ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ - ΤΜΗΜΑ ΦΥΣΙΚΗΣ, ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ
Δ.Π.Μ.Σ. «ΜΕΤΑΠΤΥΧΙΑΚΟ ΔΙΠΛΩΜΑ ΕΙΔΙΚΕΥΣΗΣ ΣΤΟΝ ΗΛΕΚΤΡΟΝΙΚΟ ΑΥΤΟΜΑΤΙΣΜΟ (Η/Α)»

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ολοκληρωμένο σύστημα παρακολούθησης και αυτόνομης οδήγησης οχήματος με χρήση τεχνικών ανάλυσης εικόνας

Αριστείδης Ι. Οικονόμου (Α.Μ.: 2016520)

Επιβλέπων: Ιωάννης Καλατζής, Αναπληρωτής Καθηγητής ΠΑ.Δ.Α.

Αθήνα, Ιανουάριος 2020

ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ - ΤΜΗΜΑ ΦΥΣΙΚΗΣ, ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ
Δ.Π.Μ.Σ. «ΜΕΤΑΠΤΥΧΙΑΚΟ ΔΙΠΛΩΜΑ ΕΙΔΙΚΕΥΣΗΣ ΣΤΟΝ ΗΛΕΚΤΡΟΝΙΚΟ ΑΥΤΟΜΑΤΙΣΜΟ (Η/Α)»

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ολοκληρωμένο σύστημα παρακολούθησης και αυτόνομης οδήγησης οχήματος με χρήση τεχνικών ανάλυσης εικόνας

Αριστείδης Ι. Οικονόμου (Α.Μ.: 2016520)

Επιβλέπων: Ιωάννης Καλατζής, Αναπληρωτής Καθηγητής ΠΑ.Δ.Α.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή:

.....	.....	.....
Ι. Καλατζής	Π. Ασβεστάς	Δ. Ρεΐσης
Αναπλ. Καθηγητής ΠΑ.Δ.Α.	Αναπλ. Καθηγητής ΠΑ.Δ.Α.	Αναπλ. Καθηγητής Ε.Κ.Π.Α.

Αθήνα, Ιανουάριος 2020

.....

Αριστείδης Ι. Οικονόμου

Μεταπτυχιακός Διπλωματούχος Ειδίκευσης στον Ηλεκτρονικό Αυτοματισμό (Η/Α)

Περίληψη

Στις μέρες μας ολοένα και περισσότερες εγκαταστάσεις, όπως αποθήκες, νοσοκομεία και ξενοδοχεία, διαχειρίζονται τις μεταφορές των αντικειμένων τους μέσα στον χώρο τους χρησιμοποιώντας ρομποτικά οχήματα. Τα συγκεκριμένα οχήματα είναι εξοπλισμένα με ειδικές κάμερες και αισθητήρες και είτε χαρτογραφούν και πλοηγούνται στον χώρο ή ακολουθούν σημάνσεις, κατά κύριο λόγο γραμμές στο δάπεδο, και πλοηγούνται με βάση αυτές. Με την παρούσα εργασία επιχειρείται η δημιουργία και κατασκευή ενός ολοκληρωμένου συστήματος παρακολούθησης και αυτόνομης οδήγησης οχημάτων εξ' ολοκλήρου με χρήση τεχνικών ανάλυσης εικόνας.

Αρχικά, παρουσιάζεται η γενική δομή του συστήματος, το οποίο αποτελείται από τρία στάδια: τα αυτόνομα οχήματα, τον κεντρικό εξυπηρετητή και τον διαχειριστή. Για την επικοινωνία μεταξύ των τριών σταδίων αναπτύχθηκαν ειδικά λογισμικά, όπου βασικό ρόλο διαδραματίζει αυτό του εξυπηρετητή. Εν γένει, αφού ταυτοποιηθούν οι χρήστες του συστήματος, η μεταξύ τους επικοινωνία βασίζεται στην αποστολή μηνυμάτων στον εξυπηρετητή και στη συνέχεια στην προώθησή τους από τον δεύτερο στους παραλήπτες. Η επικοινωνία βασίζεται σε ασφαλή, ασύγχρονα και αμφίδρομα πρωτόκολλα με σκοπό το κάθε στάδιο του συστήματος να γνωρίζει ανά πάσα στιγμή την κατάσταση των υπόλοιπων σταδίων.

Επιπλέον, αναφορικά με τη λειτουργία του συστήματος, ο διαχειριστής δύναται να αναθέτει μέσω του εξυπηρετητή διαδρομές στα οχήματα, τα οποία κινούμενα σε ειδικό χώρο με σημάνσεις στο δάπεδο και χρησιμοποιώντας αποκλειστικά τεχνικές ανάλυσης εικόνας, πλοηγούνται από τον κόμβο εκκίνησης προς αυτόν του τερματισμού. Ο ειδικός χώρος περιλαμβάνει σταυροδρόμια, τα οποία καλούνται τα οχήματα να αναγνωρίσουν και με τη βοήθεια του εσωτερικού τους χάρτη να αποφασίσουν προς ποια κατεύθυνση πρέπει να πορευτούν, ώστε να φτάσουν στον προορισμό τους.

Τέλος, εκτελείται μία δοκιμή λειτουργίας του συστήματος πλοήγησης σε εργαστηριακό περιβάλλον υπό μη βέλτιστες συνθήκες. Αρχικά, ο διαχειριστής επιλέγει και αναθέτει σε ένα όχημα μία επιθυμητή διαδρομή, οπότε αυτό εκκινεί την πλοήγησή του. Κατά τη διάρκειά της δοκιμής ελέγχεται η επιτυχία και η ορθότητα της επικοινωνίας μεταξύ διαχειριστή, εξυπηρετητή και οχήματος καθώς και των αλγορίθμων όρασης και πλοήγησης του οχήματος, καθώς αυτό καλείται να διέλθει από διαφορετικού τύπου σταυροδρόμια και υπό κακές συνθήκες φωτισμού. Διαπιστώνεται ότι η επικοινωνία διεξάγεται ομαλά και κατά τον αναμενόμενο τρόπο και το όχημα καταλήγει στον επιθυμητό προορισμό. Ωστόσο, η απλότητα του συστήματος και οι περιορισμοί που περιέχει καθιστούν απαραίτητη την περαιτέρω μελέτη και ανάπτυξή του. Συμπερασματικά, κρίνεται πως το αναπτυχθέν σύστημα της παρούσας μελέτης αποτελεί μία σημαντική βάση για την ανάπτυξη πολυπλοκότερων συστημάτων παρακολούθησης και αυτόνομης οδήγησης οχημάτων εξ' ολοκλήρου με χρήση τεχνικών ανάλυσης εικόνας.

Λέξεις κλειδιά: ROS, ρομποτικά οχήματα, αυτόνομα οχήματα, αυτόνομη οδήγηση, OpenCV, επεξεργασία εικόνας, συστήματα παρακολούθησης, ενσωματωμένα συστήματα, Arduino, μικροελεγκτής, ESP8266

Abstract

Nowadays, more and more facilities, such as warehouses, hospitals and hotels, manage the transportation of their objects within their premises using robotic vehicles. These vehicles are equipped with special cameras and sensors and can either map and navigate on site or follow markings, mainly on the floor, and navigate through them. The present thesis attempts to create and construct an integrated tracking and autonomous vehicle driving system using image analysis techniques solely.

Firstly, the general structure of the system is presented, which consists of three stages: the autonomous vehicles, the central server and the administrator. Specific software was developed for the communication between the three stages, where the server plays the most substantial role. In general, once the users of the system have been identified, the communication between them is based on sending messages to the server and then forwarding them to the respective recipient. The communication is based on secure, asynchronous and two-way protocols, so that each stage of the system is at any time aware of the status of the other stages.

In addition, with respect to the operation of the system, the administrator may assign routes to the vehicles by means of the server, whereafter the vehicles can navigate from the starting node to the destination node moving through a dedicated space with markings on the floor and using exclusively image analysis techniques. The special area includes crossroads, which the vehicles have to identify and, using their internal map, they decide the required travel direction in order to reach their destination.

Finally, a test of the navigation system’s operation was carried out in a laboratory environment under non-optimal conditions. Initially, the administrator selected and assigned a desired route to one vehicle, so that the latter started its navigation. During this test, it was examined whether the communication between the operator, the server and the vehicle as well as the vehicle's vision and navigation algorithms

were correct and successful, since the vehicle had to pass through different types of crossroads and under poor lighting conditions. It was found that the communication was carried out smoothly and as expected and that the vehicle reached the desired destination. However, the simplicity of the system as well as its limitations make further study and development necessary. In conclusion, the developed system of the present study is considered an important basis for the development of more complex tracking and autonomous vehicle driving systems using image analysis techniques solely.

Keywords: ROS, robotic vehicles, autonomous vehicles, autonomous driving, OpenCV, image processing, tracking systems, embedded systems, Arduino, microcontroller, ESP8266

Ευχαριστίες

Στο πλαίσιο της διπλωματικής μου εργασίας θα ήθελα να ευχαριστήσω θερμά όλους, όσοι με οποιονδήποτε τρόπο βοήθησαν στην περάτωσή της. Αρχικά, θα ήθελα να εκφράσω τις ευχαριστίες μου στον επιβλέποντα αναπληρωτή καθηγητή ΠΑΔΑ κ. Ιωάννη Καλατζή για την εμπιστοσύνη που έδειξε στο πρόσωπό μου με την ανάθεση της εν λόγω εργασίας. Επιπλέον, θα ήθελα να ευχαριστήσω θερμά τον αναπληρωτή καθηγητή ΠΑΔΑ κ. Παντελεήμονα Ασβεστά ο οποίος ήταν πάντα πρόθυμος να με βοηθήσει στην εκπόνηση της διπλωματικής μου εργασίας. Τέλος, μεγάλη ευγνωμοσύνη οφείλω στους γονείς μου Ιωάννη και Πηνελόπη, οι οποίοι αποτέλεσαν στήριγμα κατά τη διάρκεια των σπουδών μου, αλλά και στις ευρύτερες επιλογές στη ζωή μου.

Περιεχόμενα

1. Εισαγωγή

1.1 Σκοπός διπλωματικής εργασίας

Στις μέρες μας ολοένα και περισσότερες εγκαταστάσεις, όπως αποθήκες, νοσοκομεία και ξενοδοχεία, διαχειρίζονται τις μεταφορές αντικειμένων τους στον χώρο τους χρησιμοποιώντας ρομποτικά οχήματα. Τέτοιου είδους οχήματα είναι εξοπλισμένα με ειδικές κάμερες και αισθητήρες και είτε χαρτογραφούν και πλοηγούνται στον χώρο είτε ακολουθούν σημάνσεις, κατά κύριο λόγο γραμμές στο δάπεδο, και πλοηγούνται με βάση αυτές.

Σκοπός της παρούσας μελέτης είναι η δημιουργία και υλοποίηση ενός ολοκληρωμένου συστήματος παρακολούθησης και αυτόνομης οδήγησης οχημάτων εξ' ολοκλήρου με χρήση τεχνικών ανάλυσης εικόνας διατηρώντας όμως χαμηλό το συνολικό κόστος. Η υλοποίηση του συγκεκριμένου συστήματος γίνεται στα πλαίσια πρωτοτυποποίησης (prototyping) με στόχο να λειτουργήσει πιλοτικά με ένα ιδιοκατασκευασμένο όχημα σε χώρο συγκεκριμένων συνθηκών και με ειδικές σημάνσεις χωρίς όμως αυτό να συνεπάγεται πως οι συνθήκες αυτές είναι απόλυτα ιδανικές για το σύστημα.

Κατά την υλοποίηση του συστήματος ιδιαίτερη έμφαση δίνεται στην ασφάλεια που πρέπει να το διέπει τόσο στην ταυτοποίηση των συμμετεχόντων είτε αυτοί είναι χρήστες ή οχήματα όσο και στην ακεραιότητα των δεδομένων τόσο κατά τη μεταφορά όσο και κατά την αποθήκευσή τους. Τέλος, ιδιαίτερη έμφαση δίνεται στην επεκτασιμότητα του λογισμικού του οχήματος ώστε να είναι εύκολη η ενσωμάτωση επιπλέον αισθητήρων σε αυτό, αλλά και η ανάπτυξη και ενσωμάτωση πιο σύνθετων αλγορίθμων ανάλυσης εικόνας και πλοήγησης.

1.2 Δομή διπλωματικής εργασίας

Η εργασία αποτελείται από έξι κεφάλαια τα οποία παρουσιάζονται ακολούθως:

1ο Κεφάλαιο: Εισαγωγή

Στο πρώτο, παρόν, κεφάλαιο γίνεται μία εισαγωγή στο θέμα και στον στόχο της διπλωματικής εργασίας καθώς και μία σύντομη περιγραφή του περιεχομένου κάθε κεφαλαίου.

2ο Κεφάλαιο: Δομή και λειτουργία του συστήματος

Στο δεύτερο κεφάλαιο παρουσιάζεται η δομή και η λειτουργία του προς υλοποίηση συστήματος. Παρουσιάζονται τα μέρη από τα οποία αποτελείται το σύστημα, ο ρόλος του καθενός μέσα σε αυτό καθώς και ο τρόπος με τον οποίον επικοινωνούν μεταξύ τους.

3ο Κεφάλαιο: Υλοποίηση του Υλικού του αυτόνομου οχήματος

Στο τρίτο κεφάλαιο παρουσιάζεται η υποδομή, ο τρόπος κατασκευής καθώς και τα κριτήρια επιλογής των συστατικών του αυτόνομου οχήματος το οποίο χρησιμοποιείται στο υλοποιούμενο σύστημα.

4ο Κεφάλαιο: Υλοποίηση του Λογισμικού του συστήματος

Στο τέταρτο κεφάλαιο περιγράφεται η σχεδίαση και η υλοποίηση του λογισμικού (software) σε όλα τα επίπεδα του ολοκληρωμένου συστήματος. Πιο συγκεκριμένα, περιγράφεται τι είναι ένα ROS σύστημα, το λογισμικό του εξυπηρετητή του συστήματος, η διεπαφή του διαχειριστή του συστήματος, οι αλγόριθμοι όρασης και πλοήγησης του οχήματος καθώς και το λογισμικό του μικροελεγκτή ο οποίος ελέγχει την κίνηση του οχήματος.

5ο Κεφάλαιο: Δοκιμή του συστήματος

Στο πέμπτο κεφάλαιο δημιουργείται ένα προς εκτέλεση σενάριο δοκιμής για το υλοποιημένο σύστημα ώστε να παρουσιαστεί και να ελεγχθεί αν στην ολότητά του είναι λειτουργικό. Μέσα από την εκτέλεση της δοκιμής διαφαίνονται οι αδυναμίες της υλοποίησης και εξάγονται ορισμένα συμπεράσματα για την έκβασή της.

6ο Κεφάλαιο: Βελτιώσεις και Συμπεράσματα

Τέλος, στο έκτο κεφάλαιο παρουσιάζονται, αρχικά, μερικές από τις βελτιώσεις τις οποίες επιδέχεται το σύστημα και, στη συνέχεια εξάγονται τα τελικά συμπεράσματα για την πορεία και την έκβαση της παρούσας εργασίας.

2. Δομή και λειτουργία του συστήματος

2.1 Εισαγωγή

Στο παρόν κεφάλαιο παρουσιάζεται η δομή και η λειτουργία του προς υλοποίηση συστήματος. Παρουσιάζονται τα μέρη από τα οποία αποτελείται το σύστημα, ο ρόλος του καθενός μέσα σε αυτό καθώς και ο τρόπος με τον οποίον επικοινωνούν μεταξύ τους.

2.2 Δομή του συστήματος

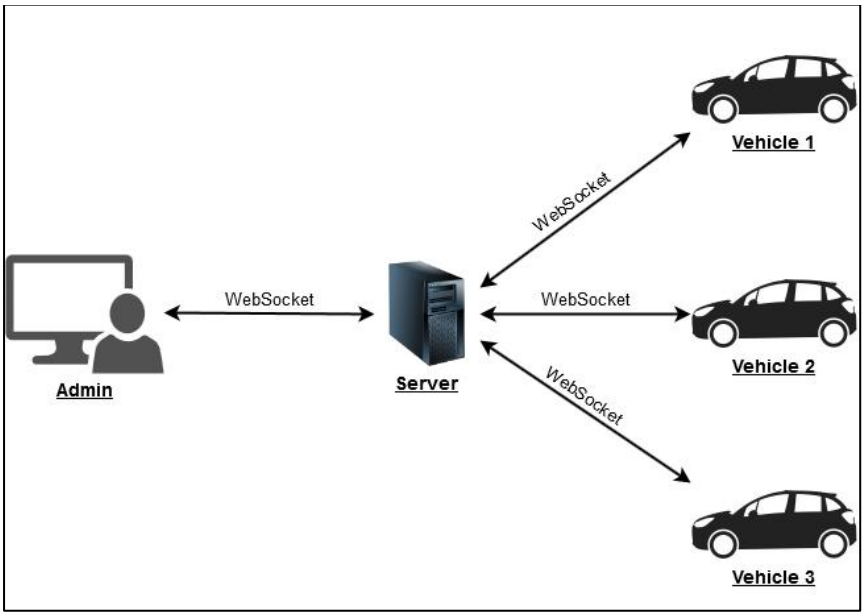
Στη γενικευμένη του μορφή το σύστημα το οποίο έχει υλοποιηθεί απαρτίζεται από τρία στάδια τα οποία παρουσιάζονται ακολούθως.

Το πρώτο στάδιο περιλαμβάνει τα ειδικά διαμορφωμένα οχήματα τα οποία βρίσκονται σε έναν ομοίως ειδικά διαμορφωμένο χώρο χωρίς φυσικά εμπόδια, του οποίου το δάπεδο διαθέτει ένα δίκτυο γραμμών. Σε αυτόν τον χώρο το εκάστοτε όχημα καλείται με τη χρήση μόνο της κάμερας που φέρει και τεχνικών ανάλυσης εικόνας να ανιχνεύει και να ακολουθεί γραμμές καθώς και να αποφασίζει προς ποια κατεύθυνση πρέπει να στρίβει όταν μέσω των γραμμών αναγνωρίζει σταυροδρόμια.

Το δεύτερο στάδιο περιλαμβάνει τον διαχειριστή του συστήματος, ο οποίος μέσω μιας διαδικτυακής εφαρμογής είναι σε θέση να αλληλεπιδρά με τα οχήματα και είτε να τα χειρίζεται απομακρυσμένα ή να τους μεταφέρει διαδρομές τις οποίες αυτά θα εκτελούν. Στην περίπτωση που μεταφέρει την επιθυμητή διαδρομή στο όχημα, εκείνο γνωρίζοντας το σημείο εκκίνησής του και με βάση τα σταυροδρόμια

τα οποία ανιχνεύει κατά την πορεία του πάνω στο δίκτυο των γραμμών στρίβει αυτόνομα έως ότου να φτάσει στον προορισμό του.

Τέλος, το τρίτο στάδιο του συστήματος και πιο σημαντικό για την ορθή λειτουργία του είναι ο εξυπηρετητής. Ο συγκεκριμένος είναι υπεύθυνος για τη σύνδεση του διαχειριστή και των οχημάτων με το σύστημα και την αμφίδρομη και ασύγχρονη μεταβίβαση πληροφοριών μεταξύ των δύο. Η σύζευξη μεταξύ των τριών σταδίων επιτυγχάνεται με τη χρήση του πρωτοκόλλου Διαδικτύου (Internet Protocol) και του πρωτοκόλλου WebSocket. Στην ακόλουθη εικόνα παρουσιάζεται η γενικευμένη μορφή του συστήματος, περιλαμβάνοντας τα τρία στάδια που αναφέρθηκαν τα οποία χωρίζονται μέσω του κοινού πρωτοκόλλου επικοινωνίας.



Εικόνα 2.1. *Γενικευμένη μορφή του συστήματος*

2.3 Παρουσίαση λειτουργίας του συστήματος

Όπως αναφέρθηκε προηγουμένως το σημαντικότερο στάδιο για την ορθή λειτουργία του συστήματος είναι το τρίτο, δηλαδή ο εξυπηρετητής. Αυτό συμβαίνει διότι για να εκκινήσει οποιαδήποτε διαδικασία πρέπει πρώτα είτε ο διαχειριστής ή τα οχήματα να έχουν ταυτοποιηθεί από τον εξυπηρετητή. Αν δεν ταυτοποιηθούν τότε δεν αποτελούν μέρος του συστήματος. Από την άλλη πλευρά, όταν ταυτοποιηθούν ο εξυπηρετητής δημιουργεί μέσω αυτού έναν αμφίδρομο δίαυλο επικοινωνίας μεταξύ του διαχειριστή και των οχημάτων χρησιμοποιώντας το πρωτόκολλο WebSocket.

Αναλυτικότερα, όταν στο σύστημα είναι ενεργός μόνο ο εξυπηρετητής και ενεργοποιείται ένα όχημα τότε το δεύτερο αποστέλλει ένα αίτημα σύνδεσης επιδεικνύοντας την ταυτότητά του. Η ταυτότητα του οχήματος είναι ένα αναγνωριστικό το οποίο έχει παραχθεί και υπογραφεί από τον εξυπηρετητή και έχει εμφωλευτεί μέσα σε αυτό. Μόλις ο εξυπηρετητής λάβει την ταυτότητα του συγκεκριμένου οχήματος ελέγχει με τη χρήση της βάσης δεδομένων του αν είναι γνήσια και σε ισχύ και αναλόγως επιτρέπει στο όχημα να συνδεθεί στο σύστημα ή όχι. Κατ' αντιστοιχία, όταν ο διαχειριστής θελήσει να συνδεθεί στο σύστημα εισάγει στη διαδικτυακή εφαρμογή που του παρέχει ο εξυπηρετητής τα στοιχεία του, όνομα και κωδικό. Ο εξυπηρετητής λαμβάνει αυτά τα στοιχεία, τα διασταυρώνει και επιστρέφει στον διαχειριστή την πλατφόρμα ελέγχου των οχημάτων συμπληρωμένη με τα οχήματα τα οποία είναι συνδεδεμένα στο σύστημα.

