται στα οχήματα σταδιακά, απαιτώντας ανθρώπινη επίβλεψη και αποτελεσματική συνεργασία ανθρώπου-ρομπότ μέσω εξελιγμένων διεπαφών.

Η αυτόνομη οδήγηση είναι ένας από τους βασικούς τομείς εφαρμογής της τεχνητής νοημοσύνης. Τα αυτόνομα οχήματα είναι εξοπλισμένα με πολλαπλούς αισθητήρες, όπως κάμερες, ραντάρ και lidar, που τους βοηθούν να κατανοήσουν καλύτερα το περιβάλλον και το πλάνο της διαδρομής τους. Αυτοί οι αισθητήρες παράγουν τεράστιο όγκο δεδομένων. Για να κατανοήσουν τα δεδομένα που παράγονται από αυτούς τους αισθητήρες, τα αυτοκίνητα αυτά χρειάζονται δυνατότητες επεξεργασίας παρόμοιες με έναν υπερυπολογιστή [6]. Οι εταιρείες που αναπτύσσουν τέτοια συστήματα βασίζονται σε μεγάλο βαθμό στην τεχνητή νοημοσύνη, με τη μορφή μηχανικής μάθησης και βαθιάς εκμάθησης, για την αποτελεσματική επεξεργασία του τεράστιου όγκου δεδομένων και την εκπαίδευση και επικύρωση των αυτόνομων συστημάτων οδήγησής τους.

Από τον Μάρτιο του 2018, 52 εταιρείες διέθεταν την άδεια να δοκιμάζουν αυτόνομα οχήματα στους δρόμους της Πολιτείας της Καλιφόρνιας [60] [68]. Παραδόξως, λίγοι απλοί πολίτες έχουν βιώσει ακόμα διαδρομές με αυτόνομο όχημα. Αυτή η προσωπική απειρία μπορεί να δυσχεράνει τον γενικό πληθυσμό να κρίνει τη δυνητική χρησιμότητα, καλώς ή κακώς, τέτοιων οχημάτων. Ωστόσο, ο κοινωνικός αντίκτυπος αυτών των οχημάτων θα είναι σίγουρα πιο εκτεταμένος από μια απλή αλλαγή στο ταξίδι μεταξύ της άμεσης προέλευσης και προορισμού. Για

παράδειγμα, τα επόμενα χρόνια μπορεί να μην είναι απαραίτητο να κατέχουν αυτοκίνητο οι πολίτες, ειδικά όταν θα μπορούν να καλέσουν ένα χρησιμοποιώντας μια εφαρμογή smartphone και έχοντας πλήρη εμπιστοσύνη ότι θα φτάσει μέσα σε λίγα λεπτά ή και δευτερόλεπτα. Ορισμένες μελέτες έχουν δείξει ότι έως και το 30% ή περισσότερο της κυκλοφορίας που κυκλώνει τους δρόμους στο κέντρο της πόλης στην πραγματικότητα αναζητά χώρο στάθμευσης [55]. Η αναζήτηση μπορεί να γίνει περιττή όταν το όχημα οδηγεί το ίδιο για να παραλάβει τον επόμενο χρήστη του, όπως φαίνεται να υποδηλώνουν κάποιες προβλέψεις σχετικά με τη χρήση εφαρμογών όπως το Uber [39]. Φυσικά, τα προβλήματα στάθμευσης δεν είναι σε καμία περίπτωση η μόνη διάσταση της αλλαγής που θα επέλθει από τα αυτοοδηγούμενα αμάξια.

Ένα σημαντικό ζήτημα σε μια τέτοια θεματολογία είναι το τι καταλαβαίνουν οι άνθρωποι όταν ακούνε αυτόνομο αμάξι. Μπορεί το κοινό να θεωρεί ότι τα οχήματα αυτά δεν απαιτούν καμία απολύτως είσοδο οδηγού. Ωστόσο, αυτή η αντίληψη δεν καταγράφει πολλές από τις σημαντικές διαφορές μεταξύ των προτεινόμενων αυτόνομων αμάξιών και των σημερινών, ημιαυτόματων οχημάτων στο δρόμο. Τα ημιαυτόματα οχήματα παρέχουν διάφορες μορφές βοήθειας οδηγού για να βοηθήσουν τον οδηγό που παραμένει στον απόλυτο έλεγχο. Τα πλήρως αυτόνομα οχήματα έχουν σχεδιαστεί για να οδηγούν μόνα τους. Αυτές οι διαφορετικές μορφές προώθησης οχημάτων έχουν ταξινομηθεί σε μια ιεραρχία που συγκρίνει τον έλεγχο του οδηγού με τον έλεγχο του οχήματος. Η ιεραρχία περιγράφεται στα επίπεδα ελέγχου της Εταιρείας Μηχανικών Αυτοκινήτου (Society of Automotive Engineers SAE) [56] [24]. Παρόλο που δεν θα αναλυθεί συγκεκριμένα καθένα από αυτά τα επίπεδα σε αυτη την πτυχιακή εργασία, είναι ζωτικής σημασίας να σημειωθεί ότι πολλές δημόσιες παραδοχές σχετικά με προηγμένες δυνατότητες οχημάτων μπορεί να είναι άστοχες. Έτσι, τα άτομα μπορεί κάλλιστα να υποθέσουν ότι τέτοια αυτοοδηγούμενα αμάξια διαθέτουν πολύ περισσότερη ευφυΐα και λειτουργική ικανότητα από ό, τι στην πραγματικότητα. Τέτοιες υποθέσεις μπορεί να αποδειχθούν κρίσιμες, αν όχι μοιραίες.

Το αυτοκίνητο χωρίς οδηγό έχει τη δυνατότητα να κάνει τον ανθρώπινο χειριστή του τόσο περιττό όσο έγινε το άλογο για την μεταφορά κατά την τεχνολογική ανάπτυξη. Οι οδηγοί φορτηγών και ταξί μπορεί να χρειαστεί να βρουν νέες μορφές απασχόλησης, μερικοί ίσως επιβλέποντας αυτά τα μεμονωμένα οχήματα από τηλεφωνικά κέντρα τηλεχειρισμού. Ωστόσο, οι θέσεις εργασίας στον νέο τομέα των μεταφορών μπορεί κάλλιστα να μειωθούν, καθώς θα έχουν αλλάξει ριζικά σε άλλους τομείς η αυτοματοποίηση και η αναδυόμενη αυτονομία μηχανών [23]. Είναι αλήθεια ότι θα δημιουργηθούν ορισμένες θέσεις εργασίας, π.χ. στη διατήρηση τέτοιων πολυπληθών αυτόνομων οχημάτων και η πρόσβαση στην μεταφορά για όσους βρίσκονται σε οικονομικά υποβαθμισμένες περιοχές θα μπορούσε να βελτιωθεί με τις υπηρεσίες αυτόνομων οχημάτων. Μελέτες [15] [3] δείχνουν ότι, γενικά, οι κοινωνικές αλλαγές που προκύπτουν από την εισαγωγή αυτών των καινοτομιών είναι πιθανό να είναι εκτεταμένες. Φυσικά, είναι πιθανό ότι πολλά οχήματα που οδηγούνται από τον άνθρωπο θα παραμείνουν στους δρόμους για τις επόμενες δεκαετίες. Τα αυτοκίνητα χωρίς οδηγό μπορούν να παρέχουν κινητικότητα σε όσους δεν μπορούν να οδηγήσουν σωματικά, όπως παιδιά, άτομα με ειδικές ανάγκες ή αδύναμοι ηλικιωμένοι. Ωστόσο, για τέτοιους πληθυσμούς τα προβλήματα εισόδου και εξόδου από το όχημα παραμένουν, τονίζοντας ότι η κινητικότητα είναι κάτι

περισσότερο από το ταξίδι μόνο με το αυτοκίνητο.

Οι μηχανικοί που αναπτύσσουν αυτόνομα αυτοκίνητα πρέπει να εξετάσουν σοβαρά ηθικά ερωτήματα. Συνήθως, αυτά θεωρούνται παραλλαγές στο πείραμα ηθικής σκέψης γνωστό ως «πρόβλημα του τρόλεϊ». Πρέπει ένα αυτοκίνητο τεχνητής νοημοσύνης να στραφεί και να τραυματίσει έναν πεζό εάν η εναλλακτική λύση είναι να συνεχίσει ευθεία μπροστά και να τραυματίσει περισσότερους αριθμούς. Πρέπει ένα αυτοκίνητο χωρίς οδηγό να προστατεύει τον επιβάτη πάνω από όλα ή να θυσιάσει τον άνθρωπο μέσα για το καλύτερο καλό σε τέτοιες συνθήκες. Πιο συγκεκριμένα, πώς ακριβώς κωδικοποιούμε αυτές τις αντίστοιχες ηθικές και ηθικές αρχές σε μια διάταξη λογισμικού που δημιουργείται συχνά από πολλούς σχεδιαστές και προγραμματιστές. Τέτοια προβλήματα είναι πολύ πιο πολύπλοκα από την απλή διχοτόμηση που εκφράζεται στο πρόβλημα του τρόλεϊ. Είναι σχεδόν βέβαιο ότι η επίλυση τέτοιων ζητημάτων πρέπει να υπερβαίνει κατά πολύ τη δομή του ίδιου του προγραμματισμού για να επιτευχθεί πλήρης δημόσια αποδοχή [16]. Μια μελέτη που έθεσε τέτοιες ερωτήσεις σε αρκετές εκατοντάδες εργαζόμενους μέσω της υπηρεσίας Amazon Mechanical Turk έδειξε ότι το κοινό παραμένει αντιφατικό για τέτοια θέματα [53]. Οι άνθρωποι είναι υπέρ των αυτοκινήτων που θυσιάζουν τον επιβάτη για να σώσουν άλλους ανθρώπους, αλλά δεν θα ήθελαν να είναι επιβάτες σε ένα αυτοκίνητο προγραμματισμένο με αυτόν τον τρόπο. Αυτές οι επιλογές είναι οι τύποι σεναρίων που εξετάζουν οι μηχανικοί κατά το σχεδιασμό των τεχνητά ευφυών μηχανών τους, αλλά ο καθοριστικός προγραμματισμός που θα καθιερωθεί ακόμα δεν έχει διαμορφωθεί. Φυσικά, αυτή τη στιγμή οι περισσότεροι άνθρωποι εξακολουθούν να στερούνται άμεσης εμπειρίας από αυτοοδηγούμενα αμάξια. Έρευνα κινδύνου υποδηλώνει ότι η εμπειρία και οι πληροφορίες θα οδηγήσουν τους ανθρώπους να αρχίσουν να δέχονται αυτά τα αμάξια, αλλά αυτή η τάση δεν ισχύει πάντα σε όλα τα πλαίσια [16] [61]. Αυτή η αποδοχή εξαρτάται από τον στρόβιλο της κοινής γνώμης και τέτοιες απόψεις βασίζονται στο πώς οι άνθρωποι βιώνουν τις δικές τους αλληλεπιδράσεις με ουσιαστικά όλες τις μορφές της τρέχουσας τεχνολογίας.

Τηπάρχουν πολλές προβλέψεις για το πώς τα αυτόνομα αυτοκίνητα θα αλλάξουν τη ζωή μας, βασισμένες σε προσεγγίσεις τόσο από φιλοσοφική όσο και από επιστημονική άποψη. Η σύγχρονη κοινωνία χαρακτηρίζεται από την κινητικότητα, με τα αυτοκίνητα να παίζουν σημαντικό ρόλο με περίπου 1 δισεκατομμύριο από αυτά να χρησιμοποιούνται παγκοσμίως κατακλύζοντας τους δρόμους, τα οποία προβλέπεται να ξεπεράσουν σε αριθμό τα 2.5 δισεκατομμύρια το 2050 [49]. Σύμφωνα με τους ειδικούς, ο εξοπλισμός αυτών των οχημάτων με “έξυπνο οδηγό” ή με πλήρως αυτόνομες λειτουργίες θα οδηγήσει τελικά σε πτώση των ποσοστών τροχαίων ατυχημάτων κατά 90% και σε μείωση κατά 60% των εκπομπών διοξειδίου του αυτοκινήτου (λόγω του αποτελεσματικού σχεδιασμού της τροχιάς), γεγονός που καθιστά τα αυτόνομα οχήματα αρκετά φιλικά προς το περιβάλλον με άμεσο θετικό αντίκτυπο στον άνθρωπο και στην φύση [5].

Με βάση τα παραπάνω, το κύριο κίνητρο για αυτή την πτυχιακή εργασία θα αφορά την κατασκευή ενός συστήματος ικανό να ανιχνεύει και να αναγνωρίζει σηματοφόρους καιθώς και το χρώμα που θα έχουν, μαζί με αμάξια τα οποία βρίσκονται σε κοντινή απόσταση. Ο απότερος σκοπός είναι στο να χρησιμεύσουν να φρενάρει

το αμάξι τις κατάλληλες στιγμές ώστε να μην υπάρξει κάποιο ατύχημα. Τα συστήματα αυτά θα εφαρμοστούν σε μη αυτόνομο όχημα ώστε να μπορούν αντικειμενικά να αξιολογηθούν οι επιδόσεις τους. Λόγω του γεγονότος ότι η ανάπτυξη και ο έλεγχος αλγορίθμων για αυτόνομα οχήματα στον πραγματικό κόσμο είναι μια δαπανηρή και χρονοβόρα διαδικασία, θα χρησιμοποιηθεί ένας προσομοιωτής. Επίσης, ένα ακόμα γεγονός που κινητροδοτεί την υλοποίηση αυτής της πτυχιακής είναι ότι όλα τα αναπτυγμένα εργαλεία λογισμικού που θα εφαρμοστούν στον προσομοιωτή θα είναι δωρεάν διαθέσιμα και ανοιχτού κώδικα, συμβάλλοντας με αυτόν τον τρόπο στην κοινότητα της ερευνητικής αυτόνομης οδήγησης. Για να γίνει η ανίχνευση των σηματοδοτών, των χρωμάτων τους και των κοντινών αμάξιών, το σύστημα που θα αναπτυχθεί θα επωφελείται από το πόρους του υπολογιστή για να πραγματοποιεί ανίχνευση αντικειμένων μέσα σε μια εικόνα. Το μοντέλο που θα εκπαίδευται, θα πρέπει να λειτουργεί σε πραγματικό χρόνο, να είναι ανοιχτού κώδικα, να έχει αποτελέσματα υψηλής ακρίβειας, να είναι εύκολο στη χρήση και εύκολο να ενσωματωθεί σε έναν κώδικα Python καθώς και να διαθέτει υψηλή ικανότητα γενίκευσης. Σε αυτό το σημείο πρέπει να αναφερθεί ότι οι εικόνες που θα χρησιμοποιηθούν για τη φάση εκπαίδευσης, επικύρωσης και δοκιμής θα είναι αποκλειστικά μέσα από τον προσομοιωτή με ένα ευρύ φάσμα διάφορων καιρικών συνθηκών.

Το object-detection model που θα χρησιμοποιηθεί και ταιριάζει καλύτερα σε όλες αυτές τις αναφερόμενες απαιτήσεις είναι το σύστημα You Only Look Once (YOLO) [48]. Ο YOLO βλέπει ολόκληρη την εικόνα κατά την διάρκεια των φάσεων εκπαίδευσης και δοκιμών, κωδικοποιώντας πληροφορίες σχετικά με τις κλάσεις αντικειμένων καθώς και τις εμφανίσεις τους χρησιμοποιώντας ένα συρόμενο παράθυρο σε πολλές θέσεις σε μια εικόνα, έτσι ώστε με την χρήση ενός μόνο νευρωνικού δικτύου να γίνεται η πρόβλεψη των οριακών πλαισίων και πιθανοτήτων κλάσης απευθείας από πλήρεις εικόνες σε μια αξιολόγηση. Με αυτόν τον τρόπο, αποφεύγεται να εκτελείται ένας ταξινομητής χιλιάδες φορές πάνω από μια εικόνα.

Όσον αφορά τον προσομοιωτή, υπάρχουν μεγάλες εταιρείες όπως η Microsoft ή η Toyota που κάνουν τεράστιες προόδους στην έρευνα για την αυτόνομη οδήγηση. Και οι δύο έχουν εφαρμόσει τους δικούς τους προσομοιωτές ανοιχτού κώδικα, βασισμένους στη μηχανή παιχνιδιών Unreal Engine [65] για την υποστήριξη της ανάπτυξης, της εκπαίδευσης και της επικύρωσης των συστημάτων αυτόνομης οδήγησης. Η Microsoft ξεκίνησε το 2017 την οπτική και φυσική προσομοίωση για αυτόνομα οχήματα AirSim [51] και το ερευνητικό ίνστιτούτο Toyota τον προσομοιωτή CARLA [14]. Και τα δύο μοιάζουν αρκετά καθώς είναι χτισμένα σε Unreal Engine, έχουν ευέλικτα APIs όπου οι χρήστες μπορούν να ελέγχουν όλες τις πτυχές που σχετίζονται με την προσομοίωση και μπορούν να ορίσουν διαφορετικών ειδών αισθητήρων, συμπεριλαμβανομένων και των LIDARs μεταξύ άλλων. Οι λόγοι οι οποίοι καθιστούν εύλογη την χρήση του προσομοιωτή CARLA σε αυτή την πτυχιακή εργασία είναι ότι έχει μια μεγάλη κοινότητα χρηστών/ερευνητών για αλληλεπίδραση με προγραμματισμένο χάρτη πορείας. Άξιο λόγου είναι κιόλας ότι το CARLA είναι πολύ γνωστό από όλους στην κοινότητα έρευνας αυτόνομης οδήγησης και αναπτύσσεται συνεχώς από ερευνητές του Ισπανικού πανεπιστημίου: the Autonomous University of Barcelona [28]. Το CARLA όπως φαίνεται στο σχήμα 1.1 παρέχει ρεαλιστικές αστικές διατάξεις (συμπεριλαμβανομένων κτιρίων, οχημάτων, πεζών) και ένα ευρύ φάσμα περιβαλλοντικών συνθηκών, γεγονός που



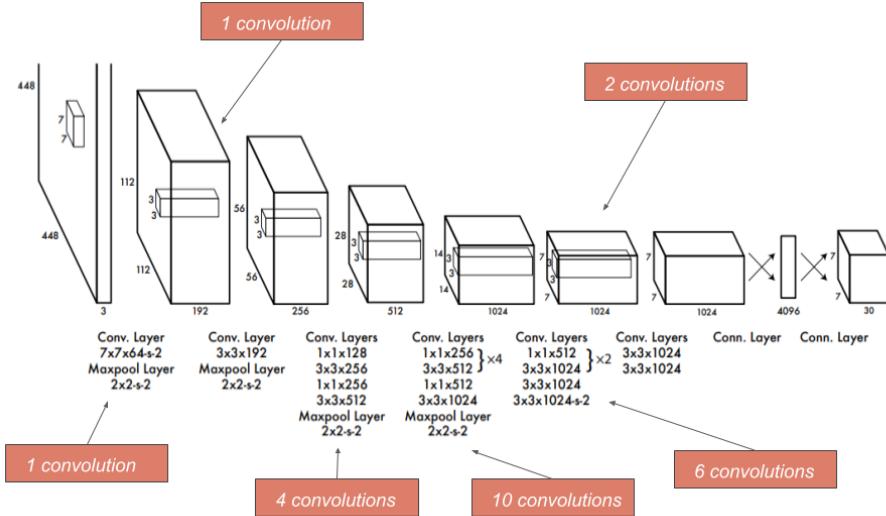
Σχήμα 1.1: Δρόμος πόλης στο Carla [14]

αναδεικνύει την ομοιότητα του με τον πραγματικό κόσμο. Αυτός ο υψηλός βαθμός ομοιότητας επιτρέπει την αναπαραγωγή του συστήματος που θα αναπτυχθεί σε αυτή την πτυχιακή να δύναται να εφαρμοστεί και σε ένα πραγματικό αυτοκίνητο.

## 1.1 Επιστημονική Περιοχή

Τα τελευταία χρόνια, η ανίχνευση αντικειμένων αποτελεί πρόβλημα για την αποτελεσματική “όραση” του υπολογιστή. Υπάρχουν πολλά μοντέλα για την αναγνώριση διαφορετικών αντικειμένων τα οποία αναπτύχθηκαν παρέχοντας τιμές υψηλής ακρίβειας. Ωστόσο, η πλειοψηφία αυτών απασχολεί τις ίδιες τεχνικές: εξαγωγή ενός συνόλου ισχυρών χαρακτηριστικών από την εικόνα εισόδου (όπως SIFT [34], Haar [64], HOG [17], Convolutional Layers [50]), χρήση ταξινομητών ή τοπικοποιητών ώστε να προσδιορίζουν τα επιθυμητά αντικείμενα στον εικονικό χώρο του προσομοιωτή με την εκτέλεση τους να λαμβάνει χώρα σε διάφορες τοποθεσίες και κλίμακες που αφορούν μια δοκιμαστική εικόνα. Υπάρχουν πιο πρόσφατες προσεγγίσεις όπως το R-CNN [35] που χρησιμοποιεί μεθόδους πρότασης περιοχής ενδιαφέροντος για να δημιουργεί πρώτα δυνητικά όρια πλαισίων σε μια εικόνα και στη συνέχεια να εκτελεί έναν ταξινομητή σε αυτά τα προτεινόμενα πλαισία. Και για τις δύο περιπτώσεις, το να συμβαίνει χιλιάδες φορές η εκτέλεση ενός ταξινομητή σε μια εικόνα συνεπάγει χιλιάδες αξιολογήσεις νευρωνικών δικτύων για την ανίχνευση αντικειμένων. Αυτή η διαδικασία απαιτεί πολλούς υπολογιστικούς πόρους, αποτρέπει τη γενίκευση και μπορεί να εισάγει μια καθυστέρηση μεγαλύτερη από την αναπαραγωγή των αντικειμένων. Επιπλέον, χρειάζεται μια φάση μετά την επεξεργασία που θα είναι εξολοκλήρου αφιερωμένη στην βελτίωση των πολλαπλών πλαισίων οριοθέτησης, την

εξάλειψη των διπλότυπων ανιχνεύσεων και στην ανάκτηση των ορίων πλαισίων η οποία πρέπει να γίνεται βάση τον αριθμό των αντικειμένων της σκηνής [21].



Σχήμα 1.2: Αρχιτεκτονική του YOLO [48]

To 2016 ο Joseph Redmon παρουσίασε το σύστημα You Look Only Once (YOLO) [48], που είναι μια νέα προσέγγιση στην ανίχνευση αντικειμένων. Ο YOLO είναι μια ενοποιημένη δομή που πλαισιώνει την ανίχνευση αντικειμένων ως πρόβλημα παλινδρόμησης χωρίς την χρήση ενός πολύπλοκου αγωγού που δίνει την δυνατότητα στο σύστημα να είναι εξαιρετικά γρήγορο. Εκτελεί εικόνες σε πραγματικό χρόνο με 45 frames ανά δευτερόλεπτο και επεξεργάζεται μέχρι ροή βίντεο επίσης σε πραγματικό χρόνο με λιγότερο από 25 χιλιοστά του δευτερολέπτου καθυστέρησης.

Αντί να χρησιμοποιεί ο YOLO ένα συρόμενο παράθυρο σε πολλές τοποθεσίες, ως κορυφαία μέθοδος ανίχνευσης όπως και ο R-CNN, βλέπει ολόκληρη την εικόνα κατά τη διάρκεια της εκπαίδευσης και της φάσης δοκιμών, κωδικοποιώντας σιωπηρά πληροφορίες σχετικά με τις τάξεις τους, καθώς και τις εμφανίσεις τους, αποφεύγοντας να κάνει κάποια λάθη ανίχνευσης αντικειμένων στο παρασκήνιο των εικόνων. Για την εκτέλεση της διαδικασίας ανίχνευσης, το σύστημα διαιρεί την εικόνα εισόδου σε ένα πλέγμα  $S \times S$  και ένα ενιαίο συνελικτικό νευρωνικό δίκτυο (CNN) προβλέπει ταυτόχρονα για κάθε κελί πλέγματος πολλαπλά οριακά πλαίσια, την τιμή εμπιστοσύνης τους και τις πιθανότητες των κλάσεων τους για κάθε περιοχή. Τα κουτιά ορίου είναι σταθμισμένα από τις προβλεπόμενες πιθανότητες και εφαρμόζεται η μη μέγιστη τεχνική καταστολής για να διωρυθούν οι πολλαπλές ανιχνεύσεις όταν εντοπίζονται αντικείμενα κοντά στο όριο πολλών κελιών.

Εξετάζοντας τον σχεδιασμό του δικτύου YOLO, η αρχιτεκτονική του είναι εμπνευσμένη από το μοντέλο GoogLeNet για ταξινόμηση εικόνας [62] αλλά αντί για 22 στρώσεις βαθύ CNN, έχει 24 ακολουθούμενες από 2 πλήρως συνδεδεμένα στρώματα. Για να μειωθεί ο χώρος των χαρακτηριστικών από τα προηγούμενα στρώματα,  $1 \times 1$  συνελικτικές στρώσεις εναλλάσσονται μεταξύ των στρωμάτων όπως φαίνεται στο σχήμα 1.2. Τα συνελικτικά στρώματα είναι προπονημένα στο ImageNet [18] τα οποία εκτελούν την εργασία ταξινόμησης στη μισή ανάλυση

(είσοδος εικόνας 224 x 224) και στη συνέχεια διπλασιάζεται για την τελική ανίχνευση. Ενώ ανιχνεύει μικρά αντικείμενα που εμφανίζονται σε ομάδες, ο YOLO εξακολουθεί να υστερεί σε σχέση με συστήματα ανίχνευσης τελευταίας τεχνολογίας, αλλά οι φωτογραφίες που θα χρησιμοποιηθούν, με σκοπό την υλοποίηση του συστήματος, είναι αρκετά μεγάλες ώστε να αποφευχθεί κάποιο θέμα στην διαδικασία εντόπισης των αντικειμένων που θα ανιχνεύονται. Οπότε η μη τελειότητα του YOLO δεν πρέπει να θεωρείται ένας τεράστιος περιορισμός για να ληφθεί υπόψη.

## 1.2 Κοινωνικός Αντίκτυπος

Είναι αυτονόητο ότι ως επί το πλείστον η ενσωμάτωση των αυτόνομων οχημάτων στη κοινωνία θα αποφέρει θετικά αποτελέσματα. Όμως, μια αντικειμενική έρευνα πρέπει να εστιάσει και στην άλλη όψη του νομίσματος. Τίποτα δεν έχει μόνο θετική πλευρά και αυτό πρέπει να ληφθεί υπόψιν πριν από οποιοδήποτε συμπέρασμα πόσο μάλλον όταν το ζήτημα έχει σχέση με μια τόσο μεγάλη κοινωνική αλλαγή που θα επηρεάσει παγκοσμίως όλη την ανθρωπότητα.

### 1.2.1 Θετικός Αντίκτυπος

Σε ατομικό επίπεδο, η πλήρως αυτοματοποιημένη οδήγηση μπορεί να χαμηλώσει το άγχος που σχετίζεται με τους μετακινούμενους επειδή ο χρόνος στο όχημα μπορεί να χρησιμοποιηθεί όπως επιθυμούν εκείνοι όπως για φαγητό, ύπνο, παρακολούθηση τηλεόρασης ή μέχρι και για να εργάζονται [5] [27] [20]. Ένα άλλο πιθανό ατομικό όφελος σχετίζεται με τη βελτίωση της πρόσβασης στην κινητικότητα, ειδικά για ηλικιωμένους και άτομα με σωματική αναπηρία, διασφαλίζοντας την ασφαλή συμμετοχή αυτών των πληθυσμών στην κυκλοφορία. Επιπλέον, η αυτόνομη οδήγηση μπορεί να μειώσει τα οικονομικά βάρη ενός σοφέρ και σε ορισμένες περιπτώσεις να αυξάνει την πρόσβαση στην εκπαίδευση και στις επαγγελματικές ευκαιρίες [4] [27] [41].

Σε κοινωνικό επίπεδο, τα πλήρως αυτοματοποιημένα οχήματα έχουν τη δυνατότητα να ενισχύσουν την ασφάλεια των οχημάτων, εξαλείφοντας πρακτικά τα ανθρωπογενή σφάλματα και λάθη που επηρεάζουν την απόδοση της οδήγησης και προκύπτουν από φαινόμενα όπως η γήρανση, οι ασθένειες, το άγχος, η κόπωση, η απειρία ή ακόμα και η κατάχρηση παράνομων ουσιών [9] [52]. Έχει εκτιμηθεί ότι τα αυτοκίνητα χωρίς οδηγό μπορούν να μειώσουν τα θύματα τροχαίων δυστυχημάτων έως και 90% μέχρι τα μέσα του αιώνα [10]. Επιπλέον, θα υπάρχει και οικολογικό όφελος, που απορρέει από μειώσεις στις εκπομπές διοξειδίου του άνθρακα και στην κατανάλωση καυσίμου [22]. Η αυτόνομη οδήγηση αναμένεται επίσης να βελτιστοποιήσει και την χρήση της γης, επειδή τα οχήματα θα μπορούν να αποβιβάσουν επιβάτες σε πυκνές μητροπολιτικές περιοχές πριν οδηγηθούν σε κοντινό χώρο στάθμευσης [4].

### 1.2.2 Αρνητικός Αντίκτυπος

Δυστυχώς, δεν μπορεί να αποκλειστεί η πιθανότητα ότι λόγω τεχνολογικών ελαττωμάτων ή καταστάσεων για τις οποίες δεν είναι προετοιμασμένη η τεχνολογία

που είναι ενσωματωμένη στο όχημα, ενδέχεται να συμβούν ατυχήματα που αφοράνε συγκεκριμένα την αυτόνομη οδήγηση και θα ήταν απίθανο να συμβούν με άνθρωπο οδηγό. Ακόμη και αν το αποτέλεσμα ενός ατυχήματος ήταν μόνο μικρές βλάβες, θα είχε σαφείς νομικές και οικονομικές επιπτώσεις που θα έπρεπε να αντιμετωπιστούν. Αυτά θα περιπλέκονταν ανάλογα το πόσο μεγάλη θα ήταν η ζημιά από το ατύχημα όπως η περίπτωση σοβαρού ατυχήματος με αυτόματο πιλότο αυτοκινητόδρομου [67].

Αυτόνομα οχήματα που βασίζονται στη σύνδεση διαδικτύου, θα δημιουργήσουν νέους κίνδυνους [45]. Μέχρι σήμερα τα οχήματα λειτουργούν ανεξάρτητα το ένα από το άλλο. Όμως, με την αυτοματοποίηση τους, η αυτόνομα καθοδηγούμενη κυκλοφορία θα συνδέεται σε κάποιο βαθμό μέσω κέντρων ελέγχου και δικτύωσης για ανάγκες όπως να κατευθύνει την κυκλοφορία βέλτιστα χρησιμοποιώντας διασυνδεδεμένους αυτόματους πιλότους [67]. Ο έλεγχος μεγάλου αριθμού οχημάτων κατά πάσα πιθανότητα θα διεξάγεται μέσω λογισμικού που είναι πανομοιότυπο στη θεμελιώδη δομή του. Αυτό θα μπορούσε, να οδηγήσει σε ταυτόχρονη βλάβη ή δυσλειτουργία μεγάλου αριθμού οχημάτων με βάση το ίδιο πρόβλημα λογισμικού. Αυτά τα προβλήματα δεν θα είναι διαχειρίσιμα από άποψη γεωγραφίας, διάρκειας και κλίμακας της ζημίας.

