
1.5em



Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης
Πολυτεχνική Σχολή
Τμήμα Ηλεκτρολόγων Μηχανικών &
Μηχανικών Γπολογιστών
Τομέας Ηλεκτρονικής και Γπολογιστών

Διπλωματική Εργασία

Ανάπτυξη διεπαφής αυτόνομου οχήματος για
προσαρμογή της συμπεριφοράς του μέσω
παραμέτρων και για απομακρυσμένο έλεγχο του

Εκπόνηση:
Παπαδάμ Στέφανος
ΑΕΜ: 8885

Επίβλεψη:
Αν. Καθ. Συμεωνίδης Ανδρέας
Δρ. Τσαρδούλιας Εμμανουήλ

The robots are going to help us find our crystal...
— Dillon, *Hey Beau*

Είμαστε κάτι διάχυτες αισθήσεις,
χωρίς ελπίδα να συγκεντρωθούμε.
Στα νεύρα μας μπερδεύεται όλη η φύσις
— Κώστας Καρυωτάκης

ΕΥΧΑΡΙΣΤΙΕΣ

Με την ολοκλήρωση της συγκεκριμένης διπλωματικής ολοκληρώνεται και ο κύκλος μου ως προπτυχιακός φοιτητής στο Τμήμα Ηλεκτρολόγων Μηχανικών του ΑΠΘ. Το μεγαλύτερο ευχαριστώ το οφείλω στην οικογένεια μου για τη στήριξη, την αγάπη και το κουράγιο που έδειξαν όλα αυτά τα χρόνια βοηθώντας με έμπρακτα να πραγματοποιήσω τα όνειρα μου. Θα ήθελα, επίσης, να ευχαριστήσω τον Αναπληρωτή Καθηγητή κ. Ανδρέα Συμεωνίδη για την εμπιστοσύνη που έδειξε στο πρόσωπο μου αναθέτοντας μου το συγκεκριμένο θέμα εργασίας και επιτρέποντας μου να ασχοληθώ με τον τομέα των αυτόνομων οχημάτων που ανήκει στα ενδιαφέροντα μου. Θα ήθελα φυσικά να ευχαριστήσω και τον Μεταδιδακτορικό Ερευνητή Δρ. Εμμανουήλ Τσαρδούλια για τις συμβουλές, την καθοδήγηση και τις πολύτιμες συμβουλές που μου παρείχε σε όλη τη διάρκεια εκπόνησης της εργασίας. Κλείνοντας, ένα ειδικό ευχαριστώ αξιζουν οι φίλοι μου που έκαναν τα χρόνια μου ως φοιτητής ξεχωριστά και μου πρόσφεραν αξέχαστες και όμορφες στιγμές όλο αυτό το διάστημα.

Στέφανος Παπαδάμ

Περίληψη

Οι μετακινήσεις των ανθρώπων μέσα στη διάρκεια των χρόνων αποτελούσαν αναπόσπαστο κομμάτι της καθημερινότητας τους. Για τον λόγο αυτό από τον 18ο αιώνα άρχισαν οι προσπάθειες κατασκευής του πρώτου αυτοκινήτου. Μέσα στο πέρασμα των χρόνων οι βιομηχανίες κατασκεύαζαν όλο και πιο σύγχρονα αυτοκίνητα με αποτέλεσμα να προσφέρουν εύκολες και άνετες μετακινήσεις στους ανθρώπους. Τα τελευταία χρόνια η καινοτομία που ερευνάται εκτενώς από μεγάλες εταιρίες και πανεπιστήμια είναι η ανάπτυξη αυτόνομων οχημάτων τα οποία θα αλλάξουν οριστικά τον παραδοσιακό τρόπο λειτουργίας των αυτοκινήτων. Ορισμένα μόνο από τα προβλήματα στα οποία θα συμβάλλει δραστικά η τεχνολογία των αυτόνομων οχημάτων είναι η εξοικονόμηση σημαντικού χρόνου στην καθημερινότητα των ανθρώπων, η μείωση των τροχαίων ατυχημάτων και κατ' επέκταση η μεγαλύτερη ασφάλεια των μετακινήσεων, η συμβολή στην οικονομία καυσίμων και η ελάττωση της μόλυνσης του περιβάλλοντος.

Μέχρι σήμερα δεν έχει υπάρξει κατασκευή ενός πλήρως αυτόνομου οχήματος που να λειτουργεί χωρίς καμία παρέμβαση από τον άνθρωπο. Όταν κάτι τέτοιο συμβεί τότε το όχημα θα μπορεί να γνωρίζει πλήρως με μεγάλη ακρίβεια τις συνθήκες που επικρατούν στο εξωτερικό του περιβάλλον, θα λαμβάνει τις σωστές αποφάσεις σε κάθε χρονική στιγμή και θα μπορεί να ανταλλάσσει πληροφορίες με άλλα οχήματα έτσι ώστε να συνεργάζεται για την καλύτερη λειτουργία της κυκλοφορίας. Ωστόσο, απαιτείται μεγάλη προσπάθεια και έρευνα έτσι ώστε το όχημα να ανταποκρίνεται με επιτυχία στον μεγάλο αριθμό διαφορετικών σεναρίων που επικρατούν μέσα στην κίνηση. Η υλοποίηση ενός συστήματος αυτόνομης οδήγησης απαιτεί να λυθεί με επιτυχία το πρόβλημα της αντίληψης του εξωτερικού περιβάλλοντος, της επιλογής σωστής συμπεριφοράς και της ομαλής μετάβασης στον τελικό προορισμό του υπακούοντας τους κανόνες της κυκλοφορίας και αποφεύγοντας στατικά και δυναμικά εμπόδια. Για την επίλυση αυτών απαιτείται ο κατάλληλος εξοπλισμός αισθητήρων τελευταίας τεχνολογίας που θα δέχονται τα ερεθίσματα του περιβάλλοντος, θα αναλύονται από κάποια κεντρική μονάδα και θα παράγεται η σωστή απόφαση.

Η παρούσα διπλωματική εργασία ασχολείται με την ανάπτυξη μιας διεπαφής χρήστη που διαθέτει ένα αυτόνομο όχημα. Παράλληλα υλοποιεί το αυτόνομο όχημα πάνω στο οποίο εφαρμόζεται αυτή η διεπαφή. Η συγκεκριμένη διεπαφή δίνει απομακρυσμένες εντολές στο όχημα σχετικά με την κίνηση του και του παρέχει συγκεκριμένες παραμέτρους έτσι ώστε να προσαρμόζει την συμπεριφορά του ανάλογα με τις επιθυμίες του χρήστη. Το σύστημα της αυτόνομης οδήγησης που αναπτύχθηκε αποτελεί ένα μεμονωμένο σύστημα που χρησιμοποίησε μια παραλλαγή αρθρωτής αρχιτεκτονικής. Η υλοποίηση του συστήματος πραγματοποιείται σε γλώσσα πρограмματισμού Python και τα πειράματα εκτελούνται στον προσομοιωτή CARLA. Το αυτόνομο όχημα για τη λειτουργία του παρέχει λύσεις στα προβλήματα της αντίληψης, της σχεδίασης τροχιάς, της επιλογής συμπεριφοράς και του κινηματικού ελέγχου. Η διεπαφή υλοποιείται στο βασισμένο σε ροές γραφικό περιβάλλον NodeRED και επικοινωνεί με το σύστημα του αυτόνομου οχήματος με χρήση MQTT

πρωτοκόλλου και του μεσολαβητή μηνυμάτων Mosquitto. Με αυτό τον τρόπο σκοπός της διπλωματικής ήταν να δίνονται απομακρυσμένα εντολές στο αυτόνομο όχημα καθορίζοντας τον βαθμό νομιμότητας και επιθετικότητας που επιθυμεί ο χρήστης να έχει το όχημα του. Έγινε επομένως μία προσπάθεια να ενταχθεί το αυτόνομο όχημα στην ακμάζουσα τεχνολογία του Διαδικτύου των Πραγμάτων (IoT) αποτελώντας έναν από τους διάφορους κόμβους που συνδέονται μεταξύ τους σε ένα τέτοιο δίκτυο.

Το σύστημα ελέγχθηκε ως προς τη σωστή εκτέλεση των εντολών που δίνονται από την διεπαφή εξετάζοντας τη σωστή επικοινωνία μεταξύ τους και ως προς τη συμπεριφορά στην οποία προσαρμόζεται το όχημα όταν λαμβάνει τις τιμές νομιμότητας και επιθετικότητας από το χρήστη. Τα πειράματα εκτελέστηκαν στο περιβάλλον του CARLA για διάφορες τιμές των δύο παραμέτρων και η αποτελεσματικότητα του ελέγχθηκε ως προς τις παραβιάσεις, την ολοκλήρωση της διαδρομής και την ασφάλεια με την οποία κινήθηκε το όχημα.

Λέξεις κλειδιά: αυτόνομη οδήγηση, αυτόνομα οχήματα, απομακρυσμένος έλεγχος αυτοκινήτου, MQTT, CARLA, NodeRED

Στέφανος Παπαδάμ

stefanospapadam@gmail.com

Τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Γιπολογιστών

Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης

Σεπτέμβριος 2021

Title

Autonomous vehicle's interface development for adjusting behavior through parameters and for remote controlling

Abstract

The transportation of people over the years was an integral part of their daily lives. For this reason, the first efforts to manufacture a car began in the 18th century. Over the years, the industries were constructing more and more modern cars to offer the people easy and comfortable transportations. In recent years, the most known companies and universities research to create autonomous vehicles, which will change the way ordinary cars work. Some of the main problems that the technology of self-driving cars will contribute drastically are the saving of significant time in people's daily lives, the reduction of road accidents and consequently the safer transportations, the contribution to fuel economy and the reduction of environment's pollution.

To date, it has not been constructed a fully autonomous vehicle that works without any human intervention. When this technology becomes a reality, the vehicle will have a full and precise perception of the external environment's conditions, will make the right decisions every time, and will be able to exchange information with other vehicles to cooperate for better operation of the overall traffic. However, a lot of effort and research is demanded in the section of autonomous driving to create a vehicle that corresponds successfully to numerous scenarios and conditions that prevail inside the traffic. The implementation of such a system requires solving the problems of the external environment's perception, the right behavior choice, and the safe and smooth transition to the final destination by obeying the traffic rules and avoiding dynamic and static obstacles. To solve the aforementioned problems, it is needed suitable equipment that includes state-of-the-art sensors which will take as input the environment's measurements. The measurements will be analyzed by a central processing unit and finally, the right decision will be taken.

The specific thesis deals with the development of a user interface that could be used by an autonomous vehicle. Concurrently, an operational self-driving car is developed in which the user interface is adapted. The user interface can send remote commands to the autonomous vehicle about its motion and can provide it with particular parameters to adjust its behavior according to the user's desires. The developed autonomous driving system is an ego-only system that uses a variation of a modular architecture. The system's development is implemented by using Python programming language and the experiments are conducted in CARLA Simulator. To operate normally, the autonomous vehicle provides solutions for the problems of perception, path planning, behavior selection, and kinematic control. The user interface is implemented in the flow-based programming tool that is called NodeRED and communicates with the overall system by using the MQTT protocol and Mosquitto message broker. This way indicates that the purpose of the particular thesis is the ability to send remote commands to the autonomous vehicle, defining the rate of aggressiveness and legality that the user wants to have the vehicle. As a result, the autonomous vehicle can be included in the ascending technology of the Internet of Things, being part of a network with devices that are connected between them.

The system has been tested for its correct execution of the remote commands that are sent by the interface, examining the successful communication between them and the behavior's adaptation when the values of aggressiveness and legality are sent. The experiments were conducted in the environment of the CARLA simulator for different values of the parameters and the effectiveness has been tested as to the road violations, the route completion, and the safety the vehicle had during its motion.

Keywords: autonomous driving, autonomous vehicles, remote control vehicles, MQTT, CARLA, NodeRED

Stefanos Papadam
stefanospapadam@gmail.com
Electrical & Computer Engineering Department
Aristotle University of Thessaloniki, Greece
September 2021

Περιεχόμενα

Ευχαριστίες	iii
Περίληψη	v
Abstract	viii
Ακρωνύμια	xviii
1 Εισαγωγή	1
1.1 Περιγραφή του Προβλήματος	6
1.2 Σκοπός - Συνεισφορά της Διπλωματικής Εργασίας	6
1.3 Διάρθρωση της Αναφοράς	7
2 Επισκόπηση της Ερευνητικής Περιοχής	8
2.1 State-of-the-art συστήματα αυτόνομης οδήγησης	8
2.1.1 Εμπορικά συστήματα αυτόνομης οδήγησης	8
2.1.2 Ερευνητικά - Ακαδημαϊκά συστήματα αυτόνομης οδήγησης	11
2.2 Αρχιτεκτονικές σε συστήματα αυτόνομης οδήγησης	12
2.2.1 Μεμονωμένα και Συνδεδεμένα συστήματα	13
2.2.2 Αρθρωτές και Διατερματικές Μεθοδολογίες	13
2.3 Γλικό (hardware) σε συστήματα αυτόνομης οδήγησης	16
3 Εργαλεία Software	25
3.1 Προσομοιωτής CARLA	25
3.2 Εργαλείο κατασκευής της διεπαφής Node Red	27
3.3 Πρωτόκολλα επικοινωνίας	29
3.4 Γλώσσα προγραμματισμού Python	30
3.5 Βιβλιοθήκες και Κλάσεις	31
4 Γλοποιήσεις	34
4.1 Αρχιτεκτονική συστήματος	34
4.2 Περιγραφή της διεπαφής του NodeRED	35
4.2.1 Περιγραφή του γραφικού περιβάλλοντος	36
4.2.2 Περιγραφή των ροών του NodeRED (low level υλοποίηση)	48
4.3 Περιγραφή της λειτουργίας του Mosquitto broker	55
4.4 Περιγραφή των υλοποιήσεων του αυτόνομου οχήματος	56
4.4.1 Περιγραφή αρχιτεκτονικής του αυτόνομου οχήματος	56
4.4.2 Υποσύστημα Επικοινωνίας	57
4.4.3 Υποσύστημα Καθορισμού και Σχεδίασης Τροχιάς	59
4.4.4 Υποσύστημα Αντίληψης	66
4.4.5 Υποσύστημα Επιλογής Συμπεριφοράς	76

ΠΕΡΙΕΧΟΜΕΝΑ

4.4.6 Υποσύστημα Ελέγχου	102
5 Πειράματα - Αποτελέσματα	108
5.1 Εισαγωγή	108
5.2 Μετρικές Αξιολόγησης	108
5.3 Συνθήκες πειραμάτων	110
5.4 Αποτελέσματα Πειραμάτων	113
6 Συμπεράσματα	124
6.1 Γενικά Συμπεράσματα	124
6.2 Προβλήματα	124
6.3 Μελλοντικές επεκτάσεις	125
Βιβλιογραφία	128

Κατάλογος Σχημάτων

1.1	Επίπεδα αυτονομίας οχημάτων	3
2.1	Αρθρωτή (Modular) Αρχιτεκτονική σε μεμωνομένο (ego-only) σύστημα	14
2.2	Αρθρωτή (Modular) Αρχιτεκτονική σε συνδεδεμένο (connected) σύστημα	14
2.3	Εικόνα που λαμβάνει η κάμερα	18
2.4	Όχημα εξοπλισμένο με Radar	19
2.5	Όχημα εξοπλισμένο με Lidar	20
2.6	Όχημα εξοπλισμένο με υπερηχητικό (ultrasonic) αισθητήρα	21
2.7	Τρόπος λειτουργίας του GPS	22
2.8	Αυτόνομο όχημα της Waymo εξοπλισμένο με τους αισθητήρες	23
2.9	Όχημα με τις θέσεις και το εύρος λειτουργίας κάθε αισθητήρα	23
3.1	Η αρχιτεκτονική πελάτη - εξυπηρετητή του CARLA	26
3.2	Το περιβάλλον του CARLA σε τυχαίο χάρτη	27
3.3	Το γραφικό περιβάλλον του Node-RED	28
3.4	Ροή στο περιβάλλον του NodeRED	29
3.5	Ο τρόπος λειτουργίας του MQTT πρωτοκόλλου	30
3.6	Οι βιβλιοθήκες της Python που χρησιμοποιήθηκαν	33
4.1	Η αρχιτεκτονική της συγκεκριμένης διπλωματικής	36
4.2	Η διεπαφή του συστήματος	37
4.3	Το σύνολο οδηγιών της διεπαφής	38
4.4	Η πόλη του CARLA με τα σημεία ενδιαφέροντος	40
4.5	Dropdown μενού επιλογής τοποθεσίας ή κατεύθυνσης	40
4.6	Επιλογές κατεύθυνσης σε τυχαία διασταύρωση	41
4.7	Μηνύματα της διεπαφής σε διάφορες περιπτώσεις	42
4.8	Διασταύρωση με πολλαπλές επιλογές κατεύθυνσης	42
4.9	Διαφορετικά μονοπάτια για την επιλογή 30 μέτρων FORWARD	43
4.10	Τα πλήκτρα FORWARD, RIGHT TURN, LEFT TURN και STRAIGHT	44
4.11	Τα πλήκτρα ENTER, DONE και CANCEL	44
4.12	Τυχαία τροχιά κίνησης	46
4.13	Η καρτέλα Handle Vehicle της διεπαφής	47
4.14	Η καρτέλα Behavior της διεπαφής	48
4.15	JSON μεταβλητή	50
4.16	Ροές στο NodeRED για την υλοποίηση των πλήκτρων	51
4.17	Ροές στο NodeRED για την υλοποίηση των ειδοποιήσεων	53

ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

4.18 Ροές στο NodeRED για την υλοποίηση των sliders	54
4.19 Ροές στο NodeRED για την υλοποίηση των dropdown μενού	55
4.20 Η λογική Black Box	56
4.21 Αρχιτεκτονική Αυτόνομου Συστήματος	57
4.22 Τα waypoints μιας τυχαίας διαδρομής	61
4.23 Κατευθυνόμενος γράφος του CARLA	64
4.24 Βέλτιστες διαδρομές υπολογισμένες μέσω του A*	65
4.25 Το σύστημα αντίληψης του αυτοκινήτου	70
4.26 Τα σημεία ενδιαφέροντος πάνω στο αυτοκίνητο	73
4.27 Απεικόνιση κοντινότερων οχημάτων - εμποδίων.	75
4.28 Παράδειγμα επιλογής συμπεριφοράς.	78
4.29 Παράδειγμα παράλειψης κάποιων waypoints για διαφορετικές ταχύτητες.	90
4.30 Έξοδος PID ελεγκτή	105
4.31 Διάγραμμα ενός συστήματος με PID ελεγκτή	105
5.1 Η τοπολογία της πόλης στην οποία εκτελέστηκαν τα πειράματα	111
5.2 Η κάτοψη της πόλης με τις λεπτομέρειες της	112
5.3 Ραβδόγραμμα για την μετρική "Βαθμολογία Οδήγησης"	121
5.4 Ραβδόγραμμα για την μετρική "Ολοκλήρωση Διαδρομής"	121
5.5 Ραβδόγραμμα για την μετρική "Ποινή Παραβάσεων"	122
5.6 Ραβδόγραμμα για την μετρική "Αριθμός δεξιών αλλαγών λωρίδας"	122
5.7 Ραβδόγραμμα για την μετρική "Αριθμός αριστερών αλλαγών λωρίδας"	123
5.8 Ραβδόγραμμα για την μετρική "Μέση Ταχύτητα"	123

Κατάλογος πινάκων

4.1 Πίνακας βαρών των κανόνων (γραμμές) σε σχέση με τις συμπεριφορές (στήλες), από τον οποίο προκύπτει η βέλτιστη συμπεριφορά κάθε στιγμή	87
4.2 Πίνακας που απεικονίζει τις σχετικές ταχύτητες δύο οχημάτων και τις αντίστοιχες ασφαλείς αποστάσεις που πρέπει να διατηρούν.	88
4.3 Τα τελικά κέρδη του Longitudinal ελεγκτή.	105
4.4 Τα τελικά κέρδη του Lateral ελεγκτή.	106
5.1 Πίνακας με τις τιμές των παραβάσεων. Οι μικρότερες τιμές αντιστοιχούν και σε μεγαλύτερη παράβαση.	110
5.2 Μετρικές αξιολόγησης αυτονομίας οδήγησης για διαφορετικές τιμές των δύο sliders και κανένα δυναμικό εμπόδιο.	113
5.3 Μετρικές αξιολόγησης της ταχύτητας και του αριθμού αλλαγών λωρίδας για διαφορετικές τιμές των δύο sliders και κανένα δυναμικό εμπόδιο.	114
5.4 Μετρήσεις των αριθμών κάθε παράβασης για διαφορετικές τιμές των δύο sliders και κανένα δυναμικό εμπόδιο.	114
5.5 Μετρικές αξιολόγησης αυτονομίας οδήγησης για διαφορετικές τιμές των δύο sliders με 10 οχήματα και 10 πεζούς.	115
5.6 Μετρικές αξιολόγησης της ταχύτητας και του αριθμού αλλαγών λωρίδας για διαφορετικές τιμές των δύο sliders με 10 οχήματα και 10 πεζούς.	115
5.7 Μετρήσεις των αριθμών κάθε παράβασης για διαφορετικές τιμές των δύο sliders με 10 οχήματα και 10 πεζούς.	116
5.8 Μετρικές αξιολόγησης αυτονομίας οδήγησης για διαφορετικές τιμές των δύο sliders με 30 οχήματα και 30 πεζούς.	116
5.9 Μετρικές αξιολόγησης της ταχύτητας και του αριθμού αλλαγών λωρίδας για διαφορετικές τιμές των δύο sliders με 30 οχήματα και 30 πεζούς.	117
5.10 Μετρήσεις των αριθμών κάθε παράβασης για διαφορετικές τιμές των δύο sliders με 30 οχήματα και 30 πεζούς.	117
5.11 Μετρικές αξιολόγησης αυτονομίας οδήγησης για διαφορετικές τιμές των δύο sliders με 60 οχήματα και 60 πεζούς.	118
5.12 Μετρικές αξιολόγησης της ταχύτητας και του αριθμού αλλαγών λωρίδας για διαφορετικές τιμές των δύο sliders με 60 οχήματα και 60 πεζούς.	118

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

5.13 Μετρήσεις των αριθμών κάθε παράβασης για διαφορετικές τιμές των δύο sliders με 60 οχήματα και 60 πεζούς.	119
5.14 Μετρικές αξιολόγησης αυτονομίας οδήγησης για διαφορετικές τιμές των δύο sliders με 90 οχήματα και 90 πεζούς.	119
5.15 Μετρικές αξιολόγησης της ταχύτητας και του αριθμού αλλαγών λωρίδας για διαφορετικές τιμές των δύο sliders με 90 οχήματα και 90 πεζούς.	120
5.16 Μετρήσεις των αριθμών κάθε παράβασης για διαφορετικές τιμές των δύο sliders με 90 οχήματα και 90 πεζούς.	120

Κατάλογος Αλγορίθμων

4.1	Class PublisherMQTT	59
4.2	Class SubscriberMQTT	60
4.3	Handle Turn	67
4.4	Specify Destination	68
4.5	Find Front and Rear Vehicles	72
4.6	Recognize Traffic Lights	74
4.7	Recognize Traffic Signs	74
4.8	Calculate Current Velocity	80
4.9	Right / Left Lane Change	82
4.10	Speed Up	83
4.11	Slow Down	83
4.12	Keep Velocity	84
4.13	Stop	84
4.14	Check New Triggers	97
4.15	Evaluate	98
4.16	Apply Maneuver	99
4.17	Regulate Speed	101
4.18	Follow Trajectory	103

Ακρωνύμια Εγγράφου

Παρακάτω παρατίθενται ορισμένα από τα πιο συχνά χρησιμοποιούμενα ακρωνύμια της παρούσας διπλωματικής εργασίας:

API	→ Application Programming Interface
GPS	→ Global Positioning System
IMU	→ Inertial measurement unit
IoT	→ Internet of Things
LiDAR	→ Light Detection and Ranging
MCDM	→ Multiple-criteria decision-making
MQTT	→ Message Queuing Telemetry Transport
PID	→ Proportional Integral Derivative
ROS	→ Robot Operating System

1

Εισαγωγή

Η σύγχρονη δομή της κοινωνίας και οι γρήγοροι ρυθμοί της έχουν αθήσει τον άνθρωπο στην ανάγκη να δημιουργήσει τρόπους και μέσα τα οποία θα του επιτρέπουν να μετακινείται ευκολότερα και γρηγορότερα σε μακρινότερους προορισμούς. Στη σημερινή εποχή για τους περισσότερους ανθρώπους οι μετακινήσεις με κάποιο μεταφορικό μέσο αποτελούν σημαντικό και συχνά απαραίτητο κομμάτι της καθημερινότητας τους προκειμένου να εξυπηρετήσουν τις προσωπικές τους ανάγκες. Είναι γενικότερα γνωστό πως ο άνθρωπος αναζητούσε συνεχώς καλύτερους τρόπους για να εξυπηρετεί τις καθημερινές ανάγκες μετακίνησης του ξεκινώντας από ζωήλατα οχήματα και καταλήγοντας στα σημερινά σύγχρονα οχήματα. Αν και οι πρώτες προσπάθειες δημιουργίας αυτοκινήτου χρονολογούνται περίπου στο 1670 όπου εμφανίστηκαν τα πρώτα ατμήλατα, το μεγάλο βήμα για την κατασκευή του πρώτου αυτοκινήτου έγινε το 1886 από τον Καρλ Μπενζ ο οποίος θεωρείται ο εφευρέτης του πρώτου σύγχρονου αυτοκινήτου¹ που λειτουργούσε με κινητήρα. Τα αυτοκίνητα έφεραν μεγάλη άνεση στη ζωή των ανθρώπων προσδίδοντας ανεξαρτησία στις μετακινήσεις τους και μειώνοντας σημαντικά το χρόνο μεταφορά τους. Επίσης, η μεγάλη ανάπτυξη στον τομέα της αυτοκινητοβιομηχανίας είχε ως αποτέλεσμα το αυτοκίνητο να μη θεωρείται πλέον ένα πολυτελές μέσο που μπορούσαν να αντέξουν μόνο οι ευκατάστατες ομάδες ανθρώπων αλλά ένα μέσο προσιτό σχεδόν σε κάθε καταναλωτή.

Ωστόσο, παρά τις μειωμένες σε χρόνο αποστάσεις, τις άμεσες δυνατότητες για ταξίδια αναψυχής και εργασίας και την ευκολία στη χρήση τους, ο αριθμός των αυτοκινήτων που χυκλοφορούν στους δρόμους έχει αυξηθεί σημαντικά δημιουργώντας μεγάλα προβλήματα χυκλοφοριακής συμφόρησης, μεγάλες αναμονές σε πολυσύχναστους δρόμους, μόλυνση του περιβάλλοντος, σπατάλη ενέργειας και φυσικών πόρων, κωλυσιεργίες σε καθημερινές δραστηριότητες και φυσικά αύξηση των τροχαίων ατυχημάτων. Μάλιστα, στο τελευταίο έχει συμβάλει σημαντικά και ο ανθρώπινος παράγοντας καθώς όλο και περισσότερα αυτοχήματα προκαλούνται από την

¹<https://en.wikipedia.org/wiki/Car>

ΚΕΦΑΛΑΙΟ 1. ΕΙΣΑΓΩΓΗ

κόπωση, την κατανάλωση αλκοόλ, την παραβίαση των σηματοδοτών του δρόμου και των λοιπών κανόνων κυκλοφορίας από τους οδηγούς προκαλώντας σοβαρά τροχαία ατυχήματα που κοστίζουν υλικές ζημιές αλλά και ανθρώπινες ζωές. Συγκεκριμένα το 2014 εκτιμάται ότι από τα 32,675 ατυχήματα που προκλήθηκαν το 94% αυτών οφείλεται σε λάθος του οδηγού, το 31% συμπεριλαμβανε μεθυσμένους οδηγούς και το 10% αφηρημένους οδηγούς [1].

Για τους λόγους αυτούς μεγάλες εταιρίες κατασκευής αυτοκινήτων και πανεπιστήμια έχουν στρέψει το ενδιαφέρον της έρευνας τους προς την κατασκευή αυτόνομων οχημάτων. Ο όρος αυτόνομο όχημα αναφέρεται σε ένα όχημα το οποίο ελέγχεται από κάποιον υπολογιστή και έχει τη δυνατότητα να πλοηγείται με ασφάλεια στους δρόμους χωρίς να συγκρούεται με στατικά ή δυναμικά εμπόδια, να εξικειώνεται με το περιβάλλον του μέσω αισθητήρων, να είναι σε θέση να παίρνει αποφάσεις για τη συμπεριφορά που θα ακολουθήσει και να λειτουργεί πλήρως με μικρή ή καμία ανθρώπινη παρέμβαση [2].

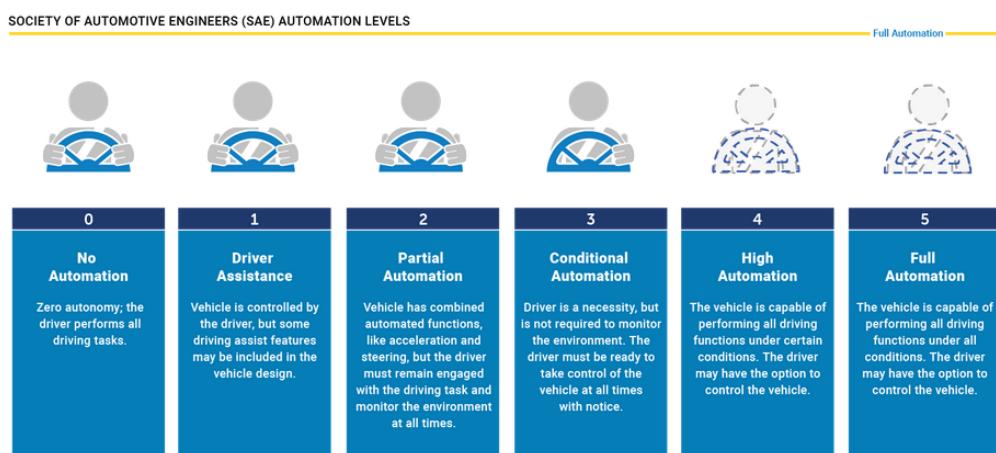
Η τεχνολογία των αυτόνομων οχημάτων που αναπτύσσεται συνεχώς παρέχει την προοπτική να μειώσει σημαντικά τα ατυχήματα που προκαλούνται από ανθρώπινο λάθος σώζοντας ένα μεγάλο αριθμό ανθρώπινων ζωών είτε πρόκειται για οδηγούς, επιβάτες είτε για πεζούς ή δικυκλιστές. Αρκεί να αναλογιστεί κανείς ότι πάνω από 35,000 άνθρωποι χάνουν τη ζωή τους από τροχαίο ατύχημα στις Η.Π.Α κάθε χρόνο, για να καταλάβει πόσο κομβική θα είναι η συμβολή της αυτόνομης οδήγησης στη ζωή των ανθρώπων² ³. Επίσης, πέρα από την ασφάλεια θα μπορούν να προσφέρουν σημαντικές διευκολύνσεις στην προσωπική ζωή των ανθρώπων καθώς δίνουν τη δυνατότητα σε άτομα με κινητικά προβλήματα ή προβλήματα όρασης αλλά και άτομα που δεν επιθυμούν να γνωρίζουν τον χειρισμό ενός αυτοκινήτου να μετακινούνται ανεξάρτητα με το δικό τους όχημα χωρίς να εξαρτάται η ζωή και η μετακίνηση τους από άλλα πρόσωπα. Η οικονομία είναι ένας ακόμη τομέας που μπορεί να επηρεαστεί σημαντικά από τη συγκεκριμένη τεχνολογία μειώνοντας την περιττή κατανάλωση καυσίμων χρησιμοποιώντας τα αυτοκίνητα με οικονομικότερο τρόπο αλλά και μειώνοντας τα κόστη από τις υλικές ζημιές των ατυχημάτων. Παράλληλα, λειτουργώντας το συνολικό δίκτυο των αυτόνομων οχημάτων με μια πιο ομαλή ροή της κυκλοφορίας μπορεί να μειώσει το χρόνο που ξοδεύει ο άνθρωπος στη διαδικασία της οδήγησης και να του προσφέρει μεγαλύτερη παραγωγικότητα στην καθημερινή του ζωή απαλλάσσοντας τον παράλληλα από το περιττό στρες που προκαλείται μέσω αυτής της καθημερινής συνήθειας [3]. Αυτός ο χρόνος θα μπορεί να εξικονομείται από την προσωπική ζωή των ανθρώπων αποφεύγοντας χρονοβόρες διαδικασίες που περιλαμβάνει η οδήγηση όπως η αναζήτηση στάθμευσης του αυτοκινήτου τους αλλά και η επιστροφή τους εκεί. Επιπρόσθετα, θα δώσει τη δυνατότητα να αυτοματοποιηθούν συγκεκριμένες εργασίες όπως οι διανομές προϊόντων οι οποίες σήμερα πραγματοποιούνται με μεγάλα φορτηγά οχήματα δίνοντας τη δυνατότητα να εκτελούνται γρηγορότερα και ασφαλέστερα. Τα μέσα μαζικής μεταφοράς αποτελούν έναν τομέα που μπορεί να επηρεαστεί άμεσα από τη συγκεκριμένη καινοτομία παρέχοντας την ευελιξία για ποιοτικότερη, ασφαλέστερη και γρηγορότερη μεταφορά των ανθρώπων στους προορισμούς τους.

Κάποιες από τις λειτουργίες που έχουν ήδη αναπτυχθεί και συμβάλλουν στην

²<https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety>

³<https://www.synopsys.com/automotive/autonomous-driving-levels.html>

καλύτερη οδηγική εμπειρία είναι το αυτόματο παρκάρισμα [4], η διατήρηση της λωρίδας κυκλοφορίας [5], η διατήρηση της ταχύτητας του αυτοκινήτου και η αυτόματη προσπέραση του μπροστινού οχήματος [6]. Είναι εμφανές λοιπόν πως πρόκειται για λειτουργίες που βοηθούν στην οδήγηση του οχήματος αλλά δεν προσφέρουν ένα πλήρες μοντέλο αυτόνομης οδήγησης όπου δε χρειάζεται η ανθρώπινη παρέμβαση. Για να γίνει περισσότερο κατανοητός αυτός ο διαχωρισμός ανάμεσα στις διαφορετικές λειτουργίες που προσφέρει ένα αυτόνομο όχημα, έχουν οριστεί 6 επίπεδα αυτονομίας από την Society of Automotive Engineers (SAE)⁴. Ο βαθμός αυτός είναι μεγαλύτερος για τα οχήματα που εκτελούν περισσότερες αυτόνομες λειτουργίες⁵. Παρακάτω απεικονίζονται (σχήμα 1.1) και περιγράφονται συνοπτικά τα έξι επίπεδα αυτονομίας που μπορεί να διαθέτει ένα όχημα:



Σχήμα 1.1: Επίπεδα αυτονομίας και οι βασικές λειτουργίες του κάθε επιπέδου.

Πηγή: <https://www.safercar.gov/technology-innovation/automated-vehicles-safety>

- **Επίπεδο 0 (Καμία αυτονομία)** Τα περισσότερα οχήματα σήμερα ανήκουν σε αυτό το επίπεδο αυτονομίας. Πρόκειται για τα οχήματα στα οποία ο οδηγός είναι υπεύθυνος για όλες τις λειτουργίες που εκτελεί το όχημα όπως αλλαγή ταχυτήτων, πέδηση, αύξηση ή μείωση της ταχύτητας και αλλαγές κατεύθυνσης. Η αυτονομία που παρέχει αυτό το επίπεδο περιορίζεται σε λειτουργίες του τύπου προειδοποιήσεις για ενδεχόμενες συγκρούσεις, προειδοποιήσεις για τα τυφλά σημεία (blind spots) του δρόμου όπου δεν υπάρχει ορατότητα από τον οδηγό ή κάποιο σύστημα επείγουσας πέδησης (emergency braking system) σε περίπτωση απότομου φρεναρίσματος. Όπως γίνεται αντιληπτό αυτές οι διαδικασίες δεν προσφέρουν κάποια αυτοματοποιημένη λειτουργία στο όχημα αλλά είναι βοηθητικές στον χειροκίνητο χειρισμό (manually controlled) από τον οδηγό.
- **Επίπεδο 1 (Υποβοήθηση του οδηγού / "hands on")** Αυτό το επίπεδο αποτελεί το χαμηλότερο επίπεδο αυτονομίας που μπορεί να διαθέτει ένα όχημα, δεδομένου ότι το προηγούμενο δεν προσφέρει κάποια αυτόματη λειτουργία όπως

⁴<https://www.sae.org/>

⁵<https://www.synopsys.com/automotive/autonomous-driving-levels.html>

αναφέρθηκε. Τα οχήματα αυτού του επιπέδου διαθέτουν κάποιο σύστημα προσαρμογής της ταχύτητας ή της κατεύθυνσης (adaptive cruise control) διατηρώντας ασφαλείς αποστάσεις από το μπροστινό όχημα ή η διατήρηση της λωρίδας στην οποία βρίσκεται το όχημα (lane keeping system). Ο οδηγός έχει τον πλήρη έλεγχο του αυτοκινήτου και είναι υπεύθυνος για την ταχύτητα και την κατεύθυνση του οχήματος αλλά μπορεί να βοηθηθεί από κάποια από τις παραπάνω λειτουργίες. Οι παραπάνω υπηρεσίες εκτελούνται ξεχωριστά και δεν μπορούν να συνδυαστούν.

- **Επίπεδο 2 (Μερική αυτοματοποίηση οδήγησης / "hands off")** Το επίπεδο αυτό προσφέρει ένα προηγμένο σύστημα υποβοηθούμενης οδήγησης (ADAS) στο οποίο το όχημα μπορεί να ελέγχει την ταχύτητα και την κατεύθυνση του ταυτόχρονα χωρίς την βοήθεια του οδηγού κάτω υπό ορισμένες συνθήκες. Ωστόσο, ο οδηγός θα πρέπει συνεχώς να παρακολουθεί το περιβάλλον στο οποίο κινείται το όχημα έτσι ώστε να πάρει τον έλεγχο του αυτοκινήτου αμέσως σε περίπτωση που χρειαστεί. Η ευθύνη για την συνολική διαδικασία κίνησης του αυτοκινήτου ανήκει στον οδηγό του οχήματος.
- **Επίπεδο 3 (Υπό όρους αυτοματοποιημένη οδήγηση / "eyes off")** Το όχημα διαθέτει ένα αυτοματοποιημένο σύστημα οδήγησης (ADS) το οποίο μπορεί να εκτελέσει όλες τις διαδικασίες που εκτελεί ένα όχημα χωρίς την παρέμβαση του οδηγού υπό συγκεκριμένες συνθήκες. Διαθέτει δυνατότητες αντίληψης του περιβάλλοντος και μπορεί να εκτελέσει ολοκληρωμένες λειτουργίες όπως αυτόματη προσπέραση προπορευόμενου οχήματος, η αλλαγή λωρίδας κίνησης και η ολοκλήρωση μιας διαδρομής με αρχικό και τελικό προορισμό. Παρ' όλα αυτά, ο οδηγός θα πρέπει να βρίσκεται συνεχώς σε επαγρύπνηση σε περίπτωση που οι συνθήκες αλλάξουν και το όχημα του ζητήσει να παρέμβει.
- **Επίπεδο 4 (Αυτοματοποίηση υψηλού επιπέδου / "mind off")** Σε αυτό το επίπεδο το όχημα αναλαμβάνει όλες τις λειτουργίες που εκτελεί και ένας οδηγός. Η διαφορά με το επίπεδο 3 είναι ότι εδώ υπάρχουν λειτουργίες για έκτακτες καταστάσεις που μπορεί να βρεθεί το όχημα φροντίζοντας από μόνο του για τον σωστό έλεγχο του αυτοκινήτου και την ασφάλεια των επιβατών. Στις περισσότερες περιπτώσεις δε χρειάζεται καθόλου ανθρώπινη παρέμβαση αλλά και πάλι το σύστημα εξακολουθεί να λειτουργεί υπό συγκεκριμένες συνθήκες δίνοντας για αυτό το λόγο τη δυνατότητα στον οδηγό να παρέμβει σε περίπτωση που κριθεί αναγκαίο. Τα συστήματα αυτά μέχρι τώρα είναι πλήρως λειτουργικά σε λιγότερο περίπλοκα περιβάλλοντα και σε λιγότερο ακραίες καιρικές συνθήκες. Εξασφαλίζοντας ένα τέτοιο περιβάλλον το σύστημα μπορεί να λειτουργήσει άριστα.
- **Επίπεδο 5 (Πλήρης αυτοματοποίηση / "steering wheel optional")** Τα οχήματα αυτού του επιπέδου είναι ικανά να πλοηγηθούν με ασφάλεια και να ανταποκριθούν σε οποιεσδήποτε συνθήκες και οποιαδήποτε σενάρια κίνησης του αυτοκινήτου. Το όχημα εξασφαλίζει πλήρως την ασφάλεια των επιβατών και δε χρειάζεται καμία παρέμβαση από τους επιβαίνοντες. Επίσης, οι επιβαίνοντες δε χρειάζεται να παρακολουθούν το περιβάλλον κατά τη διάρκεια

της διαδρομής ούτε να παρέμβουν κάποια στιγμή. Τα συγκεκριμένα οχήματα μπορούν να λειτουργήσουν και χωρίς επιβάτες δηλαδή δίνοντας τη δυνατότητα απλά να τα κληθούν στο μέρος των επιβατών. Η κατασκευή τους δεν περιλαμβάνει κάποιο σύστημα με το οποίο μπορεί να παρέμβει ο άνθρωπος όπως σύστημα πέδησης/επιτάχυνσης ή τιμόνι.

Παρά τα μεγάλα οφέλη που μπορεί να επιφέρει η συγκεκριμένη τεχνολογία στις ζωές των ανθρώπων και εκτός από τις τεχνικές δυσκολίες που υπάρχουν μέχρι την ολοκλήρωση της, ένα μεγάλο θέμα που τίθεται υπό συζήτηση τα τελευταία χρόνια είναι το νομικό πλαίσιο κάτω από το οποίο πρόκειται να λειτουργήσουν τα αυτόνομα οχήματα. Ο λόγος που γίνεται αυτή η συζήτηση αφορά κυρίως την απόδοση ευθυνών σε περίπτωση που το σύστημα ενός αυτόνομου οχήματος δεν έχει την αναμενόμενη ανταπόκριση σε κάποιο σενάριο της κυκλοφορίας και προκληθεί ατύχημα. Ήδη έχουν θεσμοθετηθεί κάποιοι κανόνες στις Η.Π.Α και στην Ευρώπη σχετικά με την εύρυθμη λειτουργία αυτών των οχημάτων. Από το 2012 και έπειτα κάποιες από τις πολιτείες των Η.Π.Α όπως η Νεβάδα, το Μίσιγκαν, η Φλόριντα, η Καλιφόρνια και η Ουάσινγκτον νομιμοποίησαν τη χρήση αυτόνομων οχημάτων για συλλογή δεδομένων και έλεγχο της ορθής λειτουργίας τους (testing). Παρόμοια και στην Ευρώπη, η Γαλλία και το Ηνωμένο Βασίλειο επέτρεψαν την κυκλοφορία αυτόνομων οχημάτων σε δημόσιους αυτοκινητόδρομους για την δοκιμή της λειτουργίας τους από το 2013 και έπειτα, προϋποθέτοντας σε κάθε περίπτωση την υποχρεωτική ύπαρξη οδηγού έτοιμου να επέμβει σε περίπτωση έκτακτης ανάγκης⁶.

Μέχρι το επίπεδο αυτονομίας 3 από τα επίπεδα που έχει προτείνει η SAE μπορεί στο συνολικό σύστημα του αυτόνομου οχήματος να προστεθεί μια λειτουργική διεπαφή με την οποία θα μπορεί ο χρήστης να αλληλεπιδρά με το όχημα του. Γίνεται σαφές ότι όσο μεγαλύτερη λειτουργικότητα έχει αυτή η διεπαφή τόσο μειώνεται η αυτονομία του οχήματος. Φυσικά, τα αυτόνομα οχήματα επίπεδου 4 και 5 μπορούν να διαθέτουν μια διεπαφή που να προσφέρει καθαρά ενημερωτικές πληροφορίες στο χρήστη. Με αυτό τον τρόπο υπάρχει η δυνατότητα το αυτόνομο όχημα να ενταχθεί σε μια νέα τεχνολογία που ακμάζει τα τελευταία χρόνια και ονομάζεται Διαδίκτυο των Πραγμάτων⁷. Ο όρος Διαδίκτυο των Πραγμάτων ή Internet of Things (IoT) αναφέρεται σε ένα δίκτυο επικοινωνίας στο οποίο συνδέεται ένα σύνολο συσκευών με σκοπό να ανταλλάσουν δεδομένα και πληροφορίες και να συνεργάζονται για κάποια διαδικασία. Οι συσκευές που συνδέονται στο δίκτυο μπορεί να είναι οικιακές συσκευές, οχήματα και κάθε άλλο ηλεκτρονικό μέσο που μπορεί να συνδέεται στο διαδίκτυο όπως λογισμικό, αισθητήρες, συστήματα ασφαλείας κ.α. Με αυτό τον τρόπο δημιουργείται ένα δίκτυο με ένα σύνολο συνδεδεμένων συσκευών σε αυτό τις οποίες ο χρήστης μπορεί να χειρίζεται απομακρυσμένα και να τους αναθέτει διάφορα καθήκοντα. Η φιλοσοφία του IoT στηρίζεται στην ύπαρξη μιας πλατφόρμας όπου θα συγκεντρώνονται οι λειτουργικότητες των διαφόρων συσκευών και θα μοιράζονται οι κατάλληλες εντολές. Μέσα σε αυτή την πλατφόρμα μπορεί να ενταχθεί η διεπαφή ενός αυτόνομου οχήματος από την οποία ο χρήστης έχει τη δυνατότητα να χειρίζεται το όχημα του είτε απομακρυσμένα είτε εντός του οχήματος.

⁶<https://en.wikipedia.org/wiki/Car>

⁷https://en.wikipedia.org/wiki/Internet_of_things

1.1 ΠΕΡΙΓΡΑΦΗ ΤΟΥ ΠΡΟΒΛΗΜΑΤΟΣ

Το πρόβλημα της αυτόνομης οδήγησης απασχολεί την επιστημονική κοινότητα εδώ και αρκετά χρόνια προσπαθώντας συνεχώς να προσφέρει καλύτερες και ποιοτικότερες λύσεις στις μετακινήσεις των ανθρώπων και στην οδηγική τους εμπειρία. Για να φτάσει στο σημείο ένα όχημα να θεωρείται πλήρως αυτόνομο θα πρέπει να αντιμετωπίσει επιτυχώς ενδιάμεσα υποπροβλήματα τα οποία θα το οδηγήσουν στον τελικό προορισμό με ασφάλεια παίρνοντας κάθε στιγμή τις σωστές αποφάσεις. Κάποια από αυτά τα υποπροβλήματα της αυτόνομης οδήγησης είναι η αντίληψη που έχει για το περιβάλλον στο οποίο κινείται (perception), η θέση του στην ευρύτερη περιοχή που θα κινηθεί (localization), συνήθως αυτή η θέση αφορά την τοποθεσία μέσα σε ένα χάρτη πόλης, χώρας κλπ., η διαδρομή που θα ακολουθήσει για να φτάσει στον τελικό προορισμό του (global planning) τηρώντας τους κανόνες κυκλοφορίας που ορίζει η συγκεκριμένη πόλη. Στο σημείο αυτό έρχεται να προστεθεί ένα ακόμη πρόβλημα που συνδέεται με το global planning και αφορά την ολοκλήρωση της διαδρομής τηρώντας τους κανόνες κυκλοφορίας και σεβόμενο τα υπόλοιπα οχήματα και τους πεζούς (local planning) επιλέγοντας ανά τακτά χρονικά διαστήματα την καλύτερη τοπική κατεύθυνση και ταχύτητα για την ολοκλήρωση της συνολικής διαδρομής.

Παράλληλα, είναι ιδιαίτερα σημαντικό το όχημα να παρέχει κατάλληλες πληροφορίες και προειδοποιήσεις κυρίως στον οδηγό του οχήματος σε περίπτωση που χρειαστεί να επέμβει σε κάποια έκτακτη κατάσταση ή απλά να τον ενημερώσει για την κατάσταση και τις συνθήκες στις οποίες βρίσκεται το αυτοκίνητο εκείνη τη στιγμή. Τέτοιου είδους πληροφορίες μπορεί να παρέχει και στους εξωτερικούς παράγοντες που μπορεί να επηρεάσουν την κίνηση του οχήματος όπως οι πεζοί και τα υπόλοιπα οχήματα ενημερώνοντας τους για τις κινήσεις που πρόκειται να πραγματοποιήσει. Φυσικά, όπως γίνεται κατανοητό ένα τέτοιο σύστημα αναφέρεται σε οχήματα που βρίσκονται μέχρι το τέταρτο επίπεδο αυτονομίας διότι όπως αναφέρθηκε και παραπάνω στο πέμπτο επίπεδο το αυτόνομο όχημα στηρίζεται αποκλειστικά στη δική του ικανότητα και αντίληψη για το περιβάλλον και δε χρειάζεται καμία παρέμβαση από τον υποτιθέμενο οδηγό. Η μόνη πληροφορία που θα μπορούσε να παρέχει σε ένα όχημα πέμπτου επιπέδου θα μπορούσε να είναι καθαρά ενημερωτική για τους επιβάτες χωρίς να τους προτρέπει να επέμβουν. Το μέσο για να καταστεί δυνατή αυτή η αλληλεπίδραση του αυτοκινήτου με τον οδηγό και τους υπόλοιπους επιβαίνοντες είναι συνήθως μια διεπαφή (interface) που διαθέτει το όχημα και επιτρέπει στον άνθρωπο να παρεμβαίνει μεταβάλλοντας την κατάσταση του οχήματος όπως την κατεύθυνση και την ταχύτητα ή απλά να ενημερώνεται για το εξωτερικό περιβάλλον του αυτοκινήτου.

1.2 ΣΚΟΠΟΣ - ΣΥΝΕΙΣΦΟΡΑ ΤΗΣ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

Η παρούσα διπλωματική έχει σκοπό να δημιουργήσει μια διεπαφή (interface) η οποία θα ενσωματώνεται σε ένα σύστημα αυτόνομης οδήγησης και θα χρησιμοποιείται από τον οδηγό για να καθορίσει διάφορες παραμέτρους και συμπεριφορές που επιθυμεί να ακολουθήσει το όχημα κατά τη διάρκεια της κίνησης του. Το σύ-

στημα αυτόνομης οδήγησης θα μπορεί να ανταποκρίνεται στα περισσότερα σενάρια κίνησης και στις περισσότερες συνθήκες χωρίς να προκαλεί συγκρούσεις με άλλα στατικά ή δυναμικά εμπόδια οδηγώντας τους επιβάτες με ασφάλεια στον τελικό προορισμό τους. Για να είναι πλήρως λειτουργικό ένα τέτοιο σύστημα θα πρέπει να προσφέρει μία ικανοποιητική λύση στα υπόλοιπα υποπροβλήματα της αυτόνομης οδήγησης και να λαμβάνει τις αποφάσεις αρκετά γρήγορα ώστε να αποφεύγει πιθανά ατυχήματα. Σε αυτό το σύστημα ενσωματώνεται και η προαναφερόμενη διεπαφή η οποία θα δίνει τη δυνατότητα να επικοινωνεί ο οδηγός με το όχημα. Η συγκεκριμένη διπλωματική προσπαθεί να ενσωματώσει το αυτόνομο όχημα μέσα στο Διαδίκτυο των Πραγμάτων⁸ (IoT) και να αποτελέσει μέρος αυτού ανάμεσα στα υπόλοιπα στοιχεία που πιθανώς να συνδέονται απομακρυσμένα μεταξύ τους. Για το λόγο αυτό η υλοποίηση της διεπαφής βασίζεται στη χρήση τεχνικών και πρωτοκόλλων IoT δίνοντας έτσι τη δυνατότητα στο χρήστη να χειρίζεται το όχημα του είτε βρίσκεται εντός του οχήματος είτε απομακρυσμένα. Εκτός αυτού βέβαια το σύστημα θα είναι σε θέση να λαμβάνει και να δημοσιεύει πληροφορίες από και προς το νέφος (cloud) που θα συμβάλλουν στην εύρυθμη λειτουργία της κυκλοφορίας. Ο χρήστης θα μπορεί να καθορίζει πέρα από τις απλές λειτουργίες ενός οχήματος όπως αλλαγές λωρίδας και κατεύθυνσης και μια πιο αφηρημένη συμπεριφορά του οχήματος προσδιορίζοντας τον βαθμό στον οποίο επιθυμεί να τηρεί το όχημα του τους νόμους αλλά και γενικότερα το πόσο παθητικά ή επιθετικά επιθυμεί να κινηθεί.

1.3 ΔΙΑΡΘΡΩΣΗ ΤΗΣ ΑΝΑΦΟΡΑΣ

Η διάρθρωση της παρούσας διπλωματικής εργασίας είναι η εξής:

- **Κεφάλαιο 1:** Γίνεται μία εισαγωγή στο γενικότερο πρόβλημα και τα χαρακτηριστικά της αυτόνομης οδήγησης παρουσιάζοντας τα επίπεδα αυτονομίας ενός οχήματος και περιγράφοντας το πρόβλημα που πρόκειται να επιλυθεί.
- **Κεφάλαιο 2:** Γίνεται ανασκόπηση της ερευνητικής περιοχής που αφορά την αυτόνομη οδήγηση τόσο σε εμπορικό όσο και σε ακαδημαϊκό επίπεδο και το υλικό που χρησιμοποιείται για την λειτουργία του συστήματος.
- **Κεφάλαιο 3:** Παρουσιάζονται τα εργαλεία που χρησιμοποιήθηκαν στις υλοποιήσεις και τα πειράματα.
- **Κεφάλαιο 4:** Περιγράφεται η διεπαφή που υλοποιήθηκε στο NodeRED και οι αλγόριθμοι που χρησιμοποιούνται για τη λειτουργία του αυτόνομου οχήματος.
- **Κεφάλαιο 5:** Παρουσιάζεται αναλυτικά η μεθοδολογία των πειραμάτων και τα αποτελέσματα.
- **Κεφάλαιο 6:** Παρουσιάζονται τα τελικά συμπεράσματα, τα προβλήματα που προέκυψαν κατά την εκπόνηση της εργασίας και προτείνονται θέματα για μελλοντική μελέτη, αλλαγές και επεκτάσεις.

⁸https://en.wikipedia.org/wiki/Internet_of_things

2

Επισκόπηση της Ερευνητικής Περιοχής

2.1 STATE-OF-THE-ART ΣΥΣΤΗΜΑΤΑ ΑΥΤΟΝΟΜΗΣ ΟΔΗΓΗΣΗΣ

Στην υποενότητα αυτή γίνεται μια προσπάθεια συλλογής και παρουσίασης μερικών από τα σημαντικότερα projects που έχουν αναπτυχθεί στον τομέα της αυτόνομης οδήγησης. Ο συγκεκριμένος τομέας έχει απασχολήσει τις μεγαλύτερες εταιρίες και βιομηχανίες παγκοσμίως και έχουν επενδυθεί μεγάλα χρηματικά ποσά για την έρευνα και ανάπτυξη τέτοιων συστημάτων. Εκτός αυτού βέβαια, διακεκριμένα πανεπιστήμια στον κόσμο έχουν επικεντρώσει το ενδιαφέρον τους στην έρευνα της ανάπτυξης αυτόνομων οχημάτων και στην αλληλεπίδραση τους με το εξωτερικό περιβάλλον προσφέροντας σημαντικές λύσεις στα ανοικτά ζητήματα.

2.1.1 Εμπορικά συστήματα αυτόνομης οδήγησης

Στην παράγραφο αυτή περιγράφονται σύντομα μερικές από τις σημαντικότερες εταιρίες που έχουν ασχοληθεί ενεργά με τον τομέα της αυτόνομης οδήγησης τα τελευταία χρόνια. Πρόκειται είτε για αυτοκινητοβιομηχανίες που αναπτύσσουν λογισμικό για οχήματα που παράγουν οι ίδιοι, είτε για εταιρίες που αναπτύσσουν λογισμικό το οποίο εφαρμόζεται σε οχήματα άλλων αυτοκινητοβιομηχανιών.

Οι εταιρίες αυτές έχουν προσπαθήσει να δώσουν λύση είτε σε οχήματα τα οποία προορίζονται για προσωπική χρήση των ανθρώπων είτε σε οχήματα τα οποία προορίζονται για χρήση από τους ανθρώπους σαν υπηρεσία μεταφοράς παρομοιάζοντας τα με ταξί.

Αρχικά παρουσιάζονται τα οχήματα της πρώτης κατηγορίας:

- **Tesla:** Η Tesla⁹ ξεκίνησε τον σχεδιασμό αυτόνομου οχήματος τον Οκτώβριο του 2015 αναπτύσσοντας την έκδοση 7 για το λογισμικό του Autopilot. Τον Ιανουάριο του 2016 αναβάθμισε το λογισμικό της στην έκδοση 7.1 η οποία περιείχε

⁹<https://www.tesla.com/>

2.1. STATE-OF-THE-ART ΣΥΣΤΗΜΑΤΑ ΑΥΤΟΝΟΜΗΣ ΟΔΗΓΗΣΗΣ

επιπλέον χαρακτηριστικά όπως αυτόνομο παρκάρισμα χωρίς την επίβλεψη οδηγού. Τον Νοέμβριο του 2020 το λογισμικό της Tesla είχε καταταγεί στο επίπεδο 2 αυτονομίας από τη SAE¹⁰. Το υλικό (hardware) του συγκεκριμένου συστήματος περιλαμβάνει 8 περιβάλλουσες κάμερες για να προσφέρει 360 μοίρες όψη για πάνω από 250 μέτρα, 12 υπέρυθρους αισθητήρες (ultrasonic sensors) για εντοπισμό των εμποδίων και έναν ενσωματωμένο υπολογιστή (onboard computer) ο οποίος προσφέρει 40 φορές περισσότερη υπολογιστική ισχύ από την προηγούμενη γενιά. Τα παλαιότερα μοντέλα της Tesla διέθεταν και ένα εμπρόσθιο ραντάρ για να έχει τη δυνατότητα να βλέπει μέσω βροιχής, ομίχλης, σκόνης αλλά τα μοντέλα μετά το 2021 θα βασίζουν το σύστημα αντίληψης τους χυρίως στη μηχανική όραση μέσω των καμερών¹¹. Το λογισμικό Autopilot¹² περιλαμβάνει λειτουργίες όπως η αντιστοίχιση της ταχύτητας του αυτοκινήτου με την ταχύτητα των γύρω αυτοκινήτων (Traffic-Aware Cruise Control) και η βοήθεια διατήρησης της λωρίδας κυκλοφορίας (Autosteer). Η Tesla διαθέτει και το λογισμικό Full Self-driving Capability το οποίο προσφέρει πιο εξειδικευμένες λειτουργίες όπως η αυτόνομη στάθμευση, η αναγνώριση σημάτων, η αυτόματη αλλαγή λωρίδας και ο χειρισμός του οχήματος σε στενά περιβάλλοντα.