Ο διαχειριστής διαθέτει δύο δυνατότητες στο σύστημα. Η πρώτη είναι ο απομακρυσμένος χειρισμός του εκάστοτε οχήματος μέσω της χρήσης του πληκτρολογίου του. Η δεύτερη δυνατότητα που διαθέτει είναι η ανάθεση στο εκάστοτε όχημα μίας διαδρομής για αυτόνομη οδήγηση με γνωστή αρχή σε γνωστό χάρτη. Ο συγκεκριμένος χάρτης και ο τρόπος δημιουργίας του από μία φυσική διαδρομή αναλύεται εκτενέστερα σε ακόλουθο κεφάλαιο, επιγραμματικά

όμως πρόκειται για έναν γράφο ο οποίος προκύπτει από τις κάθετες τομές των ευθειών που είναι αποτυπωμένες στο δάπεδο του χώρου όπου κινούνται τα οχήματα. Οι τομές των ευθειών που σχηματίζουν σταυροδρόμια στον πραγματικό χώρο, δηλαδή διαθέτουν παραπάνω από μία επιλογές δρόμων, αποτυπώνονται σε κόμβους στον γράφο και οι σύνδεσμοι των κόμβων αντιστοιχούν στους πραγματικούς δρόμους που ενώνουν τα σταυροδρόμια. Μόλις το όχημα λάβει την διαδρομή, αρχίζει να λαμβάνει και να επεξεργάζεται μία προς μία και σε πραγματικό χρόνο τις εικόνες που καταγράφει η κάμερά του. Η πληροφορία, η οποία επιδιώκεται να εξαχθεί από την εικόνα είναι πόσες και ποιες ευθείες ανιχνεύονται σε αυτήν. Οι ευθείες είναι αυτές οι οποίες καθορίζουν αν το όχημα βρίσκεται σε κάποιο σταυροδρόμι, αν υπάρχουν κάθετες μεταξύ τους ευθείες, ή σε απλό δρόμο, αν είναι μεταξύ τους παράλληλες. Όταν το όχημα βλέπει απλό δρόμο μπροστά του τότε απλά τον ακολουθεί έως ότου εντοπίσει κάποιο σταυροδρόμι. Όταν εντοπίσει σταυροδρόμι τότε προσπαθεί να εντοπίσει το σχήμα του από τις ευθείες που το συνιστούν. Όταν εντοπίσει το σχήμα του τότε ανατρέχει σε έναν εσωτερικό χάρτη που διαθέτει το κάθε όχημα στο λογισμικό του και προκύπτει από τον χάρτη του διαχειριστή για να δει ποια κατεύθυνση πρέπει να ακολουθήσει. Ο συγκεκριμένος χάρτης στην ουσία περιέχει όλους τους κόμβους του χάρτη του διαχειριστή και για κάθε έναν γνωρίζει ποια κατεύθυνση πρέπει να ακολουθήσει το όχημα ώστε να βρεθεί από τον προηγούμενο κόμβο στον επόμενο μέσω του τρέχοντος που μόλις εντόπισε. Αν ταυτοποιηθεί το σταυροδρόμι μέσω του εσωτερικού χάρτη τότε το όχημα ξεκινά να πορεύεται προς την κατάλληλη κατεύθυνση. Αν δεν ταυτοποιηθεί, δηλαδή δεν έχει επιλογές στις κατευθύνσεις όσες δείχνει ο εσωτερικός χάρτης, τότε το όχημα ακυρώνει τη διαδρομή του και ενημερώνει τον διαχειριστή.

Εδώ πρέπει να σημειωθεί πως και στις δύο δυνατότητες χειρισμού που διαθέτει ο διαχειριστής, δηλαδή του απομακρυσμένου χειρισμού μέσω πληκτρολογίου αφενός και της αυτόνομης πλοήγησης αφετέρου, η καταγραφή της κάμερας του οχήματος, είτε καθαρή ή επεξεργασμένη, μεταφέρεται στην πλατφόρμα του ώστε

αυτός να είναι σε θέση να επιβλέπει την πορεία του οχήματος, ενώ επιπλέον, στην περίπτωση της αυτόνομης οδήγησης ο χάρτης του διαχειριστή ενημερώνεται κάθε φορά που το όχημα περνά από κάποιον κόμβο.

3. Υλοποίηση του Υλικού του αυτόνομου οχήματος

3.1 Εισαγωγή

Στο παρόν κεφάλαιο παρουσιάζεται η υποδομή και ο τρόπος κατασκευής του αυτόνομου οχήματος το οποίο χρησιμοποιείται στο υλοποιούμενο σύστημα. Τα κριτήρια με βάση τα οποία επιλέχτηκαν τα επιμέρους τμήματα του οχήματος είναι η απλότητα, το κόστος, η λειτουργικότητα, η σωστή επικοινωνία-ελεγξιμότητα και η πρόβλεψη ανάγκης για μελλοντική επεκτασιμότητα του υλικού.

3.2 Επιλογή οχήματος

Για την υλοποίηση του συστήματος επιλέχθηκε ένα όχημα σχετικά ογκώδες, το οποίο να είναι σε θέση να ενσωματώσει τα απαραίτητα, ογκώδη αυτά καθαυτά, συστατικά στοιχεία, τα οποία θα περιγραφούν αργότερα. Επιπλέον κριτήριο επιλογής ήταν το όχημα να διαθέτει κινητήρες ικανούς να παρέχουν επαρκή ροπή ώστε να κινούν το φορτίο του. Πέραν τούτου, κρίθηκε σκόπιμο το όχημα να είναι ερπυστριοφόρο με δύο τροχούς, ώστε να μπορεί να εκτελεί επιτόπιες περιστροφές και να είναι ευκολότερα υλοποιήσιμο προγραμματιστικά το μοντέλο οδήγησής του αλλά και ο χειρισμός του. Για τους παραπάνω λόγους και σε συνδυασμό με το χαμηλό του κόστος, το όχημα το οποίο επιλέχτηκε εν τέλει είναι ένας τηλεχειριζόμενος εκσκαφέας, ο "*Dickie R/C Mighty Εκσκαφέας 1:14*" της εταιρίας "*Dickie*", ο οποίος φαίνεται στην ακόλουθη εικόνα.

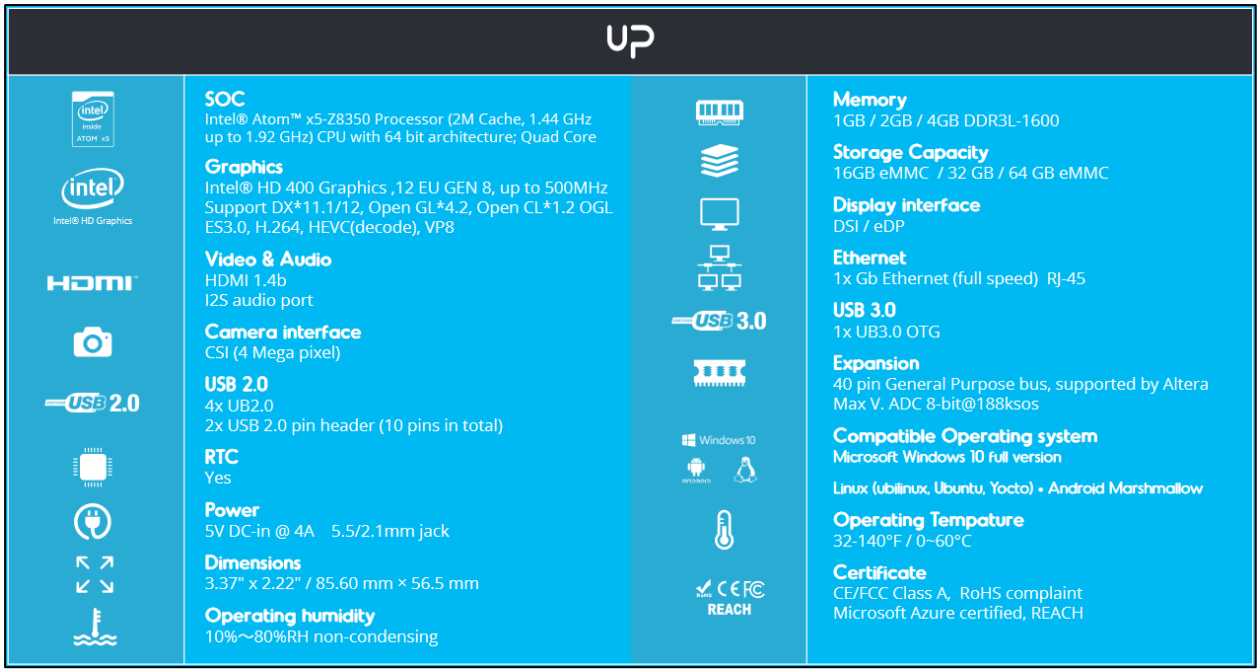
Εικόνα 3.1. *Ο εκσκαφέας Dickie R/C Mighty*
[Πηγή: https://www.moustakastoys.gr/gr/thlekateythynomena/ohimata-me-kalodio/dickie-rc-mighty-ekskafeas-114-203412782-401953012782/?]

Το μήκος της συσκευασίας του οχήματος είναι 23.5 cm, το πλάτος της 70 cm, το ύψος της 35 cm και το βάρος του οχήματος ανέρχεται στα 2.4 kg. Όπως διαφαίνεται από τα στοιχεία του, πρόκειται για ένα σχετικά ογκώδες και βαρύ όχημα, ωστόσο από αυτό τελικά διατηρήθηκε μόνο το σύστημα των τροχών με το σασί του.

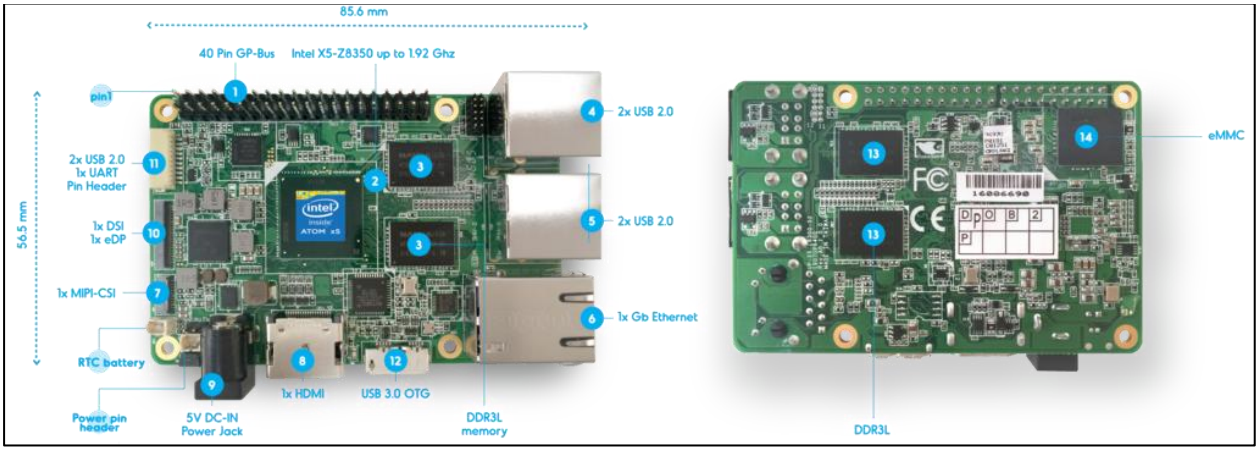
3.3 Επιλογή επεξεργαστικής μονάδας

Η επεξεργαστική μονάδα αποτελεί την πρωταρχική συνιστώσα σε ένα αυτόνομο όχημα, δεδομένου ότι πρέπει να εκτελεί πολύ γρήγορα πολύπλοκες μαθηματικές πράξεις σχεδόν σε πραγματικό χρόνο. Για την ορθή επιλογή της έπρεπε να ληφθούν αρκετές παράμετροι υπόψη όπως το κόστος, η κατανάλωση και η επίδοση.

Ως επεξεργαστική μονάδα επιλέχτηκε η πλατφόρμα *UP* της εταιρίας *UP Board*. Η συγκεκριμένη επιλογή προήλθε από τη χρήση του αναπτυξιακού πακέτου *INTEL® REALSENSE™ ROBOTIC DEVELOPMENT KIT (RDK)* το οποίο επίσης περιλαμβάνει την κάμερα *Intel® RealSense™ R200* η οποία χρησιμοποιήθηκε και παρουσιάζεται παρακάτω. Το συγκεκριμένο αναπτυξιακό πακέτο είναι αρκετά γνωστό στον χώρο των αυτόνομων οχημάτων και του ρομποτισμού και η κάμερά του έχει χρησιμοποιηθεί και στο χαμηλού κόστους ρομποτικό όχημα TurtleBot3 Waffle της εταιρίας ROBOTIS. Η πλατφόρμα UP είναι μια μικρού μεγέθους πλακέτα με υψηλές επιδόσεις και χαμηλή κατανάλωση η οποία ενσωματώνει τον τετραπύρηνο 64bit επεξεργαστή (CPU) της Intel, τον Intel® ATOM™ x5-Z8350 ο οποίος χρονίζεται έως τα 1.92GHz. Ο εσωτερικός επεξεργαστής γραφικών (GPU) είναι ο νέος Intel Gen 8 HD 400 με 12 επεξεργαστικές μονάδες χρονισμένες έως τα 500MHz με σκοπό να προσφέρουν υψηλή επίδοση στην επεξεργασία 3D γραφικών. Η πλατφόρμα UP είναι επίσης εξοπλισμένη με 4GB DDR3L μνήμη RAM και 32GB eMMC αποθηκευτικό χώρο. Η πλακέτα της διαθέτει επίσης θύρες USB2 και USB3 για τη σύνδεση περιφερειακών συσκευών καθώς και πολλαπλούς γενικού σκοπού ακροδέκτες εισόδου-εξόδου (GPIO pins). Στις ακόλουθες εικόνες παρουσιάζονται συνοπτικά τα χαρακτηριστικά της πλακέτας UP καθώς και η ίδια η πλακέτα.



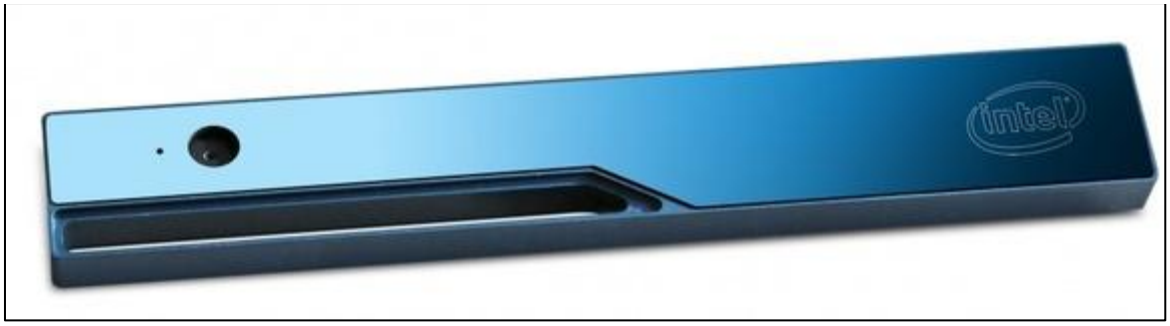
Εικόνα 3.2. *Τα χαρακτηριστικά της πλατφόρμας UP*
[Πηγή: https://up-board.org/wp-content/uploads/datasheets/UPDatasheetV8.5.pdf]



Εικόνα 3.3. *Η εμπρόσθια (αριστερά) και οπίσθια (δεξιά) όψη της πλατφόρμας UP με τα συστατικά της [Πηγή: https://up-board.org/wp-content/uploads/2018/10/layout-1024x361.png]*

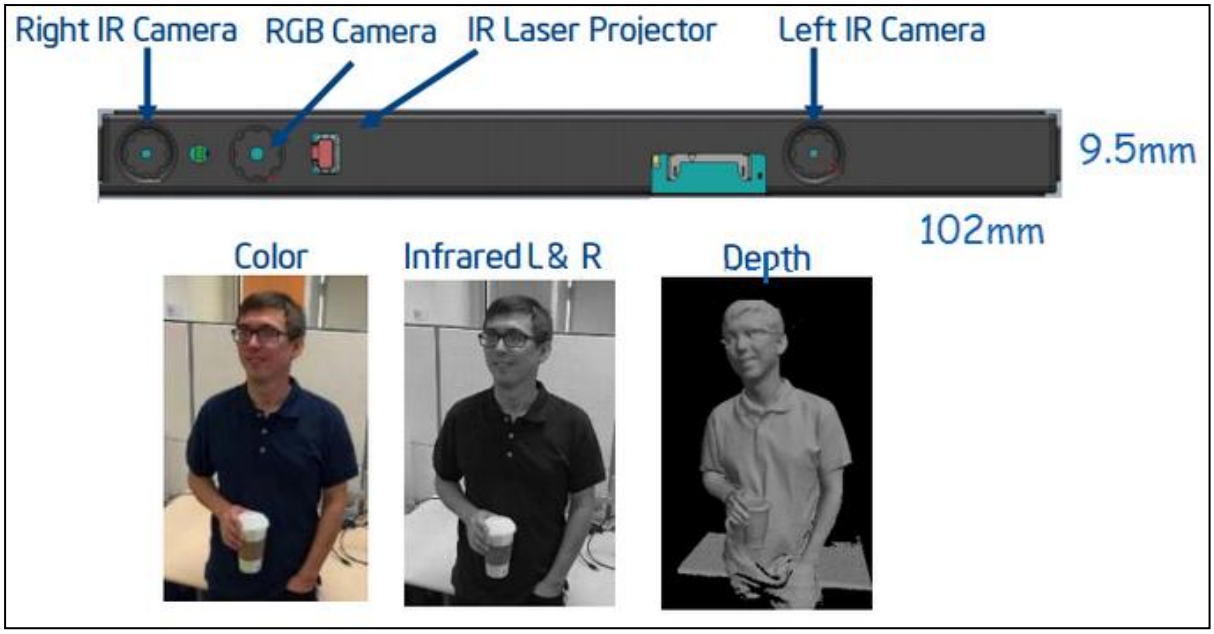
3.4 Επιλογή κάμερας

Όπως αναφέρθηκε προηγουμένως το όχημα είναι εξοπλισμένο με την κάμερα Intel® RealSense™ R200. Η συγκεκριμένη κάμερα στην πραγματικότητα περιέχει 3 κάμερες οι οποίες παρέχουν RGB (χρώμα) εικόνα και στερεοσκοπικές IR εικόνες για την παραγωγή εικόνας βάθους [1]. Με τη βοήθεια ενός προβολέα λέιζερ κατηγορίας 1 στην περιοχή των 850 nm, η κάμερα εκτελεί 3D σάρωση για την αντίληψη σκηνής και την ενισχυμένη φωτογράφηση. Η εσωτερική της εμβέλεια κυμαίνεται στο εύρος 0.5-3.5 m και η εξωτερική της μέχρι τα 10 m. Να σημειωθεί πως η εμβέλεια εξαρτάται πολύ από τον τύπο της κάμερας και τον φωτισμό.



Εικόνα 3.4. *Η κάμερα Intel® RealSense™ R200*
[Πηγή: https://images.techhive.com/images/article/2016/01/r200-100637539-large.jpg]

Ένα από τα πιο συναρπαστικά πράγματα σχετικά με την κάμερα R200 είναι η μεγαλύτερη ικανότητα σάρωσης και η νέα μέθοδος μέτρησης βάθους. Η R200 περιλαμβάνει δύο στερεοσκοπικές IR κάμερες καθώς και μία RGB. Δεδομένου ότι η κάμερα είναι λιγότερο ευαίσθητη στην IR ακτινοβολία, μπορεί να χρησιμοποιηθεί και σε εξωτερικούς χώρους.



Εικόνα 3.5. *Η δομή της κάμερας Intel® RealSense™ R200 [1]*

Η έγχρωμη κάμερα είναι ικανή να συλλαμβάνει 32-bit RGBA εικόνες, ανάλυσης έως 1080p και με ρυθμό έως 60FPS χρησιμοποιώντας σταθερή εστίαση και αναλογία διαστάσεων 16:3. Η RGB κάμερα έχει ελαφρώς μεγαλύτερο οπτικό πεδίο (FOV) από τις διπλές IR κάμερες, αλλά δεν προορίζεται για χρήση ως αυτόνομη κάμερα. Οι διπλές κάμερες χρησιμοποιούν σταθερό λόγο ευκρίνειας 4:3 με γωνίες οπτικού πεδίου 70x59x46. Οι διαθέσιμες αναλύσεις των καμερών με βάση τον ρυθμό λήψης τους παρουσιάζονται στον ακόλουθο πίνακα.

FPS	DEPTH	RGB
60	320x240	640x480
60	480x360	320x240 ή 640x480
30	320x240	640x480 ή 1920x1080
30	480x360	320x240, 640x480 ή 1920x1080

Πίνακας 3.1. *Διαθέσιμες αναλύσεις των καμερών με βάση τα διαθέσιμα FPS [1]*

Τέλος, η ισχύς τροφοδοσίας της R200 κυμαίνεται από 0 έως 100mW (σε κατάσταση αναμονής) και από 1.0 έως 1.6W όταν είναι ενεργή.

Πάνω στην κάμερα ενσωματώθηκαν τρία LEDs τύπου 5050, κίτρινου χρώματος (3000K). Η ανάγκη για τη χρήση των LEDs προέκυψε κατά τη διάρκεια των δοκιμών του αλγορίθμου όρασης του οχήματος, καθώς με την ύπαρξη τους αυξάνεται η αντίθεση και μειώνεται ο θόρυβος στην εικόνα δημιουργώντας παράλληλα σταθερές συνθήκες φωτισμού.