Ακόμη και μετά από ένα επιτυχημένο λανσάρισμα στην αγορά, μπορεί να προκύψουν ατυχίες ή ακόμα και δυσλειτουργίες του συστήματος που θα μπορούσαν να αποτελέσουν δυσφήμηση για τις επηρεαζόμενες μάρκες αυτοκινήτων [66] [72]. Το σύνθετο λογισμικό είναι αδύνατο να δοκιμαστεί στο σύνολό του, πράγμα που σημαίνει ότι στην πραγματική χρήση μπορεί να προκύψουν απροσδόκητα προβλήματα [66]. Υπό αυτή την έννοια, η χρήση είναι ουσιαστικά μια άλλη δοκιμαστική φάση και οι χρήστες είναι οι δοκιμαστές. Το κατά πόσον οι οδηγοί θα αποδεχτούν να χρησιμοποιηθούν ως «υποκείμενα δοκιμής» μένει να φανεί. Οποιαδήποτε τέτοια προθυμία θα ήταν πιθανώς ανύπαρκτη σε περιπτώσεις που σχετίζονται με την ασφάλεια.

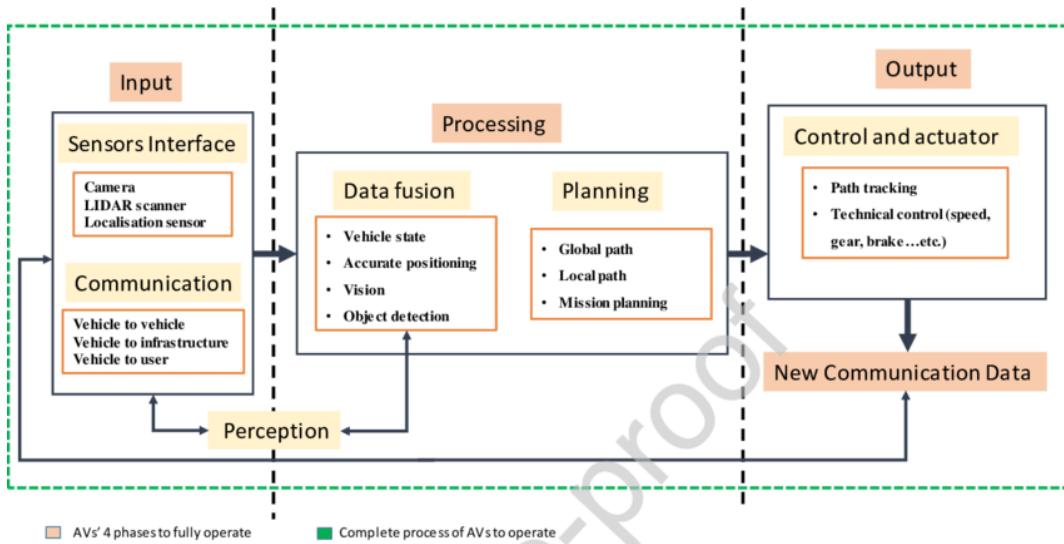
### 1.3 Αρχιτεκτονική Υλοποίησης Αυτόνομου Οχήματος

Τα αυτόνομα οχήματα συγκαταλέγονται στην γενικότερη κατηγορία των συστημάτων που έχουν σχέση με την αυτονομία των λογισμικών. Η αρχιτεκτονική υλοποίησης των συστημάτων αυτών ποικίλει ανάλογα με την εκάστοτε εφαρμογή που υλοποιείται και ανάλογα το τι στοχεύει, ενώ τα υλικά/χαρακτηριστικά της συσκευής του αυτόνομου οχήματος επηρεάζουν καθοριστικά το πως διαμορφώνεται το λογισμικό που θα είναι υπεύθυνο για τον έλεγχο του. Η φύση των αυτοκινούμενων οχημάτων τους δίνουν τη δυνατότητα να κινούνται με απόλυτη ελευθερία χωρίς όρια σε ένα άγνωστο περιβάλλον και χώρο. Όμως, η διαχείριση των απλών χαρακτηριστικών κρίνεται πιο απαιτητική σε σχέση με αυτή του γενικού αυτόνομου οχήματος, διότι η ακτίνα δράσης που έχουν οριοθετείται συνήθως σε ένα πολύ συγκεκριμένο περιβάλλον και απόλυτα γνώριμο από την αρχή. Στην συγκεκριμένη πτυχιακή εργασία δεν θα μας απασχολήσει αυτό κανώς ασχολούμαστε μόνο με την δημιουργία λογισμικού για την αυτόνομη οδήγηση ενός εικονικού οχήματος και δεν δημιουργούμε μια ρομπότ-συσκευή.

Της πάροχουν κάποιοι κανόνες και βασικές αρχές, που διέπουν όλα αυτά τα συστήματα, χωρίς αυτό να συνεπάγει ότι δεν έχει την δυνατότητα ο εκάστοτε σχεδιαστής να προσαρμόσει την αρχιτεκτονική του συστήματος που κατασκευάζει ανάλογα με τις δικές του ανάγκες. Ένα σύστημα αυτόνομου οχήματος απαρτίζεται από τρία βασικά στάδια που γίνονται ευδιάλογα στην εικόνα 1.3:

- Καταγραφή Δεδομένων Περιβάλλοντος (Input)
- Επεξεργασία Των Δεδομένων (Processing)
- Εξαγωγή Αποτελεσμάτων (Output)

Λόγω της πιθανής ασάφειας του σχήματος της εικόνας, θα αναλυθεί παρακάτω η βασική λειτουργία και σκοπιμότητα του κάθε σταδίου μέσα σε ένα σύστημα με χαρακτηριστικά αυτονομίας.



Σχήμα 1.3: Συνδεδεμένη Αρχιτεκτονική Συστήματος Αυτόνομου Οχήματος [11]

### 1.3.1 Input

Αρχικά, ένα αυτόνομο όχημα πρέπει να αντιλαμβάνεται τον κόσμο στον οποίο δρα και να τον απλοποιεί στο μέγιστο, αγνοώντας άχρηστες λεπτομέρειες που δεν ωφελούν κάπου. Η πρόσβαση στην πληροφορία του περιβάλλοντος του εικονικού κόσμου από ένα εικονικό αυτόνομο όχημα μοιάζει αρκετά με την διαδικασία που ακολουθεί ένα ρομπότ για να λάβει πληροφορίες από τον πραγματικό κόσμο. Και οι 2 διαδικασίες γίνονται μέσω αισθητήρων. Κάθε εικονικό αυτόνομο όχημα είναι εξοπλισμένο με διαφορετικό σύνολο αισθητήρων, ανάλογα με την εφαρμογή που στοχεύεται να υλοποιηθεί. Για παράδειγμα, ένα αυτόνομο όχημα χρειάζεται μία κάμερα για να βλέπει μπροστά και μία για πίσω.

Γενικά οι πληροφορίες που μπορεί να λάβει το σύστημα έρχονται συνήθως από αισθητήρες οι οποίοι ποικίλουν ανάλογα τις εκάστοτε ανάγκες. Οι πιο κλασσικοί αισθητήρες είναι μια κάμερα RGB, ένα Lidar Scanner και ένας τοπικοποιητής [59].

Δεν υπάρχει όριο στο ποιοι αισθητήρες θα στέλνουν δεδομένα και πόσοι από αυτοί θα είναι συνδεδεμένοι στο αμάξι. Επίσης, δεν υπάρχει κανένας περιορισμός και στο που θα τοποθετηθούν. Όλα εξαρτώνται από την χρίση του ειδικού και την φύση του προβλήματος που θέλει να λύσει. Επιπρόσθετα, κάποιες φορές έρχονται πληροφορίες και από τους *actors* του περιβάλλοντος όπως είναι άλλα οχήματα, πεζοί και οι υποδομές που υπάρχουν.

### 1.3.2 Processing

Το στάδιο αυτό έχει να κάνει με 2 σκέλη. Πρώτον, το σκέλος της αντίληψης τους περιβάλλοντος που έχει το όχημα, το οποίο αφορά τον τρόπο που αντιλαμβάνεται τα δεδομένα από αυτό ώστε να είναι έτοιμα για επεξεργασία. Το δεύτερο σκέλος σχετίζεται με την σχεδίαση που αφορά τις δυνατότητες που έχει το αυτόνομο όχημα να κάνει μέσα στο περιβάλλον ώστε να χριθεί η κατάλληλη δράση μέσα από κατάλληλη επεξεργασία των δεδομένων.

Τα δεδομένα που μπορεί να εκλάβει το όχημα μπορούν να προκύψουν και από κάποια αντίληψη πέρα από αισθητήρες που καταγράφουν εικονικά στοιχεία. Για παράδειγμα, τέτοια δεδομένα είναι η καταστάσεις του αμαξιού δηλαδή αν βρίσκεται σε κίνηση ή είναι σταματημένο, αν η θέση του στον χώρο συνάδει με τους κανόνες οδικής κυκλοφορίας και το αν έχει εντοπίσει κάποιο αντικείμενο στον χώρο μέσω ενσωματωμένου λογισμικού. Επίσης, στο κομμάτι της αντίληψης συγκαταλέγεται και το να συλλέγει πληροφορίες που σχετίζονται με τους *actors*. Τέτοιες πληροφορίες θα μπορούσαν να είναι καταστάσεις άλλων αμαξιών ή αν κάποιος πεζός τραυματίστηκε από κάποιο άλλο όχημα. Το πλεονέκτημα που έχει η εικονική πραγματικότητα έναντι του πραγματικού κόσμου είναι ότι αυτά τα δεδομένα μπορούν να παρθούν χωρίς να χρειαστεί να υπάρξει ούτε σωματική ούτε υλική ζημία.

Τέλος, βασικότερο κομμάτι της επεξεργασίας των παραπάνω δεδομένων αποτελεί το υποσύστημα του σχεδιασμού. Σε αυτό το υποσύστημα αναλαμβάνεται η χάραξη πορείας του οχήματος στον δρόμο και γενικότερα στον χώρο. Εκτός από μια επιμυητή πορεία μπορεί να υπάρχει και κάτι πιο συγκεκριμένο όπως το όχημα να φτάσει σε ένα συγκεκριμένο σημείο. Όλα αυτά τα αναλαμβάνει το κόμματι του σχεδιασμού το οποίο μέσα από τους στόχους που έχει θέσει ο ειδικός που ασχολείται με την υλοποίηση του λογισμικού, ενώνει τα δεδομένα για να χριθεί μέσα από αυτά ποια θα είναι στο τέλος η τελική απόφαση κίνησης του αυτόνομου οχήματος.

### 1.3.3 Output

Το στάδιο της εξαγωγής αποτελεσμάτων αφορά τον έλεγχο του οχήματος. Είναι το τελικό επίπεδο του λογισμικού και υπάρχει απαραίτητα σε κάθε σύστημα με χαρακτηριστικά αυτονομίας, καθώς είναι υπεύθυνο για την τελική δράση που θα παρθεί από το συνολικό σύστημα που έχει δημιουργηθεί [19]. Συνήθως μέσα από προσεγγίσεις Machine Learning [37] ή πιο καυθορισμένους Agents [36] παράγεται ένα αποτέλεσμα, αφού έχουν σταλθεί τα δεδομένα που έχουν συλλεχθεί και τροποποιηθεί από τα δύο παραπάνω στάδια. Αυτό το αποτέλεσμα είναι η τελική απόφαση για το πως το αμάξι θα φερθεί στον χώρο. Μπορεί να είναι είτε μια ιδανική

και βέλτιστη πορεία που μπορεί να ακολουθηθεί από το όχημα ή ένα αριθμητικό αποτέλεσμα που σχετίζεται με μια από τις βασικές μεταβλητές ενός οχήματος όπως την ταχύτητα, το φρένο και το γκάζι. Αξίζει να αναφερθεί ότι αυτό το τελικό αποτέλεσμα μπορεί να πάρει και τον ρόλο του δεδομένου (*input*) στην μετέπειτα πορεία ώστε να ανατροφοδοτήσει με την σειρά του τον επαναλαμβανόμενο κύκλο των σταδίων της αρχιτεκτονικής αυτόνομου οχήματος.

## 1.4 Σκοπός και Συνεισφορά της Πτυχιακής Εργασίας

Το κίνητρο που οδηγεί την εκπόνηση αυτής της πτυχιακής εργασίας είναι η κατασκευή και εγκατάσταση ενός συστήματος, χρησιμοποιώντας το μοντέλο YOLO, το οποίο είναι ικανό να ανιχνεύει και να αναγνωρίζει σηματοφόρους με τα χρώματα τους και κοντινά αυτοκίνητα, παίρνοντας δεδομένα από τον εικονικό κόσμο του CARLA. Αυτό το σύστημα πρέπει να είναι λειτουργικό και αποδοτικό τόσο στον επεξεργαστή όσο και στη κάρτα γραφικών και ικανό να λειτουργεί σε πραγματικό χρόνο.

Επιπλέον, θα αναπτυχθούν τέσσερις εφαρμογές για τις ανάγκες αυτής της εργασίας. Η πρώτη θα είναι μια προειδοποιητική εφαρμογή, για να ειδοποιεί τον χρήστη ότι υπάρχει σηματοδότης, αναγνωρίζοντας το χρώμα του καθώς θα προειδοποιεί και για την ύπαρξη κοντινού αμάξιού στον δρόμο. Η δεύτερη θα είναι μια εφαρμογή ελέγχου, που θα ελέγχει την ορθή διαδικασία του φρεναρίσματος αν αντικριστεί κόκκινος και κίτρινος σηματοδότης ή υπερβολικά κοντινό αμάξι. Δηλαδή, σε αυτές τις συνθήκες θα μειώνεται στο 0% η ταχύτητα του αμάξιού και δεν θα μπορεί να κινηθεί καθόλου εκτός από την περίπτωση που έχει εντοπιστεί κοντινό αμάξι μπροστά. Τότε, θα μπορεί το αμάξι μόνο να οπισθοχωρήσει για να αποφύγει την δυνητική σύγκρουση αλλά προφανώς δεν θα μπορεί να προχωρήσει μπροστά. Η τρίτη εφαρμογή θα ασχολείται με την εμφάνιση στον χώρο του πλαισίου του αντικειμένου που εντοπίζεται από τον YOLO ώστε να υπάρχει και οπτική σιγουριά και ακρίβεια στο ότι το δίκτυο εκπαιδεύτηκε σωστά και αναγνωρίζει και το τι εντοπίζει και το που το εντοπίζει. Τέλος, η τελευταία εφαρμογή αφορά καταμέτρηση αριθμητικών συχνοτήτων στις εμφανίσεις των αντικειμένων που μας ενδιαφέρουν όπως το πόσα κόκκινα φανάρια υπάρχουν στο σύνολο εκπαίδευσης. Επίσης, θα εξάγει στατιστικά δεδομένα όπως το ποσοστό ορθής πρόβλεψης του YOLO σε ένα σύνολο ελέγχου τα οποία θα παίζουν καθοριστικό ρόλο στο ποιος είναι ο καλύτερος τρόπος να εκπαιδευτεί το δίκτυο για να έχει τα καλύτερα δυνατά αποτελέσματα. Περισσότερες λεπτομέρειες θα αναφερθούν στο κεφάλαιο 4.

Επίσης, θα δημιουργηθεί ένα μεγάλο σύνολο δεδομένων εικόνων μέσα από τον εικονικό κόσμο του CARLA για να συνεισφέρει στη μεγάλη κοινότητα ερευνητών και χρηστών που αναπτύσσουν συνεχώς τον συγκεκριμένο προσομοιωτή οδήγησης. Μαζί με τις φωτογραφίες θα ακολουθούν και άλλα αρχεία τα οποία θα έχουν αποθηκευμένες πληροφορίες που αφοράν το σημείο εντόπισης ενός ή και παραπάνω αντικειμένων με σκοπό να γίνει η εκπαίδευση του δικτύου. Αυτές οι πληροφορίες θα χρησιμεύσουν ώστε ο ημιαυτόματος οδηγός που θα υπάρξει να μπορέσει να γίνει ακόμα καλύτερος από τους προγραμματιστές και συντηρητές του CARLA με ακόμα μεγαλύτερη ακρίβεια στο τρόπο συμπεριφοράς του. Επιπρόσθετα, για να μπορεί ανα πάσα στιγμή να τροποποιηθεί το λογισμικό από τρίτους ειδικούς, με σκόπο πάντοτε την βελτίωση του, όλο το λογισμικό που θα αναπτυχθεί θα είναι

εύκολα εκτελέσιμο, δωρεάν διαθέσιμο και πλήρως τεκμηριωμένο σε κάθε πτυχή της χρήσης του.

## Κεφάλαιο 2

# Μηχανή Προσομοίωσης **Carla**

### 2.1 Γενικές Πληροφορίες

Όσον αφορά τον προσομοιωτή, σημαντικές τεχνολογικές επαιρείες όπως η Microsoft & Toyota έχουν αναπτύξει το δικό τους υπερσύγχρονο και ανοιχτού κώδικα συστήματα προσομοίωσης αυτόνομης οδήγησης. Στο πιο μακρινό παρελθόν, ανήκει ο προσομοιωτής αγωνιστικών αυτοκινήτων TORCS [7] ο οποίος χυλοφόρησε το 1997 και ενέπνευσε τους ερευνητές να κάνουν αρκετές βελτιώσεις στο εικονικό πεδίο αυτόνομης οδήγησης γνωρίζοντας ότι οι δοκιμές αλγόριθμων για τα αυτόνομα οχήματα στον πραγματικό κόσμο είναι μια δαπανηρή και χρονοβόρα διαδικασία. Αφενός, η Microsoft ξεκίνησε το 2017 την οπτική και φυσική υψηλής πιστότητας προσομοίωση για αυτόνομα οχήματα AirSim [51]. Αυτός ο προσομοιωτής βασίζεται στη παιχνιδομηχανή «Unreal Engine» [65] η οποία είναι μια πλατφόρμα για τους ερευνητές να δοκιμάσουν με βαθιά και ενισχυτική μάθηση, αλγόριθμους που μπορούν να εφαρμοστούν σε αυτόνομα οχήματα. Από την άλλη πλευρά, το ερευνητικό ίνστιτούτο του Κέντρου Υπολογιστών της Toyota χρηματοδότησε τον προσομοιωτή CARLA [14], ο οποίος είναι προσομοιωτής ανοιχτού κώδικα για αυτόνομη οδήγηση βασισμένος επίσης στη παιχνιδομηχανή Unreal Engine. Και οι δύο προσομοιωτές έχουν αρκετές κοινές τομές: έχουν ευέλικτο API όπου οι χρήστες μπορούν να ελέγχουν όλες τις πτυχές που σχετίζονται με την προσομοίωση, παρέχουν και οι δύο ρεαλιστικές αστικές διατάξεις, ένα ευρύ φάσμα καιρικών συνθηκών και επίσης οι χρήστες μπορούν να ορίσουν πολυποίκιλα είδη αισθητήρων και να τους προσθέσουν στα εικονικά οχήματα κατορθώνοντας έτσι να πάρουν κάποιες εισόδους από το περιβάλλον του προσομοιωτή.

Η κύρια διαφορά μεταξύ αυτών των δύο προηγμένων προσομοιωτών είναι ότι το CARLA ασχολείται με μια μεγάλη κοινότητα χρηστών και ερευνητών για να αλληλεπιδράσει. Τα εργαλεία επικοινωνίας είναι:

1. Ένας ιστότοπος Git Hub
2. Ένας λογαριασμός στο Twitter
3. Ένα κανάλι Discord
4. Ένα κανάλι Youtube

Επίσης, η εξέλιξή του CARLA η οποία πραγματοποιείται σε σημαντικό πανεπιστήμιο επιτρέπει να είναι πολύ γνωστό σε όλη την κοινότητα που ασχολείται με την αυτόνομη οδήγηση. Το CARLA παρέχει πολύ ρεαλιστικές αστικές διατάξεις

(συμπεριλαμβανομένων κτιρίων, οχημάτων, πεζών) και ένα ευρύ φάσμα περιβαλλοντικών συνθηκών, που αναδεικνύει στο έπακρο την ομοιότητα του ως προσομοιωτή και εικονικό περιβάλλον, με τον πραγματικό κόσμο.

Καθώς το CARLA ασχολείται με μια μεγάλη κοινότητα χρηστών και ερευνητών, είναι συνεχώς σε εξέλιξη. Μόνο από το 2017 υπήρχαν ήδη 34 εκδόσεις που κυκλοφόρησαν. Δυστυχώς, η προγραμματισμένη και σταθερή έκδοση του CARLA για Windows (CARLA 0.8.2) δεν είναι η τελευταία έκδοση που έχει αναπτυχθεί στον έτος 2022. Όμως, για τους σκοπούς αυτής της πτυχιακής εργασίας, συγκεντρώνει τα απαιτούμενα χαρακτηριστικά που χρειαζόμαστε για την γραφική προσομοίωση και για το κομμάτι του πηγαίου κώδικα. Η έκδοση του CARLA 0.8.2 έχει δύο διαθέσιμα σενάρια: Town01 και Town02, και πολλές άλλες παραμέτρους που είναι εφικτό να οριστούν και να τροποποιηθούν, όπως οι καιρικές συνθήκες, ο αριθμός οχημάτων ή πεζών, μεταξύ άλλων.

Ο προσομοιωτής **Carla**, που είναι χτισμένος πάνω σε γλώσσα υψηλού επιπέδου την C++ [46], μπορεί να ελεγχθεί με εξωτερικό **script** που μπορεί να τροποποιήσει τις περισσότερες πτυχές της προσομοίωσης ανάλογα με το τι επιθυμεί ο εκάστοτε προγραμματιστής. Το Python API που παρέχεται από τους ερευνητές περιέχει μια γραφική διεπαφή χρήστη (GUI) η οποία εφαρμόζεται με το PyGame [40] όπου ο χρήστης μπορεί να ελέγχει το όχημά του με το χέρι δηλαδή με χρήση του πληκτρολογίου και του ποντικιού. Επιπλέον, ορισμένοι αισθητήρες όπως «συστήματα ανίχνευσης και εμβέλειας απεικόνισης φωτωγραφιών» (LIDARs), κάμερες RGB ή Depth Map [8] μπορούν να προστεθούν στο όχημα για να αποκτήσουν πληροφορίες για το περιβάλλον. Η ανάκτηση των μετρήσεων, ώστε να είναι εφικτό το όχημα να λάβει μερικές αποφάσεις ταχεώς, συμβαίνει κατά τη διαδικασία οδήγησης σε πραγματικό χρόνο.

## 2.2 Επικοινωνία **Carla** με τον Υπολογιστή

Οι **clients** είναι ένα από τα κύρια στοιχεία της αρχιτεκτονικής του CARLA. Συνδέονται με τον **server**, ανακτούν πληροφορίες και εντολές για αλλαγές. Αυτό γίνεται μέσω σεναρίων. Ο **client** αναγνωρίζει τον εαυτό του και συνδέεται με τον κόσμο για να λειτουργήσει με την γραφική προσομοίωση του εικονικού κόσμου. Επιπλέον, οι **clients** μπορούν να έχουν πρόσβαση σε προηγμένες μονάδες του CARLA, έχοντας την δυνατότητα να εκτελούν παρτίδες εντολών. Αυτό χρησιμεύει για βασικά πράγματα, όπως η δημιουργία πολλών **actors**.

Για την δημιουργία ενός **client** δύο πράγματα χρειάζονται. Η διεύθυνση IP που την αναγνωρίζει και δύο θύρες TCP για επικοινωνία με τον **server**. Μια προαιρετική τρίτη παράμετρος ορίζει την ποσότητα των νημάτων εργασίας [1]. Από προεπιλογή, το CARLA χρησιμοποιεί τοπική IP κεντρικού υπολογιστή και την θύρα 2000 για σύνδεση, αλλά αυτά μπορούν να αλλάξουν κατά βούληση. Η δεύτερη θύρα θα είναι πάντα η  $n+1$ , 2001 σε αυτήν την περίπτωση [1]. Μόλις δημιουργηθεί ο **client**, πρέπει να οριστεί το χρονικό όριο για την σύνδεση του με τον **server**. Αυτό περιορίζει όλες τις λειτουργίες δικτύωσης, ώστε να μην μπλοκάρουν τον **client** για πάντα. Εάν η σύνδεση αποτύχει, θα εμφανιστεί μήνυμα σφάλματος. Είναι εφικτό να υπάρχουν πολλοί **clients** συνδεδεμένοι στον **server**, καθώς είναι σύνηθες να τρέχουν παραπάνω από ένα σενάριο ταυτόχρονα. Η εργασία σε ένα σύστημα πολλαπλών **clients** με προηγμένες δυνατότητες που παρέχει το CARLA, όπως ο

διαχειριστής κίνησης, είναι βέβαιο ότι θα κάνει την επικοινωνία Carla-Υπολογιστή πιο περίπλοκη [1].

Ένας client μπορεί να συνδεθεί και να ανακτήσει τον τρέχοντα κόσμο αρκετά εύκολα. Μπορεί επίσης να λάβει μια λίστα με τους διαθέσιμους χάρτες για να αλλάξει τον τρέχοντα. Αυτό θα καταστρέψει τον εικονικό κόσμο της δεδομένης χρονικής στιγμής και θα δημιουργήσει έναν νέο [1]. Κάθε αντικείμενο μέσα στον χώρο έχει ένα αναγνωριστικό ID ή ένα σεναριακό αριθμό επεισοδίου που το προσδιορίζει. Κάθε φορά που ο client καλεί συναρτήσεις για την φόρτιση ή την επαναφόρτιση του κόσμου ο προηγούμενος κόσμος είναι δεδομένο ότι καταστρέφεται. Ένας νέος δημιουργείται από την αρχή με ένα νέο επεισόδιο να λαμβάνει χώρα. Η Unreal Engine δεν επανεκκινείται στη διαδικασία αυτή [1].

Ο κόσμος ως object έχει πρόσβαση σε ορισμένες προηγμένες διαμορφώσεις που έχουν άμεση σχέση με την προσομοίωση. Αυτές καθορίζουν τις συνθήκες απόδοσης, τα χρονικά βήματα προσομοίωσης και το συγχρονισμό μεταξύ clients και local server.

Για να επιτευχθούν ομαλά όλα τα παραπάνω, έχοντας αποτελεσματικό συγχρονισμό του Carla με τον υπολογιστή, πρέπει να υπάρξει επιτυχής ενεργοποίηση και άνοιγμα ενός local server πριν από την λειτουργία οποιουδήποτε προγράμματος και script που υπάρχει μέσα στο Carla. Όπως αναφέρθηκε και προηγουμένως, από προεπιλογή ο server που αναμένουν να “ακούσουν” οι clients είναι τοπικός στη θύρα 2000. Αυτό βοηθάει διότι δεν χρειάζονται τροποποιήσεις στο πρόγραμμα που δημιουργεί τους clients για να κατορθώσουν συνδεθούν στον server.

Η δημιουργία ενός server στην έκδοση του CARLA 0.8.2 γίνεται μέσω cmd εντολής [29], αφού έχει προηγηθεί να έχει κατέβει η έκδοση αυτή στον υπολογιστή. Η συγκεκριμένη πτυχιακή εργασία πραγματοποιείται πάνω σε Windows οπότε η εντολή που αναγράφεται τρέχει μόνο σε αυτό το λογισμικό. Με δεδομένο ότι η ρίζα, στην οποία είναι τοποθετημένο το παράθυρο cmd, είναι το path όπου βρίσκεται η εφαρμογή του Carla, η εντολή που πρέπει να τρέξει είναι η εξής:

```
CarlaUE4.exe -carla-server -benchmark -fps=FPS -windowed -ResX=X -ResY=Y
```

Η εντολή -benchmark υποδεικνύει στο Carla ότι ο server που ενεργοποιείται αποτελεί το σημείο αναφοράς όλης της γραφικής προσομοίωσης και δεν επιτρέπεται σε κανέναν άλλον server να συνδεθεί ταυτόχρονα. Με την εντολή -fps εκχωρείται ο ρυθμός προβολής των εικόνων που επιτυμεί ο εκάστοτε προγραμματιστής να πραγματοποιείται στην οπτικοποίηση του γραφικού περιβάλλοντος από τον υπολογιστή του. Συνήθως, άμα ο εκάστοτε υπολογιστής δεν διαθέτει ισχυρή κάρτα γραφικών θα ήταν σοφό να μην είναι πολύ υψηλή η τιμή εκχώρησης που θα έχει. Η εντολή -windowed στην ουσία αποσκοπεί στο να μετατρέψει τον server σε παράθυρο προγράμματος με όλες τις ιδιότητες που έχει ένα παράθυρο. Τέλος, αν έχει εκτελεστεί προηγουμένως η εντολή -windowed τότε οι εντολές -ResX και -ResY ορίζουν το μέγεθος σε μήκος και σε πλάτος, που θα έχει το παράθυρο που θα εμφανιστεί στον υπολογιστή του χρήστη.

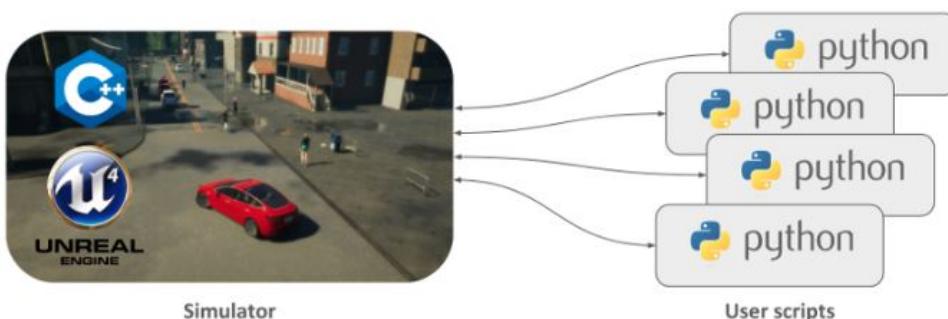
Συμπερασματικά, αφού έχουν λυθεί όλα τα τεχνικά ζητήματα που έχουν σχέση με την ενεργοποίηση του τοπικού server και την δημιουργία ενός ή παραπάνω clients στο κόσμο του Carla, τότε και μόνο τότε αξιοποιούνται όλοι οι πόροι του εκάστοτε υπολογιστή μέσα στο γραφικό περιβάλλον. Πραγματοποιείται στην

ουσία η επικοινωνία Carla-Τυπολογιστή, η οποία μελλοντικά θα καταστήσει εφικτό να υλοποιηθούν εφαρμογές που έχουν σχέση με την αυτόνομη οδήγηση.

### 2.3 Δυνατότητες

Το Carla κατασκευάστηκε από την αρχή για να χρησιμεύσει ως ευέλικτο API για την αντιμετώπιση μιας σειράς εργασιών που εμπλέκονται στο πρόβλημα της αυτόνομης οδήγησης. Ένας από τους κύριους στόχους του είναι να βοηθήσει στον εκδημοκρατισμό της έρευνας και ανάπτυξης της αυτόνομης οδήγησης, χρησιμεύοντας ως εργαλείο στο οποίο μπορούν να έχουν εύκολη πρόσβαση και προσαρμογή οι χρήστες. Για να γίνει αυτό, ο προσομοιωτής πρέπει να πληροί τις απαιτήσεις διαφορετικών περιπτώσεων στο γενικό πρόβλημα της οδήγησης όπως είναι η εκμάθηση πολιτικών οδήγησης και οι αλγόριθμοι αντίληψης εκπαίδευσης. Το Carla χρησιμοποιεί το πρότυπο OpenDRIVE [54] για να καθορίσει τους δρόμους και τις αστικές ρυθμίσεις. Ο έλεγχος της προσομοίωσης παρέχεται μέσω ενός API που χειρίζεται σε Python και C++, το οποίο βελτιώνεται συνεχώς. Προκειμένου να εξομαλύνει τη διαδικασία ανάπτυξης, εκπαίδευσης και επικύρωσης συστημάτων οδήγησης, το CARLA εξελίχθηκε σε ένα οικοσύστημα έργων, που χτίστηκε γύρω από την κύρια πλατφόρμα από την κοινότητα. Σε αυτό το πλαίσιο, είναι σημαντικό να γίνουν κατανοητά ορισμένα πράγματα για το πώς λειτουργεί ο προσομοιωτής αυτός ώστε να γίνουν αντιληπτές οι δυνατότητες του.