- **Audi:** Το 2017 η Audi παρείχε το μοντέλο Audi A8 το οποίο είναι το πρώτο μοντέλο παγκοσμίως το οποίο συγκαταλέγεται σε αυτονομία επιπέδου 3 σύμφωνα με τη SAE¹³. Η συγκεκριμένη τεχνολογία η οποία ονομάζεται Traffic Jam Pilot προσφέρει βοήθεια στον οδηγό για τις καταστάσεις μεγάλης κίνησης στους δρόμους για ταχύτητες μέχρι 60 km/h όπου οι δύο λωρίδες χωρίζονται με νησίδα απαλλάσσοντας τον από τη ρουτίνα της πόλης. Ο οδηγός δεν είναι υποχρεωμένος να επιτηρεί συνέχεια το αυτοκίνητο παρά μόνο σε μερικές στιγμές. Όταν το σύστημα φτάνει στο όριο του ειδοποιεί τον οδηγό να πάρει πάλι τον έλεγχο του αυτοκινήτου. Επίσης, προσφέρει την τεχνολογία Audi AI remote parking pilot και την Audi AI garage pilot με την οποία ο οδηγός μπορεί να δώσει εντολή μέσω της εφαρμογής myAudi app για να παρκάρει αυτόνομα το όχημα του χωρίς επιτήρηση ή να παρκάρει εντός του γκαράζ του σπιτιού του. Το υλικό (hardware) που διαθέτει το όχημα της Audi είναι ένας μεγάλου εύρους αισθητήρας Radar που σαρώνει το μπροστινό μέρος του αυτοκινήτου, τέσσερα μεσαίου εύρους αισθητήρες Radar στις γωνίες του αυτοκινήτου, μια μπροστινή κάμερα που εντοπίζει γραμμές κυκλοφορίας, πεζούς και άλλα οχήματα, ultrasonic sensors και τέσσερις κάμερες που παρακολουθούν το περιβάλλον γύρω από το αυτοκίνητο και ένα Laser Scanner το οποίο προσφέρει μεγάλης ακρίβειας δεδομένα για πάνω από 80 μέτρα με γωνία 145 μοιρών και μια υπέρυθρη κάμερα που βοηθάει στη νυχτερινή οδήγηση. Διαθέτει ένα κεντρικό ελεγκτή (assistance controller) ο οποίος συγχωνεύει όλες τις μετρήσεις του περιβάλλοντος και εξάγει μία κατάσταση για την κίνηση που

¹⁰<https://www.sae.org/>

¹¹<https://www.tesla.com/support/transitioning-tesla-vision>

¹²<https://www.tesla.com/autopilot>

¹³<https://www.audi-mediacenter.com/en/on-autopilot-into-the-future-the-audi-vision-of-autonomous-driving-9305/the-new-audi-a8-conditional-automated-at-level-3-9307>

επικρατεί¹⁴. Η Audi σχεδιάζει να παράξει το Audi AI:ME το οποίο θα παρέχει level 4 αυτονομία¹⁵.

Το πρόβλημα της αυτόνομης οδήγησης απασχολεί εδώ και αρκετά χρόνια την επιστημονική κοινότητα ελκύοντας όλο και περισσότερες εταιρείες στην έρευνα και χρηματοδότηση τέτοιων έργων. Αξίζουν να αναφερθούν επιγραμματικά κάποιες επιπλέον από αυτές όπως οι Volvo, Apple, Lyft, Mercedes Benz, Argo AI (Ford, Volkswagen), Cruise (General Motors, Honda) κ.α. Έπειτα παρουσιάζονται κάποιες από τις εταιρίες που προσφέρουν τα οχήματα για μαζικότερη μεταφορά όπως οι υπηρεσίες μεταφοράς (ταξί):

- **Waymo (Google):** Τον Αύγουστο του 2012 η Google ανακοίνωσε πως τα οχήματα της έχουν ολοκληρώσει πάνω από 500,000 χιλιόμετρα αυτόνομης οδήγησης χωρίς να προκαλέσουν ατύχημα. Το Μάιο του 2014 εξήγαγε ένα νέο πρωτότυπο αυτόνομο όχημα χωρίς σύστημα διεύθυνσης, πετάλι γκαζιού και πετάλι φρένου καθιστώντας το εντελώς αυτόνομο. Το project ονομάστηκε Firefly και είχε σκοπό να εκδοθεί σαν πλατφόρμα για πειράματα και εκμάθηση και όχι για μαζική παραγωγή. Το 2015 η Google παρείχε το πρώτο εντελώς αυτόνομο ταξίδι σε δημόσιο δρόμο στο Austin του Texas και ήταν το πρώτο ταξίδι που δε συνοδεύεται από τη βοήθεια οδηγού. Το Δεκέμβριο του 2016 μετονομάστηκε το project σε Waymo και έγινε ξεχωριστό τμήμα της εταιρείας Alphabet. Η Waymo ξεκίνησε να τεστάρει αυτόνομα minivans χωρίς οδηγό ασφαλείας σε δημόσιους δρόμους της Αριζόνα τον Οκτώβριο του 2017. Τον Ιανουάριο του 2018 ανακοινώνει ότι θα προσφέρει στην Αριζόνα υπηρεσίες κλήσης αυτοκινήτου καλούμενη Waymo One. Τον Μάρτιο του 2020 η Alphabet επίσημα ανακοινώνει ότι θα παρέχει το Waymo Via το οποίο είναι αυτόνομο φορτηγό για μετακίνηση αγαθών. Το καλοκαίρι της ίδιας χρονιάς ανακοινώνει πως θα παρέχει το λογισμικό της σε αυτοκίνητα της εταιρείας Volvo. Τα οχήματα της Waymo προσφέρουν αυτονομία επιπέδου 4^{16 17}. Το υλικό (hardware) του Waymo project αποτελείται από LiDAR τοποθετημένο στην κορυφή του αυτοκινήτου το οποίο προσφέρει 360 μοίρες οπτικό πεδίο με πάνω από 300 μέτρα εύρος συμβάλλοντας στον εντοπισμό της απόστασης των εμποδίων, τέσσερα περιμετρικά LiDAR τα οποία τοποθετούνται σε τέσσερις περιμετρικές θέσεις του αυτοκινήτου για να εντοπίζουν τα κοντινά εμπόδια και βοηθούν στο να διασχίζει το όχημα μικρά κενά σε κατάσταση μεγάλης κίνησης στην πόλη, κάμερες οι οποίες παρέχουν 360 μοίρες οπτικό πεδίο για πάνω από 500 μέτρα και βοηθούν στην αναγνώριση περισσότερων λεπτομερειών όπως φανάρια και πεζοί χρησιμοποιώντας τεχνικές Μηχανικής Μάθησης συμβάλλοντας έτσι στην κατανόηση του περιβάλλοντος, Radar το οποίο χρησιμοποιείται ως βοήθημα στα LiDAR και τις κάμερες για τον εντοπισμό αντικειμένων σε καταστάσεις βροχής, ομίχλης και σκόνης¹⁸.

¹⁴[https://www.audi-mediacenter.com/en/the-fourth-generation-of-the-audi-a8-heading-into-a-new-era-9403/audi-ai-and-driver-assistance-systems-9413](https://www.audi-mediacenter.com/en/the-fourth-generation-of-the-audi-a8-heading-into-a-new-era-9403/)

¹⁵<https://www.audi.com/en/experience-audi/models-and-technology/concept-cars/audi-aime.html>

¹⁶https://en.wikipedia.org/wiki/Self-driving_car

¹⁷<https://en.wikipedia.org/wiki/Waymo>

¹⁸<https://blog.waymo.com/2020/03/introducing-5th-generation-waymo-driver.html>

2.1. STATE-OF-THE-ART ΣΥΣΤΗΜΑΤΑ ΑΥΤΟΝΟΜΗΣ ΟΔΗΓΗΣΗΣ

- **Baidu:** Αποτελεί εταιρεία στην Κίνα η οποία εργάζεται στο project Apollo με σκοπό να δημιουργήσει το Apollo Robotaxi¹⁹ το οποίο θα προσφέρει αυτονομία level 4. Έχει ήδη ξεκινήσει κάποιες δοκιμαστικές διαδρομές στην Changsha και σε άλλες πόλεις. Ξεκίνησε την υλοποίηση του έργου της τον Σεπτέμβριο του 2019. Δίνει τη δυνατότητα στον άνθρωπο να καλεί το όχημα Robotaxi μέσω της εφαρμογής Baidu Map app ή του Baidu app το οποίο προσφέρει υπηρεσίες μεταφοράς (ταξί). Το υλικό του αποτελείται από διάφορους αισθητήρες όπως LiDAR, GPS, κάμερες και Radar. Μερικές από τις ιδιότητες που έχει είναι η έξυπνη αλλαγή λωρίδας, διάσχιση στενών δρόμων, διάσχιση διασταύρωσεων, απροστάτευτη αριστερή στροφή κλπ. Έχει δημιουργήσει και μια ανοιχτή πλατφόρμα (open platform) η οποία βοηθάει στην ανάπτυξη προγραμμάτων για αυτόνομα οχήματα²⁰.

Η Uber επίσης είναι μία εταιρία που έστρεψε και πάλι το ενδιαφέρον της στην αυτόνομη οδήγηση σε ό,τι αφορά την παροχή υπηρεσιών όπως επίσης και αρκετές καινούργιες εταιρίες (start-ups) που ιδρύθηκαν για αυτό τον σκοπό όπως οι Zoox, Nuro, EasyMile και Wayve.

2.1.2 Ερευνητικά - Ακαδημαϊκά συστήματα αυτόνομης οδήγησης

Σπουδαία πανεπιστήμια στον κόσμο με δεδομένη αναγνώριση στον τομέα της έρευνας έχουν ξεκινήσει πολύ σημαντικά projects τα οποία προσπαθούν να δημιουργήσουν λύσεις στον τομέα της αυτόνομης οδήγησης. Αναφέρονται παρακάτω μερικά από τα πιο ενδιαφέροντα projects που έχουν ξεκινήσει από πανεπιστήμια στον κόσμο:

- **MIT:** Έχει δημιουργήσει την πλατφόρμα Moral Machine²¹ στην οποία συμμετέχουν εθελοντικά άτομα από όλο τον κόσμο όπου μπορούν να απαντήσουν σε ερωτήματα σχετικά με τις αποφάσεις που πρέπει να πάρει ένα αυτόνομο όχημα όταν βρεθεί σε κατάσταση όπου το ατύχημα είναι αναπόφευκτο. Επίσης, έχει δημιουργήσει την εταιρεία Optimus Ride²² η οποία εργάζεται πάνω στην παραγωγή αυτόνομων οχημάτων. Τα οχήματα τους είναι εξαθέσια και διαθέτουν οχτώ κάμερες και τρεις LiDAR αισθητήρες σαν αυτούς των υπολοίπων εταιρειών για να πλοηγούνται στο περιβάλλον αλλά το κύριο χαρακτηριστικό τους είναι το machine vision με το οποίο αναγνωρίζουν αντικείμενα και μπορούν να κάνουν προβλέψεις για το περιβάλλον. Στόχος της εταιρείας είναι να προσφέρει αυτόνομη οδήγηση επιπέδου 4 προσφέροντας ασφαλή μεταφορά σε πελάτες μέσα σε καθορισμένες περιοχές. Τα οχήματα αυτή τη στιγμή πλοηγούνται με δύο επιβλέποντες έναν οδηγό και έναν υπεύθυνο λογισμικού.
- **Stanford University:** Έχει αναπτύξει το project MARTY²³ με το οποίο δημιούργησε ένα αυτόνομο όχημα μοντέλου DeLorean το οποίο έμαθε να πραγ-

¹⁹<https://apollo.auto/robotaxi/index.html>

²⁰<https://apollo.auto/developer.html>

²¹<https://www.moralmachine.net/>

²²<https://news.mit.edu/2019/optimus-ride-self-driving-0809>

²³<https://news.stanford.edu/2019/12/20/autonomous-delorean-drives-sideways-move-forward/>

ματοποιεί αυτόματο γλίστρημα στο οδόστρωμα. Το project ξεκίνησε το 2015 και ο λόγος δημιουργίας του ήταν να μπορέσει να δημιουργήσει ένα όχημα το οποίο θα ανταποκρίνεται επιτυχώς σε καταστάσεις ολισθηρού οδοστρώματος χρησιμοποιώντας όλη την τριβή μεταξύ του δαπέδου και των λάστιχων για να αποφύγει έτσι να βλάψει το όχημα. Το όχημα υπολογίζει την πιο ομαλή τροχιά που μπορεί να ακολουθήσει. Στην ίδια λογική δοκιμάστηκαν και τα οχήματα Niki, ένα αυτόνομο Volkswagen GTI, και ένα αυτόνομο Audi TTS τα οποία μπορούσαν αυτόνομα να παράγουν ολισθηρή κίνηση αφού χρησιμοποιήθηκε ένα νευρωνικό δίκτυο το οποίο εκπαιδεύτηκε από δεδομένα τα οποία αποκτήθηκαν από διαδρομές που πραγματοποίησαν έμπειροι οδηγοί στα δυο οχήματα. Έτσι είχαν τη δυνατότητα να παράγουν κίνηση υψηλής ταχύτητας και χαμηλής τριβής. Το νευρωνικό δίκτυο εκπαιδεύτηκε με πάνω από 200000 δείγματα (trajectories) που αποκτήθηκαν από ολισθηρά οδοστρώματα όπως χιόνι και πάγος²⁴.

- **Oxford University:** Το 2014 καθηγητές του πανεπιστημίου της Οξφόρδης δημιούργησαν την start-up Oxbotica²⁵ η οποία ασχολείται με την παραγωγή λογισμικού για αυτόνομα οχήματα. Αποτελεί την πρώτη εταιρεία που δοκίμασε τη λειτουργία του οχήματος της στους δρόμους του Ηνωμένου Βασιλείου χρησιμοποιώντας το RobotCar²⁶. Συνεργαζόμενη με άλλες εταιρείες δημιούργησε το project DRIVEN το οποίο έχει ως στόχο να δημιουργήσει αυτονομία level 4. Το λογισμικό της Oxbotica δεν εξαρτάται από GPS και μπορεί να εφαρμοστεί σε οποιοδήποτε περιβάλλον.

2.2 ΑΡΧΙΤΕΚΤΟΝΙΚΕΣ ΣΕ ΣΥΣΤΗΜΑΤΑ ΑΥΤΟΝΟΜΗΣ ΟΔΗΓΗΣΗΣ

Κάθε προσπάθεια που έχει γίνει μέχρι σήμερα για επίλυση του προβλήματος των αυτόνομων οχημάτων είτε έχει βασιστεί σε κάποια ήδη υπάρχουσα αρχιτεκτονική συστήματος είτε έχει προτείνει κάποια καινούργια χρησιμοποιώντας πρωτοποριακές τεχνικές. Αυτό έχει ως αποτέλεσμα να έχουν προταθεί αρκετοί τρόποι αντιμετώπισης του προβλήματος με την κάθε αρχιτεκτονική να χωρίζει το πρόβλημα σε διαφορετικά υποπροβλήματα και να τα συνδέει με διαφορετικούς τρόπους φτάνοντας και στην τελική λύση του προβλήματος. Να σημειωθεί σε αυτό το σημείο ότι ο όρος αρχιτεκτονική αναφέρεται στην οργάνωση του συστήματος ως σύνθεση εξαρτημάτων, στον τρόπο και στα πρωτόκολλα επικοινωνίας με τα οποία συνδέονται τα διαφορετικά εξαρτήματα, τις δομές με τις οποίες θα ελέγχεται το κάθε υποσύστημα και τελικά την φυσική υλοποίηση του. Παρόλο που έχουν προταθεί αρκετές αρχιτεκτονικές κατά καιρούς, αυτές οι αρχιτεκτονικές παρουσιάζουν συχνά πολλές ομοιότητες στον τρόπο προσέγγισης με αποτέλεσμα να μπορεί να γίνει μια γενική κατηγοριοποίηση όπως έχει προταθεί και σε σχετική έρευνα [7].

Στη συγκεκριμένη έρευνα [7] γίνεται μια κατηγοριοποίηση στα συστήματα σε δύο μεγάλες κατηγορίες: τα μεμονωμένα (ego-only systems) και τα συνδεδεμένα (connected systems).

²⁴<https://news.stanford.edu/2019/03/27/autonomous-driving-unknown/>

²⁵<https://www.oxbotica.com/>

²⁶<https://ori.ox.ac.uk/robots/robotcar/>

2.2.1 Μεμονωμένα και Συνδεδεμένα συστήματα

Τα μεμονωμένα συστήματα έχουν ως κύριο χαρακτηριστικό το γεγονός ότι εκτελούν και επιλύουν όλες τις επιμέρους λειτουργίες εντός του αυτόνομου οχήματος με τη βοήθεια ενός κεντρικού υπολογιστή που ανήκει στο όχημα και δε χρησιμοποιούν λειτουργίες τρίτων εξωτερικών συστημάτων. Αυτό σημαίνει ότι έχουν αποκλειστικά όλη την ευθύνη της κίνησης τους και είναι ανεξάρτητα από άλλα συστήματα. Τα περισσότερα σύγχρονα συστήματα [8], [9], [10] χαρακτηρίζονται ως μεμονωμένα λόγω των δυσκολιών που αντιμετωπίζουν τα συνδεδεμένα συστήματα.

Από την άλλη πλευρά όταν γίνεται λόγος για συνδεδεμένα συστήματα τότε πρόκειται για ένα γενικότερο οικοσύστημα του οποίου τα επιμέρους συστήματα είναι σε θέση να επικοινωνούν μεταξύ τους, να ανταλλάσουν πληροφορίες και να εκμεταλλεύονται τα δεδομένα μεταξύ τους. Όπως είναι κατανοητό ένα αυτόνομο όχημα μπορεί να βρίσκεται μέσα σε ένα τέτοιο σύστημα και να εκμεταλλεύεται πληροφορίες από το εξωτερικό περιβάλλον του χωρίς να χρειάζεται να λύνει όλα τα επιμέρους προβλήματα. Η λογική αυτή αναφέρεται συχνά ως V2X [11] η οποία μεταφράζεται ως "vehicle-to-everything" και αφορά την επικοινωνία του οχήματος με πεζούς (V2P), με οχήματα (V2V), με το δίκτυο (V2N), με συσκευές (V2D) και με όποια άλλη οντότητα συμπεριλαμβάνεται σε αυτό το γενικότερο πλαίσιο. Αυτή η τεχνολογία στην ουσία έχει σκοπό να μοιράσει το γενικότερο πρόβλημα ανάμεσα σε διαφορετικές οντότητες. Η τεχνολογία η οποία χρησιμοποιείται για την συγκεκριμένη υλοποίηση είναι τα Vehicular ad hoc Networks (VANETs) τα οποία συνδέουν το κάθε όχημα σαν ξεχωριστό κόμβο του συστήματος και του επιτρέπουν να επικοινωνεί με τους υπόλοιπους κόμβους του συστήματος [12]. Μέχρι σήμερα δεν έχει δοθεί κάποια εύρωστη λύση στο πρόβλημα αυτό αλλά εκτιμάται ότι η υλοποίηση του θα μειώσει σημαντικά πολλά ατυχήματα μεταξύ των οχημάτων.

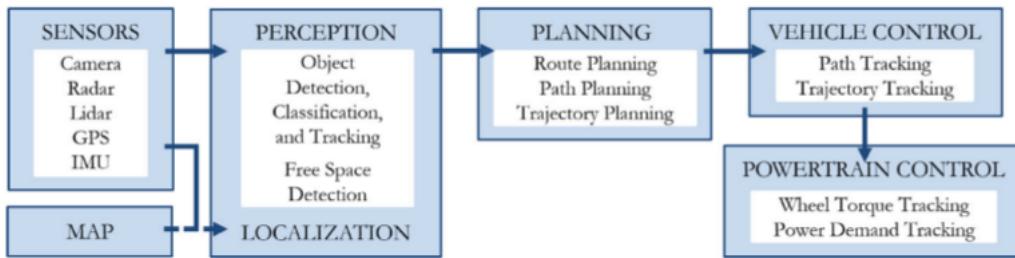
2.2.2 Αρθρωτές και Διατερματικές Μεθοδολογίες

Σε κάθε μία από αυτές τις δύο κατηγορίες, η υλοποίηση του συστήματος δηλαδή η δομή της αρχιτεκτονικής μπορεί να οριστεί με διαφορετικούς τρόπους και επίπεδα (layers). Δύο βασικοί τρόποι υλοποίησης είναι οι αρθρωτές αρχιτεκτονικές (modular) και οι διατερματικές (end-to-end) αρχιτεκτονικές.

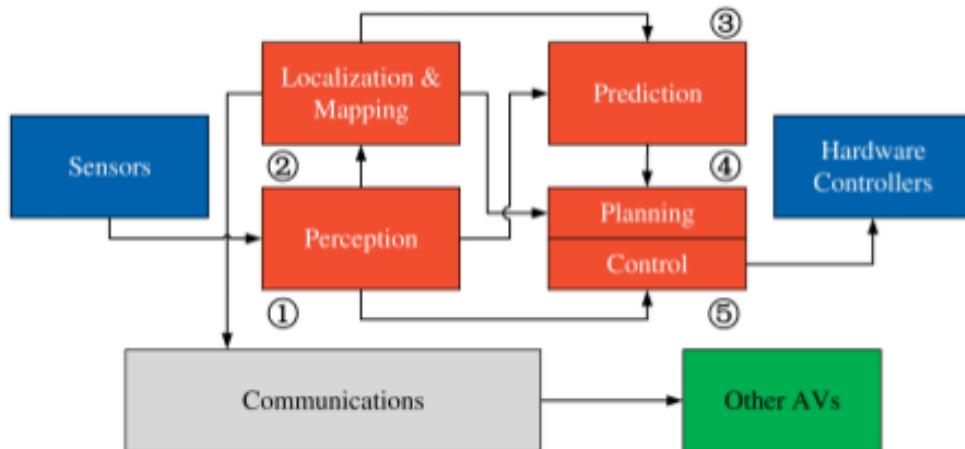
Στην πρώτη μέθοδο η αρχιτεκτονική χωρίζεται σε επιμέρους συστατικά τα οποία ενώνει ξεκινώντας από τις εισόδους/μετρήσεις των αισθητήρων και καταλήγοντας στις εξόδους των ενεργοποιητών (actuators). Στην ουσία η μέθοδος αυτή χωρίζει την αρχιτεκτονική σε συγκεκριμένα υποσυστήματα. Τα υποσυστήματα αυτά γενικότερα έχουν παρόμοιες λειτουργίες στην προτεινόμενη βιβλιογραφία [13], [14], [15], [16], [17], [18], [19] και διαχωρίζονται με παρόμοιο τρόπο. Σαν γενικά υποσυστήματα μπορούν να θεωρηθούν τα εξής υποσυστήματα: αντίληψη (perception), εντοπισμός θέσης (localization), λήψη απόφασης (decision making), σχεδιασμός διαδρομής (path planning), έλεγχος (control), διεπαφή (interface). Φυσικά, κάποιο υποσύστημα μπορεί να περιέχεται σε κάποιο άλλο όπως για παράδειγμα το υποσύστημα εντοπισμού θέσης (localization) στο υποσύστημα αντίληψης (perception) ή το υποσύστημα σχεδιασμού διαδρομής (path planning) και το υποσύστημα λήψης απόφασης (decision making) να αποτελούν ένα υποσύστημα αλλά αυτό εξαρτάται από την κάθε σχεδί-

ΚΕΦΑΛΑΙΟ 2. ΕΠΙΣΚΟΠΗΣΗ ΤΗΣ ΕΡΕΥΝΗΤΙΚΗΣ ΠΕΡΙΟΧΗΣ

αση. Αυτό που πρέπει να τονιστεί είναι ότι στη συγκεκριμένη μέθοδο τα επιμέρους υποσυστήματα θα πρέπει να λειτουργούν σωστά, να παράγουν μία ευανάγνωστη έξοδο και να επικοινωνούν με τον κατάλληλο τρόπο μεταξύ τους ανεξάρτητα από το πως έχουν διαχωριστεί. Κάθε ένα από αυτά τα υποσυστήματα αναλύεται παρακάτω διότι η συγκεκριμένη διπλωματική έχει βασιστεί σε αυτή τη λογική. Παρακάτω φαίνονται (σχήμα 2.1 και σχήμα 2.2) παραδείγματα αρθρωτής αρχιτεκτονικής ενός μεμονωμένου και ενός συνδεδεμένου συστήματος όπου διαχρίνονται τα παραπάνω υποσυστήματα και ο τρόπος που συνδέονται.



Σχήμα 2.1: Αρθρωτή (modular) αρχιτεκτονική σε μεμονωμένο σύστημα (ego-only) και τα επιμέρους υποσυστήματα της [16].



Σχήμα 2.2: Αρθρωτή (modular) αρχιτεκτονική σε συνδεδεμένο (connected) και τα επιμέρους υποσυστήματα της [16].

Η δεύτερη μέθοδος με την οποία μπορεί να υλοποιηθεί ένα σύστημα που ανήκει στις δύο παραπάνω κατηγορίες είναι η από άκρη σε άκρη μέθοδος (end-to-end) η οποία αποτελεί μια πρωτοποριακή μέθοδο των τελευταίων χρόνων και έχει ως σκοπό να λύσει απευθείας το πρόβλημα παίρνοντας ως είσοδο τις μετρήσεις των αισθητήρων και παράγοντας κατευθείαν τις εισόδους των ενεργοποιητών. Ουσιαστικά, η διαφορά με την προηγούμενη μέθοδο είναι ότι αποφεύγει να χωρίσει το πρόβλημα σε επιμέρους υποπροβλήματα και να τα επιλύσει ξεχωριστά αλλά αντιθέτως προσπαθεί να λύσει κατευθείαν το πρόβλημα. Σε γενικές γραμμές αυτή η

2.2. ΑΡΧΙΤΕΚΤΟΝΙΚΕΣ ΣΕ ΣΥΣΤΗΜΑΤΑ ΑΥΤΟΝΟΜΗΣ ΟΔΗΓΗΣΗΣ

μεθοδολογία βασίζεται σε τεχνικές μηχανικής μάθησης και πιο συγκεκριμένα διαχρίνεται σε τρεις μικρότερες ενότητες. Η πρώτη ενότητα περιλαμβάνει την εποπτευόμενη μάθηση (supervised learning) [20] με την οποία μία συγκεκριμένη δομή όπως για παράδειγμα ένα νευρωνικό δίκτυο εκπαιδεύεται κατάλληλα με έναν ικανοποιητικό αριθμό δειγμάτων πριν τη λειτουργία του συστήματος και έπειτα το εκπαιδευμένο σύστημα χρησιμοποιείται για τη λήψη των σωστών αποφάσεων. Αυτό φυσικά απαιτεί τη λήψη κατάλληλων δεδομένων για την εκπαίδευση. Για παράδειγμα, υπάρχουν τεχνικές στις οποίες το σύστημα δέχεται ως είσοδο την εικόνα των καμερών του αυτοκινήτου και παράγει ένα σύνολο από εντολές ελέγχου της διεύθυνσης και της ταχύτητας του οχήματος. Μια ακόμη τεχνική η οποία μπορεί να χαρακτηριστεί ως end-to-end είναι η ενισχυμένη μάθηση (reinforcement learning) [21] στην οποία το όχημα μαθαίνει να κινείται μέσα από την εμπειρία δίνοντας του κατάλληλα βραβεία (rewards) σε περίπτωση σωστής κίνησης και αντίθετα κατάλληλες ποινές (penalties) σε περίπτωση λανθασμένης απόφασης. Η κύρια διαφορά τους βρίσκεται στο γεγονός ότι η εποπτευόμενη μάθηση προσφέρει λύση στην οποία το σύστημα εκπαιδεύεται προηγουμένως και αν έχει εκπαιδευτεί σωστά δεν προβλέπεται να αποτύχει ενώ στη δεύτερη περίπτωση το σύστημα μαθαίνει την επιθυμητή συμπεριφορά μέσα από επαναλαμβανόμενα σφάλματα κατά τη διάρκεια της κίνησης. Και οι δύο τεχνικές είναι πολλά υποσχόμενες και έχουν αρχίσει να αναπτύσσονται σημαντικά χωρίς βέβαια να έχει παρουσιαστεί ακόμη ένα σύστημα πλήρως λειτουργικό που να βασίζεται αποκλειστικά σε τεχνικές μηχανικής μάθησης. Ωστόσο, το κύριο μειονέκτημα των end-to-end τεχνικών είναι ότι δεν είναι σαφώς προγραμματισμένες τεχνικές οπότε ενδεχομένως να μην προσφέρουν την κατάλληλη ασφάλεια στο σύστημα.

Άλλες αρχιτεκτονικές

Εκτός από την ομαδοποίηση που έγινε παραπάνω ανάμεσα στις αρχιτεκτονικές που έχουν κοινά χαρακτηριστικά έχουν προταθεί κατά καιρούς πολλές ακόμη αρχιτεκτονικές που χρησιμοποιούν διαφορετικά υποσυστήματα/επίπεδα (layers) για να επιλύσουν το γενικότερο πρόβλημα. Κάποια ενδιαφέροντα επίπεδα (layers) που αξίζουν να αναφερθούν και χρησιμοποιούνται από διάφορες αρχιτεκτονικές αναφέρονται στη συνέχεια.

Ένα από αυτά είναι το επίπεδο δικτύου (Network layer) το οποίο έχει ως κύριο σκοπό τη σύνδεση και την ανταλλαγή πληροφοριών η οποία περιλαμβάνει πρόσβαση δικτύου, μετάδοση και έλεγχο του δικτύου [22]. Το επίπεδο αυτό είναι υπεύθυνο για τις επικοινωνίες που είναι βασισμένες στο δίκτυο ανάμεσα στις διαφορετικές οντότητες της κυκλοφορίας, έχοντας ως στόχο την απόκτηση δεδομένων σχετικών με την κυκλοφορία. Σε αυτό το επίπεδο εντάσσονται η επικοινωνία του αυτοκινήτου με το εσωτερικό του (intra vehicular) όπως σύστημα προειδοποίησης ατυχήματος, σύστημα αλλαγής λωρίδας, αυτόνομο παρκάρισμα κλπ. αλλά και η επικοινωνία με τον έξω κόσμο όπως επικοινωνία με άλλα οχήματα (V2V) ή πεζούς (V2P) [23].

Έπειτα υπάρχει, το επίπεδο ασφάλειας (Security layer) το οποίο συνήθως συνδέεται και επικοινωνεί με όλα τα υπόλοιπα επίπεδα έτσι ώστε να εξασφαλίζει την ασφάλεια ολόκληρου του συστήματος και να επιτελεί λειτουργίες όπως είναι η διαθεσιμότητα του συστήματος, ο έλεγχος πρόσβασης, η αυθεντικοποίηση των

δεδομένων και η αντιμετώπιση κάθε είδους επίθεση [24].

Το edge επίπεδο (layer) το οποίο λόγω της μεγάλης αύξησης των δεδομένων κυρίως από τα συνδεδεμένα συστήματα (connected systems) προσφέρει επεξεργασία των δεδομένων σε πραγματικό χρόνο. Ο όγκος των πληροφοριών είναι πολύ μεγάλος και για αυτό παράγονται σημαντικές καθυστερήσεις όταν τα δεδομένα επεξεργάζονται από έναν κεντρικό υπολογιστή. Για αυτό το λόγο το συγκεκριμένο στρώμα χρησιμοποιεί μία φυσική συσκευή η οποία βρίσκεται κοντά στην πηγή δεδομένων και βοηθάει στην ανάλυση τους. Η λειτουργία της είναι να γνωστοποιεί τα αποτελέσματα από την ανάλυση των δεδομένων που συλλέχθηκαν από τα γεγονότα της κίνησης και έπειτα να σχηματίζει ένα μοντέλο λήψης αποφάσεων αναλαμβάνοντας κάποια από τα καθήκοντα του υπολογιστικού νέφους (cloud computing) βελτιώνοντας έτσι τη συνολική απόδοση του κέντρου δεδομένων (cloud data center) [25].

Το εικονικό επίπεδο (Virtual Layer) υπάρχει σε αρκετές αρχιτεκτονικές και χρησιμεύει για περιπτώσεις όπου το όχημα κινείται στο δρόμο και εισέρχεται σε περιοχές με χαμηλή ή και καθόλου σύνδεση στο διαδίκτυο και θα πρέπει να δημιουργηθεί μία εικονική οντότητα του αυτοκινήτου στο νέφος στην οποία θα αποθηκεύεται η τελευταία κατάσταση του οχήματος και των αισθητήρων. Όταν το αυτοκίνητο αποκτήσει ξανά σύνδεση στο διαδίκτυο τότε ανακτά την τελευταία του κατάσταση από αυτό το επίπεδο [26].

Εκτός από τα προαναφερθείσα επίπεδα αρχιτεκτονικών τα οποία χρησιμοποιούνται συχνά υπάρχουν ακόμη περισσότερα τα οποία αποτελούν ξεχωριστές κατηγορίες και δεν ομαδοποιούνται για αυτό και δεν έγινε αναφορά σε αυτά. Σε κάθε περίπτωση είναι στην κρίση του σχεδιαστή και εξαρτάται από τις απαιτήσεις του συστήματος ποια αρχιτεκτονική είναι η καταλληλότερη σε κάθε πρόβλημα αυτόνομης οδήγησης και η επιλογές ποικίλουν.

2.3 ΥΛΙΚΟ (HARDWARE) ΣΕ ΣΥΣΤΗΜΑΤΑ ΑΥΤΟΝΟΜΗΣ ΟΔΗΓΗΣΗΣ

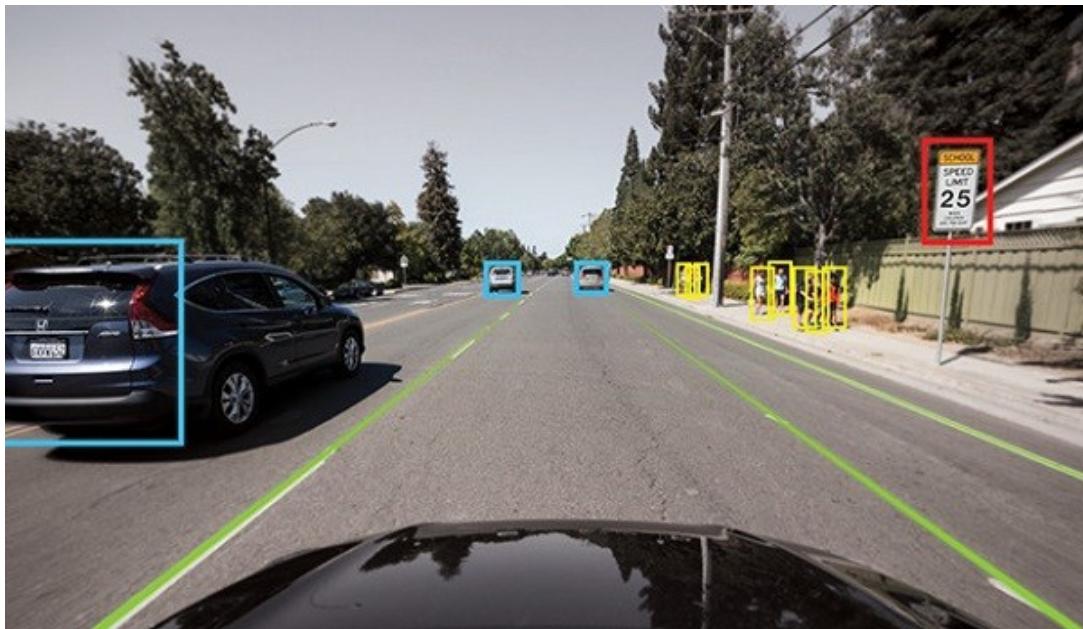
Στην ενότητα αυτή περιγράφονται τα κομμάτια του υλικού (hardware) ή αλλιώς αισθητήρες που χρησιμοποιούνται σήμερα σε ένα αυτόνομο όχημα είτε πραγματικό είτε σε περιβάλλον προσομοίωσης (simulation). Ως αισθητήρας²⁷ ορίζεται μία συσκευή που ανιχνεύει ένα φυσικό μέγεθος και παράγει μία έξοδο η οποία μπορεί να μετρηθεί και να αξιοποιηθεί. Οι αισθητήρες αποτελούν το μέσο για να αντιληφθεί το όχημα τον περιβάλλοντα χώρο για να μπορέσει στη συνέχεια να λάβει σωστές αποφάσεις. Οι αισθητήρες τόσο στον τομέα της αυτόνομης οδήγησης όσο και στη ρομποτική μπορούν να χωριστούν σε συγκεκριμένες κατηγορίες οι οποίες παρατίθενται παρακάτω:

- **Παθητικοί:** Θεωρούνται οι αισθητήρες οι οποίοι δεν εκπέμπουν κάποιο σήμα προκειμένου να μετρήσουν το μέγεθος που τους ενδιαφέρει αλλά αντίθετα απλώς λαμβάνουν σαν είσοδο τη μετρούμενη ποσότητα. Για τον λόγο αυτό οι μετρήσεις τους δεν επηρεάζονται από άλλες εκπομπές συσκευών οπότε και μειώνουν την πιθανότητα σφάλματος σε αυτές.

²⁷<https://en.wikipedia.org/wiki/Sensor>

2.3. ΥΛΙΚΟ (HARDWARE) ΣΕ ΣΥΣΤΗΜΑΤΑ ΑΥΤΟΝΟΜΗΣ ΟΔΗΓΗΣΗΣ

- **Κάμερα:** Οι κάμερες έχουν τη δυνατότητα να λαμβάνουν ως είσοδο εικόνα από το περιβάλλον, η οποία σε πολλές περιπτώσεις είναι και έγχρωμη (RGB κάμερες) δίνοντας στο όχημα τη δυνατότητα να πραγματοποιεί εργασίες όπως αναγνώριση του χρώματος του φαναριού, αναγνώριση σημάτων και γενικότερα να έχει πλήρη γνώση του περιβάλλοντος του σε περίπτωση που πραγματοποιείται σωστά η μηχανική όραση. Ωστόσο, οι κάμερες που λειτουργούν μόνες τους χωρίς τον συνδυασμό κάποιας άλλης όπως οι monocular, έχουν το αρνητικό ότι δεν μπορούν να αναγνωρίσουν το βάθος της εικόνας και κατά συνέπεια την απόσταση από τα αντικείμενα που αναγνωρίζει. Ωστόσο, η χρήση δύο όμοιων καμερών σε γνωστή απόσταση με γνωστό μετασχηματισμό μεταξύ τους (stereo κάμερες) έρχεται να δώσει λύση σε αυτό το πρόβλημα χρησιμοποιώντας τεχνικές που επιτρέπουν τον υπολογισμό του βάθους των αντικειμένων της εικόνας. Επίσης, ένα ακόμη μειονέκτημα των καμερών εντοπίζεται στο γεγονός ότι επηρεάζονται οι μετρήσεις τους από τις καιρικές συνθήκες και από καταστάσεις έντονου φωτισμού δυσκολεύοντας την αναγνώριση των εμποδίων μέσα στην εικόνα. Από εκεί και πέρα βέβαια οι κάμερες αποτελούν μια σχετικά φθηνή λύση προσφέροντας παράλληλα μια πολύ μεγάλη ποσότητα πληροφορίας του περιβάλλοντος για αυτό και είναι από τους αισθητήρες που χρησιμοποιούνται συχνά σε συστήματα αυτόνομης οδήγησης. Εκτός βέβαια από τις απλές monocular κάμερες υπάρχουν και άλλα είδη καμερών που προσφέρουν μεγαλύτερες δυνατότητες. Οι Omnidirectional κάμερες, για παράδειγμα, προσφέρουν θέαση 360 μοιρών βοηθώντας στη λύση προβλημάτων όπως ο εντοπισμός της θέσης, η πλοήγηση στο χώρο και η χαρτογράφηση της περιοχής. Επίσης, υπάρχουν οι Event κάμερες οι οποίες εντοπίζουν γεγονότα μέσα στην εικόνα συγκρίνοντας τη φωτεινότητα των pixels διαδοχικών εικόνων και συμβάλλοντας έτσι στον εντοπισμό δυναμικών εμποδίων αλλά και οι κάμερες βάθους οι οποίες προσφέρουν μια εικόνα σε ασπρόμαυρη κλίμακα (greyscale) υποδεικνύοντας το βάθος του κάθε αντικειμένου στην εικόνα [25]. Στο [σχήμα 2.3](#) απεικονίζεται η εικόνα που λαμβάνει ένα όχημα σαν είσοδο από την κάμερα του και τα αντικείμενα τα οποία αναγνωρίζει μέσω της διαδικασίας της μηχανικής όρασης.
- **Ενεργητικοί:** Θεωρούνται οι αισθητήρες οι οποίοι για να λειτουργήσουν εκπέμπουν κάποιο σήμα και μετρούν την ποσότητα η οποία αντανακλάται από το περιβάλλον. Αυτό σημαίνει ότι μπορούν να λειτουργήσουν και κάτω από δύσκολες καιρικές συνθήκες αλλά και συνθήκες έντονου φωτισμού. Οι μετρήσεις ωστόσο μπορεί να επηρεάζονται από άλλες συσκευές που λειτουργούν με τον ίδιο τρόπο δημιουργώντας θόρυβο στις μετρήσεις τους. Οι ενεργητικοί αισθητήρες που θα αναφερθούν παρακάτω μπορούν να καλύψουν αρκετά από τα προβλήματα των καμερών που αναφέρθηκαν προηγουμένως λειτουργώντας παράλληλα με αυτές.
 - **Radar (Radio Detection and Ranging):** Το Radar λειτουργεί εκπέμποντας ραδιοκύματα τα οποία επιστρέφουν σε ένα δέκτη μετρώντας έτσι το χρόνο επιστροφής και συνεπώς και την απόσταση του αντικειμένου.



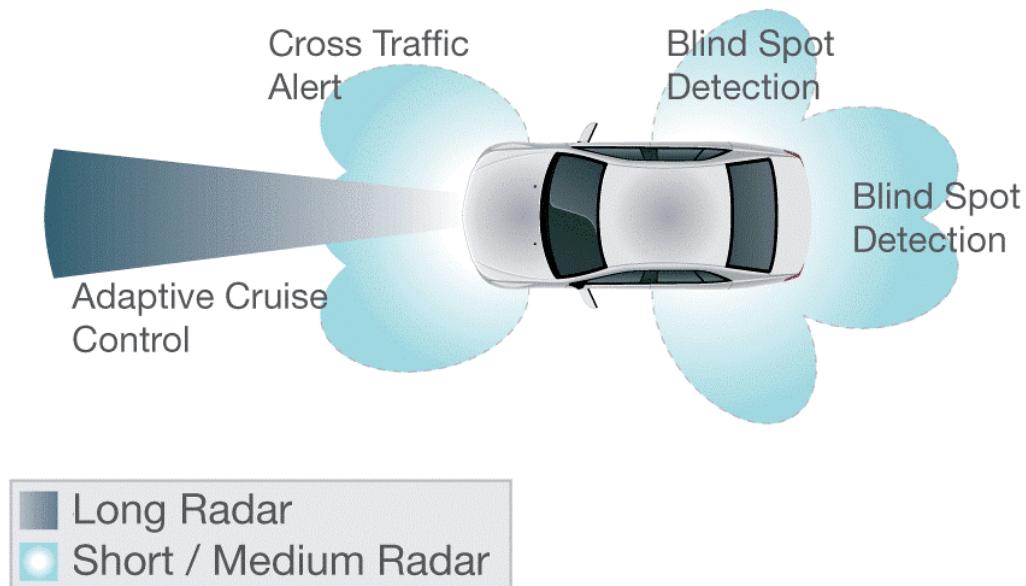
Σχήμα 2.3: Η εικόνα που λαμβάνει ένα όχημα σαν είσοδο από την κάμερα του και τα αντικείμενα τα οποία αναγνωρίζει μέσω της μηχανικής όρασης.

Πηγή: [https://www.rs-online.com/designspark/
lidar-radar-digital-cameras-the-eyes-of-autonomous-vehicles](https://www.rs-online.com/designspark/lidar-radar-digital-cameras-the-eyes-of-autonomous-vehicles)

Μπορούν να λύσουν το πρόβλημα των καμερών όταν οι συνθήκες είναι δύσκολες προσφέροντας πληροφορία για την απόσταση των αντικειμένων στον περιβάλλοντα χώρο. Ωστόσο, δεν μπορούν να αναγνωρίσουν το είδος του αντικειμένου που εντόπισαν. Έχουν χαμηλό βάρος και κόστος προσφέροντας για αυτό το λόγο ευελιξία. Ακόμη, Η απόσταση που μετρούν είναι μεγάλη δίνοντας τους τη δυνατότητα να έχουν αντίληψη του περιβάλλοντος για αρκετά μέτρα μακριά αλλά από την άλλη πλευρά η ακρίβεια τους χαρακτηρίζεται ως μέτρια. Τα αποτελέσματα τους είναι καλύτερα σε κινούμενα αντικείμενα παρά σε στατικά εμπόδια [25]. Σε αντίθεση με άλλους αισθητήρες απόστασης που μετρούν τη διαφορά της απόστασης ανάμεσα σε δύο διαδοχικές μετρήσεις, το Radar χρησιμοποιεί το φαινόμενο Ντόπλερ μετρώντας τη διαφορά στη συχνότητα του κύματος αν αυτό έρχεται προς το μέρος του αυτοκινήτου ή φεύγει από αυτό.

- **LiDAR (Light Detection and Ranging):** Το LiDAR σε γενικές γραμμές έχει παρόμοιο τρόπο λειτουργίας με το Radar αλλά λειτουργεί εκπέμποντας υπέρυθρη φωτεινή ακτινοβολία σε σύγκριση με τα ραδιοκύματα που εκπέμπει το Radar. Έχει μεγάλη ακρίβεια αλλά χαρακτηρίζεται από μέτριο εύρος μέτριο εύρος το οποίο συνήθως είναι κάτω από τα 200 μέτρα. Οι συσκευές συνήθως είναι περιστρεφόμενες δίνοντας τη δυνατότητα να αποτυπωθεί ο γύρω χώρος σαν ένα σύνολο σημείων νέφους (Point Cloud Map) δίνοντας στο όχημα τη δυνατότητα να γνωρίζει το περιβάλλον μέσα στο οποίο κινείται. Η τεχνική με την οποία μετράει τις αποστάσεις από

2.3. ΥΛΙΚΟ (HARDWARE) ΣΕ ΣΥΣΤΗΜΑΤΑ ΑΥΤΟΝΟΜΗΣ ΟΔΗΓΗΣΗΣ

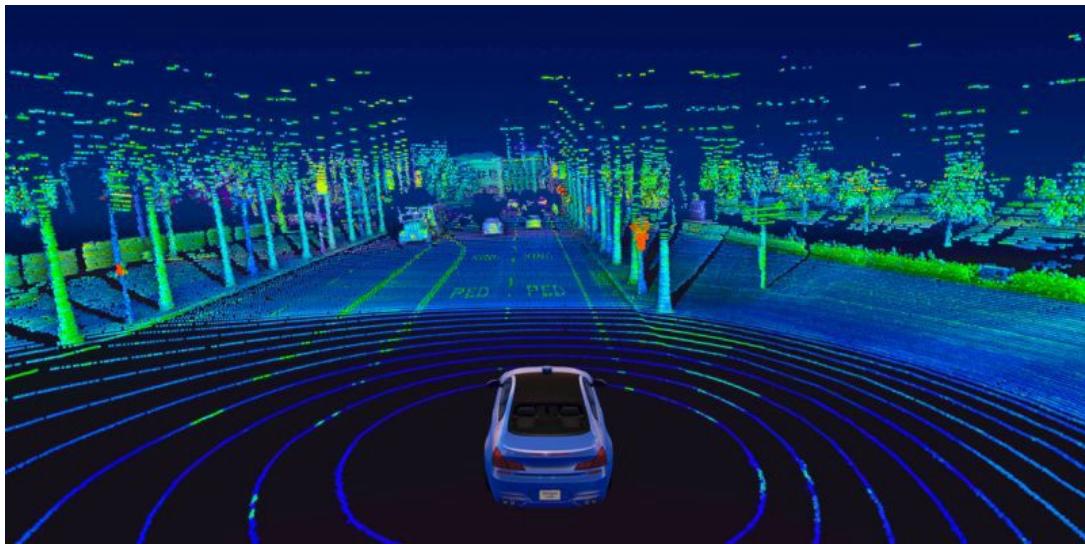


Σχήμα 2.4: Ένα όχημα εξοπλισμένο με Radar μικρού και μεσαίου εύρους σε γαλάζιο χρώμα αλλά και μεγαλύτερου εύρους σε μπλε χρώμα.

Πηγή: <https://riversonicsolutions.com/>

τα εμπόδια είναι υπολογίζοντας το χρόνο που κάνει το κύμα να επιστρέψει στην πηγή του και κατ’ επέκταση μπορεί και να υπολογίσει και την απόσταση από αυτό. Επίσης, η λειτουργία του LiDAR επηρεάζεται από τις δύσκολες καιρικές συνθήκες όπως το χιόνι και η ομίχλη αδυνατώντας έτσι να αντιμετωπίσουν το κύριο πρόβλημα των καμερών. Το κύριο αρνητικό τους είναι το μεγάλο κόστος τους αλλά και το μέγεθός τους με τα περισσότερα μοντέλα να είναι αρκετά μεγάλα σε μέγεθος. Συνήθως προτιμώνται τα μικρότερα για λόγους αεροδυναμικής και αλλά και οικονομίας χώρου επάνω στο όχημα [25].

- **Υπερηχητικοί (Ultrasonic) αισθητήρες:** Οι υπερηχητικοί αισθητήρες εκπέμπουν υπερηχητικά κύματα και μετρούν την απόσταση από τον χρόνο που κάνει το κύμα να επιστρέψει στην πηγή. Αποτελούν μια οικονομική λύση τόσο σε κόστος όσο και στον χώρο που καταλαμβάνουν επάνω στο όχημα καθιστώντας τους πλέον ως μια αρκετά βιολική λύση. Μπορούν και λειτουργούν κάτω από δύσκολες καιρικές συνθήκες όπως ομίχλη, βροχή, χιόνι αλλά και κατά τις βραδινές ώρες. Είναι συνήθως κατασκευασμένοι ώστε να έχουν μεγάλο οριζόντιο και μικρό κατακόρυφο εύρος για να αποφεύγουν τις αντανακλάσεις του εδάφους. Από την άλλη πλευρά βέβαια διαθέτουν μικρό εύρος αλλά και περιορισμένη ακρίβεια στις μετρήσεις τους. Για αυτό το λόγο χρησιμοποιούνται ως βοηθητικές συσκευές στο συνολικό πρόβλημα αντίληψης του οχήματος μαζί με άλλες συσκευές κα-



Σχήμα 2.5: Ένα όχημα εξοπλισμένο με LiDAR ο χάρτης σημείων νέφους που δημιουργεί.

Πηγή: <https://medium.com/geekculture/how-lidar-fits-into-the-future-of-autonomous-driving-29fc296052bc>

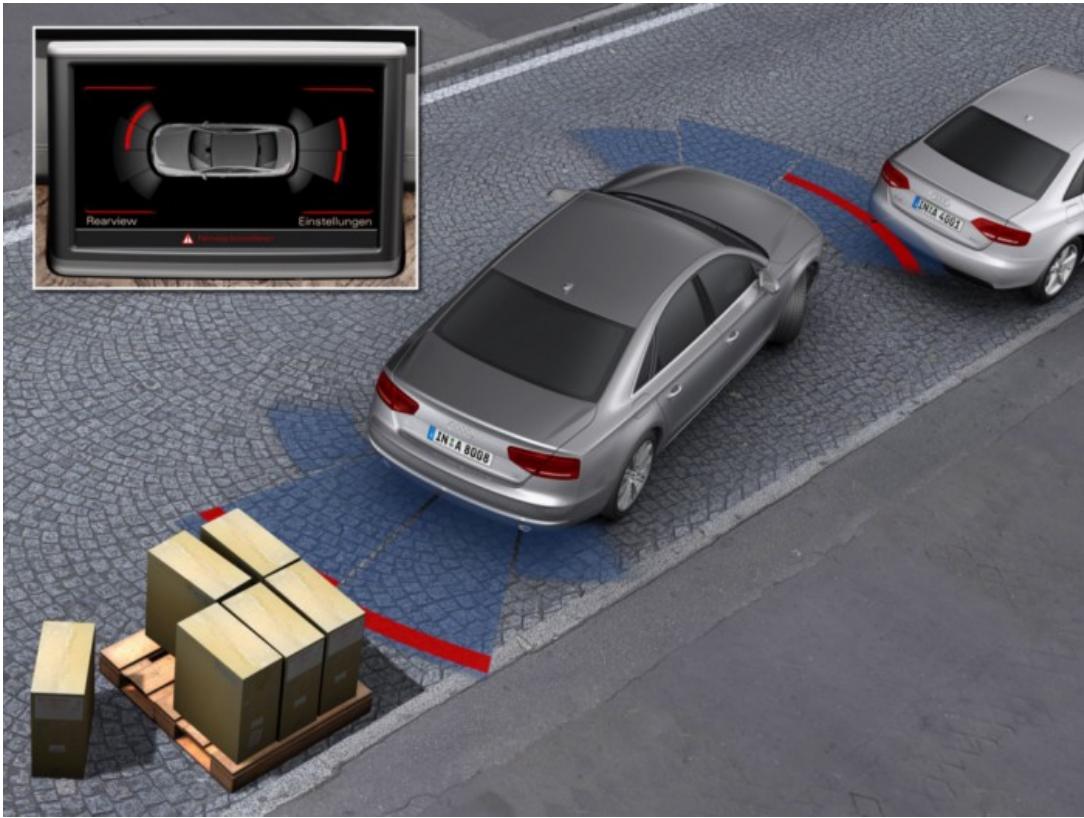
λύτερης ακρίβειας. Χρησιμοποιούνται κυρίως σε στενά περιβάλλοντα για τη βελτίωση της πλοήγησης αλλά και σε περιπτώσεις όπως η αυτόνομη στάθμευση για να εντοπίζει τα κοντινά εμπόδια [25]. Στο **σχήμα 2.6** φαίνεται ένα όχημα το οποίο επιχειρεί να σταθμεύσει πλάγια με τη βοήθεια οπίσθιου και εμπρόσθιου ultrasonic αισθητήρα.

Οι δύο κατηγορίες αισθητήρων που αναφέρθηκαν παραπάνω αφορούν κυρίως τον τρόπο με τον οποίο λαμβάνουν την απαιτούμενη πληροφορία διαχωρίζοντας τους σε αυτούς που παίρνουν ενεργά τις μετρήσεις τους εκπέμποντας και μετρώντας κάποιο σήμα και σε αυτούς που απλώς λαμβάνουν τη μετρούμενη ποσότητα από το περιβάλλον. Εκτός από αυτές τις δύο κατηγορίες οι αισθητήρες μπορούν να καταταγούν και ανάλογα το περιβάλλον από το οποίο λαμβάνουν την πληροφορία. Οι δύο αυτές κατηγορίες περιγράφονται συνοπτικά στη συνέχεια:

- **Proprioceptive sensors:** Αυτή η κατηγορία αισθητήρων περιλαμβάνει συσκευές οι οποίες υπολογίζουν στοιχεία από την κατάσταση του οχήματος όπως η ταχύτητα, η επιτάχυνση και η γωνία κίνησης του οχήματος δηλαδή μεγέθη απαραίτητα για την ασφαλή πλοήγηση του αυτοκινήτου. Τέτοιοι αισθητήρες είναι οι wheel encoders οι οποίοι μετρώντας τον αριθμό των στροφών του κινητήρα μπορούν και μετρούν την οδομετρία, οι IMU οι οποίοι διαθέτουν ένα γυροσκόπιο το οποίο μετρά τη γωνιακή ταχύτητα σε τρεις άξονες και ένα επιταχυνσιόμετρο το οποίο μετρά την επιτάχυνση σε τρεις ορθοκανονικούς άξονες και τα ταχύμετρα τα οποία χρησιμοποιούνται για να μετρούν την ταχύτητα [25]. Αυτές οι μετρήσεις γίνονται προσβάσιμες μέσω του CAN bus²⁸ που διαθέτουν τα σύγχρονα οχήματα.

²⁸https://en.wikipedia.org/wiki/CAN_bus

2.3. ΥΛΙΚΟ (HARDWARE) ΣΕ ΣΥΣΤΗΜΑΤΑ ΑΥΤΟΝΟΜΗΣ ΟΔΗΓΗΣΗΣ



Σχήμα 2.6: Ένα όχημα εξοπλισμένο με υπερυχητικό (ultrasonic) αισθητήρα το οποίο επιχειρεί να πραγματοποιήσει αυτόνομη στάθμευση.

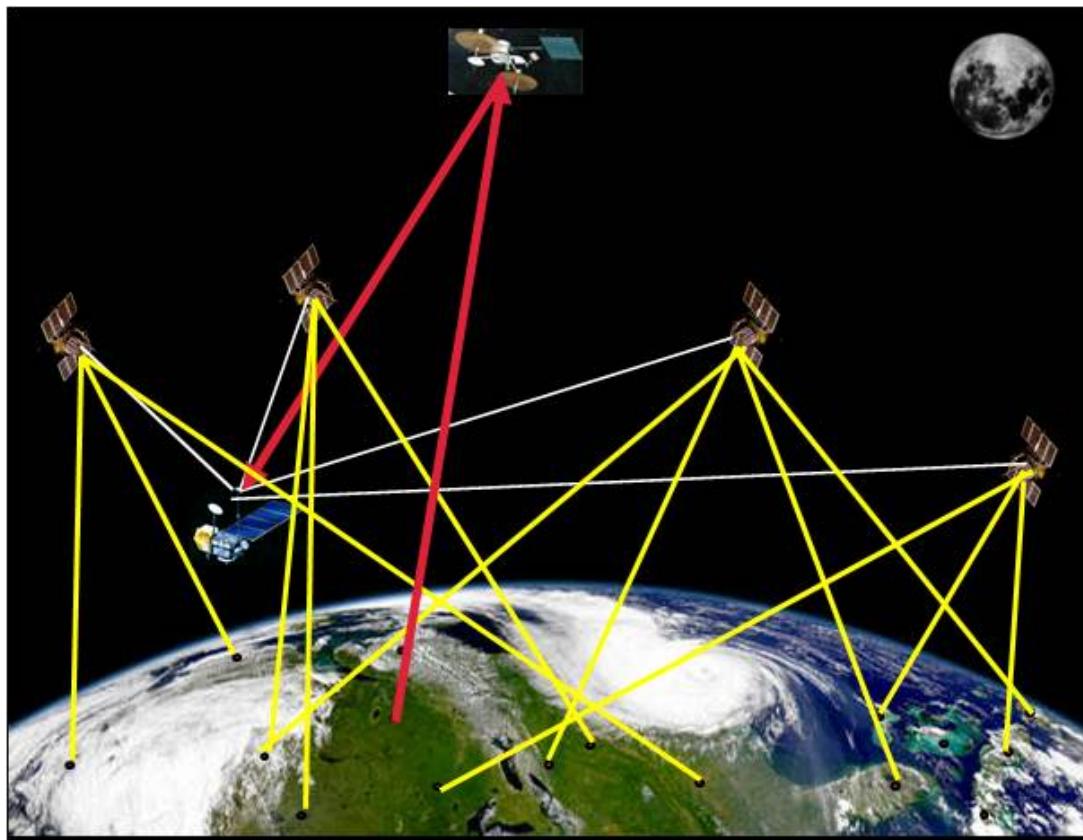
Πηγή:

<https://medium.com/@BabakShah/ultrasonic-sensors-in-self-driving-cars-d28b63be676f>

- **Exteroceptive sensors:** Οι αισθητήρες αυτής της κατηγορίας λαμβάνουν πληροφορίες από το εξωτερικό περιβάλλον του οχήματος. Οι αισθητήρες που αναφέρθηκαν παραπάνω όπως το LiDAR, το Radar, ο Ultrasonic και η κάμερα ανήκουν σε αυτή την κατηγορία. Εκτός αυτών βέβαια, τέτοιος αισθητήρας είναι το GPS (Global Positioning System) το οποίο χρησιμοποιεί δορυφόρους για να προσδιορίσει τη θέση της συσκευής. Η απόσταση από τους δορυφόρους υπολογίζεται χρησιμοποιώντας την ταχύτητα του φωτός και το χρόνο που χρειάζεται για να φτάσει στον δορυφόρο. Για να προσδιοριστεί η τρισδιάστατη θέση του οχήματος χρειάζονται μετρήσεις από τέσσερις δορυφόρους ενώ για τον υπολογισμό της δισδιάστατης θέσης απαιτούνται τρεις δορυφόροι [25]. Στο σχήμα 2.7 φαίνεται η λογική με την οποία λειτουργεί ένας αισθητήρας GPS.