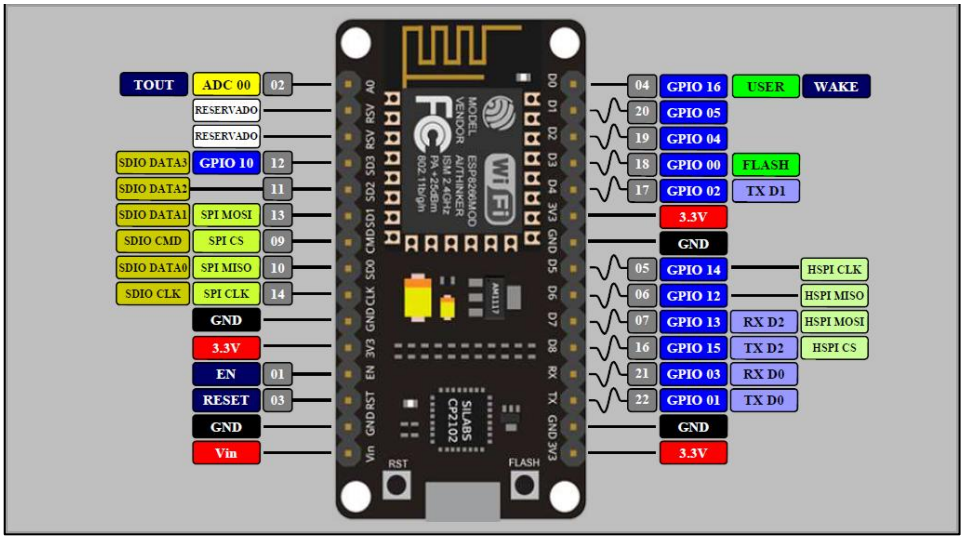
3.5 Κύκλωμα οδήγησης του αυτόνομου οχήματος

Από το όχημα αφαιρέθηκαν όλα τα εσωτερικά κυκλώματα που διέθετε και το μόνο που απέμεινε είναι οι δύο κινητήρες των τροχών του. Για να επιτευχτεί ο χειρισμός αυτών των κινητήρων από την επεξεργαστική μονάδα έγινε χρήση ενός μικροελεγκτή, ο οποίος ελέγχεται από την επεξεργαστική μονάδα και με τη σειρά του ελέγχει την ταχύτητα και τη φορά των τροχών μέσω μιας κυκλωματικής διάταξης.

Οι λόγοι για τους οποίους δεν ελέγχει η επεξεργαστική μονάδα απευθείας μέσω των γενικού σκοπού εξόδων της τους κινητήρες είναι τρεις. Αρχικά, οι κινητήρες αποτελούν ένα σχετικά μεγάλο φορτίο για να μπορέσουν να οδηγηθούν από τους ακροδέκτες χωρίς να τους καταστρέψουν και επίσης χρειάζονται μεγαλύτερη τάση από τα 5V που παρέχουν οι ακροδέκτες. Στη συνέχεια, η επεξεργαστική μονάδα διαθέτει μόνο δύο ακροδέκτες με δυνατότητα παραγωγής PWM σήματος ενώ χρειάζονται τέσσερις για την ταχύτητα περιστροφής αλλά και τη φορά των κινητήρων. Και, τέλος, είναι καλή πολιτική για λόγους επεκτασιμότητας, ευκολότερης ενσωμάτωσης και σταθερότητας, εφόσον το ROS σύστημα το υποστηρίζει, η ενσωμάτωση ενός μικροελεγκτή ως ROS κόμβου, ώστε να ελέγχει τους κινητήρες και να ενσωματώνει περιφερειακούς αισθητήρες. Αυτή είναι μια καλή πολιτική διότι η επεξεργαστική μονάδα διαθέτει λειτουργικό σύστημα το οποίο εισάγει καθυστερήσεις και αστοχίες στην επικοινωνία με περιφερειακές συσκευές, ενώ ένας μικροελεγκτής μπορεί ταχυτέρα, σταθερότερα και με

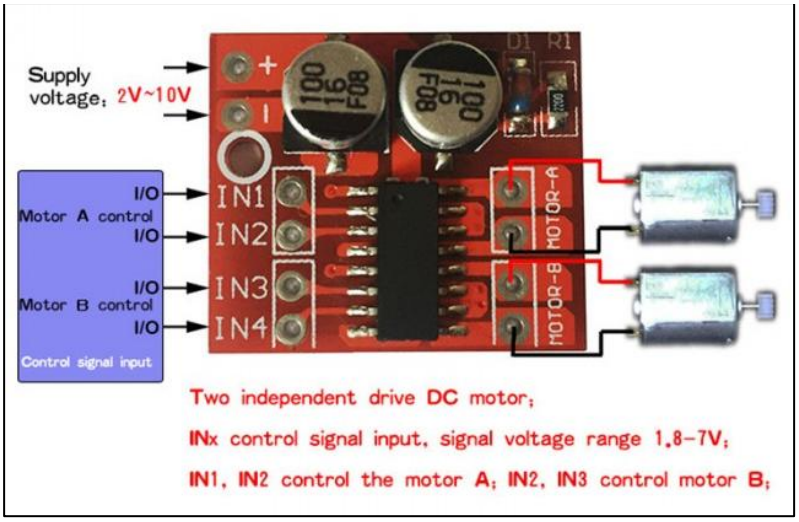
μεγαλύτερη ευκολία στην ανάπτυξη λογισμικού να συλλέγει πληροφορίες και να τις στέλνει στην επεξεργαστική μονάδα καθώς και να ελέγχει εξωτερικές συσκευές.

Ως μικροελεγκτής χρησιμοποιήθηκε ο *ESP8266* ο οποίος συνδέεται μέσω μιας USB2 θύρας με την UP πλακέτα και ελέγχει τους κινητήρες μέσω του οδηγού κινητήρων L298N Mini. Στην παρούσα φάση της υλοποίησης να αναφερθεί πως δεν χρησιμοποιείται κάποιος αισθητήρας παρά μόνο η κάμερα, αλλά όπως ειπώθηκε η ύπαρξη ενός μικροελεγκτή προσφέρει επεκτασιμότητα στην περίπτωση που καταστεί απαραίτητος κάποιος αισθητήρας. Ο ESP8266 είναι ένας γνωστός και δοκιμασμένος χαμηλού κόστους και ισχύος και μεγάλης υπολογιστικής δύναμης 32-bit μικροελεγκτής της εταιρίας Espressif Systems ο οποίος ενσωματώνει δυνατότητες WiFi. Για την ακρίβεια χρησιμοποιείται η αναπτυξιακή πλακέτα NodeMCU η οποία ενσωματώνει τον ESP8266 με πολλαπλούς εκτεθειμένους ακροδέκτες εισόδου-εξόδου και κυκλώματα τροφοδοσίας και προγραμματισμού για το τσιπ. Στη συγκεκριμένη εφαρμογή αξιοποιείται ο συγκεκριμένος μικροελεγκτής για την υπολογιστική του δύναμη καθώς είναι χρονισμένος στα 80MHz και φτάνει μέχρι και τα 160MHz διαθέτοντας 1MB Flash μνήμη. Επίσης, διαθέτει πολυάριθμους ακροδέκτες εισόδου-εξόδου τάσης 3.3V με δυνατότητα παραγωγής PWM σήματος, είναι συμβατός με την πλατφόρμα Arduino και πολυχρησιμοποιημένος από την κοινότητα, πράγμα το οποίο διευκολύνει τον προγραμματισμό του και την ενσωμάτωση περιφερειακών σε αυτόν.



Εικόνα 3.6. *Η δομή της αναπτυξιακής πλακέτας NodeMCU*
[Πηγή: http://www.ifuturetech.org/ifuture/uploads/2017/07/AMICA-NODEMCU-ESP8266-LUA-CP2102-WIFI-DEVELOPMENT-MODULE-IOT-gujarat.png]

Ο *L298N Mini* είναι ένας οδηγός για κινητήρες συνεχούς ρεύματος ο οποίος υποστηρίζει την οδήγηση έως δύο κινητήρων μέγιστου ρεύματος 1.5A ανά κινητήρα. Έχει τη δυνατότητα να τροφοδοτεί κινητήρες με τάση εύρους από 2V έως 10V. Ο ρόλος του οδηγού είναι να λαμβάνει τέσσερα PWM σήματα τάσης από 1.8V έως 7V από τέσσερις ακροδέκτες του μικροελεγκτή και με βάση αυτά τα σήματα να ελέγχει τους κινητήρες ως προς τη φορά τους και την ταχύτητα περιστροφής τους. Έπειτα από εργαστηριακές δοκιμές, η ανώτερη τάση τροφοδότησης των κινητήρων του οχήματος ώστε αυτοί να μην υπερθερμαίνονται είναι τα 9V. Για τον λόγο αυτόν συνδέεται όπως φαίνεται παρακάτω μια πηγή τάσης 9V στον ακροδέκτη VCC της πλακέτας του οδηγού. Όπως αναφέρθηκε, ο μικροελεγκτής συνδέεται με τις εισόδους του οδηγού (IN1-IN4) μέσω τεσσάρων ακροδεκτών (D5-D8) οι οποίοι δίνουν μέσω των PWM σημάτων τάσεις από 0 έως 3.3V.



Εικόνα 3.7. *Η δομή της πλακέτας L298N Mini και η συνδεσμολογία της με μικροελεγκτή, κινητήρες και τροφοδοσία [Πηγή: http://www.robotpark.com/image/cache/data/PRO/71148/71148-2-700x700.jpg]*

3.6 Κύκλωμα τροφοδότησης του αυτόνομου οχήματος

Το τελευταίο βήμα και σημαντικότερο για την ολοκλήρωση του κυκλώματος του αυτονόμου οχήματος ήταν η σωστή και σταθερή τροφοδότησή του. Η εξοικονόμηση ενέργειας, όπου αυτό είναι εφικτό, είναι απαραίτητη για την αύξηση της αυτονομίας του οχήματος.

Για την τροφοδότηση των παραπάνω συσκευών πρόεκυψε η ανάγκη για μια κεντρική πηγή τροφοδοσίας τάσης 12V από μια μπαταρία τύπου Lead Acid. Στην προκείμενη περίπτωση επιλέχτηκε η SPA 12-2.3Ah της εταιρίας Sunlight. Η συγκεκριμένη ανάγκη πρόεκυψε διότι απαιτείται, αρχικά, μια τάση 12V για την τροφοδότηση της συστοιχίας των τριών LEDs της κάμερας. Στη συνέχεια, η ανώτερη τάση τροφοδότησης των κινητήρων του οχήματος είναι τα 9V. Τέλος, η συγκεκριμένη πηγή τροφοδοτεί και την επεξεργαστική μονάδα με 5V η οποία τροφοδοτεί τον μικροελεγκτή ο οποίος ελέγχει τους κινητήρες του οχήματος, την Wi-Fi κεραία και την κάμερα μέσω των θυρών USB2 και USB3 αντίστοιχα. Οι λόγοι που επιλέχτηκε ο συγκεκριμένος τύπος μπαταρίας έναντι π.χ. ενός power bank είναι το χαμηλότερο κόστος, το υψηλότερο επίπεδο τάσης και το μεγαλύτερο ρεύμα εξόδου. Το αρνητικό φυσικά με τη συγκεκριμένη επιλογή είναι ο όγκος και

το βάρος της μπαταρίας τα οποία δυσκολεύουν την κίνηση των κινητήρων του οχήματος με συνέπεια την αύξηση της κατανάλωσης. Βέβαια το συγκεκριμένο αρνητικό καθιστά πιο ρεαλιστικές τις συνθήκες κίνησης του οχήματος. Στον ακόλουθο πίνακα παρουσιάζονται τα χαρακτηριστικά της μπαταρίας.

Applications

- Uninterruptible Power Supplies (UPS)
- Electric Power Systems (EPS)
- Emergency backup power supplies
- Electronic apparatus and equipment
- Communication power supplies
- DC power supplies
- Auto control system

Specifications

Nominal Voltage	12 V	
Number of cells	6	
Design Life	10 years	
Dimensions	Length	178 mm
	Width	35 mm
	Height	61 mm
	Total Height	67 mm
Approx. Weight	0.80 kg	
Nominal Capacity (25°C)	20 hours rate (0.115 A, 10.5 V)	2.30 Ah
	10 hours rate (0.210 A, 10.5 V)	2.10 Ah
	5 hours rate (0.390 A, 10.5 V)	1.95 Ah
	1 hour rate (1.500 A, 9.6 V)	1.50 Ah
Max. Discharge Current (25°C)	34.5 A (5s)	
Max. Charging Current (25°C)	0.69 A	
Internal Resistance Fully Charged battery (25°C)	60 mOhms	
Self-Discharge (25°C)	3% of capacity declined per month (average)	
Operating Temperature Range	Discharge : -15~50°C	
	Charge : -10~50°C	
	Storage : -20~50°C	
Short Circuit Current	74.5 A	
Charge Methods: Constant Voltage Charge (25°C)	Cycle use	2.40-2.45 Vpc
	Temperature compensation	-30 mV/°C
	Standby use	2.25-2.30 Vpc
	Temperature compensation	-18 mV/°C

Εικόνα 3.8. *Ο πίνακας χαρακτηριστικών της μπαταρίας SPA 12-2.3Ah*
[Πηγή: http://energypower.gr/wp-content/uploads/2016/08/spa-12-2.3_vrla.pdf]

Για την τροφοδότηση της πλακέτας UP και των περιφερειακών της απαιτείται μια σταθερή πηγή τάσης 5V και μεγίστου ρεύματος 5A, δηλαδή το πολύ 25W ισχύος με πολύ χαμηλές απώλειες ισχύος. Για τον λόγο αυτόν χρησιμοποιήθηκε ο μεταβλητής εξόδου υποβιβαστής και σταθεροποιητής τάσης (DC to DC Step Down Converter) XL4005 ο οποίος λαμβάνει στην είσοδό του εύρος τάσεων από 4V έως 38V και δίνει στην έξοδό του εύρος από 1.25V έως 32V και μπορεί να διαχειρίζεται ρεύματα μέχρι 5A και ισχύ μέχρι και 75W. Ο συγκεκριμένος υποβιβαστής τάσης έρχεται σε πολύ χαμηλό κόστος, με βαθμό απόδοσης στη μεταφορά ισχύος από την είσοδό του στην έξοδό του έως και 96% και είναι ικανός να παρέχει στο συγκεκριμένο κύκλωμα την αναγκαία ισχύ χωρίς να υπόκειται σε φθορές.

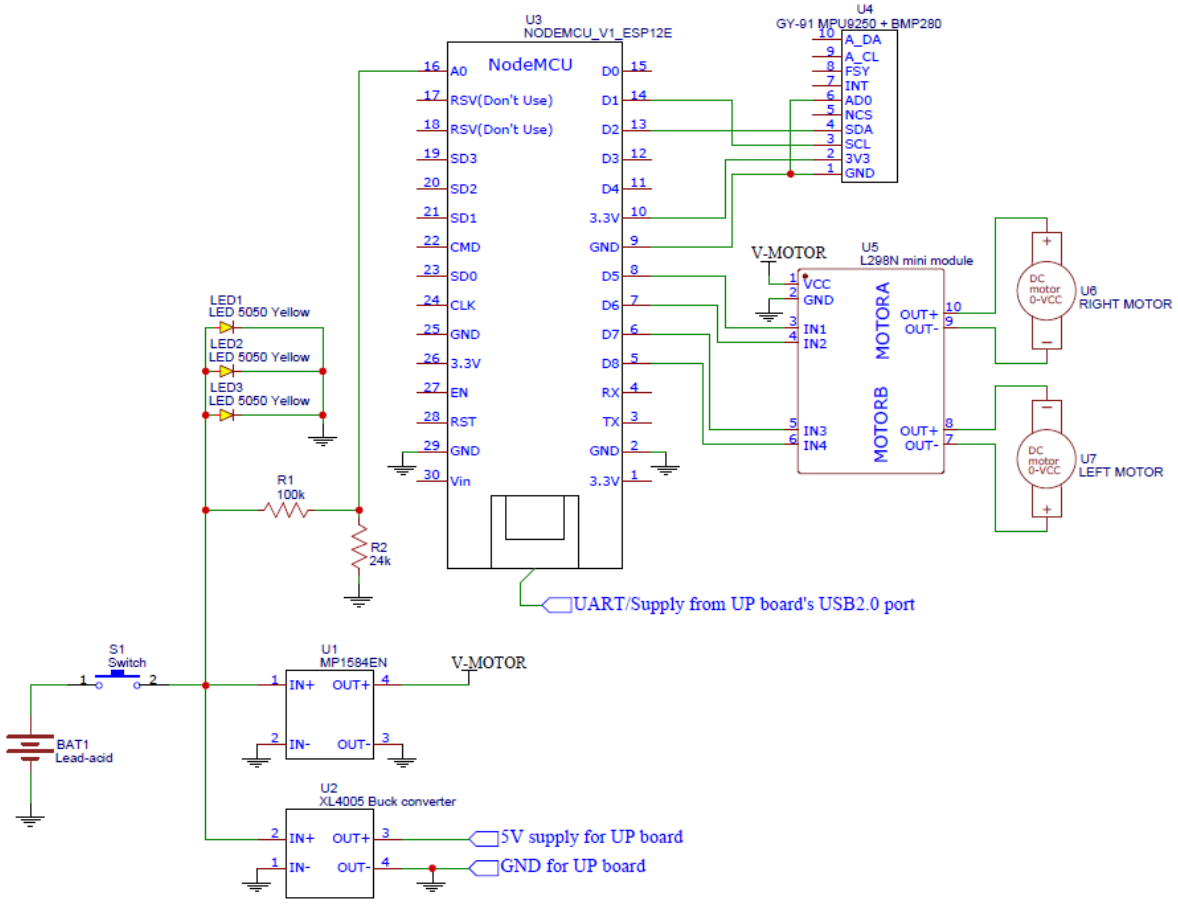
Εικόνα 3.9. *Ο υποβιβαστής τάσης XL4005*
[Πηγή: https://www.cableworks.gr/images/thumbnails/343/300/detailed/3/XL4005.jpg]

Για την τροφοδότηση του κυκλώματος οδήγησης των τροχών του οχήματος χρησιμοποιήθηκε ομοίως ένας μεταβλητής εξόδου υποβιβαστής τάσης, ο MP1584EN, ώστε να υπάρχει η δυνατότητα να μεταβάλλεται η μέγιστη τάση οδήγησης των κινητήρων στο επιθυμητό επίπεδο, στη συγκεκριμένη περίπτωση 9V. Ο συγκεκριμένος μετατροπέας λαμβάνει στην είσοδό του εύρος τάσεων από 4V έως 28V και δίνει στην έξοδό του εύρος από 0.8V έως 20V και μπορεί να διαχειρίζεται ρεύματα μέχρι 3A. Έρχεται σε πολύ χαμηλό κόστος, μικρό μέγεθος και με βαθμό απόδοσης έως και 92%.

Εικόνα 3.10. *Ο υποβιβαστής τάσης MP1584EN [Πηγή: https://www.cableworks.gr/images/thumbnails/343/300/detailed/3/MP1584EN_converter.jpg]*

3.7 Το κυκλωματικό σχηματικό του αυτόνομου οχήματος

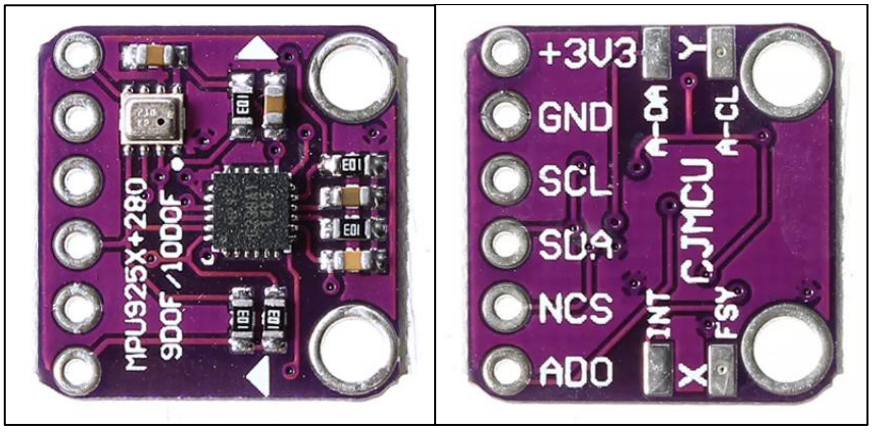
Έχοντας καλύψει πλέον όλα τα μέρη από τα οποία αποτελείται το αυτόνομο όχημα, επόμενο βήμα αποτελεί η σχεδίαση και η κατασκευή του κυκλώματος με βάση τις παραπάνω επιλογές. Στην ακόλουθη εικόνα παρουσιάζεται το σχηματικό διάγραμμα του κυκλώματος το οποίο προέκυψε συνδυάζοντας τα επιλεγμένα στοιχεία.



Εικόνα 3.11. *Το σχηματικό διάγραμμα του κυκλώματος του οχήματος*

Είναι φανερό πως στο σχηματικό διάγραμμα του κυκλώματος υπάρχουν επιπλέον ορισμένα στοιχεία τα οποία δεν αναφέρθηκαν παραπάνω και αναλύονται ακολούθως.