Ο προσομοιωτής Carla αποτελείται από μια κλιμακούμενη αρχιτεκτονική client-server. Ο server είναι υπεύθυνος για οτιδήποτε σχετίζεται με την ίδια την προσομοίωση: απόδοση αισθητήρα, υπολογισμός της φυσικής, ενημερώσεις για την κοσμική κατάσταση και τους συντελεστές της. Καθώς στοχεύει σε ρεαλιστικά αποτελέσματα, η καλύτερη εφαρμογή θα ήταν η λειτουργία του server με μια αποκλειστική κάρτα γραφικών, ειδικά όταν πρόκειται για μηχανική μάθηση [63]. Η πλευρά του client αποτελείται από ένα άθροισμα μονάδων που ελέγχουν τη λογική των actors στη σκηνή και θέτουν τις συνθήκες περιβάλλοντος. Αυτό επιτυγχάνεται με την αξιοποίηση των CARLA APIs (σχήμα 2.1) που έχει προαναφερθεί, ένα επίπεδο που μεσολαβεί μεταξύ client-server το οποίο εξελίσσεται συνεχώς για να παρέχει νέες λειτουργίες.



Σχήμα 2.1: Επικοινωνία Προσομοιωτή με APIs [63]

Το σχήμα 2.1 συνοψίζει τη βασική δομή του προσομοιωτή. Η κατανόηση του Carla όμως είναι κάτι πολύ περισσότερο από αυτό, καθώς πολλά διαφορετικά χαρακτηριστικά και στοιχεία συνυπάρχουν μέσα του. Μερικά από αυτά παρατίθενται

παρακάτω, για να γίνει αντιληπτή η προοπτική των δυνατοτήτων του τι μπορεί να επιτύχει το *Carla*. Μερικές από τις δυνατότητες που προσφέρονται από τον προσομοιωτή είναι οι εξής [63]:

- Διαχειριστής Κυκλοφορίας (Traffic manager)
- Αισθητήρες (Sensors)
- Μηχανή Εγγραφής (Recorder)
- ROS bridge and Autoware implementation
- Ευέλικτες Τροποποιήσεις Παραμέτρων (Open assets)
- Διαχείριση Εκτέλεσης Σεναρίου (Scenario runner)

Ο διαχειριστής κυκλοφορίας αποτελεί ένα ενσωματωμένο σύστημα που αναλαμβάνει τον έλεγχο των οχημάτων εκτός από το όχημα που χρησιμοποιείται για εκμάθηση. Λειτουργεί ως αγωγός που παρέχεται από το *Carla* για να αναδημιουργήσει περιβάλλοντα που μοιάζουν με αστικά κέντρα τα οποία έχουν ρεαλιστικές συμπεριφορές. Τα οχήματα βασίζονται στους αισθητήρες για τη διανομή πληροφοριών από το περιβάλλον τους. Στο CARLA οι αισθητήρες είναι ένα συγκεκριμένο είδος actor που συνδέεται με το όχημα και τα δεδομένα που λαμβάνουν μπορούν να ανακτηθούν και να αποθηκευτούν για να επιτευχθεί μελλοντικά η διαδικασία εκπαίδευσης.

Η δυνατότητα της εγγραφής χρησιμοποιείται για την αναπαράσταση μιας προσομοίωσης βήμα προς βήμα για κάθε ηθοποιό στον κόσμο. Παρέχει πρόσβαση σε οποιαδήποτε στιγμή στο χρονοδιάγραμμα οπουδήποτε στον κόσμο, καθιστώντας ένα εξαιρετικό εργαλείο ανίχνευσης. Ως θέμα καθολικότητας, το *Carla* εργάζεται για την ενσωμάτωση του προσομοιωτή και σε άλλα περιβάλλοντα εκμάθησης. Τα στοιχεία που αφοράνε τον έλεγχο των καιρικών συνθηκών και της διάταξης του χώρου μπορούν να προσαρμοστούν και να δημιουργηθούν νέα, ακολουθώντας απλές οδηγίες και εντολές από τον εκάστοτε προγραμματιστή.

Τέλος, προκειμένου να διευκολυνθεί η διαδικασία εκμάθησης για τα οχήματα, το *Carla* παρέχει μια σειρά από διαδρομές που περιγράφουν διαφορετικές καταστάσεις προς επανάληψη. Αυτά τα έθεσαν οι δημιουργοί ως πρόκληση του *Carla*, ανοιχτή σε όλους τους προγραμματιστές να δοκιμάσουν τις λύσεις τους ώστε να φτάσουν στο leaderboard της εφαρμογής.

## 2.4 Συνθήκες Περιβάλλοντος

Αρχικά πρέπει να αποσαφηνιστεί τι εννοούμε με τις συνθήκες περιβάλλοντος. Ένα όχημα εισέρχεται σε ένα κόσμο και πρέπει να οριστεί τι ακριβώς επηρεάζει την όλη πορεία του. Με δεδομένο ότι βρίσκεται σε έναν εικονικό κόσμο, πρέπει οτιδήποτε επηρεάζει ένα πραγματικό αμάξι να επηρεάζει και το εικονικό, για την επίτευξη αντικειμενικής ρεαλιστικής επίλυσης οποιουδήποτε προβλήματος αυτόματης οδήγησης. Οι συνθήκες περιβάλλοντος αναφέρονται σε 3 τομείς:

1. Τις καιρικές συνθήκες

2. Τις υλικές υποδομές
3. Την αλληλεπίδραση με *actors*

#### 2.4.1 Καιρικές Συνθήκες

Το *Carla* επιτρέπει στον χρήστη να επιλέξει ανάμεσα σε 15 προεπιλογές καιρικών συνθηκών [70]. Μέσα από αυτές είναι εφικτό το όχημα που περιπλανιέται στον χώρο, να δοκιμαστεί σε διάφορες καταστάσεις και να συγχριθεί η απόδοση του σε οποιαδήποτε καιρική συνθήκη βρεθεί. Αυτό βοηθάει κυρίως πάνω στην καθολικότητα οποιουδήποτε προβλήματος που αφορά αυτόνομη οδήγηση καθώς το αυτοκίνητο θέλουμε να αποδίδει υπό κάθε καιρική συγκυρία και όχι μόνο σε συγκεκριμένες. Για αυτό στο έργο που πραγματεύεται η συγκεκριμένη πτυχιακή εργασία, είναι σημαντικό να υπάρχει η δυνατότητα αλλαγής του καιρού. Ο τύπος του φωτός, η γωνία πρόσπτωσής του και η ποσότητα φωτός επηρεάζει τα αντικείμενα καθώς διαφέρουν οπτικά ανάλογα με τις καιρικές συνθήκες που επικρατούν. Σε ένα σύστημα αναγνώρισης αντικειμένων αυτό παίζει καθοριστικό παράγοντα καθώς υπάρχει κίνδυνος αν δεν ληφθεί υπόψιν, το δίκτυο που θα εκπαιδευτεί να αναγνωρίζει αντικείμενα μόνο σε συγκεκριμένες καιρικές συνθήκες και να σφάλει σε διαφορετικές.

#### 2.4.2 Υλικές Υποδομές

Με τις υλικές υποδομές με λίγα λόγια εννοείται ο χάρτης στον οποίο περιπλανιέται το αυτοκίνητο. Είναι από πριν προγραμματισμένος να εμπεριέχει την διαρρύθμιση του δρόμου, τα κτήρια και τα πεζοδρόμια. Ένας χάρτης περιλαμβάνει τόσο το τρισδιάστατο μοντέλο μιας πόλης όσο και τον ορισμό του δρόμου [38]. Ο ορισμός του δρόμου ενός χάρτη βασίζεται σε ένα αρχείο *OpenDRIVE*, μια τυποποιημένη, σχολιασμένη μορφή ορισμού δρόμου. Ο τρόπος με τον οποίο το πρότυπο *Open DRIVE* ορίζει δρόμους, λωρίδες και διασταυρώσεις καθορίζει τη λειτουργικότητα του *Python API* και το σκεπτικό πίσω από τις αποφάσεις που λαμβάνονται από ένα λογισμικό [38]. Το *Python API* λειτουργεί ως σύστημα αναζήτησης υψηλού επιπέδου για την πλοιόγηση σε αυτούς τους δρόμους. Κάθε κτήριο, παγκάκι, φωτιστικό δρόμου σε έναν χάρτη, έχει ένα σύνολο σχετικών μεταβλητών. Σε αυτές τις μεταβλητές περιλαμβάνεται ένα μοναδικό αναγνωριστικό που μπορεί να χρησιμοποιηθεί για την εναλλαγή της ορατότητας αυτού του αντικειμένου στον χάρτη [38]. Μπορεί να χρησιμοποιηθεί το *Python API* για να ανακτηθούν τα αναγνωριστικά κάθε αντικειμένου περιβάλλοντος με βάση τη σημασιολογική τους ετικέτα.

#### 2.4.3 Αλληλεπίδραση με **Actors**

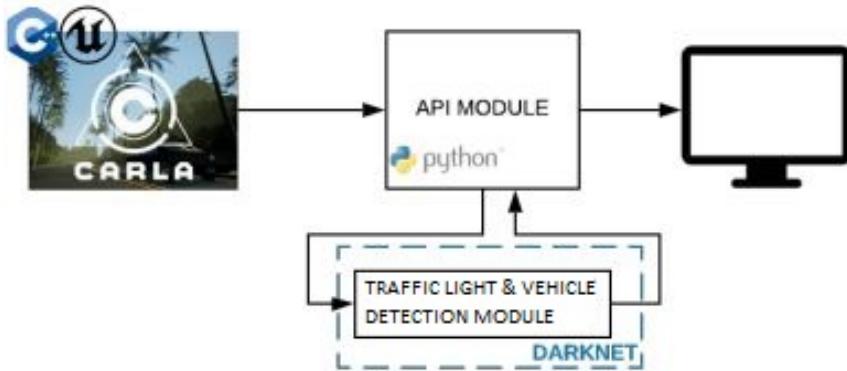
Οι *actors* στο *Carla* είναι τα στοιχεία που εκτελούν ενέργειες εντός της προσομοίωσης και μπορούν να επηρεάσουν και άλλους *actors*. Οι *actors* στο *Carla* περιλαμβάνουν οχήματα, πεζούς και επίσης αισθητήρες, σήματα κυκλοφορίας, φανάρια και τον ίδιο τον θεατή [2]. Είναι ζωτικής σημασίας να υπάρχει πλήρης κατανόηση για το πως όλοι οι *actors* αλληλεπιδρούν μεταξύ τους. Για παράδειγμα, η επιλογή ενσωμάτωσης αισθητήρα παίζει καθοριστικό ρόλο για το πως ένα όχημα

αντιλαμβάνεται το περιβόλλον καθώς στην πράξη αποτελεί τα “μάτια” του προς αυτό. Αξίζει να αναφερθεί ότι τα άλλα οχήματα, οι τυχαιοί περαστικοί και οι σηματοφόροι, προσδίδουν στην ουσία κατά την οδήγηση στον εικονικό χόσμο την ρεαλιστικότητα και την τυχαιότητα που μπορεί να υπάρξει και στον πραγματικό χόσμο.

## Κεφάλαιο 3

# Υλοποίηση Μοντέλου Αντίληψης

Σε αυτό το κεφάλαιο της πτυχιακής εργασίας, πρόκειται να εξηγηθεί διεκπεραιωτικά η διαδικασία για το πώς κατορθώθηκε να σχεδιαστεί και έπειτα να εκπαιδευτεί το μοντέλο ανίχνευσης αντικειμένων.



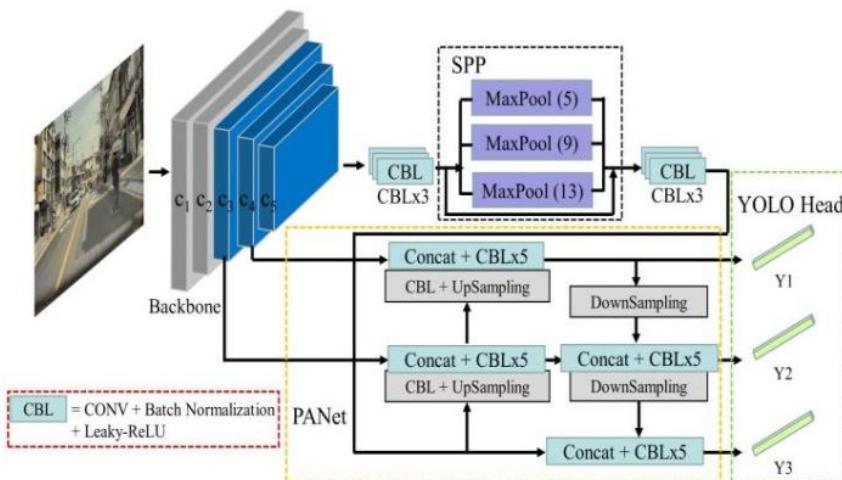
Σχήμα 3.1: Οπτική λειτουργία του συνολικού έργου

Το σχήμα 3.1 δείχνει τον “άγωγό” του έργου και τους κύριους παράγοντες που συμμετέχουν σε αυτό: το Darknet, την μονάδα ανίχνευσης σηματοφόρων και κοντινών οχημάτων (YOLO) και ο γραφικός προσομοιωτής CARLA. Το Darknet είναι ένα πλαίσιο νευρωνικών δικτύων ανοιχτού κώδικα όπου κατορθώνεται σε πραγματικό χρόνο να αναπτυχθεί αποτελεσματικά το σύστημα ανίχνευσης αντικειμένων YOLO [30]. Αυτό το σύστημα είναι εκπαιδευμένο να αναγνωρίζει τους σηματοφόρους του προσομοιωτή CARLA μαζί με το παρόν χρώμα τους και επιπρόσθετα να εντοπίζει και κοντινά οχήματα. Στη συνέχεια, το εκπαιδευμένο μοντέλο ενσωματώνεται στον προσομοιωτή. Μετά την τροποποίηση κάποιου από τον πηγαίο κώδικα του προσομοιωτή CARLA, το πλέον εκπαιδευμένο μοντέλο YOLO είναι σε θέση να ανιχνεύσει το χρώμα που θα έχει οποιοσδήποτε σηματοδότης ενός δρόμου καθώς και το αν υπάρχει κίνδυνος σύγκρουσης με κάποιο άλλο όχημα και να λάβει βάση αυτών τις ανάλογες αποφάσεις οι οποίες πρέπει να παρθούν. Επομένως, πρόκειται παρακάτω να εξηγηθούν αυτά τα κύρια μέρη του έργου που υλοποιούνται μέσα από αυτή την πτυχιακή εργασία και τις διαδικασίες που ακολουθήθηκαν σε κάθε βήμα εκτέλεσης.

### 3.1 Darknet και YOLO

Ο YOLO είναι ένα υπερσύγχρονο σύστημα ανίχνευσης αντικειμένων σε πραγματικό χρόνο που σε σύγκριση με άλλους παρόμοιους ανιχνευτές ή αλγόριθμους γενικότερα που χυλοφορούν, είναι εξαιρετικά γρήγορος, αποδοτικός και ακριβής. Το σύστημα του που χρησιμοποιείται στην συγχεκριμένη εργασία, είναι χτισμένο στο Darknet [30], ένα πλαίσιο νευρωνικού δικτύου ανοιχτού κώδικα που υποστηρίζει τόσο την κεντρική μονάδα διαδικασίας (CPU) όσο και τη μονάδα γραφικών διεργασιών (GPU). Το Darknet έχει προγραμματιστεί με κύρια βάση την γλώσσα C και το CUDA [44]. Προκειμένου να γίνει ένα σύστημα λειτουργικό στα περισσότερα περιβάλλοντα που υπάρχουν και με χαμηλό υπολογιστικό κόστος, το Darknet έχει εφαρμοστεί στην CPU και όχι στην GPU. Για αυτό, η διαδικασία εγκατάστασης του Darknet υπολογίστηκε ακολουθώντας τη τεκμηρίωση που δημοσιεύτηκε στον ιστότοπο του δημιουργού του [30]. Όπως λέει και ο ίδιος ο δημιουργός του Darknet: «Το Darknet στην CPU είναι γρήγορο αλλά είναι 500 φορές γρηγορότερα στη GPU.» [30]. Άρα με κέντρο άξονα την CPU θα πορευτεί η όλη υλοποίηση πάνω στην εκπαίδευση του YOLO από εδώ και πέρα.

Το επόμενο βήμα είναι να εκπαιδεύσουμε το δίκτυο να ανιχνεύει τα προσαρμοσμένα αντικείμενά μας, σε αυτή την περίπτωση, τους σηματοδότες του CARLA στους δρόμους, τα χρώματα τους και τα κοντινά οχήματα. Για αυτό, χρησιμοποιήθηκε το μικροσκοπικό μοντέλο της τέταρτης έκδοσης Yolo (Yolov4).



Σχήμα 3.2: Συνολική δομή του YOLOv4 [32]

Ο YOLO είναι σύστημα ανίχνευσης αντικειμένου σε πραγματικό χρόνο βασισμένο χυρίως στην παλινδρόμηση, το οποίο προβλέπει τις κλάσεις και τα οριακά πλαίσια για ολόκληρη την εικόνα σε μία μονάχα εκτέλεση του αλγορίθμου (εξού και το όνομα **YOU ONLY LOOK ONCE**). Ο σχεδιασμός του δικτύου είναι εμπνευσμένος από το μοντέλο GoogLeNet για ταξινόμηση εικόνας [62] αλλά αντί για 22 βαθιά Στρώματα Convolutional Neural Networks (CNN), ο YOLO έχει 16 επίπεδα και η πλήρης τέταρτη έκδοση μοντέλου έχει 53 συνελικτικά στρώματα με μεγέθη  $1 \times 1$  και  $3 \times 3$ , και κάθε στρώμα συνέλιξης συνδέεται με ένα στρώμα

κανονικοποίησης παρτίδας (BN) και για συνάρτηση ενεργοποίησης Mish [32]. Αξιό αναφοράς είναι το τι συντελεί την συγκεκριμένη έκδοση καθώς είναι και αυτή που βασίζεται πάνω στα αποτελέσματα της όλη η συγκεκριμένη εργασία.

Ο YOLOv4 αποτελείται από [12]:

- Ραχοκοκαλιά (Backbone): CSPDarknet53 [69]
- Λαιμό (Neck): SPP [26], PAN [33]
- Κεφαλή (Head): YOLOv3[47]

Όλοι οι ανιχνευτές αντικειμένων λαμβάνουν μια εικόνα για είσοδο και συμπλέζουν τα χαρακτηριστικά προς τα κάτω μέσω μιας ραχοκοκαλιάς συνελικτικού νευρωνικού δικτύου [58]. Στην ταξινόμηση εικόνων, αυτές οι ραχοκοκαλιές είναι το τέλος του δικτύου και μπορεί να γίνει πρόβλεψη από αυτές. Στην ανίχνευση αντικειμένων, πολλά πλαίσια οριοθέτησης πρέπει να σχεδιάζονται γύρω από τις εικόνες μαζί με την ταξινόμηση, επομένως τα στρώματα χαρακτηριστικών της συνελικτικής ραχοκοκαλιάς πρέπει να αναμειγνύονται και να συγκρατούνται το ένα υπό το φως του άλλου. Ο συνδυασμός των στρωμάτων χαρακτηριστικών της ραχοκοκαλιάς λαμβάνει χώρα στον λαιμό [58].

Είναι επίσης χρήσιμο να χωρίζονται οι ανιχνευτές αντικειμένων σε δύο κατηγορίες: ανιχνευτές ενός σταδίου και ανιχνευτές δύο σταδίων. Η ανίχνευση γίνεται στη κεφαλή [58]. Οι ανιχνευτές δύο σταδίων αποσυνδέουν το έργο του εντοπισμού και της ταξινόμησης αντικειμένων για κάθε πλαίσιο οριοθέτησης. Οι ανιχνευτές ενός σταδίου κάνουν τις προβλέψεις για τον εντοπισμό και την ταξινόμηση αντικειμένων ταυτόχρονα [58].

Η κύρια διαφορά μεταξύ των δύο μοντέλων YOLO και GoogleNet, δηλαδή αυτό που συνεπάγεται, είναι ότι το πρώτο μοντέλο είναι γρηγορότερο από το δεύτερο, ωστόσο, είναι λιγότερο ακριβές. Αφού το σύστημα λειτουργεί στην CPU, είναι λογικό να υπερισχύει η άποψη του να χρησιμοποιηθεί το γρηγορότερο μοντέλο, ελαχιστοποιώντας τον επιπρόσθετο χρόνος φόρτωσης που εισάγεται κατά την ενσωμάτωση της μονάδας αυτής στον προσομοιωτή. Επιπλέον, το μικροσκοπικό μοντέλο λειτουργεί καλύτερα από το πλήρες μοντέλο για περιορισμένα περιβάλλοντα και για την ανίχνευση μικρών και μεγάλων αντικειμένων εκτός από την ανίχνευση μικρών αντικειμένων που εμφανίζονται συγκεκριμένα σε ομάδες [12].

Στις περιπτώσεις που αντιμετωπίζει αυτή η πτυχιακή εργασία, οι σηματοφόροι στους δρόμους και η λάμψη του χρώματος που έχουν, αλλάζουν το μέγεθός τους ενώ το όχημα πλησιάζει σε αυτούς. Αυτό ισχύει και για την λάμψη από το χρώμα και το φως των οχημάτων στο δρόμο. Οπότε αυτό το χαρακτηριστικό του δικτύου ταιριάζει απόλυτα στις ανάγκες των προβλημάτων που αποσκοπεί αυτή η εργασία να επιλύσει. Επίσης, δεν εμφανίζονται σε ομάδες όπως για παράδειγμα θα εμφανίζονταν σε ομάδες τυχαίοι πεζοί οι οποίοι σε ένα άλλο πρόβλημα θα ηταν επιψυμητό υποθετικά να αναγνωριστούν από κάποιο σύστημα ανίχνευσης το οποίο ιδανικά να ειδικεύεται στις ομάδες. Ωστόσο, το δίκτυο πρέπει να διαμορφωθεί σωστά για να το εκπαιδευτεί ορθά και να βελτιώσει με αυτόν τον τρόπο όσο γίνεται την ακρίβειά του. Οι αλλαγές που έγιναν στις παραμέτρους των στρωμάτων δικτύων που διαμορφώνουν το μοντέλο εξηγούνται στην ακόλουθη ενότητα 3.2.

### 3.2 Τροποποιήσεις Δικτύου και Υπερπαραμέτρων

Αρχικά, η διαμόρφωση του δικτύου προσαρμόστηκε στην περίπτωση που πρέπει να αντιμετωπιστεί για την επίτευξη των επιθυμητών εφαρμογών. Το πρώτο βήμα ήταν η αλλαγή του batch size από 1 σε 64. Αυτή η μεταβλητή καθορίζει τον αριθμό δειγμάτων (εικόνων) που θα διαδοθούν μέσω του δικτύου σε κάθε επανάληψη. Επιλέγοντας έναν batch size μικρότερο από τον συνολικό αριθμό εικόνων εκπαίδευσης ο οποίος είναι 7289, έχει ορισμένα πλεονεκτήματα. Ένα από αυτά είναι ότι απαιτείται λιγότερος χώρος μνήμης κατά τη διάρκεια της διαδικασίας εκπαίδευσης. Αυτό συμβαίνει ως αποτέλεσμα του δίκτυου να εκπαιδεύεται με λιγότερα δείγματα. Ταυτόχρονα εκπαιδεύεται και πιο γρήγορα γιατί τα βάρη του νευρωνικού δικτύου ενημερώνονται μετά από κάθε διάδοση. Από την άλλη, όσο μικρότερο είναι το batch size, τόσο λιγότερο ακριβής θα είναι η τελική εκτίμηση. Από προεπιλογή, ορίζεται μια στοχαστική παρτίδα (batch size ίσο με 1), αλλά ένα μέγευθος παρτίδας 64 προσφέρεται καλύτερα με το δίκτυο αρχιτεκτονικής του Yolov4 [30].

Επίσης, ο αριθμός των μέγιστων επαναλήψεων άλλαξε: η μεταβλητή max\_batches ορίζεται πολλαπλασιάζοντας το 2000 επί τον αριθμό των κλάσεων. Αυτό συμβαίνει γιατί επικρατεί η άποψη ότι 2000 είναι οι ιδανικές επαναλήψεις ανά τάξη που απαιτούνται για την κατάλληλη εκπαίδευση του δικτύου. Στην περίπτωσή που αντιμετωπίζει η συγκεκριμένη εργασία υπάρχουν 4 τάξεις οι οποίες έχουν δηλωθεί πριν την εκπαίδευση στο objs.names αρχείο. Αυτές οι κλάσεις είναι:

1. Red (για σηματοδότη με κόκκινο χρώμα)
2. Green (για σηματοδότη με πράσινο χρώμα)
3. Yellow (για σηματοδότη με κίτρινο χρώμα)
4. Vehicle\_InFront (για κοντινό μπροστινό όχημα)

Επομένως, το λογικό επακόλουθο είναι να χρειαστούν  $2000 * 4 = 8000$  επαναλήψεις για την εκπαίδευση το μοντέλο. Ο ρυθμός εκμάθησης παραμένει από προεπιλογή στο 0,001.

Στη συνέχεια, το 80% και το 90% της μέγιστης τιμής επαναλήψεων θα αποτελέσουν τον αριθμό των βημάτων που θα υπάρξουν στο δίκτυο. Οπότε η σχετική μεταβλητή ορίζεται ως “6400,7200” δηλαδή το 80% και 90% του 8000. Άξιο αναφοράς είναι ότι έγινε μια καυθοριστική αλλαγή στο δίκτυο. Η ανάλυση του δικτύου μειώθηκε από 412x412 σε 224x224. Αυτό συνέβη λόγω των διαθέσιμων πόρων που υπήρχαν για την υλοποίηση της εργασίας και κατ’ επέκταση για την εκπαίδευση του νευρωνικού δικτύου. Χρειαζόταν να μειωθεί ο όγκος των δεδομένων ώστε να είναι εφικτό το να μπορεί να πραγματοποιηθεί η εκπαίδευση. Παρόλο το ρίσκο της μείωσης της ακρίβειας, όπως θα φανεί παρακάτω κιόλας τα αποτελέσματα είναι αξιοπρεπέστατα, δείχνοντας ότι ο YOLO προσφέρει αποτελεσματικότητα ακόμα και υπό περιορισμούς.

Μια ακόμα μεταβλητή που έμεινε ως έχει ειναι η burn in η οποία ισούται με 1000. Δεν έχει κάποια λειτουργική αξία στην εκπαίδευση όμως παίζει τον ρόλο της στην έκθεση και παρουσίαση των αποτελεσμάτων που θα ακολουθήσει στο κεφάλαιο 5. Ουσιαστικά αποθηκεύει την πρόοδο του δικτύου ανά τον αριθμό επαναλήψεων που έχει οριστεί. Στη συγκεκριμένη περίπτωση ανά 1000 επαναλήψεις.

Επίσης, μαζί με την αποθήκευση της προόδου του δικτύου αποθηκεύεται και ένα διάγραμμα που οπτικοποιεί πόσο έχει μειωθεί το σφάλμα του δικτύου δηλαδή με λίγα λόγια πόσο έχει βελτιωθεί η ακρίβεια του. Ο παρακάτω πίνακας 3.1 δείχνει τις προσαρμοσμένες παραμέτρους για τη διαμόρφωση του δικτυού κατά την προπονητική διαδικασία.

Διαμόρφωση Δικτύου Εκπαίδευσης			
batch	64	exposure	1.5
subdivisions	128	hue	0.1
width	224	mosaic	1
height	224	learning_rate	0.001
channels	3	burn_in	1000
momentum	0.949	max_batches	8000
decay	0.0005	policy	steps
angle	0	steps	6400,7200
saturation	1.5	scales	0.1,0.1

Πίνακας 3.1: Τιμές διαμόρφωσης του YOLO από αρχείο yolov4-objs.cfg

Το CSPDarknet53 [69] ως κεφαλή του YOLOv4 περιέχει 29 συνελικτικά στρώματα  $3 \times 3$  και ένα δεκτικό πεδίο  $725 \times 725$  [12]. Όλες οι συναρτήσεις ενεργοποίησης στον YOLOv4 αντικαθίστανται με την leaky-ReLU που απαιτεί λιγότερο υπολογισμό στην ενημέρωση που γίνεται στα βάρη. Όμως, το συνελικτικό επίπεδο πριν από το επίπεδο Yolo χρησιμοποιεί μια γραμμική συνάρτηση ενεργοποίησης. Το SPPnet αυξάνει αποτελεσματικά το δεκτικό πεδίο του μοντέλου μέσω διαφορετικών στρωμάτων max-pooling με μέγεθος 5, 9 και 13 [32]. Το PANet χρησιμοποιεί από πάνω προς τα κάτω και από κάτω προς τα πάνω προσεγγίσεις για την επανειλημένη εξαγωγή χαρακτηριστικών. Τρία κεφάλια YOLO με μεγέθη  $19 \times 19$ ,  $38 \times 38$ ,  $76 \times 76$  χρησιμοποιούνται για τη σύντηξη και αλληλεπιδρούν με χάρτες χαρακτηριστικών διαφορετικής κλίμακας για την ανίχνευση αντικειμένων διαφορετικών μεγεθών [32]. Σε αυτή τη μελέτη του μικροσκοπικού μοντέλου Yolov4, η αρχική δομή του μοντέλου τροποποιείται για να μειώσει την πολυπλοκότητα του υπολογισμού χρησιμοποιώντας την μέθοδο κλαδέματος στρώσης [32].

Για να εκπαιδευτεί σωστά το δίκτυο λαμβάνοντας υπόψη το σύνολο δεδομένων εισόδου, έγιναν κάποιες αλλαγές κατά την εκτέλεση. Πρώτον, σε καθένα από τα 3 επίπεδα Yolo, άλλαξε ο αριθμός των τάξεων στον αριθμό των αντικειμένων που μαθαίνει το μοντέλο δηλαδή στο 4. Επίσης, σε κάθε συνελικτικό επίπεδο πριν από κάθε επίπεδο Yolo, ο αριθμός των φίλτρων τροποποιήθηκε σύμφωνα με αυτόν τον κανόνα: φίλτρα =  $(\text{τάξεις}+5) * 3$ . Όπότε ο αριθμός των filters ορίστηκε ως  $(4+5)*3=27$ . Η τυχαία τιμή (random value) ορίστηκε στο 1 για να αυξηθεί η ακρίβεια του YOLO εκπαιδεύοντάς τον σε διαφορετικές αναλύσεις, μεγέθη και επιτρεπόμενες εικόνες γενίκευσης σε διαφορετικά μεγέθη αντικειμένων.

Μόλις το μοντέλο ορίστηκε, προετοιμάστηκε το σύνολο δεδομένων εκπαίδευσης και επικύρωσης. Για αυτό, χρησιμοποιήθηκε μία από τις λειτουργίες του προσωμοιωτή CARLA, η καταγραφή στιγμιότυπών. Αναμίχθηκαν διαφορετικές καιρικές

καταστάσεις σε διαφορετικά σενάρια. Επιπλέον, για να υπάρξει συνεισφορά στην έρευνα που καταβάλλεται για την αυτοοδήγηση, ολόκληρα τα φιλτραρισμένα σύνολα δεδομένων που χρησιμοποιήθηκαν για τις ανάγκες τις εργασίας, με περισσότερα από 7000 χαρέ το καθένα, βρίσκονται στον ιστότοπο [Github](#) που προετοιμάστηκε για αυτό το έργο. Μπορεί ο οποιοσδήποτε να επισκεφτεί τον ιστότοπο, να κατευθυνθεί στο [link](#) του σχετικού [GoogleDrive](#) που υπάρχει στην περιγραφή και να κατεβάσει τα σύνολα φωτογραφιών εκπαίδευσης και επικύρωσης εντελώς δωρεάν.

### 3.3 Διαδικασία Εκπαίδευσης Δικτύου

Σε αυτή την ενότητα θα αναλυθούν οι βασικοί πυλώνες που αποτελούν την διαδικασία υλοποίησης του μοντέλου αντίληψης. Θα εκτεθούν τα μέσα που χρησιμοποιήθηκαν και πως αυτά αξιοποιήθηκαν και έδωσαν τα εφόδια στη συνέχεια να αναπτυχθούν οι 4 εφαρμογές της εργασίας, που αναλύονται στο κεφάλαιο 4, μέσα στο γραφικό περιβάλλον του [Carla](#).