Θέση αισθητήρων

Εκτός από το είδος των αισθητήρων που θα χρησιμοποιήσει ένα αυτόνομο όχημα για να μπορεί να αντιληφθεί τον εξωτερικό χώρο πολύ σημαντικό ρόλο παίζει και η θέση στην οποία θα τοποθετηθούν. Οι κάμερες συνήθως τοποθετούνται στην οροφή του οχήματος είτε κοιτώντας προς τα εμπρός είτε προς τα πίσω για να έχουν πλήρη



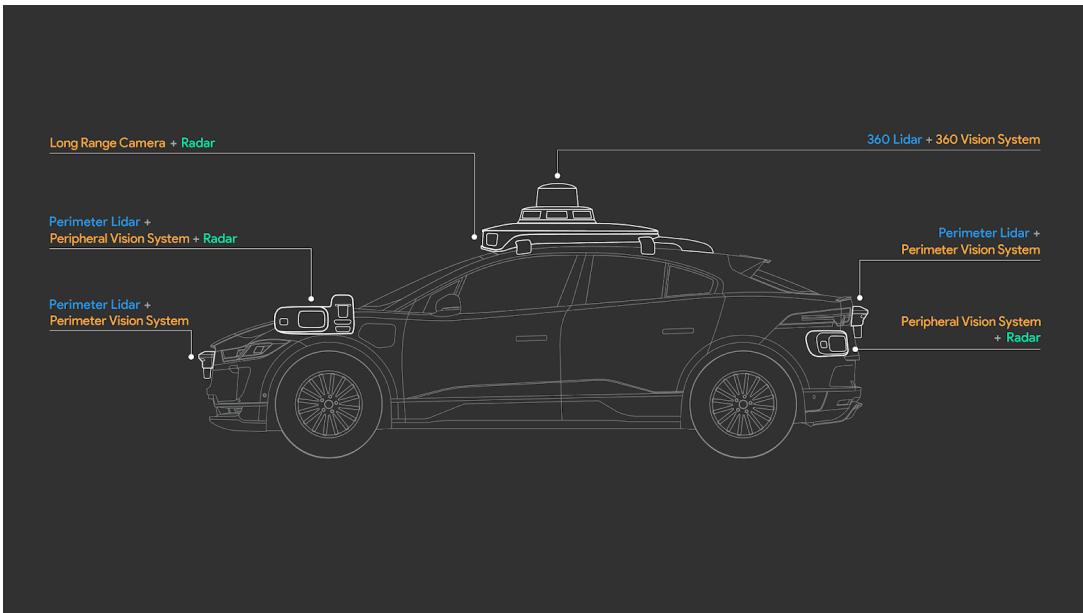
Σχήμα 2.7: Ο τρόπος με τον οποίο ο αισθητήρας GPS λαμβάνει και συνδυάζει τις μετρήσεις από τους δορυφόρους.

Πηγή: https://www.nasa.gov/directorates/heo/scan/communications/policy/policy_gnss.html

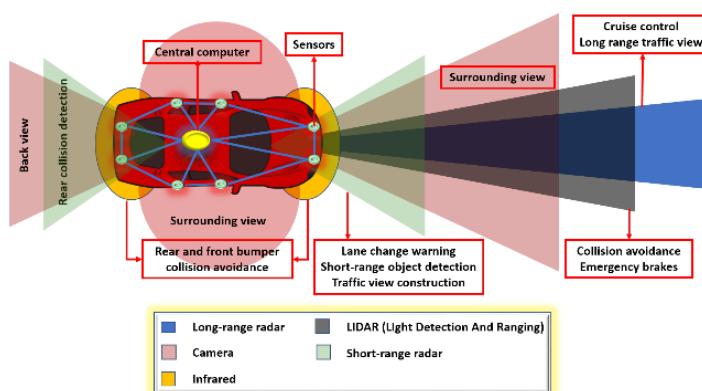
εικόνα του γύρω χώρου και να μπορούν να αναγνωρίζουν οχήματα, πεζούς, σήματα, φωτεινούς σηματοδότες, γραμμές κυκλοφορίας κλπ. Οι αισθητήρες LiDAR επίσης τοποθετούνται συνήθως στην οροφή του οχήματος ειδικά όταν πρόκειται για περιστρεφόμενους αισθητήρες προκειμένου να μπορούν να έχουν θέαση 360 μοιρών και να μπορούν να χαρτογραφούν σωστά την περιοχή. Για τον εντοπισμό κοντινών εμποδίων και τυφλών σημείων χρησιμοποιούνται όπως αναφέρθηκε οι αισθητήρες Radar και οι υπεροχητικοί (ultrasonic) για αυτό και τοποθετούνται συνήθως σε πλαϊνά ή χαμηλά σημεία γύρω από το αυτοκίνητο. Μια τυπική διάταξη των αισθητήρων πάνω σε ένα αυτόνομο όχημα της Waymo²⁹ φαίνεται στο σχήμα 2.8. Στο σχήμα 2.9 φαίνεται ένα όχημα σχεδιασμένο με τις πιθανές θέσεις των αισθητήρων, τη θέση του κεντρικού υπολογιστή και το εύρος λειτουργίας του κάθε αισθητήρα.

²⁹<https://blog.waymo.com/2020/03/introducing-5th-generation-waymo-driver.html>

2.3. ΥΛΙΚΟ (HARDWARE) ΣΕ ΣΥΣΤΗΜΑΤΑ ΑΥΤΟΝΟΜΗΣ ΟΔΗΓΗΣΗΣ



Σχήμα 2.8: Τυπική διάταξη των αισθητήρων σε ένα όχημα της Waymo.
Πηγή: <https://blog.waymo.com/2020/03/introducing-5th-generation-waymo-driver.html>



Σχήμα 2.9: Απεικονίζεται ένα όχημα με τις ακριβείς θέσεις που τοποθετούνται οι αισθητήρες και το εύρος λειτουργίας του κάθε αισθητήρα [2].

3

Εργαλεία Software

Στο κεφάλαιο αυτό παρουσιάζονται τα βασικά εργαλεία που χρησιμοποιήθηκαν κατά τη διάρκεια της συγκεκριμένης διπλωματικής τόσο στις αρχικές υλοποιήσεις όσο και στα πειράματα που εκτελέστηκαν στο τέλος. Συγκεκριμένα, παρουσιάζεται ο προσομοιωτής CARLA που χρησιμοποιήθηκε σαν περιβάλλον του οχήματος, το εργαλείο Node Red που χρησιμοποιήθηκε για την κατασκευή της διεπαφής, τα πρωτόκολλα για την επικοινωνία του κάθε υποσυστήματος και η γλώσσα προγραμματισμού Python που χρησιμοποιήθηκε για να προγραμματιστεί η λειτουργία του οχήματος μαζί με τις διάφορες βιβλιοθήκες της.

3.1 ΠΡΟΣΟΜΟΙΩΤΗΣ CARLA

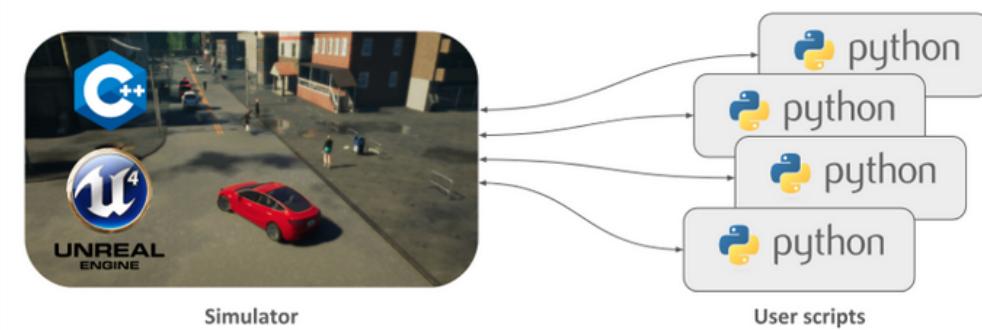
Στην ενότητα αυτή παρουσιάζεται ο προσομοιωτής CARLA [27] που χρησιμοποιήθηκε για να πλοηγηθεί το αυτόνομο όχημα σε ένα ρεαλιστικό περιβάλλον τόσο ως προς τα στατικά όσο και ως προς τα δυναμικά εμπόδια που υπάρχουν σε ένα πραγματικό χώρο κίνησης. Ο CARLA αποτελεί ένα περιβάλλον προσομοίωσης το οποίο αναπτύχθηκε με σκοπό να συνεισφέρει στην έρευνα και την ανάπτυξη συστημάτων αυτόνομης οδήγησης προσφέροντας στους προγραμματιστές ένα εύκολο στη χρήση και ρεαλιστικό περιβάλλον προκειμένου να μετρήσουν την απόδοση των προγραμμάτων τους. Αποτελεί ένα εργαλείο ανοικτού κώδικα το οποίο βασίζεται στη μηχανή Unreal Engine (UE4)³⁰ πάνω στην οποία κατασκευάζει τα γραφικά στοιχεία του και τρέχει την προσομοίωση. Επίσης, ο καθορισμός των δρόμων και των κανόνων τους βασίζεται στο πρότυπο OpenDRIVE³¹ το οποίο προσφέρει τη δυνατότητα κατασκευής οδικού δικτύου με σαφώς καθορισμένους κανόνες. Ο CARLA λειτουργεί με τη λογική πελάτη-εξυπηρετητή (client-server) όπου η πλευρά του εξυπηρετητή είναι υπεύθυνη για οτιδήποτε σχετίζεται με την προσομοίωση όπως η πα-

³⁰<https://www.unrealengine.com/en-US/>

³¹<https://www.opendrive.com/>

ΚΕΦΑΛΑΙΟ 3. ΕΡΓΑΛΕΙΑ SOFTWARE

ροχή σωστών μετρήσεων στους αισθητήρες, ο υπολογισμός των φυσικών κινήσεων των διαφόρων συντελεστών (actors) και η ομαλή ανανέωση και ροή του κόσμου της προσομοίωσης. Από την άλλη πλευρά, μπορούν να υπάρχουν ένας ή περισσότεροι πελάτες (clients) οι οποίοι ελέγχουν και μεταβάλλουν την κατάσταση των συντελεστών (actors) μέσω του API που παρέχει ο CARLA το οποίο είναι γραμμένο και μπορεί να χρησιμοποιηθεί σε Python ή C++. Οποιοσδήποτε αλγόριθμος που πρόκειται να εφαρμοστεί στο συγκεκριμένο περιβάλλον θα πρέπει να γραφεί σε μία από τις δύο γλώσσες. Η λογική αυτή απεικονίζεται στο [σχήμα 3.1](#).



Σχήμα 3.1: Η αρχιτεκτονική πελάτη - εξυπηρετητή του CARLA.

Πηγή: https://carla.readthedocs.io/en/0.9.11/start_introduction/

Ο προσομοιωτής παρέχει τη δυνατότητα στους χρήστες να χρησιμοποιούν ελεύθερα όλες τις δυνατότητες που προσφέρει όπως τα οχήματα τα οποία είτε κινούνται αυτόνομα με ρεαλιστικό τρόπο είτε προγραμματίζονται ρητά από το χρήστη για να ακολουθούν μια συγκεκριμένη κατεύθυνση, τους πεζούς οι οποίοι μπορούν να ακολουθήσουν τυχαίες ή συγκεκριμένες συμπεριφορές ανάλογα με τον προγραμματισμό τους, διάφορες καιρικές συνθήκες και καταστάσεις φωτισμού του περιβάλλοντος αλλά και στατικά εμπόδια όπως κτίρια, σήματα, φωτεινού σηματοδότες δημιουργώντας έτσι ένα περιβάλλον όπου το όχημα συναντάει όσο περισσότερα στοιχεία του πραγματικού κόσμου είναι δυνατό και δοκιμάζεται σε πραγματικές συνθήκες κίνησης. Επίσης, παρέχει έτοιμες πόλεις με διαφορετικά χαρακτηριστικά η καθεμία ως προς τη δομή των δρόμων και των κτιρίων προσφέροντας έτσι τη δυνατότητα να δοκιμάζονται οι αλγόριθμοι σε διαφορετικά περιβάλλοντα αλλά και δυνατότητες κατασκευής μιας εντελώς προσαρμοσμένης πόλης ανάλογα με τις επιθυμίες του χρήστη. Ακόμη, σημαντική είναι η ευκολία στη χρήση των αισθητήρων καθώς παρέχει μια μεγάλη ποικιλία σύγχρονων αισθητήρων που χρησιμοποιούν τα αυτόνομα οχήματα δίνοντας παράλληλα τη δυνατότητα να λαμβάνουν μετρήσεις σε πραγματικό χρόνο.

Εκτός βέβαια των χαρακτηριστικών που διαθέτει ενσωματωμένα, ο προσομοιωτής CARLA προσφέρει και μεγάλη ευελιξία στην επικοινωνία και ενσωμάτωση τρίτων εξωτερικών προγραμμάτων και πλατφορμών. Για παράδειγμα, η πλατφόρμα ROS (Robot Operating System)³² χρησιμοποιείται για τον προγραμματισμό διάφορων ρομποτικών εφαρμογών και πολύ συχνά γίνεται χρήση και σε συστήματα αυτόνο-

³²<https://www.ros.org/>

3.2. ΕΡΓΑΛΕΙΟ ΚΑΤΑΣΚΕΥΗΣ ΤΗΣ ΔΙΕΠΑΦΗΣ NODE RED

μης οδήγησης. Ο CARLA διαθέτει ένα καλά καθορισμένο σύστημα επικοινωνίας με το ROS μέσω του πακέτου ROS bridge³³ παρέχοντας έτσι την ευελιξία να ανταλλάσσει μηνύματα και να χρησιμοποιεί τις δυνατότητες του ROS. Επιπλέον, σημαντική είναι και η ευκολία που προσφέρει στη συνεργασία με προγράμματα που προσφέρουν σενάρια κίνησης στα οδικά δίκτυα όπως το SUMO³⁴ και το PTV-Vissim³⁵ τα οποία προσφέρουν τη δυνατότητα κατασκευής προσαρμοσμένων σεναρίων κίνησης δημιουργώντας έτσι ένα πιο περίπλοκο και αληθινό περιβάλλον για το αυτόνομο όχημα που πρόκειται να δοκιμαστεί.

Οι αλγόριθμοι που αναπτύχθηκαν στη συγκεκριμένη διπλωματική δοκιμάστηκαν στο περιβάλλον προσομοίωσης CARLA για τους λόγους που αναφέρθηκαν παραπάνω. Παρακάτω στο [σχήμα 3.2](#) απεικονίζεται το γραφικό περιβάλλον του CARLA με τυχαία οχήματα και πεζούς.



Σχήμα 3.2: Το περιβάλλον του CARLA σε τυχαίο χάρτη.

Πηγή: <https://www.unrealengine.com/en-US/spotlights/carla-democratizes-autonomous-vehicle-r-d-with-free-open-source-simulator>

3.2 ΕΡΓΑΛΕΙΟ ΚΑΤΑΣΚΕΥΗΣ ΤΗΣ ΔΙΕΠΑΦΗΣ NODE RED

Το εργαλείο Node-RED³⁶ αποτελεί ένα προγραμματιστικό εργαλείο το οποίο χρησιμοποιείται για προγραμματισμό βασισμένο σε ροές (flow-based). Ο όρος flow-based programming αναφέρεται στον τρόπο με τον οποίο μπορεί να περιγραφεί η συμπεριφορά μιας εφαρμογής σαν ένα δίκτυο από μαύρα κουτιά (black-boxes) ή αλλιώς κόμβους (nodes) όπως αναφέρονται στο Node-RED. Κάθε κόμβος έχει έναν σαφώς καθορισμένο σκοπό και μια συγκεκριμένη λειτουργία που πρόκειται να επιτελέσει. Διαθέτει τα δικά του δεδομένα τα οποία είτε παράγει είτε λαμβάνει από

³³https://carla.readthedocs.io/en/0.9.11/ros_installation/

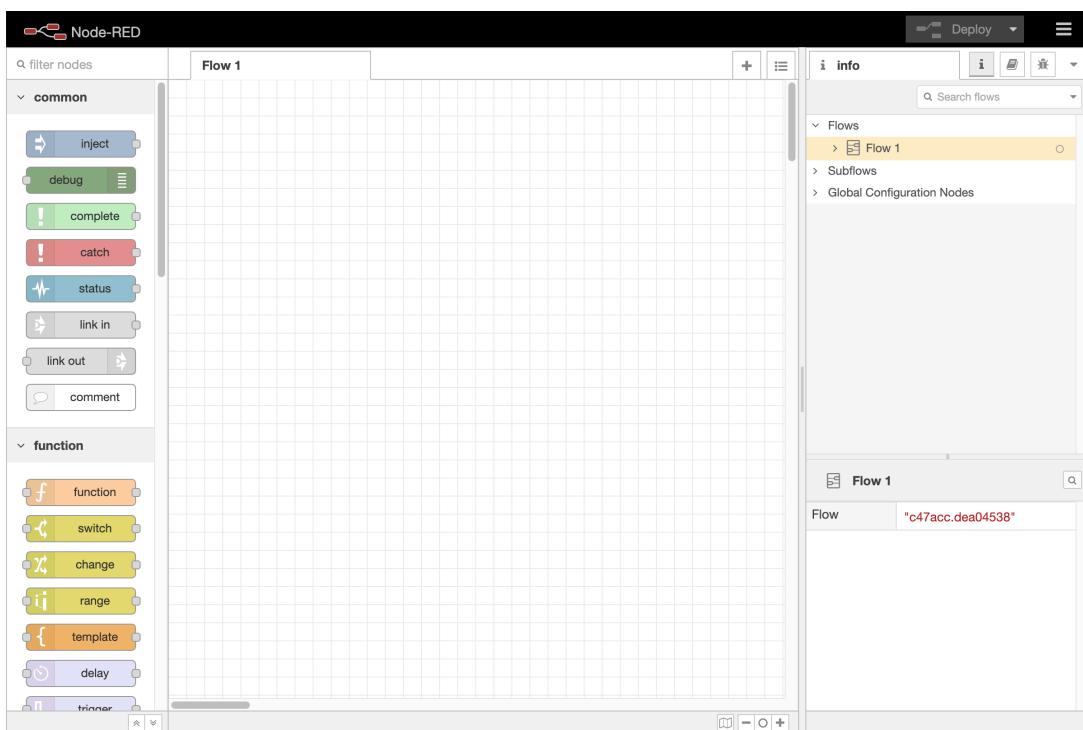
³⁴<https://www.eclipse.org/sumo/>

³⁵<https://www.eclipse.org/sumo/>

³⁶<https://nodered.org/>

ΚΕΦΑΛΑΙΟ 3. ΕΡΓΑΛΕΙΑ SOFTWARE

άλλους κόμβους και τα χρησιμοποιεί στη λειτουργία του ή τα μεταβιβάζει σε άλλους κόμβους με τους οποίους συνδέεται. Το Node-RED αποτελεί ένα πολύ φιλικό προς το χρήστη εργαλείο διότι δε χρειάζεται να έχει πλήρη γνώση του κώδικα του προγράμματος αλλά απλώς να γνωρίζει τη ροή του. Μπορεί εύκολα να προγραμματιστεί συνδέοντας κατάλληλα τους κόμβους του Node-RED και δίνοντας επιπλέον τη δυνατότητα να ενώνονται συσκευές, διάφορα API αλλά και υπηρεσίες διαδικτύου μεταξύ τους. Για να επιτευχθούν οι λειτουργίες του παρέχει έναν συντάκτη (editor) ο οποίος ανοίγει σε ένα πρόγραμμα περιήγησης (browser). Χρησιμοποιώντας την παλέτα του γραφικού περιβάλλοντος ο χρήστης μπορεί να προγραμματίσει την εφαρμογή του μέσω των κόμβων. Επίσης, παρέχει δυνατότητες προσαρμογής της ροής του προγράμματος μέσω της γλώσσας προγραμματισμού JavaScript δημιουργώντας προσαρμοσμένες λειτουργίες. Στο [σχήμα 3.3](#) απεικονίζεται το γραφικό περιβάλλον του Node-RED στο οποίο μπορεί ο χρήστης να χρησιμοποιήσει κόμβους από την αριστερή στήλη και να τους συνδέσει δημιουργώντας μία ροή (flow) προγράμματος.



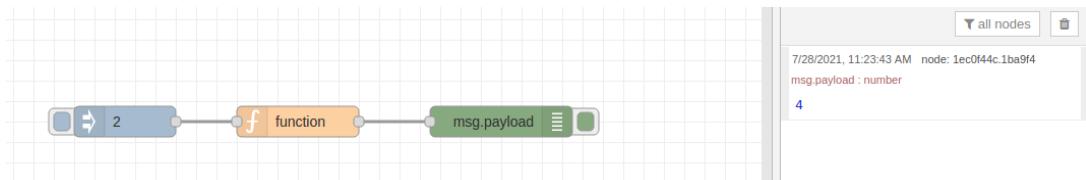
Σχήμα 3.3: Το γραφικό περιβάλλον του Node-RED στο οποίο μπορεί ο χρήστης να χρησιμοποιήσει κόμβους από την αριστερή στήλη και να τους συνδέσει δημιουργώντας μία ροή (flow) προγράμματος.

Πηγή: <https://nodered.org/docs/user-guide/editor/workspace/>

Το εργαλείο Node-RED αποτελεί ένα από τα πιο ευρέως χρησιμοποιούμενα προγράμματα για IoT εφαρμογές. Η ευκολία στη σύνδεση διαφορετικών στοιχείων το καθιστά ιδανικό για εφαρμογές όπου συνδέονται πολλές και διαφορετικές συσκευές και υπηρεσίες. Επίσης, διαθέτει ένα μεγάλο εύρος από κόμβους σχεδιασμένους για την κατασκευή της διεπαφής μιας εφαρμογής μέσω του επιπλέον πακέτου

3.3. ΠΡΩΤΟΚΟΛΛΑ ΕΠΙΚΟΙΝΩΝΙΑΣ

node-red-dashboard³⁷. Στη συγκεκριμένη διπλωματική η κατασκευή της διεπαφής βασίστηκε στο εργαλείο Node-RED. Ένα παράδειγμα μίας απλής ροής φαίνεται στο σχήμα 3.4 όπου απεικονίζεται μια απλή ροή στο περιβάλλον του NodeRED που περιλαμβάνει έναν κόμβο που προωθεί τον αριθμό 2 (μπλε κόμβος / inject node) στον επόμενο κόμβο που τον διπλασιάζει (πορτοκαλί κόμβος / function node) και τον προωθεί στον επόμενο κόμβο (πράσινος κόμβος / debug node) που τον τυπώνει στην οθόνη.



Σχήμα 3.4: Μια απλή ροή στο περιβάλλον του NodeRED που περιλαμβάνει έναν κόμβο που προωθεί τον αριθμό 2 (μπλε κόμβος / inject node) στον επόμενο κόμβο που τον διπλασιάζει (πορτοκαλί κόμβος / function node) και τον προωθεί στον επόμενο κόμβο (πράσινος κόμβος / debug node) που τον τυπώνει στην οθόνη.

3.3 ΠΡΩΤΟΚΟΛΛΑ ΕΠΙΚΟΙΝΩΝΙΑΣ

Προκειμένου να μπορέσει ένα σύστημα αυτόνομης οδήγησης να λειτουργήσει σωστά θα πρέπει να εξασφαλίζεται η ομαλή επικοινωνία μεταξύ των διαφόρων υποσυστημάτων του. Τα υποσυστήματα τα οποία χρησιμοποιούνται για την αντίληψη, την επιλογή συμπεριφοράς, τον σχεδιασμό της τροχιάς και γενικότερα ό,τι αφορά την αλγορίθμική προσέγγιση του προβλήματος έχουν υλοποιηθεί σε Python όπως αναφέρεται και στην επόμενη υποενότητα χρησιμοποιώντας τις βιβλιοθήκες του CARLA. Αυτό σημαίνει ότι μπορούν και λειτουργούν στο ίδιο υλικό του συστήματος οπότε και η επικοινωνία τους είναι προστή μέσω της ίδιας της γλώσσας.

Εκτός όμως από το αλγορίθμικό κοιμάτι, υπάρχει και η διεπαφή του συστήματος η οποία, όπως αναφέρθηκε προηγουμένως, έχει υλοποιηθεί στην πλατφόρμα Node-RED. Η διεπαφή μπορεί να τρέχει είτε στο ίδιο υλικό με το υπόλοιπο κομμάτι του προγράμματος είτε απομακρυσμένα από κάποιο άλλο σύστημα δημιουργώντας έτσι την ανάγκη για χρήση κάποιου εξωτερικού συστήματος επικοινωνίας. Για την επικοινωνία αυτή, λοιπόν, χρησιμοποιήθηκε το πρωτόκολλο MQTT³⁸ το οποίο χρησιμοποιείται συχνά σε IoT εφαρμογές διότι αποτελεί ένα αρκετά ελαφρύ και εύχρηστο πρωτόκολλο επικοινωνίας το οποίο μπορεί και συνδέει διάφορες απομακρυσμένες συσκευές μεταξύ τους.

Το MQTT πρωτόκολλο χρησιμοποιεί δύο κύριες διαδικτυακές οντότητες οι οποίες είναι ο μεσίτης μηνυμάτων (message broker) και οι πελάτες (clients)³⁹. Ο μεσίτης μηνυμάτων (message broker) είναι ένας εξυπηρετητής (server) ο οποίος λαμβάνει όλα τα μηνύματα από τους πελάτες (clients) και τα δρομολογεί στους κατάλληλους

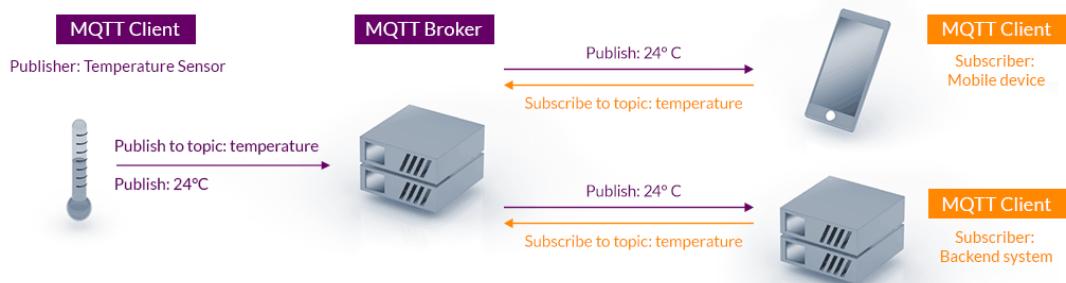
³⁷<https://flows.nodered.org/node/node-red-dashboard>

³⁸<https://mqtt.org/>

³⁹<https://en.wikipedia.org/wiki/MQTT>

ΚΕΦΑΛΑΙΟ 3. ΕΡΓΑΛΕΙΑ SOFTWARE

προορισμούς δηλαδή άλλους πελάτες (clients). Ένας MQTT πελάτης μπορεί να είναι οποιαδήποτε συσκευή από μικροεπεξεργαστές έως και πλήρως λειτουργικούς εξυπηρετητές οι οποίοι μπορούν και τρέχουν μια MQTT βιβλιοθήκη και μπορούν να συνδέονται σε έναν MQTT broker. Η πληροφορία στα MQTT πρωτόκολλα είναι οργανωμένη σε topics και η λειτουργία του βασίζεται στη λογική publish/subscribe. Οι publishers είναι οι πελάτες (clients) οι οποίοι δημοσιεύουν τα δεδομένα που διαθέτουν στο topic που τους ενδιαφέρει. Από την άλλη πλευρά οι subscribers είναι αυτοί οι πελάτες (clients) οι οποίοι λαμβάνουν τα δεδομένα από το topic που τους ενδιαφέρει. Στην περίπτωση του MQTT πρωτοκόλλου, όταν ένας publisher διαθέτει κάποια δεδομένα τα οποία επιθυμεί να μοιραστεί με άλλους clients, τότε στέλνει ένα ανάλογο μήνυμα με τα δεδομένα στον broker του συστήματος. Έπειτα, ο broker διανέμει τα δεδομένα σε όσους clients έχουν εγγραφεί (subscribe) στο topic που δημοσιεύτηκαν τα δεδομένα. Με αυτή τη λογική ούτε οι publishers χρειάζεται να γνωρίζουν δεδομένα όπως η τοποθεσία των subscribers ούτε οι subscribers τις αντίστοιχες πληροφορίες των publishers. Ο broker που χρησιμοποιείται για τη διανομή των μηνυμάτων είναι ο Mosquitto⁴⁰ ο οποίος είναι πλήρως συμβατός με το πρωτόκολλο MQTT. Για τη μετάδοση των μηνυμάτων βασίζεται στο πρωτόκολλο TCP⁴¹. Για την αρυπτογράφηση των μηνυμάτων και τη γενικότερη ασφάλεια του συστήματος χρησιμοποιεί TLS⁴² και για την αυθεντικοποίηση των πελατών είναι συμβατό με σύγχρονα πρωτόκολλα όπως το OAuth⁴³. Στο σχήμα 3.5 φαίνεται ο τρόπος λειτουργίας του MQTT και η λογική publish / subscribe στην οποία βασίζεται.



Σχήμα 3.5: Ο τρόπος λειτουργίας του MQTT και η λογική publish / subscribe στην οποία βασίζεται.

Πηγή: <https://mqtt.org/>

3.4 ΓΛΩΣΣΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ PYTHON

Η γλώσσα προγραμματισμού Python⁴⁴ αποτελεί μία υψηλού επιπέδου γλώσσα προγραμματισμού η οποία υποστηρίζει τα στοιχεία του αντικειμενοστραφούς προ-

⁴⁰<https://mosquitto.org/>

⁴¹https://en.wikipedia.org/wiki/Transmission_Control_Protocol

⁴²https://en.wikipedia.org/wiki/Transport_Layer_Security

⁴³<https://en.wikipedia.org/wiki/OAuth>

⁴⁴<https://www.python.org/>

γραμματισμού και χρησιμοποιείται σε μια μεγάλη ποικιλία εφαρμογών. Η απλή σύνταξη της και η εύκολη κατανόηση της δίνει τη δυνατότητα στους προγραμματιστές να συντάσσουν καθαρό και ευανάγνωστο κώδικα σε μικρό χρονικό διάστημα, για αυτό και προτιμάται συχνά η χρήση της. Επίσης, προσφέρει μία μεγάλη ποικιλία σε βιβλιοθήκες οι οποίες είναι γραμμένες σε Python και μπορούν εύκολα να εγκατασταθούν και να ενσωματωθούν στον κύριο κώδικα του προγράμματος. Το κύριο μειονέκτημα της είναι ο χρόνος εκτέλεσης των προγραμμάτων της καθώς στις περισσότερες περιπτώσεις είναι αρκετά μεγάλος ειδικά όταν πρόκειται για υψηλή πολυπλοκότητα των αλγορίθμων. Η συγκεκριμένη διπλωματική έχει υλοποιηθεί αποκλειστικά από τη γλώσσα προγραμματισμού Python για τους λόγους που αναφέρθηκαν παραπάνω αλλά και επειδή ο προσωμοιωτής CARLA⁴⁵ που χρησιμοποιήθηκε παρέχει ένα εξαιρετικά λειτουργικό API⁴⁶ γραμμένο σε Python διαθέτοντας μια μεγάλη ποικιλία από αντικείμενα και μεθόδους σχετικά με την αυτόνομη οδήγηση, τα οποία περιγράφονται εκτενώς στο επίσημο documentation του API.

3.5 ΒΙΒΛΙΟΘΗΚΕΣ ΚΑΙ ΚΛΑΣΕΙΣ

Στις περισσότερες μεγάλες εφαρμογές που περιλαμβάνουν υλοποίηση κώδικα για να καταστεί εφικτή η υλοποίηση του τόσο σε επίπεδο λογικής όσο και σε επίπεδο απόδοσης είναι συχνά απαραίτητο να χρησιμοποιηθούν εξωτερικές βιβλιοθήκες οι οποίες παρέχουν έτοιμες υλοποιήσεις σε μαθηματικούς υπολογισμούς, επεξεργασία και οργάνωση των δεδομένων, επεξεργασία εικόνας κ.α. Εφόσον, όπως αναφέρθηκε προηγουμένως, η γλώσσα προγραμματισμού που χρησιμοποιήθηκε είναι η Python, έγινε χρήση και κάποιων βιβλιοθηκών γραμμένων σε Python, για να επιτευχθεί ευκολότερα και πιο αξιόπιστα η υλοποίηση. Οι βιβλιοθήκες αυτές παρατίθενται παρακάτω:

- **NumPy**⁴⁷: Το κύριο χαρακτηριστικό που προσφέρει η βιβλιοθήκη NumPy, είναι η ευκολία στη χρήση πινάκων (arrays) και μητρώων (matrices) και η εφαρμογή μαθηματικών πράξεων σε αυτές. Υποστηρίζει παράλληλα μια μεγάλη ποικιλία μαθηματικών συναρτήσεων από τον τομέα της γραμμικής άλγεβρας, της μαθηματικής ανάλυσης και πολλούς ακόμη προσφέροντας ταυτόχρονα καλύτερη απόδοση σε σύγκριση με χειροκίνητες υλοποιήσεις.
- **SciPy**⁴⁸: Η βιβλιοθήκη αυτή χρησιμοποιείται κυρίως για μαθηματικές πράξεις. Η διαφορά της με τη NumPy βρίσκεται στο ότι αυτή διαθέτει περισσότερα εργαλεία για μαθηματικές πράξεις όπως η ολοκλήρωση, η γραμμική παρεμβολή, οι διάφοροι μετασχηματισμοί (Fourier, Laplace κλπ.), η επεξεργασία εικόνας, οι στατιστικές συναρτήσεις κ.α. και επικεντρώνεται σε γενικότερου σκοπού δομές δεδομένων και όχι μόνο σε πίνακες και μήτρες.

⁴⁵<https://carla.org/>

⁴⁶https://carla.readthedocs.io/en/latest/python_api/

⁴⁷<https://numpy.org/>

⁴⁸<https://www.scipy.org/>

ΚΕΦΑΛΑΙΟ 3. ΕΡΓΑΛΕΙΑ SOFTWARE

- **pandas⁴⁹**: Η συγκεκριμένη βιβλιοθήκη χρησιμοποιείται για τον καλύτερο χειρισμό και οργάνωση των δεδομένων και την ευκολότερη ανάλυση τους. Τα δεδομένα οργανώνονται σε Dataframes δηλαδή πίνακες με προσαρμοσμένα ονόματα στηλών και γραμμών. Προσφέρει μεγάλη ευκολία στην ένωση, στον ανασχηματισμό των διαστάσεων, στο φιλτράρισμα κλπ. των Dataframes. Χρησιμοποιείται ευρέως σε προβλήματα Μηχανικής Μάθησης.
- **commlib-py**: Η συγκεκριμένη βιβλιοθήκη υλοποιήθηκε σε παλαιότερη εργασία από το εργαστήριο του ISSEL⁵⁰ και χρησιμοποιείται για την αποστολή και τη λήψη μηνυμάτων από και προς τους clients που είναι υλοποιημένοι σε Python. Για την περίπτωση που χρησιμοποιείται ο MQTT broker για τη διανομή των μηνυμάτων, όπως στη συγκεκριμένη διπλωματική, τότε η commlib-py χρησιμοποιεί εσωτερικά για την ανταλλαγή των μηνυμάτων τη βιβλιοθήκη paho⁵¹. Η βιβλιοθήκη αυτή παρέχει υποστήριξη σε clients υλοποιημένους σε Python για την καλύτερη ανταλλαγή μηνυμάτων.

Το Python API που παρέχει ο CARLA προσομοιωτής διαθέτει ένα σύνολο από κλάσεις οι οποίες χρησιμοποιούνται για την υλοποίηση των αλγορίθμων που εκτελούνται πάνω στο αυτόνομο όχημα και για το συνολικό στήσιμο των σεναρίων κίνησης, του εξωτερικού περιβάλλοντος, των καιρικών συνθηκών και γενικότερα για οποιαδήποτε άλλη λειτουργικότητα υπάρχει κατά την εκτέλεση της προσομοίωσης. Παρακάτω αναλύονται δύο από τις βασικότερες κλάσεις που διαθέτει το API και χρησιμοποιήθηκαν για την υλοποίηση του αυτόνομου οχήματος:

- **Waypoint**: Το Waypoint⁵² αποτελεί μία κλάση του CARLA η οποία περιγράφεται ως ένα τρισδιάστατο κατευθυνόμενο σημείο πάνω στο χάρτη. Διαθέτει ένα μετασχηματισμό ο οποίος προσδιορίζει τη θέση ανάλογα με τις συντεταγμένες του και τον προσανατολισμό ανάλογα με τη λωρίδα του σημείου πάνω στο χάρτη. Επίσης, διαθέτει πληροφορίες σχετικά με το δρόμο και τη λωρίδα κυκλοφορίας στην οποία ανήκει όπως για παράδειγμα ο τύπος διαγράμμισης της αριστερής και δεξιάς λωρίδας, το πλάτος της λωρίδας που ανήκει, εάν βρίσκεται μέσα σε διασταύρωση. Όλη η πληροφορία που διαθέτει η κλάση Waypoint λαμβάνεται από το OpenDRIVE αρχείο πάνω στο οποίο βασίζεται ο χάρτης που τρέχει η προσομοίωση. Με λίγα λόγια αποτελούν πλήρως καθορισμένα σημεία επάνω στους δρόμους του χάρτη τα οποία όταν δοθούν ως είσοδος σε ένα σύστημα τότε το σύστημα μπορεί να έχει πλήρη γνώση για τα σημεία στα οποία θα κινηθεί. Στην παρούσα εργασία ολόκληρη η τροχιά που διαγύει το αυτόνομο όχημα αποτελεί ένα σύνολο από waypoints τα οποία προσπελαύνει διαδοχικά το όχημα για να φτάσει στον τελικό προορισμό του.
- **Actor**: Η κλάση Actor προσδιορίζει όλα εκείνα τα στοιχεία μέσα στο περιβάλλον τα οποία παίζουν κάποιο ρόλο στην προσομοίωση ή μπορούν να κινηθούν εντός του περιβάλλοντος. Περιλαμβάνει στοιχεία όπως οι πεζοί, τα οχήματα, οι αισθητήρες, οι σημάνσεις των δρόμων κλπ. Διαθέτουν ένα μοναδικό id με

⁴⁹<https://pandas.pydata.org/>

⁵⁰<https://issel.ee.auth.gr/>

⁵¹<https://www.eclipse.org/paho/>

⁵²https://carla.readthedocs.io/en/0.9.12/python_api/#carlawaypoint



(α') Λογότυπο της βιβλιοθήκης NumPy
Πηγή: <https://numpy.org/>



(β') Λογότυπο της βιβλιοθήκης SciPy
Πηγή: <https://www.scipy.org/>



(γ') Λογότυπο της βιβλιοθήκης Pandas
Πηγή: <https://pandas.pydata.org/>

Σχήμα 3.6: Οι βιβλιοθήκες της Python που χρησιμοποιήθηκαν

το οποίο μπορούν να αναγνωρίζονται μέσα στο περιβάλλον αλλά και ένα αναγνωριστικό του τύπου τους. Επίσης, οι κλάσεις Vehicle, Walker κλπ που υλοποιούν κάθε Actor ξεχωριστά κληρονομούν την κλάση Actor που σημαίνει ότι διαθέτουν όλες τις μεθόδους του. Η κλάση Actor περιλαμβάνει πολύ σημαντικές μεθόδους όπως μεθόδους εύρεσης της θέσης, του προσανατολισμού, της ταχύτητας και της επιτάχυνσης δίνοντας τη δυνατότητα στο σύστημα να γνωρίζει την κατάσταση όλων των actors. Στην παρούσα εργασία το σύνολο των οχημάτων, πεζών, σημάνσεων και αισθητήρων που έχουν χρησιμοποιηθεί αποτελούν αντικείμενα της κλάσης Actor.

4

Υλοποιήσεις

Η παρούσα διπλωματική καταπιάνεται με το πρόβλημα της ανάπτυξης μιας αναλυτικής διεπαφής που θα μπορεί να χρησιμοποιεί ο χρήστης είτε εντός του οχήματος είτε απομακρυσμένα δίνοντας του τη δυνατότητα να καθορίζει τη συμπεριφορά του αυτοκινήτου όπως επιθυμεί απαλλάσσοντας τον από τις περισσότερες και κουραστικές χειροκίνητες κινήσεις που απαιτούνται κατά την κλασική οδήγηση. Εκτός αυτού, κεντρικό στοιχείο ανάλυσης στην εργασίας αποτελεί η δυνατότητα που έχει ο χρήστης να ορίζει τη συμπεριφορά που επιθυμεί να ακολουθήσει το όχημα του ως προς την τήρηση των κανόνων και τον βαθμό επιθετικότητας δίνοντας τιμές για τις δύο παραμέτρους. Η συγκεκριμένη εργασία μελετά τον βαθμό στον οποίο επηρεάζουν αυτές οι τιμές την εύρυθμη λειτουργία του οχήματος εντός της πόλης.

Οι υλοποιήσεις μπορούν να χωριστούν σε δύο μεγάλες κατηγορίες λόγω και των εργαλείων που χρησιμοποιεί η κάθε κατηγορία:

- Ανάπτυξη της διεπαφής (interface) στο flow-based προγραμματιστικό εργαλείο NodeRED.
- Ανάπτυξη αλγορίθμων και επίλυση των υποπροβλημάτων αυτόνομης οδήγησης στον προσδομοιωτή CARLA και υλοποίηση ενός συστήματος επικοινωνίας με τη διεπαφή.

Στις επόμενες υποενότητες περιγράφεται η αρχιτεκτονική που σχεδιάστηκε για τη λειτουργία του συστήματος και αναλύονται τα επιμέρους υποσυστήματα που διαθέτει.

4.1 ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΣΥΣΤΗΜΑΤΟΣ

Στην υποενότητα αυτή περιγράφεται η αρχιτεκτονική η οποία σχεδιάσθηκε για τους σκοπούς της συγκεκριμένης εργασίας. Η αρχιτεκτονική του συστήματος που

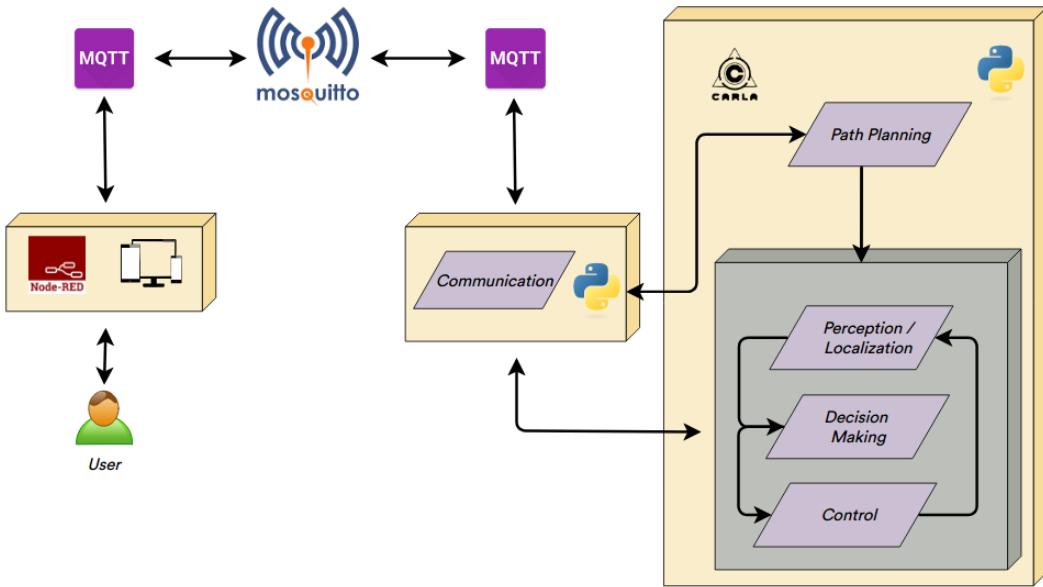
4.2. ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΔΙΕΠΑΦΗΣ ΤΟΥ NODERED

απεικονίζεται και στο [σχήμα 4.1](#) αποτελείται από τρία κεντρικά υποσυστήματα όπου το καθένα εσωτερικά έχει τη δική του λειτουργία και το δικό του σκοπό και επικοινωνεί με τα υπόλοιπα τμήματα του συνολικού συστήματος για την εύρυθμη λειτουργία του. Οι τρεις κεντρικοί κόμβοι του συστήματος είναι: α) Η διεπαφή υλοποιημένη σε NodeRED η οποία βρίσκεται στο διαδίκτυο και μπορεί να χρησιμοποιηθεί μόνο online, β) Ο broker μηνυμάτων υλοποιημένος από την πλατφόρμα Mosquitto, που συγκεντρώνει τα δεδομένα και τα διανέμει στα άλλα δύο υποσυστήματα, γ) Το υποσύστημα του αυτοκινήτου που τρέχει όλους τους αλγορίθμους στον προσομοιωτή CARLA.

Η ροή με την οποία λειτουργεί το σύστημα σε γενικές γραμμές ξεκινά από τα αριστερά στο [σχήμα 4.1](#) όπου φαίνεται ο τρόπος με τον οποίο ο χρήστης (οδηγός) εκκινεί ή παρεμβαίνει στη λειτουργία του οχήματος είτε βρίσκεται εντός του οχήματος είτε απομακρυσμένα. Μέσω, λοιπόν, του γραφικού περιβάλλοντος του NodeRED είναι σε θέση να δίνει εντολές για συγκεκριμένα καθήκοντα που επιθυμεί να εκτελεί το όχημα του εντός της οδικής κυκλοφορίας και να καθορίζει τη συμπεριφορά του. Οι εντολές που δίνει είναι γενικά αφηρημένες όπως ο καθορισμός του προορισμού που θέλει να φτάσει, ο βαθμός επιθετικότητας που θέλει να έχει στο δρόμο κλπ. Αυτό σημαίνει ότι δεν μπορεί να παρέμβει χειροκίνητα σε κάθε λειτουργία του αυτοκινήτου για αυτό και πρόκειται για ένα αυτόνομο όχημα υπό επιτήρηση. Έπειτα, μέσω του πρωτοκόλλου MQTT που αναφέρθηκε στο [υποκεφάλαιο 3.3](#) μεταβιβάζονται τα μηνύματα, που έρχονται από τη διεπαφή, στον Mosquitto broker του συστήματος. Εκεί γίνεται η δρομολόγηση των μηνυμάτων στους κατάλληλους clients. Με τη λογική publish / subscribe στα κατάλληλα topics γίνεται η διανομή των μηνυμάτων στο σύστημα του αυτοκινήτου που τρέχει στη συγκεκριμένη εργασία στον προσομοιωτή CARLA. Σε αυτή την περίπτωση, επίσης, μεσολαβεί το πρωτόκολλο MQTT για τη μεταφορά των μηνυμάτων. Το τρίτο υποσύστημα χωρίζεται σε δύο επιμέρους υποσυστήματα. Το ένα μέρος είναι το τμήμα επικοινωνίας το οποίο λαμβάνει τα μηνύματα που στέλνονται από το χρήστη μέσω της ροής που περιγράφηκε προηγουμένως και τα στέλνει στο δεύτερο τμήμα το οποίο είναι και αυτό που υλοποιεί όλους τους αλγορίθμους του οχήματος και προσαρμόζει τη λειτουργία του αυτοκινήτου ανάλογα με τις επιθυμίες του. Το υποσύστημα επικοινωνίας επικοινωνεί αρχικά μία φορά με το υποσύστημα σχεδίασης τροχιάς για να καθορίσει τον τελικό προορισμό πριν ξεκινήσει το όχημα την πορεία του. Η τροχιά που θα ακολουθήσει περνά στο υποσύστημα της κίνησης (γκρι πλαίσιο) για να ξεκινήσει η πορεία του οχήματος. Με το υποσύστημα που πραγματοποιεί την κίνηση του οχήματος (γκρι πλαίσιο) επικοινωνεί συνεχώς το υποσύστημα επικοινωνίας κατά τη διάρκεια της πορείας για ανταλλαγή πληροφοριών κίνησης. Για να ενημερώσει το συνολικό υποσύστημα του CARLA τον οδηγό για οποιοδήποτε θέμα ακολουθείται η αντίθετη πορεία των μηνυμάτων ακριβώς με την ίδια λογική.

4.2 ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΔΙΕΠΑΦΗΣ ΤΟΥ NODERED

Στην ενότητα αυτή θα περιγραφεί η διεπαφή (interface) η οποία κατασκευάστηκε στο NodeRED και δίνει εντολές στο όχημα για τον καθορισμό της συμπεριφοράς του. Η περιγραφή της λειτουργίας θα ξεκινήσει με την ανάλυση του γραφικού



Σχήμα 4.1: Η αρχιτεκτονική που σχεδιάστηκε για τη συγκεκριμένη διπλωματική.

περιβάλλοντος που βλέπει και παρεμβαίνει ο χρήστης και έπειτα θα αναλυθούν οι ροές που υλοποιήθηκαν πίσω από το interface.

4.2.1 Περιγραφή του γραφικού περιβάλλοντος

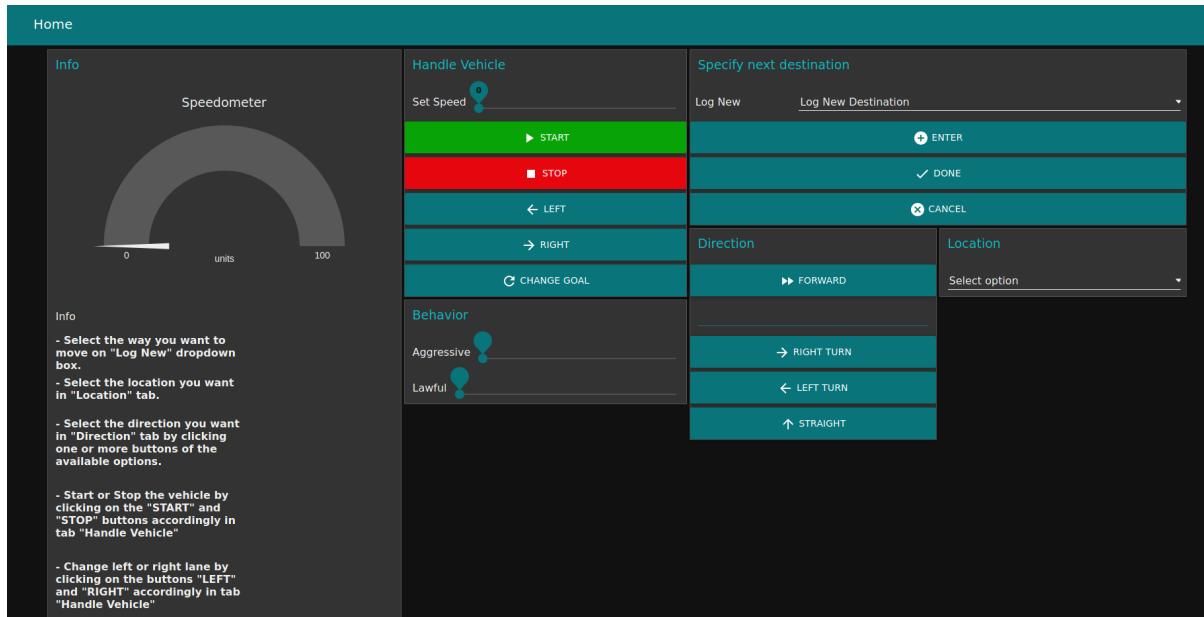
Όπως αναφέρθηκε για να ξεκινήσει το όχημα ή να παρέμβει στη λειτουργία του, ο χρήστης χρησιμοποιεί τη διεπαφή που υλοποιήθηκε στο NodeRED. Η διεπαφή αυτή με τα ξεχωριστά κομμάτια της φαίνεται στο [σχήμα 4.2](#). Η λειτουργία του interface θα μπορούσε να χωριστεί σε τρεις επιμέρους κατηγορίες ανάλογα και με τον σκοπό που εξυπηρετεί η κάθε κατηγορία λειτουργιών. Η ομαδοποίηση που μπορεί να γίνει είναι η παρακάτω:

- Πληροφορίες:** Περιλαμβάνει την Καρτέλα Info και τις ειδοποιήσεις (notifications).
- Ενέργειες πριν την εκκίνηση του οχήματος:** Περιλαμβάνει τις καρτέλες Specify next destination, Direction και Location.
- Ενέργειες κατά την πορεία του οχήματος:** Περιλαμβάνει τις καρτέλες Handle Vehicle και Behavior.

Κατηγορία πληροφοριών

Η πρώτη κατηγορία περιλαμβάνει μόνο την καρτέλα Info και βρίσκεται στα αριστερά του interface. Η καρτέλα Info περιλαμβάνει χαρακτηριστικά που αφορούν κυρίως την ενημέρωση και το feedback που λαμβάνει ο οδηγός από το σύστημα. Πιο συγκεκριμένα η καρτέλα αυτή ξεκινώντας από το πάνω μέρος της διαθέτει ένα ταχύμετρο (Speedometer) στο οποίο απεικονίζεται η ταχύτητα του αυτοκινήτου κάθε

4.2. ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΔΙΕΠΑΦΗΣ ΤΟΥ NODERED

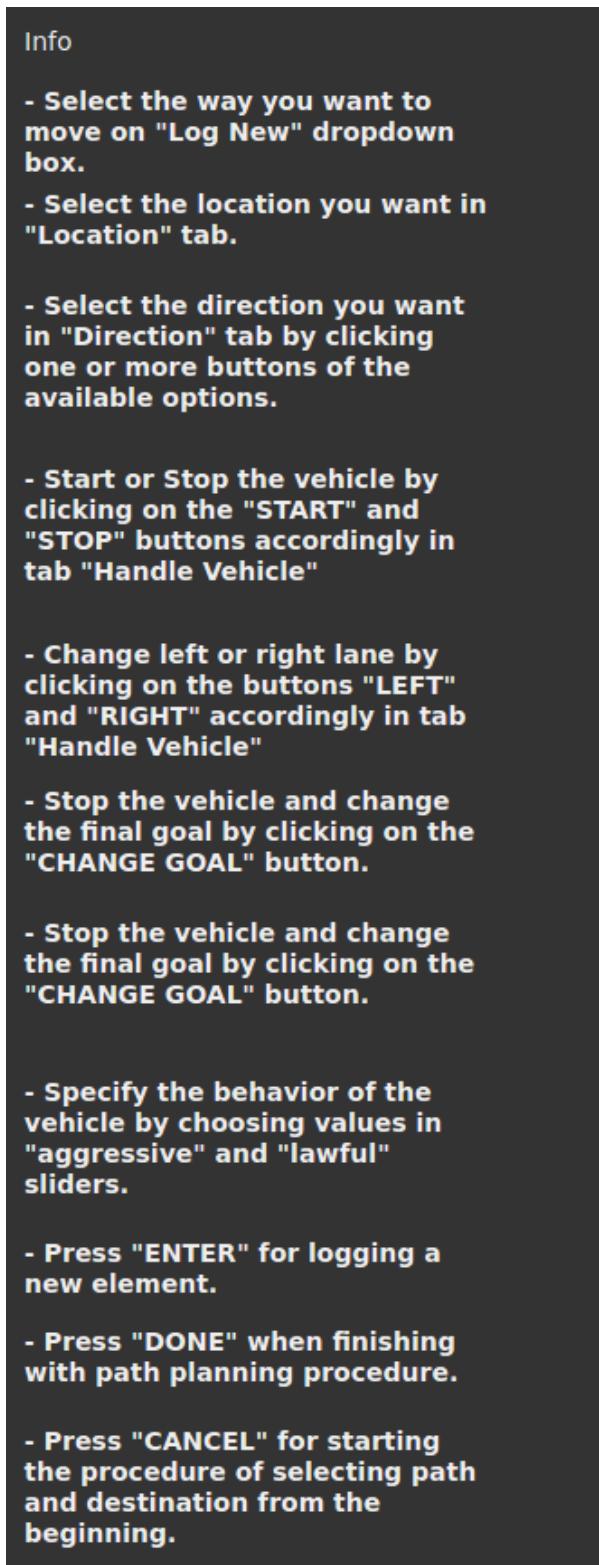


Σχήμα 4.2: Η διεπαφή που χρησιμοποιεί ο χρήστης για να παρέμβει στο όχημα.

χρονική στιγμή και δε διαφέρει σε κάτι από το κλασικό ταχύμετρο που διαθέτουν όλα τα οχήματα. Έπειτα, στο κάτω μέρος της ίδιας καρτέλας τυπώνονται οι οδηγίες τις οποίες πρέπει να ακολουθήσει ο χρήστης για να θέσει σε λειτουργία το όχημα. Οι οδηγίες αυτές αφορούν τη λειτουργία του κάθε κουμπιού και τη σειρά με την οποία πρέπει να χρησιμοποιούνται. Στο [σχήμα 4.3](#) απεικονίζονται όλες οι οδηγίες που εμφανίζονται στη διεπαφή για τον τρόπο χρήσης του κάθε στοιχείου της. Ανάμεσα στο ταχύμετρο και την ενότητα με τις οδηγίες υπάρχει μια μπάρα (text bar) στην οποία τυπώνονται διάφορες πληροφορίες είτε πριν την εκκίνηση του οχήματος είτε κατά τη διάρκεια της πορείας του. Για παράδειγμα, όταν ο χρήστης σχεδιάζει την πορεία που θα ακολουθήσει με τρόπο που θα αναλυθεί παρακάτω τότε στο συγκεκριμένο text bar τυπώνονται οι επιλογές του για τις τοποθεσίες από τις οποίες θα περάσει και για τις κατευθύνσεις που θα ακολουθήσει στις στροφές. Επίσης, τυπώνεται κατάλληλο μήνυμα που υποδηλώνει ότι το όχημα έχει φτάσει στον προορισμό του και ρωτάει τον χρήστη αν επιθυμεί να ορίσει άλλη τροχιά. Σε αυτή την κατηγορία θα μπορούσαν να προστεθούν και οι ειδοποιήσεις που δέχεται ο χρήστης σε διάφορα σημεία της πορείας ή καθώς θέτει σε λειτουργία το όχημα. Στο πάνω μέρος της οθόνης εμφανίζονται notifications όταν πρέπει να ειδοποιηθεί ο οδηγός για μία ενέργεια που πρέπει να ακολουθήσει. Όταν πατήσει για παράδειγμα το πλήκτρο "DONE" τότε εμφανίζεται ειδοποίηση ότι πρέπει να πατήσει στη συνέχεια το πλήκτρο "START" για να ξεκινήσει και έπειτα ειδοποιείται για να ορίσει ταχύτητα στο slider "Set speed".

Κατηγορία ενεργειών πριν την εκκίνηση του οχήματος

Η δεύτερη κατηγορία περιλαμβάνει τις καρτέλες Specify next destination, Direction και Location και βρίσκεται στα δεξιά του interface. Η συγκεκριμένη κατηγορία αφορά τις ενέργειες που πρόκειται να κάνει ο χρήστης πριν ξεκινήσει το όχημα



Σχήμα 4.3: Το σύνολο οδηγιών που εμφανίζεται στη διεπαφή για τη χρήση της.

4.2. ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΔΙΕΠΑΦΗΣ ΤΟΥ NODERED

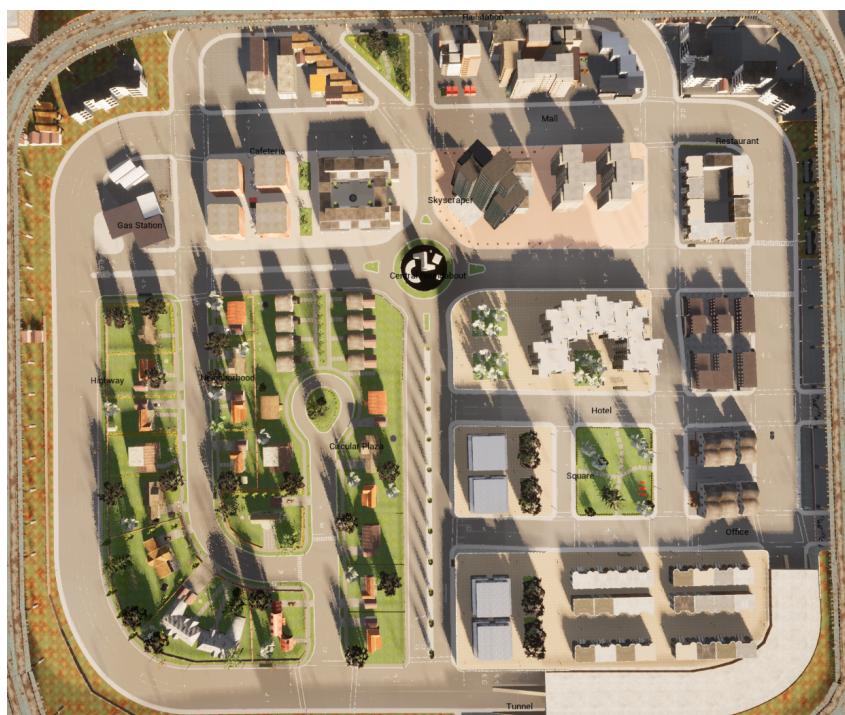
προκειμένου να προσδιορίσει τον προορισμό στον οποίο επιθυμεί να φτάσει. Αποτελεί ουσιαστικά την πρώτη ενέργεια που πρέπει να γίνει (όπως δηλώνουν και οι οδηγίες) για να γνωρίζει το όχημα την κατεύθυνση που πρόκειται να ακολουθήσει. Στο σημείο αυτό ο χρήστης έχει δύο επιλογές με τις οποίες μπορεί να προσδιορίσει την κατεύθυνση του οχήματος του. Αυτές μπορεί να τις επιλέξει από το αναπτυσσόμενο μενού (dropdown box) με την επωνυμία "Log new" όπως φαίνεται στο **σχήμα 4.5**. Οι επιλογές κίνησης που υπάρχουν είναι τα *Location* και *Direction*.