Το στοιχείο **U4** είναι η πλακέτα GY-91 η οποία αποτελείται από τον αισθητήρα βαρομετρικής πίεσης BMP-280 και τον πολλαπλό αισθητήρα MPU-9255. Η πλακέτα GY-91 επικοινωνεί με τον μικροελεγκτή μέσω του σειριακού πρωτοκόλλου I2C. Το MPU-9255 είναι ένα σύστημα το οποίο συνδυάζει δύο τσιπς. Αρχικά, περιέχει το MPU-6500, το οποίο περιέχει ένα τριών αξόνων γυροσκόπιο, ένα τριών αξόνων επιταχυνσιόμετρο και έναν ενσωματωμένο Ψηφιακό Επεξεργαστή Κίνησης (DMP™), ο οποίος είναι ικανός να επεξεργάζεται πολύπλοκους αλγορίθμους. Τέλος, περιέχει το AK8963, το οποίο είναι μια ψηφιακή πυξίδα τριών αξόνων (μαγνητόμετρο). Η πλακέτα GY-91 ενσωματώθηκε στο σύστημα του οχήματος για λόγους πληρότητας και για λόγους μελέτης της συμπεριφοράς του οχήματος στα μεγέθη τα οποία μετρά. Αυτό σημαίνει πως δεν επενεργεί στην οδήγηση του οχήματος αλλά μελετάται η συμπεριφορά του για μελλοντική εξέλιξη του αλγορίθμου οδήγησης και χρήση σε αλγόριθμο εντοπισμού θέσης του οχήματος στον χώρο.



Εικόνα 3.12. *Η πλακέτα GY-91 του IMU module [Πηγή: https://www.banggood.com/MPU9250BMP280-10DOF-GY-91-Acceleration-Gyroscope-Compass-Nine-Shaft-Sensor-Module-For-Arduino-p-1100982.html?rmmds=myorder&cur_warehouse=CN]*

Το δικτύωμα των αντιστατών **R1** και **R2** συνιστά έναν διαιρέτη τάσης του οποίου η έξοδος καταλήγει σε μια αναλογική είσοδο του μικροελεγκτή για τη μέτρηση της στάθμης της μπαταρίας. Οι λόγοι για τους οποίους δεν συνδέεται κατευθείαν η μπαταρία στην αναλογική είσοδο του μικροελεγκτή είναι δύο. Ο

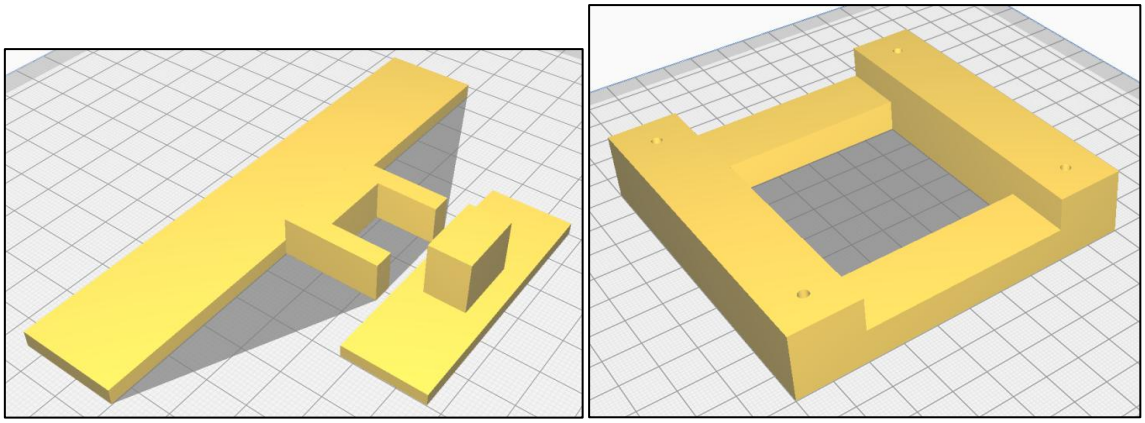
πρώτος λόγος είναι πως οι ακροδέκτες του μικροελεγκτή αντέχουν τιμές τάσεων έως 3.3V ενώ η μπαταρία μπορεί να λάβει μέγιστη τιμή τάσης τα 15V. Με τη χρήση του διαιρέτη τάσης με λόγο διαίρεσης 1/5.1667 απεικονίζεται το εύρος τάσης της μπαταρίας στο εύρος 0-3.2V. Ο δεύτερος λόγος είναι πως με τη χρήση της **R1**, η οποία έχει μεγάλη τιμή αντίστασης, ο μικροελεγκτής κατά την εκκίνησή του δεν μπορεί να βρεθεί σε κάποια απροσδιόριστη κατάσταση όπου να τροφοδοτείται από τον συγκεκριμένο ακροδέκτη.

Τα στοιχεία **LED1-3** είναι τα LEDs τα οποία τοποθετηθήκαν επάνω στην κάμερα και τροφοδοτούνται απευθείας από την κεντρική πηγή τροφοδοσίας με 12V καθώς αυτό είναι το επίπεδο τάσης τροφοδοσίας τους. Η κατανάλωση και των τριών LEDs ανέρχεται στα 612mW καθώς στα 12V καταναλώνουν 17mA το καθένα, δηλαδή 3*17mA*12V=612mW.

Τέλος, το στοιχείο **S1** πρόκειται για έναν κεντρικό διακόπτη δύο θέσεων ο οποίος αντέχει σχετικά μεγάλα DC ρεύματα και χρησιμοποιείται για την ενεργοποίηση και απενεργοποίηση όλου του κυκλώματος του οχήματος. Να σημειωθεί πως στο κύκλωμα του οχήματος δεν προστέθηκε κάποιο υποκύκλωμα για την τροφοδότηση της μπαταρίας. Η μπαταρία φορτίζεται από τροφοδοτικό πάγκου με τη μέθοδο σταθερού ρεύματος και σταθερής τάσης (CC/CV), κατά την οποία η τάση φόρτισης αυξάνεται μέχρι να φτάσει το ανώτατο όριο των 14.2V, όπου το μέγιστο ρεύμα φόρτισης 0.7A αρχίζει να μειώνεται σταδιακά λόγω του κορεσμού.

3.8 Κατασκευή του αυτόνομου οχήματος

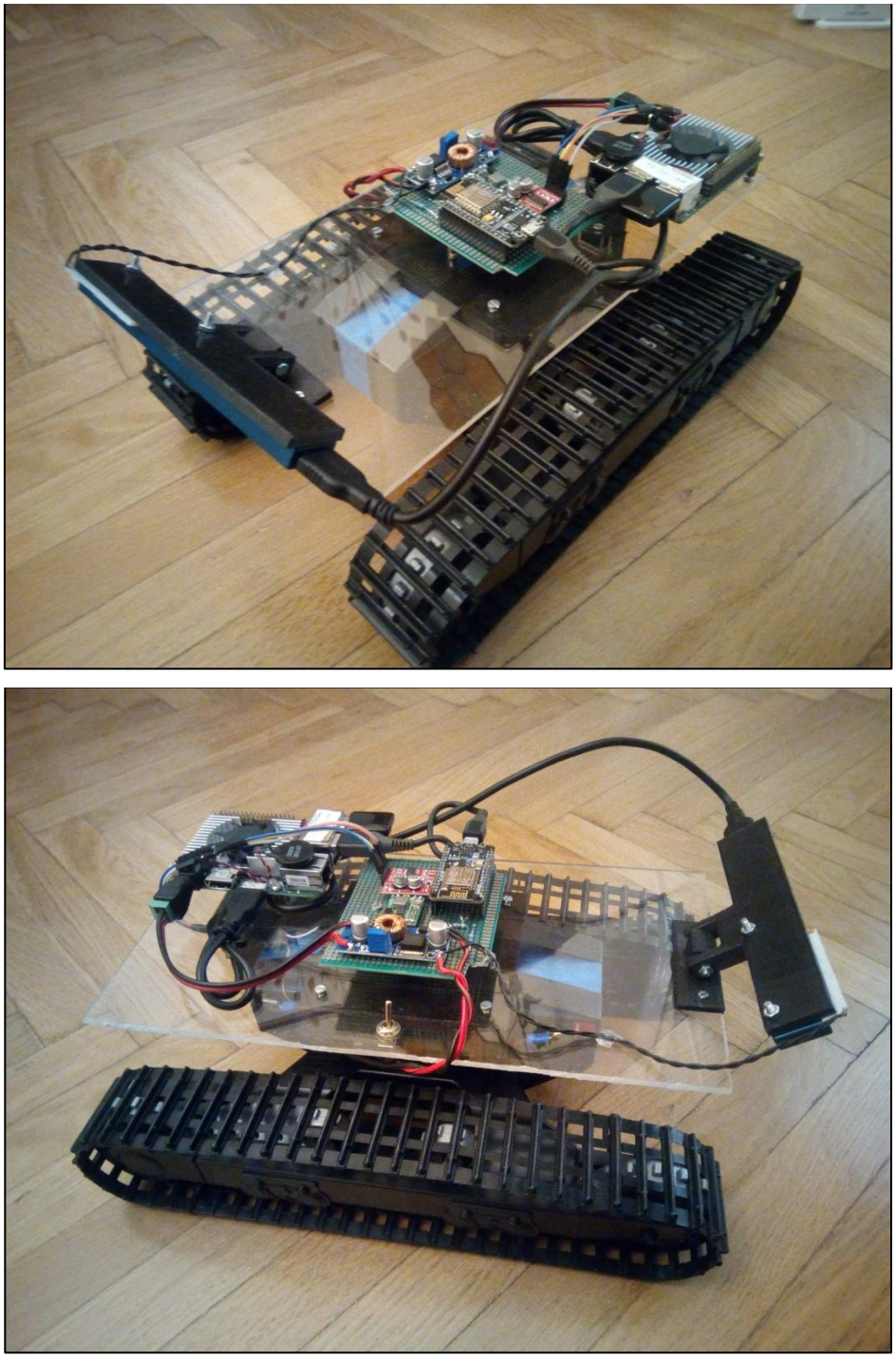
Για την κατασκευή του οχήματος όπως ειπώθηκε προηγουμένως χρειάστηκε να αφαιρεθεί όλο το άνω μέρος του οχήματος και να μείνει μόνο το κάτω του μέρος, αυτό των τροχών με το σασί τους. Παράλληλα, αφαιρέθηκαν και όλα τα εσωτερικά κυκλώματα τα οποία το όχημα διέθετε εκτός από τους κινητήρες των τροχών. Το επόμενο βήμα είναι η στερέωση της μπαταρίας πάνω στο κέντρο του σασί του οχήματος ώστε το κέντρο βάρους του οχήματος να διατηρηθεί χαμηλά και κεντρικά. Αυτό είναι απαραίτητο καθώς η μπαταρία είναι σχετικά ογκώδης και βαριά και αν τοποθετηθεί ψηλότερα υπάρχει πιθανότητα το όχημα να αναποδογυρίσει σε κάποια ανωμαλία του δρόμου. Για να στερεωθεί η μπαταρία στο σασί τροποποιήθηκε λίγο το άνω μέρος του σασί ώστε αυτή να εγκλωβιστεί και να προσανατολιστεί παράλληλα με το όχημα και χρησιμοποιήθηκαν διπλής όψης και μονωτικές ταινίες ώστε να κολληθεί και να πακτωθεί πάνω σε αυτό. Στη συνέχεια, σχεδιάστηκε ένα τρισδιάστατο μοντέλο μιας βάσης και εκτυπώθηκε με τη χρήση του 3D εκτυπωτή Anet A8 με PLA μαύρο πολυμερές υλικό, η οποία αγκαλιάζει το άνω μέρος της μπαταρίας και επάνω σε αυτή στερεώνονται τα κυκλώματα. Για την ακρίβεια τα κυκλώματα στερεώνονται πάνω σε ένα φύλλο πλεξιγκλάς μήκους 29 cm, πλάτους 15 cm και πάχους 0.3 cm, το οποίο με τη σειρά του βιδώνεται κεντρικά πάνω στην πλαστική βάση. Στα πλαίσια της κατασκευής του οχήματος σχεδιάστηκε επίσης και εκτυπώθηκε μια αρθρωτή βάση για την κάμερα πάνω στην οποία κολλήθηκε και η ταινία με τα τρία LEDs. Στην ακόλουθη εικόνα παρουσιάζονται τα δύο τρισδιάστατα σχέδια.



Εικόνα 3.13. *3D μοντέλο της βάσης της κάμερας (αριστερά) και της βάσης στήριξης πλατφόρμας (δεξιά)*

Η βάση της κάμερας και τα LEDs της τοποθετήθηκαν στο εμπρόσθιο μέρος του οχήματος στην άκρη του πλεξιγκλάς και σε γωνία 45 μοιρών ώστε να φωτίζεται επαρκώς και να καταγράφεται η περιοχή ακριβώς μπροστά από το όχημα.

Στην τελική του μορφή το όχημα ζυγίζει 1.8 kg, είναι δηλαδή κατά 400 gr ελαφρύτερο από το αρχικό παρά την προσθήκη της ογκώδους μπαταρίας. Αυτό είναι ένα θετικό στοιχείο καθώς η μείωση του συνολικού βάρους ευνοεί τη μικρή αύξηση της τάσεως τροφοδοσίας των τροχών και αρά της ταχύτητάς τους χωρίς αυτοί όμως να καταστρέφονται. Τέλος, από τις αλλαγές επηρεάστηκαν και τα γεωμετρικά χαρακτηριστικά του οχήματος το οποίο έχει πλέον μήκος 32 cm, πλάτος 22 cm, ύψος 17 cm. Στις ακόλουθες εικόνες παρουσιάζεται η τελική μορφή του αυτόνομου οχήματος.



Εικόνα 3.14. *Τελική μορφή του αυτόνομου οχήματος*

4. Υλοποίηση του Λογισμικού του συστήματος

4.1 Εισαγωγή

Στο παρόν κεφάλαιο περιγράφεται η σχεδίαση και η υλοποίηση του λογισμικού (software) σε όλα τα επίπεδα του ολοκληρωμένου συστήματος. Πιο συγκεκριμένα, περιγράφεται τι είναι ένα ROS σύστημα, το λογισμικό του εξυπηρετητή του συστήματος, η διεπαφή του διαχειριστή του συστήματος, οι αλγόριθμοι όρασης και πλοήγησης του οχήματος καθώς και το λογισμικό του μικροελεγκτή ο οποίος ελέγχει την κίνηση του οχήματος.

4.2 Το λογισμικό του εξυπηρετητή (server)

Ο εξυπηρετητής (server) αποτελεί τον συνδετικό κρίκο μεταξύ του διαχειριστή του συστήματος και των οχημάτων τα οποία βρίσκονται στο πεδίο. Παρά τη μεγάλη σημασία του για το σύστημα, η υλοποίησή του είναι σχετικά απλή.

Αρχικά, ο server είναι γραμμένος στη γλώσσα προγραμματισμού JavaScript λόγω της χρήσης του πολύ γνωστού και ασφαλούς framework Node.js. Πιο συγκεκριμένα, ο server αποτελείται από έναν HTTPS server ο οποίος παρέχει στον διαχειριστή τη διεπαφή όποτε εκείνος τη ζητήσει και παράλληλα διαχειρίζεται το κομμάτι της αυθεντικοποίησης τόσο του διαχειριστή όσο και των οχημάτων του συστήματος. Ο διαχειριστής αυθεντικοποιείται μέσω της αποστολής των στοιχείων username και password του από τη διεπαφή, ενώ τα οχήματα αποστέλλουν την ταυτότητα τους μέσω ενός JSON Web Token (JWT) το οποίο έχει υπογραφεί από τον server και έχει αποθηκευτεί στο αρχείο παραμέτρων του

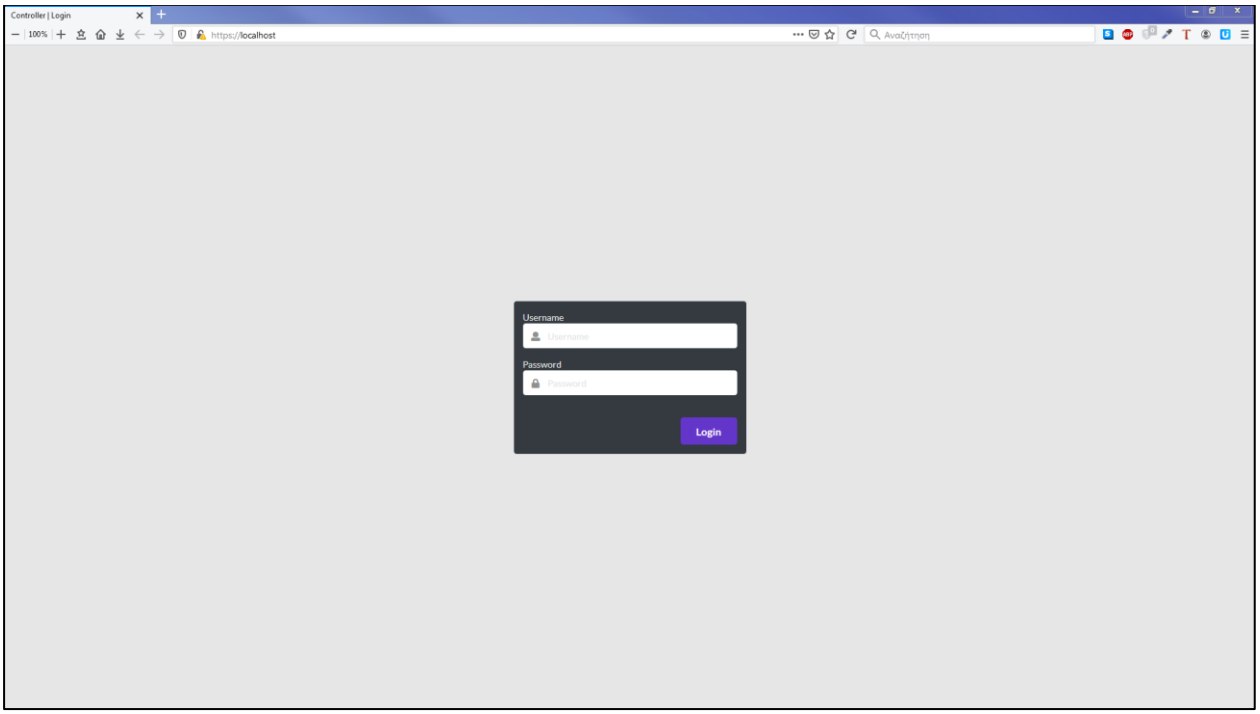
προγράμματος του οχήματος. Να σημειωθεί πως τα στοιχεία των χρηστών διατηρούνται σε μια βάση δεδομένων τύπου SQLite. Μόλις αυθεντικοποιηθούν οι χρήστες, ο HTTPS server τους παρέχει έναν αμφίδρομο τρόπο επικοινωνίας με τον ίδιο μέσω του πρωτοκόλλου WebSocketSecure (WSS) το οποίο υλοποιείται με τη χρήση της βιβλιοθήκης Socket.IO. Η συγκεκριμένη αρχιτεκτονική χρησιμοποιήθηκε διότι σε ένα σύστημα πραγματικού χρόνου, όπως το υλοποιούμενο, οι χρήστες πρέπει να ενημερώνονται άμεσα σε κάθε γεγονός και να γνωρίζουν ο ένας την κατάσταση του άλλου αλλά και την κατάσταση του server και να πράττουν ανάλογα. Για παράδειγμα, αν ο server βρεθεί εκτός λειτουργίας τα οχήματα τα οποία βρίσκονται σε κατάσταση αυτόνομης πλοήγησης αντιλαμβάνονται άμεσα την απώλειά του και σταματούν έως ότου αυτός επανέλθει.

Ένας από τους κύριους ρόλους του server στο σύστημα είναι να ανιχνεύει πότε κάποιος χρήστης συνδέεται ή αποσυνδέεται και να ενημερώνει άμεσα τους υπόλοιπους χρήστες για το συγκεκριμένο γεγονός. Εκτός αυτού, ένας σημαντικός ρόλος του είναι να προωθεί μηνύματα από τον εκάστοτε αποστολέα χρήστη προς τους όποιους παραλήπτες χρήστες, είτε είναι οχήματα ή ο διαχειριστής. Τέλος, είναι υπεύθυνος να ελέγχει θέματα εξουσιοδότησης σε εντολές οι οποίες ανταλλάσσονται μεταξύ των χρηστών ώστε να προλαμβάνει τυχόν κακόβουλες ανεπιθύμητες ενέργειες, όπως για κάποιο λόγο ένα όχημα να δώσει εντολή σε ένα άλλο γεγονός το οποίο απαγορεύεται, καθώς μόνο ο διαχειριστής δύναται να δίνει εντολές στα οχήματα.