#### 3.3.1 Συλλογή Δεδομένων

Το άλφα και το ωμέγα για να υπάρξει εκπαίδευση σε οποιονδήποτε τομέα του Machine Learning είναι η ύπαρξη δεδομένων που θα τεθούν υπό επεξεργασία για την εξαγωγή αποτελεσμάτων ή και συμπερασμάτων. Άλλοτε αυτά τα δεδομένα είναι αριθμητικά δεδομένα διάφορων τιμών και άλλοτε είναι φωτογραφίες όπου μας ενδιαφέρει η αριθμητική πληροφορία που εμπεριέχουν. Στη περίπτωση μας, τα δεδομένα είναι φωτογραφίες όπου θέλουμε να μεταφραστούν `pixel` προς `pixel` ανάλογα με τις τιμές RGB που έχει το καθένα. Αυτό δεν θα χρειαστεί να ληφθεί υποψήν στην υλοποίηση του δικτύου καθώς ο YOLO μέσα από το Darknet το κάνει αυτόματα. Η έγνοια που πρέπει να υπάρξει είναι η σωστή προετοιμασία των δεδομένων για εκπαίδευση στην σωστή μορφή.

Μέσα από το [Carla](#) είναι εφικτό να ξεκινήσει μια γραφική προσομοίωση οδήγησης όπου ένα αφάξι είναι στον δρόμο και περιπλανιέται ομαλά με άλλα οχήματα και πεζούς τριγύρω. Οπότε η όλη διαδικασία ξεκινάει με το αφάξι να βρίσκεται στον έτοιμο αυτόματο πιλότο που προσφέρει το [Carla](#) και να κινείται μέσω αυτού. Στη συνέχεια χρειάζεται ένας `actor` ο οποίος θα έχει τον ρόλο της καταγραφής των δεδομένων που “βλέπει” το αφάξι. Για αυτό τον λόγο μέσα από το σχετικό `script test.py` όπου υλοποιήθηκε ότι αναλύεται στην εργασία, προστίθεται στο όχημα μια κάμερα RGB. Είναι προγραμματισμένη βάση της θέσης τοποθέτησης της, να αποθηκεύει εικόνες που αντιπροσωπεύουν το τι συμβαίνει στην μπροστινή οπτική του αφάξιού. Ένα παράδειγμα τέτοιας φωτογραφίας φαίνεται στην εικόνα 3.3.

Οι διαστάσεις που επιλέχθηκαν για τις φωτογραφίες εκπαίδευσης είναι 800 πλάτος και 600 ύψος. Επιλέχθηκαν τέτοιες διαστάσεις διότι όπως θα εξηγηθεί παρακάτω, προορίζεται χειροκίνητα να αποθηκεύεται η χρησιμότητα της πληροφορίας που προσφέρουν αυτές οι φωτογραφίες. Αυτό σημαίνει ότι αν ήταν μικρές σε διάσταση θα κάνανε πιο δύσκολη αυτή την διαδικασία για έναν άνθρωπο, πράγμα που θα γίνει πλήρως αντιληπτό στην επόμενη υποενότητα. Επίσης, είναι άξιο σχολιασμού η ταχύτητα αποθήκευσης φωτογραφιών που συμβαίνει σε πραγματικό χρόνο. Η μεταβλητή `fps` στον τοπικό `server` έχει οριστεί ως 28 λόγω των διαθέσιμων πόρων του τοπικού υπολογιστή. Με δεδομένο αυτό μέσα από το λογισμικό



Σχήμα 3.3: Παράδειγμα εικόνας εκπαίδευσης

που αναπτύχθηκε, ανά 7 frames αποθηκεύεται και μια φωτογραφία. Είναι αρκετά λειτουργικό αυτό ως προγραμματιστική επιλογή καθώς πολλές εικόνες είναι πανομοιότυπες αν είναι υπερβολικά συχνή η εντολή καταγραφής εικόνας. Αυτό ως γεγονός δεν εξυπηρετεί την ορθή εκπαίδευση του δικτύου καθώς είναι επιθυμητό τα δεδομένα να διαφέρουν όσο γίνεται μεταξύ τους ώστε να προστίθεται ουσιώδης πληροφορία κατά την διάρκεια της εκπαίδευσης.

Σημαντικό είναι επίσης το κριτήριο που υπήρχε πίσω από το τελικό σύνολο εκπαίδευσης που χρησιμοποιήθηκε. Μέσα από τις φωτογραφίες που τραβήχτηκαν επιλέχθηκαν 7289. Αυτό έγινε με σκοπό να μην είναι υπερβολικά τα δεδομένα που έχει να επεξεργαστεί ο τοπικός υπολογιστής και ταυτόχρονα να εγγυύνται κατά την εκπαίδευση αυτά τα δεδομένα, ένα αξιόλογο αποτέλεσμα.

Διαμόρφωση Δεδομένων Εκπαίδευσης		
Κλάση	Συχνότητα	Ποσοστό ως προς το σύνολο
Red	293	4%
Green	306	4.1%
Yellow	290	3.9%
Vehicle_InFront	431	5.9%
-	6055	83%

Πίνακας 3.2: Ποσοτική ανάλυση του συνόλου εκπαίδευσης

Τα κριτήρια που αποτέλεσαν βασικό ρόλο στην διαμόρφωση των δεδομένων είναι ότι πρώτα απ' όλα οι περισσότερες φωτογραφίες δεν θα έπρεπε να δείχνουν κάποιο αντικείμενο προς ανίχνευση. Στατιστικά το αμάξι κατά την πορεία του στον δρόμο δεν ανιχνεύει τίποτα. Κατά δεύτερον, πρέπει να είσαι σχεδόν ισόποσα κατανευημένες οι φωτογραφίες που δείχνουν ένα αντικείμενο προς εντοπισμό. Λίγο

παραπάνω έμφαση όμως θα πρέπει να δοθεί προς τον εντοπισμό των οχημάτων καθώς είναι πιο σύνηθες συμβάν ένα κοντινό μπροστινό όχημα. Με αυτά υπόψιν, όπως φαίνεται και στον πίνακα 3.2, στο περίπου οι 6000 φωτογραφίες δεν δείχνουν τίποτα, οι 400 δείχνουν ένα όχημα και οι 900 δείχνουν έναν σηματοδότη. Από τις 900 αυτές φωτογραφίες σχεδόν ισόποσα έχουν κατανεμηθεί και το κόκκινο και το πράσινο και το κίτρινο χρώμα σηματοδότη. Αφού λοιπόν έχει ετοιμαστεί το σύνολο των φωτογραφιών, ο επόμενος στόχος είναι η προσθήκη ετικετών.

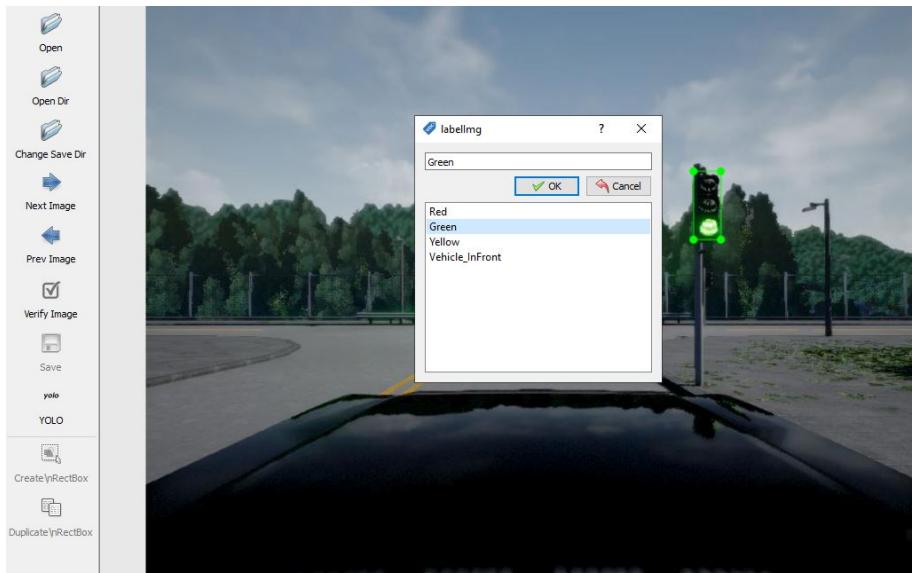
### 3.3.2 Χρήση **labelImg**

Το **labelImg** [31] είναι ένα γραφικό εργαλείο σχολιασμού εικόνας. Είναι γραμμένο σε Python και χρησιμοποιεί Qt για τη γραφική του διεπαφή. Οι σχολιασμοί αποθηκεύονται ως αρχεία xml σε μορφή PASCAL VOC, τη μορφή που χρησιμοποιείται από το ImageNet ή σε txt. Επιπλέον, υποστηρίζει επίσης μορφές YOLO που θα χρησιμοποιηθεί για την συγκεκριμένη εργασία και CreateML [31]. Ουσιαστικά η εφαρμογή αυτού του εργαλείου αποσκοπεί σε 2 στόχους:

1. Την δημιουργία αρχείων που εμπεριέχουν την γνώση του τι περιέχει η κάθε φωτογραφία.
2. Τις απαραίτητες πληροφορίες που χρειάζεται ο YOLO για να εκπαιδευτεί.

Η χρήση της εφαρμογής είναι εξαιρετικά απλή. Αρχικά, στις ρυθμίσεις πρέπει να επιλεχθεί ότι τα δεδομένα που επιθυμούνται να παραχθούν από τις φωτογραφίες είναι για μελλοντική εκπαίδευση στον αλγόριθμο YOLO. Έπειτα αφού το ποινηθεί η εφαρμογή στο path file όπου βρίσκονται οι φωτογραφίες ξεκινάει η διαδικασία της προσθήκης ετικετών. Για κάθε φωτογραφία πρέπει να πραγματοποιηθεί μια χειροκίνητη επαναλαμβανόμενη διαδικασία. Πρώτα απ' όλα πρέπει να δημιουργηθεί ένα πλαίσιο που θα ορίσουμε εμείς μέσα στην φωτογραφία. Μέσα σε αυτό το πλαίσιο θα βρίσκεται το ποινηθεμένο το αντικείμενο που υπάρχει στην εικόνα με όσο μεγαλύτερη ακρίβεια γίνεται όσον αφορά το ύψος και το πλάτος του πλαισίου. Αν δεν υπάρχει κάποιο αντικείμενο που ζητάει χρήστης να ανιχνεύεται κατά την εκπαίδευση, απλά πρέπει να προχωρήσει στην επόμενη εικόνα χωρίς να δημιουργήσει κανένα πλαίσιο. Αυτή η διαδικασία κάνει πλήρως αντιληπτό το γιατί στην υποενότητα 3.3.1 τονίστηκε ότι χρειάζεται οι εικόνες να είναι σε ένα φυσιολογικό μέγεθος. Η δημιουργία του πλαισίου θα ήταν επίπονη σε φωτογραφίες μικρής ανάλυσης. Επίσης, αφού οριστεί ένα πλαίσιο σε μια εικόνα, έπειτα πρέπει να επιλεχθεί σε ποια κλάση ανήκει το αντικείμενο του πλαισίου. Στη περίπτωση της εργασίας αυτής είναι 4 οι κλάσεις οι οποίες έχουν αναφερθεί προηγουμένως και αναγράφονται στο πίνακα 3.2.

Σε αυτό το σημείο, πρέπει να επισημανθεί ότι η ποιότητα των δεδομένων δεν σταματάει στην συλλογή των δεδομένων αλλά και στον τρόπο που θα τους προστεθεί ετικέτα. Άμα το αμάξι εντόπιζε τους σηματοδότες από μεγάλη απόσταση μαζί με το χρώμα τους θα ήταν δύσκολο να ξεκαθαριστεί το χρονικό πλαίσιο που πρέπει να λάβει κάποια δράση. Αν εντοπιζόταν από μεγάλη απόσταση ένα όχημα θα ήταν δύσκολο να οριστεί το πότε αυτό είναι επικινδυνό σαν γεγονός και πότε είναι ακίνδυνο. Για αυτό επιτηδευμένα στην προσθήκη ετικετών, όποτε κρινόταν ότι τα αντικείμενα προς εντοπισμό είχαν μια μακρινή απόσταση δεν τους προστέθηκε καμία ετικέτα. Αυτό εξυπηρετεί ώστε το δίκτυο να εκπαιδευτεί να εντοπίζει τα



Σχήμα 3.4: Προσθήκη ετικέτας σε πράσινο σηματόδοτη μέσω LabelImg.

επιιψυμητά αντικείμενα μόνο σε σχετική κοντινή απόσταση. Το όφελος από αυτή την απόφαση είναι ότι όταν εντοπιστούν τα αντικείμενα, είναι ξεκάθαρο ότι την ίδια στιγμή πρέπει να ληφθεί κάποια απόφαση για το πως θα συμπεριφερθεί το αμάξι στον χώρο. Για παράδειγμα αν εντοπιστεί σηματοφόρος με κόκκινο χρώμα επειδή έχει εκπαιδευτεί το δίκτυο να εντοπίζει τους σηματόφορους μόνο από κοντινή απόσταση, πρέπει το αμάξι να σταματήσει ακαριαία. Άμα εντόπιζε το δίκτυο το χρώμα του σηματοφόρου από πιο μακριά, θα χρειαζόταν και άλλη υπολογιστική δύναμη και πράξεις για να αποσαφηνιστεί το πότε πρέπει το αμάξι να πατήσει φρένο. Αυτό ως δεδομένο, δεν θα συνέφερε καθόλου ένα σύστημα πραγματικού χρόνου όσον αφορά την απόδοση του.

Τέλος, πρέπει να σχολιαστούν και τα αποτελέσματα που παράγονται έπειτα από την χρήση της συγκεκριμένης εφαρμογής. Αφού όλη η παραπάνω διαδικασία γίνει για κάθε φωτογραφία, τότε παράλληλα με κάθε φωτογραφία δημιουργείται ένα αρχείο txt με το ίδιο όνομα που είναι αποθηκευμένη η εικόνα. Αυτό το αρχείο περιέχει 5 μεταβλητές. Η πρώτη είναι ο αριθμός της κλάσης του αντικειμένου που εντοπίζεται στην εικόνα και οι άλλες 4 μεταβλητές έχουν να κάνουν με το που βρίσκεται το πλαίσιο του αντικειμένου στην φωτογραφία. Και οι 4 μεταβλητές βρίσκονται στο διάστημα [0,1] καθώς η πληροφορία που προσφέρουν είναι αναλογική με το ύψος και το πλάτος της εκάστοτε εικόνας. Οι 2 πρώτες μεταβλητές έχουν να κάνουν με τις συντεταγμένες που βρίσκεται το κέντρο του πλαισίου και οι άλλες 2 έχουν να κάνουν με το μήκος και το πλάτος που έχει το πλαίσιο. Αυτές οι πληροφορίες στη συνέχεια της εκπαίδευσης θα εισαχθούν ως δεδομένα στον αλγόριθμο YOLO και θα μάθει μέσω αυτών να παράγει δικά του πλαισια που θα εντοπίζουν αντικείμενα.

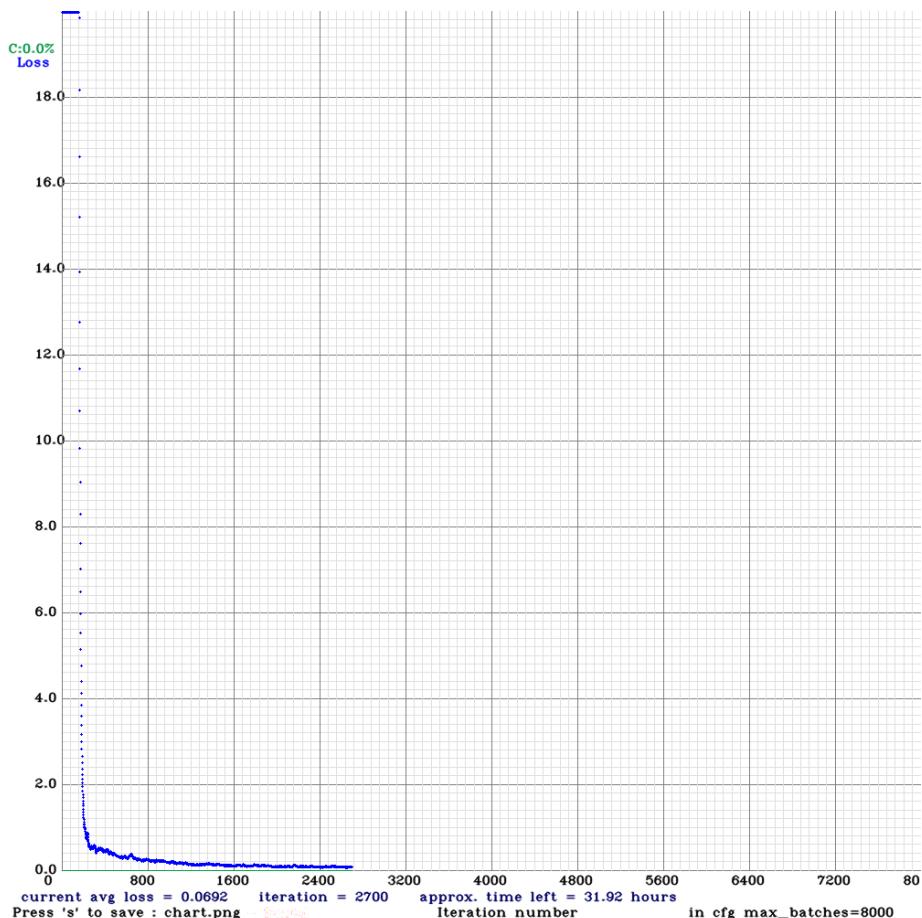
### 3.3.3 Εκπαίδευση YOLO

Αφού όλα έχουν ετοιμαστεί από τις διαδικασίες που αναλύθηκαν στις παραπάνω υποενότητες 3.3.1 και 3.3.2, έφτασε η στιγμή της εκπαίδευσης του δικτύου όπου

Θα εξάγει στη συνέχεια τα αποτελέσματα που θέλουμε. Η εκπαίδευση πραγματοποιείται πολύ απλοϊκά με εντολή μέσω cmd παραθύρου. Εφόσον στο παράθυρο που θα ανοιχτεί, βρίσκεται η ρίζα του στο path του darknet.exe η εντολή που πρέπει να τρέξει είναι η εξής:

```
darknet.exe detector train data/objs.data cfg/yolov4-objs.cfg
yolov4.conv.137
```

Οι παράμετροι που ορίζονται μέσα από την εντολή αυτή έχουν να κάνουν με τις σχετικές τροποποιήσεις του δικτύου και το που βρίσκονται τα δεδομένα εκπαίδευσης καθώς και το ποιες είναι οι επιθυμητές κλάσεις εντοπισμού. Επιπρόσθετα, η τελευταία παράμετρος έχει να κάνει με ήδη εκπαιδευμένα βάρη που διαθέτει το Darknet για τον YOLOv4. Η συγκεκριμένη ύπαρξη αυτών των βαρών εξυπηρετεί στο να επιταχύνει την ορθή διαδικασία εκπαίδευσης καθώς γίνεται η φόρτιση τους πάνω στο δίκτυο στην αρχική φάση εκπαίδευσης.



Σχήμα 3.5: Διάγραμμα μέσου σφάλματος εκπαίδευσης

Αφού εκτελεστεί η παραπάνω εντολή ουσιαστικά αναμένεται η εκπαίδευση του δικτύου. Όπως αναφέρθηκε και στην ενότητα 3.2, εφόσον ορίστηκε η μεταβλητή burn\_in στο 1000 τότε ανά 1000 επαναλήψεις θα αποθηκεύονται τα νέα εκπαιδευμένα βάρη που θα προκύπτουν μέσα από τις φωτογραφίες και τις πληροφορίες που εμπεριέχουν τα txt αρχεία. Σε αυτό το σημείο να αναφερθεί ότι από επιλογή η διαδικασία εκπαίδευσης σταματάει στις 2700 επαναλήψεις και ας έχει οριστεί στον σχετικό πίνακα 3.1 ότι θα σταματάει στις 8000. Αυτό συμβαίνει για 2 βασικούς

λόγους. Πρώτον, για να υπάρξει ένα γρήγορο αξιόλογο αποτέλεσμα βάση των διαθέσιμων πόρων του τοπικού υπολογιστή. Ο δεύτερος λόγος αφορά πιο λειτουργικό παράγοντα. Επειδή αντικειμενικά δεν είναι μεγάλος ο όγκος δεδομένων από τις φωτογραφίες εκπαίδευσης, υπάρχει μεγάλος κίνδυνος να συμβεί overfitting [73] αν οι επαναλήψεις φτάσουν σε μεγάλο αριθμό. Αυτό θα οδηγούσε στο δυσάρεστο αποτέλεσμα το δίκτυο να μην αποδίδει καλά σε δεδομένα πέρα από αυτά που εκπαιδεύτηκε. Οπότε για να διατηρηθεί η κρίση του δικτύου στο άγνωστο σε φυσιολογικά επίπεδα, επιλέχθηκε η πρόωρη διακοπή εκπαίδευσης.

Αφού τελειώσει η εκπαίδευση, η οποία διαρκεί περίπου 12 ώρες, το Darknet παράγει ένα γράφημα που αφορά την πορεία εκπαίδευσης του δικτύου και δείχνει την πορεία πτώσης του μέσου σφάλματος. Όπως φαίνεται και στην εικόνα 3.5 του διαγράμματος, το μέσο σφάλμα στο τέλος της εκπαίδευσης ισούται με 0.0692 έναν ικανοποιητικά χαμηλό αριθμό.

### 3.4 Εξαγωγή Αποτελεσμάτων μέσω **Darknet**

Στις προηγούμενες ενότητες 3.2 και 3.3 εξηγήθηκε αναλυτικά το πως τροποποιήθηκε το νευρωνικό δίκτυο YOLO και πως υποβλήθηκε στην διαδικασία εκπαίδευσης. Όλα αυτά οργανικά οδηγούν στον απλό στόχο της εξαγωγής αποτελεσμάτων και το πως πραγματοποιείται αυτή μέσα από το Darknet.

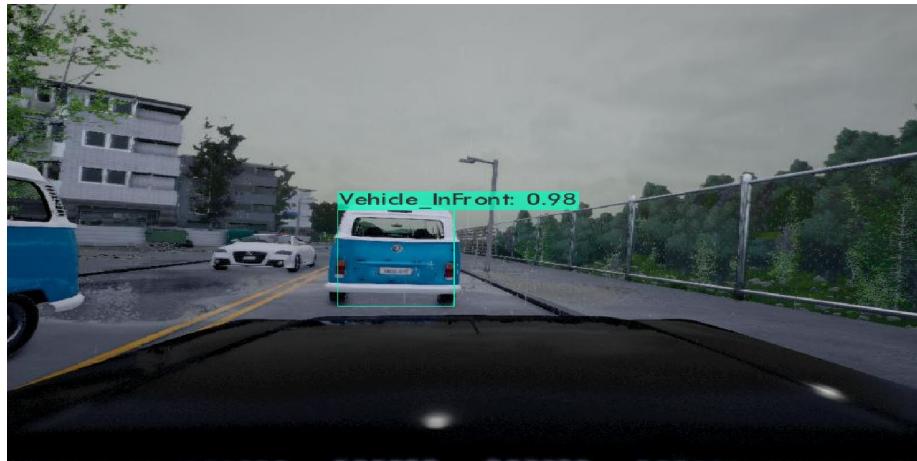
Αρχικά, πρέπει με ρίζα πάντα το path όπου βρίσκεται το darknet.exe στον εκάστοτε τοπικό υπολογιστή, να ανοιχτεί ένα cmd παράθυρο. Αφού το παράθυρο είναι ενεργό, για να ξεκινήσει η όλη διαδικασία χρειάζεται πολύ απλά να τρέξει μια συγκεκριμένη εντολή η οποία είναι η παρακάτω:

```
darknet.exe detector test data/objs.data cfg/yolov4-objs.cfg backup
/yolov4-objs_last.weights -ext_output -save_labels
```

Οι κύριες παραμέτροι βασίζονται κυρίως στο που βρίσκονται τα αρχεία που χρειαζόνται, όπως οι μεταβλητές του δικτύου YOLO και τα αποθηκευμένα εκπαιδευμένα βάρη. Αφού τρέξει η παραπάνω εντολή τότε φορτίζει το νευρωνικό δικτύου με τις παραμέτρους του και αφού ετοιμαστεί για χρήση ζητάει από τον χρήστη ένα path φωτογραφίας. Από αυτή την φωτογραφία θα προσπαθήσει να εντοπίσει το εκπαιδευμένο δίκτυο αν υπάρχει κάποια από τις κλάσεις που εκπαιδεύτηκε να αναγνωρίζει, αν εντοπίζει κάτι να το οριοθετεί σε συγκεκριμένο πλαίσιο και να συμπληρώνει ποσοστιαία την σιγουριά που έχει για την πρόβλεψη που έκανε.

Αφού γίνει η πρόβλεψη, εκτός από οπτικό αποτέλεσμα που φαίνεται μέσα από την εικόνα 3.6, οι αριθμοί που μας ενδιαφέρουν ως μεταβλητές και αποτελέσματα αναγράφονται στο παράθυρο cmd. Καθώς γίνεται η πρόβλεψη του τι εμπεριέχει η φωτογραφία που επιλέχθηκε, δημιουργείται παράλληλα και ένα αρχείο txt. Αυτό το αρχείο εμπεριέχει τον αριθμό της κλάσης που προέβλεψε το δίκτυο και 4 ποσοστιαίες μεταβλητές που σχετίζονται με το μέγεθος του πλαισίου που υπάρχει το αντικείμενο στην φωτογραφία. Οι 2 πρώτες μεταβλητές έχουν να κάνουν με το κέντρο του πλαισίου στην φωτογραφία και οι άλλες 2 με το ύψος και το πλάτος του πλαισίου όπως εξηγήθηκε και στην υποενότητα 3.3.2.

Όλη αυτή η διαδικασία της εκπαίδευσης με την παραγωγή των συγκεκριμένων αποτελεσμάτων θα οδηγήσει στο κεφάλαιο 4 που εξηγούνται οι εφαρμογές οι οποίες υλοποιήθηκαν μέσα από αυτά τα αποτελέσματα του Darknet και της επιτυχής



Σχήμα 3.6: Εντοπισμός οχήματος με 98% βεβαιότητα

σύνδεσης του με το Carla .Τα αποτελέσματα μέσα στο εκάστοτε txt αρχείο που παράγεται ανα πρόβλεψη, αποτελούν καθοριστικό ρόλο σχεδόν σε όλες τις εφαρμογές που υλοποιήθηκαν παρακάτω. Για παράδειγμα, η χλάση που προέβλεψε το δίκτυο επηρεάζει και το τι θα εκτυπωθεί στην εφαρμογή της ενότητας 4.1 και το τι απόφαση θα πάρει το λογισμικό για το αμάξι στην εφαρμογή της 4.2 ενότητας.

## Κεφάλαιο 4

# Εφαρμογές στο Carla

Σε αυτό το κεφάλαιο θα εκτεθούν όλες οι εφαρμογές που δημιουργήθηκαν με τις δυνατότητες που προσφέρει το γραφικό περιβάλλον του CARLA και με την συνδεσμότητα των APIs και του Darknet. Ουσιαστικά κάθε εφαρμογή που θα αναλυθεί στοχεύει μόνο σε ένα πράγμα: την ομαλή αυτόνομη οδήγηση και την ελάττωση ατυχών συμβάντων και ατυχημάτων. Οι εφαρμογές οι οποίες τέθηκαν προς υλοποίηση είναι οι εξής:

1. Εφαρμογή προειδοποίησης
2. Εφαρμογή Ελέγχου
3. Εφαρμογή Πλαισίου Αντικειμένων
4. Εφαρμογή Προσπέλασης Δεδομένων

Οι τρεις πρώτες εφαρμογές έχουν να κάνουν άμεσα με την οδήγηση που συμβαίνει σε πραγματικό χρόνο. Αντιθέτως, η τέταρτη έχει να κάνει κυρίως με την καταμέτρηση των δεδομένων και την εξαγωγή στατιστικών πορισμάτων. Είναι ισάξιας σημασίας με τις προηγούμενες διότι αυτή η εφαρμογή ουσιαστικά θα κατορθώσει να οδηγήσει σε συμπεράσματα όπως αν είναι λειτουργικό το δίκτυο και αν μπορεί να αποδώσει σε πραγματικό χρόνο κατά την οδήγηση στον δρόμο. Επίσης, θα εξάγει και το ποσοστό ακρίβειας του YOLO μέσα από δεδομένα ελέγχου.

### 4.1 Εφαρμογή προειδοποίησης

Αυτή η εφαρμογή αναπτύχθηκε ώστε με ένα μήνυμα να κατορθώνει το λογισμικό να ειδοποιεί τον χρήστη για το τι συμβαίνει στον χώρο. Ουσιαστικά, το τι εντοπίζεται μέσα από το δίκτυο ως αντικείμενο, χρησιμοποιείται ως πληροφορία για να γίνει αντιληπτό στον χρήστη ότι ο YOLO είτε θεωρεί ότι είναι όλα ομαλά είτε ότι μπορεί να υπάρξει κάποιος κίνδυνος που καλό θα ήταν να αποφευχθεί. Η συγκεκριμένη εφαρμογή δημιουργεί μια ένδειξη μηνύματος για:

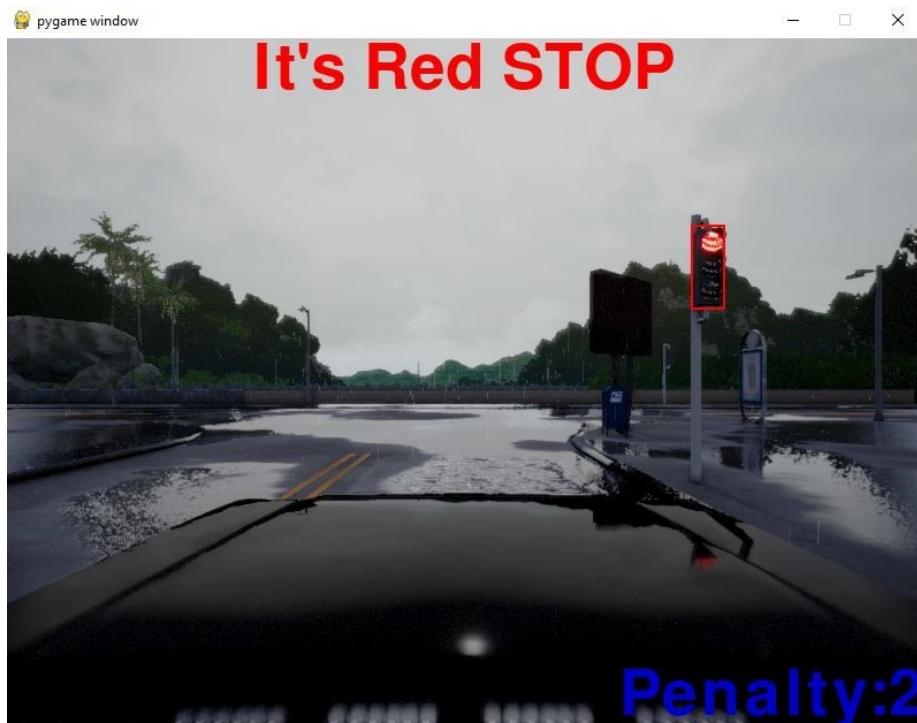
- Εντοπισμό σηματοδότη και του χρώματος του.
- Εντοπισμό οχήματος.
- Λανθασμένη προσπέραση σηματοδότη.