Η επιλογή *Location* δίνει τη δυνατότητα στο χρήστη να προσδιορίσει το σημείο στο οποίο επιθυμεί να κατευθυνθεί το όχημα του πάνω στο χάρτη. Έχουν οριστεί συγκεκριμένα σημεία πάνω στο χάρτη τα οποία μπορεί να επιλέξει και είναι κυρίως σημεία ενδιαφέροντος μέσα σε μια πόλη στα οποία μπορεί να επιθυμεί να επισκεφτεί κάποιος χρήστης. Για να φτάσει το όχημα στην τοποθεσία που επιλέχθηκε υπολογίζεται, με τρόπο που θα αναλυθεί σε επόμενη ενότητα, η ελάχιστη διαδρομή από το σημείο στο οποίο βρίσκεται εκείνη τη στιγμή το όχημα μέχρι την επιθυμητή τοποθεσία. Οι τοποθεσίες αυτές είναι οι παρακάτω και φαίνονται πάνω τον χάρτη του CARLA στο **σχήμα 4.4**:

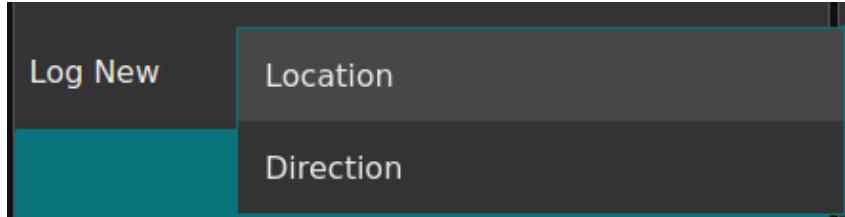
- Βενζινάδικο (Gas Station)
- Κεντρικός κυκλικός κόμβος (Central Roundabout)
- Κυκλική πλατεία (Circular Plaza)
- Τούνελ (Tunnel)
- Σιδηροδρομικός σταθμός (Railstation)
- Ουρανοξύστης (Skyscraper)
- Ξενοδοχείο (Hotel)
- Πλατεία (Square)
- Αυτοκινητόδρομος (Highway)
- Εμπορικό Κέντρο (Mall)
- Γραφείο (Office)
- Γειτονιά (Neighborhood)
- Καφετέρια (Cafeteria)
- Εστιατόριο (Restaurant)

Εφόσον ο χρήστης δώσει την επιλογή *Location* τότε αναγκαστικά θα πρέπει να μεταφερθεί στο tab που ονομάζεται *Location*. Σε αυτό το tab υπάρχει μόνο ένα αναπτυσσόμενο μενού (dropdown box) στο οποίο εμφανίζονται όλες οι επιλογές με τις τοποθεσίες που αναφέρθηκαν παραπάνω. Ο χρήστης μπορεί να επιλέξει μία ή περισσότερες τοποθεσίες και κάθε φορά που διαλέγει μία τοποθεσία θα πρέπει να πατάει το κουμπί "ENTER" για να συμπεριλάβει το σύστημα την επιθυμητή

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ



Σχήμα 4.4: Η πόλη του CARLA με τις διαθέσιμες τοποθεσίες που μπορεί να επιλέξει ο χρήστης.



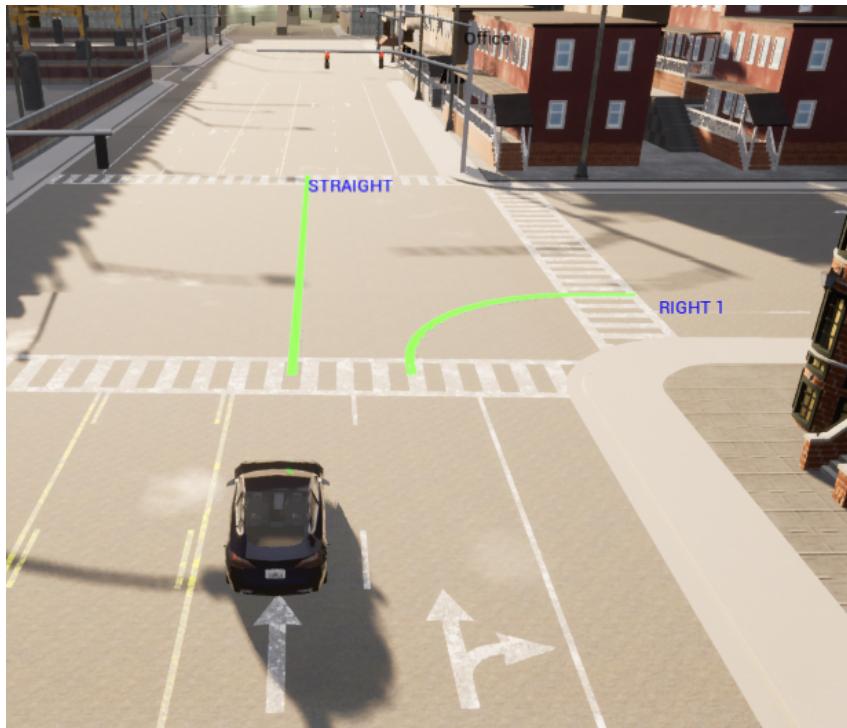
Σχήμα 4.5: Το μενού από το οποίο μπορεί ο χρήστης να επιλέξει κατεύθυνση ή τοποθεσία για το όχημα του.

τοποθεσία. Με βάση τη σειρά που επιλέχθηκαν οι τοποθεσίες δημιουργείται και η πιο σύντομη διαδρομή για να μεταφερθεί στα σημεία αυτά με τη σειρά που δόθηκαν. Για παράδειγμα, αν αρχικά ο χρήστης επιλέξει το Gas Station και έπειτα το Centre τότε το σύστημα θα βρει το συντομότερο μονοπάτι από την εκάστοτε τοποθεσία του αυτοκινήτου μέχρι το Gas Station και έπειτα από το Gas Station μέχρι το Centre. Για να ολοκληρώσει τη διαδικασία επιλογής τοποθεσιών και να προτρέψει το σύστημα να βρει το συντομότερο δρόμο για τον προορισμό του ο χρήστης θα πρέπει να επιλέξει το κουμπί DONE το οποίο δηλώνει ότι το όχημα είναι έτοιμο να ξεκινήσει.

Η δεύτερη επιλογή στο tab "Specify next destination" είναι το Direction και δίνει τη δυνατότητα στον χρήστη να επιλέξει την κατεύθυνση του οχήματος σε κάθε διασταύρωση που συναντάει από τις έγκυρες επιλογές που έχει εκείνη τη στιγμή. Για την ακρίβεια έχει τη δυνατότητα να επιλέξει την κατεύθυνση σε κάθε κομμάτι του γράφου πάνω στον οποίο είναι χτισμένο το οδικό δίκτυο στο αρχείο OpenDRIVE

4.2. ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΔΙΕΠΑΦΗΣ ΤΟΥ NODERED

που αναφέρθηκε στο [υποκεφάλαιο 3.1](#). Ο χρήστης φυσικά δεν επιλέγει τυχαία τη συμπεριφορά του στις στροφές αλλά το σύστημα των ενημερώνει για τις επιλογές κατεύθυνσης που έχει στη συγκεκριμένη διασταύρωση. Σε περίπτωση που δώσει εντολή σε μια διασταύρωση να κατεύθυνθεί σε στροφή που δεν υπάρχει στην τοπολογία του συγκεκριμένου οδικού κόμβου τότε τυπώνεται μήνυμα σφάλματος και δίνεται η δυνατότητα να δώσει ξανά επιλογή. Εφόσον ο χρήστης δώσει την επιλογή Direction τότε αναγκαστικά θα πρέπει να μεταφερθεί στο tab που ονομάζεται Direction. Σε αυτό το tab ο χρήστης μπορεί να επιλέξει την κατεύθυνση του οχήματος σε κάθε κομμάτι του οδικού δικτύου. Οι επιλογές που διαθέτει είναι οι FORWARD, RIGHT TURN, LEFT TURN και STRAIGHT. Για παράδειγμα, όταν το όχημα βρίσκεται στη διασταύρωση που φαίνεται στο [σχήμα 4.6](#) τότε οι επιλογές κατεύθυνσης είναι τα 2 μονοπάτια που απεικονίζονται με κατεύθυνση ευθεία και δεξιά. Το μήνυμα που τυπώνεται στη διεπαφή για τις διαθέσιμες επιλογές του χρήστη είναι αυτό του [σχήματος 4.7α'](#) και σε περίπτωση που δώσει την επιλογή STRAIGHT εμφανίζεται το μήνυμα της [εικόνας 4.7γ'](#). Σε περίπτωση που ο χρήστης δώσει αδύνατη επιλογή κατεύθυνσης όπως η LEFT τότε εμφανίζεται στη διεπαφή το μήνυμα της [εικόνας 4.7β'](#).



Σχήμα 4.6: Επιλογές κατεύθυνσης σε τυχαία διασταύρωση.

Οι διαθέσιμες επιλογές φαίνονται στο [σχήμα 4.10](#) και περιγράφονται παρακάτω:

- **RIGHT TURN:** Το πλήκτρο RIGHT TURN δίνει την επιλογή στο χρήστη εφόσον το επιτρέπει η τοπολογία του οδικού δικτύου να κινηθεί στα δεξιά στην επόμενη διασταύρωση που θα συναντήσει.
- **LEFT TURN:** Αντίστοιχα, το πλήκτρο LEFT TURN λειτουργεί με ακριβώς ίδιο τρόπο με το προηγούμενο πλήκτρο, δίνοντας την επιλογή στο όχημα να

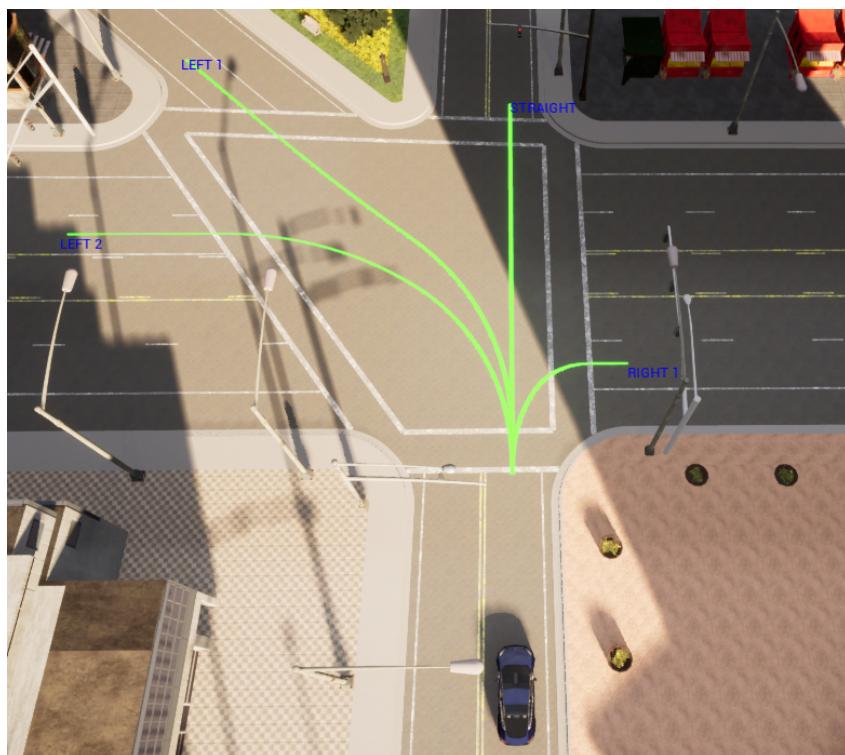
ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ



Σχήμα 4.7: Μηνύματα της διεπαφής σε διάφορες περιπτώσεις

κινηθεί αριστερά στην επόμενη διασταύρωση.

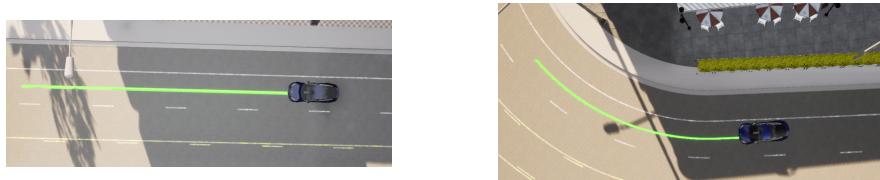
Για τις περιπτώσεις που υπάρχουν παραπάνω από μία δεξιές ή αριστερές στροφές, όπως αυτή στο [σχήμα 4.8](#), τότε κάθε στροφή συνοδεύεται και από έναν αριθμό και ο χρήστης θα πρέπει να εισάγει τον αριθμό της στροφής που επιθυμεί να ακολουθήσει στην άδεια μπάρα κάτω ακριβώς από το κουμπί FORWARD. Στην περίπτωση του [σχήματος 4.8](#) αν επιλέξει LEFT κατεύθυνση τότε θα πρέπει να εισάγει έναν εκ των αριθμών 1 και 2 για την επιλογή δρόμου.



Σχήμα 4.8: Διασταύρωση στην οποία ο χρήστης θα πρέπει να επιλέξει εκτός από την κατεύθυνση και τον αριθμό του δρόμου που θέλει να κινηθεί.

- **STRAIGHT:** Με το πλήκτρο STRAIGHT το όχημα συνεχίζει κανονικά ευθεία

4.2. ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΔΙΕΠΑΦΗΣ ΤΟΥ NODERED



- (α') Το μονοπάτι μετά την επιλογή 30 μέτρων FORWARD βρίσκεται στην ίδια ευθεία.
- (β') Το μονοπάτι μετά την επιλογή 30 μέτρων FORWARD ακολουθεί την καμπυλότητα του δρόμου.

Σχήμα 4.9: Διαφορετικά μονοπάτια για την επιλογή 30 μέτρων FORWARD.

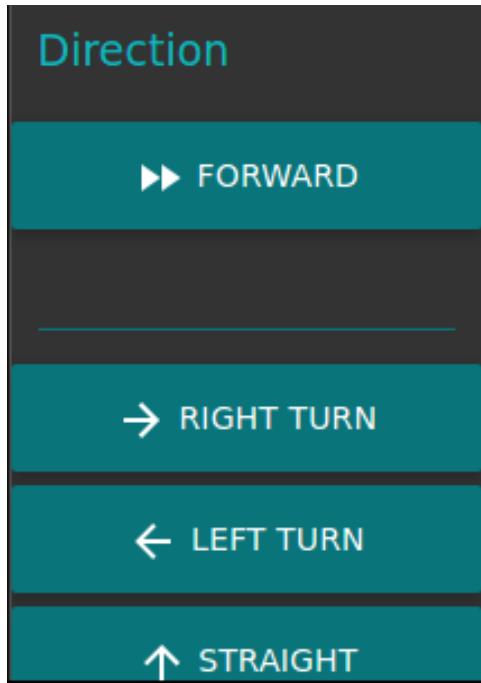
την πορεία του στη διασταύρωση χωρίς να αλλάξει κατεύθυνση.

Επισημαίνεται εδώ και για τις τρεις προηγούμενες επιλογές κίνησης ότι σε περίπτωση που το όχημα μπορεί να κατευθυνθεί προς τη φορά που ορίζει το πλήκτρο εμφανίζεται μήνυμα του τύπου “Going <direction> at the next junction”. Σε αντίθετη περίπτωση, δηλαδή όταν ο χρήστης επιλέξει κατεύθυνση στην οποία δεν επιτρέπεται να κινηθεί το όχημα ή δεν υπάρχει στην τοπολογία του χάρτη τέτοια στροφή τότε τυπώνεται το μήνυμα “Unable to go <direction> at the next junction”. Και στις δύο περιπτώσεις το <direction> αντικαθίσταται από την εκάστοτε πορεία κίνησης (ένα εκ των RIGHT, LEFT, STRAIGHT). Τα μηνύματα όπως αναφέρθηκε και σε προηγούμενη ενότητα τυπώνονται στο tab Info.

- **FORWARD:** Με το πλήκτρο FORWARD ο χρήστης δίνει εντολή στο όχημα του να προχωρήσει μπροστά όσα μέτρα επιθυμεί. Τα μέτρα που θέλει να προχωρήσει τα πληκτρολογεί στην άδεια μπάρα που υπάρχει ακριβώς κάτω από το κουμπί FORWARD, πατώντας έπειτα το πλήκτρο ENTER για να καταγραφεί η επιλογή του. Η επιλογή αυτή δε σημαίνει απαραίτητα ότι το όχημα θα κινηθεί ευθεία. Για παράδειγμα, αν ο δρόμος αναγκαστικά στρίβει προς κάποια κατεύθυνση, τότε συνεχίζει κανονικά στην πορεία του μέχρι την ολοκλήρωση των μέτρων που δόθηκαν από το χρήστη. Επίσης, αν υπάρξει διασταύρωση και δεν υπάρχει η επιλογή να κινηθεί ευθεία τότε επιλέγεται τυχαία μια από τις στροφές της διασταύρωσης. Για παράδειγμα, στο σχήμα 4.9 δίνονται δύο περιπτώσεις που έχει πατηθεί το πλήκτρο FORWARD με την εισαγωγή 30 μέτρων. Στο σχήμα 4.9α' το μονοπάτι που θα ακολουθήσει το όχημα είναι όλο στην ίδια ευθεία ενώ στο σχήμα 4.9β' το μονοπάτι ακολουθεί την καμπυλότητα του δρόμου μέχρι την ολοκλήρωση των 30 μέτρων. Το νόημα δηλαδή αυτού του πλήκτρου είναι να προχωρήσει το όχημα κάποια μέτρα πάνω στο δρόμο που ήδη βρίσκεται. Αυτή η λειτουργία έχει κυρίως λογική σε περιπτώσεις όπως όταν ο χρήστης καταλήξει σε μία τοποθεσία και θέλει να κινηθεί λίγα μέτρα ακόμη για να φτάσει στον τελικό προορισμό του ή για αποστάσεις που γνωρίζει ότι δε θέλει να αλλάξει κατεύθυνση μένοντας στον ίδιο δρόμο για τα επόμενα μέτρα που θα προσδιορίσει.

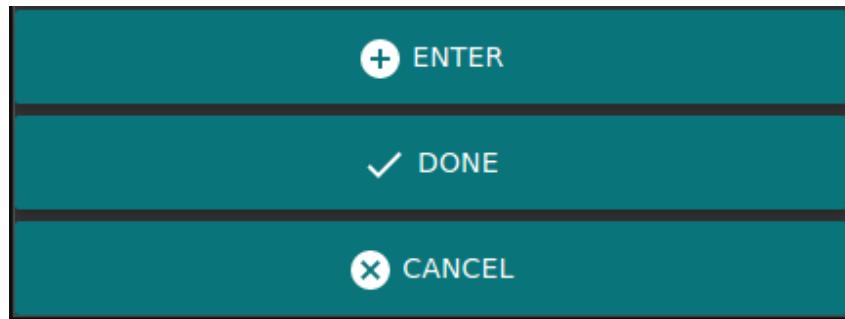
Επιπλέον, αν και αναφέρθηκε η λειτουργία κάποιων από τα κουμπιά που θα περιγραφούν παρακάτω αξίζει να αναφερθεί η γενική λειτουργία τους. Συγκεκριμένα,

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ



Σχήμα 4.10: Τα πλήκτρα FORWARD, RIGHT TURN, LEFT TURN και STRAIGHT.

ο λόγος γίνεται για τα κουμπιά ENTER, DONE και CANCEL τα οποία ολοκληρώνουν τη λειτουργία της συγκεκριμένης ομάδας χαρακτηριστικών του interface και φαίνονται στο [σχήμα 4.11](#). Παρακάτω, περιγράφεται ο σκοπός τους:



Σχήμα 4.11: Τα πλήκτρα ENTER, DONE και CANCEL.

- **ENTER:** Το κουμπί ENTER χρησιμοποιείται γενικά όταν χρειάζεται να καταγραφεί και να προστεθεί στη λειτουργία κάποιο επιπλέον καινούργιο στοιχείο. Το στοιχείο αυτό μπορεί να είναι κάποια τοποθεσία, τα μέτρα που θα χρειαστεί να προχωρήσει το όχημα ευθεία και όποια άλλη παράμετρος χρειάζεται να προστεθεί για να επιτευχθεί σωστά μία λειτουργία.
- **DONE:** Το κουμπί DONE χρησιμοποιείται όταν η διαδικασία σχεδιασμού τροχιάς από το χρήστη έχει ολοκληρωθεί, δηλαδή έχει δώσει τις επιθυμητές τοποθεσίες ή τις κατευθύνσεις στις διασταύρωσεις και επιθυμεί να περάσει στο στάδιο που το σύστημα σχεδιάζει την τροχιά και το όχημα ξεκινάει την πορεία

4.2. ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΔΙΕΠΑΦΗΣ ΤΟΥ NODERED

του. Στην ουσία δηλώνει ότι οποιαδήποτε διεργασία γενικότερα χρειαζόταν να γίνει πριν την εκκίνηση της πορείας του οχήματος έχει ολοκληρωθεί και το όχημα μπορεί να αρχίσει να κινείται.

- **CANCEL:** Κατά τη διαδικασία σχεδιασμού της τροχιάς σε περίπτωση που ο χρήστης επιθυμεί να ακυρώσει την επιλογή του για κάποια κατεύθυνση ή τοποθεσία τότε μπορεί να πατήσει το κουμπί CANCEL με το οποίο διαγράφει ό,τι έχει καταγράψει μέχρι εκείνη τη στιγμή και ξεκινάει από τη διαδικασία επιλογής προορισμού.

Συνοψίζοντας, ο χρήστης κατά τη διαδικασία επιλογής προορισμού έχει τη δυνατότητα να επιλέξει μία ή παραπάνω τοποθεσίες από τις διαθέσιμες ή να επιλέξει την κατεύθυνση στις επόμενες διασταυρώσεις που θα συναντήσει. Οι δύο αυτές επιλογές μπορούν να συνδυαστούν όσες φορές επιθυμεί ο χρήστης και με οποιαδήποτε σειρά εφόσον είναι επιτρεπτές από την τοπολογία του οδικού δικτύου. Με αυτό τον τρόπο ο χρήστης έχει τη δυνατότητα να σχεδιάσει μια εντελώς προσαρμοσμένη (custom) διαδρομή στην οποία θέλει να κινηθεί το όχημα του. Για παράδειγμα, δίνεται ένα τυχαίο σενάριο κίνησης με τη σειρά που επιλέγεται από το χρήστη και η παραγόμενη τροχιά απεικονίζεται στο [σχήμα 4.12](#):

1. Επιλογή Direction → STRAIGHT (μπλε χρώμα).
2. Επιλογή Direction → RIGHT (μπλε χρώμα).
3. Επιλογή Direction → FORWARD → 15 meters (πράσινο χρώμα).
4. Επιλογή Location → Highway (πορτοκαλί χρώμα).
5. Επιλογή Direction → STRAIGHT (μπλε χρώμα).
6. Επιλογή Location → Gas Station (πορτοκαλί χρώμα).

Κατηγορία ενεργειών κατά την πορεία του οχήματος

Η τρίτη κατηγορία περιλαμβάνει τις καρτέλες Handle Vehicle και Behavior και βρίσκεται στο μεσαίο τμήμα του interface. Η συγκεκριμένη κατηγορία αφορά τις ενέργειες που μπορεί να κάνει ο χρήστης κατά τη διάρκεια της πορείας του αυτοκινήτου σε περίπτωση που θέλει να μεταβάλλει τη συμπεριφορά του. Σε αυτή την ομάδα λειτουργιών ο χρήστης μεταφέρεται αφού ολοκληρώσει τη διαδικασία επιλογής προορισμού από την προηγούμενη κατηγορία και μπορεί να παραμείνει σε αυτή και να παρεμβαίνει στο όχημα καθ' όλη τη διάρκεια της διαδρομής. Όπως αναφέρθηκε και προηγουμένως εφόσον έχει πατηθεί το πλήκτρο DONE σημαίνει ότι έχει δοθεί εντολή στο σύστημα να σχεδιαστεί η τροχιά που ο χρήστης επιθυμεί να ακολουθήσει.

Έπειτα, ο χρήστης μεταφέρεται στην καρτέλα "Handle vehicle" το οποίο φαίνεται στο [σχήμα 4.13](#) για να ορίσει την ταχύτητα του οχήματος και να ξεκινήσει την πορεία του. Τα πλήκτρα της συγκεκριμένης καρτέλας περιγράφονται

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ



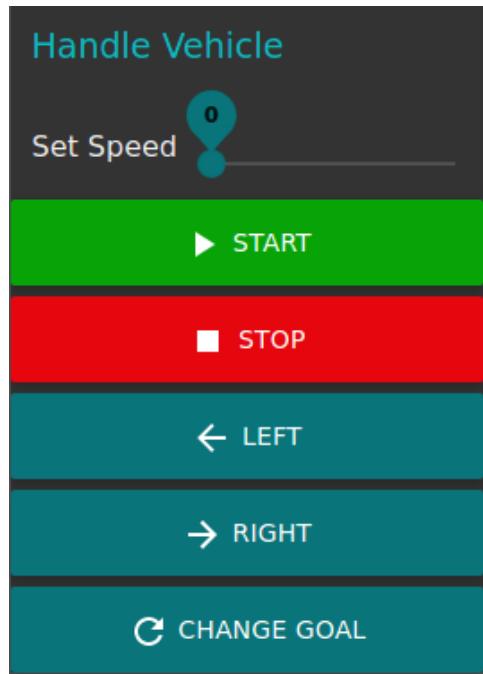
Σχήμα 4.12: Σχεδιασμένη τροχιά με συνδυασμό κατευθύνσεων και τοποθεσιών.

- **START:** Αρχικά, το σύστημα εμφανίζει ενημερωτικό μήνυμα στο χρήστη το οποίο τον προτρέπει να πατήσει το πράσινο πλήκτρο START από τη συγκεκριμένη καρτέλα για να ξεκινήσει η λειτουργία του οχήματος.
- **STOP:** Το κόκκινο πλήκτρο STOP μόλις πατηθεί σταματάει αμέσως το όχημα μηδενίζοντας την ταχύτητα του. Για να ξεκινήσει και πάλι το όχημα θα πρέπει να πατηθεί το πλήκτρο START που περιγράφηκε παραπάνω.
- **Slider "Set Speed":** Πριν ξεκινήσει το όχημα, θα πρέπει να γνωρίζει την ταχύτητα με την οποία επιθυμεί να ξεκινήσει οπότε εμφανίζεται και πάλι μήνυμα στο οποίο ο χρήστης ενημερώνεται ότι πρέπει να θέσει αρχική ταχύτητα από το slider "Set speed". Μόλις ο χρήστης θέσει την αρχική ταχύτητα του οχήματος τότε θα πρέπει να πατήσει και πάλι το πλήκτρο START για να ξεκινήσει το όχημα να κινείται. Για να αλλάξει την ταχύτητα του οχήματος οποιαδήποτε στιγμή τότε μπορεί να τη θέσει από το ίδιο slider.
- **RIGHT / LEFT:** Με τα πλήκτρα RIGHT και LEFT ο οδηγός έχει τη δυνατότητα να αλλάξει λωρίδα κυκλοφορίας καθώς κινείται δεξιά και αριστερά

4.2. ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΔΙΕΠΑΦΗΣ ΤΟΥ NODERED

αντίστοιχα.

- **CHANGE GOAL:** Η λειτουργικότητα του συγκεκριμένου tab ολοκληρώνεται με το πλήκτρο CHANGE GOAL με το οποίο το όχημα σταματάει την κίνηση του και το σύστημα επιστρέφει στο σημείο σχεδίασης τροχιάς από το χρήστη. Στην ουσία το συγκεκριμένο πλήκτρο δίνει τη δυνατότητα στον χρήστη να αλλάξει τον προορισμό του και τη διαδρομή του αφού ξεκινήσει το όχημα.



Σχήμα 4.13: Η καρτέλα Handle Vehicle του interface από το οποίο μπορεί να χειρίζεται το όχημα του.

Η κατηγορία αυτή ολοκληρώνεται με την καρτέλα Behavior η οποία φαίνεται στο [σχήμα 4.14](#). Η συγκεκριμένη καρτέλα έχει κομβικό ρόλο στη συγκεκριμένη διπλωματική καθώς στην ουσία μελετάται όπως θα φανεί και στα πειράματα παρακάτω η συμπεριφορά του οχήματος για διάφορες τιμές των sliders που θα περιγραφούν. Η καρτέλα Behavior διαθέτει δύο sliders, τα aggressive και lawful τα οποία λαμβάνουν διαχριτές τιμές από το 0 έως το 10 και περιγράφονται συνοπτικά παρακάτω:

- **Aggressive:** To slider Aggressive δηλώνει το ποσοστό επιθετικότητας που θέλει να έχει το όχημα κατά την πορεία του στο δρόμο. Η τιμή 10 υποδηλώνει ότι ο οδηγός θέλει να είναι όσο επιθετικότερος γίνεται, η τιμή 0 όσο το δυνατόν πιο προσεκτικός και όλες οι ενδιάμεσες τιμές αντιπροσωπεύουν μια αύξουσα επιθετική συμπεριφορά όσο η τιμή του aggressive πλησιάζει προς την τιμή 10. Αν και δεν υπάρχει σαφής ορισμός της επιθετικής συμπεριφοράς οδήγησης, η NHTSA⁵³ έχει κάνει μία προσπάθεια να ομαδοποιήσει κάποιες συμπεριφορές που θεωρούνται επιθετικές στον δρόμο⁵⁴. Τέτοιες συμπεριφορές μπορούν

⁵³https://en.wikipedia.org/wiki/National_Highway_Traffic_Safety_Administration

⁵⁴https://en.wikipedia.org/wiki/Aggressive_driving

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

να χαρακτηριστούν οι εξής: οι πολλές προσπεράσεις άλλων οχημάτων, οι μεγάλες ταχύτητες, οι συχνές αλλαγές λωρίδας, η πολύ κοντινή απόσταση από τα μπροστινά οχήματα και η αδιαφορία για πεζούς και λοιπά οχήματα μέσα στην κυκλοφορία. Αυτές είναι και οι συμπεριφορές επιθετικότητας που έχουν υλοποιηθεί στο πλαίσιο της συγκεκριμένης διπλωματικής και όσο μεγαλύτερη είναι η τιμή του aggressive τόσο πιθανότερο είναι να εκδηλώσει τέτοιου είδους συμπεριφορές.

- **Lawful:** Το δεύτερο slider ονομάζεται Lawful και δηλώνει το ποσοστό νομιμότητας που θέλει να ακολουθεί το όχημα κατά την πορεία του στο δρόμο. Πιο συγκεκριμένα, η τιμή 10 υποδηλώνει ότι το όχημα πρόκειται να κινηθεί τηρώντας απόλυτα τους νόμους της κυκλοφορίας, η τιμή 0 υποδηλώνει ότι θα έχει τη μέγιστη αδιαφορία ως προς τους κανόνες κυκλοφορίας και όλες οι ενδιάμεσες τιμές υποδηλώνουν αύξουσα νόμιμη συμπεριφορά όσο πλησιάζει η τιμή του slider προς την τιμή 10. Οι νόμιμες συμπεριφορές που έχουν συμπεριληφθεί στη συγκεκριμένη διπλωματική είναι η τήρηση των ορίων ταχύτητας, η σωστή συμπεριφορά σε φωτεινούς σηματοδότες και σημάνσεις STOP και η τήρηση των γραμμών κυκλοφορίας χωρίς να πατάει πάνω σε συνεχείς ή κίτρινες γραμμές. Όσο μεγαλύτερη, δηλαδή, είναι η τιμή του lawful τόσο πιο πιθανό είναι να εκδηλωθούν τέτοιες συμπεριφορές από το όχημα.

Φυσικά, η τιμή του ενός slider δεν αναιρεί την τιμή του άλλου δηλαδή ο οδηγός είναι σε θέση να ορίσει όποιες τιμές θέλει για τα δύο sliders αφήνοντας το όχημα μόνο του να προσαρμόζει τη συμπεριφορά του ανάλογα με αυτές τις τιμές.



Σχήμα 4.14: Η καρτέλα Behavior του interface από το οποίο μπορεί να αλλάξει τον βαθμό της νόμιμης και της επιθετικής συμπεριφοράς του οχήματος του.

4.2.2 Περιγραφή των ροών του NodeRED (low level υλοποίηση)

Η συγκεκριμένη υποενότητα εξηγεί τις ροές (flows) που έχουν υλοποιηθεί στο γραφικό περιβάλλον του NodeRED για να παράγουν το interface του χρήστη που περιγράφηκε στην προηγούμενη υποενότητα. Αποτελεί στην ουσία την αναλυτική περιγραφή του low level κομματιού της υλοποίησης του NodeRED. Για την καλύτερη ανάγνωση της συγκεκριμένης υποενότητας έγινε μία ομαδοποίηση ανάμεσα στις ροές που έχουν παρόμοιο τρόπο υλοποίησης έτσι ώστε να παρουσιαστούν μία φορά οι ροές και οι κόμβοι με παρόμοιο τρόπο λειτουργίας. Η πρώτη παράγραφος αυτής της υποενότητας κάνει μια γενική περιγραφή των όλων των κόμβων που

χρησιμοποιήθηκαν στις ροές και έπειτα στις επόμενες παραγράφους ακολουθεί μια πιο ειδική περιγραφή της κάθε ροής.

Γενική περιγραφή των κόμβων

Στη συγκεκριμένη παράγραφο περιγράφεται η γενική λειτουργία που προσφέρει σε μια ροή κάθε κόμβος που χρησιμοποιήθηκε. Οι κόμβοι που χρησιμοποιήθηκαν είναι οι παρακάτω:

- **Function:** Ο Function⁵⁵ κόμβος επιτρέπει να τρέχει στο εσωτερικό του κώδικας γραμμένος σε JavaScript και να επεξεργάζεται στην ίδια γλώσσα τα μηνύματα που περνάνε σαν είσοδος στον συγκεκριμένο κόμβο. Η είσοδος στον κόμβο function είναι ένα αντικείμενο το οποίο καλείται "msg" και συνήθως διαθέτει μία ιδιότητα η οποία καλείται "msg.payload" και περιέχει συνήθως το κύριο σώμα του μηνύματος.
- **JSON:** Ο JSON κόμβος⁵⁶ χρησιμοποιείται για να μετατρέψει μια αλφαριθμητική JSON μεταβλητή σε JavaScript αντικείμενο που αναπαριστά μια JSON μεταβλητή.
- **MQTT (Publisher):** Ο MQTT Publisher κόμβος χρησιμοποιείται για να δημοσιεύει τα δεδομένα σε έναν MQTT broker⁵⁷ όπως o Mosquitto. Τα δεδομένα δημοσιεύονται στο topic που ορίζεται στη διαμόρφωση του κόμβου. Επίσης, χρειάζεται να οριστεί η IP του broker και το port που χρησιμοποιεί για να επικοινωνήσει με τον broker.
- **MQTT (Subscriber):** Ο συγκεκριμένος κόμβος λειτουργεί σαν subscriber και χρησιμοποιείται για να λαμβάνει τα δεδομένα από το topic με το οποίο επικοινωνεί. Όπως και στην περίπτωση του κόμβου MQTT Publisher, θα πρέπει να οριστεί η IP του Mosquitto broker και το port που χρησιμοποιεί.
- **Button:** Ο κόμβος Button δημιουργεί ένα πλήκτρο στη διεπαφή και πατώντας το στέλνεται το μήνυμα που ορίζεται στο πεδίο "Payload".
- **Notification:** Ο κόμβος Notification χρησιμοποιείται για να εμφανίζεται μία pop-up ειδοποίηση στην οθόνη. Το μέρος της οθόνης που εμφανίζεται η ειδοποίηση ορίζεται στη διαμόρφωση του κόμβου.
- **Gauge:** Ο κόμβος Gauge λειτουργεί σαν μετρητής ο οποίος εμφανίζει αναλογικά τη μετρούμενη ποσότητα.
- **Text:** Ο κόμβος αυτός δέχεται ως είσοδο ένα μήνυμα αλφαριθμητικού τύπου το οποίο εμφανίζει στο περιβάλλον της διεπαφής και δεν μπορεί να μετατραπεί ή να δεχτεί επεξεργασία από το χρήστη αλλά παραμένει στην οθόνη της διεπαφής σαν κείμενο.

⁵⁵ <https://nodered.org/docs/user-guide/writing-functions>

⁵⁶ <https://cookbook.nodered.org/basic/convert-json>

⁵⁷ <https://cookbook.nodered.org/mqtt/connect-to-broker>

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

- **Text Input:** Ο κόμβος αυτός δημιουργεί μία κενή μπάρα στη διεπαφή στην οποία ο χρήστης μπορεί να εισάγει δεδομένα.
- **Slider:** Ο κόμβος Slider υλοποιεί τα sliders που εμφανίζονται στη διεπαφή στα οποία ο χρήστης μπορεί να ορίσει την τιμή που επιθυμεί για το εκάστοτε slider ολισθαίνοντας πάνω στην μπάρα που δημιουργείται. Η μπάρα του slider έχει συγκεκριμένο εύρος το οποίο ορίζεται στη διαμόρφωση του κόμβου.
- **Dropdown Box:** Ο κόμβος dropdown box δημιουργεί ένα αναπτυσσόμενο μενού στη διεπαφή από το οποίο ο χρήστης μπορεί να επιλέξει πατώντας οποιαδήποτε επιλογή θέλει από τις διαθέσιμες επιλογές που υπάρχουν.

Υλοποίηση των πλήκτρων (buttons)

Στη συγκεκριμένη παράγραφο έχουν συγκεντρωθεί οι υλοποιήσεις όλων των πλήκτρων που αναφέρθηκαν παραπάνω διότι οι ροές που τρέχουν για να προκύψει το αποτέλεσμα τους έχουν ακριβώς τον ίδιο τρόπο λειτουργίας. Οι ροές αυτές απεικονίζονται στο [σχήμα 4.16](#). Για την ευκολότερη ανάγνωση και διευκόλυνση της ροής του κειμένου θα αναλυθεί ενδεικτικά μόνο η πρώτη ροή του σχήματος δηλαδή αυτή που υλοποιεί το πλήκτρο START. Οι υπόλοιπες ροές του σχήματος λειτουργούν κατ' αναλογία με ακριβώς την ίδια λογική. Η ροή της υλοποίησης του πλήκτρου START αποτελείται από τρεις ξεχωριστούς κόμβους:

- **Button:** Στον συγκεκριμένο κόμβο το μήνυμα που στέλνεται πατώντας πάνω στο πλήκτρο είναι αλφαριθμητικού (string) τύπου και περιέχει το αλφαριθμητικό "start". Το μήνυμα περνάει σαν έξοδος στον επόμενο κόμβο ο οποίος είναι τύπου function. Στα υπόλοιπα πλήκτρα το μήνυμα που περνάει είναι είτε αλφαριθμητικού τύπου περιέχοντας την αντίστοιχη εντολή είτε boolean τύπου περιέχοντας τιμή "Alethyst" για να γνωρίζει ο επόμενος κόμβος ότι το κουμπί έχει πατηθεί.
- **Function:** Στον συγκεκριμένο function κόμβο το μήνυμα που έρχεται σαν είσοδος από τον Button κόμβο μετατρέπεται σε μια JSON⁵⁸ μορφή μηνύματος και περνάει σαν έξοδος στον επόμενο κόμβο. Το μήνυμα παίρνει τη μορφή που φαίνεται στο [σχήμα 4.15](#) και είναι απαραίτητο να βρίσκεται σε αυτή τη μορφή για να περάσει στον επόμενο κόμβο και να μπορεί να λειτουργήσει σωστά με τη βιβλιοθήκη commlib-py που αναλύθηκε στο [υποκεφάλαιο 3.5](#) και είναι υπεύθυνη για τη μετάδοση των μηνυμάτων στο περιβάλλον του CARLA. Η διαφορά με τους κόμβους function των υπόλοιπων πλήκτρων βρίσκεται στο πεδίο "data" της JSON μεταβλητής και πιο συγκεκριμένα στο πεδίο του με όνομα "trigger" το οποίο παίρνει ένα όνομα αντίστοιχο της λειτουργίας του.

{ "data": { "trigger": msg.payload}, "meta": { "timestamp": msg.time, "properties": { "content_type": "application/json", "content_encoding": "utf8"}}}

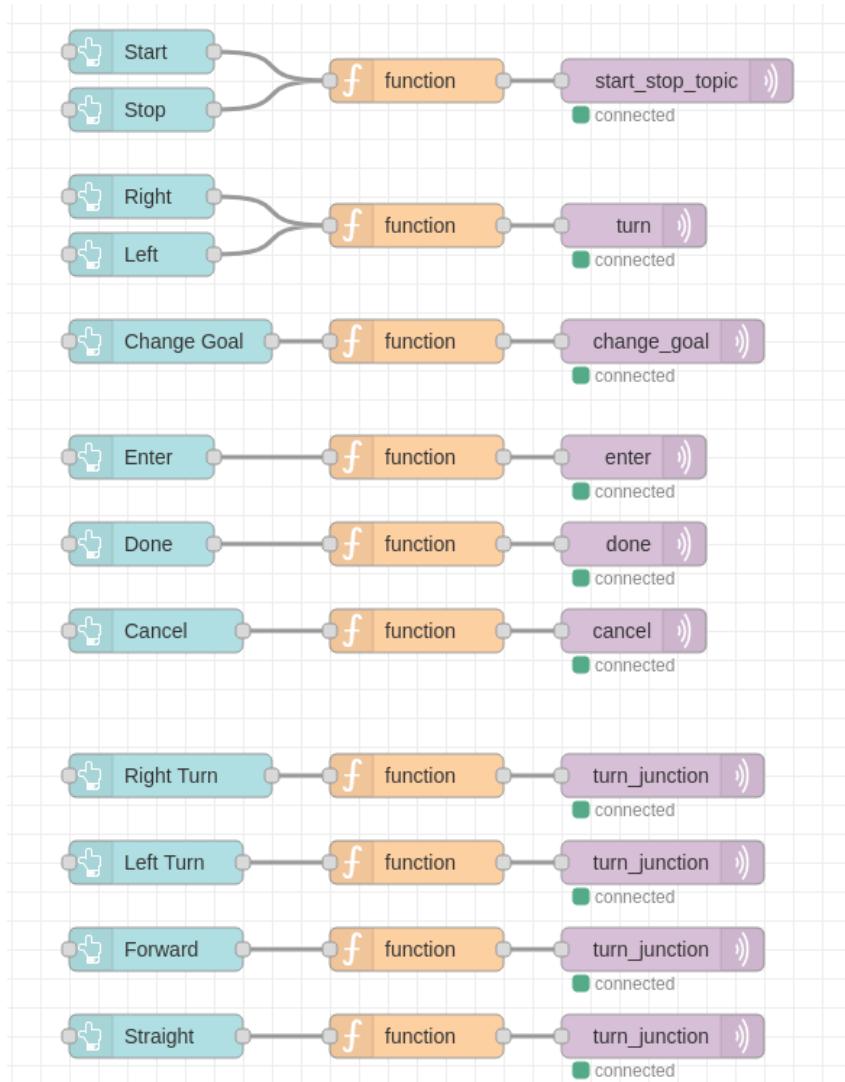
Σχήμα 4.15: Απεικονίζεται η μορφή της JSON μεταβλητής στην οποία μετατρέπεται το μήνυμα.

⁵⁸<https://www.json.org/json-en.html>

4.2. ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΔΙΕΠΑΦΗΣ ΤΟΥ NODERED

- MQTT (Publisher):** Στη συγκεκριμένη περίπτωση ο κόμβος MQTT Publisher δημιουργεύει τα δεδομένα στον Mosquitto broker και τα κάνει διαθέσιμα στους clients που ενδιαφέρονται να τα χρησιμοποιήσουν. Τα δεδομένα που δέχεται σαν είσοδος είναι η έξοδος του κόμβου function που περιγράφηκε προηγουμένως και το μήνυμα που δημιουργεύει είναι η JSON μεταβλητή με τα δεδομένα. Το όνομα του κόμβου υποδηλώνει σε αυτή την περίπτωση το topic στα οποία δημοσιεύονται τα δεδομένα. Η μόνη διαφορά ανάμεσα στους MQTT κόμβους των άλλων πλήκτρων είναι το όνομα του topic.

Συνοψίζοντας, η γενική λειτουργία αυτής της ροής που υλοποιεί τη λογική όλων των πλήκτρων, αρχίζει από τη διέγερση του κόμβου Button από τον οποίο ξεκινάει το μήνυμα το οποίο περνάει και μετατρέπεται σε κατάλληλη μορφή στον κόμβο function και καταλήγει τελικά στον κόμβο MQTT για να δημοσιευτεί και να γίνει διαθέσιμο στο συγκεκριμένο topic για τους ενδιαφερόμενους clients.



Σχήμα 4.16: Απεικονίζονται οι ροές που υλοποιήθηκαν για να εμφανίζονται τα πλήκτρα της διεπαφής.

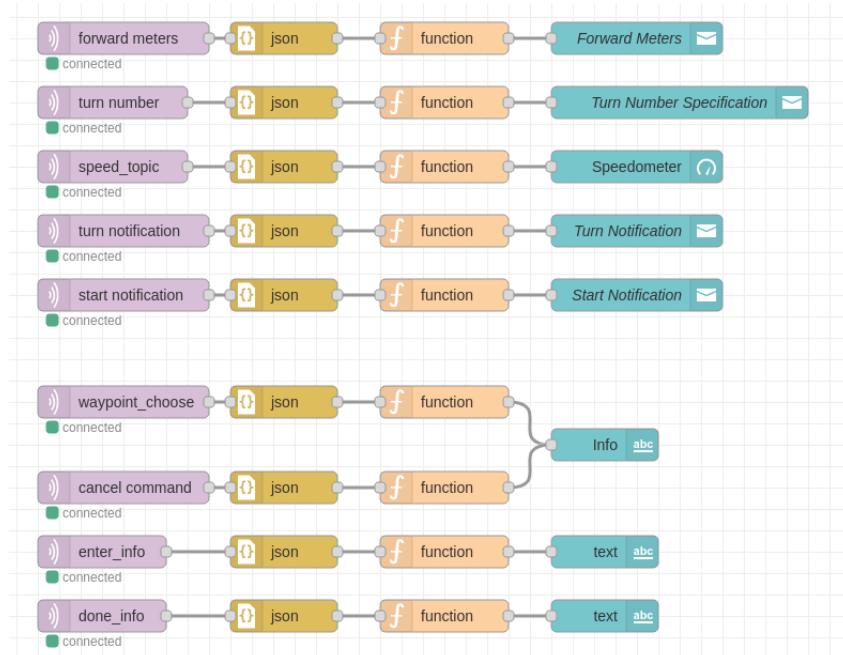
Υλοποίηση των ειδοποιήσεων και των πληροφοριών

Στη συγκεκριμένη υποενότητα αναλύονται οι ροές των πληροφοριών και των ειδοποιήσεων καθώς παρουσιάζουν παρόμοιο τρόπο υλοποίησης μεταξύ τους. Οι ροές αυτές φαίνονται στο [σχήμα 4.17](#) και θα ακολουθηθεί και εδώ ένας παρόμοιος τρόπος ανάλυσης με την προηγούμενη παράγραφο περιγράφοντας ενδεικτικά την πρώτη ροή του σχήματος και επισημαίνοντας τις διαφορές με τις υπόλοιπες ροές. Η πρώτη ροή περιλαμβάνει τρεις κόμβους και υλοποιεί την ειδοποίηση που προτρέπει το χρήστη να ορίσει τα μέτρα που θέλει να κινηθεί μπροστά το όχημα του μετά το πάτημα του πλήκτρου FORWARD:

- **MQTT (Subscriber):** Τα δεδομένα που λαμβάνει από το ορισμένο topic έχουν δημοσιευτεί από κάποιον client της Python που έχει στείλει μήνυμα στο topic του Mosquitto broker. Το όνομα του κόμβου υποδηλώνει σε και αυτή την περίπτωση, το topic από το οποίο λαμβάνονται τα δεδομένα. Η διαφορά με τους υπόλοιπους MQTT (Subscriber) κόμβους των άλλων ειδοποιήσεων είναι το όνομα του topic.
- **JSON:** Το μήνυμα που λαμβάνει ο MQTT Subscriber είναι σε αλφαριθμητική JSON μορφή και το μεταβιβάζει στον κόμβο JSON για να το μετατρέψει σε JavaScript αντικείμενο.
- **Function:** Σε αυτή την περίπτωση ο κόμβος Function δέχεται ως είσοδο το JSON αντικείμενο του προηγούμενου κόμβου και η μετατροπή που κάνει στο μήνυμα είναι να διατηρήσει μόνο την τιμή της "data.value" ιδιότητας στο "msg". Το μήνυμα δηλαδή που περνάει στον επόμενο κόμβο έχει πλέον μετατραπεί από ολόκληρο JSON αντικείμενο σε μια απλή αριθμητική, αλφαριθμητική ή λογική τιμή. Αναφέρονται και οι τρεις περιπτώσεις γιατί διαφέρουν οι τιμές που λαμβάνονται ανάλογα με την υλοποίηση της ροής.
- **Notification:** Ο κόμβος Notification χρησιμοποιείται για να εμφανίζει ειδοποίηση στο χρήστη στο πάνω δεξιά μέρος της οθόνης. Η ειδοποίηση είναι ένα pop-up παράθυρο που εμφανίζεται για κάποια δευτερόλεπτα και εμφανίζει το μήνυμα που δέχεται ως είσοδο από τον προηγούμενο function κόμβο. Στην περίπτωση της ειδοποίησης το μήνυμα είναι μια αλφαριθμητική τιμή με περιεχόμενο την ειδοποίηση προς το χρήστη.
- **Gauge:** Στη συγκεκριμένη περίπτωση ο μετρητής λειτουργεί σαν ταχόμετρο και εμφανίζει την ταχύτητα του οχήματος την εκάστοτε χρονική στιγμή. Να σημειωθεί εδώ ότι ο μετρητής μπαίνει στη θέση του κόμβου Notification όπως φαίνεται και στο [σχήμα 4.17](#). Η υπόλοιπη ροή για τη λειτουργία του ταχυμέτρου παραμένει ακριβώς ίδια με τη ροή που υλοποιεί τις ειδοποιήσεις.
- **Text:** Ο κόμβος Text χρησιμοποιείται στις οδηγίες χρήσης της διεπαφής που αναφέρθηκαν προηγουμένως και στο feedback που δίνεται στο χρήστη καθώς επιλέγει τα σημεία από τα οποία θέλει να περάσει το όχημα του. Σημειώνεται και εδώ, ότι η υλοποίηση της υπόλοιπης ροής των κόμβων text είναι ακριβώς ίδια με αυτή των ειδοποιήσεων.

4.2. ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΔΙΕΠΑΦΗΣ ΤΟΥ NODERED

Συνολικά, οι ροές που αναλύθηκαν σε αυτή την κατηγορία έχουν ίδιο τρόπο υλοποίησης είτε πρόκειται για ειδοποιήσεις, text inputs ή μετρητές. Με λίγα λόγια, σε όλες τις περιπτώσεις αυτής της κατηγορίας η ροή ξεκινάει από τον MQTT Subscriber κόμβο ο οποίος δέχεται τα δεδομένα από το αντίστοιχο topic, τα μεταβιβάζει σε έναν JSON κόμβο για να μετατραπούν σε JavaScript αντικείμενο ο οποίος με τη σειρά του τα μεταφέρει σε function κόμβο ο οποίος κρατά από το αντικείμενο μόνο το κύριο μέρος του μηνύματος. Από εκεί τα επεξεργασμένα δεδομένα μεταφέρονται στον τελικό κόμβο (notification, gauge, text input) ο οποίος είναι και ο κόμβος που εμφανίζεται στο interface και ανάλογα τη λειτουργία του εμφανίζει τα δεδομένα στην οθόνη με κατάλληλο τρόπο.



Σχήμα 4.17: Απεικονίζονται οι ροές που υλοποιήθηκαν για να εμφανίζονται οι ειδοποιήσεις και οι καρτέλες πληροφοριών της διεπαφής.

Υλοποίηση των sliders

Στην παρακάτω παράγραφο θα γίνει ανάλυση του τρόπου με τον οποίο λειτουργούν οι ροές οι οποίες παράγουν sliders που περιέχει η διεπαφή. Οι ροές για τις οποίες γίνεται λόγος φαίνονται στο [σχήμα 4.18](#). Ενδεικτικά και πάλι θα αναλυθεί η πρώτη ροή η οποία περιλαμβάνει την υλοποίηση του slider "Aggressive" με το οποίο ο χρήστης ρυθμίζει το ποσοστό επιθετικότητας που θέλει να έχει το όχημα του στο δρόμο. Η συγκεκριμένη ροή περιλαμβάνει έξι κόμβους οι οποίοι αναλύονται παρακάτω:

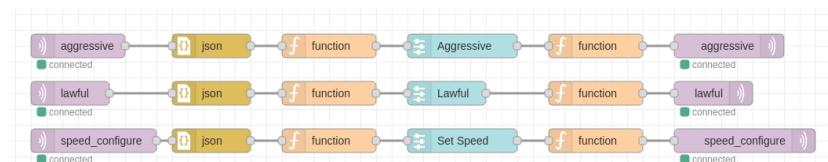
- **MQTT (Subscribe):** Ο κόμβος MQTT δέχεται ως είσοδο τα δεδομένα από έναν client που έχει δημοσιευθεί στο αντίστοιχο topic. Στη συγκεκριμένη ομάδα ροών ο MQTT Subscriber δε χρησιμοποιείται για να πάρει δεδομένα τα οποία θα αλλάξουν την τιμή του slider διότι η τιμή αυτή καθορίζεται από τον χρήστη.

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

Η τιμή που δέχεται αυτός ο Subscriber είναι πάντα μηδενική. Ο σκοπός του είναι να μηδενίζει τις τιμές των sliders όταν ξεκινά η λειτουργία του interface και να μη μένουν παλιές τιμές αποθηκευμένες στα sliders.

- **JSON:** Ο κόμβος JSON χρησιμοποιείται για τη μετατροπή του μηνύματος που μεταβιβάζει ο Subscriber σε JavaScript αντικείμενο.
- **Function:** Ο κόμβος function έχει τη γενική λειτουργικότητα που αναφέρθηκε παραπάνω και στη συγκεκριμένη περίπτωση χρησιμοποιείται για να κρατήσει στο μήνυμα μόνο το μέρος "data.aggressive", στην περίπτωση του aggressive slider, του συνολικού μηνύματος το οποίο και μεταβιβάζει στον επόμενο κόμβο.
- **Slider:** Ο συγκεκριμένος κόμβος slider δέχεται ως είσοδο το μήνυμα από τον function κόμβο το οποίο όπως αναφέρθηκε είναι μηδενικό μήνυμα για να καθαρίσουν τα sliders από παλιές τιμές κατά την εκκίνηση. Τα sliders aggressive και lawful παίρνουν τιμές στο διάστημα (0, 10) και με βήμα τη μονάδα που σημαίνει ότι παίρνουν όλες τις ακέραιες τιμές σε αυτό το διάστημα. Το slider "Set Speed" με το οποίο θέτει ο χρήστης την ταχύτητα του οχήματος δέχεται ακέραιες τιμές στο διάστημα (0, 100). Η τιμή που θέτει ο χρήστης στον slider κόμβο μεταβιβάζεται στον επόμενο function κόμβο.
- **MQTT (Publish):** Ο MQTT Publisher κόμβος δέχεται ως είσοδο την τιμή που έχει θέσει ο χρήστης στο slider. Μεσολαβεί βέβαια μεταξύ του MQTT και του Slider ένας function κόμβος ο οποίος μετατρέπει την έξοδο του slider σε JSON μορφή διότι όπως αναφέρθηκε και παραπάνω τα μηνύματα θα πρέπει να δημοσιεύονται στα topics σε κατάλληλη μορφή JSON. Ο MQTT Publisher δέχεται το JSON μήνυμα και το δημοσιεύει στο κατάλληλο topic από το οποίο οι clients της Python θα μπορούν να γνωρίζουν τις τιμές των sliders.

Με λίγα λόγια, το αριστερό κομμάτι της συγκεκριμένης ομάδας ροών, δηλαδή οι τρεις κόμβοι μέχρι τον κόμβο του slider, χρησιμοποιείται για να καθαρίσει με μηδενική τιμή τον κόμβο slider σε περίπτωση που έχουν διατηρηθεί παλιές τιμές. Έπειτα ο κόμβος slider υλοποιεί το slider που εμφανίζεται στην οθόνη και δίνει τη δυνατότητα στο χρήστη να θέτει τις τιμές που επιθυμεί και η ροή της πληροφορίας καταλήγει σε έναν MQTT Publisher ο οποίος κάνει γνωστό στο σύστημα τις επιλογές τιμών του χρήστη.



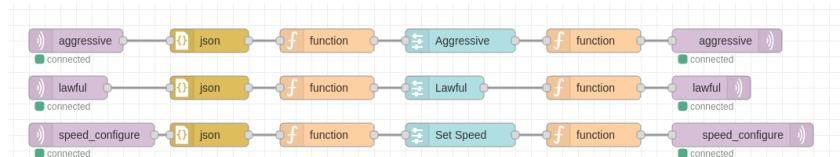
Σχήμα 4.18: Απεικονίζονται οι ροές που υλοποιήθηκαν για να εμφανίζονται τα sliders της διεπαφής.

4.3. ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΛΕΙΤΟΥΡΓΙΑΣ ΤΟΥ MOSQUITTO BROKER

Υλοποίηση των αναπτυσσόμενων μενού (dropdown boxes) και των text inputs

Η περιγραφή των υλοποιήσεων των ροών ολοκληρώνεται σε αυτή την παράγραφο με την περιγραφή των ροών που εμφανίζουν τα αναπτυσσόμενα μενού και το text input στη διεπαφή. Αυτή η κατηγορία ροών περιλαμβάνει έξι κόμβους η καθεμία και φαίνεται στο [σχήμα 4.19](#). Στη διεπαφή υπάρχουν συνολικά δύο αναπτυσσόμενα μενού. Το πρώτο είναι αυτό που χρησιμοποιείται για την επιλογή του τρόπου κατεύθυνσης (επιλογές Location ή Direction) και το δεύτερο είναι αυτό από το οποίο μπορεί ο χρήστης να επιλέξει τοποθεσία πάνω στο χάρτη. Το text input είναι το στοιχείο εκείνο στο οποίο προσδιορίζονται τα μέτρα που επιθυμεί ο χρήστης να κινηθεί μπροστά σε περίπτωση που πατήσει προηγουμένως το πλήκτρο FORWARD.

Η λειτουργία της ροής αυτής της κατηγορίας είναι αρκετά όμοια με την υλοποίηση των ροών των sliders οπότε θα επισημανθούν οι ομοιότητες και οι διαφορές τους. Η υλοποίηση της ροής ενός αναπτυσσόμενου μενού ξεκινάει και εδώ με μία τριάδα κόμβων (MQTT, JSON, Function) που χρησιμοποιούνται για να καθαρίσουν την όποια τιμή έχει μείνει στο αναπτυσσόμενο μενού από προηγούμενη λειτουργία. Η διαφορά με την προηγούμενη υλοποίηση είναι ότι η έξοδος που φτάνει τελικά στην είσοδο του dropdown box μετά από την επεξεργασία που έχει δεχθεί είναι ένα κενό αλφαριθμητικό και όχι μία μηδενική τιμή όπως προηγουμένως. Έπειτα, υπάρχει ο κύριος κόμβος της ροής που είναι το αναπτυσσόμενο μενού ή το text input. Στα αναπτυσσόμενα μενού χρειάζεται να οριστούν μόνο οι επιλογές που θα δίνονται στο χρήστη για να επιλέξει. Στο text input χρειάζεται να οριστεί μόνο ο τύπος της εισόδου που θα δίνεται στην μπάρα σαν είσοδος. Τελικά, το μήνυμα καταλήγει, όπως και στην περίπτωση των sliders, αφού πρώτα μετατραπεί σε JSON μορφή από έναν κόμβο Function, σε έναν MQTT Publisher ο οποίος το κάνει διαθέσιμο στους clients της Python για περαιτέρω λειτουργίες.