4.3 Η διεπαφή του διαχειριστή (admin)

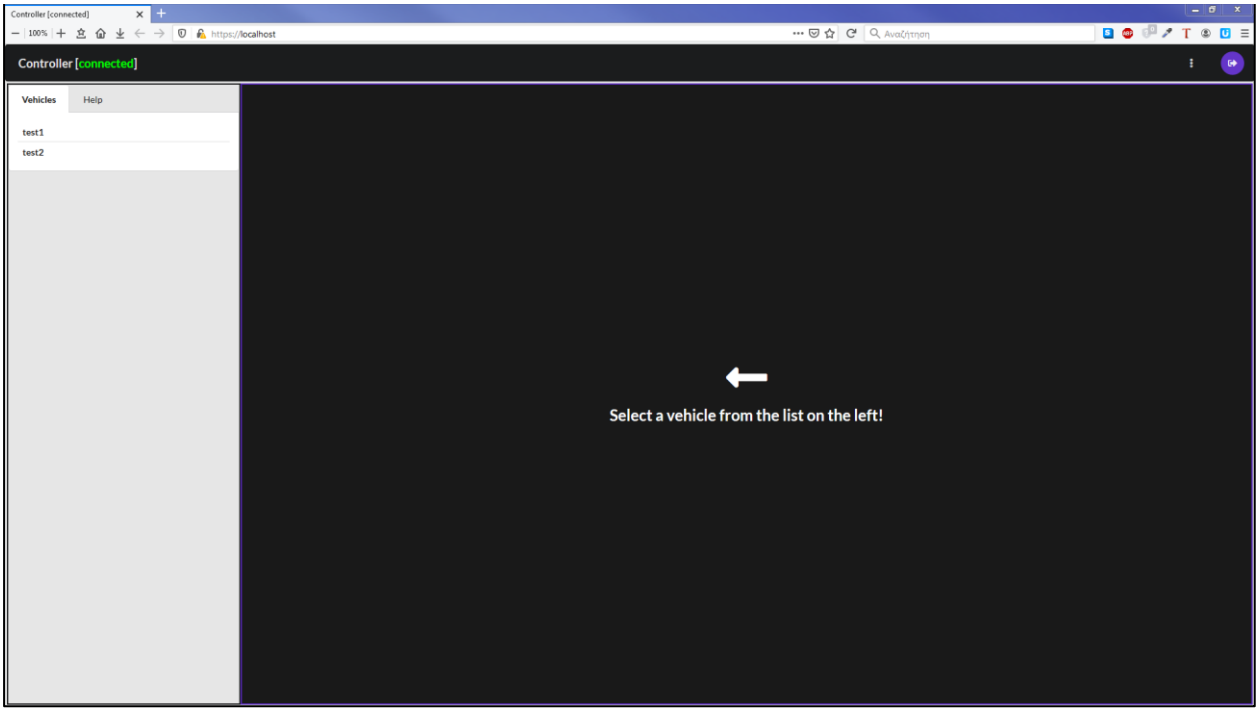
Ο διαχειριστής του συστήματος είναι υπεύθυνος για την καθοδήγηση και την επιτήρηση των οχημάτων. Για να συμβεί όμως αυτό χρειάζεται μια διεπαφή μέσω της οποίας να μπορεί να βλέπει πόσα οχήματα έχουν συνδεθεί στο δίκτυο και ποια είναι η κατάστασή τους καθώς και να μπορεί να τα διαχειριστεί. Για τον λόγο αυτόν στο πλαίσιο της εργασίας υλοποιήθηκε μια διεπαφή την οποία ζητά ο διαχειριστής από τον server. Η συγκεκριμένη διεπαφή υλοποιήθηκε με τη βοήθεια της βιβλιοθήκης Semanti-UI και για τη λειτουργικότητά της έγινε χρήση των JavaScript βιβλιοθηκών jQuery, Cytoscape.js και Socket.IO.

Όταν ο διαχειριστής ζητά πρόσβαση στη διεπαφή, ο server του επιστρέφει την ακόλουθη φόρμα εισόδου ώστε να εισαγάγει τα στοιχεία του.



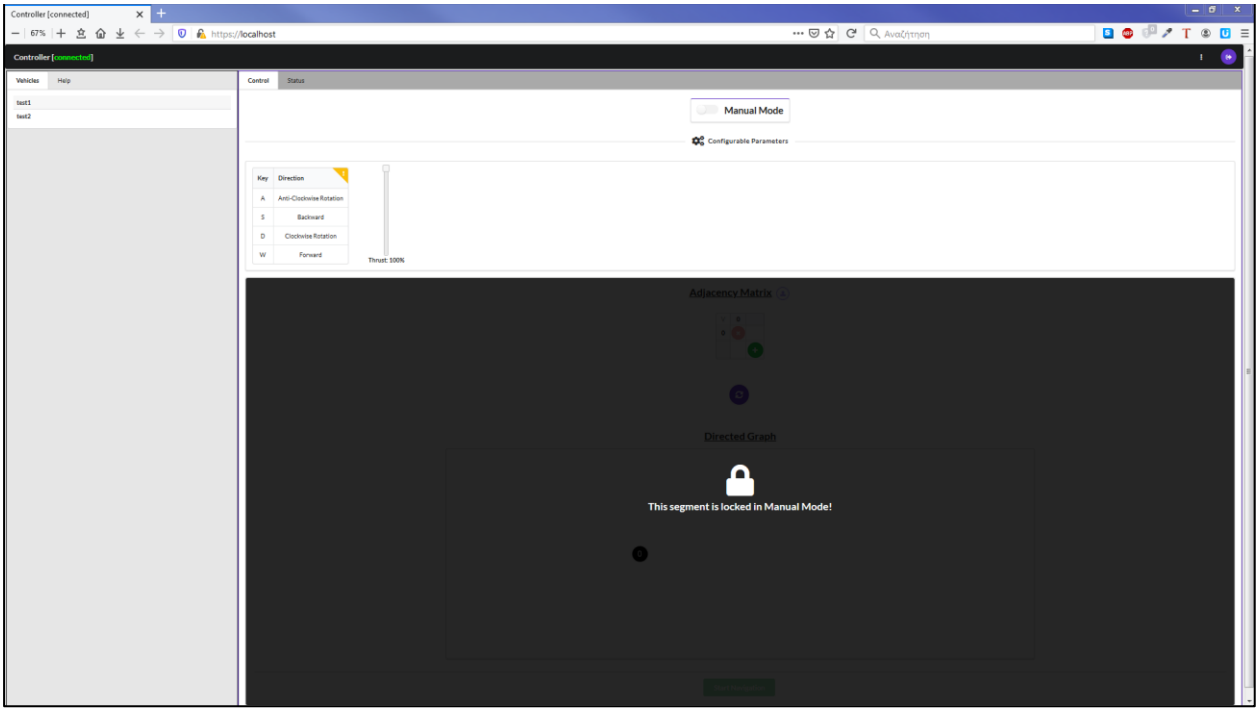
Εικόνα 4.1. *Φόρμα εισόδου της διαχειριστικής διεπαφής*

Μόλις εισαγάγει τα στοιχεία του και αυτά είναι ορθά, ο server τον ανακατευθύνει στην κεντρική σελίδα της διαχειριστικής διεπαφής.

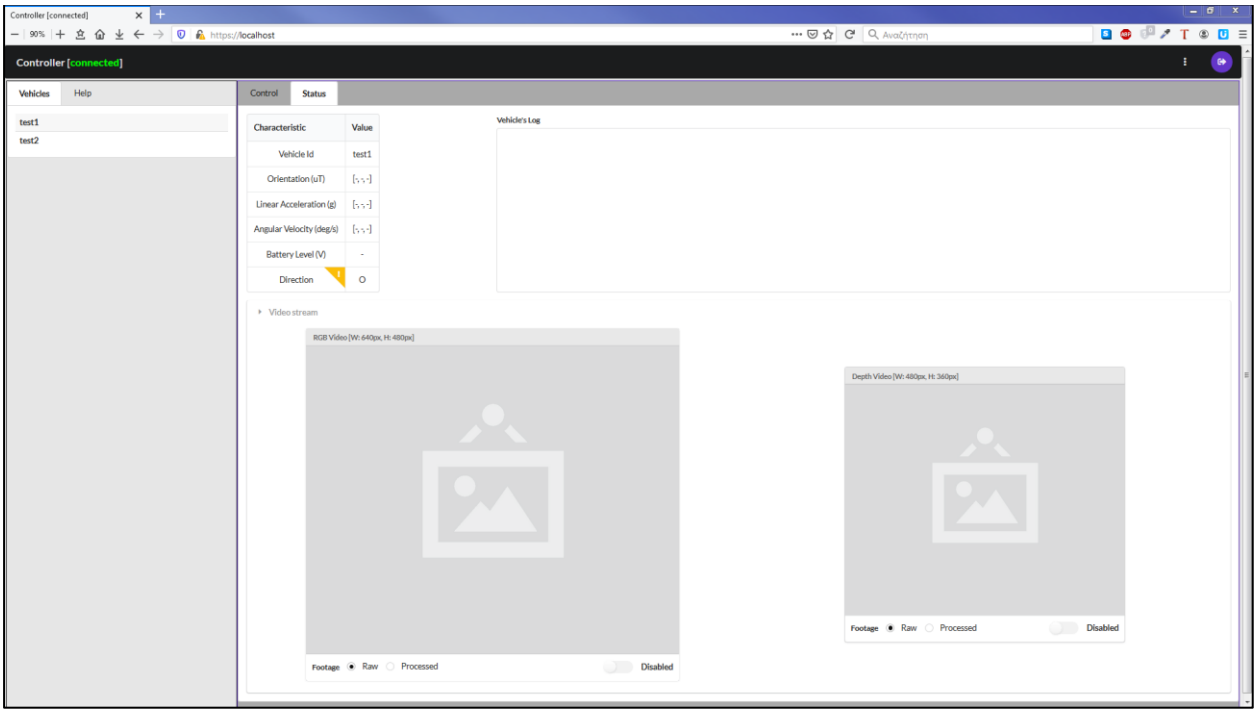


Εικόνα 4.2. *Κεντρική σελίδα διαχειριστικής διεπαφής*

Η συγκεκριμένη διεπαφή αποτελείται από τρία μέρη. Αρχικά αποτελείται από την μπάρα στο άνω μέρος της, η οποία περιλαμβάνει επιλογές και δείχνει αν ο server είναι ενεργός και η διαχειριστική διεπαφή είναι συνδεδεμένη με αυτόν. Επιπλέον αποτελείται από τη λίστα στο αριστερό μέρος της διεπαφής, η οποία περιλαμβάνει όλα τα συνδεδεμένα στο δίκτυο οχήματα από τα οποία μπορεί ο διαχειριστής να επιλέγει όποιο επιθυμεί. Και, τέλος, το μεγαλύτερο της μέρος είναι το παράθυρο διαχείρισης και παρακολούθησης του οχήματος το οποίο ενεργοποιείται όταν επιλεγεί κάποιο όχημα. Το εν λόγω παράθυρο περιλαμβάνει δύο καρτέλες, μία για τον χειρισμό του οχήματος (*Control*), είτε χειροκίνητα ή αυτόνομα, και μία για την παρακολούθηση διαφόρων μεγεθών του οχήματος (*Status*), όπως την τάση της μπαταρίας του, το τι βλέπει η κάμερά του και την αποτύπωση μηνυμάτων για διάφορα γεγονότα που συμβαίνουν εσωτερικά του.



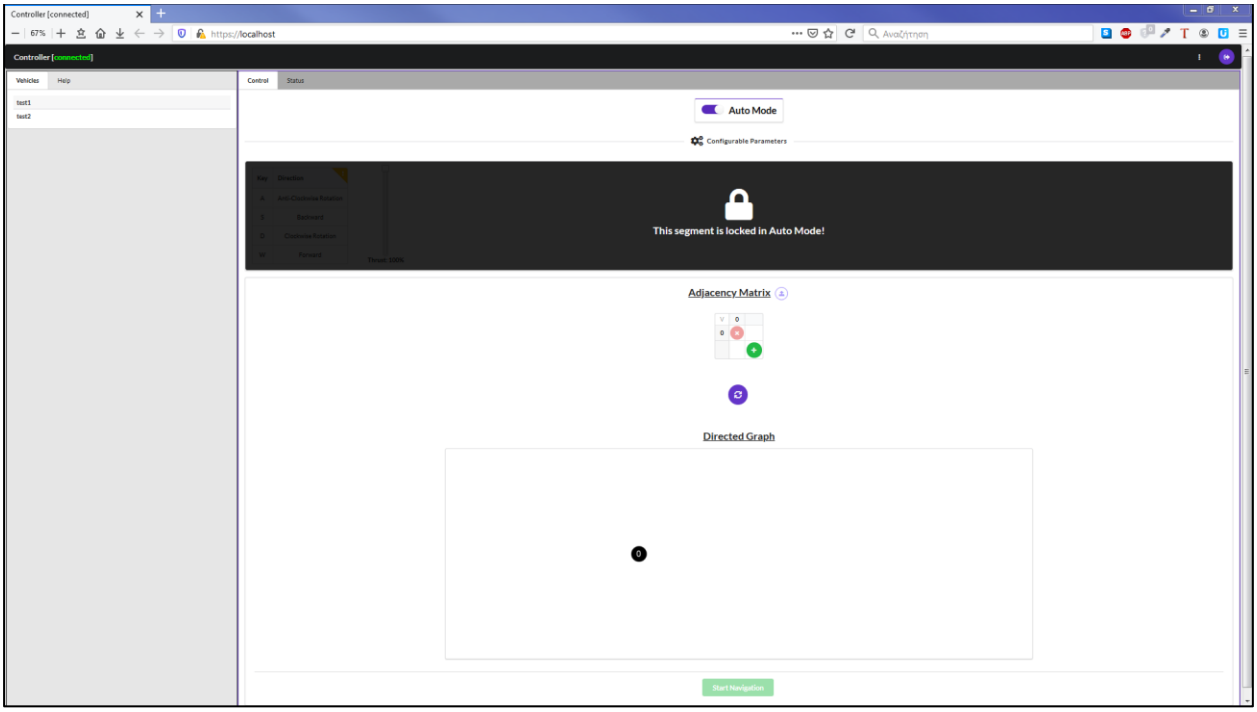
Εικόνα 4.3. *Καρτέλα χειρισμού (Control) οχήματος με επιλεγμένο τον χειροκίνητο χειρισμό*



Εικόνα 4.4. *Καρτέλα παρακολούθησης (Status) οχήματος*

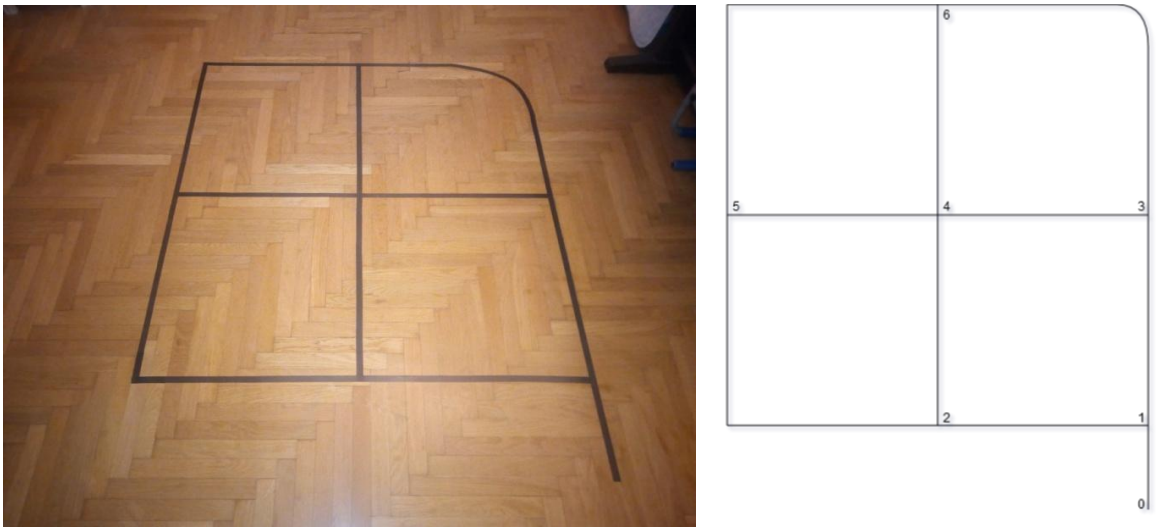
Στην καρτέλα Control ο διαχειριστής μέσω ενός κουμπιού στο άνω μέρος έχει τη δυνατότητα να εναλλάσσει τον χειρισμό του οχήματος από χειροκίνητο σε αυτόνομο. Στον χειροκίνητο χειρισμό μπορεί να χειρίζεται το όχημα μέσω του

πληκτρολογίου με τα αναγραφόμενα πλήκτρα και να ελέγχει ποσοστιαία την ταχύτητα του οχήματος. Τα πράγματα γίνονται πιο σύνθετα κατά την αυτόνομη λειτουργία όπου ο διαχειριστής καλείται να αποτυπώσει σε γράφο τον χώρο στον οποίο κινείται το όχημα.



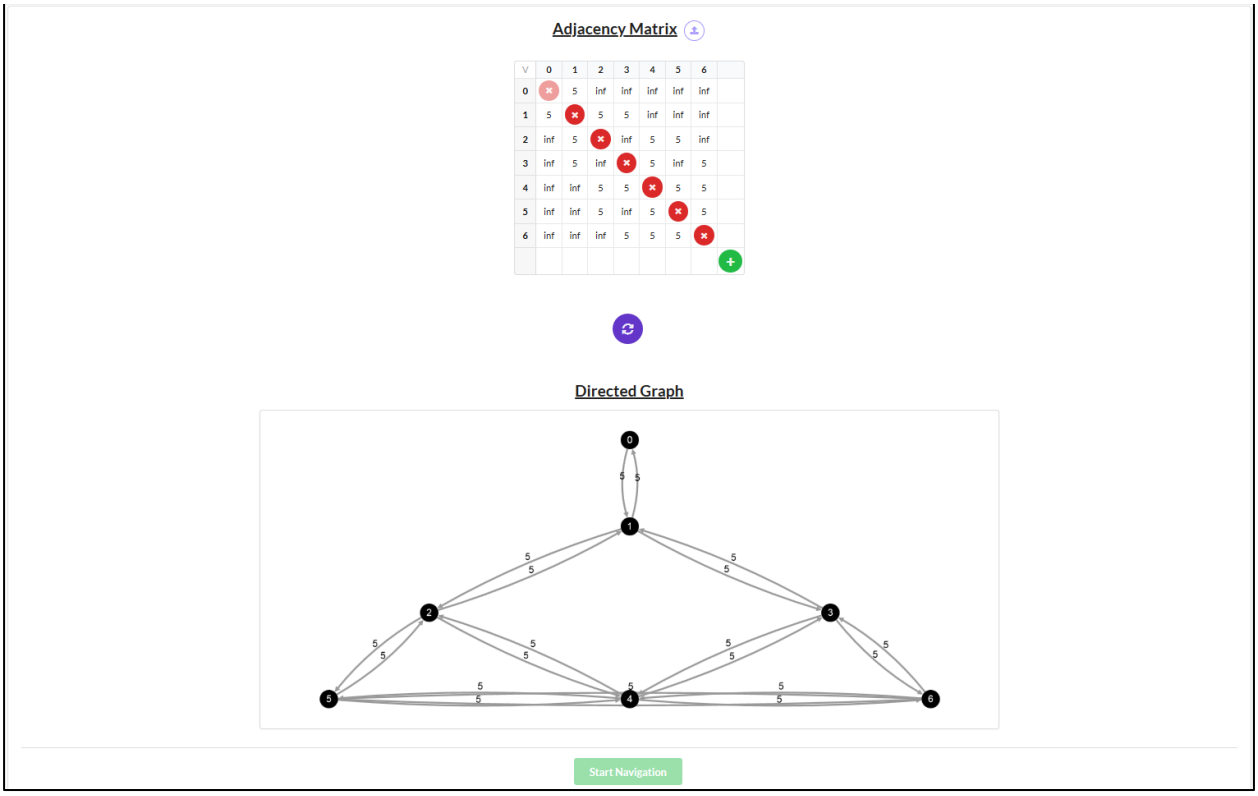
Εικόνα 4.5. *Καρτέλα χειρισμού (Control) οχήματος με επιλεγμένο τον αυτόνομο χειρισμό*

Στην παρούσα υλοποίηση επειδή το όχημα αναζητά σταυροδρόμια όταν πλοηγείται, εύλογη απόρροια είναι πως οι κόμβοι του γράφου τους οποίους συμπληρώνει ο διαχειριστής είναι τα σταυροδρόμια. Έτσι, για παράδειγμα, αν στον ειδικό δρόμο που κατασκευάστηκε για τις ανάγκες της υλοποίησης εντοπιστούν και απαριθμηθούν οι κόμβοι προκύπτει το ακόλουθο αποτέλεσμα.



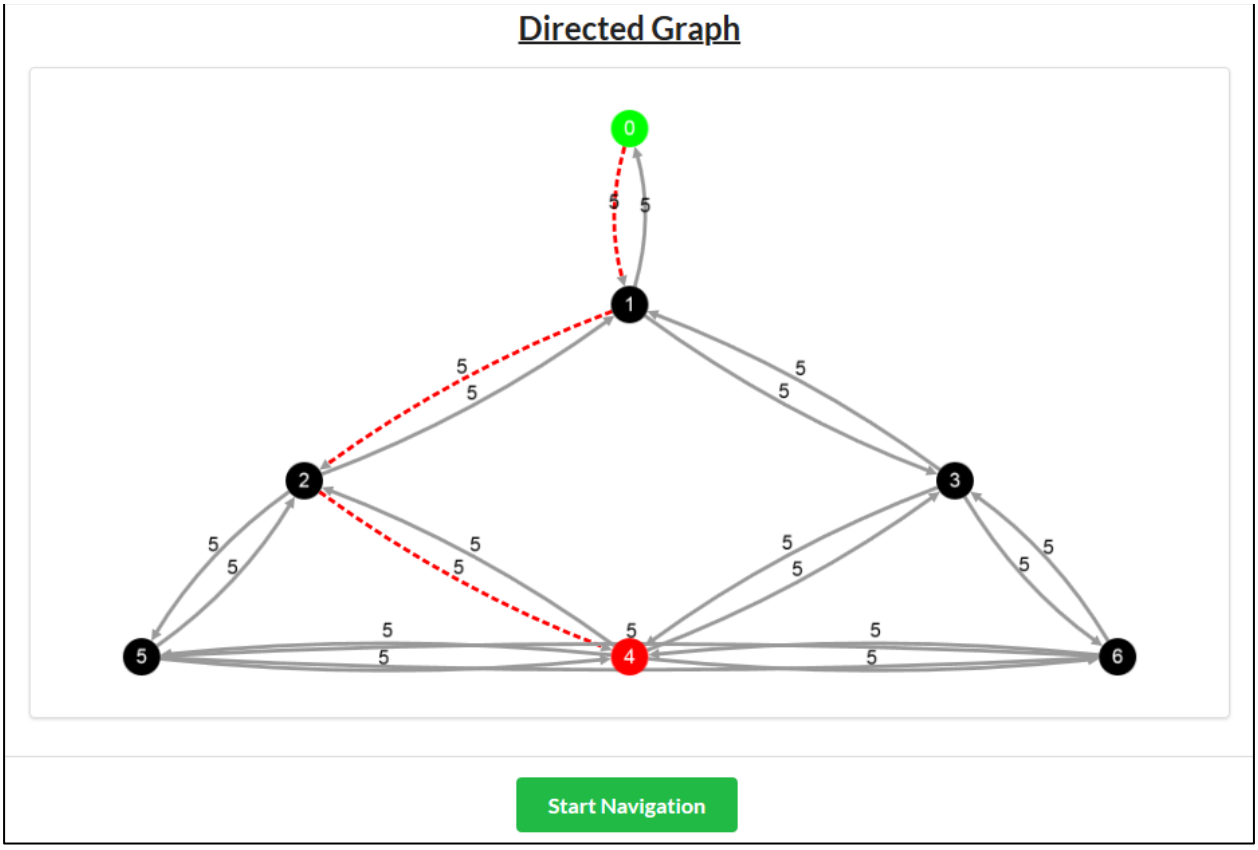
Εικόνα 4.6. *Πραγματική διαδρομή (αριστερά) με τον αντίστοιχο γράφο της (δεξιά)*

Να σημειωθεί πως ως κόμβοι, όπως φαίνεται και από την ανωτέρω αντιστοίχηση, θεωρούνται μόνο τα σταυροδρόμια τα οποία έχουν παραπάνω από μία επιλογές δρόμων. Έχοντας ο διαχειριστής εντοπίσει τους κόμβους μπορεί με δύο τρόπους να τους εισαγάγει στη διεπαφή, είτε πατώντας το κουμπί προσθήκης κόμβου στον πίνακα γειτνίασης για κάθε κόμβο εισάγοντας το βάρος της ακμής μεταξύ των δύο κόμβων ή να φορτώσει στη διεπαφή ένα JSON αρχείο με συμπληρωμένο τον πίνακα γειτνίασης. Όταν ολοκληρώσει τη συμπλήρωση του πίνακα, πατώντας το κουμπί συγχρονισμού πίνακα γειτνίασης και γράφου εμφανίζεται το ακόλουθο αποτέλεσμα. Ο γράφος είναι κατευθυνόμενος και το γραφικό αποτέλεσμα καθορίζεται από το τι εισάγει ο διαχειριστής στον πίνακα γειτνίασης.