Η ουσιαστική εξυπηρέτηση που προσφέρεται μέσα από τις δυνατότητες αυτής της εφαρμογής είναι να γίνεται ακόμα πιο αισθητό σε έναν οδηγό το τι συμβαίνει στον δρόμο. Του ενισχύει το αίσθημα κίνδυνου και επιβίωσης και τον αναγκάζει να είναι περισσότερο σε ετοιμότητα. Αυτό εξυπηρετεί στο να είναι σε θέση να λάβει δράση για να αποφύγει κάποιο πιθανό ατύχημα ή κάποια οδική παράβαση αφού έχει λάβει έγκαιρα μια προειδοποίηση.

#### 4.1.1 Εντοπισμός Σηματοδότη

Καθώς εντοπίζεται ένας σηματοδότης στον δρόμο από το δίκτυο που εκπαιδεύτηκε, καλείται να αναγνωρίσει τι χρώμα είναι. Ανάλογα του τι χρώμα είναι θα οριστεί και το αντίστοιχο μήνυμα που είναι επιθυμητό να εμφανιστεί στον χρήστη/οδηγό. Αν το χρώμα του σηματοδότη είναι κόκκινο τότε η ένδειξη που εμφανίζεται είναι «It's Red STOP». Αυτή η ένδειξη ουσιαστικά θα εμφανιστεί μόνο αν το αμάξι βρίσκεται κοντά σε σηματοδότη που έχει κόκκινο χρώμα. Οπότε σε αυτή την περίπτωση πρέπει να σταματήσει αμέσως το αμάξι αλλιώς ουσιαστικά παραβιάζει τον κώδικα οδικής κυκλοφορίας, κάτι που δεν είναι αποδεχτό. Αν υπό τις ίδιες συνθήκες εντοπιστεί κίτρινος σηματοδότης τότε το μήνυμα προειδοποίησης είναι το «It's Yellow BRAKE». Ουσιαστικά σε αυτή την περίπτωση, ο οδηγός ειδοποιείται να αρχίσει άμεσα την διαδικασία φρεναρίσματος καθώς μετά το κίτρινο χρώμα έπειται το κόκκινο που εκεί θα επιβάλλεται να είναι σταματημένο το όχημα έτσι και αλλιώς. Ως τελευταία ένδειξη προς τον χρήστη, αποτελεί η ένδειξη για εντοπισμό πράσινου σηματοφόρου η οποία είναι «It's Green GO». Ο στόχος της είναι να κάνει ξεκάθαρο στον χρήστη ότι το όχημα πρέπει να σταματήσει να βρίσκεται σε ακίνητη θέση και να τεθεί σε κίνηση και περιπλάνηση μέσα στον δρόμο. Επίσης, όταν δεν εντοπίζεται ούτε σηματοφόρος ούτε μπροστινό όχημα, η ένδειξη στον χρήστη είναι απλά «GO». Πρακτικά ισοδυναμεί σε λογική με τον εντοπισμό του πράσινου σηματοφόρου αφού πάλι πρέπει να βρίσκεται σε κίνηση το όχημα και να μην είναι σταματημένο εμποδίζοντας έτσι την κυκλοφορία.

Σε αυτό το σημείο πρέπει να επισημανθεί κάτι που αποτελεί βασικό ρόλο. Και αυτό είναι το που θα εμφανιστεί ακριβώς το επιθυμητό μήνυμα προειδοποίησης για τους σηματοδότες και τα χρώματα τους. Σε αυτό το πρόβλημα, η λύση είναι το *pygame*. Κατά την εκτέλεση του λογισμικού, εκτός από τον *server* του *Carla* ανοίγει και ένα παράθυρο *pygame*. Η κύρια χρησιμότητα που έχει είναι κυρίως για τον έλεγχο του αμαξιού από τον χρήστη, δηλαδή μέσω του πληκτρολογίου. Επίσης, σε πραγματικό χρόνο δείχνει το τι “βλέπει” η κάμερα RGB η οποία είναι τοποθετημένη πάνω στο όχημα. Όμως, όσον αφορά τις εφαρμογές προειδοποιήσεις η δυνατότητα που έχει ενδιαφέρον είναι το ότι είναι εφικτό να οριστεί ένα πλαίσιο σε οποιοδήποτε σημείο του παραθύρου που θα γράψει ότι επιθυμεί ο εκάστοτε προγραμματιστής. Στην περίπτωση που αναλύεται τώρα, αυτό που πρέπει να γίνει είναι η καταγραφή επιθυμητών μηνυμάτων ανάλογα το χρώμα του σηματοδότη. Αυτό κατορθώνεται πολύ εύκολα μέσω του *pygame* καθώς με τις συναρτήσεις που έχει ως πακέτο, ορίζονται ευκολά οι επιθυμητές συντεταγμένες για να εμφανιστεί το κάθε μήνυμα. Επίσης, με την ίδια ευκολία ορίζεται το μέγεθος των γραμμάτων μαζί με το χρώμα τους. Στην εικόνα 4.1 φαίνεται ξεκάθαρα πως αναγράφεται ένα προειδοποιητικό μήνυμα κατά τον εντοπισμό σηματοδότη. Όταν το μήνυμα αφορά

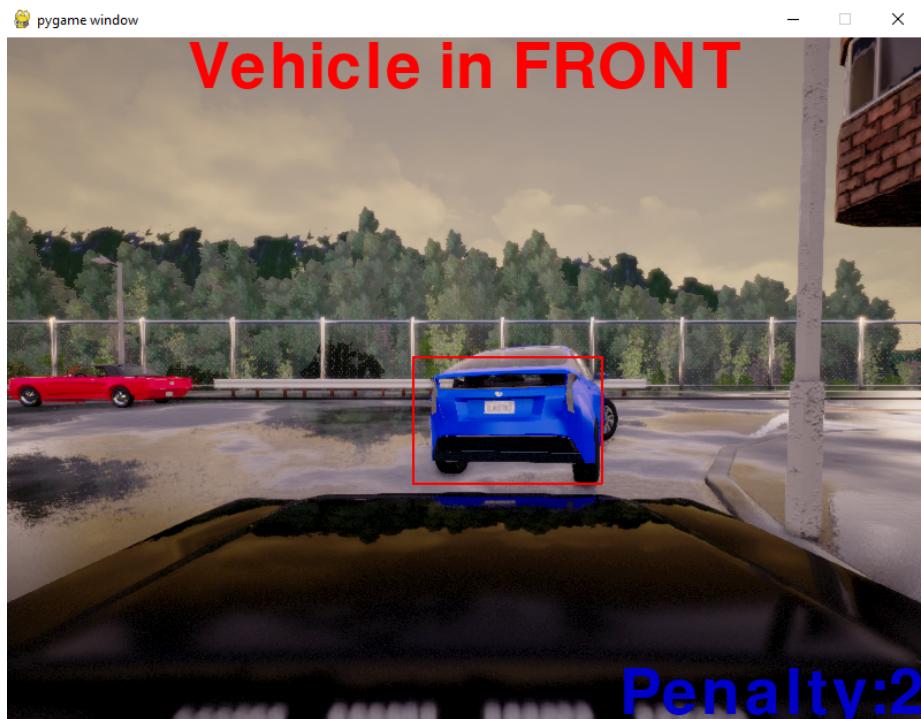


Σχήμα 4.1: Μήνυμα προειδοποίησης κόκκινου σηματοδότη μέσα από παράθυρο pygame.

την εμφάνιση κόκκινου σηματοδότη τα γράμματα είναι κόκκινα και το ίδιο μοτίβο ακολουθείται για κάθε χρώμα σηματοδότη.

#### 4.1.2 Εντοπισμός οχήματος

Ο ανίχνευση στον χώρο ενός κοντινού μπροστινού οχήματος από το δίκτυο, γίνεται με την ίδια διαδικασία σχεδόν που αναλύθηκε και στην προηγούμενη υποενότητα όπου εξηγείται τι συμβαίνει με τους σηματοδότες. Το δίκτυο μέσω της κάμερας RGB προσπαθεί σε πραγματικό χρόνο να εντοπίσει την ύπαρξη κάποιου επιθυμητού αντικειμένου. Αν εντοπίσει μπροστινό όχημα τότε ακολουθεί η ένδειξη "Vehicle in FRONT". Αυτό σημαίνει ότι έχει εντοπιστεί σε πολύ κοντινή απόσταση ένα άλλο όχημα και καλό θα ήταν να φρενάρει ώστε να αποφευχθεί κάποια πιθανότητα μετωπικής σύγκρουσης. Η διαδικασία εμφάνισης στον χρήστη/οδηγό είναι η ίδια ακριβώς με ότι αναφέρθηκε προηγουμένως για τους σηματοδότες μέσω της χρήσης του pygame. Το χρώμα που επιλέγεται για τα γράμματα του μηνύματος προειδοποίησης όπως φαίνεται και στην εικόνα 4.2 είναι το κόκκινο. Ουσιαστικά πρέπει να υπάρχει η ίδια αντικειμενική από το όχημα και στον εντοπισμό κόκκινου σηματοδότη και στο μπροστινό όχημα δηλαδή το ακαριαίο φρενάρισμα. Οπότε για λογούς ομαλής λογικής οπτικοποίησης, επιλέχθηκε το κόκκινο χρώμα για την συγκεκριμένη προειδοποίηση του οχήματος. Πρέπει όμως να ληφθεί υπόψιν ένα ακόμα ζήτημα που αφορά ταυτόχρονα και την προηγούμενη υποενότητα και την συγκεκριμένη. Εφόσον το σύστημα εκπαιδεύτηκε να εντοπίζει ταυτόχρονα και σηματοδότες και οχήματα στον χώρο, αν εντοπιστούν και τα 2 τότε πρέπει να ξεκαθαριστεί το μήνυμα προειδοποίησης που θα εμφανίζεται στον χρήστη. Τέτοια



Σχήμα 4.2: Μήνυμα προειδοποίησης κοντινού μπροστινού οχήματος μέσα από παράθυρο pygame.

Ζητήματα πάντοτε αφήνονται στην προσωπική χρίση του εκάστοτε προγραμματιστή. Όμως στη συνέχεια, θα αναλυθεί μια πλήρως αντικειμενική οπτική γωνία στο ζήτημα.

Δεν είναι επιθυμητό να γεμίσει με πολλά λόγια το pygame παράθυρο καθώς θα αφαιρείται σημαντική ορατότητα από τον χρήστη. Αυτό του αφαιρεί την δυνατότητα να αντιλαμβάνεται πως καταγράφονται τα δεδομένα και πως τα λαμβάνει ο YOLO ως input. Οπότε πρέπει να προσδιοριστεί η αξία της πληροφορίας που θα λαμβάνει στην προειδοποίηση χωρίς μακρυγορίες.

Με δεδομένο ότι εντοπίστηκε και μπροστινό όχημα και σηματοδότης, ανεξάρτητα από το χρώμα του σηματοδότη, είναι δεδομένο ότι το αυτοκίνητο πρέπει να σταματήσει ώστε να μην υπάρξει πιθανότητα τρακαρίσματος. Αυτό οδηγεί στο λογικό συμπέρασμα ότι η πληροφορία του χρώματος του σηματοδότη είναι περιττή πληροφορία ως ειδοποίηση. Οπότε επιλέγεται υπό αυτές τις συνθήκες η μόνη ειδοποίηση που θα εμφανιστεί να αφορά το μπροστινό όχημα και όχι τον σηματοδότη.

Τέλος, πρέπει να τονιστεί ότι η συγκεκριμένη απόφαση δεν εκμηδενίζει την πληροφορία του σηματοδότη έναντι του κοντινού οχήματος. Στην επόμενη ενότητα θα γίνει πλήρως αντιληπτή η αξία του ταυτόχρονου εντοπισμού τους. Όμως, όσον αφορά την ειδοποίηση του οδηγού καθώς κινείται στον δρόμο, πρέπει να επιλεχθεί ένας άμεσος τρόπος προειδοποίησης. Αυτό για να συμβεί χρειάζεται προφανώς να ζυγιστεί η αξία της πληροφορίας που ενδιαφέρει τον χρήστη περισσότερο την εκάστοτε στιγμή ώστε να μπορεί να την επεξεργαστεί γρήγορα και να λάβει μια απόφαση. Βέβαια στην επόμενη ενότητα που θα αναλυθεί το σύστημα ελέγχου δεν έχει τόσο καθοριστικό ρόλο η κρίση του χρήστη βάση της προειδοποίησης, καθώς θα παίρνει πρωτοβουλίες και μόνο του το όχημα.

### 4.1.3 Λανθασμένη Προσπέραση Σηματοδότη

Όπως μπορεί να γίνει αντιληπτό από τις εικόνες 4.1 και 4.2, κάτω δεξιά με μπλε γράμματα αναγράφεται ένας μετρητής ο οποίος ορίζεται ως η μεταβλητή *penalty*. Η συγκεκριμένη μεταβλητή πάνω στο λογισμικό που δημιουργήθηκε, έχει τρεις χρησιμότητες στο έργο:

1. Την καταγραφή της συχνότητας λανθασμένων προσπεράσεων σηματοδότη.
2. Την ύπαρξη ενός παραπάνω μέτρου σύγκρισης απόδοσης του δικτύου.
3. Την υπάρξη ενός παραπάνω μέτρου σύγκρισης μεταξύ άλλων δικτύων που εκπαιδεύτηκαν για τον ίδιο στόχο.

Το βασικό ερώτημα που μπορεί να προκύψει είναι το πότε ψεωρείται λανθασμένη η προσπέραση ενός σηματοδότη κοια κριτήρια αποτελούν αυτό το οδικό λάθος γενικότερα. Όπως και στον πραγματικό κόσμο, η απάντηση είναι η ίδια. Δηλαδή λανθασμένη προσπέραση είναι όταν προσπεραστεί ο σηματοδότης ενώ το χρώμα που δείχνει είναι είτε κόκκινο είτε κίτρινο. Αυτό συμβαίνει, διότι αυτά τα χρώματα αντιπροσωπεύουν ξεκάθαρα την άμεση διαταγή στο όχημα να λάβει τέλος η ταχύτητα του και να μείνει ακίνητο μέχρι να φανεί το πράσινο χρώμα στον σηματοδότη. Οπότε είναι ξεκάθαρη και αξιοχατάκριτη παρατήρηση που προκύπτει βάση τον κώδικα οδικής κυκλοφορίας. Εφόσον αναπτύσσεται ένα σύστημα αρκετά ικανό να εντοπίζει τους σηματοδότες και τα χρώματα τους, όταν σπατάλη πόρων το να μην μπορεί το σύστημα να ανιχνεύσει μια τέτοια περίπτωση κατά την οδήγηση στον δρόμο.

Αφού το *penalty* αφορά και την απόδοση του οχήματος κατά την οδήγηση και κυρίως την οπτική επίπληξη από το σύστημα, όταν μπορούσε κάποιος να δηλώσει ότι όταν λογική η προσμέτρηση της περίπτωσης να συμβεί μετωπική σύγκρουση με άλλο όχημα. Και αυτή η περίπτωση αποτελεί ένα βασικό κριτήριο του πόσο καλά κινείται ένα όχημα. Εφόσον το όχημα εκπαιδεύεται κυρίως μέσα από το σύστημα στο πότε οφείλει να φρενάρει, αποτελεί ένα αντικειμενικό μέτρο σύγκρισης. Όμως, λαμβάνοντας ως κριτήριο τον πραγματικό κόσμο, αυτό το γεγονός μπορεί να μην οφείλεται στον χρήστη/οδηγό αλλά στο λάθος ενός άλλου οδηγού. Είναι αρκετά πολύπλοκη, χρονοβόρα και δαπανηρή η προγραμματιστική διαδικασία που όταν χρειαστεί ώστε το σύστημα να μπορεί να αντιληφθεί πότε αντικειμενικά φταίει ο οδηγός και όχι ο εξωτερικός οδηγός για να τον κατηγορήσει. Επίσης, μια σύγκρουση με αμάξι είναι κάτι ξεκάθαρο ενώ το να περάσει με κόκκινο φανάρι το όχημα είναι κάτι που μπορεί να μην προσέξει ο οδηγός και να το καταλάβει μόνο από την ειδοποίηση του συστήματος. Οπότε, για όλους τους παραπάνω λόγους δεν λήφθηκε η περίπτωση ατυχήματος με άλλο όχημα στη συγκεκριμένη μεταβλητή *penalty* που αναλύεται.

Αυτό που πρέπει να απασχολήσει στη συνέχεια είναι το πως αξιοποιεί τις πληροφορίες το δίκτυο ώστε να κατανοήσει πότε έχει γίνει η λανθασμένη προσπέραση που έχει προαναφερθεί. Μέσα από το *script test.py* που αναπτύχθηκε για τις ανάγκες της εργασίας, ένα συγκεκριμένο αποσπασματικό κομμάτι κώδικα που φαίνεται στην εικόνα 4.3, έχει γραφτεί για την συγκεκριμένη επίτευξη αντίληψης του οχήματος. Ουσιαστικά ο *YOLO* κοιτάει τις εικόνες συνεχώς και ενημερώνει τον χρήστη για το χρώμα που έχει ο σηματοδότης. Η φυσιολογική διαδικασία που πρέπει να ακολουθηθεί είναι με την εξής σειρά:

- Εντοπισμός κίτρινου σηματοδότη (προαιρετικά)
- Εντοπισμός κόκκινου σηματοδότη
- Εντοπισμός πράσινου σηματοδότη
- Κανένας εντοπισμός σηματοδότη

```
if its_red:
    its_red=False
    penalty+=1
detected_colour_loop = "GO"
```

Σχήμα 4.3: Απόσπασμα κώδικα της περίπτωσης που δεν εντοπίστηκε κανένα αντικείμενο ενώ πρίν η ένδειξη έδειχνε κόκκινο ή κίτρινο σηματοδότη.

Το λάθος προκύπτει στο αν από εκεί που εντοπίζεται κόκκινος ή κίτρινος σηματοδότης, ξαφνικά δεν εντοπίζεται τίποτα. Δηλαδή, δεν “έίδε” ποτέ πράσινο χρώμα ο YOLO. Τότε σε αυτή την περίπτωση θεωρείται ότι συνέβη η λανθασμένη προσπέραση σηματοδότη και η μεταβλητή `penalty` αυξάνεται κατά ένα. Χρησιμοποιείται το ρήμα “θεωρείται” για λόγους που θα αναλυθούν στην συνέχεια καθώς δεν έχει πάντα έτσι η κατάσταση.

Δυστυχώς, όταν αναπτύσσεται ένα δίκτυο το ποσοστό ακρίβειας του δεν είναι ποτέ 100%. Πάντα υπάρχει πιθανότητα να συμβεί κάτι λάθος κατά τον εντοπισμό αντικειμένου σε μια εικόνα. Τα δύο βασικότερα λάθη είναι είτε να εντοπιστεί κάτι ενώ δεν υπάρχει τίποτα στην εικόνα ή το αντικείμενο που εντοπίστηκε να έχει καταταχθεί σε λάθος κλάση. Αυτές οι περιπτώσεις επηρεάζουν άμεσα την λειτουργικότητα του εντοπισμού λανθασμένης προσπέρασης σηματοδότη.

Αυτό συμβαίνει διότι ας πούμε, μπορεί ο YOLO να εντοπίσει λανθασμένα ένα κόκκινο σηματοδότη ενώ δεν υπάρχει και μετά να αντιληφθεί το λάθος και να μην υπάρχει στη συνέχεια καμία ένδειξη σηματοδότη. Όμως, αυτό αλγορίθμικά ισοδυναμεί με την περίπτωση του να εντοπίστηκε πραγματικά κόκκινος σηματοδότης και μετά να προσπεράστηκε και για αυτό έπειτα να μην εντοπίζει τίποτα ο αλγόριθμος. Οπότε, και στις δύο περιπτώσεις η μεταβλητή `penalty` θα αυξηθεί κατά ένα.

Άλλη μια παρόμοια περίπτωση είναι το να εντοπιστεί κόκκινος σηματοδότης ενώ στην πραγματικότητα στην εικόνα υπάρχει πράσινος σηματοδότης. Στην περίπτωση που μετά από το λάθος εντοπιστεί αμέσως το πράσινο χρώμα, φυσικά και δεν θα προμετρηθεί ως λάθος. Αυτό το σφάλμα δεν αιθέτησε την σειριακή διαδικασία εντοπισμού που τέθηκε ως ορθή παραπάνω. Όμως, αν μετά τον εντοπισμό του κόκκινου χρώματος έστω και στιγμιαία ο αλγόριθμος παραμένει ανίκανος να εντοπίσει το πράσινο χρώμα και δεν εντοπίσει τίποτα, τότε πάλι αυτή η περίπτωση συγκαταλέγεται στην ίδια με την πραγματική περίπτωση της προσπέρασης. Αυτό θα οδηγήσει ξανά στο να αυξηθεί το `penalty` κατά 1.

Πρέπει να ξεκαθαριστεί το πως οφείλεται μέσα από το σύστημα να λαμβάνονται υπόψιν αυτές οι πιθανές περιπτώσεις. Η άποψη που εκφράζεται μέσα από την

συγκεκριμένη εργασία είναι ότι δεν είναι λάθος σε αυτές τις περιπτώσεις να υπάρχει αύξηση της μεταβλητής `penalty`. Εκτός, από ειδοποίηση στον χρήστη είναι και μέτρο του πόσο καλά εκπαιδεύτηκε το δίκτυο και πόσο συχνά μπορεί να σφάλει σε πραγματικό χρόνο κάτι που δεν μπορεί να οριστεί εύκολα αλλιώς. Άρα, είναι καλό να συγκαταλέγονται μαζί όλες αυτές οι περιπτώσεις ώστε να γίνεται σε ένα πιο ευρύ φάσμα αντιληπτή η απόδοση του δικτύου κατά την οδήγηση. Το να διαχωριστούν αυτές οι πληροφορίες είναι ανούσια εκμετάλλευση πόρων, καθώς δεν θα υπάρξει καμία ουσιαστική πληροφορία που θα βοηθήσει τον οδηγό και τους προγραμματιστές που ασχολούνται με το λογισμικό, ώστε να το βελτιώσουν. Το κύριο ενδιαφέρον βρίσκεται γενικά στην έννοια του σφάλματος και όχι στην συγκατάταξη του.

## 4.2 Εφαρμογή Ελέγχου

Η εφαρμογή αυτή αφορά τις πρωτοβουλίες που μπορεί να πάρει το όχημα κατά την οδήγηση χωρίς καμία παρέμβαση από τον χρήστη/οδηγό. Είναι η κύρια εφαρμογή που καθιστά το σύστημα που αναπτύσσεται αυτοοδηγούμενο. Βάση των δεδομένων που παίρνει από τον αλγόριθμο YOLO κρίνει το ποια είναι η πιο σοφή απόφαση που πρέπει να παρθεί βάση πάντα του κώδικα οδικής κυκλοφορίας. Για αρχή, πρέπει να αναλυθούν οι αποφάσεις που παίρνει το αμάξι όταν εντοπίζει σηματοδότη. Μέσα από το απόσπασμα κώδικα 4.4 αναπτύσσονται αλγορίθμικά οι πρωτοβουλίες του οχήματος.

```
if its_red or ("Vehicle" in detected_colour and not back):
    control.brake=1.0
    control.throttle=0.0
    control.hand_brake = True
elif "Yellow" in detected_colour:
    if int(brk)!=1:
        brk+=0.05
        control.brake=brk
else:
    brk=0
```

Σχήμα 4.4: Απόσπασμα κώδικα ελέγχου του οχήματος.

Αν εντοπιστεί κόκκινος σηματοδότης από τον αλγόριθμο, τότε αυτό που συμβαίνει είναι το όχημα να φρενάρει ακαριαία. Αυτό συμβαίνει στον κώδικα με το να οριστεί η μεταβλητή του φρένου στη μέγιστη τιμή, η τιμή του γκαζιού στην ελάχιστη και να ενεργοποιηθεί επίσης και το χειρόφρενο του αμαξιού. Βέβαια η περίπτωση του εντοπισμού κίτρινου χρώματος στο φανάρι έχει πιο ομαλή διαδικασία φρεναρίσματος. Όπως φαίνεται και στον κώδικα 4.4 αν η τιμή της μεταβλητής που ορίζει το φρένο δεν έχει φτάσει στην μέγιστη τιμή, τότε χρήζει μιας πολύ μικρής σταδιακής αύξησης. Αυτό οδηγεί το αμάξι να ξεκινήσει την διαδικασία φρεναρίσματος άλλα με αργό ρυθμό ώστε ταυτόχρονα να συνεχίζει να πηγαίνει μπροστά και να σταματήσει τελείως όταν πια το χρώμα αλλάξει, όπως είναι φυσικό, από κίτρινο σε κόκκινο. Τέλος, όσον αφορά τους σηματοδότες, αν εντοπιστεί

πράσινο χρώμα ή απλά κανένα αντικείμενο στον χώρο, τότε στην ουσία το όχημα ελευθερώνεται και του επιτρέπεται η ελεύθερη κίνηση στον χώρο. Επίσης, η βιοηθητική μεταβλητή που χρησιμεύει στο αργό φρενάρισμα κατά τον εντοπισμό κίτρινου χρώματος, ορίζεται πάλι στο μηδέν ώστε να είναι σε ετοιμότητα για όταν ξανά εντοπιστεί κίτρινος σηματοδότης.

Ξεχωριστή κατηγορία στον τομέα της πρωτοβουλίας αποτελεί ο εντοπισμός κοντινού μπροστινού οχήματος. Εκεί πέρα τα πράγματα είναι πιο πολύπλοκα. Για αρχή όπως φαίνεται και στον κώδικα 4.4 η φυσιολογική διαδικασία που έπεται με τον εντοπισμό οχήματος μπροστά στον δρόμο είναι η ίδια με τον εντοπισμό του κόκκινου σηματοδότη. Δηλαδή, να σταματήσει ακαριαία το αμάξι ώστε να μην υπάρχει δυνατότητα σύγκρουσης και να ελευθερωθεί κινητικά μόνο όταν δεν θα εντοπίζεται αυτό το όχημα στον χώρο, πράγμα που σημαίνει ότι το αμάξι έχει το ελεύθερο να κινηθεί μπροστά. Όμως, στον πραγματικό κόσμο μια σύγκρουση μπορεί να προκύψει και από επικίνδυνη όπισθεν του μπροστινού αμαξιού. Επίσης, μπορεί να αποφευχθεί και σύγκρουση με την δυνατότητα της όπισθεν από το όχημα που ελέγχει το σύστημα. Οπότε, σε αντίθεση με όταν εντοπίζεται κόκκινος σηματοδότης, όταν το αμάξι εντοπίζει μπροστινό όχημα παραμένει με την δυνατότητα να κινηθεί προς τα πίσω, γεγονός που δεν το χαθιστά το όχημα πλήρως σταματημένο.

Ένα θέμα που πρέπει να υπολογιστεί ως παράμετρος στο θέμα του ελέγχου, είναι το πως θα συγχρονιστεί ομαλά ο εντοπισμός των οχημάτων με τον εντοπισμό των χρωμάτων του σηματοδότη. Για παράδειγμα, επειδή μπορεί να υπάρχει κίνηση στο δρόμο, το όχημα θα σταματήσει μπροστά από ένα άλλο, αλλά δεν θα εντοπίζει πρακτικά τον σηματοδότη και το χρώμα του επειδή θα βρίσκεται μακριά. Άξιο απορίας είναι αν αυτό θα έπρεπε να θεωρείται λειτουργικό λάθος. Δημιουργούνται ερωτήματα ανασφάλειας όπως το αν μπορεί να αποτελέσει λόγο να γίνει κάποιο οδικό λάθος από πρωτοβουλία του αμαξιού. Η απάντηση στους παραπάνω προβληματισμούς είναι καταφατικά ότι είναι ανούσιοι. Πρώτον, με δεδομένο ότι τα αμάξια λειτουργούν με τον αυτόνομο πιλότο που ορίζει το *Carla*, προφανώς κινούνται ομαλά στον δρόμο. Άρα, το πιο μπροστινό αμάξι όταν ο σηματοδότης δείξει πράσινο, θα ξεκινήσει να κινείται μπροστά μαζί με τα υπόλοιπα αμάξια, γεγονός που θα δημιουργήσει εν τέλει χώρο και στο αυτοκίνητο που είναι ελεγχόμενο από το σύστημα, για να κινηθεί χωρίς να υπάρξει κάποια ατυχής σύγκρουση.

Επίσης, ακόμα και αν επεκταθεί το θέμα στον πραγματικό κόσμο, δηλαδή έχοντας αμάξια απρόβλεπτης συμπεριφοράς, αν κανένα αμάξι δεν κινείται προφανώς οι πρωτοβουλίες που έχει προγραμματιστεί να παίρνει το αμάξι, βάση του συστήματος, παραμένουν σωστές. Είναι προτιμότερο να μείνει το αυτοκίνητο ακίνητο από το να του δοιθεί το ελεύθερο να προχωρήσει ευθεία, ενώ βλέπει ότι μπροστά υπάρχει όχημα και η πιθανότητα σύγκρουσης είναι μεγάλη, ακόμα και αν είναι ακίνητο για αρκετό χρονικό διάστημα. Αυτά τα θέματα δυστυχώς δεν μπορεί να τα λύσει η τεχνητή νοημοσύνη και απασχολούν ξεκάθαρα την ανθρωπολογικό κομμάτι της οδήγησης και μόνο.

### 4.3 Εφαρμογή Πλαισίου Αντικειμένων

Όπως φαίνεται και από τις εικόνες 4.1 και 4.2 υπάρχει ένα πλαίσιο το οποίο ουσιαστικά αναλαμβάνει τον ρόλο να εντοπίσει με οπτικό τρόπο το τι αντιλαμβάνεται ως

αντικείμενο ο YOLO κατά την οδήγηση. Όπως έχει τονιστεί και σε προηγούμενη ενότητα 3.4, εκτός από την κλάση που ανήκει ένα αντικείμενο, ο YOLO προβλέπει και μεταβλητές που έχουν να κάνουν σχέση με το πλαίσιο στο οποίο εντοπίζεται μέσα στην εικόνα το αντικείμενο που ανιχνεύτηκε. Όμως, ενώ το Darknet έχει αυτόματα αυτή την λειτουργία, στο σύστημα οδήγησης που αναπτύσσεται μέσω του Carla, δεν μπορεί να μεταφερθεί αυτούσια, οπότε πρέπει να οριστεί εκ νέου μέσω του pygame.

Αρχικά οι σχετικές μεταβλητές που αφοράνε την δημιουργία ενός πλαισίου ορίζονται στο μηδέν και παραμένουν πάντα έτσι μέχρι να εντοπιστεί κάποιο αντικείμενο στον χώρο. Αυτό σημαίνει ότι όταν δεν εντοπίζεται τίποτα “δημιουργείται” ένα πλαίσιο με 0 ύψος και 0 πλάτος που στην ουσία δεν υπάρχει και δεν φαίνεται τίποτα οπτικά στον χρήστη. Στη συνέχεια, όταν εντοπίζεται ένα όχημα στις σχετικές μεταβλητές εκχωρούνται οι τιμές που υπολογίστηκαν από τον αλγόριθμο YOLO. Έπειτα, όπως αποφαίνεται και από τον κώδικα 4.5, ξεκινάει η διαδικασία εμφάνισης του πλαισίου αντικειμένου. Επειδή το pygame ξεκινάει την διαδικασία δημιουργίας του από το σημείο που βρίσκεται πάνω αριστερά, πρέπει πρακτικά να αξιοποιηθεί η γνώση για την τοποθεσία του κέντρου του πλαισίου. Με λίγα λόγια, για να μεταφερθεί το κέντρο στο πάνω αριστερά άκρο πρέπει να υπάρξει στον x άξονα αρνητική μετατόπιση κατά το μισό του πλάτους του πλαισίου και στον y άξονα αρνητική μετατόπιση κατά το μισό του ύψους του πλαισίου. Το σύστημα το λαμβάνει αυτό υπόψιν και έτσι πορεύεται με την εμφάνιση του πλαισίου στον χώρο.

```
pygame.draw.rect(surface, (r,g,b), pygame.Rect(x_left-(width//2), y_top-(height//2), width, height),2)
pygame.draw.rect(surface, (r,g,b), pygame.Rect(x_left2-(width2//2), y_top2-(height2//2), width2, height2),2)
basicfont = pygame.font.SysFont(None, 80)
text_detected = basicfont.render(detected_colour, True, (r,g,b))
textrec = text_detected.get_rect()
textrec.top = surface.get_rect().top
textrec.midtop = surface.get_rect().midtop
surface.blit(text_detected, textrec)
```

Σχήμα 4.5: Απόσπασμα κώδικα δημιουργίας πλαισίων αντικειμένων.