Σχήμα 4.19: Απεικονίζονται οι ροές που υλοποιήθηκαν για να εμφανίζονται τα αναπτυσσόμενα μενού και το text input της διεπαφής.

4.3 ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΛΕΙΤΟΥΡΓΙΑΣ ΤΟΥ MosQUITTO BROKER

Η περιγραφή της αρχιτεκτονικής του συστήματος ξεκίνησε από την ανάλυση της διεπαφής που χτίστηκε στο NodeRED διότι είναι το πρώτο πράγμα με το οποίο έρχεται σε επαφή ο χρήστης καθώς θέτει σε λειτουργία το σύστημα. Η σχεδίαση της αρχιτεκτονικής συνεχίζεται με τον broker Mosquitto που χρησιμοποιήθηκε για τη διανομή των μηνυμάτων. Το συγκεκριμένο κομμάτι της αρχιτεκτονικής χρησιμοποιήθηκε σε λογική black box ⁵⁹ όπως φαίνεται στο [σχήμα 4.20](#). Δηλαδή αποτελεί ένα

⁵⁹ https://en.wikipedia.org/wiki/Black_box

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

σύστημα στο οποίο στέλνονται κάποιες είσοδοι και παράγονται κάποιες έξοδοι χωρίς να γνωρίζει το υπόλοιπο σύστημα την εσωτερική λειτουργία του. Οι clients που έχουν υλοποιηθεί σε Python δέχονται από τα κατάλληλα topics αυτές τις εξόδους του broker και ενεργούν κατάλληλα στο όχημα.



Σχήμα 4.20: Η λογική Black Box.

Πηγή: https://en.wikipedia.org/wiki/Black_box

4.4 ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΥΛΟΠΟΙΗΣΕΩΝ ΤΟΥ ΑΥΤΟΝΟΜΟΥ ΟΧΗΜΑΤΟΣ

Το σημαντικότερο ίσως υποσύστημα της αρχιτεκτονικής που ακολουθήθηκε και περιγράφηκε στο **υποκεφάλαιο 4.1** είναι το τμήμα αυτό το οποίο υλοποιεί όλους τους αλγορίθμους που χρησιμοποιούνται για να κινηθεί σωστά το αυτόνομο όχημα. Το μέρος αυτό της αρχιτεκτονικής τρέχει στον προσομοιωτή CARLA και οι επιμέρους αλγόριθμοι υλοποιήθηκαν σε Python. Στην παρούσα ενότητα θα παρουσιαστεί η αρχιτεκτονική των υποσυστημάτων που αφορούν την κίνηση και αλληλεπίδραση του οχήματος με το περιβάλλον και θα γίνει η ανάλυση του καθενός από αυτά. Στην ουσία αφορά το μέρος εκείνο του συνολικού συστήματος το οποίο δεν είναι ορατό στο χρήστη αλλά τρέχει στο πίσω μέρος του συστήματος και φροντίζει για την ορθή λειτουργία του.

4.4.1 Περιγραφή αρχιτεκτονικής του αυτόνομου οχήματος

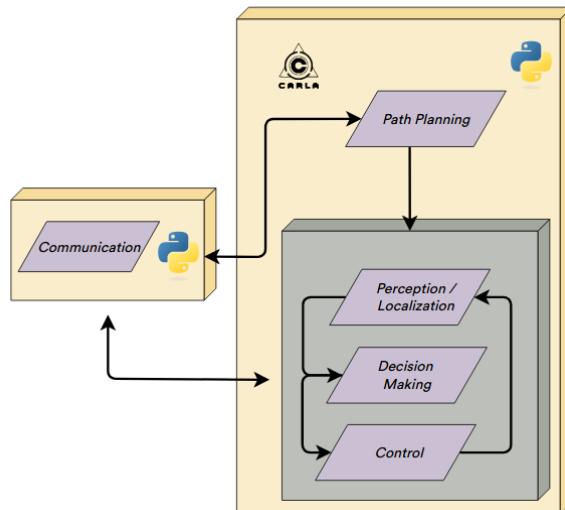
Στην παρούσα διπλωματική, για την κατασκευή των διάφορων υποσυστημάτων του αυτόνομου οχήματος, χρησιμοποιήθηκε μία αρθρωτή (modular) αρχιτεκτονική παρόμοια με τις [13], [14], [15], [16], [17], [18], [19] ως προς τα βασικά υποσυστήματα που διαθέτει προσθέτοντας κάποια επιπλέον τα οποία βοήθησαν στη σωστή σχεδίαση της συγκεκριμένης αρχιτεκτονικής. Ο λόγος που επιλέχθηκε αυτή η αρχιτεκτονική ήταν η ξεκάθαρη διάκριση στην υλοποίηση που προσφέρει ανάμεσα στα υποσυστήματα της. Αυτό σημαίνει ότι το κάθε υποσύστημα μπορεί να υλοποιηθεί ανεξάρτητα από το άλλο και να επικοινωνούν μεταξύ τους μόνο για να ανταλλάσουν τα απαραίτητα δεδομένα. Το συνολικό πρόβλημα ακολουθώντας αυτή τη λογική μετατρέπεται σε ένα σαφώς καθορισμένο σύνολο από επιμέρους προβλήματα τα οποία λύνοντας τα και συνδέοντας τις λύσεις τους με κατάλληλους τρόπους οδηγεί το συνολικό σύστημα σε ορθή λειτουργία. Τα επιμέρους υποσυστήματα που συνθέτουν την αρχιτεκτονική του αυτόνομου οχήματος στην παρούσα διπλωματική είναι:

- **Υποσύστημα Επικοινωνίας:** Το υποσύστημα αυτό είναι υπεύθυνο για τη μεταβίβαση και τη διαχείριση των ερεθισμάτων που στέλνει στο όχημα ο χρήστης μέσω της διεπαφής του NodeRED.

4.4. ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΥΛΟΠΟΙΗΣΕΩΝ ΤΟΥ ΑΥΤΟΝΟΜΟΥ ΟΧΗΜΑΤΟΣ

- **Υποσύστημα Σχεδίασης Τροχιάς:** Το υποσύστημα αυτό είναι υπεύθυνο για τη σωστή σχεδίαση της τροχιάς πάνω στον χάρτη που πρόκειται να κινηθεί, την οποία επιθυμεί να ακολουθήσει ο χρήστης με το όχημα του.
- **Υποσύστημα Αντίληψης:** Το συγκεκριμένο υποσύστημα αποτελεί τα "μάτια" του αυτόνομου οχήματος εντοπίζοντας πιθανά δυναμικά και στατικά εμπόδια στο περιβάλλον μέσα στο οποίο κινείται αλλά και τους ελεύθερους χώρους στους οποίους ου επιτρέπεται η κίνηση.
- **Υποσύστημα Επιλογής Συμπεριφοράς:** Το υποσύστημα αυτό χρησιμοποιείται για την επιλογή της καταλληλότερης συμπεριφοράς κάθε χρονική στιγμής ανάλογα με τις συνθήκες που επικρατούν εκείνη τη στιγμή.
- **Υποσύστημα Ελέγχου:** Το υποσύστημα ελέγχου υλοποιεί στην ουσία έναν ελεγκτή ο οποίος είναι και υπεύθυνος για την προγραμματική κίνηση και τη σωστή ακολούθηση της τροχιάς. Παράγει ένα σύνολο από τιμές οι οποίες δίνουν εντολή στους τροχούς του οχήματος για την ταχύτητα και την κατεύθυνση που θα ακολουθήσουν.

Στο [σχήμα 4.21](#) φαίνεται το μέρος της συνολικής αρχιτεκτονικής που αφορά το αυτόνομο όχημα και περιλαμβάνει τα υποσυστήματα που αναφέρθηκαν παραπάνω. Έπειτα ακολουθεί μία ανάλυση του τρόπου με τον οποίο αναπτύχθηκε το κάθε υποσύστημα, η επικοινωνία μεταξύ τους και οι αλγόριθμοι που υλοποιήθηκαν εσωτερικά των υποσυστημάτων.



Σχήμα 4.21: Το μέρος της συνολικής αρχιτεκτονικής που αφορά τη λειτουργία του αυτόνομου οχήματος.

4.4.2 Υποσύστημα Επικοινωνίας

Το υποσύστημα επικοινωνίας είναι ένα από τα σημαντικότερα υποσυστήματα στη δομή της αρχιτεκτονικής του αυτόνομου οχήματος διότι αποτελεί ουσιαστικά τη

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

”γέφυρα” μεταξύ των υπόλοιπων υποσυστημάτων του οχήματος και της διεπαφής του χρήστη. Όπως αναφέρθηκε και προηγουμένως, οι εντολές που στέλνει ο χρήστης από τη διεπαφή μεταφέρονται στον Mosquitto broker και από εκεί με το MQTT πρωτόκολλο στο αυτόνομο όχημα. Το υποσύστημα επικοινωνίας είναι το τμήμα του αυτόνομου οχήματος που θα λάβει τα μηνύματα από τον broker και θα τα μοιράσει στα υπόλοιπα υποσυστήματα του αυτοκινήτου ανάλογα και με τη φύση της εντολής που εστάλη.

Τα κύρια υποσυστήματα με τα οποία αλληλεπιδρά είναι το υποσύστημα Επιλογής Συμπεριφοράς και το υποσύστημα Ελέγχου. Στο πρώτο επιδρά κυρίως με τις τιμές των aggressive και lawful που δίνονται από το χρήστη και επηρεάζουν τις αποφάσεις που παίρνει εντός της κυκλοφορίας. Στο δεύτερο επιδρά κυρίως με εντολές εκκίνησης, σταματήματος, αλλαγής λωρίδας στις οποίες ο χρήστης θέλει να επέμβει απευθείας στον έλεγχο του αυτοκινήτου χωρίς να του δώσει το περιθώριο να αποφασίσει. Το συγκεκριμένο υποσύστημα χρησιμοποιείται και για την αντίθετη πορεία των μηνυμάτων δηλαδή όταν κάποιο από τα άλλα υποσυστήματα του οχήματος θέλει να στείλει κάποιο μήνυμα στον broker και μετέπειτα στον χρήστη μέσω του NodeRED τότε το πραγματοποιεί μέσω του υποσυστήματος επικοινωνίας. Τα μηνύματα που ακολουθούν αυτή την πορεία είναι κυρίως ενημερωτικού τύπου σε περίπτωση που το σύστημα του αυτοκινήτου θέλει να προειδοποιήσει ή απλά να ενημερώσει το χρήστη για τις συνθήκες λειτουργίας του αυτοκινήτου και την πορεία του μέσα στην κυκλοφορία. Με τα δύο συστήματα που αναφέρθηκαν, το υποσύστημα επικοινωνίας αλληλεπιδρά σε πραγματικό χρόνο, δηλαδή όσο το όχημα βρίσκεται σε πορεία ανταλλάσσοντας συνεχώς ερεθίσματα είτε με κατεύθυνση προς το σύστημα είτε προς το χρήστη. Εκτός των δύο αυτών υποσυστημάτων βέβαια, το υποσύστημα επικοινωνίας ανταλλάσσει μηνύματα με το υποσύστημα Σχεδίασης τροχιάς. Η διαφορά με αυτό το υποσύστημα βρίσκεται στο γεγονός ότι τα μηνύματα είναι οι επιλογές προορισμού και κατεύθυνσης που δίνονται από το χρήστη και δίνονται μια φορά στη διαδικασία σχεδιασμού της τροχιάς σε αντίθεση με την επικοινωνία που γίνεται σε πραγματικό χρόνο με τα υπόλοιπα υποσυστήματα όπως αναφέρθηκε προηγουμένως.

Τεχνικά μιλώντας το υποσύστημα αυτό υλοποιεί τους clients που είναι subscribers και publishers υλοποιημένοι σε Python. Οι clients αυτοί μέσω της σύνδεσης με τα καταλληλα topics είτε δημοσιεύουν (publish) στον Mosquitto broker τα μηνύματα που προορίζονται για το χρήστη είτε καταγράφουν (subscribe) τις εντολές που δέχεται ο broker από το χρήστη και προορίζονται προς το όχημα.

Περιγραφή Αλγορίθμου

Παρακάτω εμφανίζονται οι δύο κλάσεις οι οποίες υλοποιήθηκαν και λειτουργούν σαν Publishers και Subscribers με σκοπό την εύρυθμη επικοινωνία με τον broker. Η υλοποίηση των δύο κλάσεων και οι κλάσεις και τα αντικείμενα που χρησιμοποιούνται (για παράδειγμα η κλάση ConnectionParameters) βασίζεται στη λειτουργία της commlib-*py* που αναφέρθηκε στο [υποκεφάλαιο 3.5](#).

Η κλάση PublisherMQTT είναι αυτή η οποία υλοποιεί τις μεθόδους των Publishers και διαθέτει δύο μεθόδους. Η πρώτη μέθοδος καλείται ”initialize” και αρχικοποιεί όλες τις απαραίτητες μεταβλητές για τη δημιουργία επικοινωνίας. Για την επί-

4.4. ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΥΛΟΠΟΙΗΣΕΩΝ ΤΟΥ ΑΥΤΟΝΟΜΟΥ ΟΧΗΜΑΤΟΣ

τευξή της σύζευξης με τον broker είναι απαραίτητο να είναι γνωστό το hostname του broker και το port στο οποίο τρέχει για αυτό και δηλώνονται ως προϋπόθεση όπως φαίνεται στον αλγόριθμο. Με τη δεύτερη μέθοδο η οποία καλείται "publish" το μήνυμα "msg" που δίνεται ως όρισμα δημοσιεύεται στο topic που έχει οριστεί προηγουμένως. Στη συγκεκριμένη περίπτωση οι μεταβλητές host και port λαμβάνουν τις τιμές "test.mosquitto.org" και 1883 αντίστοιχα.

Η κλάση SubscriberMQTT υλοποιεί τις μεθόδους των Subscribers και διαθέτει τρεις κύριες μεθόδους. Η πρώτη μέθοδος ονομάζεται και εδώ "initialize" και αρχικοποιεί τις μεταβλητές που είναι απαραίτητες για τη σύνδεση. Και σε αυτή την περίπτωση προαπαιτείται η γνώση του hostname και του port με τα οποία θα συνδεθεί και τα οποία έχουν τις τιμές που αναφέρθηκαν στην προηγούμενη παράγραφο. Εσωτερικά, η μέθοδος "initialize" καλεί τη μέθοδο "subscribe" η οποία αρχικοποιεί τον Subscriber που θα λειτουργήσει και τον θέτει σε αδιάκοπη σύνδεση μέσω της συνάρτησης run(). Η πιο σημαντική μέθοδος της Subscriber κλάσης είναι η "dataCallback" η οποία καλείται αυτόματα κάθε φορά που θα εντοπιστεί καινούργιο δεδομένο στο αντίστοιχο topic. Αυτός είναι και ο λόγος που ο Subscriber συνδέεται αδιάκοπα με τον broker. Εντός της dataCallback ελέγχεται η ορθότητα του topic σύνδεσης και σε περίπτωση που υπάρχουν δεδομένα σε αυτό τότε αποθηκεύονται στη "value" μεταβλητή. Επίσης, η μεταβλητή που καλείται changedValue χρησιμοποιείται εσωτερικά για να γνωρίζουν τα υπόλοιπα υποσυστήματα ότι το topic έχει δεχθεί καινούργιο δεδομένο. Τα ονόματα των μεταβλητών που αναφέρονται εδώ είναι εικονικά διότι, σημειώνεται ότι για κάθε ξεχωριστό topic υπάρχει διαφορετική κλάση Subscriber η οποία αρχικοποιεί τις μεταβλητές με πιο αντιπροσωπευτικά ονόματα και πιο εξειδικευμένες λειτουργίες. Η κεντρική ιδέα του Subscriber πάντως είναι αυτή που περιγράφηκε.

Αλγόριθμος 4.1 Class PublisherMQTT

Require: host, port

```
1: function INITIALIZE(topic)
2:   self.topic = topic
3:   connectionParams = ConnectionParameters(host, port)
4:   self.publisher = Publisher(topic, connectionParams)
5: end function
6: function PUBLISH(msg)
7:   publisher.publish(msg)
8: end function
```

4.4.3 Υποσύστημα Καθορισμού και Σχεδίασης Τροχιάς

Το υποσύστημα καθορισμού και σχεδίασης τροχιάς είναι από τα πρώτα υποσυστήματα τα οποία ενεργοποιούνται κατά τη λειτουργία του συστήματος. Το συγκεκριμένο υποσύστημα είναι υπεύθυνο για την πλήρη χάραξη της τροχιάς δημιουργώντας τα διαδοχικά waypoints τα οποία θα ακολουθεί ένα-ένα το όχημα για να φτάσει στον τελικό προορισμό του. Είναι το υποσύστημα εκείνο το οποίο καλείται πριν το όχημα ξεκινήσει να κινείται δηλαδή στη φάση που ο χρήστης καθορίζει τα σημεία

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

Αλγόριθμος 4.2 Class SubscriberMQTT

Require: host, port

```
1: function INITIALIZE(topic)
2:     self.topic = topic
3:     connectionParams = ConnectionParameters(host, port)
4:     self.value = 0
5:     self.changedValue = False
6:     subscribe()
7: end function
8: function SUBSCRIBE
9:     sub = Subscriber(topic, dataCallback, connectionParams)
10:    sub.run()
11: end function
12: function DATA CALLBACK(msg)
13:     if self.topic == correctTopic then
14:         self.value = msg['topic']
15:         self.changedValue = True
16:     else
17:         return
18:     end if
19: end function
```

και τις κατευθύνσεις που θέλει να ακολουθήσει πάνω στο χάρτη. Όπως αναλύθηκε και στη [ενότητα 4.2.1](#) οι ενέργειες του χρήστη χωρίζονται σε αυτές που πράττει πριν την εκκίνηση του οχήματος και σε αυτές που πραγματοποιεί όσο το όχημα κινείται. Το συγκεκριμένο υποσύστημα ενεργοποιείται κατά την πρώτη φάση ενεργειών και στην ουσία αποτελεί το κεντρικό αλγορίθμικό τμήμα που διαχειρίζεται τις επιλογές του χρήστη ως προς τις τοποθεσίες και κατευθύνσεις που θέλει να ακολουθήσει. Το υποσύστημα καθορισμού και σχεδίασης της τροχιάς συνδέεται με το υποσύστημα επικοινωνίας, το οποίο αποτελεί τη γέφυρα μεταξύ της διεπαφής και του συστήματος του αυτοκινήτου. Το υποσύστημα επικοινωνίας μεταβιβάζει στο υποσύστημα σχεδίασμού της τροχιάς τις προθέσεις του χρήστη για τους προορισμούς και την αντίθετη πορεία ακολουθούν μηνύματα προειδοποίησης σε περίπτωση που κάποια κίνηση δεν μπορεί να πραγματοποιηθεί ή μηνύματα επιβεβαίωσης σχετικά με την καταγραφή των προορισμών.

Κάθε τοποθεσία ή κατεύθυνση που δίνεται από τον οδηγό μεταφράζεται σε ένα συγκεκριμένο waypoint πάνω στο χάρτη. Ετσι σχηματίζεται μία ακολουθία από σημεία πάνω στο χάρτη από τα οποία θα περάσει το όχημα. Προκειμένου η μετάβαση σε αυτά τα σημεία να γίνει με όσο το δυνατόν πιο ομαλό και ασφαλή τρόπο τα σημεία αυτά ενώνονται με κοντινότερα waypoints που απέχουν συνήθως μέχρι ένα μέτρο μεταξύ τους και κάνουν τη συνολική τροχιά πιο ομαλή. Δημιουργείται έτσι ένας πίνακας όπως αυτός που φαίνεται στη [εξίσωση 4.1](#) με όλα τα waypoints από τα οποία θα πρέπει να περάσει το όχημα για να ολοκληρώσει τη διαδρομή του. Ο **αλγόριθμος 4.4** υλοποιεί την πρώτη διαδικασία κατά την οποία ο χρήστης δίνει τις επιλογές του στο σύστημα σχετικά με τις διαδρομές και τις τοποθεσίες που

4.4. ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΥΛΟΠΟΙΗΣΕΩΝ ΤΟΥ ΑΥΤΟΝΟΜΟΥ ΟΧΗΜΑΤΟΣ



Σχήμα 4.22: Τα waypoints w_X που περιέχει το διάνυσμα 4.1 για μια ευθεία διαδρομή 30 μέτρων.

Θέλει να επισκεφτεί. Ο αλγόριθμος παίρνει τις επιλογές του χρήστη τις μεταφράζει σε συντεταγμένες πάνω στο χάρτη και επιστρέφει έναν πίνακα όπως αυτόν της εξίσωσης 4.1 περιλαμβάνοντας όλα τα σημεία από τα οποία πρέπει να περάσει το όχημα για να ολοκληρώσει την πορεία του και να φτάσει στον προορισμό του. Στο σχήμα 4.22 απεικονίζεται ένα παράδειγμα με τα waypoints w_X που περιέχει το διάνυσμα 4.1 για μια ευθεία διαδρομή 30 μέτρων και πρέπει να ακολουθήσει το όχημα ένα προς ένα για να φτάσει στο τέλος της ευθείας.

$$waypoints = [waypoint^{(1)}, \ waypoint^{(2)}, \dots, \ waypoint^{(i)}, \dots, \ waypoint^{(n)}] \quad (4.1)$$

Περιγραφή αλγορίθμου επιλογής τροχιάς

Η μόνη προϋπόθεση για να λειτουργήσει ο αλγόριθμος 4.4 είναι να δοθεί ως είσοδος η αρχική τοποθεσία του οχήματος. Έπειτα ξεκινάει μια άπειρη δομή επανάληψης η οποία τερματίζεται όταν ο χρήστης πατήσει το κουμπί DONE το οποίο, όπως αναλύθηκε και προηγουμένως στην ενότητα που περιγράφηκε η διεπαφή υποδηλώνει ότι η διαδικασία καταγραφής των σημείων που θέλει να επισκεφτεί ο χρήστης έχει ολοκληρωθεί και το όχημα ξεκινάει τη λειτουργία του. Στην ουσία στον αλγόριθμο αυτό είναι η συνθήκη τερματισμού της άπειρης επανάληψης που θα επιτρέψει στο όχημα να πάει στο επόμενο στάδιο. Εσωτερικά, αυτής της επανάληψης ο αλγόριθμος υλοποιεί τη λογική επιλογής τοποθεσίας ή κατεύθυνσης και διαθέτει μια εσωτερική άπειρη δομή επανάληψης η οποία τρέχει όσο ο χρήστης δίνει επιλογές κατεύθυνσης στο σύστημα. Η λειτουργία της επανάληψης εσωτερικά ξεκινάει στη γραμμή 3 όπου ελέγχεται αν έχει πατηθεί το κουμπί CANCEL με το οποίο ακυρώνονται όλες οι επιλογές του χρήστη, βγαίνει από την επανάληψη και η διαδικασία επιλογής ξεκινά από την αρχή. Έπειτα στη μεταβλητή "action" εκχωρείται η επιλογή του χρήστη σχετικά με τον τρόπο που επιθυμεί να κινηθεί και ο αλγόριθμος ακολουθεί δύο διαδρομές.

Στην περίπτωση που δοθεί η επιλογή "Location" ο αλγόριθμος καλεί τη συνάρτηση "handle_locations" η οποία καταγράφει όλες τις τοποθεσίες από τις οποίες επιθυμεί να περάσει το όχημα και τις αποθηκεύει σε έναν πίνακα για να υπάρχει αποθηκευμένη η σειρά με την οποία θα τις επισκεφτεί και τις επιστρέφει στον πίνακα "end_waypoints". Επίσης, τυπώνει και ανάλογα μηνύματα ενημέρωσης προς το χρήστη σχετικά με τις καταγεγραμμένες τοποθεσίες. Στη συνέχεια, μέσω της

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

συνάρτησης "trace_route" υπολογίζονται η ελάχιστες διαδρομές μεταξύ των διαδοχικών προορισμών που έχει επιλέξει ο χρήστης και επιστρέφονται στον πίνακα "custom_waypoints" όλα τα σημεία της τροχιάς που έχει σχηματιστεί. Ανανεώνεται επιπλέον το "current_waypoint" το οποίο διαθέτει το τελευταίο σημείο της συνολικής τροχιάς για να γνωρίζει ο αλγόριθμος από που να ξεκινήσει σε περίπτωση που ο χρήστης επιλέξει επιπλέον προορισμούς και τέλος στον πίνακα "waypoints" αποθηκεύονται όλα τα σημεία της τροχιάς από τα οποία πρέπει να περάσει το όχημα είτε προέρχονται από την επιλογή "Location" είτε από την επιλογή "Direction".

Στην περίπτωση που δοθεί η επιλογή "Direction" το σύστημα εμφανίζει τις διαθέσιμες επιλογές κατεύθυνσης που μπορεί να κινηθεί μέσω της συνάρτησης "turn_info". Αφού το σύστημα περιμένει μέχρι να δοθεί επιλογή από το χρήστη τότε στις περιπτώσεις που δοθεί μία εκ των "RIGHT", "LEFT" και "STRAIGHT" τότε ο έλεγχος του προγράμματος μεταφέρεται στη συνάρτηση "handle_turn" η οποία διαχειρίζεται τις επιλογές κατεύθυνσης και επιστρέφει τα waypoints που ενώνουν την εκάστοτε τοποθεσία με την κατάληξη της στροφής που έχει επιλεγεί. Η περίπτωση της επιλογής "FORWARD" διαχειρίζεται από ξεχωριστή συνάρτηση η οποία επιστρέφει επίσης τα ανάλογα waypoints. Όπως και στην περίπτωση του "Location" ανανεώνονται οι μεταβλητές current_waypoint και waypoints αντίστοιχα και η επανάληψη τερματίζεται για να δοθεί εκ νέου επιλογή από το χρήστη.

Εφόσον η διαδικασία αυτή ολοκληρωθεί και ο χρήστης δώσει όλες τις επιλογές τοποθεσίας ή κατεύθυνσης που επιθυμεί να ακολουθήσει η εξωτερική άπειρη επανάληψη τερματίζεται και τυπώνονται στη διεπαφή ανάλογα μηνύματα που σηματοδοτούν την έναρξη της πορείας του οχήματος. Ο έλεγχος του προγράμματος μεταφέρεται πλέον σε καινούργια συνάρτηση η οποία μετακινεί το όχημα και καθορίζει την επιλογή της συμπεριφοράς του και θα αναλυθεί σε επόμενη ενότητα. Μόλις το όχημα ολοκληρώσει τη διαδρομή του ο αλγόριθμος τυπώνει στην οθόνη της διεπαφής μήνυμα ενημερώσεις σχετικά με την ολοκλήρωση της διαδρομής και ο αλγόριθμος 4.4 ξεκινάει από την αρχή τη λειτουργία του σε περίπτωση που ο χρήστης επιθυμεί να μετακινηθεί σε άλλο προορισμό.

Περιγραφή συναρτήσεων

Στη συγκεκριμένη υποενότητα θα περιγραφούν οι σημαντικότερες συναρτήσεις και οι αλγόριθμοι που χρησιμοποιήθηκαν στον αλγόριθμο 4.4 για την πλήρη λειτουργία του:

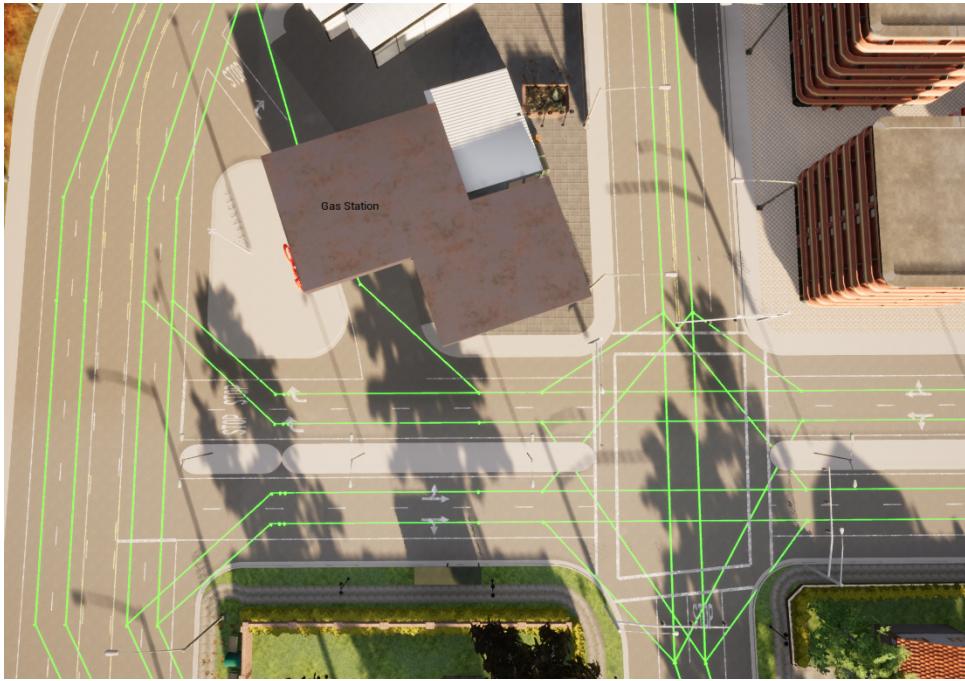
- **Εύρεση Ελάχιστου Μονοπατιού:** Στο μέρος εκείνο του αλγορίθμου όπου ο χρήστης επιλέγει έναν ή περισσότερους τελικούς προορισμούς θα πρέπει να βρεθεί η βέλτιστη διαδρομή που συνδέει διαδοχικά αυτούς τους προορισμούς. Για τον λόγο αυτό χρησιμοποιήθηκε το γεγονός ότι κάθε χάρτης του CARLA διαθέτει σε κάθε λωρίδα κυκλοφορίας έναν κόμβο στην αρχή και έναν στο τέλος της λωρίδας. Οι κόμβοι αυτοί είναι στην ουσία waypoints που υποδηλώνουν τα όρια της λωρίδας. Δημιουργείται με αυτόν τον τρόπο σε ολόκληρο το χάρτη ένας κατευθυνόμενος γράφος με κόμβους τα waypoints της κάθε λωρίδας. Ο γράφος απεικονίζεται στο σχήμα 4.23 όπου φαίνονται με βέλη οι κατευθύνσεις των ακμών οι οποίες συμβαδίζουν με την κατεύθυνση του δρόμου. Να σημειωθεί εδώ ότι σαν μονοπάτι δε χρησιμοποιούνται οι ακμές

4.4. ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΥΛΟΠΟΙΗΣΕΩΝ ΤΟΥ ΑΥΤΟΝΟΜΟΥ ΟΧΗΜΑΤΟΣ

του γράφου για τη μετακίνηση του οχήματος αλλά ένα σύνολο από κοντινότερα σημεία τα οποία σχηματίζουν καμπυλωτές διαδρομές και οδηγούν με ασφάλεια το όχημα από κόμβο σε κόμβο. Επομένως, για να βρεθεί η βέλτιστη διαδρομή χρησιμοποιήθηκε ο αλγόριθμος A^* ο οποίος αποτελεί έναν από τους πιο αποδοτικούς αλγορίθμους εύρεσης ελάχιστου μονοπατιού σε γράφο. Ο A^{*60} αποτελεί έναν αλγόριθμο ο οποίος λειτουργεί σε γράφο, με θετικά βάροντας και προσπαθεί σε κάθε βήμα του να ελαχιστοποιήσει ένα συγκεκριμένο κόστος μεταξύ του κόμβου που βρίσκεται και του τελικού κόμβου. Σε κάθε βήμα του ελέγχει τους επόμενους κόμβους στους οποίους μπορεί να κινηθεί δημιουργώντας έτσι συνεχώς έναν δέντρο με τους δυνατούς κόμβους και επιλέγει εκείνον που ικανοποιεί μια συγκεκριμένη συνθήκη. Το κόστος που προσπαθεί να ελαχιστοποιήσει σε κάθε επανάληψη δίνεται από τη συνάρτηση $f(n) = g(n) + h(n)$ όπου η μεταβλητή n αποτελεί τον κόμβο που εξετάζεται, η συνάρτηση g αποτελεί το κόστος του κόμβου, δηλαδή το άθροισμα των βαρών των ακμών, από την αρχή μέχρι τον κόμβο n και η συνάρτηση h η οποία ονομάζεται "heuristic" και είναι αυτή που υπολογίζει το κόστος από τον κόμβο n μέχρι τον τελικό κόμβο. Στη συγκεκριμένη περίπτωση οι ακμές του γράφου είναι οι λωρίδες κυκλοφορίας. Οπότε, ως συνάρτηση g που χρησιμοποιήθηκε είναι το μήκος της κάθε λωρίδας μέχρι τον κόμβο n και η συνάρτηση h που χρησιμοποιήθηκε είναι η απόσταση του κόμβου n από τον τελικό προορισμό. Η λογική του αλγορίθμου A^* υλοποιείται μέσω της συνάρτησης "trace_route" που καλείται στον [αλγόριθμο 4.4](#) και παρέχεται έτοιμη από το CARLA API. Κατά τη διαδικασία εύρεσης του επόμενου βέλτιστου κόμβου λαμβάνονται υπόψη και οι κανόνες κυκλοφορίας και η τοπολογία του χάρτη στον οποίο τρέχει ο αλγόριθμος. Η διαδικασία εύρεσης ελάχιστου μονοπατιού τερματίζει όταν βρεθεί ο τελικός κόμβος. Ο αλγόριθμος επιστρέφει έναν πίνακα από waypoints τα οποία συνδέουν τον αρχικό και τον τελικό προορισμό με βέλτιστο τρόπο. Στο [σχήμα 4.24](#) απεικονίζονται κάποια παραδείγματα βέλτιστων διαδρομών.

- **Έλεγχος κατεύθυνσεων:** Δίνοντας την επιλογή "Direction" ο χρήστης, επιλέγει την κατεύθυνση που θέλει να ακολουθήσει στην επόμενη διασταύρωση. Όπως φαίνεται και στον [αλγόριθμο 4.4](#) η συνάρτηση "handle_turn" είναι αυτή που χειρίζεται τις εντολές κατεύθυνσης που δίνει ο οδηγός. Όποια εντολή κατεύθυνσης και να δοθεί το σύστημα καλεί την ενσωματωμένη συνάρτηση του CARLA (next_until_lane_end(1.0)) η οποία γεμίζει με waypoints που απέχουν 1 μέτρο την αρχή και το τέλος μιας λωρίδας ή μιας διασταύρωσης για να προστεθούν στα συνολικά waypoints της τροχιάς. Το κύριο πρόβλημα που διαχειρίζεται η συνάρτηση "handle_turn" είναι να μπορέσει να αντιληφθεί την κατεύθυνση ως προς το όχημα των τελικών waypoints. Το CARLA API δε διαθέτει κάποια πληροφορία σχετικά με τη σχετική θέση των waypoints δηλαδή πόσο δεξιά ή αριστερά είναι κάποιο waypoint σε σχέση με κάποιο άλλο οπότε προκύπτει μέσω μαθηματικών υπολογισμών. Πιο συγκεκριμένα, οι συντεταγμένες όλων των σημείων εντός του χάρτη δίνονται ως προς κάποιο κεντρικό σύστημα συντεταγμένων που βρίσκεται στο κέντρο του χάρτη. Για να μπορέ-

⁶⁰https://en.wikipedia.org/wiki/A*_search_algorithm



Σχήμα 4.23: Κατευθυνόμενος γράφος που παρέχεται από τον CARLA προσομοιωτή.

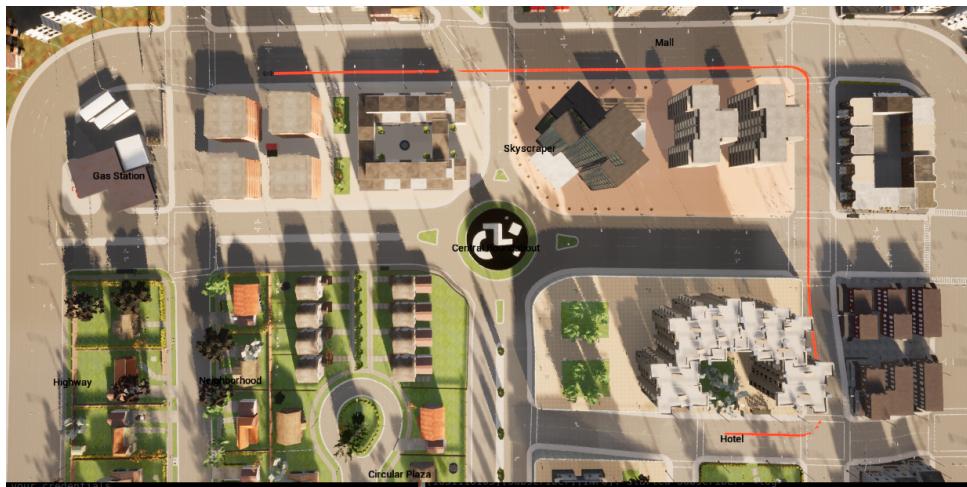
σει ο αλγόριθμος να αντιληφθεί τη σχετική θέση δύο σημείων μετατρέπει το ένα σημείο θεωρώντας ως αρχή τις συντεταγμένες του άλλου σημείου. Το σύστημα συντεταγμένων του CARLA είναι αριστερόστροφο με πάνω τον άξονα Z για αυτό και χρησιμοποιείται ο πίνακας που φαίνεται στη [εξίσωση 4.2](#) για τη μετατροπή των συντεταγμένων. Η γωνία θ είναι η στροφή του αρχικού σημείου ως προς τον άξονα Z (γωνία yaw). Οι υπολογισμοί για τη μετατροπή των συντεταγμένων ενός σημείου B ως προς ένα σημείο A υλοποιούνται όπως στη [εξίσωση 4.3](#). Σημειώνεται ότι ο δείκτης κάτω δεξιά δηλώνει το σημείο στο οποίο ανήκει η συντεταγμένη και ο πάνω δεξιά είναι το σύστημα ως προς το οποίο είναι εκφρασμένη με το W να δηλώνει το αρχικό κεντρικό σύστημα συντεταγμένων. Αφού γίνει η μετατροπή τότε συγχρίνονται τα πρόσημα των x, y συντεταγμένων. Όταν οι x, y συντεταγμένες είναι ομόσημες τότε το σημείο B βρίσκεται στα δεξιά του σημείου A και όταν είναι ετερόσημες το σημείο B βρίσκεται στα αριστερά του σημείου A.

$$R = \begin{bmatrix} \cos(\theta), & \sin(\theta), & 0 \\ -\sin(\theta), & \cos(\theta), & 0 \\ 0, & 0, & 1 \end{bmatrix} \quad (4.2)$$

$$\begin{bmatrix} x_B^{(A)} \\ y_B^{(A)} \\ z_B^{(A)} \end{bmatrix} = R \cdot \left(\begin{bmatrix} x_B^{(W)} \\ y_B^{(W)} \\ z_B^{(W)} \end{bmatrix} - \begin{bmatrix} x_A^{(W)} \\ y_A^{(W)} \\ z_A^{(W)} \end{bmatrix} \right) \quad (4.3)$$

Για την περίπτωση που το σημείο B βρίσκεται περίπου στην ίδια ευθεία με το σημείο A τότε δεν πραγματοποιείται αλλαγή συντεταγμένων αλλά ακολουθείται μια διαφορετική τεχνική. Αυτή η περίπτωση καλύπτει την εντολή

4.4. ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΥΛΟΠΟΙΗΣΕΩΝ ΤΟΥ ΑΥΤΟΝΟΜΟΥ ΟΧΗΜΑΤΟΣ



(α') Ελάχιστο μονοπάτι με αρχική θέση την Καφετέρια και τελική το Ξενοδοχείο.



(β') Ελάχιστο μονοπάτι με αρχική θέση την Καφετέρια και τελική τη Γειτονιά.

Σχήμα 4.24: Βέλτιστες διαδρομές υπολογισμένες μέσω του A*.

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

STRAIGHT. Λαμβάνεται το μοναδιαίο διάνυσμα του σημείου B και του σημείου A και υπολογίζεται η γωνία μεταξύ των δύο μέσω της αντίστροφης εφαπτομένης του εσωτερικού τους γινομένου. Μετά από υπολογισμούς προκύπτει ότι στρογγυλοποιημένα αυτή η γωνία είναι 0.8 μοίρες. Η λογική που περιγράφηκε παραπάνω και για τις τρεις επιλογές που μπορεί να δοθούν από το χρήστη αποτυπώνεται στον **αλγόριθμο 4.3**. Να σημειωθεί εδώ ότι στην περίπτωση που έχει επιλεγεί η εντολή RIGHT ή LEFT για μια διασταύρωση και υπάρχουν παραπάνω από μια δεξιές ή αριστερές στροφές τότε όπως αναφέρθηκε και στην ενότητα της διεπαφής ζητείται να δοθεί και ο αριθμός της στροφής. Με αυτό τον τρόπο συμπληρώνονται και τα αντίστοιχα waypoints που πρέπει να προστεθούν στον συνολικό πίνακα.

Συνοψίζοντας, η συνάρτηση "handle_turn" είναι αυτή που αποφασίζει ποια waypoints θα προστεθούν στα συνολικά waypoints της τροχιάς μετά την εντολή που θα δώσει ο χρήστης. Όταν οι εντολές του χρήστη είναι LEFT ή RIGHT χρησιμοποιείται η τεχνική της πρώτης παραγράφου και όταν η εντολή είναι STRAIGHT χρησιμοποιείται η τεχνική της δεύτερης.

- **Επιλογή FORWARD:** Σε περίπτωση που ο χρήστης δώσει την επιλογή FORWARD με X μέτρα προώθησης τότε ο έλεγχος του **αλγορίθμου 4.4** μεταφέρεται στη συνάρτηση "handle_forward" η οποία είναι εξαιρετικά απλή στην υλοποίηση και γεμίζει τον συνολικό πίνακα των waypoints με σημεία τα οποία απέχουν μεταξύ τους 1 μέτρο και βρίσκονται στην ίδια λωρίδα με το όχημα. Η διαδικασία ολοκληρώνεται όταν συμπληρώσει X σημεία στον πίνακα και σε περίπτωση που η λωρίδα του οχήματος δε συνεχίζει στην τοπολογία του χάρτη τότε αυτόματα προστίθενται στον πίνακα τυχαία δεξιά ή αριστερά τα επόμενα σημεία μέχρι να συμπληρωθούν τα X μέτρα.

Συνοψίζοντας, στη συγκεκριμένη ενότητα παρουσιάστηκε ο τρόπος με τον οποίο οι εντολές κίνησης και οι επιλογές προορισμού που δίνονται από το χρήστη επεξεργάζονται με κατάλληλες τεχνικές από το σύστημα και μετατρέπονται σε ένα σύνολο από σημεία πάνω στο χάρτη δημιουργώντας έτσι μια ομαλή τροχιά που συνδέει το αρχικό σημείο που βρίσκεται το όχημα με τον τελικό προορισμό περνώντας από όλους τους ενδιάμεσους προορισμούς που δόθηκαν και υπακούοντας όλες τις πιθανές εντολές του οδηγού στις διασταύρωσεις που συναντά. Όλη αυτή η διαδικασία υλοποιείται από τον **αλγόριθμο 4.4** μαζί με τις επιμέρους υλοποιήσεις των συναρτήσεων που περιγράφηκαν.

4.4.4 Υποσύστημα Αντίληψης

Το πρόβλημα της Αντίληψης

Στο πρόβλημα της αυτόνομης οδήγησης ένα από τα σημαντικότερα βήματα που πρέπει να γίνει για να μπορεί το όχημα να κινείται με ασφάλεια εντός του δρόμου είναι η γνώση της περιοχής γύρω από το όχημα. Για την ακρίβεια, δεν αρκεί το όχημα να λαμβάνει δεδομένα από το περιβάλλον όπως για παράδειγμα εικόνα από κάμερα ή μετρήσεις από κάποιο LiDAR αλλά θα πρέπει να παράγει γνώση

Αλγόριθμος 4.3 Handle Turn

Require: start_waypoint, turn_choice

```

1: waypoints ← [start_waypoint]
2: paths ← paths in the junction from the same, right and left lane if exist
3: for each available path do
4:   temp_waypoints ← path.next_until_lane_end(1.0)
5:   angle ← calculate_angle(temp_waypoints[0], temp_waypoints[end])
6:   converted_coor ← change_coordinate_system(temp_waypoints[end])
7:   if angle = 0.8 ∧ turn_choice == "STRAIGHT" then
8:     Print in Node RED the message 'Going STRAIGHT at the next junction'
9:     overall_waypoints ← append temp_waypoints
10:    break the loop
11:   else if converted_coor.x * converted_coor.y > 0 ∧ turn_choice = "RIGHT" then
12:     if more than one RIGHT turns exist then
13:       choice_of_turn ← the number of the turn
14:       temp_waypoints ← path[choice_of_turn].next_until_lane_end(1.0)
15:     end if
16:     Print in Node RED the message 'Turn RIGHT at the next junction'
17:     overall_waypoints ← append temp_waypoints
18:     break the loop
19:   else if converted_coor.x * converted_coor.y < 0 ∧ turn_choice = "LEFT" then
20:     if more than one LEFT turns exist then
21:       choice_of_turn ← the number of the turn
22:       temp_waypoints ← path[choice_of_turn].next_until_lane_end(1.0)
23:     end if
24:     Print in Node RED the message 'Turn LEFT at the next junction'
25:     overall_waypoints ← append temp_waypoints
26:     break the loop
27:   end if
28: end for
29: return overall_waypoints

```

Αλγόριθμος 4.4 Specify Destination

Require: Start Point

```

1: while True do
2:   while True do
3:     if CANCEL button has been pushed then
4:       break the internal loop
5:     end if
6:     action  $\leftarrow$  User's choice for "Location" or "Direction"
7:     if action == "Location" then
8:       end_waypoints  $\leftarrow$  handle_locations(current_waypoint)
9:       custom_waypoints  $\leftarrow$  trace_route(end_waypoints)
10:      current_waypoint  $\leftarrow$  last waypoint of custom_waypoints array
11:      waypoints  $\leftarrow$  add to waypoints the custom_waypoints
12:      break the internal loop
13:    else if action == "Direction" then
14:      turn_info(current_waypoint)           # Show the options for direction
15:      Wait until an option will be given
16:      turn  $\leftarrow$  user's choice for direction
17:      if turn == "RIGHT" then
18:        custom_waypoints  $\leftarrow$  handle_turn(current_waypoint, "RIGHT")
19:      else if turn == "LEFT" then
20:        custom_waypoints  $\leftarrow$  handle_turn(current_waypoint, "LEFT")
21:      else if turn == "STRAIGHT" then
22:        custom_waypoints  $\leftarrow$  handle_turn(current_waypoint, "STRAIGHT")
23:      else if turn == "FORWARD" then
24:        custom_waypoints  $\leftarrow$  handle_forward(current_waypoint)
25:      end if
26:      waypoints  $\leftarrow$  add to waypoints the custom_waypoints
27:      current_waypoint  $\leftarrow$  last waypoint of custom_waypoints array
28:      break the internal loop
29:    end if
30:  end while
31:  if DONE button has been pushed then
32:    Trajectory has been selected so break the external loop
33:  end if
34: end while
35: Print in Node RED the message 'Waypoint selection has completed! Press START to begin!'
36: Wait until START button has been pushed
37: Print in Node RED the message 'Your car is on! Choose velocity to begin!'
38: Call the function that turns the car on to follow the specified trajectory
39: if vehicle has reached its destination then
40:   Print in Node RED the message 'You have reached your destination! Define a new route to continue!'
41: end if
```

4.4. ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΓΛΟΠΟΙΗΣΕΩΝ ΤΟΥ ΑΥΤΟΝΟΜΟΥ ΟΧΗΜΑΤΟΣ

από αυτά τα δεδομένα δηλαδή μέσα από κατάλληλες πράξεις να αντιλαμβάνεται τι πραγματικά συμβαίνει στον γύρω χώρο του. Ένα καλό σύστημα αυτόνομης οδήγησης περιλαμβάνει αντίληψη χώρου 360 μοίρες γύρω από το όχημα η οποία επιτυγχάνεται μέσα από κατάλληλους αισθητήρες που είναι εγκατεστημένοι στο όχημα. Κάθε αισθητήρας, ανάλογα με τα χαρακτηριστικά του όπως το εύρος λειτουργίας του, χρησιμοποιείται και για διαφορετικό σκοπό. Οι αισθητήρες δέχονται τα διάφορα ερεθίσματα και ένα κεντρικό σύστημα αντίληψης τα χρησιμοποιεί για να συνθέσει μία συνολική εικόνα για το εξωτερικό περιβάλλον του οχήματος. Η διαδικασία αυτή ονομάζεται sensor fusion. Σε αυτό το υποσύστημα υλοποιούνται διαδικασίες όπως εντοπισμός αντικειμένων (object detection) βασισμένο σε εικόνα το οποίο αναφέρεται στον καθορισμό της τοποθεσίας και του μεγέθους των διάφορων στατικών και δυναμικών αντικειμένων όπως άλλα οχήματα, πεζοί, σημάνσεις, φανάρια κλπ. Πραγματοποιεί έτσι μια κατάταξη στα διάφορα αντικείμενα του περιβάλλοντος. Εκτός από εντοπισμό (detection) γίνεται και παρακολούθηση (tracking) της συμπεριφοράς των αντικειμένων πραγματοποιώντας μια πρόβλεψη για τη μελλοντική τους κίνηση αποφεύγοντας έτσι τις συγκρούσεις. Επίσης, εδώ πραγματοποιείται και η αναγνώριση λωρίδας (lane detection) που καθορίζει ουσιαστικά την περιοχή του δρόμου πάνω στην οποία μπορεί να κινηθεί το όχημα με ασφάλεια. Το υποσύστημα αντίληψης είναι ένα από τα βασικότερα υποσυστήματα γενικότερα σε ένα αυτόνομο όχημα καθώς έχει τη μεγαλύτερη ευθύνη για την πλοήγηση του αυτοκινήτου μέσα στο χώρο και την αποφυγή των στατικών και δυναμικών εμποδίων. Είναι στην ουσία το σύστημα που αντικαθιστά τα μάτια του οδηγού στη συνολική διαδικασία της οδήγησης.

Γλοποίηση της Αντίληψης

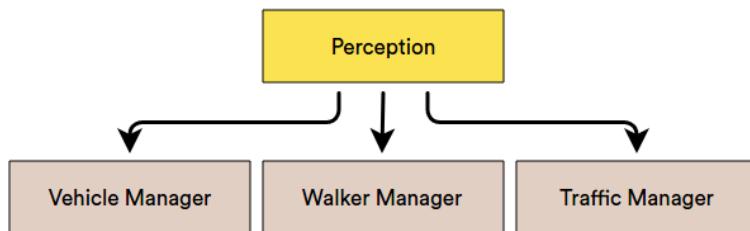
Παρά τη μεγάλη ανάπτυξη και την ακρίβεια που προσφέρουν οι αισθητήρες που έχουν αναπτυχθεί για τη λύση του προβλήματος αντίληψης του οχήματος αλλά και τους προηγμένους αλγορίθμους που προσφέρουν σωστή ανάλυση όλων των δεδομένων εισόδου από το εξωτερικό περιβάλλον, στη συγκεκριμένη διπλωματική εργασία χρησιμοποιήθηκε μια πιο απλουστευμένη λύση στο πρόβλημα αντίληψης του οχήματος στην οποία βασικό ρόλο έπαιξε και η χρήση του CARLA ως προσομοιωτή του συνολικού συστήματος.

Συγκεκριμένα, ο προσομοιωτής του CARLA διαθέτει τρισδιάστατους χάρτες μέσα στους οποίους πλοιηγείται το όχημα. Οι χάρτες αυτοί περιλαμβάνουν τόσο τα τρισδιάστατα αντικείμενα στο χώρο όσο και τον σαφή καθορισμό των δρόμων και των κανόνων του. Το οδικό δίκτυο ενός χάρτη βασίζει τη λειτουργία του σε ένα OpenDRIVE αρχείο όπως αναφέρθηκε και στο [υποκεφάλαιο 3.1](#). Το αρχείο αυτό περιλαμβάνει όλη τη χρήσιμη πληροφορία που χρειάζεται ένα όχημα για να αντιληφθεί το εξωτερικό περιβάλλον του, αλλά και τον τρόπο με τον οποίο λειτουργεί το οδικό δίκτυο δηλαδή οι κατευθύνσεις και τα όρια των λωρίδων κυκλοφορίας, οι κανόνες αλλά και οι διάφορες σημάνσεις. Η πληροφορία αυτή λαμβάνεται από το όχημα και μετατρέπεται σε χρήσιμη γνώση όπως για παράδειγμα ο υπολογισμός των αποστάσεων και των κατευθύνσεων των δυναμικών εμποδίων. Επίσης, η πληροφορία που λαμβάνεται από τον προσομοιωτή είναι περιορισμένη ως προς το εύρος της έτσι ώστε να προσομοιώνει με τον ρεαλιστικότερο δυνατό τρόπο τα

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

δεδομένα που λαμβάνει ένα πραγματικό όχημα με αληθινούς αισθητήρες. Το υποσύστημα της αντίληψης λαμβάνει υπόψη του όλα τα στατικά και δυναμικά εμπόδια που βρίσκονται εντός του δρόμου κυκλοφορίας. Τα στατικά αντικείμενα που βρίσκονται εκτός του δρόμου κυκλοφορίας όπως κτίρια, παγκάκια κλπ δε λαμβάνονται υπόψη παρά μόνο τα δυναμικά εμπόδια και οι πιθανές σημάνσεις. Επίσης, λαμβάνει πληροφορίες σχετικά με την κατεύθυνση των λωρίδων κυκλοφορίας. Η υλοποίηση του υποσυστήματος αντίληψης που έγινε στη συγκεκριμένη διπλωματική χωρίστηκε σε τρία επιμέρους υποσυστήματα με το καθένα να αναλαμβάνει και διαφορετικό καθήκον ως προς τη συνολική αντίληψη του εξωτερικού περιβάλλοντος. Όπως φαίνεται και στο [σχήμα 4.25](#) τα επιμέρους υποσυστήματα χωρίζονται ως προς το είδος των εμποδίων που εντοπίζουν είναι τα:

- Υποσύστημα εντοπισμού οχημάτων
- Υποσύστημα αναγνώρισης σημάνσεων και φωτεινών σηματοδοτών
- Υποσύστημα εντοπισμού πεζών



Σχήμα 4.25: Το σύστημα αντίληψης του αυτοκινήτου.

Υποσύστημα εντοπισμού οχημάτων

Το συγκεκριμένο υποσύστημα είναι υπεύθυνο για την ανίχνευση δυναμικών εμποδίων και συγκεκριμένα ασχολείται με τον εντοπισμό των οχημάτων που βρίσκονται σε μια ακτίνα 50 μέτρων από το αυτόνομο όχημα. Η λογική με την οποία εντοπίζονται τα κοντινότερα οχήματα βασίζεται στη χρήση των λωρίδων κυκλοφορίας στις οποίες κινούνται τα οχήματα τη στιγμή που καλείται το υποσύστημα εντοπισμού οχημάτων για να τα βρει. Κάθε όχημα κινείται πάνω σε μια γραμμή κυκλοφορίας η οποία διαθέτει ένα μοναδικό αναγνωριστικό το οποίο παρέχει το API του CARLA και που βοηθάει στον ακριβή προσδιορισμό της θέσης των οχημάτων.

Ο [αλγόριθμος 4.5](#) χρησιμοποιείται για τον εντοπισμό όλων των εμπρόσθιων και οπίσθιων οχημάτων που ανήκουν στην ίδια λωρίδα και κατ' επέκταση και των κοντινότερων από αυτά. Στις γραμμές 1 και 2 υπολογίζεται το εμπρόσθιο και οπίσθιο σημείο του οχήματος προσθέτοντας και αφαιρώντας αντίστοιχα, την έκταση (extent) του οχήματος ως προς τον άξονα x. Τα σημεία αυτά φαίνονται καλύτερα και στο [σχήμα 4.26](#). Έπειτα, ακολουθεί μία δομή επανάληψης η οποία προσπελαύνει όλα τα οχήματα του χάρτη και εντοπίζει αυτά που ανήκουν στην ίδια λωρίδα και έχουν

4.4. ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΥΛΟΠΟΙΗΣΕΩΝ ΤΟΥ ΑΥΤΟΝΟΜΟΥ ΟΧΗΜΑΤΟΣ

σχεδόν ίδιο προσανατολισμό με το αυτόνομο όχημα, τοποθετώντας τα στη λίστα των οχημάτων με την ίδια λωρίδα κατεύθυνσης όπως φαίνεται στην [εξίσωση 4.4](#).

$$vehicles_in_lane = [vehicle^{(1)}, \ vehicle^{(2)}, \ \dots, \ vehicle^{(i)}, \ \dots, \ vehicle^{(n)}] \quad (4.4)$$

Από τα οχήματα που ανήκουν στην ίδια λωρίδα κατεύθυνσης με το αυτόνομο όχημα συγκρίνεται η απόσταση τους από το εμπρόσθιο και το οπίσθιο σημείο του αυτόνομου οχηματος όπως φαίνεται και στο [σχήμα 4.26](#) και συγκρίνοντας αυτές τις δύο τιμές τοποθετούνται σε λίστες με τα εμπρόσθια ή οπίσθια οχήματα αντίστοιχα. Εκτός από τα οχήματα, σε αντίστοιχες λίστες τοποθετούνται και οι υπολογισμένες αποστάσεις όπως αυτές της [εξίσωσης 4.5](#).

$$distances_from_front_side = \begin{bmatrix} front_side_distance_from_vehicle^{(1)}, \\ front_side_distance_from_vehicle^{(2)}, \\ \vdots \\ front_side_distance_from_vehicle^{(i)}, \\ \vdots \\ front_side_distance_from_vehicle^{(n)} \end{bmatrix} \quad (4.5)$$

Φτάνοντας σε αυτό το σημείο ο αλγόριθμος γνωρίζει όλα τα οχήματα εντός της προβλεπόμενης ακτίνας που βρίσκονται μπροστά και πίσω του. Από τις δύο αυτές λίστες υπολογίζονται οι ελάχιστες τιμές δηλαδή οι μικρότερες αποστάσεις και με αυτό τον τρόπο το σύστημα πλέον γνωρίζει το κοντινότερο μπροστινό και οπίσθιο όχημα του καθώς και τις αποστάσεις από αυτά όπως υπολογίζεται από την [εξίσωση 4.6](#) για την περίπτωση των μπροστινών οχημάτων.

$$\min_distance_index = \min_j (distances_from_front_side) \quad (4.6)$$

$$closest_front_vehicle = front_vehicles_in_lane[min_distance_index]$$

Σχετικά με τα πλαϊνά οχήματα χρησιμοποιείται η ίδια ακριβώς με τεχνική στον αλγόριθμο ε μία μικρή παραλλαγή. Για αυτό τον λόγο δε θα αναλυθεί εκτενώς ο αλγόριθμος εύρεσης των κοντινότερων πλαϊνών οχημάτων αλλά θα αναφερθεί μόνο η διαφορά με την προηγούμενη προσέγγιση. Η διαφορά έγκειται στο γεγονός ότι για τον υπολογισμό για παράδειγμα των κοντινότερων οχημάτων που βρίσκονται στη δεξιά λωρίδα χρησιμοποιείται το σημείο "R" που φαίνεται στο [σχήμα 4.27](#). Το σημείο "R" στην ουσία είναι ο καθρεπτισμός του σημείου του οχηματος εκείνη τη χρονική στιγμή στη δεξιά λωρίδα δημιουργώντας έτσι ένα εικονικό όχημα για τη διευκόλυνση των υπολογισμών. Έπειτα, εντοπίζονται τα οχήματα πάνω στο χάρτη που έχουν ίδιο αναγνωριστικό λωρίδας με το σημείο "R" οπότε και ανήκουν στη δεξιά λωρίδα του οχηματος. Οι συντεταγμένες και το αναγνωριστικό του σημείου αυτού πάνω στο χάρτη παρέχονται έτοιμα από το CARLA API. Αντίστοιχα, υπολογίζονται, κάνοντας χρήση του σημείου "L", τα οχήματα και οι αποστάσεις από την αριστερή λωρίδα. Σημειώνεται εδώ, ότι οι αποστάσεις από τα πλαϊνά οχήματα υπολογίζονται από τη θέση του οχηματος και όχι από τα σημεία "R" και "L" δεξιά και αριστερά. Με τον ίδιο ακριβώς τρόπο δημιουργούνται οι λίστες με τα κοντινότερα δεξιά και αριστερά οχήματα και τις αντίστοιχες αποστάσεις τους όπως στις [εξισώσεις 4.4](#) και [4.5](#). Προκύπτει επομένως, ότι το όχημα διαθέτει πλήρη γνώση για τα κοντινότερα οχήματα που το περιτριγυρίζουν και στις τρεις λωρίδες κυκλοφορίας.