Εικόνα 4.7. *Συμπληρωμένος πίνακας γειτνίασης και αντίστοιχος γράφος*

Σε αυτό το σημείο ο διαχειριστής μπορεί να επιλέξει μέσα από μενού κάνοντας δεξί κλικ πάνω στον κόμβο της επιθυμίας του αν θέλει να είναι κόμβος εκκίνησης ή τερματισμού. Όταν επιλεγεί ένας κόμβος εκκίνησης και ένας τερματισμού τότε υπολογίζεται και επισημαίνεται αυτόματα με τη χρήση του αλγορίθμου Dijkstra το ελάχιστο μονοπάτι ανάμεσα στους δύο κόμβους. Μόλις υπολογιστεί το ελάχιστο μονοπάτι τότε δίνεται η δυνατότητα στον διαχειριστή να εκκινήσει την αυτόνομη πλοήγηση.



Εικόνα 4.8. *Γράφος με επισημασμένο το μονοπάτι και επιλογή για έναρξη πλοήγησης*

Απαραίτητη προϋπόθεση για την εκκίνηση και τη σωστή διεξαγωγή της αυτόνομης πλοήγησης είναι ο διαχειριστής να έχει τοποθετήσει το όχημα σε δρόμο ο οποίος βρίσκεται ανάμεσα στον κόμβο εκκίνησης και στον επόμενο του και με κατεύθυνση προς τον δεύτερο, διότι στην παρούσα υλοποίηση το όχημα δεν δύναται να ταυτοποιεί τα σταυροδρόμια τα οποία συναντά, παρά μόνο να ελέγχει αν αυτά έχουν τον σωστό αριθμό κατευθύνσεων. Μόλις ο διαχειριστής εκκινήσει την πλοήγηση, η αλληλουχία των κόμβων του μονοπατιού αποστέλλεται στο όχημα το οποίο με τη σειρά του εκκινεί την πλοήγηση και η διεπαφή μεταβαίνει αυτόματα στην καρτέλα Status.

Στην καρτέλα Status, όπως παρουσιάστηκε προηγουμένως, εμπεριέχεται ένας πίνακας στον οποίο αποτυπώνονται τα στοιχεία και οι μετρήσεις που επιστρέφει το όχημα. Επιπλέον, υπάρχει μια περιοχή απεικόνισης κειμένου (Vehicle's Log) στην οποία εμφανίζονται μηνύματα στις εναλλαγές καταστάσεων του οχήματος κατά τη διάρκεια της αυτόνομης πλοήγησης. Τέλος, υπάρχει και το τμήμα στο οποίο ο

διαχειριστής μπορεί να βλέπει είτε επεξεργασμένη ή όχι την RGB εικόνα και την εικόνα βάθους από την κάμερα του οχήματος. Η καρτέλα Status παρουσιάζεται συμπληρωμένη με πραγματικά στοιχεία αποτυπωμένα στο κεφάλαιο που αφορά στη δοκιμή του συστήματος.

4.4 Το λογισμικό της επεξεργαστικής μονάδας του οχήματος

4.4.1 Robot Operating System (ROS)

Το Ρομποτικό Λειτουργικό Σύστημα (ROS ή ros) αποτελεί ένα ενδιάμεσο λογισμικό ρομποτικής, δηλαδή μια συλλογή frameworks για την ανάπτυξη ρομποτικών εφαρμογών [2]. Παρόλο που το ROS δεν είναι λειτουργικό σύστημα, παρέχει υπηρεσίες σχεδιασμένες για ένα ετερογενές σύμπλεγμα υπολογιστών, όπως αφαιρετικές μεθόδους για το υλικό, έλεγχο συσκευών χαμηλού επιπέδου, ενσωμάτωση κοινώς χρησιμοποιούμενων λειτουργιών, μετάδοση μηνυμάτων μεταξύ διαδικασιών και διαχείριση πακέτων. Τα ενεργά σύνολα διαδικασιών που βασίζονται στο ROS αναπαρίστανται σε μια αρχιτεκτονική γραφημάτων, όπου η επεξεργασία λαμβάνει χώρα σε κόμβους που μπορούν να λαμβάνουν, να αποστέλλουν και να πολυπλέκουν διαφόρων τύπων μηνύματα που αφορούν σε δεδομένα αισθητήρων, στον έλεγχο, στην κατάσταση, στη σχεδίαση, στην ενεργοποίηση και σε άλλα. Παρά τη σημασία της άμεσης αντίδρασης και της χαμηλής καθυστέρησης στις ρομποτικές εφαρμογές, το ίδιο το ROS δεν είναι λειτουργικό σύστημα πραγματικού χρόνου (RTOS). Ωστόσο, είναι δυνατή η ενσωμάτωση του ROS με κώδικα πραγματικού χρόνου. Η έλλειψη υποστήριξης για συστήματα πραγματικού χρόνου αναμένεται να αντιμετωπιστεί με τη δημιουργία του ROS 2.0, μια σημαντική αναθεώρηση του ROS API, το οποίο θα επωφελείται από τις σύγχρονες βιβλιοθήκες και τεχνολογίες για τη βασική λειτουργικότητα του

ROS και θα προσθέτει υποστήριξη για κώδικα πραγματικού χρόνου και ενσωματωμένου υλικού.

Από τις πιο συχνές ερωτήσεις που σχετίζονται με το ROS είναι η σύγκριση του ROS με άλλες ρομποτικές πλατφόρμες λογισμικού (OpenRTM, OPRoS, Player, YARP, Orocos, CARMEN, Orca, MOOS, Microsoft Robotics Studio) [3]. Αν και θα μπορούσε να γίνει μια τέτοια σύγκριση, δεν θα είχε ιδιαίτερο νόημα καθώς κάθε ένα από αυτά εξυπηρετεί διαφορετικούς σκοπούς. Το ROS επικεντρώνεται στη μεγιστοποίηση της επαναχρησιμοποίησης κώδικα στην έρευνα και στην ανάπτυξη της ρομποτικής, αντί να στοχεύει στο να αποτελέσει ρομποτική πλατφόρμα λογισμικού, ενδιάμεσο λογισμικό ή framework. Για να το υποστηρίξει αυτό, το ROS διαθέτει τα ακόλουθα χαρακτηριστικά.

- Κατανεμημένη διεργασία: Είναι προγραμματισμένο με τη λογική ελαχίστου αριθμού μονάδων εκτελέσιμων διεργασιών (κόμβοι) ώστε κάθε διεργασία να εκτελείται ανεξάρτητα και να ανταλλάσσει δεδομένα συστηματικά.
- Διαχείριση πακέτου: Πολλαπλές διεργασίες οι οποίες επιτελούν τον ίδιο σκοπό υφίστανται διαχείριση ως πακέτο, έτσι ώστε να είναι εύκολες στη χρήση και στην ανάπτυξη καθώς και στον διαμοιρασμό, στην τροποποίηση και στην αναδιανομή.
- Δημόσιο αποθετήριο: Κάθε πακέτο δημοσιοποιείται στα προτιμώμενα δημόσια αποθετήρια των προγραμματιστών (π.χ. GitHub) με καθορισμένη την άδεια χρήσης του.
- API: Κατά την ανάπτυξη ενός προγράμματος που χρησιμοποιεί το ROS, το ROS έχει σχεδιαστεί για να καλεί απλά ένα API και να το εισάγει εύκολα στον κώδικα που χρησιμοποιείται. Στον πηγαίο κώδικα ο προγραμματισμός στο ROS δεν είναι πολύ διαφορετικός μεταξύ της C++ και της Python.
- Υποστήριξη διαφόρων γλωσσών προγραμματισμού: Το πρόγραμμα ROS παρέχει μια βιβλιοθήκη με υποστήριξη σε διάφορες γλώσσες προγραμματισμού. Η βιβλιοθήκη αυτή μπορεί να εισαχθεί σε γλώσσες προγραμματισμού που είναι δημοφιλείς στον τομέα της ρομποτικής, όπως

οι Python, C++ και Lisp καθώς και σε γλώσσες όπως οι JAVA, C#, Lua και Ruby. Με άλλα λόγια, ένα πρόγραμμα ROS μπορεί να αναπτυχθεί χρησιμοποιώντας όποια γλώσσα προγραμματισμού προτιμά ο προγραμματιστής.

Αυτά τα χαρακτηριστικά του ROS έχουν επιτρέψει στους χρήστες να δημιουργήσουν ένα περιβάλλον όπου είναι δυνατόν να συνεργαστούν για την ανάπτυξη λογισμικού ρομποτικής σε παγκόσμιο επίπεδο. Η επαναχρησιμοποίηση ενός κώδικα στην έρευνα και στην ανάπτυξη της ρομποτικής γίνεται όλο και πιο συνήθης, κάτι που αποτελεί τον τελικό στόχο του ROS.

Ο Γράφος Υπολογισμού είναι το δίκτυο peer-to-peer των διεργασιών ROS που επεξεργάζονται δεδομένα μαζί [4]. Βασικές έννοιες του Γράφου Υπολογισμού του ROS είναι οι κόμβοι (nodes), ο Master, τα μηνύματα (messages), οι υπηρεσίες (services) και τα θέματα (topics), τα οποία παρέχουν όλα δεδομένα στον Γράφο με διάφορους τρόπους. Αυτές οι έννοιες υλοποιούνται στο αποθετήριο ros_comm.

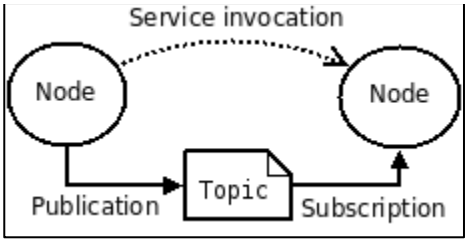
- **Κόμβοι (Nodes):** Οι κόμβοι είναι διεργασίες που εκτελούν υπολογισμό. Το ROS είναι σχεδιασμένο ώστε να είναι αρθρωτό σε λεπτή κλίμακα, όπου ένα ρομποτικό σύστημα ελέγχου συνήθως περιλαμβάνει πολλούς κόμβους. Για παράδειγμα, κάποιος κόμβος μπορεί να ελέγχει τους κινητήρες των τροχών, κάποιος κόμβος να εκτελεί εντοπισμό, κάποιος κόμβος να εκτελεί σχεδιασμό διαδρομής, κάποιος κόμβος να παρέχει γραφική απεικόνιση του συστήματος κ.ο.κ. Ένας κόμβος ROS γράφεται με τη χρήση μιας βιβλιοθήκης πελατών ROS, όπως είναι οι roscpp ή rospy.
- **Master:** Ο ROS Master παρέχει καταχώριση και αναζήτηση ονομάτων στον υπόλοιπο Γράφο Υπολογισμού. Χωρίς τον Master, οι κόμβοι δεν θα μπορούσαν να βρουν ο ένας τον άλλο, να ανταλλάξουν μηνύματα ή να επικαλεστούν υπηρεσίες.
- **Μηνύματα (Messages):** Οι κόμβοι επικοινωνούν μεταξύ τους μεταδίδοντας μηνύματα. Ένα μήνυμα είναι απλώς μια δομή δεδομένων, που περιλαμβάνει πεδία τύπων. Υποστηρίζονται οι πρότυποι τύποι αρχεγόνου (ακέραιος,

κινητής υποδιαστολής, boolean, κλπ.), όπως και οι πίνακες τύπων αρχεγόνου. Τα μηνύματα μπορούν να περιλαμβάνουν αυθαίρετες εμφωλευμένες δομές και πίνακες (όπως οι δομές C).

- **Θέματα (Topics):** Τα μηνύματα δρομολογούνται μέσω ενός συστήματος μεταφοράς με σημασιολογία δημοσίευσης/εγγραφής. Ένας κόμβος στέλνει ένα μήνυμα δημοσιεύοντάς το σε ένα δεδομένο θέμα. Το θέμα είναι ένα όνομα που χρησιμοποιείται για την αναγνώριση του περιεχομένου του μηνύματος. Ένας κόμβος που ενδιαφέρεται για ένα συγκεκριμένο είδος δεδομένων θα εγγραφεί στο κατάλληλο θέμα. Ενδέχεται να υπάρχουν πολλοί ταυτόχρονοι εκδότες (publishers) και συνδρομητές (subscribers) για ένα μόνο θέμα ή ένας μόνο κόμβος ενδέχεται να δημοσιεύσει και/ή να εγγραφεί σε πολλά θέματα. Σε γενικές γραμμές, οι εκδότες και οι συνδρομητές δεν γνωρίζουν την ύπαρξη άλλων εκδοτών και συνδρομητών. Η ιδέα είναι να αποσυνδεθεί η παραγωγή πληροφοριών από την κατανάλωσή της. Θα μπορούσε κανείς να φανταστεί ένα θέμα ως έναν δίαυλο πληκτρολογημένων μηνυμάτων. Κάθε δίαυλος έχει ένα όνομα και ο καθένας μπορεί να συνδεθεί σε αυτόν για να αποστείλει ή να λάβει μηνύματα εφόσον είναι του σωστού τύπου.
- **Υπηρεσίες (Services):** Το μοντέλο δημοσίευσης/εγγραφής αποτελεί ένα πολύ ευέλικτο μοντέλο επικοινωνίας, αλλά η μονόδρομη επικοινωνία πολλών αποστολέων προς πολλούς συνδρομητές σε πολλές περιπτώσεις δεν είναι κατάλληλη για αλληλεπιδράσεις μεταξύ αιτήσεων/απαντήσεων, οι οποίες απαιτούνται συχνά σε ένα κατανεμημένο σύστημα. Η αίτηση/απόκριση γίνεται μέσω υπηρεσιών, οι οποίες καθορίζονται από ένα ζεύγος μηνυμάτων: ένα για την αίτηση και ένα για την απόκριση. Ένας κόμβος παροχής προσφέρει μια υπηρεσία με ένα όνομα και ένας πελάτης χρησιμοποιεί την υπηρεσία αποστέλλοντας το μήνυμα αίτησης και περιμένοντας την απόκριση. Οι βιβλιοθήκες πελάτη ROS παρουσιάζουν

γενικά αυτήν την αλληλεπίδραση στον προγραμματιστή σαν να ήταν κλήση απομακρυσμένης διαδικασίας.

Ο ROS Master ενεργεί ως υπηρεσία ονομάτων στον Γράφο Υπολογισμού ROS. Αποθηκεύει πληροφορίες καταχώρησης θεμάτων και υπηρεσιών για κόμβους ROS. Οι κόμβοι επικοινωνούν με τον Master για να αναφέρουν τα στοιχεία καταχώρησής τους. Καθώς αυτοί οι κόμβοι επικοινωνούν με τον Master, μπορούν να λαμβάνουν πληροφορίες σχετικά με άλλους καταχωρημένους κόμβους και να κάνουν συνδέσεις ανάλογα με την περίπτωση. Ο Master θα κάνει επίσης επανακλήσεις σε αυτούς τους κόμβους όταν αλλάξουν αυτές οι πληροφορίες καταχώρησης, γεγονός που επιτρέπει στους κόμβους να δημιουργούν δυναμικά συνδέσεις καθώς προστίθενται νέοι κόμβοι. Οι κόμβοι συνδέονται άμεσα με άλλους κόμβους. Ο Master παρέχει μόνο πληροφορίες αναζήτησης, όπως ένας εξυπηρετητής DNS. Οι κόμβοι που εγγράφονται σε ένα θέμα θα ζητήσουν συνδέσεις από κόμβους που δημοσιεύουν αυτό το θέμα και θα καθορίσουν τη σύνδεση αυτή μέσω ενός συμφωνηθέντος πρωτοκόλλου σύνδεσης. Το πιο κοινό πρωτόκολλο που χρησιμοποιείται σε ένα ROS ονομάζεται TCPROS, το οποίο χρησιμοποιεί τυπικές υποδοχές TCP/IP. Αυτή η αρχιτεκτονική επιτρέπει την αποσυνδεδεμένη λειτουργία, όπου τα ονόματα είναι ο κύριος τρόπος με τον οποίο μπορούν να κατασκευαστούν μεγαλύτερα και πιο σύνθετα συστήματα. Τα ονόματα έχουν πολύ σημαντικό ρόλο στο ROS: οι κόμβοι, τα θέματα, οι υπηρεσίες και οι παράμετροι έχουν όλοι ονόματα. Κάθε ROS βιβλιοθήκη πελάτη υποστηρίζει την εκ νέου αντιστοίχιση μέσω της γραμμής εντολών των ονομάτων, πράγμα που σημαίνει ότι ένα μεταγλωττισμένο πρόγραμμα μπορεί να αναδιαμορφωθεί κατά τη διάρκεια της εκτέλεσης για να λειτουργήσει σε μια διαφορετική τοπολογία Γράφου Υπολογισμού.



Εικόνα 4.9. *Διάγραμμα βασικών ROS εννοιών [4]*

4.4.2 Διαδικασία ρύθμισης της επεξεργαστικής μονάδας

Στην παρούσα παράγραφο παρουσιάζεται η διαδικασία ρύθμισης της πλακέτας UP κατά την οποία εγκαθίσταται σε αυτή το λειτουργικό σύστημα Ubuntu Xenial 16.04 LTS, το ROS Kinetic οικοσύστημα και τα απαραίτητα ROS Kinetic πακέτα ώστε να χρησιμοποιηθεί η κάμερα R200 και ο μικροελεγκτής ESP8266 ως ROS κόμβοι.

Η επιλογή του λειτουργικού συστήματος Ubuntu Xenial δεν είναι τυχαία καθώς για να ενσωματωθεί η κάμερα, η νεότερη υποστηριζόμενη ROS έκδοση η οποία πρέπει να χρησιμοποιηθεί είναι η Kinetic [5] η οποία υποστηρίζεται μέχρι και τη συγκεκριμένη έκδοση του λειτουργικού συστήματος Ubuntu [6]. Για την εγκατάσταση του λειτουργικού συστήματος στην πλακέτα UP ακολουθούνται τα εξής βήματα [7] [8]:

1. Δημιουργείται ένας σκληρός δίσκος USB με δυνατότητα εκκίνησης (bootable) με το .ISO αρχείο του λειτουργικού συστήματος.
2. Συνδέεται ο bootable δίσκος μαζί με περιφερειακές συσκευές όπως πληκτρολόγιο και ποντίκι στις ελεύθερες USB2.0 θύρες και η οθόνη στην HDMI θύρα.
3. Στη συνέχεια τροφοδοτείται και ενεργοποιείται η πλακέτα UP.
4. Αφού εμφανιστεί μετά από λίγα δευτερόλεπτα το λογότυπο της εταιρίας UP στην οθόνη, εμφανίζεται αμέσως μετά το μενού της εγκατάστασης του λειτουργικού συστήματος.
5. Επιλέγεται η δεύτερη επιλογή "*Install Ubuntu*" η οποία εκκινεί τη διαδικασία της εγκατάστασης.