Ένα σημαντικό θέμα που πάλι μπορεί να γίνει αντιληπτό από τις 2 πρώτες γραμμές στο απόσπασμα κώδικα 4.5, είναι ότι υπάρχουν 2 εντολές για δημιουργία πλαισίου. Αυτό συμβαίνει επειδή πρακτικά ο αλγόριθμος YOLO έχει εκπαιδευτεί ώστε να αντιλαμβάνεται δύο αντικείμενα στον χώρο ταυτόχρονα. Οπότε αν εντοπίσει δύο αντικείμενα τότε πρέπει να ειναι και δύο τα πλαίσια που θα εμφανιστούν στο παράθυρο του pygame. Οι σχετικές μεταβλητές που αφοράνε το δεύτερο αντικείμενο προς εντόπιση αρχικοποιούνται στο μηδέν οπότε όσο δεν εντοπίζεται δεύτερο αντικείμενο στον χώρο τότε δεν εμφανίζεται και κανένα πλαίσιο. Όταν εμφανιστεί, η διαδικασία είναι η ίδια ακριβώς που αναλύθηκε παραπάνω. Αξίζει επίσης να αναφερθεί ότι όταν το σύστημα βρεθεί στην περίπτωση που δεν εντοπίζεται κανένα αντικείμενο στον χώρο μετά από περίπτωση εμφάνισης, τότε όλες οι μεταβλητές που αφοράνε το πλαίσιο πρώτου και δεύτερου αντικειμένου επιστρέφουν πάλι στο μηδέν. Αυτό έχει ως λογικό επακόλουθο στην ουσία όλα τα πλαίσια να εξαφανίζονται. Τα χρώματα που τους έχουν οριστεί είναι ανάλογα το τι εντοπίζουν. Αν εντοπίζεται όχημα ή κόκκινος σηματοδότης το χρώμα του αντίστοιχου πλαισίου γίνεται κόκκινο και ακολουθείται το ίδιο μοτίβο και για τα άλλα χρώματα.

## 4.4 Εφαρμογή Προσπέλασης Δεδομένων

Αυτή είναι η τελευταία εφαρμογή που αναπτύχθηκε για τις ανάγκες της εργασίας. Δεν αποτελεί εφαρμογή που επηρεάζει στην πράξη την αυτόνομη οδήγηση αλλά μόνο την ανάλυση των δεδομένων που προ υπήρξαν ώστε να πραγματοποιηθεί η δυνατότητα αυτονόμησης ενός οχήματος. Επίσης αναλαμβάνει η συγκεκριμένη εφαρμογή σημαντικό ρόλο στο να αξιολογήσει τα αποτελέσματα που προκύπτουν μέσα από τον YOLO. Αυτό για να συμβεί χρειάζεται να υπάρξει προσπέλαση μέσα από όλα τα δεδομένα για λόγους καταμέτρησης ώστε να γίνει πιο αντιληπτό το πως εκπαιδεύτηκε το δίκτυο αλλά και για να προκύψει μέσα από δεδομένα δοκιμής τα οποία όταν εξεταστούν ένα προς ένα, το στατιστικό πόρισμα ακρίβειας κατά τον εντοπισμό αντικειμένων στον χώρο. Η εφαρμογή όπως γίνεται αντιληπτό έχει δύο διακλαδώσεις οι οποίες είναι:

1. Προσπέλαση καταμέτρησης του **Dataset**.
2. Προσπέλαση αποδοτικότητας σε **Test Dataset**.

### 4.4.1 Προσπέλαση Καταμέτρησης του **Dataset**

Αυτή η λειτουργία έχει σκοπό να απαριθμήσει την συχνότητα εμφανίσεων των κλάσεων μέσα σε ένα ενιαίο σύνολο το οποίο μπορεί να είναι είτε για εκπαίδευση είτε για δοκιμή. Κάνει πιο ξεκαθάρα τα δεδομένα που χρησιμοποιούνται, την ποιότητα τους και βοηθούν και στην κρίση του αν το **Dataset** που εξετάζεται χρήζει κάποιας αλλαγής. Ουσιαστικά στον πίνακα 3.2 χρησιμοποιήθηκε η συγκεκριμένη προσπέλαση ώστε να υπάρξουν τα συγκεκριμένα αριθμητικά αποτελέσματα.

Όπως φαίνεται και στο παρακάτω απόσπασμα κώδικα 4.6 που δημιουργήθηκε σε `ξεχωριστό script access.py`, η διαδικασία που ακολουθείται για να επιτευχθεί η προσπέλαση είναι αρχικά να οριστεί το `path` όπου βρίσκονται οι φωτογραφίες και τα αντίστοιχα `txt` που φέρουν. Στη συνέχεια πραγματοποιείται η προσπέλαση μέσα σε `exception` ώστε αν δεν είναι αριθμημένα με την σειρά τα δεδομένα να μην τερματίσει το πρόγραμμα. Εφόσον με την χρήση της μεταβλητής `i` ορίζεται ένας σειριακός τρόπος να εξετάζονται τα αρχεία, τότε πρέπει να οριστεί η διαδικασία ελέγχου.

Η διαδικασία ελέγχου γίνεται μέσω της μεταβλητής `line` που της εκχωρείται κάθε φορά, η πληροφορία του αρχείου που εξετάζεται. Αν η μεταβλητή είναι κενή τότε η αντίστοιχη φωτογραφία δεν απεικονίζει τίποτα. Αν δεν είναι, τότε εμπεριέχει πληροφορίες και η πρώτη είναι ο αριθμός της κλάσης που εντοπίζεται. Ο εκάστοτε αριθμός φωτογραφίας που αφορά το `ti` θέλουμε να υπολογίσουμε, όπως το πόσοι κίτρινοι σηματοδότες υπάρχουν, αποθηκεύεται κάθε φορά στην λίστα `L`. Στο τέλος της επανάληψης `for` περιέχει τον αριθμό κάθε φωτογραφίας με την κλάση που ορίσαμε, και το μέγεθος της λίστας `for` είναι η πληροφορία της συχνότητας της συγκεκριμένης κλάσης. Στο παράδειγμα της εικόνας 4.6 γίνεται η καταμέτρηση το πόσες φορές εμφανίζεται κοντινό όχημα μέσα στα δεδομένα εκπαίδευσης.

```

line=""
L=[]
path='C:/Yolo_v4/darknet/build/darknet/x64/data/objs/'
for i in range(1,7290):
    try:
        with open(f'{path}Photo{i}.txt') as f:
            line = f.read()
            if len(line)!=0 and line[0]=='3':
                L.append(i)
    except FileNotFoundError:
        continue

print(f"The vehicles in front are {len(L)}")

```

Σχήμα 4.6: Απόσπασμα κώδικα προσπέλασης δεδομένων από access.py

#### 4.4.2 Προσπέλαση Αποδοτικότητας σε Test Dataset

Αυτή η λειτουργία έχει ως απώτερο σκοπό να εξμεταλλευτεί την διαδικασία προσπέλασης των αρχείων για κάτι παραπάνω από μια καταμέτρηση. Ουσιαστικά στοχεύει να κρίνει βάση του αποτελέσματος που θα εξάγει, το πόσο αποδοτικό είναι το εκάστοτε εκπαιδευμένο δίκτυο που εξετάζεται. Ως γνωστόν σε οποιονδήποτε τομέα του Machine Learning για να οριστεί η ακρίβεια της εκπαίδευσης χρειάζονται δεδομένα δοκιμής που δεν δόθηκαν ποτέ ως πληροφορία στο δίκτυο κατά την εκπαίδευση. Μόνο με αυτό τον τρόπο είναι εφικτό να εξεταστεί αντικειμενικά η απόδοση του σε άγνωστες περιπτώσεις που δεν έχει προμελετήσει και πρέπει εκείνη την στιγμή να τις επεξεργαστεί για πρώτη φορά.

Μέσα από το Darknet φορτώνεται στο πρόγραμμα το δίκτυο που εκπαιδεύτηκε. Αφού οριστεί το path που βρίσκεται το test dataset τότε ξεκινάει πάλι η διαδικασία προσπέλασης. Μπορεί η φιλοσοφία που ακολουθείται να είναι παρόμοια με αυτή στο απόσπασμα κώδικα 4.6 άλλα στη συγκεκριμένη περίπτωση έχουν ενδιαφέρονται οι φωτογραφίες καθαυτές ως αρχείο. Χρειάζεται να υπάρχουν σε ξεχωριστό φάκελο οι φωτογραφίες και τα txt αρχεία τους. Ο YOLO μέσα στον φάκελο των φωτογραφιών θα παράγει αυτόματα καινούργια txt αρχεία τα οποία θα περιέχουν τις πληροφορίες που προέβλεψε. Με την σειρά τους όμως αυτά τα αρχεία txt, θα σβήσουν τα προηγούμενα αν είναι στον ίδιο φάκελο επειδή θα έχουν ακριβώς το ίδιο όνομα. Οπότε για να αποφευχθεί αυτή η σύγχυση πρέπει να βρίσκονται σε άλλους φακέλους.

Αφού λυθούν αυτά τα τεχνικά ζητήματα τότε ξεκινάει η ουσία του θέματος. Σειριακά για κάθε φωτογραφία παράγεται το αντίστοιχο αρχείο που εμπεριέχει τις πληροφορίες πρόβλεψης από τον YOLO. Στη συνέχεια αυτά τα αρχεία, συγχρίνονται με εκείνα που αποθηκεύτηκαν από την διαδικασία προσθήκης ετικετών. Αν τα αντίστοιχα αρχεία με αυτά που έχουν ετικέτα, εμπεριέχουν την ίδια χλάση ως πληροφορία τότε ουσιαστικά αυτό μετράει ως μια σωστή πρόβλεψη και καταμετρείται σε σχετική μεταβλητή. Αν είναι λάθος απλά η διαδικασία συνεχίζει ως έχει. Στην περίπτωση που δεν υπάρχει ετικέτα σε μια φωτογραφία και το αρχείο txt της είναι κενό, τότε σωστή πρόβλεψη θεωρείται μόνο αν και το αρχείο πρόβλεψης που δημιουργηθεί είναι επίσης κενό. Όταν αυτή η επαναληπτική διαδικασία λάβει τέλος,

Θα έχει υπολογιστεί το πόσες φορές ο YOLO έκανε σωστή πρόβλεψη. Με μια διαίρεση με τον συνολικό αριθμό των δεδομένων, παράγεται εν τέλει το επιθυμητό ποσοστό ακρίβειας.

## Κεφάλαιο 5

# Πειράματα και Αποτελέσματα

Σε αυτό το κεφάλαιο θα αναλυθούν τα πειράματα που έγιναν για την μελέτη της έκβασης που έχει η αυτόνομη οδήγηση. Θα συγχριθούν δίκτυα YOLO με διαφορετικό τρόπο εκπαίδευσης μεταξύ τους, ώστε να χριθεί ποιος τρόπος εκπαίδευσης είναι πιο αποτελεσματικός. Επίσης, θα επηρεαστεί και το περιβάλλον στο οποίο βρίσκεται το όχημα ώστε να φανεί σε πραγματικό χρόνο κατά την οδήγηση, ποιες θα είναι οι αποκρίσεις του οχήματος και αν αυτές κρίνονται ορθές. Επιπρόσθετα, θα παρατεθούν στατιστικά αποτελέσματα ακρίβειας, διάγραμμα σχετικά με τα σφάλματα (μεταβλητή *penalty*) και λογικά συμπεράσματα βάση των αποτελεσμάτων που θα προκύψουν.

## 5.1 Πειράματα

Η ενότητα αυτή θα καλύψει όλες τις δοκιμές και όλα τα πειραματικά κομμάτια που έλαβαν χώρα μέσα στο γραφικό περιβάλλον του Carla με την χρήση του αλγορίθμου YOLO. Θα υπάρξει πλήρης ανάλυση της τροποποίησης του δικτύου και των αλλαγών στο περιβάλλον οι οποίες αποτελούν βασικό κριτήριο με τα αποτελέσματα τους για το γενικό συμπέρασμα της απόδοσης της αυτονόμησης των οχημάτων.

### 5.1.1 Εκπαίδευση με διαφορετικό **Dataset**

Είναι λογικό για να υπάρχει μια πιο σφαιρική άποψη για το πόσο λειτουργικός και αποτελεσματικός είναι ο YOLO κατά την οδήγηση, να ληφθεί υπόψιν και η εκπαίδευση του ίδιου δικτύου με εξολοκλήρου διαφορετικά δεδομένα εκπαίδευσης. Σε όλα τα προηγούμενα κεφάλαια όταν αναφέρονταν τα δεδομένα εκπαίδευσης αναφέρονταν μόνο στα δεδομένα του πίνακα 3.2 του 3 κεφαλαίου. Από εδώ και στο εξής τα συγκεκριμένα δεδομένα θα αναφέρονται ως **Dataset 1**. Τα καινούργια δεδομένα που χρειάστηκαν να υπάρξουν για μια δεύτερη εκπαίδευση του δικτύου θα αναφέρονται ως **Dataset 2**.

Η διαδικασία που υπήρξε για την δημιουργία του **Dataset 2** και την εκπαίδευση του δικτύου είναι ακριβώς η ίδια που αναλύθηκε και στην 3.3 ενότητα. Προφανώς, εφόσον γίνεται αναφορά για νέα δεδομένα εκπαίδευσης, η αναδιαμόρφωση και η συχνότητα εμφάνισης των κλάσεων είναι διαφορετική. Ο νέος πίνακας ανάλυσης για το **Dataset 2** είναι ο 5.1.

Ένας σημαντικός λόγος που έπρεπε να υπάρξουν 2 δεδομένα εκπαίδευσης εκτός από την εκπαίδευση 2 δικτύων, είναι και η ξεχωριστή αξιολόγηση της απόδοσης τους. Το πρόβλημα των ελλειπόντων πόρων του τοπικού υπολογιστή παραμένει

Διαμόρφωση του Dataset 2		
Κλάση	Συχνότητα	Ποσοστό ως προς το σύνολο
Red	328	4.5%
Green	312	4.2%
Yellow	324	4.4%
Vehicle_InFront	393	5.4%
-	5918	81.3%

Πίνακας 5.1: Ποσοτική ανάλυση του Dataset 2

πάντα σταυθερός περιοριστικός παράγοντας. Οπότε, για αυτό τον λόγο κρίθηκε σοφό τα δύο δεδομένα εκπαίδευσης στον τομέα των συχνοτήτων των κλάσεων και σε μέγεθος να είναι περίπου όμοια. Όμως, αυτό δεν καθιστά τα δεδομένα του Dataset 1 ανίκανα να αποτελέσουν τον ρόλο των δεδομένων δοκιμής ως προς το δίκτυο που εκπαιδεύτηκε με το Dataset 2 και αντίστροφα. Ουσιαστικά τα δεδομένα εκπαίδευσης του ενός δικτύου αποτελούν στην πράξη άγνωστα δεδομένα για το άλλο δίκτυο. Είναι μια πρακτική εκμετάλλευση πόρων καθώς μέσα από αυτή την διαδικασία θα αξιολογηθούν αντικείμενα 2 δίκτυα YOLO και μετέπειτα θα μπορούν αντικείμενα να συγχριθούν ως προς την ακρίβεια τους.

### 5.1.2 Αλλαγές στις Υπερπαραμέτρους

Ένας άλλος τρόπος με τον οποίο μπορεί να αξιολογηθούν δύο δίκτυα μεταξύ τους είναι αν εκπαιδεύονται με τα ίδια δεδομένα εκπαίδευσης αλλα με διαφορετικές παραμέτρους. Οπότε, για την ανάγκη αυτού του πειράματος εκπαιδεύτηκε ένα ακόμα δίκτυο YOLO το οποίο χρησιμοποιεί για δεδομένα εκπαίδευσης το Dataset 1 και δεν έχει υποστεί αλλαγές στις υπερπαραμέτρους που δεν θα αναφερθούν. Οι υπερπαράμετροι που τροποποιήθηκαν για να εξεταστεί πως επηρεάζεται η απόδοση του YOLO είναι οι εξής:

1. batch size
2. learning\_rate
3. momentum

Το batch size δοκιμάστηκε να αυξηθεί. Μέσα από έρευνα [57] που εφαρμόστηκε σε συγκεκριμένα μοντέλα εκπαίδευσης, φαίνεται να είναι μια πολύ πρακτική λύση η αύξηση της συγκεκριμένης παραμέτρου. Ουσιαστικά αυξάνεται ο αριθμός της πληροφορίας που παίρνει το δίκτυο κάθε φορά κατά την εκπαίδευση και με τους κατάλληλους πόρους αυτό το γεγονός φτάνει πιο γρήγορα το δίκτυο στο στάδιο της αξιοπρεπής ακρίβειας. “Όμως, μέσα από άλλες έρευνες [25] γίνεται αισθητό ότι μπορεί μια τέτοια αλλαγή να αποβεί και μοιραία καθώς δεν οδηγεί πάντα στην βελτίωση του δικτύου. Στην περίπτωση του YOLO είναι δύσκολο να οριστεί από την αρχή αν ανήκει στην κατηγορία που η αύξηση του batch size θα φέρει καλύτερα αποτελέσματα. Για αυτό κρίθηκε αναγκαίο μέσα από αυτή την εργασία να απαντηθεί αυτό το ερώτημα.

Όσον αφορά το **learning\_rate** είναι ουσιαστικά το πόσο γρήγορα το δίκτυο θα προσαρμοστεί πάνω στα δεδομένα. Γενικά συνίσταται ένας αριθμός όχι πολύ μικρός αλλά όχι και πολύ μεγάλος [25]. Βάση αυτού δοκιμάστηκε μια αύξηση μήπως η συγκεκριμένη τιμή αποφέρει πιο γρήγορα και ορθά αποτελέσματα υπό των ίδιο αριθμού επαναλήψεων.

Μια μικρή τιμή στο **momentum** μπορεί να επιβραδύνει την εκπαίδευση του συστήματος [71]. Αυτό δεν είναι καθόλου επιθυμητό όταν δεν θέλουμε να είναι χρονοβόρα η εκπαίδευση. Άρα συνεχίζει να διατηρείται σε υψηλή τιμή. Όμως, μια υπερβολικά μεγάλη τιμή προκαλεί την σύγκλιση του δίκτυου να συμβεί γρήγορα. Αυτό το γεγονός καλύτερα από αποφευχθεί γιατί ίσως δεν εκπαιδευτεί αποτελεσματικά το δίκτυο. Προτιμήθηκε λοιπόν βάση των παραπάνω, να χαμηλωθεί ελάχιστα η τιμή της συγκεκριμένης υπερπαραμέτρου.

Φαίνονται υπογραμμισμένες οι αριθμητικές αλλαγές στις υπερπαραμέτρους του YOLO στον παρακάτω πίνακα 5.2 μαζί με τις αντίστοιχες ονομασίες. Επίσης, στην προηγούμενη υποενότητα 5.1.1 το δίκτυο που εκπαιδεύτηκε με το Dataset 2, υπέστει τις ίδιες τροποποιήσεις εκτός του **momentum**. Αυτό συνέβη ώστε να υπάρξει πιο πιθανή διαφορά στο τι αποτελέσματα θα προκύψουν μέσα από όλες αυτές τις αλλαγές στην εκπαίδευση.

Διαμόρφωση Δικτύου Εκπαίδευσης			
<u>batch</u>	128	exposure	1.5
subdivisions	128	hue	0.1
width	224	mosaic	1
height	224	<u>learning_rate</u>	<u>0.005</u>
channels	3	burn_in	1000
<u>momentum</u>	0.8	max_batches	8000
decay	0.0005	policy	steps
angle	0	steps	6400,7200
saturation	1.5	scales	0.1,0.1

Πίνακας 5.2: Τιμές διαμόρφωσης του YOLO από τροποποιημένο αρχείο υπερπαραμέτρων.

### 5.1.3 Αλλαγές στις Συνθήκες Περιβάλλοντος

Για να μπορεί να αποτυπωθεί ένα ακόμα πιο σφαιρικό συμπέρασμα για το πως αποδίδει μια μορφή τεχνητής νοημοσύνης στην αυτόνομη οδήγηση, πρέπει οι συνθήκες περιβάλλοντος να είναι απρόβλεπτες. Αυτό εξυπηρετεί στο να εξεταστεί αντικειμενικά ένα λογισμικό αν αποδίδει σε όλες τις συνθήκες με την ίδια ακρίβεια. Η φύση της οδήγησης εξ ορισμού είναι κάτι που οδηγεί σιγά στο απρόβλεπτο και αν δεν μπορεί ένα σύστημα αυτοματισμού να αντιδράσει επιθυμητά σε τέτοιες καταστάσεις δεν είναι καθόλου λειτουργικό.

Όμως, πρέπει να απαντηθεί το τι ορίζεται ως συνθήκη περιβάλλοντος και μπορεί να τροποποιηθεί ως πείραμα στην περίπτωση της συγκεκριμένης εργασίας. Το δίκτυο YOLO βάση των Dataset 1 και Dataset 2 έχει ήδη εκπαιδευτεί σε αρκετές

καιρικές συνθήκες που αποτελούν την βασική αλλαγή στο περιβάλλον οδήγησης. Η απάντηση στο παραπάνω ερώτημα είναι ουσιαστικά οι **actors** που υπάρχουν στον χώρο και συγχεκριμένα οι πεζοί και τα οχήματα.

Είναι σχεδόν βέβαιο ότι η απόδοση ενός συστήματος εντοπισμού αντικειμένου επηρεάζεται άμεσα από οτιδήποτε κινείται μέσα στον χώρο. Μπορεί κάποιο αντικείμενο να μην ανήκει καν στις κλάσεις εκπαίδευσης όμως το δίκτυο να κάνει το λάθος στιγμιαία να συμπεριλάβει αυτό το αντικείμενο σαν να ανήκει σε μια από τις κλάσεις. Αυτό μπορεί να συμβεί για διάφορους λόγους. Ένας από αυτούς είναι για παράδειγμα ένα πεζός από μια συγκεκριμένη απόσταση να καταλαμβάνει τον ίδιο χώρο στο πλαίσιο με έναν σηματοδότη. Αν ο πεζός είναι ταυτόχρονα ντυμένος στα κόκκινα μπορεί να θεωρηθεί ως κόκκινος σηματοδότης. Για να εντοπιστούν τέτοιες περιπτώσεις ως σφάλματα και μάλιστα η συχνότητα που έχουν, πρέπει να αυξηθεί και ο αριθμός των οχημάτων και ο αριθμός των πεζών που συνυπάρχουν μαζί με το ελεγχόμενο όχημα στον εικονικό κόσμο του **Carla**. Όσο περισσότεροι είναι οι **actors** τόσο περισσότερο εκτίθεται το σύστημα στην πιθανότητα σφάλματος και μπορεί να κριθεί η απόδοση του σε πραγματικό χρόνο.

Το πείραμα που θα γίνει βάση των παραπάνω θα λάβει υπόψιν του 10 επαναλήψεις προσομοίωσης περιβάλλοντος μέσα στον γραφικό κόσμο οι οποίες θα διαρκούν 15 λεπτά η κάθε μια. Ανά επανάληψη θα αυξάνεται συνεχώς ο αριθμός των **actors** δηλαδή των πεζών και των οχημάτων και θα καταγράφεται κάθε φορά η τελική τιμή που θα έχει η μεταβλητή **penalty**. Όπως έχει ήδη ξεκαθαριστεί από την υποενότητα 4.1.3, η συγκεκριμένη μεταβλητή είναι η πλέον κατάλληλη βάση τον αλγορίθμικό της σκοπό για τους σκοπούς που έχουν αναφαρθεί σε αυτή την υποενότητα. Μέσω εκείνης κρίνεται το πως αποδίδει η διαδικασία εντοπισμού του **YOLO** σε πραγματικό χρόνο αφού κάθε φορά προσμετρά την περίπτωση του λανθασμένου εντοπισμού.

#### 5.1.4 Δοκιμές του **YOLO** υπό διάφορες περιπτώσεις

Καίριας σημασίας εκτός την γενική ακρίβεια απόδοσης ενός δικτύου και της ομαλής πορείας του σε πραγματικό χρόνο, είναι και τα αποτελέσματα που εξάγει σε ειδικές περιπτώσεις. Κάποια σενάρια είναι μικρή η πιθανότητα να εμφανιστούν οπότε υπάρχει η τάση να αγνοούνται. Ένα θετικό που προσφέρει το να πραγματοποιούνται οι δοκιμές της αυτόνομης οδήγησης σε γραφικό περιβάλλον, είναι ότι με λίγη τροποποίηση στο λογισμικό είναι εφικτό εύκολα να δημιουργηθεί οποιαδήποτε ακραία συνθήκη ειναι επιθυμητή να συμβεί κατά την διάρκεια της οδήγησης ώστε να δοκιμαστεί το όχημα στο τι απόφαση θα λάβει.

Τα πειράματα που έχει ενδιαφέρον να γίνουν ώστε να φανεί το τι αποτελέσματα θα παράγει ο **YOLO** αφορούν κυρίως την ταχύτητα του οχήματος. Πρέπει να αποσαφηνιστεί πως επηρεάζει η ταχύτητα την διαδικασία του εντοπισμού ενός αντικειμένου και την διαδικασία πρωτοβουλίας πάνω στον εντοπισμό που γίνεται. Για συμβεί αυτό πρέπει το ίδιο σενάριο να διεξαχθεί 3 φορές υπό τις εξής συνθήκες:

- Υπερβολικά υψηλή ταχύτητα
- Υπερβολικά χαμηλή ταχύτητα
- Φυσιολογική ταχύτητα

Ουσιαστικά αυτό που είναι επιθυμητό είναι να εξεταστεί η δυνατότητα του YOLO να μπορεί να προσαρμοστεί πάνω σε ακραίες καταστάσεις. Μπορεί να είναι ελάχιστες οι περιπτώσεις που θα χληθεί να πάρει απόφαση υπό τέτοιες συγκυρίες, όμως το να έχει την δυνατότητα να μην σφάλει ακόμα και εκεί, δημιουργεί ένα μεγαλύτερο αίσθημα ασφάλειας στο να αποφευχθεί κάποιο μελλοντικό ατύχημα κατά την οδήγηση.

Επιπρόσθετα, πρέπει να δοκιμαστούν και ακραίες καταστάσεις στις οποίες υπάρχει μεγάλη πιθανότητα να υπάρξει λάθος. Αυτό γίνεται για να δοκιμαστεί στα άκρα η αποδοτικότητα του δικτύου. Οι πειραματικές περιπτώσεις που έγιναν για αυτή την εργασία είναι:

- Αναμομή πεζού να διασχύσει τον δρόμο.
- Επιτηδεύμενη ύπαρξη 2 αντικειμένων στον χώρο.

Η αναμομή πεζού στον οπτικό χώρο του αμαξιού είναι ένα σενάριο το οποίο παίρνει αρκετή ώρα. Αναγκάζει τον YOLO πάρα πολλές φορές να εξετάζει αν το σχήμα του πεζού και η πληροφορία των χρωμάτων του, αντιστοιχεί σε κάποια από τις εκπαιδευμένες κλάσεις. Με την διαρκή κίνηση του πεζού είναι πολύ πιθανό τουλάχιστον μια φορά να γίνει λάθος στιγμιαίος εντοπισμός στην θεωρία. Όμως, επειδή η θεωρία δεν αρκεί, το πείραμα αυτό γίνεται για να φανεί στην πράξη αν θα συμβεί κάτι τέτοιο.

Η επιτηδεύμενη ύπαρξη δύο αντικειμένων στον χώρο, γίνεται κυρίως για να φανεί αν το δίκτυο έχει αντιληφθεί ότι προγραμματίστηκε με τον σκοπό να μην εστιάζει μόνο σε ένα αντικείμενο. Επίσης, βάση το Dataset 1 είναι μόνο 84 τα σενάρια που συνυπάρχουν δύο αντικείμενα σε μια φωτογραφία. Οπότε, είναι εύλογο να χρειάζεται να εξεταστεί το τι αποτελέσματα θα υπάρξουν υπό όλες αυτές τις συνθήκες.

## 5.2 Αποτελέσματα

Αυτή η ενότητα θα απασχοληθεί με την παράθεση των αποτελεσμάτων που προέκυψαν από τα παραπάνω πειράματα καθώς και την σημασία που έχουν πάνω στην αυτόνομη οδήγηση. Θα εστιάσει στην ακρίβεια του YOLO και στο που υστερεί μέσα από την εκπαίδευση των δικτύων που πραγματοποιήθηκε. Για λόγους διευκόλυνσης από εδώ και πέρα, το δίκτυο YOLO που εκπαιδεύτηκε με το Dataset 1 και και με τις υπεραπαμέτρους του πίνακα 3.1 θα ονομάζεται Δίκτυο 1. Το δίκτυο που εκπαιδεύτηκε με το Dataset 2 θα ονομάζεται Δίκτυο 2. Τέλος, το δίκτυο που εκπαιδεύτηκε με το Dataset 1 και με τις τροποποιημένες υπεραπαμέτρους του πίνακα 5.2 θα ονομάζεται Δίκτυο 3.

### 5.2.1 Στατιστικά Αποτελέσματα και Συγκρίσεις

Μέσα από την πειραματική εκπαίδευση δικτύων που αναλύθηκε στις υποενότητες 5.1.1 και 5.1.2, προέκυψαν το Δίκτυο 2 και Δίκτυο 3 αντίστοιχα. Ουσιαστικά το βασικό μοντέλο που εκπαιδεύτηκε για αυτή την εργασία είναι το Δίκτυο 1 και τα άλλα δύο αποτελούν παραλλαγές του. Παρακάτω θα φανούν οι διαφορές που προκύπτουν μέσα από την εκπαίδευση τους.

Τηλογίστηκε η γενική ακρίβεια των δικτύων μέσα από το script access.py που αναφέρεται στην ενότητα 4.4. Η βασική λογική που ακολουθείται είναι να χρησιμοποιούνται ως δεδομένα ελέγχου το dataset που δεν εκπαιδεύτηκε με αυτό το εκάστοτε δίκτυο. Οπότε, προκύπτει αυτός ο πίνακας ακρίβειας:

Ον/σια	Γενική Ακρίβεια
Δίκτυο 1	94.61%
Δίκτυο 2	94.07%
Δίκτυο 3	95.84%

Πίνακας 5.3: Γενική ακρίβεια των 3 δικτύων.

Όπως φαίνεται η γενική απόδοση και ακρίβεια του δικτυού δεν μεταβάλλεται ιδιαίτερα με όποιον τρόπο και αν εκπαιδευτεί. Διατηρείται υψηλά με πάνω από 94% ακρίβεια σε κάθε περίπτωση. Όμως όπως είναι οφθαλμοφανή, οι τροποποιήσεις που υπήρξαν για την δημιουργία του Δικτύου 3 έφεραν εν τέλει καλύτερα αποτελέσματα από ότι το πρότυπο Δίκτυο 1. Επίσης, οι αλλαγές στις παραμέτρους σε συνδυασμό με την αλλαγή των δεδομένων εκπαίδευσης φαίνεται να επηρέασε αρνητικά την τελική ακρίβεια που εν τέλει είχε το Δίκτυο 2.