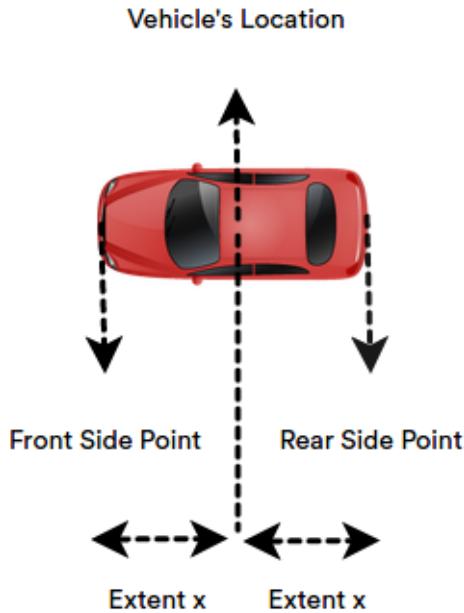
Αλγόριθμος 4.5 Find Front and Rear Vehicles

Input: List of vehicles in map

Output: Closest front and rear vehicles and distances

```

1: vehicle_front_side ← ego_vehicle_location+[ego_vehicle_bounding_box_extent_x, 0, 0]
2: vehicle_rear_side ← ego_vehicle_location-[ego_vehicle_bounding_box_extent_x, 0, 0]
3: for vehicle in vehicle_list do
4:   ego_waypoint ← The waypoint of ego_vehicle's current location in the map
5:   other_waypoint ← The waypoint of examined vehicle's current location in the map
6:   yaw_difference ← ego_vehicle_yaw - vehicle_yaw
7:   if ego_waypoint_lane_id = other_waypoint_lane_id  $\wedge$  yaw_difference < 4 then
8:     add examined vehicle in vehicles_in_lane_list
9:   end if
10:  end for
11:  for vehicle in vehicles_in_lane_list do
12:    vehicle_location ← examined vehicle's current location
13:    ego_distance_front ← distance between examined vehicle and ego_vehicle's front side
14:    ego_distance_rear ← distance between examined vehicle and ego_vehicle's rear side
15:    ego_distance ← distance between examined vehicle and ego_vehicle's location
16:    if ego_distance_front < ego_distance_rear then
17:      add examined vehicle in vehicles_in_front_list
18:      add ego_distance in distances_in_front_list
19:    else
20:      add examined vehicle in vehicles_in_rear_list
21:      add ego_distance in distances_in_rear_list
22:    end if
23:  end for
24:  front_min_index ← index in list distances_in_front_list with the minimum value
25:  if distances_in_front_list[front_min_index] < 50 then
26:    closest_front_vehicle ← vehicles_in_front_list[front_min_index]
27:    closest_distance_from_front_vehicle ← distances_in_front_list[front_min_index]
28:  else
29:    closest_front_vehicle ← None
30:    closest_distance_from_front_vehicle ← Inf
31:  end if
32:  if distances_in_rear_list[rear_min_index] < 50 then
33:    closest_rear_vehicle ← vehicles_in_rear_list[rear_min_index]
34:    closest_distance_from_rear_vehicle ← distances_in_rear_list[rear_min_index]
35:  else
36:    closest_rear_vehicle ← None
37:    closest_distance_from_rear_vehicle ← Inf
38:  end if
```



Σχήμα 4.26: Τα σημεία ενδιαφέροντος επάνω στο αυτοκίνητο.

Υποσύστημα αναγνώρισης σημάνσεων και φωτεινών σηματοδοτών

Το δεύτερο τμήμα του υποσυστήματος αντίληψης αναλαμβάνει το καθήκον να εντοπίσει και να αναγνωρίσει τους φωτεινούς σηματοδότες και τις διάφορες σημάνσεις που υπάρχουν στους δρόμους κυκλοφορίας. Ο [αλγόριθμος 4.6](#) υλοποιεί την αναγνώριση των φωτεινών σηματοδοτών λαμβάνοντας ως είσοδο το αντικείμενο του αυτοκινήτου, για να μπορεί να καλέσει τις κατάλληλες συναρτήσεις που διαθέτει ο CARLA, και επιστρέφει το χρώμα του φωτεινού σηματοδότη. Η πληροφορία για τους φωτεινούς σηματοδότες δίνεται απευθείας από τον προσομοιωτή όταν το όχημα πλησιάσει κοντά στον φωτεινό σηματοδότη. Έπειτα, ο [αλγόριθμος 4.7](#) υλοποιεί την αναγνώριση των διαφόρων σημάνσεων. Οι σημάνσεις που έχουν ενδιαφέρον είναι το STOP και τα όρια ταχύτητας για αυτό και εντοπίζονται μόνο αυτές. Γενικά, η λογική είναι όμοια με αυτή του προηγούμενου αλγορίθμου και συγκεκριμένα καλείται ο προσομοιωτής που διαθέτει έτοιμη συνάρτηση η οποία επιστρέφει τα διάφορα ορόσημα (landmarks) πάνω στο δρόμο εντός μιας ακτίνας που εδώ έχει οριστεί στα 5 μέτρα. Από όλα τα ορόσημα που εντοπίζονται, διατηρούνται μόνο αυτά με αναγνωριστικά τους αριθμούς 206 και 274 που αντιστοιχούν στις σημάνσεις ενδιαφέροντος. Έπειτα, ελέγχεται ο προσανατολισμός της σήμανσης που εντοπίστηκε και της λωρίδας κυκλοφορίας που βρίσκεται το όχημα και σε περίπτωση που συμβαδίζουν επιστρέφονται αληθείς τιμές για τις σημάνσεις.

Υποσύστημα εντοπισμού πεζών

Το υποσύστημα εντοπισμού πεζών είναι αυτό που ολοκληρώνει τη λειτουργία του συστήματος αντίληψης του οχήματος και δίνει λύση στο γρήγορο εντοπισμό των

Αλγόριθμος 4.6 Recognize Traffic Lights

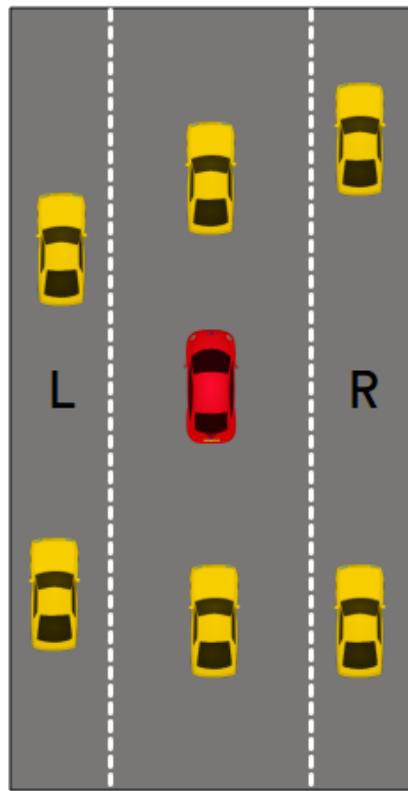
Input: Vehicle Actor Object
Output: Traffic Light State

```
1: traffic_state ← "GREEN"
2: if vehicle_actor.is_at_traffic_light() then    # vehicle_actor is the input's object
3:     traffic_light ← vehicle_actor.get_traffic_light()
4:     if traffic_light.get_state() = carla.TrafficLightState.Red then
5:         traffic_state ← "RED"
6:     else if traffic_light.get_state() = carla.TrafficLightState.Yellow then
7:         traffic_state ← "YELLOW"
8:     else if traffic_light.get_state() = carla.TrafficLightState.Green then
9:         traffic_state ← "GREEN"
10:    end if
11: end if
12: return traffic_state
```

Αλγόριθμος 4.7 Recognize Traffic Signs

Input: Vehicle's current Waypoint
Output: Close Landmarks

```
1: landmarks ← waypoint.get_landmarks(distance = 5)    # 5 is the range in meters
2: speed_limit_sign ← False
3: speed_limit_value ← 0
4: stop_sign ← False
5: for each landmark in landmarks list do
6:     if landmark.type == "274" then    # 274 is speed limit sign's ID in CARLA
7:         orientation ← landmark.orientation
8:         if current waypoint's lane id * orientation > 0 then
9:             speed_limit_sign ← True
10:            speed_limit_value ← landmark.value
11:        end if
12:    end if
13:    if landmark.type == "206" then      # 206 is STOP sign's ID in CARLA
14:        orientation ← landmark.orientation
15:        if current waypoint's lane id * orientation > 0 then
16:            stop_sign ← True
17:        end if
18:    end if
19: end for
20: return speed_limit_sign, speed_limit_value, stop_sign
```



Σχήμα 4.27: Το αυτόνομο όχημα σε κόκκινο χρώμα και τα κοντινότερα οχήματα στις τρεις λωρίδες (ίδια, αριστερή, δεξιά) σε κίτρινο χρώμα.

κοντινότερων πεζών με σκοπό την αποφυγή ατυχημάτων. Η λογική με την οποία εντοπίζονται οι πεζοί είναι όμοια με αυτή του υποσυστήματος εντοπισμού οχημάτων. Συγκεκριμένα, ένα αυτόνομο όχημα ενδιαφέρεται κυρίως για τους πεζούς που βρίσκονται εντός του δρόμου όταν για παράδειγμα οι πεζοί διασχίζουν κάποια διάβαση. Για τον λόγο αυτό λαμβάνεται από το CARLA API η τοποθεσία του κάθε πεζού. Έπειτα, συγκρίνεται το αναγνωριστικό ID της τοποθεσίας τη λωρίδας στην οποία βρίσκεται ο πεζός και το ID της λωρίδας του οχήματος και κατατάσσεται ο καθένας στην κατάλληλη λίστα πεζών. Αυτό σημαίνει ότι δημιουργούνται και πάλι τρεις λίστες με τους πεζούς που βρίσκονται εκείνη τη χρονική στιγμή στην ίδια, τη δεξιά και την αριστερή λωρίδα κυκλοφορίας αλλά και οι αποστάσεις από αυτούς για να γνωρίζει το σύστημα τον βαθμό κινδύνου σύγκρουσης με τον πεζό. Ωστόσο, πέρα από τους πεζούς που υπάρχουν εντός του δρόμου κυκλοφορίας υπάρχουν άνθρωποι που κινούνται πάνω στα πεζοδρόμια. Για αυτό το λόγο οι αποστάσεις που υπολογίζονται αφορούν και αυτούς κατατάσσοντας τους σε ξεχωριστή λίστα και διατηρώντας πάντα ένα συγκεκριμένο όριο στις αποστάσεις μαζί τους.

4.4.5 Υποσύστημα Επιλογής Συμπεριφοράς

Αφού σχεδιαστεί η κατάλληλη τροχιά που θα πρέπει το όχημα να ακολουθήσει έτσι ώστε να φτάσει στον τελικό προορισμό του θα πρέπει να ακολουθήσει κάποιες ενέργειες έτσι ώστε να την ολοκληρώσει με όσο το δυνατόν πιο ομαλό και ασφαλέστερο τρόπο. Κατά τη διάρκεια της πορείας του, αναγκάζεται να αλληλεπιδρά με άλλα στοιχεία του εξωτερικού περιβάλλοντος είτε είναι δυναμικά όπως τα οχήματα και οι πεζοί είτε είναι στατικά όπως οι οδικές σημάνσεις, τα κτίρια κ.α. Γίνεται επομένως αντιληπτό ότι κάθε χρονική στιγμή θα πρέπει να λαμβάνει κάποιες αποφάσεις σχετικά με την κίνηση που θα πραγματοποιήσει τοπικά του δρόμου έτσι ώστε να συνεργάζεται ομαλά με τα υπόλοιπα στοιχεία του οδικού δικτύου αλλά και να συμμορφώνεται κατάλληλα στους κανόνες κυκλοφορίας. Το καθήκον αυτό αναλαμβάνει το υποσύστημα επιλογής συμπεριφοράς το οποίο λαμβάνει τη χρήσιμη πληροφορία από το υποσύστημα αντίληψης και καλείται κάθε χρονική στιγμή να αξιολογήσει το περιβάλλον και τις καταστάσεις στις οποίες βρίσκονται τα στοιχεία του έτσι ώστε να λάβει τη βέλτιστη επιλογή συμπεριφοράς για εκείνο το χρονικό σημείο. Με τον όρο συμπεριφορά εννοείται μία ενέργεια ή ένας συνδυασμός ενεργειών που καλείται να πράξει το όχημα όπως η αλλαγή λωρίδας, η αύξηση ή μείωση της ταχύτητας κλπ. Ένα παράδειγμα καλής συμπεριφοράς ενός αυτόνομου οχήματος είναι όταν θα βρεθεί σε κάποιο φωτεινό σηματοδότη θα πρέπει να μειώσει την ταχύτητα του σε περίπτωση που έχει κόκκινο χρώμα ο σηματοδότης και να αναμένει σταματημένο μέχρι να αλλάξει το χρώμα του σε πράσινο όπου ξεκινάει μια νέα επιλογή συμπεριφοράς. Οι συμπεριφορές που έχουν αναπτυχθεί για το συγκεκριμένο όχημα θα αναλυθούν στη συνέχεια. Έχουν αναπτυχθεί κατά καιρούς διάφορες τεχνικές επιλογής συμπεριφοράς όπως είναι οι μηχανές πεπερασμένων καταστάσεων (FMS) που έχουν προταθεί σε αυτή την έρευνα [13] και χρησιμοποιούν ένα σύνολο από συμπεριφορές οι οποίες ονομάζονται καταστάσεις οι οποίες ενώνονται μεταξύ τους με μεταβάσεις οι οποίες ουσιαστικά δηλώνουν τους κανόνες που πρέπει να ικανοποιούνται για να μεταβεί το σύστημα από τη μία κατάσταση στην άλλη. Επίσης, η ανάπτυξη της τεχνολογίας επικοινωνίας μεταξύ των οχημάτων και των υπόλοιπων στοιχείων του οδικού δικτύου (V2X) έχει συνδράμει σημαντικά στη λύση του προβλήματος επιλογής συμπεριφοράς μειώνοντας το συνολικό φορτίο επιλογής συμπεριφοράς από το όχημα εκμεταλλευόμενη τις πληροφορίες και τις επικείμενες συμπεριφορές των άλλων οχημάτων [2]. Εκτός των ντετερμινιστικών τεχνικών βέβαια έχουν αναπτυχθεί και αλγόριθμοι επιλογής συμπεριφοράς που βασίζονται σε τεχνικές μηχανικής μάθησης και τεχνητής νοημοσύνης εκπαιδεύοντας κατάλληλες δομές μέσα από έναν μεγάλο αριθμό δεδομένων για να δώσει τη δυνατότητα στο όχημα να λαμβάνει σωστές αποφάσεις κάθε χρονική στιγμή [28].

Μέθοδος επιλογής συμπεριφοράς

Το υποσύστημα επιλογής συμπεριφοράς της παρούσας εργασίας πραγματοποιεί μια ομαλή μετάβαση από την αρχική τοποθεσία του οχήματος μέχρι τον τελικό προορισμό που έχει επιλεγεί, υπακούοντας στους κανόνες κυκλοφορίας κατά περιπτώσεις και σεβόμενο τα υπόλοιπα στοιχεία της κυκλοφορίας όπως οχήματα και πεζούς. Ωστόσο, όπως αναφέρθηκε και στη [υποκεφάλαιο 1.2](#) ο σκοπός της συγκεκριμένης εργασίας είναι να μελετήσει τον τρόπο με τον οποίο επηρεάζει τη

4.4. ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΥΛΟΠΟΙΗΣΕΩΝ ΤΟΥ ΑΥΤΟΝΟΜΟΥ ΟΧΗΜΑΤΟΣ

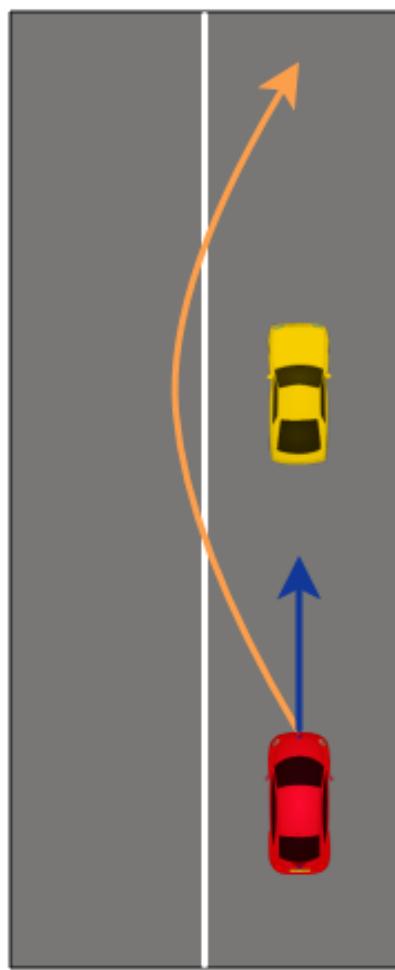
συμπεριφορά του αυτοκινήτου ένα σύνολο από τιμές που δίνονται από το χρήστη και προσαρμόζουν ανάλογα τη συμπεριφορά του οχήματος. Οι τιμές αυτές είναι προφανώς οι τιμές των sliders "aggressive" και "lawful" που δίνονται στο αυτόνομο όχημα μέσω της διεπαφής του NodeRED. Αυτό σημαίνει, ότι το υποσύστημα επιλογής συμπεριφοράς επηρεάζεται από τις τιμές αυτών των sliders και η συμπεριφορά που μπορεί να εκδηλώσει σε παρόμοια σενάρια να διαφέρει ανάμεσα στις διαφορετικές τιμές των sliders. Για παράδειγμα, στο [σχήμα 4.28](#) απεικονίζεται ένα αυτόνομο όχημα σε κόκκινο χρώμα και ένα τυχαίο όχημα που βρίσκεται μπροστά του σε κίτρινο χρώμα. Αν γίνει η υπόθεση ότι το slider "aggressive" έχει θετική τιμή και το slider "lawful" μηδενική, το οποίο μεταφράζεται ότι ο οδηγός θέλει να ακολουθήσει μία επιθετική συμπεριφορά χωρίς υπακοή στους κανόνες τότε το όχημα θα ακολουθήσει την πορτοκαλί τροχιά αγνοώντας τη συνεχή γραμμή στη μέση του δρόμου που απαγορεύεται να διασχίσει. Σε αντίθετη περίπτωση που το "aggressive" έχει μία θετική μικρή τιμή και το "lawful" μία συγκριτικά μεγαλύτερη θετική τιμή τότε το όχημα θα επιβραδύνει και θα κινηθεί πίσω από το κίτρινο αυτοκίνητο ακολουθώντας την μπλε τροχιά. Γίνεται επομένως αντιληπτό ότι το σύστημα θα προσαρμόζει τις αποφάσεις του στις προθέσεις του χρήστη που εκδηλώνονται μέσω των sliders.

Στη συγκεκριμένη διπλωματική εργασία, λόγω της ανάγκης να επιδράσουν οι τιμές των sliders επάνω στην επιλογή συμπεριφοράς αλλά και των πολλών διαφορετικών συνθηκών που υπάρχουν μέσα σε ένα περιβάλλον, η μέθοδος η οποία χρησιμοποιήθηκε είναι αυτή της ανάλυσης απόφασης πολλαπλών κριτηρίων (Multiple-criteria decision-making (MCDM))⁶¹. Η συγκεκριμένη μέθοδος χρησιμοποιείται σε πολλά προβλήματα στην καθημερινότητα, στην οικονομία, στην ιατρική κλπ. και δίνει λύση ρυθμίσεων σε θέματα όπου δεν υπάρχει μια βέλτιστη λύση αλλά αυτή εξαρτάται πολλά και συγκρουόμενα μεταξύ τους κριτήρια. Το πρόβλημα της επιλογής συμπεριφοράς είναι ένα τέτοιο πρόβλημα καθώς κάθε χρονική στιγμή οι συνθήκες που επικρατούν στο εξωτερικό περιβάλλον αλλάζουν και είναι μεγάλες σε αριθμό. Η συγκεκριμένη τεχνική έχει εφαρμοστεί με επιτυχία και σε αυτή την έρευνα [29] για αυτό και προτιμήθηκε.

Περιγραφή του πίνακα της μεθόδου Ανάλυσης Απόφασης Πολλαπλών Κριτηρίων (MCDM)

Στη μέθοδο MCDM τα δεδομένα οργανώνονται σε ένα δισδιάστατο όπου η μία διάσταση αντικατοπτρίζει συνήθως τα κριτήρια με βάση τα οποία πρέπει να εξαχθεί η βέλτιστη λύση και η άλλη διάσταση περιλαμβάνει τις διαθέσιμες επιλογές που υπάρχουν για τη λύση του προβλήματος. Υπάρχουν αρκετές τεχνικές ώστε να υπολογιστεί η βέλτιστη λύση του από τις διαθέσιμες επιλογές που υπάρχουν στη μία διάσταση. Ο δισδιάστατος αυτός πίνακας συνήθως γεμίζει κάθε κελί του με μία τιμή που αντικατοπτρίζει τον βαθμό στον οποίο επιδρά το εκάστοτε κριτήριο στη συγκεκριμένη λύση του προβλήματος. Επίσης, υπάρχουν περιπτώσεις όπου τα στοιχεία της διάστασης που περιλαμβάνει τα κριτήρια επιλογής πολλαπλασιάζονται με διαφορετικούς αριθμούς (βάρη) έτσι ώστε να δοθεί διαφορετική βαρύτητα στο κάθε κριτήριο αξιολόγησης.

⁶¹https://en.wikipedia.org/wiki/Multiple-criteria_decision_analysis



Σχήμα 4.28: Το αυτόνομο όχημα (κόκκινο χρώμα) λαμβάνει την απόφαση για προστέραση (πορτοκαλί τροχιά) του μπροστινού οχήματος (κίτρινο χρώμα) για θετικές τιμές του "aggressive" και την απόφαση να ακολουθήσει το μπροστινό όχημα (μπλε τροχιά) για θετικές τιμές του "lawful".

Στην παρούσα διπλωματική εργασία ο προαναφερόμενος δισδιάστατος πίνακας έχει οργανωθεί ως εξής:

- **Στήλες:** Οι στήλες του πίνακα περιλαμβάνουν όλες τις διαθέσιμες συμπεριφορές που μπορεί να ακολουθήσει το όχημα. Οι συμπεριφορές αυτές έχουν οργανωθεί σε δύο κατηγορίες. Η πρώτη κατηγορία περιλαμβάνει συμπεριφορές που επηρεάζουν μόνο την κατεύθυνση του οχήματος ενώ η δεύτερη περιλαμβάνει τις συμπεριφορές που επηρεάζουν την ταχύτητα του οχήματος. Κάθε συμπεριφορά έχει ξεχωριστή υλοποίηση και η καθεμία από αυτές θα αναλυθεί παρακάτω.
- **Γραμμές:** Οι γραμμές του πίνακα περιλαμβάνουν όλα τα κριτήρια με βάση τα οποία εξάγεται η βέλτιστη συμπεριφορά. Στην ουσία κάθε γραμμή αντιπρο-

4.4. ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΥΛΟΠΟΙΗΣΕΩΝ ΤΟΥ ΑΥΤΟΝΟΜΟΥ ΟΧΗΜΑΤΟΣ

σωπεύει μια ξεχωριστή συνθήκη που υπάρχει στην κυκλοφορία κάθε στιγμή. Για παράδειγμα, μία συνθήκη που υπάρχει είναι το χρώμα του κοντινότερου φωτεινού σηματοδότη το οποίο επηρεάζει άμεσα τη συμπεριφορά του οχήματος. Κάθε μία από αυτές τις συνθήκες συνοδεύεται με τον αριθμό 1 ή 0 που υποδηλώνει αν εκείνη τη στιγμή η συνθήκη αυτή έχει ενεργοποιηθεί ή όχι αντίστοιχα.

- **Κελιά:** Κάθε κελί του πίνακα περιέχει μία σταθερή τιμή f για την οποία ισχύει:

$$f_{i,j} \in [-1, 1] \quad (4.7)$$

$$\begin{aligned} i &\in 1, 2, \dots, N-1, N = \text{number of criteria} \\ j &\in 1, 2, \dots, M-1, M = \text{number of behaviors} \end{aligned}$$

Όταν το συγκεκριμένο κελί i, j έχει θετική τιμή σημαίνει ότι ο κανόνας i επηρεάζει θετικά τη συμπεριφορά j δηλαδή ενθαρρύνει την επιλογή της. Όσο μεγαλύτερη η θετική τιμή του κελιού τόσο μεγαλύτερη επίδραση στην επιλογή της j συμπεριφοράς. Αντίθετα, οι αρνητικές τιμές των κελιών υποδηλώνουν αρνητική επίδραση του i κανόνα στη j συμπεριφορά μειώνοντας τις πιθανότητες να επιλεγεί η συγκεκριμένη συμπεριφορά. Ο τρόπος που θα υπολογιστεί η βέλτιστη συμπεριφορά μέσω του πίνακα θα περιγραφεί παρακάτω.

Ο **πίνακας 4.1** είναι και αυτός που χρησιμοποιείται για την υλοποίηση της μεθόδου MCDM. Όπως είναι φανερό το όνομα κάθε στήλης αποτελεί και μια ξεχωριστή συμπεριφορά, το όνομα κάθε γραμμής αποτελεί και ένα ξεχωριστό κανόνα και η τιμή του κάθε κελιού είναι το βάρος του κανόνα στη συγκεκριμένη συμπεριφορά.

Περιγραφή συμπεριφορών

Οι συμπεριφορές οι οποίες είναι διαθέσιμες και μπορεί να ακολουθήσει ένα όχημα είναι επτά και χωρίζονται σε δύο μεγάλες κατηγορίες:

1. **Συμπεριφορές που καθορίζουν την κατεύθυνση του οχήματος**
2. **Συμπεριφορές που καθορίζουν την ταχύτητα του οχήματος**

Για την καλύτερη κατανόηση των αλγορίθμων που περιγράφουν την υλοποίηση των συμπεριφορών σημειώνονται κάποια κοινά στοιχεία των αλγορίθμων:

- **global_velocity:** Η `global_velocity` αναφέρεται στην ταχύτητα που καθορίζεται από το slider "Set Speed" της διεπαφής και είναι η ταχύτητα που είτε έχει τεθεί από το χρήστη είτε από κάποια αναγκαστική συμπεριφορά που πρόκειται να εφαρμοστεί. Η ταχύτητα δηλαδή αυτή δεν είναι η ταχύτητα που έχει το όχημα αλλά αυτή που επιθυμεί να φτάσει μέσω του συστήματος ελέγχου. Σε κάθε κύκλο του συστήματος η ταχύτητα αυτή δίνεται συνεχώς ως είσοδος και αποτελεί την ταχύτητα αναφοράς του συστήματος.

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

- **current_velocity:** Η current_velocity είναι η ταχύτητα που έχει το όχημα σε μια χρονική στιγμή. Σε πολλές περιπτώσεις μπορεί και να ταυτίζεται με την global_velocity αν το σύστημα έχει φτάσει στην επιθυμητή ταχύτητα χωρίς κάποιο περισπασμό. Ο [αλγόριθμος 4.8](#) υπολογίζει την τρέχουσα ταχύτητα του οχήματος. Λαμβάνεται αρχικά το διάνυσμα της ταχύτητας από τον CARLA, μετατρέπεται σε πίνακα και υπολογίζεται το μέτρο του. Η τιμή που επιστρέφεται πολλαπλασιάζεται με 3.6 για να μετατραπεί σε km/h.

Αλγόριθμος 4.8 Calculate Current Velocity

```
1: function CALCULATE_CURRENT_VELOCITY
2:   velocity_vector  $\leftarrow$  get velocity vector from CARLA
3:   velocity_array  $\leftarrow$  [velocity_vector.x, velocity_vector.y, velocity_vector.z]
4:   velocity_norm  $\leftarrow$  meter of velocity_array
5:   current_velocity  $\leftarrow$  3.6 * velocity_norm
6:   return current_velocity
7: end function
```

- **Publish / Subscribe velocity:** Το topic "velocity" είναι αυτό από το οποίο λαμβάνει (subscribe) το σύστημα την επιθυμητή ταχύτητα (global_velocity) που θέλει να επιτύχει. Κάθε φορά που η ταχύτητα του συστήματος πρέπει να αλλάξει εξαιτίας κάποιας διαδικασίας, η νέα ταχύτητα δημοσιεύεται (publish) στο συγκεκριμένο topic.

Στην πρώτη κατηγορία ανήκουν οι συμπεριφορές που βρίσκονται στις τρεις πρώτες στήλες του πίνακα και αναλύονται παρακάτω:

- **Left Lane Change:** Το όχημα πραγματοποιεί αλλαγή λωρίδας προς τα αριστερά.
- **Right Lane Change:** Το όχημα πραγματοποιεί αλλαγή λωρίδας προς τα δεξιά.
Στις δύο παραπάνω περιπτώσεις για την επίτευξη της αλλαγής λωρίδας θα πρέπει να αλλάξουν τα waypoints της [εξίσωσης 4.1](#) με τρόπο που θα εξηγηθεί παρακάτω στον [αλγόριθμο 4.4 alg:right-left-change](#). Ο [αλγόριθμος 4.9](#) χρησιμοποιείται για να αλλάξει το όχημα λωρίδα είτε έπειτα από επιλογή του χρήστη πατώντας ένα από τα πλήκτρα RIGHT ή LEFT είτε αναγκαστικά όταν έχει επιλεχθεί ως η βέλτιστη συμπεριφορά.

Η λειτουργία του [αλγορίθμου 4.9](#) χωρίζεται σε δύο βασικά μέρη. Το πρώτο μέρος αφορά τις περιπτώσεις που το όχημα βρίσκεται στην αρχική λωρίδα κυκλοφορίας και στις περιπτώσεις που έχει αλλάξει λωρίδα είτε προς τα δεξιά είτε προς τα αριστερά. Αυτά τα μέρη χωρίζονται από τη μεταβλητή current_state η οποία παίρνει τις τιμές "INIT", "RIGHT" και "LEFT" για την κάθε περίπτωση αντίστοιχα. Η μεταβλητή turn που δίνεται ως είσοδος υποδηλώνει την κατεύθυνση προς την οποία θέλει να κινηθεί το όχημα. Σε περίπτωση που έχει τιμή None τότε σημαίνει ότι θα συνεχίσει κανονικά την πορεία του στη λωρίδα που βρίσκεται. Η μεταβλητή current_index που χρησιμοποιείται, διατηρεί τον δείκτη στον πίνακα waypoints, ο οποίος δείχνει το waypoint στο

4.4. ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΥΛΟΠΟΙΗΣΕΩΝ ΤΟΥ ΑΥΤΟΝΟΜΟΥ ΟΧΗΜΑΤΟΣ

οποίο βρίσκεται κάθε στιγμή το όχημα. Αποτελεί global μεταβλητή στο πρόγραμμα που σημαίνει ότι ανανεώνεται σε διαφορετικά σημεία του προγράμματος και όχι μόνο στη συγκεκριμένη συνάρτηση. Στο πρώτο μέρος λοιπόν αν η μεταβλητή *turn* έχει λάβει τιμή "RIGHT" ή "LEFT" τότε υπολογίζεται το καινούργιο waypoint που πρόκειται να κατεύθυνθεί το όχημα στη νέα λωρίδα και σε περίπτωση που είναι δυνατή η μετάβαση αυτή, αντικαθίσταται το καινούργιο waypoint με το παλιό στον πίνακα με όλα τα waypoints της τροχιάς. Η μεταβλητή *current_state* ανανεώνεται με την τιμή της νέας λωρίδας στην οποία μεταβαίνει το όχημα. Σε οποιαδήποτε άλλη περίπτωση που δεν μπορεί να αλλάξει το waypoint η μεταβλητή *current_state* διατηρεί την τιμή της ως έχει. Το δεύτερο μέρος χειρίζεται τις περιπτώσεις που το όχημα δε βρίσκεται στην αρχική αλλά έχει ήδη αλλάξει λωρίδα. Ο πίνακας waypoints περιέχει τα waypoints της αρχικής τροχιάς. Αυτό σημαίνει ότι σε περίπτωση που το όχημα βρίσκεται στην αριστερή ή τη δεξιά τροχιά θα πρέπει συνεχώς να ανανεώνει τα waypoints του πίνακα για να διατηρείται στην τροχιά που βρίσκεται και να μη γυρνά στην αρχική. Η διαδικασία που εκτελείται είναι ο υπολογισμός των καινούργιων waypoints είτε δεξιά είτε αριστερά και η αντικατάσταση τους στον πίνακα waypoints. Η λειτουργία αυτή εκτελείται είτε όταν δεν έχει δοθεί εντολή να αλλάξει λωρίδα το όχημα (*turn = None*) είτε όταν δίνεται εντολή αλλαγής λωρίδας προς την ίδια κατεύθυνση με τη λωρίδα που βρίσκεται. Σε κάθε άλλη περίπτωση το όχημα μεταβαίνει στην αρχική λωρίδα.

- **Keep Straight:** Το όχημα συνεχίζει την πορεία του κανονικά στη λωρίδα που βρίσκεται. Σε αυτή την περίπτωση τα waypoints της [εξίσωσης 4.1](#) παραμένουν ανεπηρέαστα και το όχημα συνεχίζει να τα ακολουθεί.

Στη δεύτερη κατηγορία ανήκουν οι συμπεριφορές που βρίσκονται στις τέσσερις τελευταίες στήλες του πίνακα και αναλύονται παρακάτω:

- **Speed Up:** Το όχημα επιταχύνει αυξάνοντας την ταχύτητα του σε μια επιθυμητή τιμή, εφόσον η επιθυμητή τιμή είναι μεγαλύτερη από την τρέχουσα τιμή της ταχύτητας. Ο [αλγόριθμος 4.10](#) είναι αυτός που εφαρμόζεται όταν δίνεται εντολή για επιτάχυνση του οχήματος. Αφού ελεγχθεί αν υπάρχει άλλη ενέργεια σε εκκρεμότητα, υπολογίζεται η ταχύτητα που έχει το όχημα εκείνη τη στιγμή και συγκρίνεται με την επιθυμητή η οποία δίνεται ως είσοδος στη συνάρτηση. Αν είναι μικρότερη της επιθυμητής η ταχύτητα αναφοράς αυξάνεται στην τιμή της επιθυμητής και δημοσιεύεται στο topic "velocity" για να γνωρίζει το σύστημα ότι η ταχύτητα που πρέπει να φτάσει έχει αλλάξει. Σε αντίθετη περίπτωση η ταχύτητα διατηρείται ως έχει. Ο αλγόριθμος τελικά στέλνει την επιθυμητή ταχύτητα στο σύστημα ελέγχου για να εφαρμόσει την κατάλληλη αλλαγή στην κίνηση του αυτοκινήτου.
- **Slow Down:** Το όχημα επιβραδύνει μειώνοντας την ταχύτητα του σε μια επιθυμητή τιμή, εφόσον η επιθυμητή τιμή είναι μικρότερη από την τρέχουσα τιμή της ταχύτητας. Ο [αλγόριθμος 4.11](#) υλοποιεί τη διαδικασία επιβράδυνσης του οχήματος και λειτουργεί με ακριβώς ίδιο τρόπο με αυτόν της επιτάχυνσης του οχήματος. Η μόνη διαφορά εντοπίζεται στο σημείο όπου ελέγχεται η ταχύτητα

Αλγόριθμος 4.9 Right / Left Lane Change

Require: turn

```

1: if current_state == "INIT" then
2:   if turn != None then
3:     previous_waypoint ← waypoints[current_index]
4:     new_waypoint ← current waypoint's left or right waypoint
5:     if new_waypoint exists then
6:       waypoints[current_index] ← new_waypoint
7:       if new_waypoint == previous_waypoint then
8:         current_state ← "INIT"
9:       else
10:        current_state ← turn
11:      end if
12:    else
13:      current_state ← "INIT"
14:    end if
15:  else
16:    current_state ← "INIT"
17:  end if
18: else if current_state == "LEFT" ∨ current_state == "RIGHT" then
19:   if turn == current_state ∨ turn == None then
20:     previous_waypoint ← waypoints[current_index]
21:     if current_state == "LEFT" then
22:       new_waypoint ← initial waypoint's left waypoint
23:     else if current_state == "RIGHT" then
24:       new_waypoint ← initial waypoint's right waypoint
25:     end if
26:     if new_waypoint exists then
27:       waypoints[current_index] ← new_waypoint
28:       if new_waypoint == previous_waypoint then
29:         waypoints[current_index] ← waypoints[current_index + offset]
30:         current_state ← "INIT"
31:       end if
32:     else
33:       current_state ← "INIT"
34:     end if
35:   else if turn != current_state then
36:     current_state ← "INIT"
37:   end if
38: end if

```

4.4. ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΥΛΟΠΟΙΗΣΕΩΝ ΤΟΥ ΑΥΤΟΝΟΜΟΥ ΟΧΗΜΑΤΟΣ

Αλγόριθμος 4.10 Speed Up

Require: desired_velocity

```
1: if another action is in operation then
2:   return
3: end if
4: current_velocity ← calculate_current_velocity()
5: if current_velocity < desired_velocity then
6:   global_velocity ← desired_velocity
7:   Publish global_velocity in "velocity" topic
8: else
9:   Keep the current velocity
10: end if
11: Send the new velocity in control system to apply new control
12: Spend some simulation cycles in order to apply acceleration command
```

και σε αυτή την περίπτωση εφαρμόζεται η επιβράδυνση όταν η τρέχουσα ταχύτητα του οχήματος είναι μεγαλύτερη από την επιθυμητή.

Αλγόριθμος 4.11 Slow Down

Require: desired_velocity

```
1: if another action is in operation then
2:   return
3: end if
4: current_velocity ← calculate_current_velocity()
5: if current_velocity > desired_velocity then
6:   global_velocity ← desired_velocity
7:   Publish global_velocity in "velocity" topic
8: else
9:   Keep the current velocity
10: end if
11: Send the new velocity in control system to apply new control
12: Spend some simulation cycles in order to apply deceleration command
```

- **Keep Velocity:** Το όχημα διατηρεί αμετάβλητη την τρέχουσα ταχύτητα του. Ο [αλγόριθμος 4.12](#) πραγματοποιεί τη διατήρηση της ταχύτητας. Ελέγχεται αρχικά αν πραγματοποιείται άλλη διαδικασία και έπειτα καλείται η συνάρτηση που υπολογίζει την ταχύτητα που έχει το όχημα. Αν η ταχύτητα αυτή είναι μικρότερη από 0.5 km/h πράγμα που σημαίνει ότι είναι πολύ χαμηλή τότε ο αλγόριθμος διατηρεί ένα κατώτατο όριο των 10 km/h όταν διότι για χαμηλότερες ταχύτητες το όχημα κινείται πολύ αργά και δυσκολεύεται να προσομοιώσει την κατάσταση διατήρησης ταχύτητας. Σε αντίθετη περίπτωση απλά δίνει εντολή στο σύστημα ελέγχου να διατηρήσει την ίδια ταχύτητα.
- **Stop:** Το όχημα επιβραδύνει απότομα μέχρι να μηδενίσει την ταχύτητα του και να σταματήσει την κίνηση του. Ο [αλγόριθμος 4.13](#) περιγράφει τη διαδικα-

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

Αλγόριθμος 4.12 Keep Velocity

```
1: if another action is in operation then
2:   return
3: end if
4: current_velocity ← calculate_current_velocity()
5: if current_velocity < 0.5 then
6:   speed_up(10)
7: else
8:   Keep the current velocity
9: end if
10: Send the new velocity in control system to apply new control
11: Spend some simulation cycles in order to keep velocity
```

σία που ακολουθείται όταν η βέλτιστη επιλογή ταχύτητας είναι η "Stop". Σε περίπτωση που κάποια άλλη ενέργεια πραγματοποιείται χωρίς να έχει ολοκληρωθεί τότε ο αλγόριθμος επιστρέφει. Σε αντίθετη περίπτωση ελέγχεται αν το όχημα βρίσκεται εκτός της διασταύρωσης και εφαρμόζεται το σταμάτημα του αυτοκινήτου δίνοντας κατάλληλη εντολή για μηδενισμό της ταχύτητας στο σύστημα ελέγχου. Αν το όχημα βρίσκεται εντός διασταύρωσης τότε διατηρείται η ταχύτητα του για να μεταφερθεί εκτός αυτής. Αυτή η σύμβαση έχει γίνει γιατί θεωρείται ασφαλέστερο να φύγει εκτός της διασταύρωσης παρά να παραμείνει σταματημένο μέσα.

Αλγόριθμος 4.13 Stop

```
1: if another action is in operation then
2:   return
3: end if
4: if ego vehicle not in junction then
5:   global_velocity ← 0
6:   Publish zero velocity in "velocity" topic
7: else
8:   Apply current velocity
9: end if
10: Send the new velocity in control system to apply new control
11: Spend some simulation cycles in order to apply stop command
```

Οι συγκεκριμένες συμπεριφορές επιλέχθηκαν διότι αποτελούν τις πιο θεμελιώδεις κινήσεις που μπορεί να πραγματοποιήσει ένα όχημα κατά τη διάρκεια της πορείας του. Μέσα από αυτές τις συμπεριφορές μπορούν να προκύψουν και άλλες πιο σύνθετες. Για παράδειγμα, μία πολύ συχνή συμπεριφορά ενός οχήματος είναι η προσπέραση του προπορευόμενου οχήματος. Η συγκεκριμένη συμπεριφορά στην ουσία πραγματοποιείται εφόσον το επιτρέπουν οι συνθήκες με μία αριστερή αλλαγή λωρίδας, μία επιτάχυνση και μία δεξιά αλλαγή λωρίδας. Οπότε, παρόμοιες συμπεριφορές μπορούν να προκύψουν από το σύνολο των στοιχειωδών συμπεριφορών που έχουν υλοποιηθεί.

4.4. ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΥΛΟΠΟΙΗΣΕΩΝ ΤΟΥ ΑΥΤΟΝΟΜΟΥ ΟΧΗΜΑΤΟΣ

Περιγραφή κανόνων

Οι κανόνες με βάση τους οποίους εξάγεται η βέλτιστη συμπεριφορά σε κάθε κύκλο αξιολόγησης και αποτελούν τις γραμμές του πίνακα 4.1 περιγράφονται παρακάτω. Οι τιμές που επιστρέφουν οι συναρτήσεις που υλοποιούν τους κανόνες είναι *True* ή *False* και ο λόγος είναι για να γνωρίζει το υπόλοιπο μέρος του προγράμματος μόνο αν έχει ενεργοποιηθεί ή όχι ο εκάστοτε κανόνας.

- **R1** → Το αυτόνομο όχημα έχει μεγαλύτερη ταχύτητα από το κοντινότερο μπροστινό του όχημα.
- **R2** → Υπάρχει κοντινό όχημα μπροστά στο αυτόνομο όχημα στην ίδια λωρίδα.
- **R3** → Το αυτόνομο όχημα βρίσκεται κοντά σε διασταύρωση.
- **R4** → Το αυτόνομο όχημα βρίσκεται κοντά σε σήμανση STOP.
- **R5** → Το αυτόνομο όχημα βρίσκεται κοντά στον τελικό προορισμό.
- **R6** → Η μπροστινή περιοχή του οχήματος είναι ελεύθερη για αρκετό χρόνο.
- **R7** → Το αυτόνομο όχημα βρίσκεται σε κόκκινο φανάρι.
- **R8** → Η ταχύτητα του αυτόνομου οχήματος είναι μηδενική.
- **R9** → Η δεξιά λωρίδα είναι ασφαλής.
- **R10** → Η αριστερή λωρίδα είναι ασφαλής.
- **R11** → Η αριστερή λωρίδα έχει ίδια κατεύθυνση με αυτή που κινείται το όχημα.
- **R12** → Η δεξιά λωρίδα είναι πυκνότερη, έχει δηλαδή περισσότερα οχήματα, από την αριστερή.
- **R13** → Η αριστερή λωρίδα είναι πυκνότερη, έχει δηλαδή περισσότερα οχήματα, από τη δεξιά.
- **R14** → Πεζός βρίσκεται σε κοντινή απόσταση δεξιά από το όχημα.
- **R15** → Πεζός βρίσκεται σε κοντινή απόσταση αριστερά από το όχημα.
- **R16** → Πεζός βρίσκεται σε κοντινή απόσταση μπροστά από το όχημα.
- **R17** → Η διαγράμμιση στη δεξιά λωρίδα του δρόμου επιτρέπει αλλαγή λωρίδας στα δεξιά (διακεκομμένη γραμμή).
- **R18** → Η διαγράμμιση στην αριστερή λωρίδα του δρόμου επιτρέπει αλλαγή λωρίδας στα αριστερά (διακεκομμένη γραμμή).
- **R19** → Το αυτόνομο όχημα βρίσκεται σε λωρίδα διπλής κυκλοφορίας για μεγάλο χρονικό διάστημα.

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

- **R20** → Η αριστερή λωρίδα από αυτή που βρίσκεται το αυτόνομο όχημα είναι διπλής κατευθύνσεως.
- **R21** → Η ταχύτητα του αυτόνομου οχήματος είναι μεγαλύτερη από το όριο ταχύτητας του δρόμου.
- **R22** → Το αυτόνομο όχημα λόγω πιθανής δυσκολίας στην λωρίδα που κινείται πρέπει να μπει στην δεξιά του λωρίδα.
- **R23** → Το αυτόνομο όχημα λόγω πιθανής δυσκολίας στην λωρίδα που κινείται πρέπει να μπει στην αριστερή του λωρίδα.
- **R24** → Το πιθανό όχημα που εντοπίστηκε πριν κάποιο διάστημα βρίσκεται ακόμη μπροστά στο αυτόνομο όχημα.
- **R25** → Η δεξιά λωρίδα δεν είναι ασφαλής για μετακίνηση εκεί.
- **R26** → Η αριστερή λωρίδα δεν είναι ασφαλής για μετακίνηση εκεί.
- **R27** → Η διασταύρωση μπροστά από τη σήμανση STOP είναι ελεύθερη για μετακίνηση.

Περιγραφή Καθορισμού Παραμέτρων

Ένα από τα βασικά αντικείμενα μελέτης της παρούσας εργασίας, όπως αναφέρθηκε, είναι η παρατήρηση της συμπεριφοράς του αυτοκινήτου για διαφορετικές τιμές των sliders "aggressive" και "lawful". Αυτό σημαίνει ότι μελετώνται οι αποφάσεις που θα πάρει το όχημα σε κάθε κύκλο αξιολόγησης των συνθηκών. Εκτός βέβαια από τις αποφάσεις που θα ληφθούν από το όχημα οι τιμές των sliders και η κατάσταση που βρίσκεται το όχημα, καθορίζουν και συγκεκριμένες παραμέτρους στη διαδικασία της πορείας του οχήματος οι οποίες θα εξηγηθούν στη συνέχεια. Ένα μέρος του υποσυστήματος συμπεριφοράς αποτελεί το κομμάτι που συνδέεται με το υποσύστημα επικοινωνίας για να λάβει τις τιμές των sliders και τις χρησιμοποιεί για να υπολογίσει τις τιμές διαφόρων παραμέτρων. Επίσης, η ταχύτητα που έχει το όχημα συμμετέχει στον καθορισμό αυτών των παραμέτρων. Παρακάτω παρουσιάζονται οι παράμετροι που ορίζονται και ο τρόπος που λαμβάνουν τις τιμές τους:

- **Ασφαλής απόσταση από το μπροστινό όχημα:** Η απόσταση αυτή είναι η ελάχιστη απόσταση που πρέπει να έχει το αυτόνομο όχημα από το μπροστινό κοντινότερο όχημα για να το εντοπίσει. Όταν δηλαδή η απόσταση που έχουν τα δύο οχήματα πέσει κάτω από τη συγκεκριμένη αυτή τιμή τότε εντοπίζεται το μπροστινό όχημα. Η ασφαλής μπροστινή απόσταση ωστόσο δεν είναι ίδια για όλες τις ταχύτητες των οχημάτων αλλά καθορίζεται από τη σχετική τους ταχύτητα. Η σχετική ταχύτητα των οχημάτων είναι η διαφορά της ταχύτητας του αυτόνομου οχήματος από την ταχύτητα του κοντινότερου μπροστινού οχήματος. Σύμφωνα με τη [30] προκύπτουν οι ασφαλείς αποστάσεις για συγκεκριμένες σχετικές ταχύτητες των οχημάτων. Οι σχετικές ταχύτητες ξεκινούν από το 0 και καταλήγουν στην τιμή 100. Για τις σχετικές ταχύτητες που

4.4. ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΥΛΟΠΟΙΗΣΕΩΝ ΤΟΥ ΑΥΤΟΝΟΜΟΥ ΟΧΗΜΑΤΟΣ

		Behaviors						
		Left Lane change	Right Lane change	Keep Straight	Speed Up	Slow Down	Keep Velocity	Stop
Rules	R1	0	0	0	0	0	0	0
	R2	0	0	0	0	0	0	0
	R3	0	0	0	0	0	0	0
	R4	0	0	0	0	0	0	0
	R5	0	0	0	0	0	0	0
	R6	0	0	0	0	0	0	0
	R7	0	0	0	0	0	0	0
	R8	0	0	0	0	0	0	0
	R9	0	0	0	0	0	0	0
	R10	0	0	0	0	0	0	0
	R11	0	0	0	0	0	0	0
	R12	0	0	0	0	0	0	0
	R13	0	0	0	0	0	0	0
	R14	0	0	0	0	0	0	0
	R15	0	0	0	0	0	0	0
	R16	0	0	0	0	0	0	0
	R17	0	0	0	0	0	0	0
	R18	0	0	0	0	0	0	0
	R19	0	0	0	0	0	0	0
	R20	0	0	0	0	0	0	0
	R21	0	0	0	0	0	0	0
	R22	0	0	0	0	0	0	0
	R23	0	0	0	0	0	0	0
	R24	0	0	0	0	0	0	0
	R25	0	0	0	0	0	0	0
	R26	0	0	0	0	0	0	0
	R27	0	0	0	0	0	0	0

Πίνακας 4.1: Πίνακας βαρών των κανόνων (γραμμές) σε σχέση με τις συμπεριφορές (στήλες), από τον οποίο προκύπτει η βέλτιστη συμπεριφορά κάθε στιγμή

δεν ανήκουν σε κάποια από τις διακριτές τιμές του πίνακα χρησιμοποιείται η μέθοδος της γραμμικής παρεμβολής για τον υπολογισμό της ασφαλούς απόστασης σύμφωνα με τη εξίσωση 4.8. Στη εξίσωση 4.8 τα v_1 και v_2 είναι τα άνω και κάτω όρια του διαστήματος της σχετικής ταχύτητας στο οποίο ανήκει η ταχύτητα (current_velocity) που έχει εκείνη τη στιγμή το αυτόνομο όχημα. Τα x_1 και x_2 είναι τα αντίστοιχα άνω και κάτω όρια του διαστήματος των ασφαλών αποστάσεων. Η συνάρτηση max χρησιμοποιείται για να μην πέφτει η ασφαλής απόσταση κάτω από τα δέκα μέτρα.

$$safe_distance = \max(10, x_1 + (current_velocity - v_1) \frac{x_1 - x_2}{v_1 - v_2}) \quad (4.8)$$

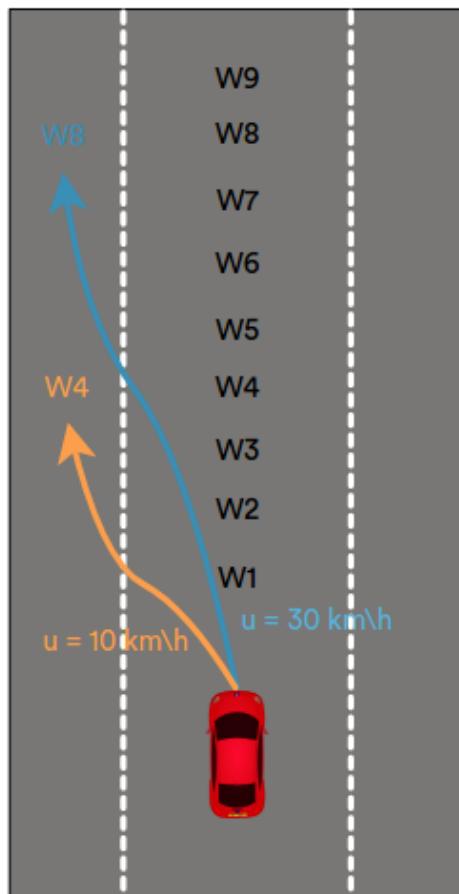
Relative Velocity (km/h)	Safe Distance (m)
0	1.5
5	4.22
10	6.72
15	9.49
20	12.54
25	15.86
30	19.46
35	23.33
40	27.49
45	31.91
50	34.50
55	38.2
60	41.56
65	44.82
70	48.4
75	52.33
80	55.4
85	60.31
90	65.45
95	70.32
100	76.34

Πίνακας 4.2: Πίνακας που απεικονίζει τις σχετικές ταχύτητες δύο οχημάτων και τις αντίστοιχες ασφαλείς αποστάσεις που πρέπει να διατηρούν.

- **Απόσταση από τα κοντινότερα μπροστινά πλάγια (δεξιά και αριστερή λωρίδα) οχήματα:** Οι αποστάσεις αυτές είναι όμοιες με την απόσταση ασφαλείας από το μπροστινό όχημα αλλά αναφέρεται στα κοντινότερα οχήματα στις πλάγιες λωρίδες κυκλοφορίας από αυτή που βρίσκεται το όχημα. Η απόσταση αυτή υποδηλώνει το όριο κάτω από το οποίο αν πέσει η απόσταση του αυτόνομου οχήματος από το κάθε εμπόδιο τότε θα εντοπιστεί το πλάγιο όχημα. Η απόσταση αυτή δεν είναι ίδια πάντα αλλά εξαρτάται από την τιμή του slider "aggressive". Συγκεκριμένα, οι τιμές του slider "aggressive" κυμαίνονται στο διάστημα [0, 10] και αντιστοιχίζονται μέσω γραμμικής παρεμβολής στο διάστημα [20, 10]. Όπως γίνεται αντιληπτό, η ελάχιστη δυνατή τιμή επιθετικότητας που μπορεί να θέσει ο οδηγός η οποία είναι μηδενική, αντιστοιχίζεται στη μεγαλύτερη δυνατή απόσταση που είναι τα 20 μέτρα. Το ίδιο ισχύει και για τη μέγιστη δυνατή τιμή επιθετικότητας η οποία αντιστοιχίζεται στην ελάχιστη απόσταση. Τα δύο διαστήματα τιμών έχουν δηλαδή αρνητική συσχέτιση. Η λογική πίσω από αυτή τη συσχέτιση στηρίζεται στο γεγονός ότι όσο πιο επιθετικός θέλει να γίνει ο οδηγός τόσο μικρότερες θα είναι και οι αποστάσεις ασφαλείας που θέλει να αφήσει από τα πλάγια εμπόδια.

4.4. ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΥΛΟΠΟΙΗΣΕΩΝ ΤΟΥ ΑΥΤΟΝΟΜΟΥ ΟΧΗΜΑΤΟΣ

- **Απόσταση από τα κοντινότερα πίσω πλάγια (δεξιά και αριστερή λωρίδα) οχήματα:** Η απόσταση αυτή αποτελεί το ελάχιστο όριο κάτω από τα οποίο όταν πέσει η απόσταση του αυτόνομου οχήματος από τα πλάγια πίσω εμπόδια τότε αυτά εντοπίζονται. Οι αποστάσεις αυτές ποικίλουν για τις διάφορες τιμές του slider "aggressive". Με τον ίδιο ακριβώς τρόπο και την ίδια λογική που ακολουθήθηκε για τον υπολογισμό των ελάχιστων ορίων των αποστάσεων από τα πλάγια μπροστινά εμπόδια, υπολογίζεται και αυτή η απόσταση. Η διαφορά βρίσκεται στο εύρος των τιμών στο οποίο μετατρέπεται με τις τιμές του slider "aggressive" να αντιστοιχίζονται στο εύρος [10, 5] αφήνοντας έτσι μεγαλύτερο περιθώριο για να έρθουν κοντύτερα τα πίσω οχήματα μιας και δεν επηρεάζουν σε τόσο μεγάλο βαθμό το αυτόνομο όχημα.
- **Αριθμός παραλειπομένων waypoints:** Κατά τις αλλαγές λωρίδας του οχήματος, είτε αυτές γίνονται λόγω κάποιας απόφασης από το υποσύστημα επιλογής συμπεριφοράς είτε πραγματοποιούνται μετά από εντολή του οδηγού, το αυτοκίνητο θα πρέπει να παραλείψει κάποια από τα waypoints της πορείας του κατά τη μετάβαση του από τη μία λωρίδα στην άλλη έτσι ώστε να γίνει πιο ομαλή η πορεία του. Ο αριθμός των waypoints που θα πρέπει να παραλειφθούν καθορίζεται από τη συγκεκριμένη παράμετρο και μεταβάλλεται σε σχέση με την ταχύτητα του οχήματος. Η ταχύτητα του οχήματος ανήκει στο διάστημα [0, 100] km/h και αντιστοιχίζεται μέσω γραμμικής παρεμβολής στο διάστημα [1, 20]. Η παρεμβαλλόμενη τιμή στρογγυλοποιείται στον κοντινότερο ακέραιο προς τα πάνω. Υπάρχει δηλαδή θετική συσχέτιση ανάμεσα στην ταχύτητα και στον αριθμό των waypoints τα οποία παραλείπονται. Αυτό είναι λογικό διότι για μεγαλύτερες ταχύτητες το όχημα θα πρέπει να μπει πιο ομαλά στην άλλη λωρίδα άρα να παραλείψει και περισσότερα waypoints. Για παράδειγμα, αν το όχημα κινείται με 10 km/h τότε τα waypoints που θα πρέπει να παραληφθούν είναι 3, ενώ αν κινείται με 30 km/h θα πρέπει να παραληφθούν 7. Στο σχήμα 4.29 φαίνονται οι δύο αυτές περιπτώσεις. Στην πρώτη περίπτωση το όχημα ακολουθεί την πορτοκαλί διαδρομή και θέλει να φτάσει το τέταρτο waypoint παραλείποντας τα δύο ενδιάμεσα και στη δεύτερη περίπτωση ακολουθεί την μπλε διαδρομή στοχεύοντας στο όγδοο waypoint. Όπως είναι αντιληπτό, η μπλε διαδρομή που αντιστοιχεί σε μεγαλύτερη ταχύτητα είναι πιο ομαλή για μετάβαση στην αριστερή λωρίδα. Ο αριθμός αυτός έχει ήδη αναφερθεί στον αλγόριθμο 4.9 με τη μεταβλητή "offset" η οποία χρησιμοποιήθηκε στην περίπτωση που το όχημα βρίσκεται σε αριστερή ή δεξιά λωρίδα και θα έπρεπε να επανέλθει στην αρχική οπότε και παρέλειπε συγκεκριμένο αριθμό από waypoints για να εισέλθει πιο ομαλά στη λωρίδα.
- **Ποσοστό μείωσης ταχύτητας:** Σε περίπτωση που οι συνθήκες είναι τέτοιες ώστε το όχημα καλείται να μειώσει την ταχύτητα τότε, θα πρέπει να αποφανθεί σχετικά με το ποσοστό αυτής της μείωσης. Το ποσοστό αυτής της μείωσης εξαρτάται από τον βαθμό επιθετικότητας που έχει επιλέξει ο οδηγός να διαθέτει το όχημα του. Η λογική που ακολουθείται είναι ότι για μεγαλύτερες τιμές του slider "aggressive", το όχημα θα έχει και μικρότερες μειώσεις στην ταχύτητα του για να διατηρήσει την επιθετική του συμπεριφορά. Συγκεκριμένα, μέσω γραμμικής παρεμβολής το εύρος [0, 10] στο οποίο ανήκουν



Σχήμα 4.29: Το αυτόνομο όχημα παραλείπει 7 waypoints στην περίπτωση που κινείται με 30 km/h ενώ στην περίπτωση που κινείται με 10 km/h παραλείπει 3 waypoints.