6. Στις επόμενες οθόνες επιλέγονται με τη σειρά οι ακόλουθες επιλογές:
    - **Welcome:** Επιλέγεται η προεπιλεγμένη γλώσσα "*English*".
    - **Preparing to install Ubuntu:** Δεν επιλέγεται καμία από τις δύο επιλογές. Η διαδικασία ενημέρωσης του λειτουργικού γίνεται αργότερα.
    - **Installation type:** Επιλέγεται η προεπιλεγμένη επιλογή "*Erase disk and install Ubuntu*".
    - **Where are you?, Keyboard layout, Who are you?:** Στις συγκεκριμένες οθόνες εισάγονται προσωπικές επιλογές.
    - Τέλος, μόλις ολοκληρωθεί η διαδικασία της εγκατάστασης επιλέγεται η επιλογή "*Restart Now*" όπου αναμένεται η εμφάνιση του λογότυπου της εταιρίας UP.
7. Αποσυνδέεται από την τροφοδοσία η UP πλακέτα, έπειτα ο bootable δίσκος από τη USB θύρα της, συνδέεται η USB κεραία Wi-Fi και επανασυνδέεται στην τροφοδοσία η UP πλακέτα.
8. Εφόσον εκκινήσει το λειτουργικό σύστημα, για να υπάρχει πρόσβαση στο Internet, αρχικά, συνδέεται η πλακέτα UP μέσω Ethernet καλωδίου στο Internet ώστε να εγκατασταθεί ο οδηγός (driver) της εξωτερικής Wi-Fi κεραίας. Αφού εγκατασταθεί ο οδηγός μέσω ενός τερματικού παραθύρου, αφαιρείται το καλώδιο Ethernet και συνδέεται η εξωτερική ασύρματη Wi-Fi κεραία σε γνωστό δίκτυο.
9. Ακολούθως και πριν την εγκατάσταση του πυρήνα (kernel) της εταιρίας UP ενημερώνεται το λειτουργικό σύστημα μέσω των ακόλουθων εντολών:
    - *sudo apt update*
    - *sudo apt -y dist-upgrade*
10. Αφού ενημερωθεί το λειτουργικό σύστημα, εισάγονται οι ακόλουθες εντολές στο τερματικό για να αντικατασταθεί ο προεπιλεγμένος πυρήνας από αυτόν της upboard:
    - *sudo add-apt-repository ppa:ubilinux/up*
    - *sudo apt update*

- *sudo apt -y install linux-upboard*
- *sudo apt -y autoremove --purge 'linux-.*generic'*

11. Επανεκκινώντας το λειτουργικό σύστημα και στη συνέχεια εκτελώντας την εντολή "*uname -r*" αν όλα έχουν πάει καλά θα πρέπει να εμφανιστεί το όνομα του νέου πυρήνα "*4.4.0.2-upboard*".

Εφόσον εγκαταστάθηκε, ενημερώθηκε και ρυθμίστηκε το λειτουργικό σύστημα της UP πλακέτας, σειρά έχει η εγκατάσταση του ROS οικοσυστήματος και ορισμένων πακέτων του. Από τη στιγμή που δημιουργήθηκε ο upboard kernel έχουν κυκλοφορήσει πολλαπλές διορθώσεις του ώστε να ενεργοποιηθούν περισσότερες λειτουργίες για την πλατφόρμα RealSense. Για τον λόγο αυτόν πριν την εγκατάσταση του ROS και των RealSense πακέτων του είναι υποχρεωτικό να ενημερωθεί ο πυρήνας. Για να επιτευχτεί αυτό εκτελείται η ακόλουθη εντολή ώστε να ενημερωθούν τα πηγαία αρχεία του apt:

- *wget -q -O - https://raw.githubusercontent.com/IntelRealSense/librealsense/rosdebian/scripts/enable_kernel_sources.sh\ | sudo /bin/bash*

Στη συνέχεια για την εγκατάσταση του ROS Kinetic και των πακέτων RealSense εκτελούνται οι ακόλουθες εντολές:

- *sudo add-apt-repository http://packages.ros.org/ros/ubuntu*
- *sudo apt-key adv --keyserver hkp://ha.pool.sks-keyserver.net --recv-key 0xB01FA116*
- *sudo apt update*
- *sudo apt -y install ros-kinetic-desktop-full python-rosinstall ros-kinetic-realsense-camera*
- *sudo rosdep init*
- *rosdep update*
- *echo "source /opt/ros/kinetic/setup.bash" >> ~/.bashrc*
- *source ~/.bashrc*

- *sudo apt-get install ros-kinetic-rosserial ros-kinetic-rosserial-python* για τη μετέπειτα επικοινωνία με τον μικροελεγκτή ESP8266 μέσω του πρωτοκόλλου σειριακής επικοινωνίας [9].

Για να λάβουν χώρα οι παραπάνω τροποποιήσεις χρειάζεται να γίνει μια επανεκκίνηση στο λειτουργικό σύστημα.

4.4.3 Το πρόγραμμα της επεξεργαστικής μονάδας

Στην παρούσα ενότητα παρουσιάζεται η δομή και η λειτουργία του προγράμματος της επεξεργαστικής μονάδας UP. Το συγκεκριμένο λογισμικό αποτελεί την καρδιά του αυτόνομου οχήματος καθώς είναι υπεύθυνο για τη συλλογή εικόνων, μετρήσεων και, όπως φαίνεται ακολούθως, τη λήψη αποφάσεων κατά την πορεία του.

Αρχικά, το συγκεκριμένο πρόγραμμα αποτελείται από τρία διαφορετικά υποπρογράμματα τα οποία αποτελούν μια κοινοπραξία κάτω από την ομπρέλα του οικοσυστήματος ROS. Πιο συγκεκριμένα, τα τρία αυτά υποπρογράμματα αποτελούν τρεις διαφορετικούς κόμβους του ROS συστήματος οι οποίοι επικοινωνούν μεταξύ τους μέσω Publishers και Subscribers σε συγκεκριμένα topics. Στις ακόλουθες παραγράφους παρουσιάζοντα οι ROS κόμβοι, οι Subscribers και οι Publishers τους και η λογική που διέπει το κάθε υποπρόγραμμα.

Το πρώτο υποπρόγραμμα είναι γραμμένο στη γλώσσα προγραμματισμού JavaScript λόγω της χρήσης του framework Node.js και αποτελεί την απαρχή του συνολικού προγράμματος. Η επιλογή του συγκεκριμένου framework είναι συνειδητή καθώς επικοινωνεί άμεσα με τον server ο οποίος είναι υλοποιημένος με την ίδια τεχνολογία και πρωτόκολλα όπως το Socket.IO υλοποιούνται πιο εύκολα όταν και τα δύο άκρα χρησιμοποιούν το ίδιο framework. Το συγκεκριμένο υποπρόγραμμα είναι υπεύθυνο, αρχικά, για την εκκίνηση του ROS Master, ο οποίος συντονίζει όλους τους ROS κόμβους, και, έπειτα, για την εκκίνηση των ROS κόμβων της κάμερας και του μικροελεγκτή. Επιπλέον, είναι υπεύθυνο για την

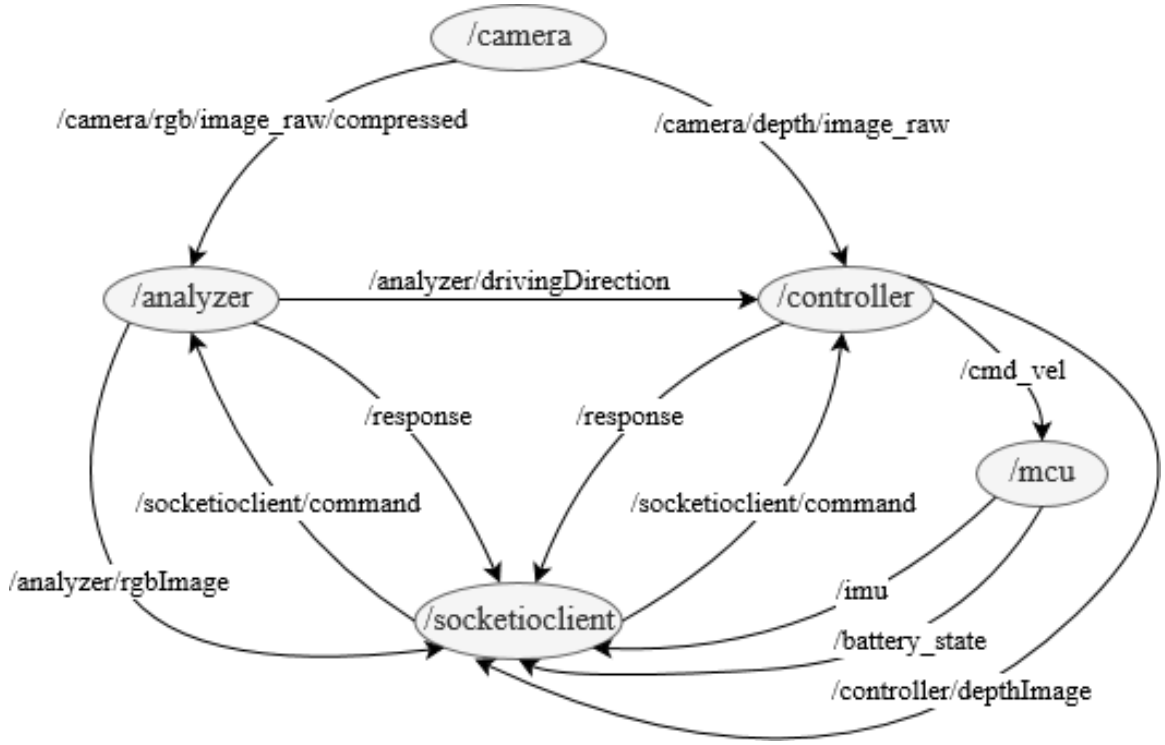
εκκίνηση των άλλων δύο κόμβων και τη μεταβίβαση μηνυμάτων μεταξύ αυτών και του διαχειριστή. Τέλος, είναι υπεύθυνο για τη σύνδεση του οχήματος στο δίκτυο και την επικοινωνία του με τον εξυπηρετητή και τον διαχειριστή. Πιο συγκεκριμένα, όταν εκκινεί το πρόγραμμα εκκινεί τον ROS Master ώστε να μπορούν στη συνέχεια να εκκινήσουν και όλοι οι υπόλοιποι κόμβοι. Μόλις εκκινήσει ο ROS Master, τότε το πρόγραμμα εκκινεί τον ROS κόμβο της κάμερας με όνομα "***/camera***", τον rosserial ROS κόμβο με όνομα "***/mcu***" ο οποίος είναι υπεύθυνος για την επικοινωνία του ROS με τον μικροελεγκτή μέσω της σειριακής θύρας USB και, τέλος, εκκινεί τα δύο υπολειπόμενα υποπρογράμματα, τα οποία αναλύονται στη συνέχεια. Αφού γίνουν οι παραπάνω εκκινήσεις το υποπρόγραμμα δημιουργεί τον δικό του ROS κόμβο με όνομα "***/socketioclient***" ο οποίος διαθέτει πέντε Subscribers και έναν Publisher. Τέλος, είναι υπεύθυνο για τη σύνδεση του οχήματος με το δίκτυο και την ανταλλαγή πληροφοριών με τον διαχειριστή και τον εξυπηρετητή. Ο τρόπος σύνδεσης του οχήματος στο δίκτυο επιτυγχάνεται με τη χρήση ενός αμφίδρομου και ασύγχρονου πρωτοκόλλου, του WebSocket. Κατά την πρώτη σύνδεση στον εξυπηρετητή και για λόγους ασφαλείας το υποπρόγραμμα στέλνει μαζί με το αίτημα σύνδεσης την ταυτότητα του οχήματος, δηλαδή ένα token το οποίο έχει παραχθεί και υπογραφεί από τον εξυπηρετητή και έχει ενσωματωθεί στο όχημα ώστε να το ταυτοποιεί. Μόλις ταυτοποιηθεί το όχημα δημιουργούνται ασύγχρονες υποδοχές μηνυμάτων και στις δύο πλευρές ώστε ανά πάσα στιγμή η κάθε πλευρά να γνωρίζει την κατάσταση της άλλης. Ο Publisher του ROS κόμβου του υποπρογράμματος είναι υπεύθυνος να προωθεί στους ROS κόμβους των άλλων δύο υποπρογραμμάτων τις προς εκτέλεση εντολές οι οποίες λαμβάνονται από τον διαχειριστή. Το topic στο οποίο στέλνει ο Publisher αυτές τις εντολές είναι το "*/socketioclient/command*" και τα μηνύματα είναι τύπου *std_msgs/String*. Ο συγκεκριμένος ROS κόμβος διαθέτει δύο Subscribers οι οποίοι λαμβάνουν μετρήσεις από τον ROS κόμβο του μικροελεγκτή στα topics "*/imu*" και "*/battery_state*" με τύπο μηνυμάτων αντίστοιχα *sensor_msgs/Imu* και *sensor_msgs/BatteryState*, τα οποία αποστέλλονται περιοδικά στην εφαρμογή του

διαχειριστή όταν αυτός είναι διαθέσιμος. Ο ROS κόμβος διαθέτει επίσης και άλλους δύο Subscribers οι οποίοι ακούν στα topics "*/analyzer/rgbImage*" και "*/controller/depthImage*" με τύπο μηνυμάτων *std_msgs/String* τα οποία λαμβάνουν εικόνες RGB ή εικόνες βάθους, επεξεργασμένες ή μη, από τα άλλα δύο υποπρογράμματα, που θα αναλυθούν ακολούθως, και τις στέλνουν στον διαχειριστή. Ο τελευταίος Subscriber του ROS κόμβου ακούει στο topic "*/response*" για μηνύματα τύπου *std_msgs/String* στο οποίο στέλνουν οι άλλοι δύο ROS κόμβοι τις απαντήσεις τους στις εντολές που τους έχουν φτάσει από το topic "*/socketioclient/command*". Να σημειωθεί πως τα μηνύματα των εντολών και των απαντήσεων αλλά και γενικά όλα τα μηνύματα τύπου *std_msgs/String* είναι ειδικά διαμορφωμένα μηνύματα JSON μορφής τα οποία έχουν αποστολέα και παραλήπτη πέρα από το περιεχόμενο τους και είναι αυτού του τύπου για να είναι πιο εύκολα προσπελάσιμα.

Το δεύτερο υποπρόγραμμα αποτελεί την όραση του αυτοκινούμενου οχήματος και το κέντρο λήψης αποφάσεων κατά την αυτόνομη οδήγηση. Είναι γραμμένο στη γλώσσα προγραμματισμού Python και είναι αρκετά εκτενές και πολύπλοκο. Κατά την εκκίνησή του δημιουργεί τον ROS κόμβο του με όνομα "***/analyzer***" και δημιουργεί τρεις Publishers και δύο Subscribers. Οι δύο από τους Publishers είναι γνωστοί καθώς είναι τα topics "*/response*" και "*/analyzer/rgbImage*" τα οποία αναφέρθηκαν ως Subscribers στον ROS κόμβο "***/socketioclient***". Ο τρίτος Publisher στέλνει στο topic "*/analyzer/drivingDirection*" μηνύματα τύπου *geometry_msgs/Twist*, ο τρόπος παραγωγής του περιεχομένου των οποίων θα αναλυθεί στη συνέχεια. Όσον αφορά στους Subscribers του ROS κόμβου, ο πρώτος είναι γνωστός καθώς είναι το topic "*/socketioclient/command*" το οποίο αναφέρθηκε ως Publisher στον ROS κόμβο "***/socketioclient***". Ο δεύτερος Subscriber ακούει στο topic "*/camera/rgb/image_raw/compressed*" του ROS κόμβου "***/camera***" στο οποίο λαμβάνονται μηνύματα τύπου *sensor_msgs/CompressedImage*, τα οποία είναι οι RGB εικόνες, οι οποίες υφίστανται επεξεργασία και αναλύονται στον παρόντα ROS κόμβο για την όραση του οχήματος. Ο αλγόριθμος επεξεργασίας και

ανάλυσης της εικόνας καθώς και των εντολών πλοήγησης είναι αρκετά σύνθετος και αναλύεται σε ξεχωριστή παράγραφο.

Το τρίτο υποπρόγραμμα είναι υπεύθυνο για τη μεταβίβαση στον μικροελεγκτή των εντολών οδήγησης, τις οποίες λαμβάνει από το δεύτερο υποπρόγραμμα. Είναι γραμμένο στη γλώσσα προγραμματισμού Python και είναι αρκετά μικρό και απλό, αλλά διαθέτει προοπτικές ανάπτυξης. Κατά την εκκίνησή του δημιουργεί τον ROS κόμβο του με όνομα "***/controller***" και δημιουργεί τρεις Subscribers και τρεις Publishers. Οι δύο από τους Publishers είναι γνωστοί καθώς είναι τα topics "*/response*" και "*/analyzer/depthImage*" τα οποία αναφέρθηκαν ως Subscribers στον ROS κόμβο "***/socketioclient***". Ο τρίτος Publisher προωθεί στον μικροελεγκτή μέσω του topic "*/cmd_vel*" μηνύματα τύπου *geometry_msgs/Twist* τα οποία λαμβάνει μέσω του πρώτου Subscriber του στο topic "*/analyzer/drivingDirection*". Ο δεύτερος Subscriber του είναι γνωστός καθώς είναι το topic "*/socketioclient/command*" το οποίο αναφέρθηκε ως Publisher στον ROS κόμβο "***/socketioclient***". Ο τρίτος Subscriber ακούει στο topic "*/camera/depth/image_raw*" του ROS κόμβου "***/camera***" στο οποίο λαμβάνονται μηνύματα τύπου *sensor_msgs/Image* τα οποία είναι οι εικόνες βάθους, οι οποίες στην παρούσα φάση της υλοποίησης δεν υφίστανται επεξεργασία ούτε αναλύονται στον παρόντα ROS κόμβο για την πλοήγηση του οχήματος, απλά αποστέλλονται στη διεπαφή του διαχειριστή για εμφάνιση. Στην ακόλουθη εικόνα παρουσιάζεται η δομή του ROS συστήματος και η διασύνδεση των επιμέρους ROS κόμβων του μέσω των αντιστοίχων topics. Να σημειωθεί πως τα εξερχόμενα βέλη παριστάνουν τους Publishers των ROS κόμβων ενώ τα εισερχόμενα παριστάνουν αντίστοιχα τους Subscribers των ROS κόμβων.



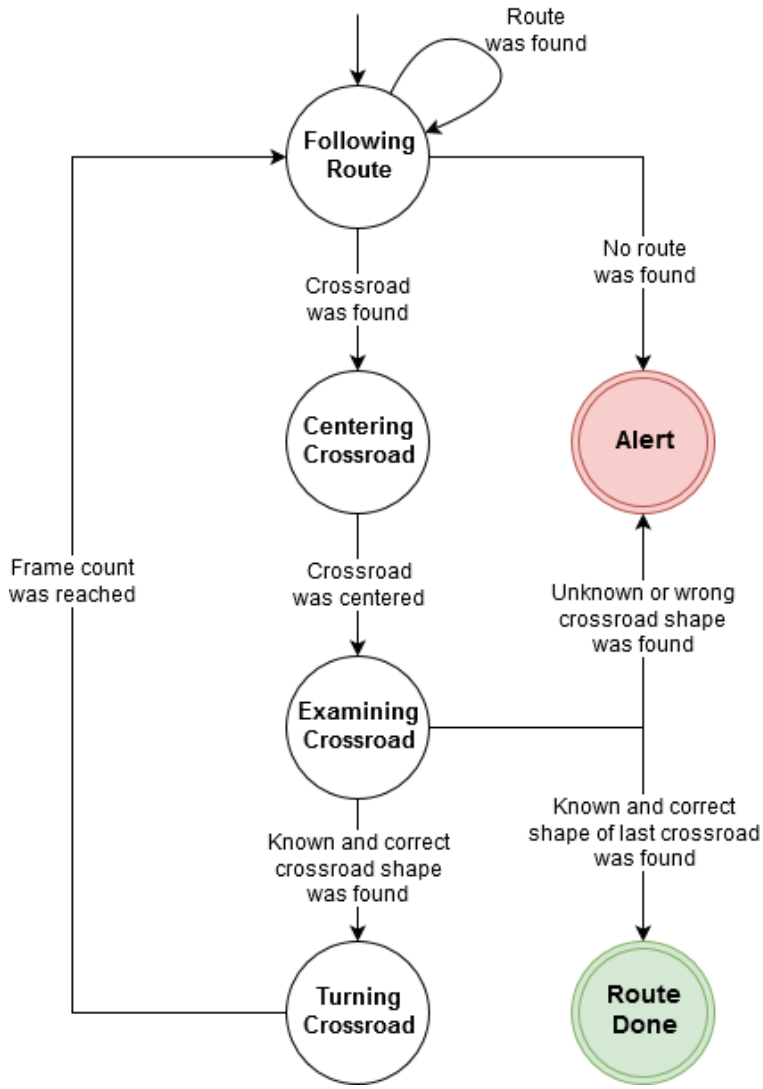
Εικόνα 4.10. *Διάγραμμα της δομής του ROS συστήματος του οχήματος*

4.4.4 Η μηχανή καταστάσεων του οχήματος

Στη συγκεκριμένη ενότητα αναλύεται ο αλγόριθμος ο οποίος εκτελείται στον ROS κόμβο "***/analyzer***" με τον οποίο το όχημα βλέπει και πλοηγείται. Όταν το όχημα κληθεί να πλοηγηθεί μόνο του, απαραίτητη προϋπόθεση είναι πως έχει λάβει το μονοπάτι της διαδρομής σαν ακολουθία αριθμών από τη διεπαφή του διαχειριστή και βρίσκεται σε δρόμο ανάμεσα στον κόμβο εκκίνησης και στον επόμενό του με κατεύθυνση προς τον δεύτερο. Ο αλγόριθμος πλοήγησης είναι βασισμένος σε μια μηχανή καταστάσεων η οποία διατηρείται εσωτερικά του υποπρογράμματος και αναλόγως με τις εικόνες που λαμβάνει ως είσοδο από την κάμερα κάνει και τις ανάλογες μεταβάσεις μεταξύ των καταστάσεων αυτών. Ακολούθως παρουσιάζονται με τη σειρά όλες οι πιθανές καταστάσεις στις οποίες μπορεί να βρεθεί το όχημα κατά την πλοήγησή του μαζί με το τι εισόδους χρειάζονται ώστε να μεταβαίνουν η κάθε μία στην επόμενη της. Οι αλγόριθμοι όρασης και πλοήγησης τους οποίους περιλαμβάνει η κάθε κατάσταση της μηχανής καταστάσεων παρουσιάζονται και αναλύονται στην ακόλουθη ενότητα.