Παρόλα αυτά, πρέπει η ακρίβεια των δικτύων να ληφθεί υπόψιν και με κάθε κλάση ξεχωριστά για να φανεί πόσο ικανά είναι να τις προβλέψουν ξεχωριστά στον χώρο, ανεξάρτητα από την γενική ακρίβεια που έχουν. Με παρόμοιο τρόπο που υπολογίστηκε και η γενική ακρίβεια, υπολογίστηκε και για κάθε κλάση μέσω της διαδικασίας με τα δεδομένα ελέγχου που προαναφέρθηκε, για το πόσες φορές εντοπίζουν σωστά την ύπαρξη της κάθε κλάσης όταν αυτή όντως υπάρχει στον χώρο. Με μια διάρεση με τον συνολικό αριθμό ύπαρξης της κλάσης στο σύνολο ελέγχου, υπολογίζεται η επιθυμητή ακρίβεια. Άρα μέσα πάλι από το script access.py παράγονται τα εξής στατιστικά:

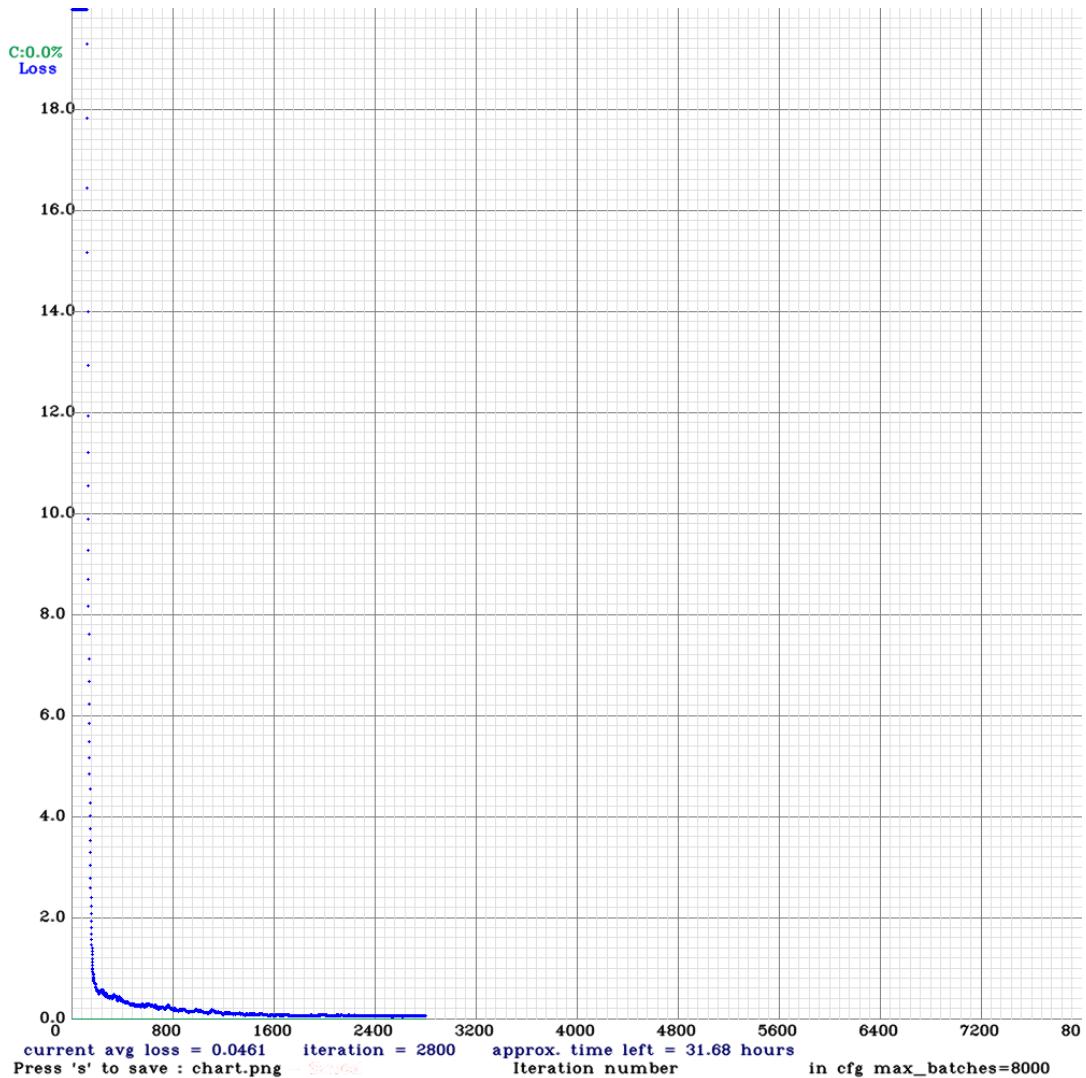
Ον/σια	Κλάση 0	Κλάση 1	Κλάση 2	Κλάση 3	-
Δίκτυο 1	95.42%	89.1%	83.02%	73.79%	96.89%
Δίκτυο 2	64.5%	67.32%	96.89%	47.79%	98.86%
Δίκτυο 3	95.42%	95.51%	90.43%	63.35%	98.36%

Πίνακας 5.4: Ακρίβεια πρόβλεψης κάθε κλάσης από τα 3 δίκτυα.

Οι κλάσεις 0,1,2,3,- αντιστοιχούν στο κόκκινο, πράσινο, κίτρινο, όχημα, τίποτα. Το “τίποτα” μπορεί να μην είναι ακριβώς κλάση αλλά συμβολίζει την περίπτωση που δεν εντοπίζεται κανένα αντικείμενο στον χώρο. Είναι καίριας σημασίας το συγκεκριμένο ποσοστό καθώς τις περισσότερες φορές στην διάρκεια της οδήγησης, δεν πραγματοποιείται εντοπισμός.

Άλλο ένα κριτήριο που πρέπει να φέρει τα δίκτυα σε σύγκριση είναι η πορεία του μέσου σφάλματος κατά την εκπαίδευση. Για το Δίκτυο 1 φαίνεται από το διάγραμμα 3.5, για το Δίκτυο 2 από το 5.1 και για το Δίκτυο 3 από το 5.2.

Τα δεδομένα από τα διαγράμματα 3.5, 5.1 και 5.2 εξάγουν τον πίνακα 5.5 ο οποίος φέρνει με πιο μεγάλη σαφήνεια τα 3 δίκτυα σε σύγκριση. Σε αυτόν τον

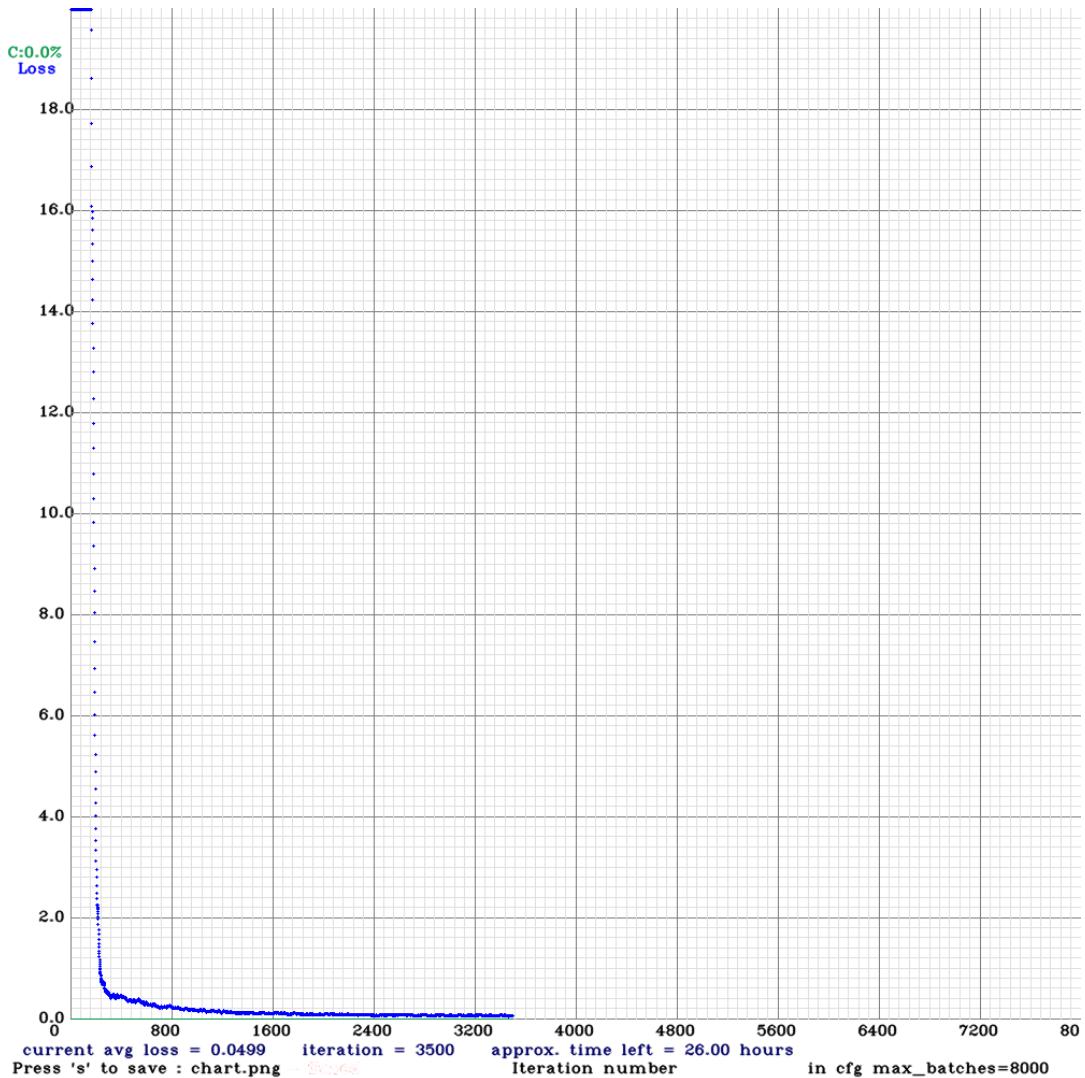


Σχήμα 5.1: Διάγραμμα μέσου σφάλματος εκπαίδευσης για Δίκτυο

2

πίνακα φαίνεται κιόλας ότι υπάρχει αντίκτυπος στο τελικό αποτέλεσμα ανάλογα των επαναλήψεων που γίνανε κατά την εκπαίδευση των δίκτυων. Το Δίκτυο 2 κατορθώνει να διαχριθεί με μικρότερο σφάλμα αν και αντιφατικά, βάση των στοιχείων στον πίνακα 5.4 υστερεί υπερβολικά στην μεμονωμένη πρόβλεψη κλάσεων σε σύγκριση με το Δίκτυο 1 και Δίκτυο 3.

Λαμβάνοντας υπόψιν όλα τα στοιχεία που έχουν παρατεθεί, δηλαδή την γενική απόδοση πρόβλεψης, την απόδοση πρόβλεψης ανά κλάση και το μέσο σφάλμα, είναι μονόδρομος το συμπέρασμα ότι το Δίκτυο 3 είναι το πιο σωστά εκπαιδευμένο. Βέβαια δεν είναι το βέλτιστο σε κάθε τομέα. Παραδείγματος χάρη, βάση του πίνακα 5.4 το Δίκτυο 2 έχει καλύτερο ποσοστό πρόβλεψης κίτρινου σηματοδότη. Όμως, στο γενικότερο σύνολο πάντα τα στατιστικά αποτελέσματα του Δικτύου 3 είναι τα πιο αξιοπρεπή.



Σχήμα 5.2: Διάγραμμα μέσου σφάλματος εκπαίδευσης για Δίκτυο

3

### 5.2.2 Αποτελέσματα μέσω του **Carla** σε real-time

Για το πείραμα στην υποενότητα 5.1.3, μέσα σε πραγματικό χρόνο καταγράφηκαν τα penalties που καταμετρήθηκαν ανά την αύξηση των actors στο εικονικό περιβάλλον του Carla. Χρησιμοποιήθηκε το Δίκτυο 3 για το πείραμα ως το αντικειμενικά καλύτερο από τα υπόλοιπα, σε συνδυασμό με τον αυτόματο πιλότο του Carla. Η λογική που ακολουθήθηκε είναι να αυξάνονται οι πεζοί ανά 10 και τα οχήματα ανά 5. Οπότε στο σύνολο οι actors αυξάνονταν ανά 15 σε κάθε δοκιμή που γινόταν. Τα αποτελέσματα φαίνονται αναλυτικά στον πίνακα 5.6 ο οποίος συγκεντρώνει τις πληροφορίες του συγκεκριμένου πειράματος.

Για να γίνει πιο ξεκάθαρη η σχέση της αύξησης των actors με το πως επηρεάζει την μεταβλητή penalty βάση των αριθμών στον πίνακα 5.1.3 δημιουργήθηκε το διάγραμμα 5.3.

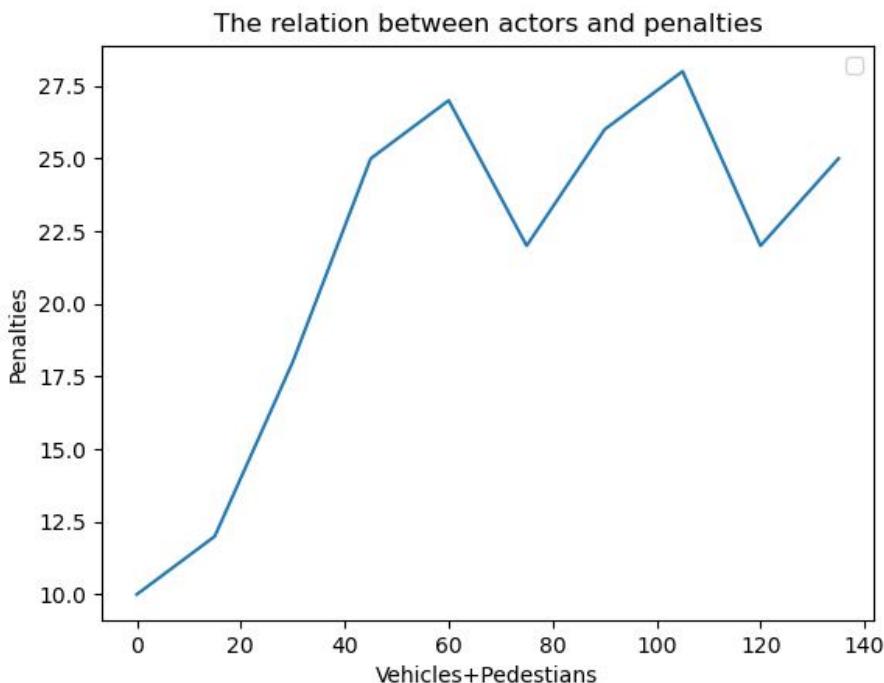
Όπως φαίνεται μέσα από το διάγραμμα 5.3, υπάρχει μια σταδιακή αύξηση του ρυθμού που αυξάνεται η μεταβλητή penalty μέχρι περίπου όταν ο αριθμός των actors είναι στο 40. Επίσης, η μικρότερη τιμή που πήρε ήταν όταν δεν υπήρχε

Ον/σια	Επαναλήψεις	Μέσο σφάλμα
Δίκτυο 1	2700	0.0692
Δίκτυο 2	2800	0.0461
Δίκτυο 3	3500	0.0499

Πίνακας 5.5: Πίνακας μέσου σφάλματος για τα 3 δίκτυα

Πεζοί+οχήματα	0	15	30	45	60	75	90	105	120	135
Penalties	10	12	18	25	27	22	26	28	22	25

Πίνακας 5.6: Αποτελέσματα του πειράματος της υποενότητας 5.1.3.



Σχήμα 5.3: Διάγραμμα σχέσης penalty-actors.

κανένας πεζός και όχημα στο εικονικό περιβάλλον. Αυτό είναι λογικό κιόλας καθώς μειώνεται η πιθανότητα να λανθασμένης πρόβλεψης όταν τα αντικείμενα στον χώρο δεν είναι πολλά. Όμως, στη συνέχεια παρατηρείται ότι δεν υπάρχει ξεκάθαρο μοτίβο στην γραφική πορεία των penalties. Συμβαίνει μια διαρκή αυξομείωση η οποία τείνει συνεχώς ανάμεσα στο 25 και στο 27. Ούτε καλυτερεύει ούτε χειροτερεύει η κατάσταση. Συνεπώς, μετά τους 40 actors η κατάσταση παραμένει κατά προσέγγιση σταθερή στο πόσο σφάλει ο YOLO κατά την διάρκεια της οδήγησης. Αυτό είναι ευχάριστο καθώς δεν θα ήταν επιθυμητό ένα συχνό γεγονός όπως ο συνοστισμός οχημάτων και πεζών να καθορίζει δραματικά την πιθανότητα λανθασμένης πρόβλεψης από το δίκτυο.

Επίσης, τα πειράματα που αναφέρονται στην υποενότητα 5.1.4 είναι επίσης σε πραγματικό χρόνο και γίνανε και αυτά με το Δίκτυο 3. Καταγραφόταν η ουσόνη του

τοπικού υπολογιστή κάτα την διάρκεια των δοκιμών, ώστε να υπάρξει φωτογραφικό υλικό στο πως απέδωσε, κάτω από τις συνθήκες που προγραμματίστηκε να βρεθεί, το ελεγχόμενο όχημα.

Όσον αφορά τα πειράματα των ταχυτήτων, για αρχή θα εξηγηθεί το απλό σενάριο που πορεύεται το αμάξι με τον αυτόματο πιλότο. Φαίνεται να εντοπίζει αμέσως στο σωστό σημείο τον σηματοφόρο όπως προδίδει και το πλαίσιο εντοπισμού στην εικόνα 5.4. Βρίσκει αμέσως το χρώμα του σηματοφόρου, σε χρόνο αρκετά γρήγορο ώστε να προβεί γρήγορα το σύστημα σε διαδικασία φρεναρίσματος αν το χρώμα είναι χόκκινο. Βέβαια αυτό ήταν αναμενόμενο καθώς όλα τα δεδομένα εκπαίδευσης του Dataset 1 και Dataset 2 είναι βάση του αυτόματου πιλότου. Όμως, αυτό που έχει ενδιαφέρον είναι το τι γίνεται αν ο χρήστης/οδηγός, ο οποίος δεν διαμέτει ικανότητα ιδανικής οδήγησης, πάρει το τιμόνι.



Σχήμα 5.4: Πείραμα απόδοσης εντοπισμού σηματοδότη από YOLO υπό φυσιολογική ταχύτητα αυτόματου πιλότου.

Η αργή ταχύτητα δεν φαίνεται να αποτελεί κανένα πρόβλημα καθώς από την εικόνα 5.5, γίνεται αντιληπτό ότι το όχημα πάλι σταματάει περίπου στην ίδια απόσταση από τον σηματοδότη, με αυτήν που σταματούσε και όταν βρισκόταν σε αυτόματο πιλότο. Επίσης, παρατηρήθηκε η ίδια συμπεριφορά στην διαδικασία εντοπισμού του χρώματος και της ταχύτητας πρωτοβουλίας του οχήματος στο φρενάρισμα. Οπότε, με ασφάλεια μπορούμε να πούμε ότι η αργή ταχύτητα δεν επηρεάζει καθόλου την αποδοτικότητα ορθού εντοπισμού. Δυστυχώς, αυτό δεν φαίνεται να ισχύει και για την γρήγορη ταχύτητα.

Στην εικόνα 5.6, τα 2 frames που έχουν αποτυπωθεί δείχνουν ότι κατά την διάρκεια υψηλής ταχύτητας, το αυτοκίνητο δεν αντιλαμβάνεται στον χώρο την ύπαρξη του σηματοδότη και το κάνει με χρονική καθυστέρηση, αφού το αμάξι πλέον τον έχει προσπεράσει. Ο YOLO αντιλαμβάνεται αυτό το γεγονός ως λάθος εντοπισμό και άμεσα τον διορθώνει. Η ένδειξη μηνύματος που φαίνεται μετά από



Σχήμα 5.5: Πείραμα απόδοσης εντοπισμού σηματοδότη από YOLO υπό αργή ταχύτητα.

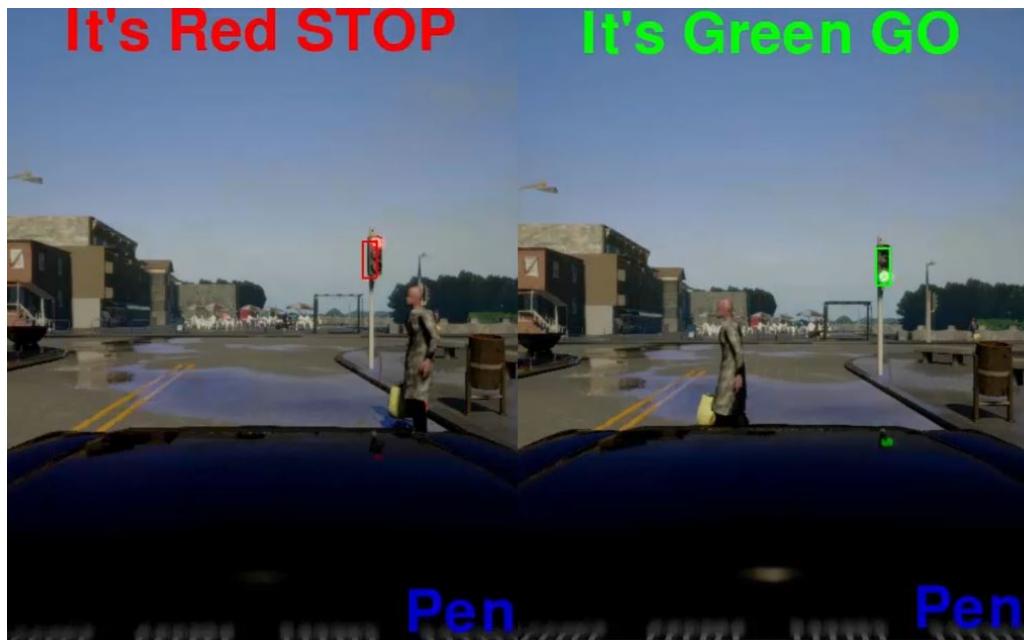
ελάχιστο χρόνο, κάνει αντιληπτό ότι δεν εντοπίζεται τίποτα στον χώρο, γεγονός που αληθεύει. Μπορεί να βγει ως συμπέρασμα ότι το σύστημα υπό υπερβολική ταχύτητα, είναι ανίκανο να πάρει την πρωτοβουλία φρεναρίσματος την κατάλληλη στιγμή και στην κατάλληλη απόσταση από τον σηματοδότη.



Σχήμα 5.6: Πείραμα απόδοσης εντοπισμού σηματοδότη από YOLO υπό γρήγορη ταχύτητα.

Ευχάριστα όμως εκπλήσσει η απόδοση του δικτύου, καθώς στο πείραμα με τον πεζό, δεν γίνεται ούτε μια στιγμή κάποιο λάθος. Όλη την ώρα σταθερά εντοπίζει τον σηματοδότη ως κόκκινο και αγνοεί εντελώς τον πεζό ως αντικείμενο, και στο τέλος επίσης ομαλώς, εντοπίζει και την αλλαγή του σηματοδότη από κόκκινο

σε πράσινο. Οπότε, η διαρκή κίνηση του πεζού και ότι αυτό επιφέρει ως οπτική αλλαγή, δεν επηρεάζει καθόλου στην απόδοση. Το πείραμα φαίνεται κιόλας στην εικόνα 5.7.



Σχήμα 5.7: Πείραμα απόδοσης εντοπισμού σηματοδότη από YOLO υπό την αναμονή πεζού να διασχίσει τον δρόμο.



Σχήμα 5.8: Πείραμα απόδοσης ταυτόχρονου εντοπισμού σηματοδότη και οχήματος από YOLO.

Παραλίγο εξίσου ευχάριστο αποτέλεσμα να είχε και το πείραμα εντοπισμού 2 αντικειμένων. Όμως εκεί υπάρχει μερικό λάθος στην όλη διαδικασία που φαίνεται και στην εικόνα 5.8. Το αμάξι, με χειροκίνητο έλεγχο μέσω του πληκτρολογίου, βρέθηκε υπό τέρμα συγκεκριμένη συνθήκη, που τοποθετείται σε απόσταση αρκετά

κοντινή ώστε να πρέπει να εντοπίσει και μπροστινό όχημα και σηματοδότη μαζί με το χρώμα του. Η καλή είδηση είναι ότι εντοπίζει με ακρίβεια και το όχημα και τον σηματοδότη στον χώρο. Η αρνητική είδηση είναι ότι όσο εντοπίζει ταυτόχρονα το όχημα, δεν αντιλαμβάνεται ότι το χρώμα του σηματοδότη είναι κίτρινο και το κατατάσσει το σύστημα σε κόκκινο. Μόνο όταν το όχημα απομακρυνθεί αρκετά από το οπτικό πεδίο του αμαξιού και δεν εντοπίζεται πλέον, ο YOLO κάνει σωστή πρόβλεψη του χρώματος. Αυτό δείχνει ότι η ύπαρξη του οχήματος και η κίνηση του στον χώρο επηρεάζει αρνητικά την τελική πρόβλεψη στο χρώμα του σηματοδότη. Τουλάχιστον το σύστημα φαίνεται ικανό να καταλάβει πότε υπάρχουν 2 αντικείμενα στον χώρο που πρέπει να ανιχνευθούν.

### 5.2.3 Σχολιασμός και Σημασία Αποτελεσμάτων

Έχει μεγάλη σημασία να αποσαφηνιστούν όλα τα παραπάνω αποτελέσματα στο πόσο θετικά είναι και στο αν ειναι λειτουργικά. Σκοπός του κάθε συστήματος αυτόνομης οδήγησης είναι η ελαχιστοποίηση των σφαλμάτων αν όχι η εξαφάνιση τους. Αυτή η φιλοσοφία φαίνεται να δικαιώθηκε από την γενική ακρίβεια των δικτύων στον πίνακα 5.3. Ο YOLO κατά κύριο λόγο αντιλαμβάνεται με μεγάλη ακρίβεια τα αντικείμενα που εκπαιδεύτηκε να αναγνωρίζει, πράγμα που αποδεικνύει την καταλληλότητα του για τον στόχο της συγκεκριμένης εργασίας. Βέβαια στον πίνακα 5.4, γίνονται πιο ξεκάθαρες και οι αδυναμίες του YOLO κατά την εκπαίδευση. Για παράδειγμα, είναι ξεκάθαρο ότι δυσκολεύεται να εντοπίσει ορθά τα οχήματα στον χώρο, συγκριτικά με τις υπόλοιπες κλάσεις. Μάλιστα στην περίπτωση του Δικτύου 2, το ποσοστό ακρίβειας είναι χειρότερο από 50%. Αυτό το γεγονός καθιστά το συγκεκριμένο δίκτυο αυτόματα μη λειτουργικό όσον αφορά την πρόβλεψη οχημάτων.

Σχετικά με το πείραμα που τα αποτελέσματα του φαίνονται στον πίνακα 5.6, ο μέσος όρος των **penalties** υπολογίζεται γύρω στο 21. Πρέπει να ξεκαθαριστεί πόσο αρνητικό είναι αυτό ως γεγονός. Όπως αναφέρθηκε σε προηγούμενο κεφάλαιο, το Carla έχει οριστεί να δουλεύει σε 28 fps και ο YOLO εξάγει δεδομένα μόνο από τα 7 frames. Εφόσον κάθε επανάληψη του πειράματος είχε διάρκεια 15 λεπτά, τότε ο YOLO εξήγαγε σε κάθε επανάληψη 6300 δεδομένα. Αν ο μέσος όρος των **penalties** διαιρεθεί με τον συνολικό αριθμό δεδομένων, παράγεται ένα ποσοστό σφάλματος τιμής 0,003%. Είναι ένας πολύ μικρός αριθμός που δείχνει πόσο καλά εκπαιδεύτηκε το δίκτυο. Όμως κατά την οδήγηση, που το παραμικρό λάθος μπορεί να αποβεί μοιραίο, το συγκεκριμένο ποσοστό πρέπει να ελαττωθεί ακόμα παραπάνω. Δεν αρκεί αντικειμενικά η ασφάλεια που προσφέρει στον οδηγό.

Τέλος, τα πειράματα υπό συγκεκριμένες συνθήκες, βγάζουν ως γενικότερο συμπέρασμα ότι δεν αποδίδει πάντα επιθυμητά ο YOLO. Το να αργήσει να προβλέψει στο πείραμα της εικόνας 5.6 το κόκκινο φανάρι, μπορεί να αποβεί μοιραίο για έναν οδηγό που έχει εμπιστευτεί πλήρως το σύστημα. Βέβαια το ότι δεν είναι ο τέλειος αλγόριθμος, δεν πρέπει να επισκιάζει ότι στο σύνολο φάνηκε αξιοπρεπής κάτω από τις περισσότερες συνθήκες που βρέθηκε. Το γεγονός ότι στις περισσότερα σενάρια υπήρχε μεγάλη πιθανότητα σφάλματος και οι σωστές αποχρίσεις ήταν περισσότερες από τις λανθασμένες, δείχνει ότι η εκπαίδευση που υπήρξε είχε όντως αποτελεσματικότητα.

## Κεφάλαιο 6

# Σύνοψη και Μελλοντική Εργασία

Αναλύοντας την απόδοση και τα συνολικά αποτελέσματα που προέκυψαν στο έργο της συγκεκριμένης εργασίας, μπορούν να εξαχθούν ορισμένα πορίσματα. Πρώτον, ο αρχικός ερευνητικός σκοπός επιτεύχθηκε ο οποίος ήταν να ενσωματωθεί ένα σύστημα ικανό να ανιχνεύει και να αναγνωρίζει έναν σηματοφόρο, το χρώμα του και τα κοντινά οχήματα, σε ένα όχημα που προηγουμένως δεν υπήρχε κάποια έξυπνη συσκευή ή λογισμικό που εντοπίζει αντικείμενα. Επιπλέον, τέσσερις εφαρμογές με κίνητρο αυτή τη διαδικασία ανίχνευσης έχουν εφαρμοστεί, δίνοντας στο σύστημα μια πιο γνωστική και αντιληπτική χρησιμότητα. Επιπλέον, αυτό το έργο συμβάλλει σε μεγάλο βαθμό στην έρευνα για την αυτόνομη οδήγηση, διότι τα σύνολα δεδομένων, ο κώδικας και τα εκπαιδευμένα μοντέλα μπορούν να βρεθούν στον ιστότοπο [Github](#) που δημιουργήθηκε για αυτό το έργο. Αυτό επιτρέπει στον καθένα να χρησιμοποιήσει ελεύθερα οτιδήποτε παράχθηκε σε αυτή την εργασία για την δική του έρευνά.

Αυτή η εργασία έκανε αντιληπτό το πόσο εύχρηστος είναι ο αλγόριθμος **YOLO** στον τομέα της ανίχνευσης αντικειμένων. Στην αυτόνομη οδήγηση αποτελεί μεγάλη σημασία το να μπορεί ένα σύστημα ενσωματωμένο στο αμάξι, να αναγνωρίσει το τι υπάρχει στον χώρο. Εφόσον αναγνωρίσει το τι βρίσκεται στον χώρο, έπειτα πρέπει να έχει ένα αρκετά αποδοτικό σύστημα που σε πραγματικό χρόνο να αποφασίζει την πιο ορθή οδική απόφαση που πρέπει να πάρει το όχημα. Στη συγκεκριμένη εργασία, έγινε ξεκάθαρο ότι αυτό δεν απέχει καθόλου από το να μπορεί υλοποιηθεί στην πραγματικότητα. Μέσα από το γραφικό περιβάλλον του **Carla** και σε συνδυασμό των περιορισμένων πόρων του τοπικού υπολογιστή, κατορθώθηκε να δημιουργηθεί ένα σύστημα υψηλής ακρίβειας και απόδοσης. Αυτό και μόνο του σαν γεγονός είναι εντυπωσιακό. Πόσο μάλλον αν ληφθεί υπόψιν η βελτίωση που μπορεί να υπάρξει τουλάχιστον στους διαθέσιμους πόρους.