οι τιμές του "aggressive" αντιστοιχίζεται στο εύρος [0.2, 0.8]. Αυτό σημαίνει ότι υπάρχει θετική συσχέτιση ανάμεσα στα δύο εύρη τιμών. Η μέγιστη τιμή επιθετικότητας (10) αντιστοιχίζεται στη μέγιστη δυνατή (0.8) διότι η καινούργια ταχύτητα προκύπτει πολλαπλασιάζοντας την τρέχουσα ταχύτητα με τη μετατετραμένη τιμή. Η τελική ταχύτητα προκύπτει μετά από εφαρμογή της συνάρτησης max ορίζοντας ως κατώτατο όριο την τιμή 10 km/h για την καλύτερη εκτέλεση της προσομοίωσης. Η εξίσωση 4.9 υπολογίζει τη νέα μειωμένη τιμή της ταχύτητας. Για παράδειγμα, αν η ταχύτητα του οχήματος είναι 30 km / h και ο βαθμός επιθετικότητας είναι 10 τότε η νέα μειωμένη ταχύτητα θα είναι $30 \times 0.8 = 24 km/h$, ενώ για βαθμό επιθετικότητας 4 η μειωμένη ταχύτητα θα είναι $30 \times 0.44 = 13.2 km/h$.

$$\begin{aligned}
 f(x) &= \text{interpolate } x \text{ from } [0, 10] \text{ to } [0.2, 0.8], \\
 \text{parameter}_{\in[0.2,0.8]} &= f(\text{aggressive_value}_{\in[0,10]}), \\
 \text{new_velocity} &= \text{parameter}_{\in[0.2,0.8]} \times \text{current_velocity}, \\
 \text{new_velocity} &= \max(10, \text{new_velocity})
 \end{aligned} \tag{4.9}$$

- Ποσοστό αύξησης ταχύτητας:** Παρόμοια λογική με αυτή της μείωσης της ταχύτητας ακολουθείται στην περίπτωση που το όχημα καλείται να αυξήσει την ταχύτητα του. Και σε αυτή την περίπτωση η νέα ταχύτητα εξαρτάται από την τιμή επιθετικότητας που έχει θέσει ο οδηγός στο όχημα. Συγκεκριμένα, οι τιμές του slider "aggressive" αντιστοιχίζονται από το διάστημα $[0, 10]$ στο διάστημα $[1.1, 2]$ μέσω γραμμικής παρεμβολής. Γίνεται επομένως αντιληπτό ότι για τη μέγιστη τιμή επιθετικότητας η ταχύτητα διπλασιάζεται. Η τελική ταχύτητα προκύπτει μετά από εφαρμογή της συνάρτησης \min ορίζοντας ως ανώτατο όριο την τιμή 70 km/h για να μπορεί να ακολουθήσει το όχημα την πορεία του χωρίς να φεύγει από αυτή και να μην αυξάνεται συνεχώς χωρίς όριο. Για μεγαλύτερες τιμές της ταχύτητας παρατηρήθηκαν φαινόμενα που το όχημα έφευγε από την πορεία του οπότε προτιμήθηκε να οριστεί ως ανώτατο όριο αύξησης της ταχύτητας τα 70 km/h. Η [εξίσωση 4.10](#) υπολογίζει τη νέα αυξημένη τιμή.

$$\begin{aligned}
 f(x) &= \text{interpolate } x \text{ from } [0, 10] \text{ to } [1.1, 2], \\
 \text{new_velocity} &= \text{parameter}_{\in[1.1,2]} \times \text{current_velocity}, \\
 \text{new_velocity} &= \min(70, \text{new_velocity})
 \end{aligned} \tag{4.10}$$

Περιγραφή Υπολογισμού Βέλτιστης Συμπεριφοράς

Το όχημα καθ' όλη τη διάρκεια της πορείας του καλείται να λάβει αποφάσεις σχετικά με την κίνηση του για την ασφαλή μεταφορά του στον τελικό προορισμό του και για την ικανοποίηση των απαιτήσεων που θέτει ο οδηγός μέσω των sliders για τη συμπεριφορά του αυτοκινήτου. Όπως σημειώθηκε και προηγουμένως οι αποφάσεις αυτές λαμβάνονται μέσω της μεθόδου MCDM και χρησιμοποιείται ο [πίνακας 4.1](#) για τον υπολογισμό της βέλτιστης συμπεριφοράς. Στην παράγραφο αυτή θα αναλυθεί ο τρόπος με τον οποίο υπολογίζεται η βέλτιστη συμπεριφορά μέσω του πίνακα αλλά και ο κύκλος αξιολόγησης για τη λήψη νέας απόφασης.

Για να κινηθεί το όχημα το πρόγραμμα χρειάζεται να τρέχει μέσα σε μια άπειρη επανάληψη στην οποία συνεχώς αξιολογεί το περιβάλλον, λαμβάνει αποφάσεις κίνησης και εφαρμόζει την κίνηση. Η επανάληψη αυτή τερματίζει όταν το όχημα φτάσει στον προορισμό του οπότε και πρέπει να τεθεί καινούργιος προορισμός από το χρήστη και όλη η διαδικασία να ξεκινήσει από την αρχή. Ένας μέρος αυτής της άπειρης επανάληψης είναι το κοιμάτι εκείνο που αξιολογεί τις συνθήκες που επικρατούν και επιλέγει κατάλληλη συμπεριφορά. Όπως φαίνεται και στον [πίνακα 4.1](#) κάθε κελί διαθέτει ένα συγκεκριμένο βάρος που υποδηλώνει τον βαθμό στον οποίο ο κανόνας i (γραμμή i) επηρεάζει τη συμπεριφορά j (στήλη j). Τα βάρη αυτά είναι συγκεκριμένα και προκαθορισμένα και δεν αλλάζουν αυτά καθ' αυτά κατά τη διάρκεια της επιλογής συμπεριφοράς. Ωστόσο, υπάρχουν κάποια φίλτρα σε μορφή

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

διανυσμάτων που λειτουργούν σαν συντελεστές και τα μεταβάλλουν προσωρινά ανάλογα με τις συνθήκες που επικρατούν. Τα διανύσματα αυτά θα εξηγηθούν στη συνέχεια:

- **Φίλτρο ενεργοποίησης κανόνων:** Το μέρος του προγράμματος που επιλέγει συμπεριφορά ξεκινά καλώντας μία προς μία τις συναρτήσεις που ενεργοποιούν τον κάθε κανόνα ξεχωριστά. Οι συναρτήσεις αυτές επιστρέφουν μία δυαδική τιμή *True* ή *False* την οποία αποθηκεύουν στην αντίστοιχη μεταβλητή. Μετά το πέρας αυτού του μέρους, υπάρχουν 27 μεταβλητές με δυαδικές τιμές που υποδηλώνουν ποιοι κανόνες έχουν ενεργοποιηθεί. Επομένως, το συνολικό σύστημα μετά την κλήση όλων των συναρτήσεων των κανόνων γνωρίζει ακριβώς τις συνθήκες που επικρατούν και είναι έτοιμο να προχωρήσει στη λήψη απόφασης. Οι κανόνες αυτοί οργανώνονται σε ένα διάνυσμα διάστασης 27×1 ίσο με τις γραμμές του [πίνακα 4.1](#) το οποίο ονομάζεται *activation_row_filter*. Οι ενεργοποιημένοι κανόνες συμπληρώνουν την τιμή 1 στην αντίστοιχη γραμμή του διανύσματος ενώ οι υπόλοιποι την τιμή 0. Το διάνυσμα έχει την παρακάτω μορφή:

$$activation_row_filter = \left[\begin{array}{l} 1 * R1 \wedge 1 * R2, \\ 1 * R2, \\ 1 * R3, \\ 1 * R4, \\ 1 * R5, \\ 1 * R6, \\ 1 * R7, \\ 1 * R8, \\ 1 * R9 \wedge (1 * R2 \vee 1 * R22), \\ 1 * R10 \wedge (1 * R2 \vee 1 * R23), \\ 1 * R11 \wedge 1 * R2 \wedge 1 * R10, \\ 1 * R12 \wedge 1 * R2, \\ 1 * R13 \wedge 1 * R2, \\ 1 * R14, \\ 1 * R15, \\ 1 * R16, \\ 1 * R17 \wedge 1 * R2, \\ 1 * R18 \wedge 1 * R2, \\ 1 * R19 \wedge 1 * R9, \\ 1 * R20 \wedge 1 * R2, \\ 1 * R21, \\ 1 * R22, \\ 1 * R23, \\ 1 * R24, \\ 1 * (\neg R9) \wedge (1 * R2 \vee 1 * R22), \\ 1 * (\neg R10) \wedge (1 * R2 \vee 1 * R23), \\ 1 * R27 \end{array} \right] \quad (4.11)$$

Το [διάνυσμα 4.11](#) λειτουργεί σαν φίλτρο στον [πίνακα 4.1](#) και συγκεκριμένα

4.4. ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΥΛΟΠΟΙΗΣΕΩΝ ΤΟΥ ΑΥΤΟΝΟΜΟΥ ΟΧΗΜΑΤΟΣ

πολλαπλασιάζεται με τις γραμμές του πίνακα με element-wise τρόπο δηλαδή όλα τα στοιχεία της γραμμής i πολλαπλασιάζονται με τον αριθμό 0 ή 1 της γραμμής i του [διανύσματος 4.11](#). Αυτό σημαίνει ότι όσοι κανόνες είναι ενεργοποιημένοι, δηλαδή έχουν τιμή 1 στην αντίστοιχη θέση του διανύσματος, τότε θα διατηρήσουν τα βάρη που έχουν στον πίνακα, ενώ όσοι κανόνες δεν έχουν ενεργοποιηθεί θα έχουν τιμή 0 και θα μηδενίσουν την αντίστοιχη γραμμή του πίνακα. Το διάνυσμα αυτό είναι το κυριότερο από τα φίλτρα διότι στην ουσία υποδηλώνει ποιες συνθήκες επικρατούν εκείνη τη στιγμή δηλαδή ποιοι κανόνες θα διατηρηθούν και ποιοι όχι.

Έπειτα θα εξηγηθεί η λογική με την οποία συμπληρώθηκε το activation_row_filter. Ένα κοινό στοιχείο που έχουν οι γραμμές είναι ότι οι κανόνες πολλαπλασιάζονται με την τιμή 1. Στην ουσία αυτό γίνεται για να μετατραπεί η λογική τιμή True ή False σε 1 ή 0 αντίστοιχα, για να πραγματοποιηθούν οι πολλαπλασιασμοί με τις γραμμές του πίνακα. Οι γραμμές του φίλτρου από ανεξάρτητους κανόνες που ενεργοποιούνται μόνοι τους και κανόνες εξαρτημένους που για να ενεργοποιηθούν χρειάζεται να είναι ενεργοποιημένοι άλλοι κανόνες. Συγκεκριμένα, οι γραμμές 2, 3, 4, 5, 6, 7, 8, 14, 15, 16, 21, 22, 23, 24 και 27 περιέχουν κανόνες ανεξάρτητους οι οποίοι ενεργοποιούνται, δηλαδή συμπληρώνουν την τιμή 0 ή 1 στη γραμμή τους, μόνο από την κλήση της συνάρτησης τους. Αυτό συμβαίνει διότι το σύστημα ενδιαφέρεται για την ενεργοποίηση τους σε κάθε επανάληψη ανεξάρτητα από τυχούσες άλλες συνθήκες. Οι γραμμές 1, 12, 13, 17, 18 και 20 περιέχουν κανόνες οι οποίοι εξαρτώνται από τον κανόνα R2. Αυτό γίνεται αντιληπτό από το λογικό AND που τους συνδέει. Ο κανόνας R2 είναι αυτός που ελέγχει αν υπάρχει μπροστινό κοντινό όχημα και θεωρείται από τους βασικότερους στην επιλογή συμπεριφοράς. Ο λόγος για τον οποίοι οι προαναφερθέντες κανόνες εξαρτώνται από τον R2 είναι γιατί δεν έχουν νόημα να εξεταστούν αν δεν είναι ενεργοποιημένος ο R2. Για παράδειγμα, ο κανόνας R1 που ελέγχει αν η ταχύτητα του μπροστινού οχήματος είναι μεγαλύτερη από την ταχύτητα του αυτόνομου οχήματος, δεν έχει πρακτικό νόημα να ελεγχθεί αν δεν υπάρχει μπροστινό όχημα. Εκτός αυτού, η λογική με την οποία κατασκευάστηκαν οι κανόνες ήταν ότι ο κύριος λόγος για να αλλάξει συμπεριφορά το όχημα και πιο συγκεκριμένα κατεύθυνση είναι να εμφανιστεί μπροστά του κάποιο όχημα. Σε διαφορετική περίπτωση δεν υπάρχει λόγος αν δεν επέμβει ο χρήστης να αλλάξει την κατεύθυνση του. Αυτός είναι και ο λόγος που οι κανόνες R12, R13, R17, R18 και R20, οι οποίοι ελέγχουν την κατάσταση των πλαινών λωρίδων, εξαρτώνται από τον R2. Με την ίδια λογική ενεργοποιούνται και οι κανόνες R9, R10, R11, R25 και R26 στους οποίους αλλάζουν οι κανόνες εξάρτησης. Όπως φαίνεται όλοι οι εξαρτημένοι κανόνες εξαρτώνται από τον κανόνα R2 για αυτό και αποτελεί τον βασικότερο κανόνα και αυτόν στον οποίο χτίστηκε η λογική των υπόλοιπων κανόνων.

- **Φίλτρο εφαρμογής aggressive τιμής:** Το φίλτρο αυτό είναι ένα διάνυσμα διαστάσεων 7×1 δηλαδή όσο και οι στήλες του πίνακα. Το διάνυσμα αυτό έχει τη μορφή της [εξίσωσης 4.13](#). Το διάνυσμα πολλαπλασιάζεται με element-wise τρόπο δηλαδή στοιχείο προς στοιχείο με τις στήλες του [πίνακα 4.1](#). Μετά τον

πολλαπλασιασμό τα βάρη κάθε στήλης αποκτούν μία νέα τιμή ανάλογα με την τιμή του κάθε στοιχείου του διανύσματος. Τα στοιχεία του διανύσματος αποτελούνται από δύο ειδών τιμές. Οι γραμμές 3,5,6,7 αποτελούνται από την τιμή 1. Αυτό σημαίνει ότι αυτές οι γραμμές παραμένουν ανεπηρέαστες μετά τον πολλαπλασιασμό. Οι γραμμές 1,2,4 πολλαπλασιάζονται με μία συγκεκριμένη τιμή. Η τιμή αυτή έχει προκύψει μετά από αντιστοίχιση του διαστήματος $[0, 10]$ στο διάστημα $[1, 2]$ μέσω γραμμικής παρεμβολής της [συνάρτησης 4.12](#)

$$\begin{aligned} f(x) &= \text{interpolate } x \text{ from } [0, 10] \text{ to } [1, 2], \\ \text{parameter}_{\in [1,2]} &= f(\text{aggressive_value}_{\in [0,10]}) \end{aligned} \quad (4.12)$$

Οι τιμές $[0, 10]$ αποτελούν το εύρος τιμών του slider "aggressive" και έχουν θετική συσχέτιση με το διάστημα $[1, 2]$. Αυτό σημαίνει ότι για μέγιστη τιμή του aggressive τα βάρη της αντίστοιχης στήλης διπλασιάζονται. Οι θέσεις που επιλέχθηκαν να τοποθετηθούν οι τιμές αυτές ανταποκρίνονται στις συμπεριφορές "LEFT LANE CHANGE", "RIGHT LANE CHANGE" και "SPEED UP" οι οποίες είναι και οι συμπεριφορές που χαρακτηρίζουν έναν επιθετικό οδηγό. Αυτό σημαίνει ότι όσο μεγαλύτερη η τιμή του aggressive τόσο περισσότερο ενισχύεται η επιλογή αυτών των συμπεριφορών λόγω του πολλαπλασιασμού των τιμών. Σε περίπτωση βέβαια που το aggressive έχει την ελάχιστη δυνατή τιμή τότε όλο το διάνυσμα έχει τιμή 1 που σημαίνει ότι τα βάρη παραμένουν ανεπηρέαστα. Αυτός είναι και ο τρόπος με τον οποίο επιδρά το slider "aggressive" πάνω στην επιλογή συμπεριφοράς.

$$\text{aggressive_column_filter} = \begin{bmatrix} \text{parameter}_{\in [1,2]}, \\ \text{parameter}_{\in [1,2]}, \\ 1, \\ \text{parameter}_{\in [1,2]}, \\ 1, \\ 1, \\ 1 \end{bmatrix} \quad (4.13)$$

- Φίλτρο εφαρμογής lawful τιμής:** Το φίλτρο αυτό αποτελεί ένα διάνυσμα διάστασης 27×1 δηλαδή ίσο με τη διάσταση των γραμμών του [πίνακα 4.1](#). Απεικονίζεται στην [εξίσωση 4.15](#) και πολλαπλασιάζεται με τις γραμμές του [πίνακα 4.1](#). Με αυτό τον τρόπο τα βάρη των γραμμών λαμβάνουν μία νέα τιμή μετά τη διαδικασία του πολλαπλασιασμού. Τα στοιχεία του διανύσματος [4.15](#) αποτελούνται είτε από τιμές ίσες με 1 και είτε από τιμές ίσες με μία συγκεκριμένη υπολογισμένη τιμή. Οι γραμμές που περιέχουν την τιμή 1 αφήνουν ανεπηρέαστα τα βάρη του πίνακα μετά τον πολλαπλασιασμό. Οι υπόλοιπες γραμμές λαμβάνουν μία τιμή η οποία προκύπτει όπως ακριβώς και στον προηγούμενο φίλτρο, δηλαδή από την αντιστοίχιση του εύρους $[0, 10]$ στο διάστημα $[1, 2]$ μέσω γραμμικής παρεμβολής χρησιμοποιώντας τη [εξίσωση 4.14](#). Το διάστημα $[0, 10]$ αποτελεί το εύρος τιμών του slider "lawful". Από την αντιστοίχιση προκύπτει ότι όταν ο χρήστης επιθυμεί να είναι απόλυτα νόμιμος τότε η βαρύτητα των συγκεκριμένων κανόνων διπλασιάζεται.

4.4. ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΥΛΟΠΟΙΗΣΕΩΝ ΤΟΥ ΑΥΤΟΝΟΜΟΥ ΟΧΗΜΑΤΟΣ

Με αυτό τον τρόπο το slider "lawful" επιδρά στη λήψη αποφάσεων και επηρεάζει τα βάρη του πίνακα. Η λογική στην οποία στηρίχθηκε είναι ότι πολλαπλασιάζοντας συγκεκριμένους κανόνες, δηλαδή γραμμές του πίνακα, με μία συγκεκριμένη τιμή που εξαρτάται από το slider "lawful" τότε οι κανόνες που επηρεάζουν τη νομιμότητα του οχήματος αποκτούν διαφορετική βαρύτητα.

$$f(x) = \text{interpolate } x \text{ from } [0, 10] \text{ to } [1, 2], \quad (4.14)$$

$$\text{parameter}_{\in [1,2]} = f(\text{lawful_value}_{\in [0,10]})$$

$$\text{lawful_row_filter} = \begin{bmatrix} 1, \\ 1, \\ 1, \\ 1, \\ 1, \\ 1, \\ \text{parameter}_{\in [1,2]}, \\ 1, \\ 1, \\ 1, \\ \text{parameter}_{\in [1,2]}, \\ 1, \\ 1, \\ 1, \\ \text{parameter}_{\in [1,2]}, \\ \text{parameter}_{\in [1,2]}, \\ 1, \\ 1, \\ \text{parameter}_{\in [1,2]}, \\ 1, \\ 1, \\ 1, \\ 1, \\ 1, \\ \text{parameter}_{\in [1,2]}, \end{bmatrix} \quad (4.15)$$

Οι κανόνες που πολλαπλασιάζονται με την παράμετρο που εξηγήθηκε παραπάνω δηλαδή επηρεάζονται από την τιμή του slider "lawful" είναι ο R7 δηλαδή ο κανόνας που ελέγχει αν βρίσκεται σε κόκκινο φανάρι το όχημα, ο R11 ο κανόνας που ελέγχει αν η αριστερή λωρίδα είναι ίδιας κατεύθυνσης, οι R17 και R18 δηλαδή οι κανόνες που ελέγχουν αν η δεξιά και η αριστερή λωρίδα αντίστοιχα έχουν διακεκομμένη διαγράμμιση, ο R21 που ελέγχει αν έχει ξεπεραστεί το όριο ταχύτητας και ο R27 που ελέγχει αν ο η διασταύρωση μπροστά από τη σήμανση STOP είναι ελεύθερη.

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

Συνολικά οι πράξεις που εφαρμόζονται πάνω στον αρχικό πίνακα φαίνονται στην [εξίσωση 4.16](#). Ο πίνακας $A_{(27 \times 7)}$ αποτελεί τον αρχικό πίνακα βαρών 4.1 και πολλαπλασιάζεται αρχικά με τον διάνυσμα $act_{(27 \times 1)}$ της [εξίσωσης 4.11](#) για να παραμείνουν στον πίνακα όσες γραμμές έχουν ενεργούς κανόνες. Έπειτα, ο παραγόμενος πίνακας με τις ενεργές γραμμές, πολλαπλασιάζεται με το διάνυσμα $law_{(27 \times 1)}$ 4.15 για να εφαρμοστεί η επίδραση του slider "lawful" στα βάρη του πίνακα. Στη συνέχεια, ο ανάστροφος του παραγόμενου πίνακα πολλαπλασιάζεται με το [διάνυσμα 4.13](#) για να εφαρμοστεί η τιμή του slider "aggressive" πάνω στον πίνακα. Τελικά, πραγματοποιείται ξανά μία αναστροφή του παραγόμενου πίνακα έτσι ώστε να προκύψει ο πίνακας W διαστάσεων 27×7 , όσο δηλαδή και ο αρχικός, από τον οποίο προκύπτει η βέλτιστη συμπεριφορά με τρόπο που θα εξηγηθεί στη συνέχεια. Να σημειωθεί ότι το σύμβολο $\cdot *$ υποδηλώνει τον "στοιχείο προς στοιχείο" πολλαπλασιασμό των πινάκων. Στην [εξίσωση 4.17](#) το $w^{(i)(j)}$ είναι το στοιχείο της γραμμής i και στήλης j του τελικού πίνακα και δείχνει τον τρόπο με τον οποίο λαμβάνει την τιμή του. Ο αρχικός πίνακας A παραμένει αναλλοίωτος κατά τους κύκλους αξιολόγησης δηλαδή τα βάρη του είναι σταθερά και οποιεσδήποτε αλλαγές εφαρμόζονται πάνω του, παράγουν τον καινούργιο πίνακα W .

$$W_{27 \times 7} = [(A_{(27 \times 7)} \cdot * \ act_{(27 \times 1)} \cdot * \ law_{(27 \times 1)})^T \cdot * \ agg_{(7 \times 1)}]^T \quad (4.16)$$

$$w^{(i)(j)} = A^{(i)(j)} \cdot act^{(i)} \cdot law^{(i)} \cdot agg^{(j)}, \quad i = 1, \dots, 27, \quad j = 1, \dots, 7 \quad (4.17)$$

Όπως αναφέρθηκε και παραπάνω το σύστημα αξιολογεί το περιβάλλον και τις συνθήκες που επικρατούν και λαμβάνει την απόφαση που μοιάζει βέλτιστη με βάση τις συνθήκες που επικρατούν εκείνη τη στιγμή. Ωστόσο, το σύστημα δε λαμβάνει αποφάσεις σε κάθε γύρο της άπειρης επανάληψης που αναφέρθηκε. Οι περίπτωσεις για να προβεί το σύστημα σε μία απόφαση και στην ουσία να αλλάξει τη συμπεριφορά του είναι 2 και αναφέρονται παρακάτω:

- 1. Αλλαγή του διανύσματος activation_row_filter:** Ο πρώτος λόγος για να κληθεί το όχημα να αξιολογήσει το περιβάλλον είναι να διαφέρει έστω και σε μία τιμή το διάνυσμα activation_row_filter από την κατάσταση του στην προηγούμενη επανάληψη. Το activation_row_filter όπως αναφέρθηκε περιέχει δυαδικές τιμές 0 ή 1 ανάλογα με την ενεργοποίηση ή όχι του αντίστοιχου κανόνα. Οπότε, σε περίπτωση που η τιμή του διανύσματος αυτού αλλάζει σε μια χρονική στιγμή τότε σημαίνει ότι οι συνθήκες του εξωτερικού περιβάλλοντος είναι διαφορετικές από αυτές που επικρατούσαν στην προηγούμενη επανάληψη οπότε και θα πρέπει να αξιολογηθεί εκ νέου η συμπεριφορά που ακολουθεί το όχημα.
- 2. Αλλαγή στις τιμές των sliders:** Σε κάθε γύρο της άπειρης επανάληψης ελέγχονται οι τιμές των sliders "aggressive" και "lawful" που έχει ορίσει ο χρήστης. Αν κάποια στιγμή τουλάχιστον μία από τις τιμές των sliders είναι διαφορετική σε σχέση με την προηγούμενη τιμή που έχει τότε το όχημα καλείται και πάλι

4.4. ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΥΛΟΠΟΙΗΣΕΩΝ ΤΟΥ ΑΥΤΟΝΟΜΟΥ ΟΧΗΜΑΤΟΣ

να επαναξιολογήσει το περιβάλλον και να λάβει καινούργια απόφαση για τη συμπεριφορά του. Αυτό συμβαίνει διότι, σε περίπτωση που οι τιμές αλλάξουν, ο χρήστης επιθυμεί το όχημα του να αλλάξει τη συμπεριφορά του οπότε και οδηγείται σε καινούργιο κύκλο αξιολόγησης.

Η λογική που περιγράφηκε για το έναυσμα του κύκλου αξιολόγησης συνοψίζεται στον [αλγόριθμο 4.14](#).

Αλγόριθμος 4.14 Check New Triggers

```

1: new_trigger ← False
2: if activation_row_filter(t) != activation_row_filter(t-1) then
3:   new_trigger ← True
4: end if
5: if aggressive(t) != aggressive(t-1) ∨ lawful(t) != lawful(t-1) then
6:   new_trigger ← True
7: end if
8: return new_trigger

```

Το επόμενο στάδιο, εφόσον ελεγχθεί αν ο συγκεκριμένος γύρος της άπειρης επανάληψης είναι γύρος στον οποίο θα πρέπει να πραγματοποιηθεί αξιολόγηση των συνθηκών, είναι να προχωρήσει το σύστημα στον υπολογισμό της βέλτιστης συμπεριφοράς. Για τον σκοπό αυτό υλοποιείται ο [αλγόριθμος 4.15](#) ο οποίος χρησιμοποιεί τον πίνακα W της [εξίσωσης 4.16](#). Όπως φαίνεται και στον αλγόριθμο υπολογίζονται τα αθροίσματα κατά στήλη όλων των συμπεριφορών. Εφόσον έχει εφαρμοστεί το φίλτρο activation_row_filter οι γραμμές που περιέχουν μη ενεργούς κανόνες θα είναι μηδενικές οπότε δεν επηρεάζουν το άθροισμα. Αφού προστεθούν οι στήλες αποθηκεύονται σε ένα διάνυσμα sum_values το οποίο περιέχει τα επτά αθροίσματα. Έπειτα, υπολογίζεται η μέγιστη τιμή των τριών πρώτων στοιχείων του διανύσματος και η μέγιστη τιμή των τεσσάρων τελευταίων. Αυτό συμβαίνει διότι τα τρία πρώτα αθροίσματα αντιπροσωπεύουν τις συμπεριφορές κατεύθυνσης οπότε πρέπει να επιλεχθεί η βέλτιστη, και τα τέσσερα τελευταία τις συμπεριφορές ταχύτητας. Από το κάθε σετ επιλέγεται η μεγαλύτερη τιμή οπότε μετά το πέρας αυτού του υπολογισμού το όχημα γνωρίζει ποια είναι η βέλτιστη πορεία που θα ακολουθήσει και πως θα προσαρμόσει βέλτιστα την ταχύτητα του στην πορεία αυτή. Σε περίπτωση που ένα από τα αθροίσματα ή και τα δύο είναι μηδέν τότε σημαίνει ότι δεν ενεργοποιείται κανένας κανόνας οπότε και επιλέγονται ως βέλτιστη συμπεριφορά τα "KEEP_STRAIGHT" και "KEEP_VELOCITY". Οι πράξεις που πραγματοποιούνται στον [αλγόριθμο 4.15](#) συνοψίζονται στη [εξίσωση 4.18](#).

$$\begin{aligned}
 & \text{sum_columns}^{(j)} = \sum_{i=1}^{27} W(i, j), \text{ where } j = 1, 2, \dots, 7 \text{ and } W \text{ is weight matrix} \\
 & \max_direction = \text{column's name of } \max(\text{sum_columns}[1 : 3]) \\
 & \max_speed = \text{column's name of } \max(\text{sum_columns}[4 : 7])
 \end{aligned} \tag{4.18}$$

Αφού επιλεχθεί η βέλτιστη συμπεριφορά από τη διαδικασία που περιγράφηκε παραπάνω, το σύστημα είναι έτοιμο να εφαρμόσει την επιλεγμένη συμπεριφορά.

Αλγόριθμος 4.15 Evaluate

Require: Matrix W with the calculated weights

```

1: if new_trigger is False then
2:     return
3: end if
4: sum_columns: vector with length equal to number of behaviors initialized with zeros
5: for j = 1 until columns' length do
6:     sum_columns[j] ← sum(W(1 : 27, j))
7: end for
8: optimal_direction_behavior_value ← max(sum_columns(1 : 3))
9: optimal_direction_behavior_index ← index(max(sum_columns(1 : 3)))
10: optimal_direction_behavior ← column's name with index optimal_direction_behavior_index
11: optimal_speed_behavior_value ← max(sum_columns(4 : 7))
12: optimal_speed_behavior_index ← index(max(sum_columns(4 : 7)))
13: optimal_speed_behavior ← column's name with index optimal_speed_behavior_index
14: if optimal_direction_behavior_value == 0 then
15:     optimal_direction_behavior ← "KEEP_STRAIGHT"
16: end if
17: if optimal_speed_behavior_value == 0 then
18:     optimal_speed_behavior ← "KEEP_VELOCITY"
19: end if
20: return optimal_direction_behavior, optimal_speed_behavior

```

Ο αλγόριθμος 4.16 υλοποιεί την εφαρμογή της κατάλληλης συμπεριφοράς. Όταν η βέλτιστη συμπεριφορά είναι η αλλαγή λωρίδας είτε προς τα δεξιά είτε προς τα αριστερά τότε μέσω της μεταβλητής *turn* ορίζεται η κατεύθυνση της αλλαγής, έπειτα ρυθμίζεται η ταχύτητα μέσω της συνάρτησης *regulate_speed* που θα αναλυθεί παρακάτω και καλείται ο αλγόριθμος αλλαγής λωρίδας 4.9. Προτού κληθεί ο αλγόριθμος αλλαγής λωρίδας ο δείκτης *current_index*, ο οποίος δείχνει ποιο waypoint του **τίναχα 4.1** πρόκειται να ακολουθήσει το όχημα, αυξάνεται κατά μία τιμή offset η οποία αναλύθηκε προηγουμένως και στην ουσία δείχνει ανάλογα με την ταχύτητα του οχήματος πόσα ενδιάμεσα waypoints θα παραληφθούν κατά την αλλαγή λωρίδας για να γίνει πιο ομαλή η μετάβαση. Στην περίπτωση που η βέλτιστη συμπεριφορά είναι η συνέχιση στην ίδια λωρίδα τότε το μόνο πράγμα που χρειάζεται να κάνει ο αλγόριθμος είναι να ρυθμίσει την ταχύτητα του. Η παραπάνω διαδικασία είναι αυτή που χρησιμοποιείται για να εφαρμοστεί η βέλτιστη συμπεριφορά κατεύθυνσης. Για την εφαρμογή της βέλτιστης συμπεριφοράς ταχύτητας χρησιμοποιείται η συνάρτηση *regulate_speed* η οποία αναλύεται παρακάτω.

Για να εφαρμοστεί η βέλτιστη συμπεριφορά κατεύθυνσης θα πρέπει να υπάρχει και η κατάλληλη ταχύτητα. Η ταχύτητα αυτή ορίζεται από την βέλτιστη συμπεριφορά ταχύτητας που έχει επιλεχθεί. Ο **αλγόριθμος 4.17** υλοποιεί την λογική με την οποία προσαρμόζεται η ταχύτητα με το βέλτιστο τρόπο και καλείται σε κάθε επιλογή συμπεριφοράς κατεύθυνσης. Για την περίπτωση που το όχημα καλείται να μειώσει την ταχύτητα του τότε υπολογίζεται η νέα μειωμένη ταχύτητα πολλαπλασιάζοντας την τρέχουσα ταχύτητα με ένα συντελεστή που ανήκει στο διάστημα

Αλγόριθμος 4.16 Apply Maneuver

Require: optimal_direction_behavior

```
1: if optimal_direction_behavior == "RIGHT_LANE_CHANGE" then
2:   if action_performed == False then
3:     turn ← "RIGHT"
4:     action_performed ← True
5:   end if
6:   Apply speed regulation by calling regulate_speed function
7:   current_index ← current_index + offset
8:   Apply lane change by calling change_lane function
9: end if
10: if optimal_direction_behavior == "LEFT_LANE_CHANGE" then
11:   if action_performed == False then
12:     turn ← "LEFT"
13:     action_performed ← True
14:   end if
15:   Apply speed regulation by calling regulate_speed function
16:   current_index ← current_index + offset
17:   Apply lane change by calling change_lane function
18: end if
19: if optimal_direction_behavior == "KEEP_STRAIGHT" then
20:   Apply speed regulation by calling regulate_speed function
21: end if
```

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

[0.2, 0.8] ανάλογα με την τιμή του slider "aggressive". Αναλυτικά ο τρόπος υπολογισμού περιγράφεται στην υποενότητα περιγραφής των παραμέτρων. Παρά την μείωση της ταχύτητας διατηρείται ένα κατώτατο όριο ίσο με 10 km/h διότι κάτω από αυτή την τιμή το όχημα κινείται πολύ αργά και δεν προσομοιώνεται σωστά η συμπεριφορά. Στην περίπτωση που υπάρχει όχημα μπροστά και η βέλτιστη συμπεριφορά κατεύθυνσης είναι η "KEEP_STRAIGHT" τότε η νέα μειωμένη ταχύτητα είναι η ίδια με την ταχύτητα του μπροστινού οχήματος προσομοιώνοντας έτσι την συμπεριφορά της ακολούθησης μπροστινού οχήματος (car following). Επίσης, αν κάποιος πεζός βρίσκεται σε πολύ κοντινή απόσταση τότε η νέα μειωμένη ταχύτητα είναι 3 km/h προσφέροντας έτσι μια πολύ χαμηλή ταχύτητα κίνησης για να αποφευχθεί το ατύχημα με τον πεζό. Αφού υπολογιστεί λοιπόν η μειωμένη ταχύτητα από τις περιπτώσεις που αναφέρθηκαν καλείται η συνάρτηση Slow Down 4.11 για να μεταβιβάσει την ταχύτητα στο σύστημα ελέγχου και να εφαρμόσει πρακτικά την επιβράδυνση του οχήματος. Αν η βέλτιστη συμπεριφορά ταχύτητας που καλείται να ακολουθήσει το όχημα είναι η επιτάχυνση, τότε αν η τρέχουσα ταχύτητα είναι μικρότερη από 1 km/h τότε η αυξημένη ταχύτητα γίνεται κατευθείαν 12 km/h για να επιταχύνει κατευθείαν το όχημα λόγω της πολύ χαμηλής ταχύτητας που είχε. Στη γενική περίπτωση που το όχημα έχει ταχύτητα μεγαλύτερη από 1 km/h τότε υπολογίζεται η νέα αυξημένη ταχύτητα πολλαπλασιάζοντας την τρέχουσα ταχύτητα με ένα συντελεστή που ανήκει στο διάστημα [1.1, 2] ανάλογα ξανά με την τιμή του slider "aggressive". Ο αναλυτικός τρόπος υπολογισμού βρίσκεται στην υποενότητα των παραμέτρων. Το ανώτατο όριο που διατηρείται είναι τα 70 km/h, διότι παραπάνω από αυτή την ταχύτητα το όχημα αδύνατεί να ακολουθήσει τα waypoints και χάνει την πορεία του λόγω μεγάλης ταχύτητας. Αφού υπολογιστεί η αυξημένη ταχύτητα καλείται η συνάρτηση Speed Up 4.10 για να μεταβιβαστεί η ταχύτητα στο σύστημα ελέγχου και να εφαρμοστεί η κατάλληλη επιτάχυνση. Για τις περιπτώσεις που οι βέλτιστες συμπεριφορές είναι οι STOP και KEEP_VELOCITY τότε απλώς καλούνται οι αλγόριθμοι 4.13 και 4.12 που έχουν ήδη περιγραφεί αναλυτικά προηγουμένως.

Αφού περιγράφηκε κάθε κομμάτι αυτής της άπειρης επανάληψης μέσα στην οποία τρέχει ο αλγόριθμος του οχήματος μέχρι να φτάσει στον τελικό προορισμό, θα παρουσιαστεί παρακάτω ο κεντρικό αλγόριθμος, δηλαδή αυτή η άπειρη επανάληψη, ο οποίος περιλαμβάνει όλα τα ξεχωριστά κομμάτια τα οποία αναφέρθηκαν και τρέχουν κατά τη διάρκεια κίνησης του οχήματος. Ο αλγόριθμος αυτός είναι ο γενικός μόνο κατά τη διάρκεια της κίνησης του οχήματος. Πριν ξεκινήσει το όχημα ο γενικός αλγόριθμος που τρέχει είναι ο αλγόριθμος καθορισμού τροχιάς 4.4 που αναλύθηκε προηγουμένως. Ο αλγόριθμος 4.18 ξεκινάει κάνοντας αρχικοποίηση τις σημαντικότερες μεταβλητές ανάμεσα στις οποίες είναι και ο αρχικός πίνακας βαρών πάνω στον οποίο εφαρμόζονται τα φίλτρα και επιλέγεται η βέλτιστη συμπεριφορά. Επίσης, αρχικοποιείται με την τιμή 0 ο δείκτης current_index ο οποίος όπως αναφέρθηκε δείχνει το waypoint του πίνακα waypoints προς το οποίο κινείται το όχημα. Έπειτα αρχίζει η άπειρη επανάληψη η οποία προσπελαύνει κάθε waypoint που προσεγγίζει το όχημα. Αρχικά, αφού υπολογιστεί η ταχύτητα του οχήματος μέσω του αλγόριθμο 4.8, ελέγχεται αν ο δείκτης current_index έχει φτάσει στο τελευταίο waypoint της τροχιάς οπότε και διακόπτεται η επανάληψη και το πρόγραμμα μεταφέρεται στον αλγόριθμο επιλογής προορισμού. Στη συνέχεια

Αλγόριθμος 4.17 Regulate Speed

Require: optimal_speed_behavior

```

1: if optimal_speed_behavior == "SLOW_DOWN" then
2:   if action_performed == False then
3:     speed ← current_velocity · (interpolated_aggressive_parameter ∈ [0.2, 0.8])
4:     speed ← max(10, speed)
5:     if vehicle_in_front ∧ optimal_direction == "KEEP_STRAIGHT" then
6:       speed ← front obstacle's velocity
7:     end if
8:     if pedestrian_closely then
9:       speed ← 3
10:    end if
11:    Call Slow Down function to apply the new velocity
12:    action_performed ← True
13:  end if
14: end if
15: if optimal_speed_behavior == "SPEED_UP" then
16:   if action_performed == False then
17:     if current_velocity > 1 then
18:       speed ← current_velocity · (interpolated_aggressive_param ∈ [1.1, 2])
19:       speed ← min(70, speed)
20:     else
21:       speed ← 12
22:     end if
23:     Call Speed Up function to apply the new velocity
24:     action_performed ← True
25:   end if
26: end if
27: if optimal_speed_behavior == "STOP" then
28:   Call Stop function in order to stop the vehicle instantly
29:   action_performed ← True
30: end if
31: if optimal_speed_behavior == "KEEP_VELOCITY" then
32:   Call Keep Velocity function in order to keep vehicle's speed as it is
33:   action_performed ← True
34: end if

```

ακολουθεί το τμήμα στο οποίο καθορίζονται οι παράμετροι του συστήματος σύμφωνα με τις τιμές των sliders όπως αναλύθηκε στην [υποενότητα 4.4.5](#). Το επόμενο τμήμα είναι το υποσύστημα αντίληψης το οποίο καλεί τους αλγορίθμους [4.6](#), [4.7](#), [4.5](#) και τη συνάρτηση εντοπισμού πεζών. Έπειτα ακολουθεί το τμήμα εκείνο το οποίο ελέγχει αν έχει πατηθεί κάποιο από τα κουμπιά με τα οποία μπορεί να επέμβει ο χρήστης στο όχημα κατά τη διάρκεια της κίνησης. Αν κάποιο από τα πλήκτρα έχει πατηθεί εφαρμόζεται και η κατάλληλη ενέργεια. Αφού ελεγχθεί το εξωτερικό περιβάλλον από το τμήμα αντίληψης και ελεγχθούν τα πλήκτρα ο αλγόριθμος περνάει στο στάδιο ελέγχου των κανόνων όπου καλούνται όλες οι συναρτήσεις των κανόνων, καθορίζονται τα φίλτρα και υπολογίζεται ο πίνακας βαρών ο οποίος θα αξιολογηθεί όπως περιγράφεται στην [υποενότητα 4.4.5](#). Έπειτα ξεκινάει το τμήμα επιλογής συμπεριφοράς ή τμήμα δράσης όπου ελέγχεται μέσω του [αλγόριθμο 4.14](#) αν υπάρχει καινούργιο ερέθισμα στο περιβάλλον που θα αλλάξει τη συμπεριφορά του οχήματος και σε περίπτωση που υπάρχει καλείται ο [αλγόριθμος 4.15](#) ο οποίος αξιολογεί το περιβάλλον και εξάγει τη βέλτιστη συμπεριφορά τη δεδομένη στιγμή. Αφού επιλεχθεί η βέλτιστη συμπεριφορά καλούνται οι αλγόριθμοι [4.16](#) και [4.17](#) για να εφαρμόσουν τις επιλεγμένες συμπεριφορές κίνησης και ταχύτητας. Το τελικό στάδιο του αλγορίθμου είναι το τμήμα ελέγχου το οποίο εφαρμόζει τις εντολές ελέγχου προκειμένου το όχημα να προσεγγίσει την επιθυμητή ταχύτητα και κατεύθυνση. Ο αλγόριθμος κλείνει ελέγχοντας την απόσταση της τρέχουσας τοποθεσίας του οχήματος από το επόμενο waypoint και σε περίπτωση που είναι μικρότερη από 2 μέτρα, που σημαίνει ότι έφτασε αρκετά κοντά στο waypoint, ο δείκτης των waypoints αυξάνεται κατά 1 για να κινηθεί το όχημα στο επόμενο waypoint. Η διαδικασία που περιγράφηκε εκτελείται σε όλη τη διάρκεια της διαδρομής μέχρι το όχημα να φτάσει τον τελικό του προορισμό.

4.4.6 Υποσύστημα Ελέγχου

Ένα από τα σημαντικότερα υποσυστήματα που λειτουργούν πάνω σε ένα αυτόνομο όχημα είναι το υποσύστημα ελέγχου. Το υποσύστημα αυτό είναι υπεύθυνο για να καθοδηγεί το αυτόνομο όχημα με όσο το δυνατόν καλύτερο τρόπο πάνω στη σχεδιασμένη τροχιά [13]. Ο έλεγχος του αυτοκινήτου θα πρέπει να εξασφαλίζει μία ασφαλή πορεία στο όχημα και να μπορεί να ανταποκρίνεται κάτω από δύσκολες συνθήκες οδήγησης όπως κεκλιμένα επίπεδα, λείες και τραχιές επιφάνειες όπως για παράδειγμα σε περίπτωση βροχής ή δύσβατου δρόμου αντίστοιχα. Όλα αυτά θα πρέπει να εξασφαλίζονται λαμβάνοντας υπόψη αρκετούς περιορισμούς που υπάρχουν κατά τη διαδικασία της οδήγησης όπως η δυναμική του οχήματος αλλά και περιορισμοί στα τεχνικά χαρακτηριστικά του όπως το σύστημα πέδησης, το σύστημα διεύθυνσης κλπ. Ταυτόχρονα με την όσο το δυνατόν καλύτερη ακολούθηση της σχεδιασμένης τροχιάς θα πρέπει να εξασφαλίζεται και μια σχετικά ομαλή κίνηση του οχήματος προσφέροντας στους επιβάτες μεγαλύτερη άνεση κατά την πορεία του οχήματος αλλά και προσομοιώνοντας όσο το δυνατόν καλύτερα την πορεία ενός χειροκίνητου οχήματος. Ουσιαστικά, το υποσύστημα ελέγχου είναι αυτό που θα στείλει τις εντολές στο σύστημα πέδησης, επιτάχυνσης και διεύθυνσης ελέγχοντας την ταχύτητα και την κατεύθυνση του οχήματος.

Στη συγκεκριμένη διπλωματική εργασία χρησιμοποιήθηκε ένας συνδυασμός δύο

Αλγόριθμος 4.18 Follow Trajectory

```

1: Initialize Weight Matrix A with the initial weights and the remaining variables
2: current_index ← 0
3: while True do
4:   current_velocity ← calculate_current_velocity()
5:   if current_index == length of waypoints array then
6:     Apply zero velocity to the control system in order to stop the vehicle
7:     Break the loop
8:   end if
9:   Publish Velocity to Interface's Speedometer
10:  # Behavior Definition Block
11:   Read sliders' values
12:   Calculate Behavior Parameters
13:   # Perception Block
14:   Call Vehicle Obstacles Manager
15:   Call Walker Obstacles Manager
16:   Call Traffic Sign and Traffic Light Manager
17:   Recognize obstacles and traffic signs and lights
18:   # Buttons Block
19:   Check if Stop button has been pressed
20:   Check if Lane Change button has been pressed
21:   Check if Change Goal button has been pressed
22:   if Stop button has been pressed then
23:     Apply Control to Stop the vehicle
24:   end if
25:   if Change Goal button has been pressed then
26:     Apply Control to Stop the vehicle
27:     Break the loop to specify new destination
28:   end if
29:   # Criteria Block
30:   Call each rule's function
31:   Specify row and column filter
32:   Calculate Weight Matrix
33:   # Action Block
34:   Check if new trigger exists
35:   if new trigger exists then
36:     Call Evaluate Function in order to choose optimal behavior
37:   end if
38:   Call Apply Maneuver Function to apply the optimal direction behavior
39:   Call Regulate Speed Function to apply the optimal speed behavior
40:   # Control Block
41:   Apply Control Commands to reach the specified velocity
42:   distance ← distance between vehicle's location and next waypoint
43:   if distance < 2 then
44:     current_index ← current_index + 1
45:   end if
46: end while

```

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

PID ελεγκτών για να ελεγχθεί η ταχύτητα και η γωνία στροφής του αυτοκινήτου αντίστοιχα. Ο PID ελεγκτής⁶² είναι ένας από τους πιο ευρέως χρησιμοποιούμενους ελεγκτές και εύκολους στη χρήση για αυτό και επιλέχθηκε. Στην ουσία είναι ένας αλγόριθμος ο οποίος υπολογίζει μία τιμή προερχόμενη από τον υπολογισμό ενός σφάλματος. Το σφάλμα στην προκειμένη περίπτωση είναι η διαφορά της πραγματικής τροχιάς που θέλει να ακολουθήσει το όχημα και της θέσης στην οποία βρίσκεται αυτή τη στιγμή. Ένας PID ελεγκτής αποτελείται από τρία στοιχεία:

- **P (Proportional):** Ο όρος P μεταφράζεται ως αναλογικός και σημαίνει ότι συνεισφέρει αναλογικά στο σφάλμα που υπάρχει. Το σφάλμα αυτό μπορεί να τροποποιηθεί χρησιμοποιώντας τον συντελεστή K_P που ονομάζεται αναλογικό κέρδος και πολλαπλασιάζεται με το σφάλμα. Το κέρδος K_P στην ουσία ρυθμίζει την ταλάντωση που είναι επιθυμητή στο σύστημα με τις μεγαλύτερες τιμές να προσφέρουν και εκτενέστερη ταλάντωση. Όταν το σύστημα είναι ένα σύστημα διεύθυνσης ενός αυτόνομου οχήματος όπως στη συγκεκριμένη περίπτωση ο αναλογικός όρος εφαρμόζει μια αναλογική διόρθωση στη στροφή του τιμονιού αναλογική με το σφάλμα που υπάρχει.
- **D (Derivative):** Ο όρος D ονομάζεται διαφορικός και επιδρά στο σφάλμα με τον παρακάτω τρόπο. Ύπολογίζεται η κλίση της συνάρτησης του σφάλματος με το χρόνο και το αποτέλεσμα που προκύπτει πολλαπλασιάζεται με το διαφορικό κέρδος K_D. Στην ουσία ο διαφορικός όρος επιδρά στην αλλαγή του σφάλματος και προσφέρει μεγαλύτερη σταθερότητα στο σύστημα. Μειώνει επίσης την ταλάντωση που δημιουργεί ο αναλογικός όρος και για την περίπτωση του αυτόνομου οχήματος μεταβάλλει γρηγορότερα τη γωνία στροφής για να γίνει πιο ομαλή η κίνηση του οχήματος.
- **I (Integral):** Ο όρος I ονομάζεται ολοκληρωτικός και επιδρά στο σφάλμα με τη λογική που περιγράφεται. Ύπολογίζεται το άθροισμα του σφάλματος μέσα σε ένα συγκεκριμένο χρονικό διάστημα το οποίο αντιπροσωπεύει το συσσωρευμένο σφάλμα που θα έπρεπε να έχει διορθωθεί μέσα στη διάρκεια του χρόνου. Η διαδικασία αυτή προφανώς ονομάζεται ολοκλήρωση και το συσσωρευμένο σφάλμα που υπολογίστηκε πολλαπλασιάζεται με το ολοκληρωτικό κέρδος K_I προσθέτοντας το αποτέλεσμα στη συνολική έξοδο του ελεγκτή. Στην ουσία ο ολοκληρωτικός όρος υπάρχει για να διορθώνει τυχόν σφάλμα που έχει συσσωρευτεί από τις συχνές αλλαγές κατεύθυνσης.

Μετά τους υπολογισμούς που αναφέρθηκαν ότι γίνονται η έξοδοι των τριών ελεγκτών προστίθενται όπως φαίνεται και στο σχήμα 4.30 και δίνονται ως είσοδος στο ελεγχόμενο σύστημα δηλαδή στη συγκεκριμένη περίπτωση στο σύστημα διεύθυνσης και ταχύτητας του οχήματος. Η έξοδος του ελεγχόμενου συστήματος ανατροφοδοτείται και πάλι στην είσοδο του συστήματος και υπολογίζεται το σφάλμα μεταξύ της αληθινής τιμής εξόδου και της τιμής αναφοράς που θέλει να πετύχει το σύστημα. Το σφάλμα αυτό τροφοδοτείται στους τρεις όρους και η διαδικασία ακολουθεί ξανά τον τρόπο υπολογισμού που περιγράφηκε. Στο σχήμα 4.31 φαίνεται και η διαγραμματική μορφή ενός PID ελεγκτή.

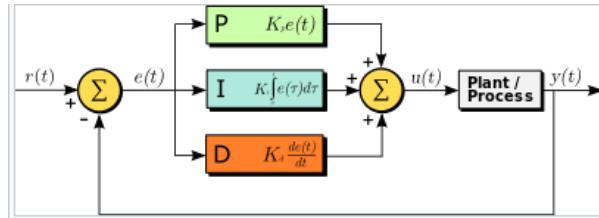
⁶²https://en.wikipedia.org/wiki/PID_controller

4.4. ΠΕΡΙΓΡΑΦΗ ΤΩΝ ΥΛΟΠΟΙΗΣΕΩΝ ΤΟΥ ΑΥΤΟΝΟΜΟΥ ΟΧΗΜΑΤΟΣ

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt}$$

Σχήμα 4.30: Έξοδος PID ελεγκτή.

Πηγή: https://en.wikipedia.org/wiki/PID_controller



Σχήμα 4.31: Διάγραμμα ενός συστήματος με PID ελεγκτή.

Πηγή: https://en.wikipedia.org/wiki/PID_controller

Το API του CARLA παρέχει έτοιμες υλοποιήσεις για τους PID ελεγκτές οι οποίες χρησιμοποιούνται και εφαρμόστηκαν σε αυτή την εργασία ⁶³. Στη συγκεκριμένη διπλωματική το πρόβλημα του ελέγχου χωρίστηκε σε δύο μέρη τα οποία περιγράφονται παρακάτω:

- **Longitudinal control:** Το πρώτο σύστημα ελέγχου αναφέρεται ως Longitudinal control και αφορά την προσαρμογή της ταχύτητας του οχήματος σε μια ταχύτητα αναφοράς. Ο ελεγκτής που χρησιμοποιήθηκε είναι ένας PID ελεγκτής. Η είσοδος του ελεγκτή είναι η ταχύτητα αναφοράς δηλαδή αυτή που επιθυμεί να φτάσει το όχημα και μετά τους υπολογισμούς ανατροφοδοτείται με την εκάστοτε ταχύτητα του οχήματος για να υπολογιστεί το αντίστοιχο σφάλμα. Η έξοδος του ελεγκτή είναι το ποσοστό γκαζιού ή φρένου που θα εφαρμοστεί την εκάστοτε χρονική στιγμή. Μετά από πειράματα και διαδικασία δοκιμής σφάλματος προέκυψαν οι τιμές που φαίνονται στον πίνακα 4.3.

K _p	K _d	K _i
1.0	0	0.05

Πίνακας 4.3: Τα τελικά κέρδη του Longitudinal ελεγκτή.

- **Lateral control:** Το δεύτερο κοιμάτι του ελέγχου αφορά την κατεύθυνση του οχήματος και τον βαθμό στον οποίο πρέπει να στρίψουν οι ρόδες του οχήματος για να ακολουθήσει με τον καλύτερο δυνατό τρόπο η σχεδιασμένη τροχιά. Το συγκεκριμένο σύστημα ελέγχου ονομάζεται Lateral control και χρησιμοποιήθηκε και σε αυτή την περίπτωση ένας PID ελεγκτής. Η είσοδος που δίνεται σε αυτή την περίπτωση στον ελεγκτή είναι το waypoint της τροχιάς που καλείται το όχημα εκείνη τη στιγμή να ακολουθήσει και αποτελεί στην ουσία την είσοδο αναφοράς. Επίσης, ανατροφοδοτείται στο σύστημα η εκάστοτε θέση του οχήματος και το σφάλμα μεταξύ αυτής και της θέσης αναφοράς δίνεται

⁶³<https://github.com/carla-simulator/carla/blob/master/PythonAPI/carla/agents/navigation/controller.py>

ΚΕΦΑΛΑΙΟ 4. ΥΛΟΠΟΙΗΣΕΙΣ

ως είσοδος στον ελεγκτή. Η έξοδος του συστήματος είναι μια εντολή με το ποσοστό που πρέπει να στρίψει το σύστημα διεύθυνσης του οχήματος για να προσεγγίσει την επιθυμητή θέση. Οι τιμές των κερδών που προέκυψαν στον lateral ελεγκτή φαίνονται στον [πίνακα 4.4](#).

Kp	Kd	Ki
1.0	0	0.07

Πίνακας 4.4: Τα τελικά κέρδη του Lateral ελεγκτή.

5

Πειράματα - Αποτελέσματα

5.1 ΕΙΣΑΓΩΓΗ

Στην παρούσα ενότητα θα παρουσιαστεί η αξιολόγηση του συστήματος αυτόνομης οδήγησης και η δυνατότητα του να ανταποκρίνεται σωστά στις διάφορες συνθήκες κίνησης. Η αξιολόγηση γίνεται κατά κύριο λόγο ως προς τη δυνατότητα του οχήματος να ολοκληρώνει τη διαδρομή που του έχει ανατεθεί, να αποφεύγει τυχούσες συγκρούσεις με στατικά ή δυναμικά εμπόδια κατά τη διάρκεια της διαδρομής αυτής και να ανταποκρίνεται σωστά στους κανόνες οδικής κυκλοφορίας. Εκτός αυτών βέβαια το σύστημα αξιολογείται ως προς τον τρόπο με τον οποίο κινείται και λαμβάνει αποφάσεις για διάφορες τιμές των δυο sliders. Τα πειράματα πραγματοποιούνται για συγκεκριμένες διαδρομές για να ελεγχθεί και η εγκυρότητα της σχεδίασης τροχιάς μέσω του NodeRED με βάση τις επιθυμίες του χρήστη. Η πόλη στην οποία εκτελούνται τα πειράματα είναι η πόλη 3 του CARLA για διαφορετικές τιμές των aggressive και lawful και των δυναμικών εμποδίων. Στο τέλος, σχολιάζονται τα τελικά συμπεράσματα που προκύπτουν από τα αποτελέσματα των πειραμάτων.