Είναι σχεδόν αυτονόητο ότι η όλη διαδικασία δημιουργίας του λογισμικού που πραγματεύεται η συγκεκριμένη εργασία, πρέπει να γίνει εξολοκλήρου σε δυνατότερα συστήματα υπολογιστή. Μια ισχυρή κάρτα γραφικών έχει βασικό ρόλο σε ένα σύστημα που εκτελείται εντός γραφικού περιβάλλοντος. Επιπρόσθετα, όταν ένα πρόβλημα αφορά επεξεργασία εικόνων, οι υπολογιστικές πράξεις μέσα από την κάρτα γραφικών αυξάνουν δραματικά την απόδοση σε χρονικό πλαίσιο. Επίσης, όλοι οι αλγόριθμοι εντοπισμού, όπως είναι και ο **YOLO**, αποδίδουν καλύτερα μέσα από την κάρτα γραφικών και όχι μέσα από τον επεξεργαστή, που η συγκεκριμένη εργασία τον χρησιμοποιεί για κάθε διαδικασία.

Ένα βασικό ζήτημα που πρέπει να συνυπολογιστεί, που πάλι πρόκυπτε από τους ελλείπεις πόρους, είναι το μέγεθος των δεδομένων εκπαίδευσης. Αντικειμενικά, αν υπήρξε τέτοια ακρίβεια με τόσο λίγα δεδομένα εκπαίδευσης, είναι τρομερά

ελπιδοφόρο το τι μπορεί να προκύψει με την ενσωμάτωση περισσότερων κατά την διάρκεια της εκπαίδευσης του YOLO. Αυτό θα καθιστούσε εφικτό, το να μην υπάρχει και ο κίνδυνος για overfitting, αν δεν διακοπεί ο αλγόριθμος στο ορισμένο max\_batches. Θα υπάρχει σίγουρα καλύτερη εκπαίδευση εντοπισμού μεμονωμένα για κάθε κλάση που έχει οριστεί.

Πέρα από την όλη διαδικασία εκπαίδευσης, με επιπλέον υπολογιστικούς πόρους, είναι εφικτό το να εκμεταλλευτεί περισσότερο το σύστημα τα δεδομένα από τον γραφικό ή και πραγματικό κόσμο στον οποίο βρίσκεται. Η συγκεκριμένη εργασία χρησιμοποιεί μόνο μια κάμερα RGB που καταγράφει τον χώρο. Θα μπορούσαν να χρησιμοποιούνται περισσότερες κάμερες από διάφορες οπτικές γωνίες του οχήματος. Αυτό θα αύξανε σημαντικά τον όγκο πληροφορίας που μπορεί να αντλήσει το σύστημα για το τι συμβαίνει τριγύρω. Ήδη η προσθήκη μιας κάμερας στην πίσω μεριά του αμαξιού, θα μπορούσε να ενισχύσει ακόμα περισσότερο το πρόβλημα της αποφυγής συγκρούσεων με άλλα οχήματα, που η συγκεκριμένη εργασία πραγματεύεται.

Τέλος, η απόδοση του συστήματος που αναπτύχθηκε, είναι ενθαρρυντική και έχει θετικό αντίκτυπο για περαιτέρω έρευνα και μελέτες στον τομέα της αυτόνομης οδήγησης. Πιο συγκεκριμένα, έχει αντίκτυπο για την πραγματοποίηση έρευνας στο πεδίο της αναγνώρισης σηματοδοτών και οχημάτων. Κυρίως όμως, υπάρχει αντίκτυπος στο να αναπτυχθούν εφαρμογές με σκοπό να γίνει η διαδικασία οδήγησης ευκολότερη και ασφαλέστερη, βοηθώντας έτσι τον καυημερινό οδηγό αυτοκινήτου.

## Παράρτημα A'

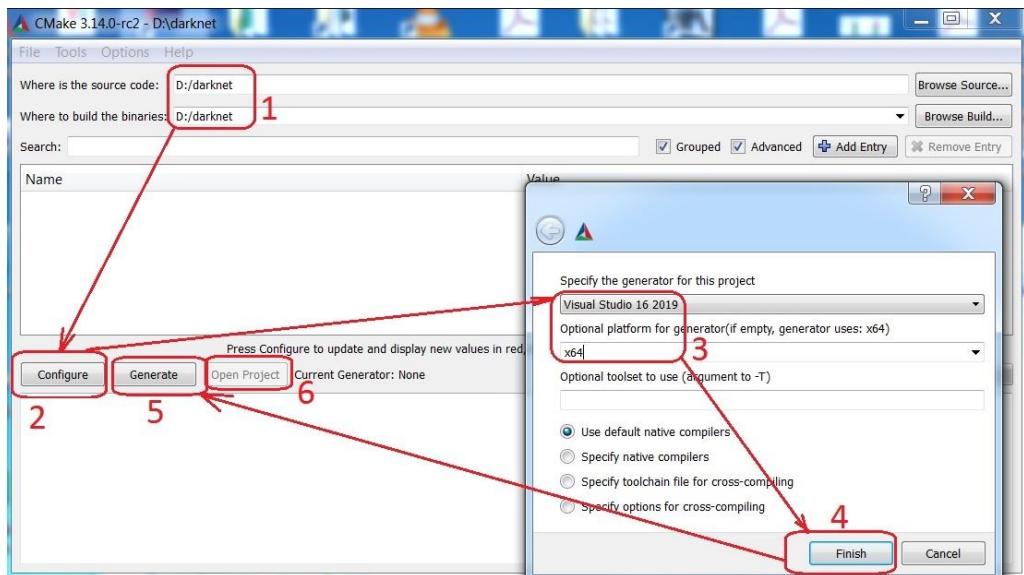
### Εγκατάσταση λογισμικού

Στο Πρόσθετο αυτό θα εξηγηθεί η μέθοδος και τρόπος να κατέβουν σε έναν τοπικό υπολογιστή τα λογισμικά που χρησιμοποιούνται για την συγχεκριμένη εργασία. Είναι απαραίτητο να διασφαλίστει το ορθό κατέβασμα δίοτι μόνο έτσι μπορεί να εκτελεστεί και ο κώδικας που συνοδεύει αυτή την εργασία. Η ανάπτυξη και εκπαίδευση πραγματοποιήθηκαν σε υπολογιστή με Windows 10.

#### A'.1 Εγκατάσταση Darknet

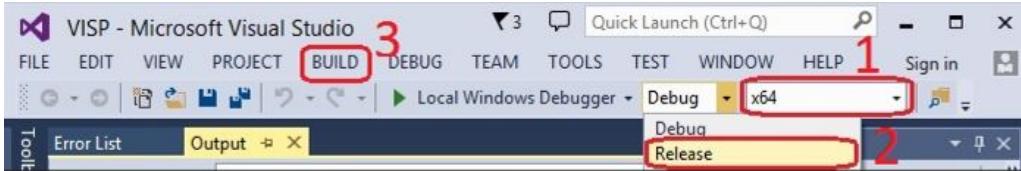
Για να λειτουργήσει ορθά το Darknet και να μπορεί κάποιος να το εκτελέσει ως πρόγραμμα, πρέπει πρώτα να κατεβάσει κάποιος το Visual Studio, το CUDA και το CMake. Έπειτα, αφού γίνει αυτό πρέπει να κατέβει και ένα συμπιεσμένο αρχείο του Darknet πατώντας τον ακόλουθο σύνδεσμο: <https://github.com/AlexeyAB/darknet/archive/master.zip>. Μετά πρέπει απλά να αποσυμπιεστεί σε ένα μέρος του υπολογιστή.

Αφού γίνουν αυτά και έχει ανοιχτεί το γραφικό περιβάλλον του CMake, πρέπει να ακολουθήσει κάποιος τις εξής τροποποιήσεις που αναγράφει η εικόνα A'.1 μέσα στο πρόγραμμα. Έχουν να κάνουν με την ενσωμάτωση του Darknet και την παραγωγή του εκτελέσιμου αρχείου του ορίζοντας τα σχετικά paths.



Σχήμα A'.1: Τροποποιήσεις στο CMake.

Τυπερα, πρέπει να ανοιχτεί το γραφικό περιβάλλον του Visual Studio και να πατηθούν με την σειρά ότι αναγράφεται αριθμημένα στην παρακάτω εικόνα Α'.2 για να παραχθεί στο ορισμένο path το αρχείο darknet.exe.



Σχήμα Α'.2: Δημιουργία εκτελέσιμου αρχείου μέσω Visual Studio.

Τέλος, αφού γίνει βέβαιο ότι το CUDA έχει εγκαταστηθεί αρμονικά στον υπολογιστή, πρέπει να ανοιχτεί ένα Windows Powershell και να εκτελεστούν μέσα σε αυτό οι εξής εντολές:

```
Set-ExecutionPolicy unrestricted -Scope CurrentUser -Force
git clone https://github.com/AlexeyAB/darknet
cd darknet
.\build.ps1 -UseVCPKG -EnableOPENCV -EnableCUDA -EnableCUDNN
```

Με την επίτευξη όλων των παραπάνω, το Darknet είναι έτοιμο για χρήση.

## A'.2 Εγκατάσταση Carla

Είναι αρκετά απλή η διαδικασία που ακολουθείται. Η έκδοση που χρησιμοποιείται είναι το Carla 0.8.2 και το μόνο που χρειάζεται είναι να κατέβει το συμπιεσμένο αρχείο που προτείνεται για υπολογιστές Windows στον παρακάτω σύνδεσμο: <https://github.com/carla-simulator/carla/releases/tag/0.8.2/>. Έπειτα απλά πρέπει να αποσυμπιεστεί οπουδήποτε στον υπολογιστή.

Προσοχή όμως διότι η συγκεκριμένη έκδοση δουλεύει μόνο σε python 3.6 , άρα δεν θα μπορεί να εκτελεστεί χανένα πρόγραμμα του Carla. Η διαδικασία για να κατέβει η έκδοση της είναι απλή μέσα από την επίσημη ιστοσελίδα της Python όμως πρέπει κατά την διαδικασία του κατεβάσματος, να επιλεχθεί η χυκλωμένη επιλογή στην εικόνα Α'.3. Αφού γίνει αυτό το Carla μπορεί επιτέλους να λειτουργήσει.

## A'.3 Εγκατάσταση labelImg

Για αρχή πρέπει πρώτα να έχει κατέβει στον υπολογιστή μια έκδοση της Python. Μετά από αυτό μέσα από Command Prompt παράθυρο πρέπει να τρέξουν οι εντολές:

```
pip install PyQt5
pip install lxml
```

Αυτές οι εντολές θα κατεβάσουν απαραίτητα προγράμματα για να μπορέσει να λειτουργήσει το labelImg. Αφού έχει γίνει αυτό τότε πρέπει πάλι σε παράθυρο από Command Prompt να τρέξουν οι εξής εντολές:

```
pyrcc4 -o libs/resources.py resources.qrc
For pyqt5, pyrcc5 -o libs/resources.py resources.qrc
python labelImg.py
python labelImg.py [IMAGE_PATH] [PRE-DEFINED CLASS FILE]
```



Σχήμα Α'.3

Αφού έχουν γίνει όλα αυτά, θα πρέπει τελικά να αναζητηθεί στον υπολογιστή ο φάκελος του labelImg. Μέσα του θα βρίσκεται το εκτελέσιμο αρχείο που θα οδηγήσει εν τέλει στο γραφικό περιβάλλον της εφαρμογής.

# Bιβλιογραφία

- [1] *1st. World and client.* [https://carla.readthedocs.io/en/latest/core\\_world/](https://carla.readthedocs.io/en/latest/core_world/). 2017.
- [2] *Actors and blueprints.* [https://carla.readthedocs.io/en/latest/core\\_actors/](https://carla.readthedocs.io/en/latest/core_actors/). 2017.
- [3] "America's Workforce and the Self-Driving Future Realizing Productivity Gains and Spurring Economic Growth". In: Securing America's Future Energy, June 2018. URL: [https://avworkforce.secureenergy.org/wp-content/uploads/2018/06/SAFE\\_AV\\_Policy\\_Brief.pdf](https://avworkforce.secureenergy.org/wp-content/uploads/2018/06/SAFE_AV_Policy_Brief.pdf).
- [4] J. C. Anderson and D. W. Gerbing. "Structural equation modeling in practice: A review and recommended two-step approach." In: *Psychological Bulletin* 103.3 (1988), 411–423. DOI: <https://doi.org/10.1037/0033-2909.103.3.411>.
- [5] James M. Anderson et al. *Autonomous Vehicle Technology A Guide for Policymakers*. "[https://www.rand.org/pubs/research\\_reports/RR443-2.html](https://www.rand.org/pubs/research_reports/RR443-2.html)". Rand Corporation, 2016, pp. 28,120. ISBN: 9780833083982. DOI: 10.7249/RR443-2. URL: [https://www.rand.org/pubs/research\\_reports/RR443-2.html](https://www.rand.org/pubs/research_reports/RR443-2.html).
- [6] *Artificial intelligence driving autonomous vehicle development.* [https://ihsmarkit.com/research-analysis/artificial-intelligence-driving-autonomous-vehicle-development.html?fbclid=IwAR10VTBDsjlsAsMH1D8FkJ-7SQA1tdCq8v4PuM5JzAmAOuQDb7iz6\\_eR53U](https://ihsmarkit.com/research-analysis/artificial-intelligence-driving-autonomous-vehicle-development.html?fbclid=IwAR10VTBDsjlsAsMH1D8FkJ-7SQA1tdCq8v4PuM5JzAmAOuQDb7iz6_eR53U). Jan. 2020.
- [7] Wymann B. *TORCS, The Open Racing Car Simulator*. <http://torcs.sourceforge.net/>. 2000.
- [8] Sebastiano Battiato et al. "Depth map generation by image classification". In: *Electronic Imaging 2004, 2004, San Jose, California, United States*. Vol. 5302. SPIE, Apr. 2004. DOI: <https://doi.org/10.1117/12.526634>. URL: <https://www.spiedigitallibrary.org/conference-proceedings-of-spie/5302/0000/Depth-map-generation-by-image-classification/10.1117/12.526634.short?SSO=1>.
- [9] Sven A. Beiker. "Legal Aspects of Autonomous Driving". In: *Santa Clara L. Rev.* 52.4 (Dec. 2012), 1144–1156. URL: <https://digitalcommons.law.scu.edu/lawreview/vol52/iss4/1/>.
- [10] Michele Bertoncello and Dominik Wee. "Ten ways autonomous driving could redefine the automotive world". In: *McKinsey Company* (Jan. 2015). URL: <https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/ten-ways-autonomous-driving-could-redefine-the-automotive-world>.

- [11] Nacer Eddine Bezai et al. "Future cities and autonomous vehicles: analysis of the barriers to full adoption". In: (May 2020), p. 33. DOI: <https://doi.org/10.1016/j.enbenv.2020.05.002>. URL: [https://www.researchgate.net/figure/Connected-autonomous-vehicle-system-architecture-overview-reproduced-from-113-124-128\\_fig5\\_341731497](https://www.researchgate.net/figure/Connected-autonomous-vehicle-system-architecture-overview-reproduced-from-113-124-128_fig5_341731497).
- [12] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. "YOLOv4: Optimal Speed and Accuracy of Object Detection". In: (Apr. 2020), p. 7. DOI: <https://doi.org/10.48550/arXiv.2004.10934>. URL: <https://arxiv.org/abs/2004.10934>.
- [13] Dominik B. O. Boesl and Bernd Liepert. "4 Robotic Revolutions - Proposing a holistic phase model describing future disruptions in the evolution of robotics and automation and the rise of a new Generation 'R' of Robotic Natives". In: International Conference on Intelligent Robots and Systems (IROS). Oct. 2016, 1262–1267. URL: <https://www.roboticgovernance.com/wp-content/uploads/2017/01/1285.pdf>.
- [14] CARLA Open-source simulator for autonomous driving research. <https://carla.org/>.
- [15] Lewis M. Clements and Kara M. Kockelman. "Economic Effects of Automated Vehicles". In: *SAGE Journals* 2606.1 (Jan. 2017), pp. 106–114. DOI: 10.3141/2606-14. URL: <https://journals.sagepub.com/doi/abs/10.3141/2606-14>.
- [16] "Communicating Science Effectively: A Research Agenda". In: National Academies of Sciences, Engineering, and Medicine [NASEM]. 2017. DOI: 10.17226/23674. URL: <https://nap.nationalacademies.org/catalog/23674/communicating-science-effectively-a-research-agenda>.
- [17] N. Dalal and B. Triggs. "Histograms of oriented gradients for human detection". In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. Vol. 1. IEEE, June 2005, 886–893. ISBN: 0-7695-2372-2. DOI: 10.1109/CVPR.2005.177. URL: <https://ieeexplore.ieee.org/document/1467360>.
- [18] Jia Deng et al. "ImageNet: A large-scale hierarchical image database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, June 2009, 248–255. ISBN: 978-1-4244-3992-8. DOI: 10.1109/CVPR.2009.5206848. URL: <https://ieeexplore.ieee.org/document/5206848>.
- [19] John Doyle, Francis A, and Allen Tannenbaum. *Feedback Control Theory*. Jan. 2009. DOI: 10.1007/978-0-387-85460-1\_1.
- [20] Anders Eugensson et al. "ENVIRONMENTAL, SAFETY, LEGAL AND SOCIETAL IMPLICATIONS OF AUTONOMOUS DRIVING SYSTEMS". In: (2013), pp. 1–15. URL: <https://www-esv.nhtsa.dot.gov/Proceedings/23/files/23ESV-000467.PDF>.

- [21] Ross Girshick et al. "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, June 2014, 580–587. ISBN: 978-1-4799-5118-5. DOI: 10.1109/CVPR.2014.81. URL: <https://ieeexplore.ieee.org/document/6909475>.
- [22] Jeffery B. Greenblatt and Susan Shaheen. "Automated Vehicles, On-Demand Mobility, and Environmental Impacts". In: *SpringerLink* 2 (July 2015), pp. 74–81. DOI: 10.1007/s40518-015-0038-5. URL: <https://link.springer.com/article/10.1007/s40518-015-0038-5>.
- [23] P. A. Hancock. "Automation: how much is too much?" In: *Taylor and Francis Online* 57.3 (Sept. 2013), pp. 449–454. DOI: 10.1080/00140139.2013.816375. URL: <https://www.tandfonline.com/doi/abs/10.1080/00140139.2013.816375>.
- [24] P. A. Hancock. "Some pitfalls in the promises of automated and autonomous vehicles". In: *Taylor and Francis Online* 62.4 (Sept. 2017), pp. 479–495. DOI: 10.1080/00140139.2018.1498136. URL: <https://www.tandfonline.com/doi/abs/10.1080/00140139.2018.1498136>.
- [25] Fengxiang He, Tongliang Liu, and Dacheng Tao. "Control Batch Size and Learning Rate to Generalize Well: Theoretical and Empirical Evidence". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., 2019. URL: <https://proceedings.neurips.cc/paper/2019/file/dc6a70712a252123c40d2adba6a11d84-Paper.pdf>.
- [26] Kaiming He et al. "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.9 (2015), pp. 1904–1916. DOI: 10.1109/TPAMI.2015.2389824.
- [27] Christoph Hohenberger, Matthias Spörrle, and Isabell M. Welpe. *Not fearless, but self-enhanced: The effects of anxiety on the willingness to use autonomous cars depend on individual levels of self-enhancement*. Vol. 116. Technological Forecasting and Social Change, Mar. 2017, pp. 40–52. DOI: <https://doi.org/10.1016/j.techfore.2016.11.011>.
- [28] Kangchan Lee Hongki Cha. "Facilitating the Development of Self-Driving Cars with Open-source Projects". In: 2021 International Conference on Information and Communication Technology Convergence (ICTC), Oct. 2021. ISBN: 978-1-6654-2384-7. DOI: 10.1109/ICTC52510.2021.9621005. URL: <https://ieeexplore.ieee.org/abstract/document/9621005>.
- [29] *Imitation Learning for Autonomous Driving in CARLA*. <https://luffca.com/2018/07/carla-imitation-learning/>. July 2018.
- [30] Redmon J. *Darknet: Open source neural networks in c*. <https://pjreddie.com/darknet/>. 2013-2016.
- [31] *LabelImg Graphical image annotation tool and label object bounding boxes*. <https://sourceforge.net/projects/labelimg.mirror/>.

- [32] Yanfen Li et al. "A Deep Learning-Based Hybrid Framework for Object Detection and Recognition in Autonomous Driving". In: 8 (Oct. 2020), pp. 194228 –194239. DOI: 10.1109/ACCESS.2020.3033289. URL: <https://ieeexplore.ieee.org/document/9238023>.
- [33] Shu Liu et al. "Path Aggregation Network for Instance Segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018. URL: [https://openaccess.thecvf.com/content\\_cvpr\\_2018/papers/Liu\\_Path\\_Aggregation\\_Network\\_CVPR\\_2018\\_paper.pdf](https://openaccess.thecvf.com/content_cvpr_2018/papers/Liu_Path_Aggregation_Network_CVPR_2018_paper.pdf).
- [34] D.G. Lowe. "Object recognition from local scale-invariant features". In: *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Vol. 99. IEEE, Sept. 1999, 1150–1157. ISBN: 0-7695-0164-8. DOI: 10.1109/ICCV.1999.790410. URL: <https://ieeexplore.ieee.org/document/790410>.
- [35] Hengliang Luo et al. "Traffic Sign Recognition Using a Multi-Task Convolutional Neural Network". In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. Vol. 19. 4. IEEE, June 2005, pp. 1100 –1111. DOI: 10.1109/TITS.2017.2714691. URL: <https://ieeexplore.ieee.org/document/7959631>.
- [36] Charles M. Macal and Michael J. North. "Agent-based modeling and simulation: ABMS examples". In: *2008 Winter Simulation Conference*. IEEE, Dec. 2008. ISBN: 978-1-4244-2707-9. DOI: 10.1109/WSC.2008.4736060. URL: <https://ieeexplore.ieee.org/abstract/document/4736060>.
- [37] Batta Mahesh. "Machine Learning Algorithms - A Review". In: *International Journal of Science and Research (IJSR)*. Vol. 9. 1. ResearchGate Impact Factor, Jan. 2020, pp. 381–386. ISBN: 978-1-4799-4584-9. URL: [https://www.researchgate.net/publication/344717762\\_Machine\\_Learning\\_Algorithms\\_-A\\_Review](https://www.researchgate.net/publication/344717762_Machine_Learning_Algorithms_-A_Review).
- [38] *Maps and navigation*. [https://carla.readthedocs.io/en/latest/core\\_map/](https://carla.readthedocs.io/en/latest/core_map/). Map and Buildings. 2017.
- [39] Aarian Marshall. "To See the Future of Cities, Watch the Curb. Yes, the Curb". In: *WIRED* (Nov. 2017). URL: <https://www.wired.com/story/city-planning-curbs/>.
- [40] Will McGugan. *Beginning Game Development with Python and Pygame From Novice to Professional*. Apress, 2007, pp. 41–67. ISBN: 978-1-59059-872-6.
- [41] M.König and L.Neumayr. *Users' resistance towards radical innovations: The case of the self-driving car*. Vol. 44. *Transportation Research Part F: Traffic Psychology and Behaviour*, Jan. 2017, pp. 42–52. DOI: <https://doi.org/10.1016/j.trf.2016.10.013>.
- [42] MOTOR VEHICLE SAFETY ISSUES. <https://injuryfacts.nsc.org/motor-vehicle/motor-vehicle-safety-issues/speeding/>.

- [43] *New rules to improve road safety and enable fully driverless vehicles in the EU.* [https://ec.europa.eu/commission/presscorner/detail/en/IP\\_22\\_4312](https://ec.europa.eu/commission/presscorner/detail/en/IP_22_4312). July 2022.
- [44] *NVIDIA CUDA Toolkit Release Notes.* <https://docs.nvidia.com/cuda/cuda-toolkit-release-notes/index.html>. 2007-2022.
- [45] Kai Rannenberg. *Opportunities and Risks Associated with Collecting and Making Usable Additional Data*. SpringerOpen, May 2016, 497–517. ISBN: 978-3-662-48845-4. DOI: 10.1007/978-3-662-48847-8. URL: [https://link.springer.com/chapter/10.1007/978-3-662-48847-8\\_24](https://link.springer.com/chapter/10.1007/978-3-662-48847-8_24).
- [46] Dmitrii Rassokhin. “The C++ programming language in cheminformatics and computational chemistry”. In: *Journal of Cheminformatics* 12.10 (Feb. 2020). DOI: <https://doi.org/10.1186/s13321-020-0415-y>. URL: <https://link.springer.com/article/10.1186/s13321-020-0415-y>.
- [47] Joseph Redmon and Ali Farhadi. “YOLOv3: An Incremental Improvement”. In: *CoRR* abs/1804.02767 (2018). arXiv: 1804.02767. URL: <http://arxiv.org/abs/1804.02767>.
- [48] Joseph Redmon et al. “You Only Look Once: Unified, Real-Time Object Detection”. In: (May 2016). DOI: <https://doi.org/10.48550/arXiv.1506.02640>. URL: <https://arxiv.org/abs/1506.02640>.
- [49] Syed A. Sattar et al. “Airborne Infectious Agents and Other Pollutants in Automobiles for Domestic Use: Potential Health Impacts and Approaches to Risk Mitigation”. In: *Journal of Environmental and Public Health* 2016 (Oct. 2016). DOI: 10.1155/2016/1548326. URL: <https://www.hindawi.com/journals/jeph/2016/1548326/>.
- [50] Hanie Sedghi, Vineet Gupta, and Philip M. Long. “The Singular Values of Convolutional Layers”. In: Published as a conference paper at ICLR 2019, May 2018. DOI: <https://doi.org/10.48550/arXiv.1805.10408>. URL: <https://arxiv.org/abs/1805.10408>.
- [51] Shital Shah et al. “AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles”. In: (July 2017). DOI: 10.48550/arXiv.1705.05065. URL: <https://arxiv.org/abs/1705.05065>.
- [52] Ravi Shanker et al. *Autonomous Cars: Self-Driving the New Auto Industry Paradigm*. <https://studylib.net/doc/8717560/autonomous-cars--self-driving-the-new-auto-industry-paradigm>. Nov. 2013.
- [53] JEAN-FRANÇOIS BONNEFONAZIM SHARIFF and IYAD RAHWAN. “The social dilemma of autonomous vehicles.” In: *Science* 352.6293 (June 2016), pp. 1573–1576. DOI: 10.1126/science.aaf2654. URL: <https://www.science.org/doi/abs/10.1126/science.aaf2654>.
- [54] Han Shi. *Automatic generation of OpenDrive roads from road measurements*. Chalmers tekniska högskola / Institutionen för data- och informationsteknik (Chalmers) Chalmers University of Technology / Department of Computer Science and Engineering (Chalmers), Nov. 2011. URL: <https://odr.chalmers.se/handle/20.500.12380/149237>.

- [55] Donald Shoup. "Cruising for Parking". In: *ACCESS Magazine*, 1(30s) (Apr. 2007). URL: <https://escholarship.org/content/qt6sn7s1x2/qt6sn7s1x2.pdf>.
- [56] Bryant Walker Smith. "SAE LEVELS OF DRIVING AUTOMATION". In: *Center for Internet and Society* (Dec. 2013). URL: <http://cyberlaw.stanford.edu/blog/2013/12/sae-levels-driving-automation>.
- [57] Samuel L. Smith, Pieter-Jan Kindermans, and Quoc V. Le. "Don't Decay the Learning Rate, Increase the Batch Size". In: *CoRR* abs/1711.00489 (2017). arXiv: 1711.00489. URL: <http://arxiv.org/abs/1711.00489>.
- [58] Jacob Solawetz. "YOLOv4 Explained". In: *roboflow* (June 2020). URL: <https://blog.roboflow.com/a-thorough-breakdown-of-yolov4/>.
- [59] Ha ryong Song et al. "Target localization using RGB-D camera and LiDAR sensor fusion for relative navigation". In: *2014 CACS International Automatic Control Conference (CACS 2014)*. IEEE, Nov. 2014. ISBN: 978-1-4799-4584-9. DOI: 10.1109/CACS.2014.7097178. URL: <https://ieeexplore.ieee.org/document/7097178>.
- [60] State of California, Department of Motor Vehicles. <https://www.dmv.ca.gov/portal/dmv/detail/vr/autonomous/permit>.
- [61] Paul C. Stern and Harvey V. Fineberg. "Understanding Risk: Informing Decisions in a Democratic Society". In: National Research Council [NRC]. 1996. DOI: 10.17226/5138. URL: <https://nap.nationalacademies.org/catalog/5138/understanding-risk-informing-decisions-in-a-democratic-society>.
- [62] Christian Szegedy et al. "Going deeper with convolutions". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2015. ISBN: 978-1-4673-6964-0. DOI: 10.1109/CVPR.2015.7298594. URL: <https://ieeexplore.ieee.org/document/7298594>.
- [63] *The simulator*. [https://carla.readthedocs.io/en/latest/start\\_introduction/](https://carla.readthedocs.io/en/latest/start_introduction/). The introduction of Carla. 2017.
- [64] Stephan Tornier. "Haar Measures". In: (June 2020), p. 13. DOI: <https://doi.org/10.48550/arXiv.2006.10956>. URL: <https://arxiv.org/abs/2006.10956>.
- [65] *Unreal Engine*. <https://www.unrealengine.com/en-US/>.
- [66] Walther Wachenfeld and Hermann Winner. *The Release of Autonomous Vehicles*. SpringerOpen, May 2016, 425–449. ISBN: 978-3-662-48845-4. DOI: 10.1007/978-3-662-48847-8. URL: [https://link.springer.com/chapter/10.1007/978-3-662-48847-8\\_21](https://link.springer.com/chapter/10.1007/978-3-662-48847-8_21).
- [67] Walther Wachenfeld et al. *Use Cases for Autonomous Driving*. SpringerOpen, May 2016. ISBN: 978-3-662-48845-4. DOI: 10.1007/978-3-662-48847-8. URL: [https://link.springer.com/chapter/10.1007/978-3-662-48847-8\\_2](https://link.springer.com/chapter/10.1007/978-3-662-48847-8_2).

- [68] Daisuke Wakabayashi. "California scraps safety driver rules for self-driving cars." In: *The New York Times* (Feb. 2018). URL: [https://www.nytimes.com/2018/02/26/technology/\(driverless-cars-california-rules.html](https://www.nytimes.com/2018/02/26/technology/(driverless-cars-california-rules.html)).
- [69] Chien-Yao Wang et al. "CSPNet: A New Backbone That Can Enhance Learning Capability of CNN". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. June 2020, pp. 390–391. URL: [https://openaccess.thecvf.com/content\\_CVPRW\\_2020/papers/w28/Wang\\_CSPNet\\_A\\_New\\_Backbone\\_That\\_Can\\_Enhance\\_Learning\\_Capability\\_of\\_CVPRW\\_2020\\_paper.pdf](https://openaccess.thecvf.com/content_CVPRW_2020/papers/w28/Wang_CSPNet_A_New_Backbone_That_Can_Enhance_Learning_Capability_of_CVPRW_2020_paper.pdf).
- [70] Weather presets. [https://carla.readthedocs.io/en/stable/carla\\_settings/](https://carla.readthedocs.io/en/stable/carla_settings/). 15 weather types. 2017.
- [71] What is momentum in neural networks. <https://deeplearning.buzz/2017/05/31/what-is-momentum-in-neural-networks/>. May 2017.
- [72] Thomas Winkle. *Development and Approval of Automated Vehicles: Considerations of Technical, Legal, and Economic Risks*. SpringerOpen, May 2016, 589–618. ISBN: 978-3-662-48845-4. DOI: 10.1007/978-3-662-48847-8. URL: [https://link.springer.com/chapter/10.1007/978-3-662-48847-8\\_28](https://link.springer.com/chapter/10.1007/978-3-662-48847-8_28).
- [73] Xue Ying. "An Overview of Overfitting and its Solutions". In: *Journal of Physics: Conference Series* 1168 (2019), p. 022022. DOI: 10.1088/1742-6596/1168/2/022022. URL: <https://doi.org/10.1088/1742-6596/1168/2/022022>.