5.2 ΜΕΤΡΙΚΕΣ ΑΞΙΟΛΟΓΗΣΗΣ

Οι μετρικές αξιολόγησης οι οποίες χρησιμοποιήθηκαν για να μελετήσουν την ποιότητα και την απόδοση του αυτόνομου οχήματος έχουν σχέση με τον αριθμό και το είδος των παραβάσεων που πραγματοποίησε το όχημα, με τη δυνατότητα ολοκλήρωσης της επιθυμητής διαδρομής και το ποσοστό παραμονής εντός διαδρομής του οχήματος σε όλη τη διάρκεια της πορείας. Οι παραβάσεις που μπορεί να πραγματοποιήσει το όχημα έχουν διαφορετικό βαθμό επικινδυνότητας καθώς συμπεριλαμβάνουν συγκρούσεις με πεζούς και οχήματα δηλαδή δυναμικά εμπόδια, συγκρούσεις με στατικά εμπόδια όπως τα πεζοδρόμια ή κάποιο μέρος εκτός ή

εντός του δρόμου, παραβάσεις στο όριο ταχύτητας του δρόμου, παραβάσεις φωτεινού σηματοδότηση και παραβάσεις σήμανσης stop. Η δυνατότητα ολοκλήρωσης της διαδρομής αναφέρεται στο ποσοστό της συνολικής διαδρομής που ολοκλήρωσε το όχημα. Το ποσοστό παραμονής εντός διαδρομής καθορίζει το ποσοστό του χρόνου στο οποίο το όχημα κινήθηκε εντός της προβλεπόμενης διαδρομής, χωρίς να έχει μεγάλες αποκλίσεις από την τροχιά, σε σχέση με τον συνολικό χρόνο που χρειάστηκε για να την ολοκληρώσει. Τα δύο αυτά ποσοστά υπολογίζονται κατά μέσω όρο για N εκτελούμενες διαδρομές. Οι προαναφερόμενες μετρικές λήφθηκαν από το CARLA Leaderboard⁶⁴ και υπολογίζονται με τον παρακάτω τρόπο:

- **Βαθμολογία οδήγησης (Driving Score):** Αποτελεί την κύρια μετρική για την αξιολόγηση του συστήματος διότι εκφράζεται ως το άθροισμα της μέσης ολοκλήρωσης της διαδρομής και του συνολικού αριθμού παραβάσεων της κυκλοφορίας. Ο αριθμός είναι το σύνολο των διαδρομών στις οποίες δοκιμάστηκε το όχημα, R_i είναι το ποσοστό ολοκλήρωσης της i -οστής διαδρομής και P_i η ποινή παραβάσεων της i -οστής διαδρομής. Η συγκεκριμένη μετρική δίνεται από τη [εξίσωση 5.1](#).

$$\text{Driving Score} = \frac{1}{N} \cdot \sum_i^N (R_i \cdot P_i) \quad (5.1)$$

- **Ολοκλήρωση Διαδρομής (Route Completion):** Η μετρική αυτή αποτελεί το ποσοστό της συνολικής διαδρομής που καλύφθηκε από το αυτόνομο όχημα υπολογισμένο κατά μέσο όρο για N διαδρομές. Υπολογίζεται από την [εξίσωση 5.2](#)

$$\text{Route Completion} = \frac{1}{N} \cdot \sum_i^N R_i \quad (5.2)$$

- **Ποινή Παραβάσεων (Infraction Penalty):** Η συγκεκριμένη μετρική συγκεντρώνει το συνολικό αριθμό παραβάσεων που πραγματοποιούνται από το αυτόνομο όχημα, ως γεωμετρική σειρά. Το αυτόνομο όχημα ξεκινάει με ιδινή τιμή τη 1.0, η οποία μειώνεται για κάθε παράβαση καθώς πολλαπλασιάζεται με τους συντελεστές παράβασης. Οι συντελεστές παράβασης δίνονται από τον [πίνακα 5.1](#) και η [εξίσωση 5.3](#) υπολογίζει την ποινή παραβάσεων. Η ποινή παράβασης P_i της i -οστής διαδρομής που αναφέρθηκε στην [εξίσωση 5.1](#) υπολογίζεται από το εσωτερικό του αθροίσματος της [εξίσωσης 5.3](#) και συγκεκριμένα δίνεται από την [εξίσωση 5.4](#).

$$\text{Infraction Penalty} = \frac{1}{N} \cdot \sum_i^N \prod_j^{ped,...,stop} (p_i^j)^{\text{infractions}_j} \quad (5.3)$$

$$P_i = \prod_j^{ped,...,stop} (p_i^j)^{\text{infractions}_j} \quad (5.4)$$

⁶⁴<https://leaderboard.carla.org/>

ΚΕΦΑΛΑΙΟ 5. ΠΕΙΡΑΜΑΤΑ - ΑΠΟΤΕΛΕΣΜΑΤΑ

Παράβαση	Τιμή Παράβασης
Σύγκρουση με πεζό	0.5
Σύγκρουση με όχημα	0.6
Σύγκρουση με στατικό εμπόδιο	0.65
Παραβίαση κόκκινου φωτεινού σηματοδότη	0.7
Παραβίαση σηματοδότη STOP	0.8
Παραβίαση ορίου ταχύτητας	0.9
Ποσοστό χρόνου εκτός δρόμου	Αριθμός ποσοστού

Πίνακας 5.1: Πίνακας με τις τιμές των παραβάσεων. Οι μικρότερες τιμές αντιστοιχούν και σε μεγαλύτερη παράβαση.

Οι μετρικές που περιγράφηκαν παραπάνω και χρησιμοποιούνται για την αξιολόγηση του αυτόνομου οχήματος της συγκεκριμένης διπλωματικής αποτελούν τις επίσημες μετρικές που χρησιμοποιούνται για τον διαγωνισμό CARLA Autonomous Driving Challenge και χρησιμοποιήθηκαν στην παρούσα εργασία για να υπάρχει μια ευρωστία σχετικά με τις μετρήσεις των πειραμάτων. Ωστόσο, επειδή το κύριο αντικείμενο μελέτης της εργασίας εκτός από την αυτόνομη πλοήγηση του οχήματος ήταν και η επίδραση των sliders aggressive και lawful στη συμπεριφορά του οχήματος χρησιμοποιήθηκαν εκτός από τις γενικές μετρικές και κάποιες επιπλέον απλούστερες μετρήσεις που αποτύπωναν την επίδραση αυτή. Συγκεκριμένα, στα πειράματα μετρήθηκαν επιπλέον:

- Αριθμός δεξιών και αριστερών αλλαγών λωρίδας.
- Η μέση ταχύτητα του οχήματος κατά τη διάρκεια της διαδρομής.

Οι μετρήσεις αυτές επιλέχθηκαν για να αποτυπωθεί ο βαθμός στον οποίο επηρεάζει το slider aggressive την ταχύτητα του οχήματος αλλά και τις αλλαγές κατεύθυνσης. Η επίδραση του slider lawful αποτυπώνεται στη μετρική "Ποινή παραβάσεων" καθώς το συγκεκριμένο slider επηρεάζει τον βαθμό νομιμότητας με τον οποίο κινείται το όχημα.

5.3 ΣΥΝΘΗΚΕΣ ΠΕΙΡΑΜΑΤΩΝ

Στην ενότητα αυτή θα γίνει μια γενική περιγραφή των πειραμάτων που εκτελέστηκαν και θα αναλυθούν οι συνθήκες κάτω από τις οποίες πραγματοποιήθηκαν, οι παραδοχές που έγιναν κατά την εκτέλεση αλλά και οι διαδρομές που ακολουθήθηκαν.

Αρχικά, η πόλη στην οποία έτρεξαν τα πειράματα ήταν η πόλη 3 του CARLA. Πρόκειται για την πιο περίπλοκη πόλη του CARLA η οποία περιλαμβάνει διασταύρωση 5 λωρίδων, κυκλικό κόμβο, μία σήραγγα, ανομοιογένεια στους δρόμους, αρκετές διασταυρώσεις τύπου 'T' και πολλές ακόμη οδικές δυσκολίες. Περιέχει φυσικά και μια μεγάλη ποικιλία σεναρίων με τα οποία έρχεται αντιμέτωπο το όχημα όπως οι διαβάσεις τις οποίες καλείται να ελέγξει, φωτεινούς σηματοδότες, αυτοκινητόδρομους, διασταυρώσεις υποχρεωτικής διακοπής της κίνησης και μονόδρομους.

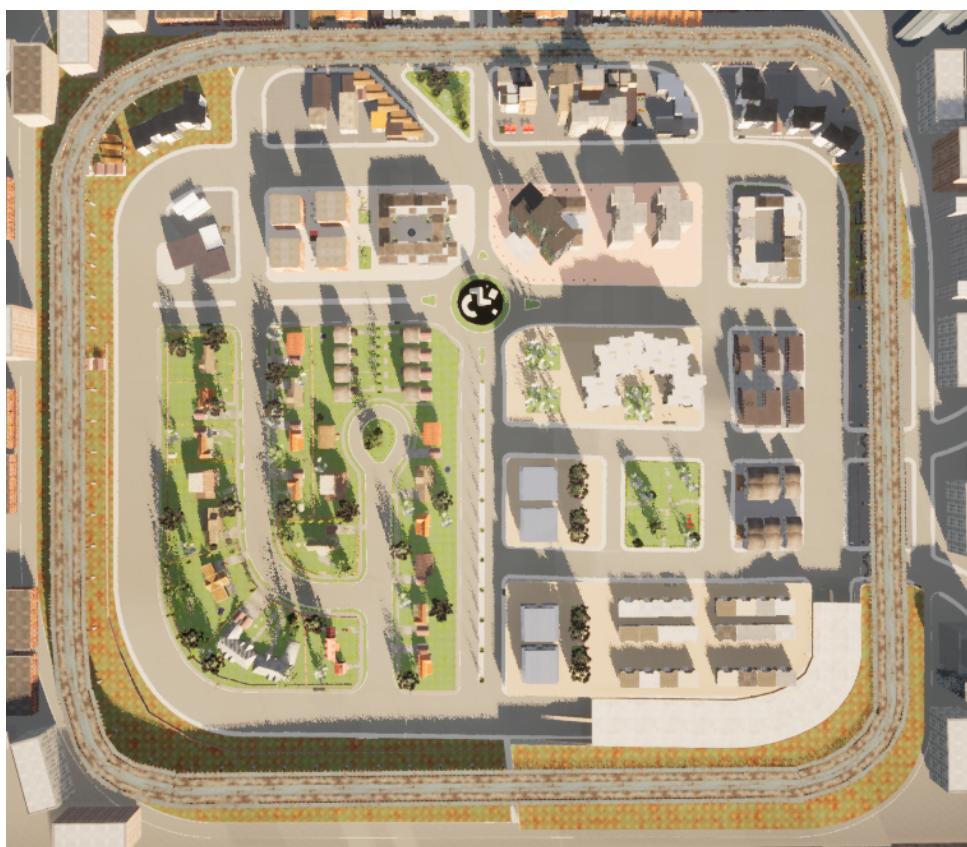
δημιουργώντας έτσι ένα αρκετά ρεαλιστικό περιβάλλον κίνησης. Αυτός ήταν και ο λόγος που επιλέχθηκε η συγκεκριμένη πόλη καθώς προσφέρει τη μεγαλύτερη δυσκολία κατά τη διαδικασία κίνησης. Η τοπολογία της πόλης φαίνεται στο [σχήμα 5.1](#) και η κάτοψη μαζί με τις λεπτομέρειες της στο [σχήμα 5.2](#).



Σχήμα 5.1: Η τοπολογία της πόλης στην οποία εκτελέστηκαν τα πειράματα.

Τα πειράματα εκτελέστηκαν για διαφορετικό αριθμό δυναμικών εμποδίων. Συγκεκριμένα, ο αριθμός των δυναμικών εμποδίων σε κάθε διεξαγωγή πειράματος αυξάνεται για να αυξάνεται έτσι και η κυκλοφοριακή συμφόρηση στους δρόμους. Με αυτό τον τρόπο το όχημα καλείται να λαμβάνει γρηγορότερα αποφάσεις λόγω της ταχείας αλλαγής των συνθηκών του περιβάλλοντος. Οι συνδυασμοί δυναμικών εμποδίων για τους οποίους δοκιμάστηκε το σύστημα είναι:

1. Κανένα όχημα και κανένας πεζός
2. Με 10 οχήματα και 10 πεζούς
3. Με 30 οχήματα και 30 πεζούς
4. Με 60 οχήματα και 60 πεζούς
5. Με 90 οχήματα και 90 πεζούς



Σχήμα 5.2: Η κάτοψη της πόλης με τις λεπτομέρειες της.

Επιπλέον, οι τιμές των δύο sliders αποτελούν την κύρια μεταβλητή των πειραμάτων και στην ουσία μελετάται η επίδραση της μεταβολής τους στην πλοήγηση του οχήματος και στη λήψη των αποφάσεων. Για αυτό το λόγο τα πειράματα εκτελέστηκαν για διαφορετικές τιμές των δύο sliders. Οι τιμές των sliders που επιλέχθηκαν είναι 0, 5 και 10 για το κάθε slider και ο λόγος που επιλέχθηκαν οι συγκεκριμένες είναι για να μελετηθεί η επίδραση στις ακραίες και μεσαίες τιμές τους. Εκτελέστηκε ένα πείραμα για κάθε συνδυασμό τους, δηλαδή συνολικά 9 διαφορετικά περιπτώσεις. Κάθε μία από τις 9 διαφορετικές περιπτώσεις εκτελέστηκε για κάθε διαφορετικό αριθμό δυναμικών εμποδίων και για κάθε διαφορετική διαδρομή στην πόλη. Συνολικά, το όχημα με τους 9 συνδυασμούς τιμών των δύο sliders, έτρεξε σε 2 διαφορετικές διαδρομές του 1 km η καθεμιά στην πόλη 3. Οι 18 αυτές περιπτώσεις δοκιμάστηκαν για 0, 10, 30, 60, 90 δυναμικά εμπόδια που σημαίνει ότι εκτελέστηκαν 90 διαφορετικά πειράματα. Εφόσον το κάθε πείραμα ακολουθούσε διαδρομή 1 km τότε συνεπάγεται ότι το αυτόνομο όχημα κάλυψε 90 km. Το όχημα έλαβε διαφορετικά ερεθίσματα από το περιβάλλον στις συγκεκριμένες διαδρομές διότι τα δυναμικά εμπόδια δημιουργούνταν τυχαία επάνω στο χάρτη και κλήθηκε να πάρει διαφορετικές αποφάσεις εξαρτώμενες από τις τιμές των sliders. Οι διαδρομές σχεδιάστηκαν έτσι ώστε το όχημα να συναντάει σημάνσεις STOP, φωτεινούς σηματοδότες και διασταυρώσεις χωρίς κάποιο σήμα έτσι ώστε να έρχεται αντιμέτωπο με πολλά διαφορετικά σενάρια κίνησης.

5.4 ΑΠΟΤΕΛΕΣΜΑΤΑ ΠΕΙΡΑΜΑΤΩΝ

Πίνακες Αποτελεσμάτων

Αφού ορίστηκαν οι συνθήκες με τις οποίες εκτελέστηκαν τα πειράματα στη συνέχεια παρουσιάζονται σε πίνακες για κάθε αριθμό δυναμικών εμποδίων από αυτούς που αναφέρθηκαν παραπάνω τα αποτελέσματα από την πλοϊγηση του αυτόνομου οχήματος μέσα στην πόλη. Για κάθε αριθμό δυναμικών εμποδίων παρουσιάζονται τρεις πίνακες.

- Ο πίνακας με τις βασικές μετρικές του Carla Leaderboard για τους διαφορετικούς συνδυασμούς των τιμών των sliders.
- Ο πίνακας με τις μετρήσεις ταχύτητας και αλλαγών λωρίδας για τους διαφορετικούς συνδυασμούς των τιμών των sliders.
- Ο πίνακας με τους αριθμούς κάθε παράβασης για τους διαφορετικούς συνδυασμούς των τιμών των sliders.

[H] Αποτελέσματα για 2 διαδρομές στην πόλη χωρίς οχήματα και πεζούς

Aggressive	Lawful	Βαθμολογία Οδήγησης	Ολοκλήρωση διαδρομής	Ποινή παραβάσεων
0	0	0	0	0
0	5	0	0	0
0	10	0	0	0
5	0	0	0	0
5	5	0	0	0
5	10	0	0	0
10	0	0	0	0
10	5	0	0	0
10	10	0	0	0
Σύνολο		0	0	0

Πίνακας 5.2: Μετρικές αξιολόγησης αυτονομίας οδήγησης για διαφορετικές τιμές των δύο sliders και κανένα δυναμικό εμπόδιο.

ΚΕΦΑΛΑΙΟ 5. ΠΕΙΡΑΜΑΤΑ - ΑΠΟΤΕΛΕΣΜΑΤΑ

[H]

Aggressive	Lawful	Δεξιές αλλαγές	Αριστερές αλλαγές	Μέση Ταχύτητα
0	0	0	0	0
0	5	0	0	0
0	10	0	0	0
5	0	0	0	0
5	5	0	0	0
5	10	0	0	0
10	0	0	0	0
10	5	0	0	0
10	10	0	0	0
Σύνολο		0	0	0

Πίνακας 5.3: Μετρικές αξιολόγησης της ταχύτητας και του αριθμού αλλαγών λωρίδας για διαφορετικές τιμές των δύο sliders και κανένα δυναμικό εμπόδιο.

Aggressive	Lawful	Σύγκρουση με πεζό	Σύγκρουση με όχημα	Σύγκρουση με στατικό εμπόδιο	Παραβίαση φωτεινού σηματοδότη	Παραβίαση stop	Παραβίαση ορίου ταχύτητας	Ποσοστό χρόνου εντός δρόμου
0	0	0	0	0	0	0	0	0
0	5	0	0	0	0	0	0	0
0	10	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
5	5	0	0	0	0	0	0	0
5	10	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0
10	5	0	0	0	0	0	0	0
10	10	0	0	0	0	0	0	0
Σύνολο		0	0	0	0	0	0	0

Πίνακας 5.4: Μετρήσεις των αριθμών κάθε παράβασης για διαφορετικές τιμές των δύο sliders και κανένα δυναμικό εμπόδιο.

Συγκριτικά Διαγράμματα Αποτελεσμάτων

Σχολιασμός Αποτελεσμάτων

Αποτελέσματα για 2 διαδρομές στην πόλη με 10 οχήματα και 10 πεζούς

Aggressive	Lawful	Βαθμολογία Οδήγησης	Ολοκλήρωση διαδρομής	Ποινή παραβάσεων
0	0	0	0	0
0	5	0	0	0
0	10	0	0	0
5	0	0	0	0
5	5	0	0	0
5	10	0	0	0
10	0	0	0	0
10	5	0	0	0
10	10	0	0	0
Σύνολο		0	0	0

Πίνακας 5.5: Μετρικές αξιολόγησης αυτονομίας οδήγησης για 2 διαφορετικές τιμές των δύο sliders με 10 οχήματα και 10 πεζούς.

Aggressive	Lawful	Δεξιές αλλαγές	Αριστερές αλλαγές	Μέση Ταχύτητα
0	0	0	0	0
0	5	0	0	0
0	10	0	0	0
5	0	0	0	0
5	5	0	0	0
5	10	0	0	0
10	0	0	0	0
10	5	0	0	0
10	10	0	0	0
Σύνολο		0	0	0

Πίνακας 5.6: Μετρικές αξιολόγησης της ταχύτητας και του αριθμού αλλαγών λωρίδας για 2 διαφορετικές τιμές των δύο sliders με 10 οχήματα και 10 πεζούς.

ΚΕΦΑΛΑΙΟ 5. ΠΕΙΡΑΜΑΤΑ - ΑΠΟΤΕΛΕΣΜΑΤΑ

Aggressive	Lawful	Σύγκρουση με πεζό	Σύγκρουση με όχημα	Σύγκρουση με στατικό εμπόδιο	Παραβίαση φωτεινού σηματοδότη	Παραβίαση stop	Παραβίαση ορίου ταχύτητας	Ποσοστό χρόνου εντός δρόμου
0	0	0	0	0	0	0	0	0
0	5	0	0	0	0	0	0	0
0	10	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
5	5	0	0	0	0	0	0	0
5	10	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0
10	5	0	0	0	0	0	0	0
10	10	0	0	0	0	0	0	0
Σύνολο		0	0	0	0	0	0	0

Πίνακας 5.7: Μετρήσεις των αριθμών κάθε παράβασης για διαφορετικές τιμές των δύο sliders με 10 οχήματα και 10 πεζούς.

Αποτελέσματα για 2 διαδρομές στην πόλη με 30 οχήματα και 30 πεζούς

Aggressive	Lawful	Βαθμολογία Οδήγησης	Ολοκλήρωση διαδρομής	Ποινή παραβάσεων
0	0	0	0	0
0	5	0	0	0
0	10	0	0	0
5	0	0	0	0
5	5	0	0	0
5	10	0	0	0
10	0	0	0	0
10	5	0	0	0
10	10	0	0	0
Σύνολο		0	0	0

Πίνακας 5.8: Μετρικές αξιολόγησης αυτονομίας οδήγησης για διαφορετικές τιμές των δύο sliders με 30 οχήματα και 30 πεζούς.

5.4. ΑΠΟΤΕΛΕΣΜΑΤΑ ΠΕΙΡΑΜΑΤΩΝ

Aggressive	Lawful	Δεξιές αλλαγές	Αριστερές αλλαγές	Μέση Ταχύτητα
0	0	0	0	0
0	5	0	0	0
0	10	0	0	0
5	0	0	0	0
5	5	0	0	0
5	10	0	0	0
10	0	0	0	0
10	5	0	0	0
10	10	0	0	0
Σύνολο		0	0	0

Πίνακας 5.9: Μετρικές αξιολόγησης της ταχύτητας και του αριθμού αλλαγών λωρίδας για διαφορετικές τιμές των δύο sliders με 30 οχήματα και 30 πεζούς.

Aggressive	Lawful	Σύγκρουση με πεζό	Σύγκρουση με όχημα	Σύγκρουση με στατικό εμπόδιο	Παραβίαση φωτεινού σηματοδότη	Παραβίαση stop	Παραβίαση ορίου ταχύτητας	Ποσοστό χρόνου εντός δρόμου
0	0	0	0	0	0	0	0	0
0	5	0	0	0	0	0	0	0
0	10	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
5	5	0	0	0	0	0	0	0
5	10	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0
10	5	0	0	0	0	0	0	0
10	10	0	0	0	0	0	0	0
Σύνολο		0	0	0	0	0	0	0

Πίνακας 5.10: Μετρήσεις των αριθμών κάθε παράβασης για διαφορετικές τιμές των δύο sliders με 30 οχήματα και 30 πεζούς.

Αποτελέσματα για 2 διαδρομές στην πόλη με 60 οχήματα και 60 πεζούς

Aggressive	Lawful	Βαθμολογία Οδήγησης	Ολοκλήρωση διαδρομής	Ποινή παραβάσεων
0	0	0	0	0
0	5	0	0	0
0	10	0	0	0
5	0	0	0	0
5	5	0	0	0
5	10	0	0	0
10	0	0	0	0
10	5	0	0	0
10	10	0	0	0
Σύνολο		0	0	0

Πίνακας 5.11: Μετρικές αξιολόγησης αυτονομίας οδήγησης για 2 διαφορετικές τιμές των δύο sliders με 60 οχήματα και 60 πεζούς.

Aggressive	Lawful	Δεξιές αλλαγές	Αριστερές αλλαγές	Μέση Ταχύτητα
0	0	0	0	0
0	5	0	0	0
0	10	0	0	0
5	0	0	0	0
5	5	0	0	0
5	10	0	0	0
10	0	0	0	0
10	5	0	0	0
10	10	0	0	0
Σύνολο		0	0	0

Πίνακας 5.12: Μετρικές αξιολόγησης της ταχύτητας και του αριθμού αλλαγών λωρίδας για 2 διαφορετικές τιμές των δύο sliders με 60 οχήματα και 60 πεζούς.

5.4. ΑΠΟΤΕΛΕΣΜΑΤΑ ΠΕΙΡΑΜΑΤΩΝ

Aggressive	Lawful	Σύγκρουση με πεζό	Σύγκρουση με οχημα	Σύγκρουση με στατικό εμπόδιο	Παραβίαση φωτεινού σηματοδότη	Παραβίαση stop	Παραβίαση ορίου ταχύτητας	Ποσοστό χρόνου εντός δρόμου
0	0	0	0	0	0	0	0	0
0	5	0	0	0	0	0	0	0
0	10	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
5	5	0	0	0	0	0	0	0
5	10	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0
10	5	0	0	0	0	0	0	0
10	10	0	0	0	0	0	0	0
Σύνολο		0	0	0	0	0	0	0

Πίνακας 5.13: Μετρήσεις των αριθμών κάθε παράβασης για διαφορετικές τιμές των δύο sliders με 60 οχήματα και 60 πεζούς.

Αποτελέσματα για 2 διαδρομές στην πόλη με 90 οχήματα και 90 πεζούς

Aggressive	Lawful	Βαθμολογία Οδήγησης	Ολοκλήρωση διαδρομής	Ποινή παραβάσεων
0	0	0	0	0
0	5	0	0	0
0	10	0	0	0
5	0	0	0	0
5	5	0	0	0
5	10	0	0	0
10	0	0	0	0
10	5	0	0	0
10	10	0	0	0
Σύνολο		0	0	0

Πίνακας 5.14: Μετρικές αξιολόγησης αυτονομίας οδήγησης για διαφορετικές τιμές των δύο sliders με 90 οχήματα και 90 πεζούς.

ΚΕΦΑΛΑΙΟ 5. ΠΕΙΡΑΜΑΤΑ - ΑΠΟΤΕΛΕΣΜΑΤΑ

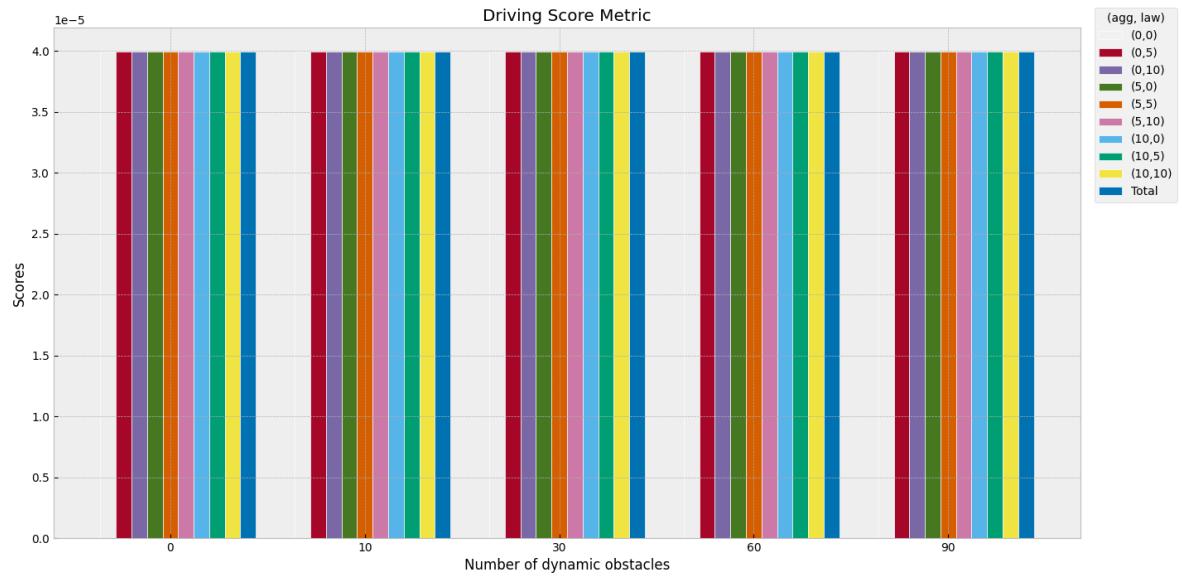
Aggressive	Lawful	Δεξιές αλλαγές	Αριστερές αλλαγές	Μέση Ταχύτητα
0	0	0	0	0
0	5	0	0	0
0	10	0	0	0
5	0	0	0	0
5	5	0	0	0
5	10	0	0	0
10	0	0	0	0
10	5	0	0	0
10	10	0	0	0
Σύνολο		0	0	0

Πίνακας 5.15: Μετρικές αξιολόγησης της ταχύτητας και του αριθμού αλλαγών λωρίδας για διαφορετικές τιμές των δύο sliders με 90 οχήματα και 90 πεζούς.

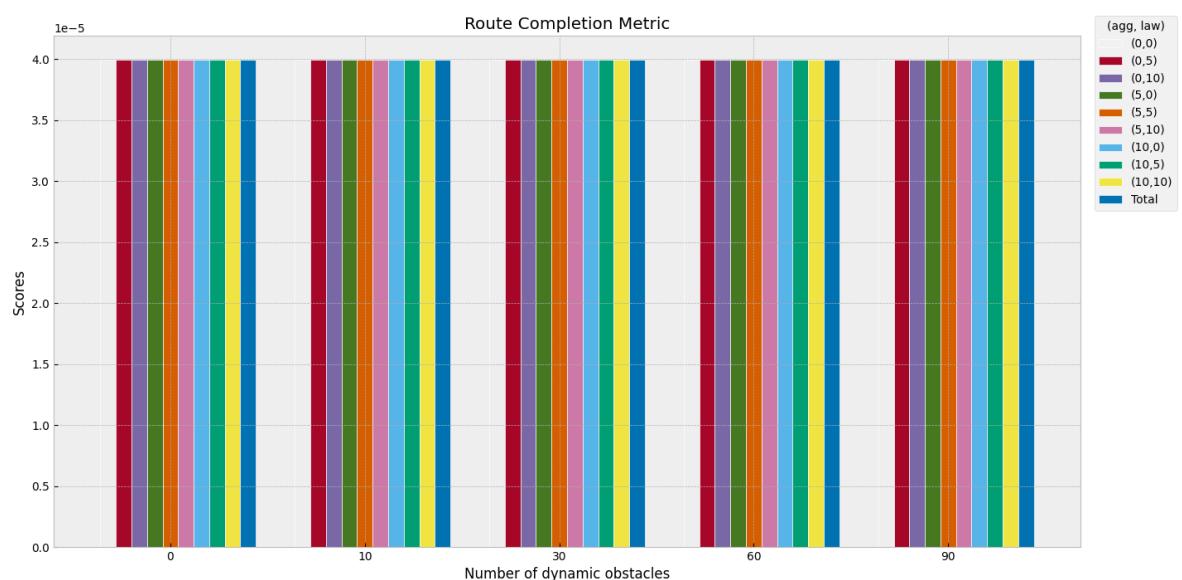
Aggressive	Lawful	Σύγκρουση με πεζό	Σύγκρουση με όχημα	Σύγκρουση με στατικό εμπόδιο	Παραβίαση φωτεινού σηματοδότη	Παραβίαση stop	Παραβίαση ορίου ταχύτητας	Ποσοστό χρόνου εντός δρόμου
0	0	0	0	0	0	0	0	0
0	5	0	0	0	0	0	0	0
0	10	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
5	5	0	0	0	0	0	0	0
5	10	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0
10	5	0	0	0	0	0	0	0
10	10	0	0	0	0	0	0	0
Σύνολο		0	0	0	0	0	0	0

Πίνακας 5.16: Μετρήσεις των αριθμών κάθε παράβασης για διαφορετικές τιμές των δύο sliders με 90 οχήματα και 90 πεζούς.

5.4. ΑΠΟΤΕΛΕΣΜΑΤΑ ΠΕΙΡΑΜΑΤΩΝ

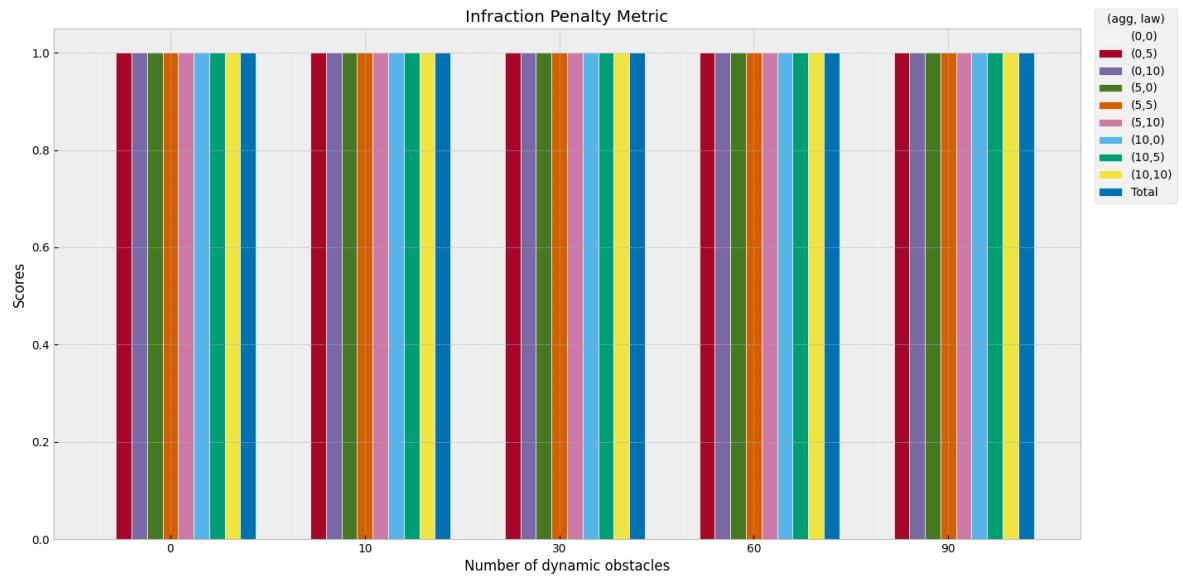


Σχήμα 5.3: Ραβδόγραμμα για τη μετρική "Βαθμολογία Οδήγησης".

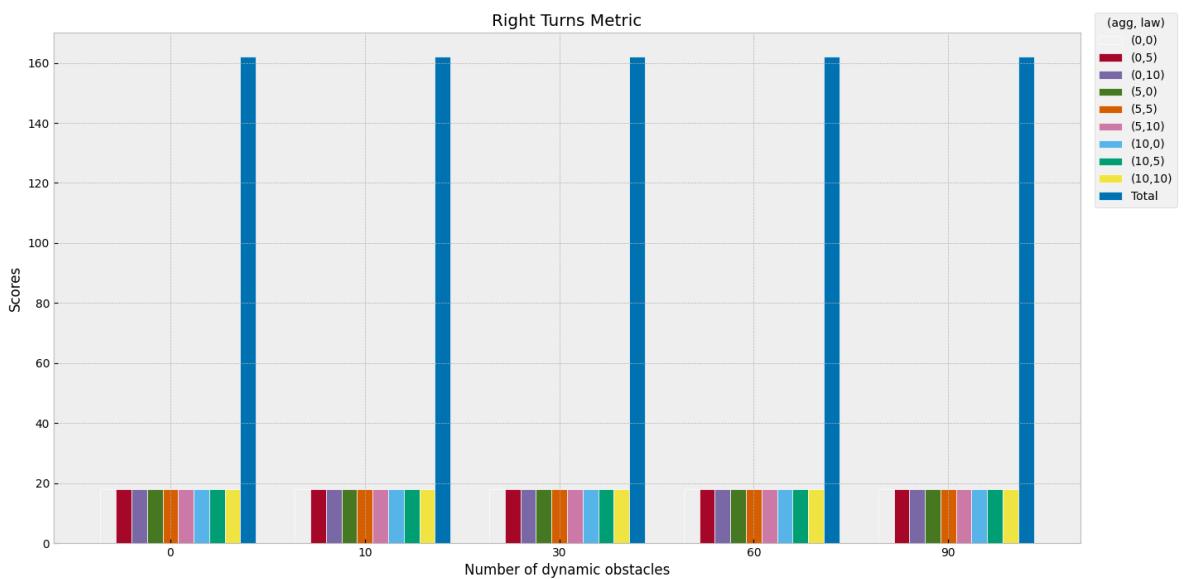


Σχήμα 5.4: Ραβδόγραμμα για τη μετρική "Ολοκλήρωση Διαδρομής".

ΚΕΦΑΛΑΙΟ 5. ΠΕΙΡΑΜΑΤΑ - ΑΠΟΤΕΛΕΣΜΑΤΑ

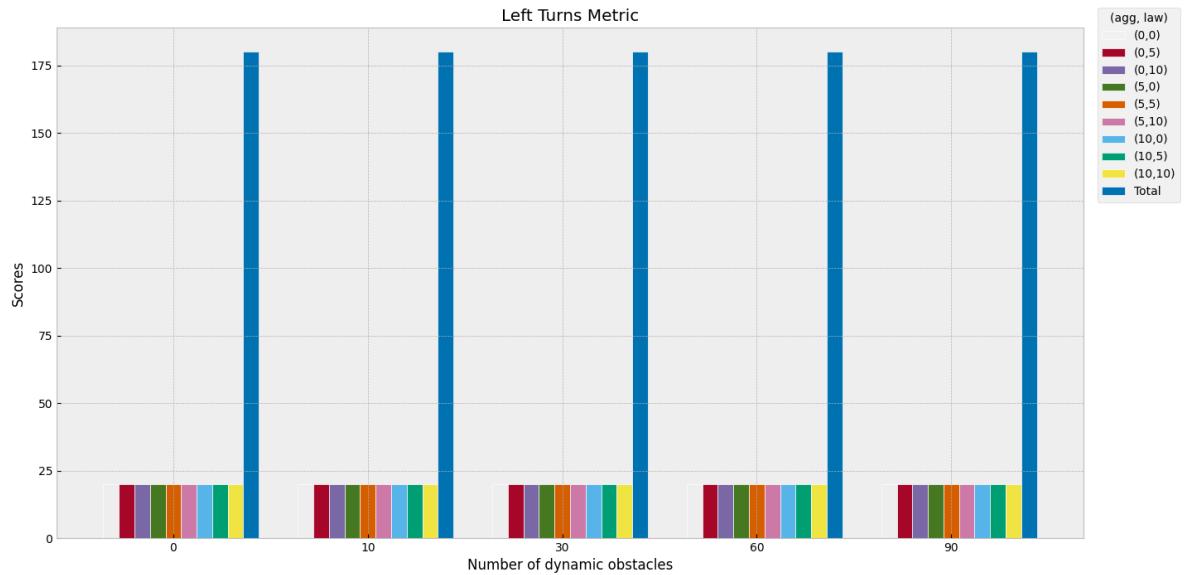


Σχήμα 5.5: Ραβδόγραμμα για τη μετρική "Ποινή Παραβάσεων". Μεγαλύτερες τιμές στο διάγραμμα υποδηλώνουν λιγότερες παραβάσεις άρα και καλύτερο αποτέλεσμα αυτονομίας.

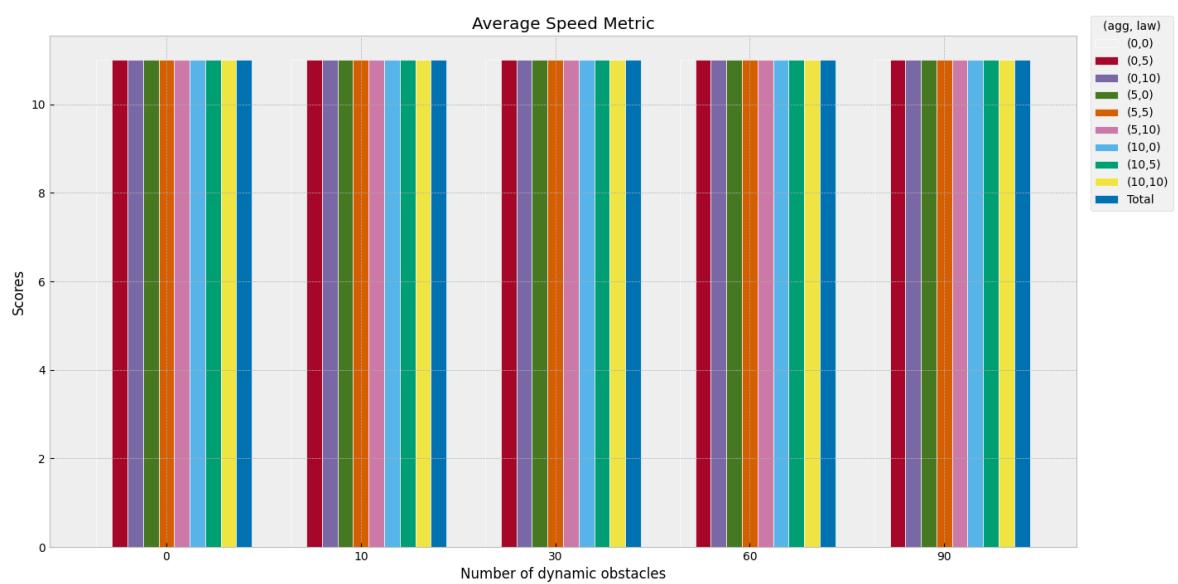


Σχήμα 5.6: Ραβδόγραμμα για τη μετρική "Αριθμός δεξιών αλλαγών λωρίδας".

5.4. ΑΠΟΤΕΛΕΣΜΑΤΑ ΠΕΙΡΑΜΑΤΩΝ



Σχήμα 5.7: Ραβδόγραμμα για τη μετρική "Αριθμός αριστερών αλλαγών λωρίδας".



Σχήμα 5.8: Ραβδόγραμμα για τη μετρική "Μέση Ταχύτητα".

6

Συμπεράσματα

Στο κεφάλαιο αυτό παρουσιάζονται η υλοποίηση του συνολικού συστήματος αυτόνομης οδήγησης και της διεπαφής αλλά και τα γενικά συμπεράσματα που προκύπτουν από την λειτουργία του συστήματος και την εκτέλεση των πειραμάτων. Οι μετρικές αξιολόγησης του συστήματος είχαν σχέση με την ολοκλήρωση της διαδρομής που κινείται το όχημα, με τον αριθμό παραβάσεων που πραγματοποιεί κατά την πορεία του όπως παραβάσεις σε φωτεινούς σηματοδότες και άλλες σημάνσεις, συγκρούσεις με στατικά και δυναμικά εμπόδια και μη τήρηση των κανόνων κυκλοφορίας. Επίσης, χρησιμοποιήθηκαν μετρικές που έδειχναν την μέση ταχύτητα κατά τη διάρκεια των διαδρομών και τον αριθμό αλλαγών λωρίδας διότι με αυτές τις μετρικές μελετήθηκε η επίδραση των δύο sliders πάνω στο όχημα. Στη συνέχεια παρουσιάζονται τα προβλήματα που εμφανίστηκαν σε όλη τη διάρκεια εκπόνησης της διπλωματικής κατά την υλοποίηση του συστήματος αλλά και κατά την εκτέλεση των πειραμάτων. Το κεφάλαιο καταλήγει με τις μελλοντικές επεκτάσεις και βελτιώσεις που θα μπορούσαν να εφαρμοστούν στη διπλωματική εργασία.

6.1 ΓΕΝΙΚΑ ΣΥΜΠΕΡΑΣΜΑΤΑ

6.2 ΠΡΟΒΛΗΜΑΤΑ

Στην υποενότητα αυτή θα αναλυθούν τα προβλήματα που εμφανίστηκαν κατά τη διάρκεια εκπόνησης της εργασίας. Αρχικά, σε ό,τι αφορά τον κώδικα και τη λειτουργικότητα του ένα σημαντικό πρόβλημα εμφανίστηκε στον πίνακα βαρών της μεθόδου MCDM διότι ρυθμίστηκαν έπειτα από αρκετές εκτελέσεις διαφορετικών σεναρίων κίνησης. Δεν υπήρχε δηλαδή κάποια αυτοματοποιημένη διαδικασία που να ρυθμίζει τα συγκεκριμένα βάρη. Με τη χειροκίνητη αλλαγή των βαρών πολλές αλλαγές ερχόταν σε σύγκρουση μεταξύ των διαφορετικών σεναρίων με αποτέλεσμα να είναι μια διαδικασία αρκετά χρονοβόρα. Το ίδιο ίσχυε και στη ρύθμιση των

6.3. ΜΕΛΛΟΝΤΙΚΕΣ ΕΠΕΚΤΑΣΕΙΣ

παραμέτρων του PID ελεγκτή χρησιμοποιώντας μια εκτενή διαδικασία δοκιμής και λάθους για την σωστή προσέγγιση των βαρών.

Η μεγαλύτερη, ωστόσο, δυσκολία που συναντήθηκε σε όλη τη διάρκεια της εργασίας ήταν οι υψηλές υπολογιστικές απαιτήσεις του προσομοιωτή. Η συνολική υλοποίηση του προγράμματος του αυτόνομου οχήματος αλλά και τα αποτελέσματα που λήφθηκαν στα πειράματα έτρεξαν σε σύστημα που διαθέτει επεξεργαστή Intel® Core™ i5-5200U CPU @ 2.20GHz × 4 με 2 πυρήνες και 4 νήματα, κάρτα γραφικών GeForce 920M με 2 GB εσωτερική μνήμη και μνήμη RAM 4 GB. Όπως γίνεται αντιληπτό τα χαρακτηριστικά του συστήματος ήταν κάτω από τα προτεινόμενα του CARLA γεγονός που δυσκόλευε αρκετά την εκτέλεση σεναρίων με πολλά οχήματα επάνω στο χάρτη. Τα σενάρια κίνησης με τα περισσότερα οχήματα οδηγούσαν το σύστημα στη χρησιμοποίηση όλων των πόρων και να τρέχει σε χαμηλότερα FPS. Λόγω των χαμηλών υπολογιστικών πόρων που διέθετε το σύστημα αποκλείστηκαν στην αρχή της υλοποίησης και άλλα λογισμικά όπως η χρήση του Autoware το οποίο προσφέρει λύσεις στα προβλήματα του perception, του localization, του planning κλπ. αλλά λόγω δυσκολίας εγκατάστασης και εκτέλεσης του προγράμματος η χρησιμοποίηση του δεν προχώρησε. Επίσης, στην αρχή της διπλωματικής η λύση του προβλήματος της αντίληψης ξεκίνησε με την έρευνα Deep Learning τεχνικών όπως η αναγνώριση πεζών, οχημάτων και σημάτων μέσω κάμερας αλλά και σε αυτή την περίπτωση η λύση απορρίφθηκε λόγω των περιορισμένων υπολογιστικών πόρων.

Τέλος, κάποια προβλήματα εμφανίστηκαν στην αρχή με την επικοινωνία του αυτόνομου οχήματος με τη διεπαφή. Το πρωτόκολλο επικοινωνίας που επιλέχθηκε στην αρχή ήταν το AMQP με χρήση του μεσολαβητή RabbitMQ αλλά παρόλο που υπάρχουν πακέτα στο NodeRED που υποστηρίζουν την επικοινωνία μέσω AMQP εν τέλει στην πράξη φάνηκε ότι είχαν λειτουργικά προβλήματα για αυτό και επιλέχθηκε το πρωτόκολλο MQTT. Ο μεσολαβητής RabbitMQ υποστηρίζει το πρωτόκολλο MQTT αλλά για διευκόλυνση της διαδικασίας χρησιμοποιήθηκε τελικά ο Mosquitto.

6.3 ΜΕΛΛΟΝΤΙΚΕΣ ΕΠΕΚΤΑΣΕΙΣ

Στην παρούσα υπενότητα αναφέρονται κάποιες από τις μελλοντικές επεκτάσεις και βελτιώσεις που θα μπορούσαν να εφαρμοστούν στην παρούσα εργασία. Οι επεκτάσεις αυτές αφορούν τόσο τη λειτουργία και τους αλγορίθμους υλοποίησης του αυτόνομου οχήματος, δηλαδή το κομμάτι που υλοποιήθηκε στον CARLA, όσο και τη λειτουργία της διεπαφής του χρήστη που υλοποιήθηκε στο NodeRED.

Η σημαντικότερη από τις βελτιώσεις που μπορεί να δεχθεί το αυτόνομο όχημα είναι η υλοποίηση του συστήματος αντίληψης. Στην παρούσα διπλωματική οποιαδήποτε πληροφορία χρειάστηκε το όχημα από το εξωτερικό περιβάλλον του όπως οι θέσεις και οι ταχύτητες άλλων οχημάτων, το είδος και η τοποθεσία των σημάνσεων, οι θέσεις των πεζών λήφθηκαν απευθείας από τον προσομοιωτή CARLA. Ένα βελτιωμένο σύστημα αντίληψης μπορεί να χρησιμοποιήσει τους αισθητήρες που παρέχει ο CARLA οι οποίοι στην ουσία αποτελούν αληθινές συσκευές και να λαμβάνει ερεθίσματα από το περιβάλλον του μέσω αυτών αντί να παρέχονται έτοιμα από τον προσομοιωτή. Σε ένα τέτοιο σύστημα το οποίο μεταξύ των υπολοίπων αισθητήρων

ΚΕΦΑΛΑΙΟ 6. ΣΥΜΠΕΡΑΣΜΑΤΑ

διαθέτει και κάμερα μπορούν να υλοποιηθούν αλγόριθμοι τεχνητής νοημοσύνης οι οποίοι θα αναγνωρίζουν τα εμπόδια και τα πιθανά σήματα που βρίσκονται γύρω από το όχημα. Ένα σύστημα αντίληψης υλοποιημένο με αυτή την προσέγγιση είναι πιο ρεαλιστικό και θα μπορούσε να εφαρμοστεί με τις κατάλληλες προσαρμογές και σε αληθινό όχημα.

Μια ακόμη βελτίωση θα μπορούσε να υπάρξει στο σύστημα κίνησης του οχήματος. Ο τρόπος με τον οποίο κινείται τοπικά το όχημα όπως αναλύθηκε προηγουμένως είναι διανύοντας ένα ένα τα waypoints της τροχιάς και αγνοώντας κάποια από αυτά σε περίπτωση αλλαγής λωρίδας για να δημιουργείται έτσι μια πιο ομαλή τροχιά. Ένα βελτιωμένο σύστημα θα διέθετε ένα σύστημα τοπικών μονοπατιών το οποίο σε κάθε βήμα κίνησης του οχήματος θα παρήγαγε ένα σύνολο από πιθανές καμπύλες κίνησης από τις οποίες θα επέλεγε τη βέλτιστη εκείνη τη χρονική στιγμή. Επιπλέον, σε ό,τι αφορά το κομμάτι της κίνησης του οχήματος θα μπορούσε να βελτιωθεί και ο ελεγκτής του συστήματος είτε ρυθμίζοντας τα βάρη του PID ελεγκτή με κάποια τεχνική βελτιστοποίησης της ρύθμισης των βαρών προσφέροντας την ομαλότερη δυνατή κίνηση είτε χρησιμοποιώντας ένας εντελώς διαφορετικό ελεγκτή όπως ο MPC.

Επίσης, στη συγκεκριμένη διπλωματική δε δόθηκε βάση στην πρόβλεψη της κίνησης και της συμπεριφοράς άλλων δυναμικών εμποδίων όπως οχήματα και πεζοί. Για την καλύτερη αποφυγή των εμποδίων και την επιλογή συμπεριφοράς του οχήματος θα ήταν μια καλή λύση να υπάρχει ένα ξεχωριστό υποσύστημα το οποίο μέσω ενός μοντέλου όπως το μαρκοβιανό θα προβλέπει τις κινήσεις των υπόλοιπων εμποδίων.

Από την πλευρά της διεπαφής στην οποία έχει πρόσβαση ο χρήστης θα μπορούσαν να υπάρξουν σημαντικές βελτιώσεις. Αρχικά, μια βασική βελτίωση βρίσκεται στο γεγονός ότι η διεπαφή θα μπορούσε να γίνει πιο δυναμική παρέχοντας στο χρήστη δεδομένα πραγματικού χρόνου όπως η εικόνα των καμερών που θα διέθετε το όχημα δίνοντας του καλύτερη εικόνα για την κατάσταση στην οποία βρίσκεται το όχημα κατά την κίνηση του όταν το χειρίζεται απομακρυσμένα. Επίσης, θα μπορούσε να προστεθεί μια απεικόνιση πραγματικού χρόνου του χάρτη μέσα στον οποίο κινείται παρακολουθώντας συνεχώς τη θέση του μέσα στην πόλη.

Επιπλέον, μια σημαντική επέκταση θα μπορούσε να είναι ο τρόπος επιλογής τοποθεσιών πάνω στο χάρτη. Όπως περιγράφηκε από την υλοποίηση παραπάνω οι τοποθεσίες που είναι διαθέσιμες στο όχημα είναι συγκεκριμένες και ορισμένες για την πόλη μέσα στην οποία κινείται το όχημα. Θα μπορούσε να υπάρξει ένα σύστημα στο οποίο ο χρήστης θα μπορεί να επιλέξει οποιοδήποτε σημείο πάνω στο χάρτη και να το αποθηκεύσει σε περίπτωση που επιθυμεί να το επιλέξει ξανά μελλοντικά. Εκτός αυτού βέβαια θα μπορούσαν να υλοποιηθούν επιπλέον πλήκτρα που θα δίνουν εντολές για άλλα σενάρια κίνησης του οχήματος όπως επιλογή για αυτόνομη στάθμευση, ολοκληρωμένη προσπέραση, ακολούθηση συμπεριφοράς εμπρόσθιου οχήματος κλπ.

Κλείνοντας, μια πολύ ενδιαφέρουσα επέκταση θα μπορούσε να είναι στην ασφάλεια των δεδομένων του συγκεκριμένου συστήματος. Με την προσθήκη ενός επιπλέον επιπέδου ασφάλειας στη συγκεκριμένη αρχιτεκτονική θα μπορούσε να μελετηθεί η ασφαλής μεταφορά των μηνυμάτων από τη διεπαφή μέχρι το αυτόνομο όχημα και αντίστροφα.

Βιβλιογραφία

- [1] Ronald R Knippling. “*Car-Truck Crashes in the National Motor Vehicle Crash Causation Survey*“. In “*Proceedings of the 8th International Driving Symposium on Human Factors in Driver Assessment, Training, and Vehicle Design: driving assessment 2015*“, pages 310–316, Salt Lake City, Utah, USA, 2015. University of Iowa.
- [2] Rasheed Hussain and Sherali Zeadally. “*Autonomous Cars: Research Results, Issues, and Future Challenges*“. IEEE Communications Surveys & Tutorials, 21(2):1275–1313, 2019. ISSN 1553-877X, 2373-745X.
- [3] Dwight A. Hennessy and David L. Wiesenthal. “*Traffic congestion, driver stress, and driver aggression*“. Aggressive Behavior, 25(6):409–423, 1999. ISSN 1098-2337.
- [4] Bai Li and Zhijiang Shao. “*A unified motion planning method for parking an autonomous vehicle in the presence of irregularly placed obstacles*“. Knowledge-Based Systems, 86:11–20, September 2015. ISSN 09507051.
- [5] Zhilu Chen and Xinming Huang. “*End-to-end learning for lane keeping of self-driving cars*“. In “*2017 IEEE Intelligent Vehicles Symposium (IV)*“, pages 1856–1860, Los Angeles, CA, USA, June 2017. IEEE. ISBN 9781509048045.
- [6] Vicente Milanés, David F. Llorca, Jorge Villagrá, Joshué Pérez, Carlos Fernández, Ignacio Parra, Carlos González, and Miguel A. Sotelo. “*Intelligent automatic overtaking system using vision for vehicle detection*“. Expert Systems with Applications, 39(3):3362–3373, February 2012. ISSN 09574174.
- [7] Ekim Yurtsever, Jacob Lambert, Alexander Carballo, and Kazuya Takeda. “*A Survey of Autonomous Driving: Common Practices and Emerging Technologies*“. IEEE Access, 8:58443–58469, 2020. ISSN 2169-3536.
- [8] Alberto Broggi, Michele Buzzoni, Stefano Debattisti, Paolo Grisleri, Maria Chiara Laghi, Paolo Medici, and Pietro Versari. “*Extensive Tests of Autonomous Driving Technologies*“. IEEE Transactions on Intelligent Transportation Systems, 14(3):1403–1415, September 2013. ISSN 1524-9050, 1558-0016.
- [9] Naoki Akai, Luis Yoichi Morales, Takuma Yamaguchi, Eijiro Takeuchi, Yuki Yoshihara, Hiroyuki Okuda, Tatsuya Suzuki, and Yoshiki Ninomiya. “*Autonomous driving based on accurate localization using multilayer LiDAR and*

dead reckoning“. In “*2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*“, pages 1–6, October 2017. ISSN: 2153-0017.

- [10] Julius Ziegler, Philipp Bender, Markus Schreiber, Henning Lategahn, Tobias Strauss, Christoph Stiller, Thao Dang, Uwe Franke, Nils Appenrodt, Christoph G. Keller, Eberhard Kaus, Ralf G. Herrtwich, Clemens Rabe, David Pfeiffer, Frank Lindner, Fridtjof Stein, Friedrich Erbs, Markus Enzweiler, Carsten Knoppel, Jochen Hipp, Martin Haueis, Maximilian Trepte, Carsten Brenk, Andreas Tamke, Mohammad Ghanaat, Markus Braun, Armin Joos, Hans Fritz, Horst Mock, Martin Hein, and Eberhard Zeeb. “*Making Bertha Drive—An Autonomous Journey on a Historic Route*“. *IEEE Intelligent Transportation Systems Magazine*, 6(2):8–20, 2014. ISSN 1939-1390.
- [11] Khadige Abboud, Hassan Aboubakr Omar, and Weihua Zhuang. “*Interworking of DSRC and Cellular Network Technologies for V2X Communications: A Survey*“. *IEEE Transactions on Vehicular Technology*, 65(12):9457–9470, December 2016. ISSN 1939-9359.
- [12] Sheralli Zeadally, Ray Hunt, Yuh-Shyan Chen, Angela Irwin, and Aamir Hassan. “*Vehicular ad hoc networks (VANETS): status, results, and challenges*“. *Telecommunication Systems*, 50(4):217–241, August 2012. ISSN 1572-9451.
- [13] Kichun Jo, Junsoo Kim, Dongchul Kim, Chulhoon Jang, and Myoungho Sunwoo. “*Development of Autonomous Car—Part II: A Case Study on the Implementation of an Autonomous Driving System Based on Distributed Architecture*“. *IEEE Transactions on Industrial Electronics*, 62(8):5119–5132, August 2015. ISSN 1557-9948.
- [14] Junsoo Kim, Kichun Jo, Dongchul Kim, Keonyup Chu, and Myoungho Sunwoo. “*Behavior and Path Planning Algorithm of Autonomous Vehicle A1 in Structured Environments*“. *IFAC Proceedings Volumes*, 46(10):36–41, June 2013. ISSN 14746670.
- [15] Johannes Betz, Alexander Wischnewski, Alexander Heilmeier, Felix Nobis, Tim Stahl, Leonhard Hermansdorfer, and Markus Lienkamp. “*A Software Architecture for an Autonomous Racecar*“. In “*2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*“, pages 1–6, Kuala Lumpur, Malaysia, April 2019. IEEE. ISBN 9781728112176.
- [16] Tulga Ersal, Ilya Kolmanovsky, Neda Masoud, Necmiye Ozay, Jeffrey Scruggs, Ram Vasudevan, and Gábor Orosz. “*Connected and automated road vehicles: state of the art and future challenges*“. *Vehicle System Dynamics*, 58(5):672–704, May 2020. ISSN 0042-3114, 1744-5159.
- [17] Rui Fan, Jianhao Jiao, Haoyang Ye, Yang Yu, Ioannis Pitas, and Ming Liu. “*Key Ingredients of Self-Driving Cars*“. *arXiv:1906.02939 [cs, eess]*, August 2019. arXiv: 1906.02939.
- [18] Sebastian Thrun. “*Toward robotic cars*“. *Communications of the ACM*, 53(4):99–106, April 2010. ISSN 0001-0782.

ΒΙΒΛΙΟΓΡΑΦΙΑ

- [19] “*CaRINA Intelligent Robotic Car: Architectural design and applications*“. Journal of Systems Architecture, 60(4):372–392, April 2014. ISSN 1383-7621.
- [20] D. Pomerleau. “*ALVINN: An Autonomous Land Vehicle in a Neural Network*“. In “*NIPS*“, 1988.
- [21] AhmadEL Sallab, Mohammed Abdou, Etienne Perot, and Senthil Yogamani. “*Deep Reinforcement Learning framework for Autonomous Driving*“. Electronic Imaging, 2017(19):70–76, January 2017. ISSN 2470-1173.
- [22] Fangchun Yang, Jinglin Li, Tao Lei, and Shangguang Wang. “*Architecture and key technologies for Internet of Vehicles: a survey*“. Journal of Communications and Information Networks, 2(2):1–17, June 2017. ISSN 2509-3312.
- [23] Khondokar Fida Hasan, Tarandeep Kaur, Md Mhedi Hasan, and Yanming Feng. “*Cognitive Internet of Vehicles: Motivation, Layered Architecture and Security Issues*“. arXiv:1912.03356 [cs], November 2019. arXiv: 1912.03356.
- [24] Juan Contreras-Castillo, Sheralli Zeadally, and Juan Antonio Guerrero-Ibañez. “*Internet of Vehicles: Architecture, Protocols, and Security*“. IEEE Internet of Things Journal, 5(5):3701–3709, October 2018. ISSN 2327-4662.
- [25] Baofeng Ji, Xueru Zhang, Shahid Mumtaz, Congzheng Han, Chunguo Li, Hong Wen, and Dan Wang. “*Survey on the Internet of Vehicles: Network Architectures and Applications*“. IEEE Communications Standards Magazine, 4(1):34–41, March 2020. ISSN 2471-2833.
- [26] Asma Alshehri and Ravi Sandhu. “*Access Control Models for Cloud-Enabled Internet of Things: A Proposed Architecture and Research Agenda*“. In “*2016 IEEE 2nd International Conference on Collaboration and Internet Computing (CIC)*“, pages 530–538, November 2016.
- [27] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. “*CARLA: An Open Urban Driving Simulator*“. In “*Proceedings of the 1st Annual Conference on Robot Learning*“, pages 1–16, 2017.
- [28] Laurene Claussmann, Marc Revilloud, Sébastien Glaser, and Dominique Gruyer. “*A study on al-based approaches for high-level decision making in highway autonomous driving*“. In “*2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*“, pages 3671–3676, Banff, AB, October 2017. IEEE. ISBN 9781538616451.
- [29] Andrei Furda and Ljubo Vlacic. “*Enabling Safe Autonomous Driving in Real-World City Traffic Using Multiple Criteria Decision Making*“. IEEE Intelligent Transportation Systems Magazine, 3(1):4–17, 2011. ISSN 1939-1390.
- [30] Tesfaye Hailemariam Yimer, Chao Wen, Xiaozhuo Yu, and Chaozhe Jiang. “*A Study of the Minimum Safe Distance between Human Driven and Driverless Cars Using Safe Distance Model*“. arXiv:2006.07022 [physics], June 2020. arXiv: 2006.07022.