Μόλις το όχημα λάβει την εντολή και το μονοπάτι για την αυτόνομη πλοήγηση εισέρχεται στην αρχική κατάσταση η οποία είναι η ακολούθηση του δρόμου (**Following Route**). Στη συγκεκριμένη κατάσταση το υποπρόγραμμα επεξεργάζεται και αναλύει κάθε εικόνα που λαμβάνει από την κάμερα και ελέγχει αν υπάρχει κάποιο σταυροδρόμι. Στην περίπτωση που δεν βρεθεί σταυροδρόμι, ελέγχει αν ακόμα βρίσκεται σε δρόμο ή όχι. Αν βρίσκεται σε δρόμο τότε παραμένει στην ίδια κατάσταση ακολουθώντας δρόμους, αλλιώς μεταβαίνει στην κατάσταση έκτακτης ανάγκης (**Alert**) στην οποία το όχημα ακυρώνει τη διαδρομή, ακινητοποιείται και ενημερώνει τον διαχειριστή. Στην περίπτωση που ανιχνευτεί σταυροδρόμι τότε το υποπρόγραμμα μεταβαίνει στην κατάσταση στην οποία το όχημα κινείται με κοφτές κινήσεις ώστε να ταυτίσει το κέντρο του σταυροδρομιού με το κέντρο της εικόνας (**Centering Crossroad**). Το συγκεκριμένο βήμα εξυπηρετεί στην αναγνώριση του σχήματος του σταυροδρομιού όπως παρουσιάζεται ακολούθως. Μόλις το όχημα κεντράρει το σταυροδρόμι στην εικόνα τότε μεταβαίνει στην κατάσταση εξέτασης του σταυροδρομιού (**Examining Crossroad**) για την εύρεση του σχήματός του. Στην περίπτωση που βρεθεί το σχήμα του σταυροδρομιού, εξετάζεται αν όντως έπρεπε να βρεθεί αυτού του τύπου σταυροδρόμι χρησιμοποιώντας έναν εσωτερικό χάρτη και, τέλος, επιλέγει με τη βοήθεια αυτού προς ποια κατεύθυνση πρέπει να πορευτεί, ώστε να μεταβεί στον επόμενο κόμβο του μονοπατιού. Ο εσωτερικός χάρτης πρόκειται για ένα JSON αρχείο το οποίο είναι απαραίτητο να το διαθέτουν όλα τα οχήματα και περιλαμβάνει όλους τους κόμβους του χάρτη του διαχειριστή, με πόσους κόμβους συνδέεται ο κάθε κόμβος και σε ποια κατεύθυνση πρέπει να στρίψει το όχημα εάν έχει βρεθεί στο σταυροδρόμι του κόμβου *Β* ερχόμενο από τον κόμβο *Α* και με προορισμό τον κόμβο *Γ*. Αν όλα πάνε καλά τότε το όχημα γνωρίζει προς τα πού πρέπει να κατευθυνθεί και μεταβαίνει για ορισμένο αριθμό καρέ, δηλαδή ορισμένο χρόνο, στην κατάσταση στροφής (**Turning Crossroad**), αλλιώς μεταβαίνει στην κατάσταση Alert. Μετά το πέρας του αριθμού των καρέ το όχημα μεταβαίνει στην αρχική κατάσταση Following Route αναζητώντας νέο σταυροδρόμι. Να σημειωθεί

πως αν το σχήμα του σταυροδρομιού είναι μια απλή γωνία τότε το υποπρόγραμμα δεν ασχολείται καθόλου με τον εσωτερικό χάρτη και μεταβαίνει απευθείας στην κατάσταση Turning Crossroad. Επιπροσθέτως, να σημειωθεί πως αν το σταυροδρόμι ανήκει στον τελευταίο κόμβο του μονοπατιού τότε το όχημα ακινητοποιείται, μεταβαίνει στην κατάσταση ολοκλήρωσης της διαδρομής (**Route Done**) και ενημερώνει τον διαχειριστή. Τέλος, σε κάθε αλλαγή κατάστασης το όχημα αποστέλλει στον διαχειριστή μηνύματα ενημέρωσης τα οποία προβάλλονται στο πλαίσιο της διαχειριστικής διεπαφής Vehicle's Log. Στην ακόλουθη εικόνα παρουσιάζεται το διάγραμμα ροής της μηχανής καταστάσεων του υποπρογράμματος με τις μεταβάσεις της.



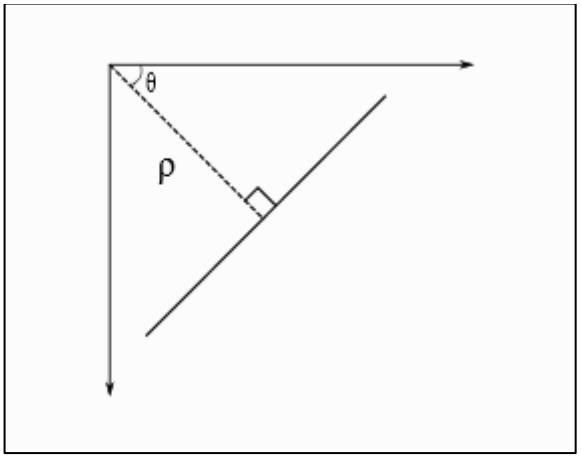
Εικόνα 4.11. *Διάγραμμα ροής της μηχανής καταστάσεων του υποπρογράμματος*

4.4.5 Οι αλγόριθμοι όρασης και πλοήγησης του οχήματος

Η λογική πίσω από την όραση του οχήματος βασίζεται κατά κύριο λόγο στη χρήση μεθόδων ανίχνευσης ακμών (Edge Detection) και ευθειών (Line Detection) σε εικόνα στην οποία η φωτεινότητα των δρόμων που ακολουθεί το όχημα έχει επαρκή αντίθεση σε σχέση με τη φωτεινότητα του δαπέδου, για παράδειγμα μαύροι δρόμοι σε λευκό δάπεδο. Σε αυτό το σημείο να σημειωθεί πως από πειραματικές δοκιμές προέκυψε πως την όραση του οχήματος βοηθούν και σταθεροποιούν οι σχετικά σταθερές και ομοιόμορφες συνθήκες φωτισμού. Για τον λόγο αυτόν κρίθηκε αναγκαία η τοποθέτηση των LEDs πάνω από την κάμερα με σκοπό να παρέχεται μια σταθερή συνιστώσα φωτισμού στην περιοχή ενδιαφέροντος, η οποία δεν θα δημιουργεί θαμβώσεις και θα είναι μεγαλύτερης έντασης από τον περιβάλλοντα φωτισμό, ώστε οι εναλλαγές του περιβάλλοντα φωτισμού να μην επηρεάζουν ιδιαίτερα τους αλγορίθμους όρασης. Στην παρούσα υλοποίηση οι δρόμοι έχουν δημιουργηθεί με χρήση μαύρης μονωτικής ταινίας και το δάπεδο είναι ξύλινο χρώματος σκούρου καφέ, παρόλα αυτά υπάρχει αντίθεση αλλά και αρκετός θόρυβος από τα μορφολογικά χαρακτηριστικά του δαπέδου.

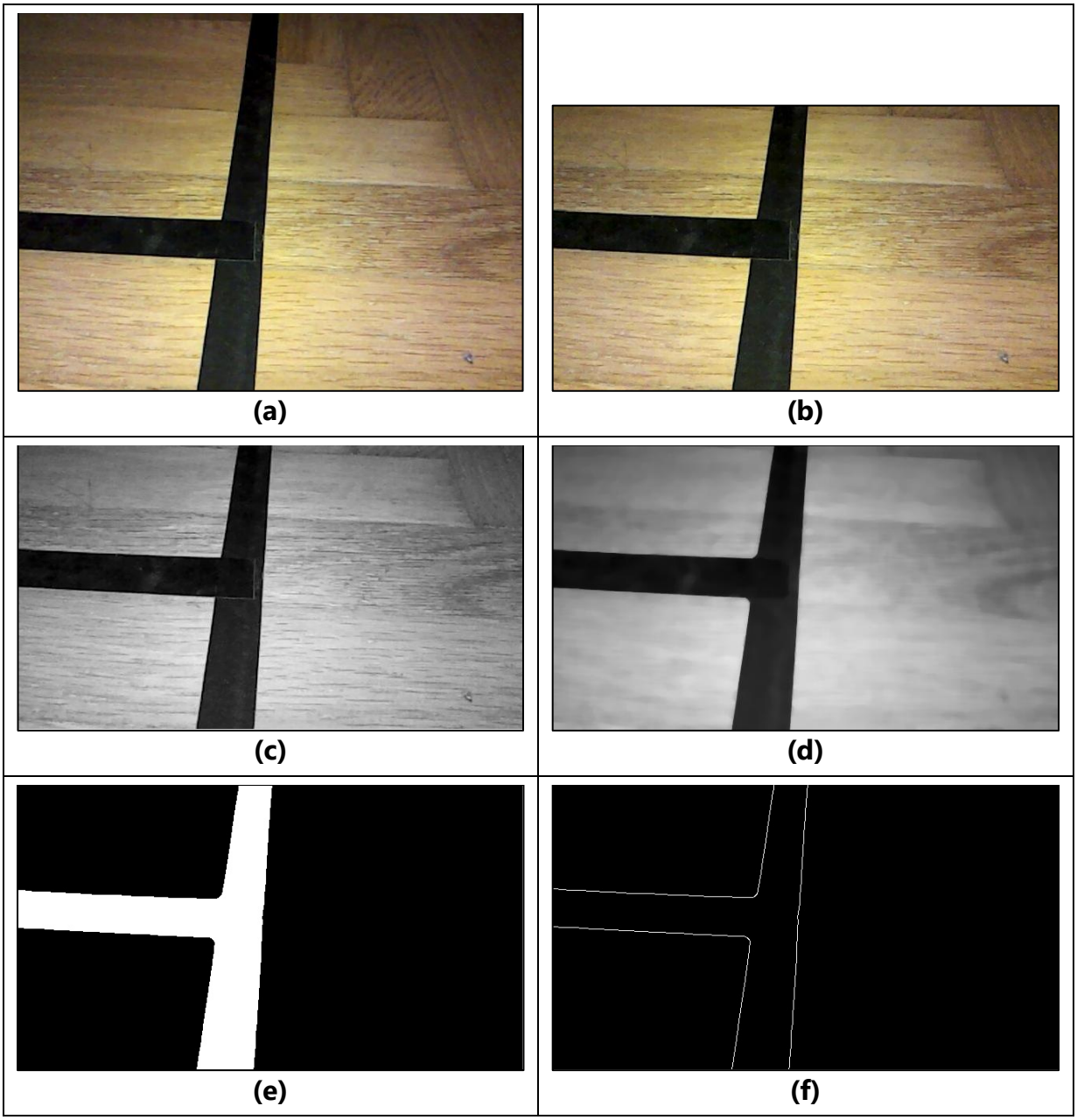
Οι αλγόριθμοι οι οποίοι εκτελούνται σε κάθε κατάσταση χρειάζονται ως είσοδο μια συγκεκριμένη μορφή εικόνας καθώς και ποιες ευθείες έχουν ανιχνευτεί σε αυτή. Αρχικά, η κάμερα του οχήματος λαμβάνει RGB εικόνες πλάτους 640 και ύψους 480 pixels. Η συγκεκριμένη εικόνα υπόκειται σε μια σειρά μεθόδων επεξεργασίας με τη βοήθεια της βιβλιοθήκης OpenCV με σκοπό να μειωθεί το μέγεθός της, να αφαιρεθούν οι άχρηστες και να εξαχθούν μόνο οι χρήσιμες πληροφορίες της. Αρχικά, από την εικόνα αφαιρείται το 1/4 από το άνω μέρος του ύψους της, δηλαδή πλέον γίνεται 640x360 pixels, ώστε να μην υπάρχουν περιττές πληροφορίες παρασκηνίου οι οποίες παράλληλα μπορούν να μπερδέψουν τους αλγόριθμους. Στη συνέχεια, η εικόνα μετατρέπεται από έγχρωμη (RGB) σε εικόνα κλίμακας του Γκρι (Gray scale) καθώς οι αλγόριθμοι βασίζονται κυρίως στην ανίχνευση ακμών και όχι στα χρώματα. Μέσω της συγκριμένης αλλαγής

εξοικονομείται επεξεργαστική ισχύς, δεδομένου ότι πλέον η εικόνα διαθέτει μόνο ένα κανάλι φωτεινότητας και όχι τρία. Έπειτα, στο ξύλινο δάπεδο υπάρχουν πολλές μικρές λεπτές σκούρες γραμμές και δεδομένου ότι οι αλγόριθμοι ανίχνευσης ακμών είναι επιρρεπείς σε θόρυβο, δηλαδή σε μεγάλες εναλλαγές των εντάσεων φωτεινότητας, κρίθηκε αναγκαία η εφαρμογή φίλτρου θολώματος (Median Blur) ώστε να εξομαλυνθούν οι περιοχές που εισάγουν θόρυβο. Στη συνέχεια στην εικόνα εφαρμόζεται η απλή μέθοδος κατωφλίωσης, ώστε αυτή να μετατραπεί σε δυαδική και να επιβιώσουν ως λευκά μόνο τα πολύ σκούρα μέρη της, δηλαδή η μαύρη ταινία. Ακολούθως, εφαρμόζεται η μέθοδος ανίχνευσης ακμών Canny η οποία με κατάλληλη παραμετροποίηση εξάγει μόνο το περίγραμμα της μαύρης ταινίας. Επιπρόσθετα, εφαρμόζεται η μέθοδος διαστολής (Dilation) καθώς τα περιγράμματα της μεθόδου Canny έχουν πλάτος ένα pixel και η επόμενη μέθοδος χρειάζεται πιο παχιές και συμπαγείς γραμμές. Τέλος, εφαρμόζεται η μέθοδος ανίχνευσης ευθειών Hough Line Transform η οποία εντοπίζει ευθείες γραμμές σε μια εικόνα και τις επιστρέφει εκφρασμένες σε πολικές συντεταγμένες, δηλαδή την κάθετη απόσταση $\boldsymbol{\rho}$ της ευθείας από την αρχή των αξόνων και τη γωνία $\boldsymbol{\theta}$ η οποία σχηματίζεται μεταξύ της κάθετης στην ευθεία και του οριζόντιου άξονα και μετράται αντίθετα προς τη φορά των δεικτών του ρολογιού.



Εικόνα 4.12. *Μετατροπή ευθείας σε πολικές συντεταγμένες [Πηγή: https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_houghlines/py_houghlines.html]*

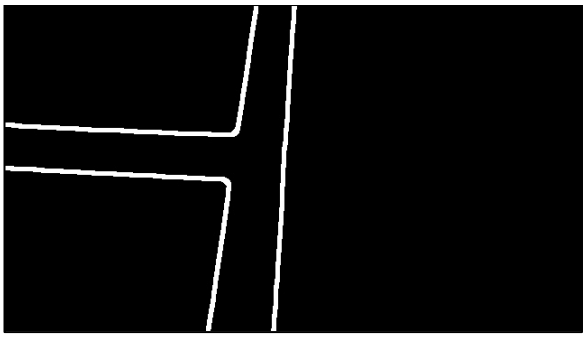
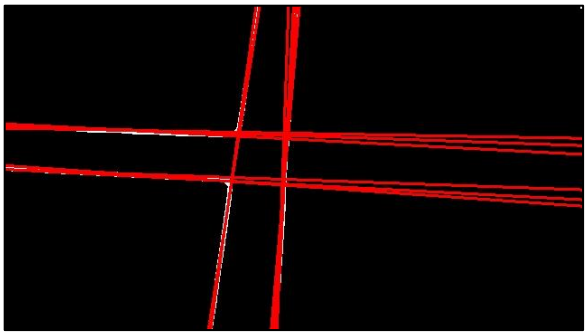
Τα δύο τελευταία βήματα παράγουν την εικόνα και το σύνολο των ευθειών που υπάρχουν σε αυτή αντίστοιχα, με βάση τα οποία λαμβάνουν αποφάσεις οι αλγόριθμοι των καταστάσεων. Στον ακόλουθο πίνακα παρουσιάζονται με τη σειρά τα στάδια επεξεργασίας από τα οποία διέρχεται η αρχική εικόνα.



(a) **(b)**

(c) **(d)**

(e) **(f)**

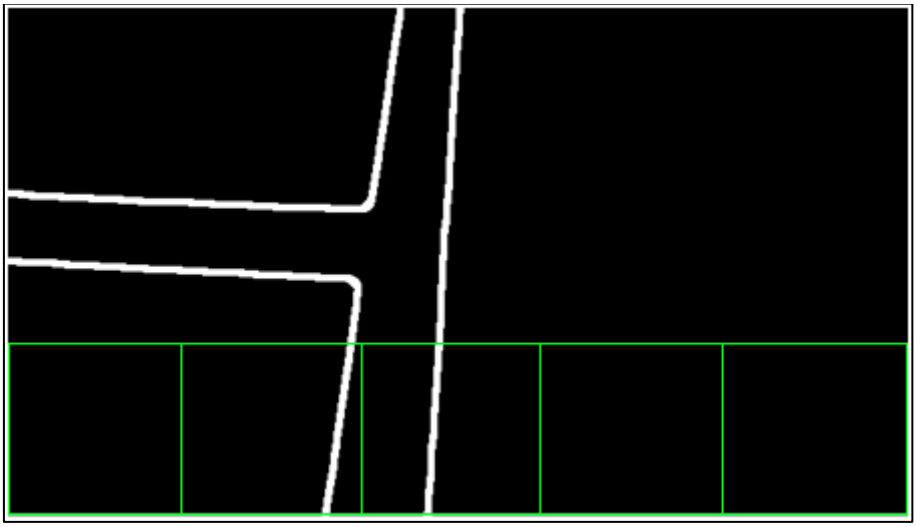
	
(g)	**(h)**

Πίνακας 4.1. *(a) Αρχική εικόνα, (b) μειωμένη σε ύψος εικόνα, (c) μετατροπή εικόνας σε gray scale, (d) εφαρμογή Median Blur, (e) κατωφλίωση εικόνας, (f) αποτέλεσμα μεθόδου Canny, (g) εφαρμογή διαστολής και (h) εφαρμογή μεθόδου ανίχνευσης ευθειών και απεικόνιση αυτών πάνω στην εικόνα (g)*

Να σημειωθεί πως στην Εικόνα (h) του πίνακα οι κόκκινες γραμμές αντιπροσωπεύουν τις ευθείες τις οποίες εντόπισε ο αλγόριθμος όρασης στο τελευταίο του βήμα και οι οποίες απεικονίζονται με κόκκινο χρώμα και εκτείνονται από τη μία άκρη της εικόνας στην άλλη.

Αρχικά, κατά την κατάσταση Following Route και ενώ το όχημα κινείται, ελέγχεται αν υπάρχει κάθετη ευθεία στον τρέχοντα προσανατολισμό (heading) του οχήματος. Το heading ανανεώνεται σε κάθε καρέ και είναι εκείνη η γωνία, από τις γωνίες των ευθειών που προκύπτουν από την προαναφερθείσα διαδικασία, η οποία έχει τη μικρότερη απόλυτη διαφορά από το heading του προηγούμενου καρέ. Στην περίπτωση που εντοπιστεί γωνία με διαφορά από το heading μεταξύ 60 και 120 μοιρών τότε το όχημα ακινητοποιείται και μεταβαίνει στην κατάσταση Centering Crossroad διότι υπάρχει υποψία για ύπαρξη σταυροδρομιού. Αν όμως δεν εντοπιστεί κάποια κάθετη στο heading γωνία τότε ανανεώνεται το heading λαμβάνοντας την τιμή της γωνίας που είναι πιο κοντά στην τρέχουσα τιμή του και το όχημα συνεχίζει να ακολουθεί τον δρόμο. Ο τρόπος με τον οποίο τον ακολουθεί υλοποιείται με τη συνάρτηση **followRoute**, η οποία στηρίζεται στην ιδέα ότι έχει γίνει καλό φιλτράρισμα στην αρχική εικόνα και λαμβάνεται μια εικόνα ως είσοδος όπως η Εικόνα (g) από τον προηγούμενο πίνακα. Εν συνεχεία, από την εικόνα απομονώνεται το κάτω 1/3 του ύψους της διότι αποτελεί την περιοχή

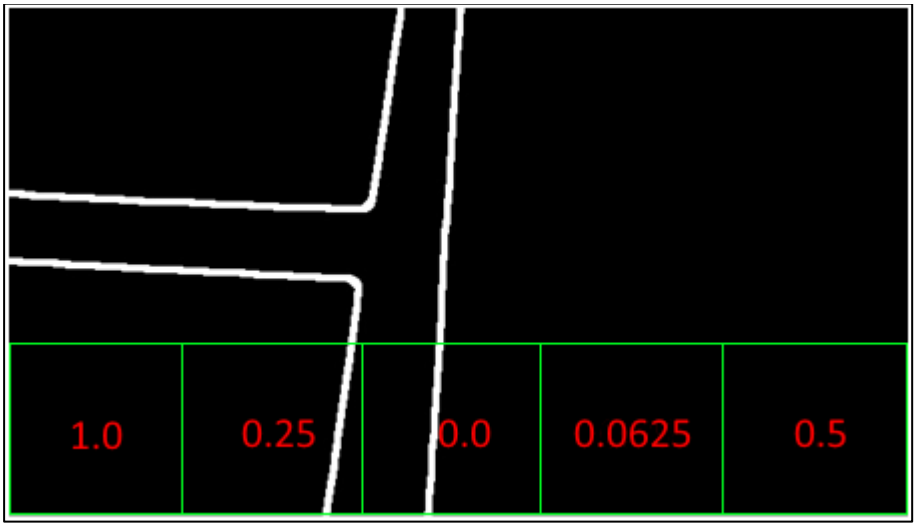
ενδιαφέροντος και χωρίζεται κατά μήκος σε ένα πλέγμα πέντε περιοχών ίσου μήκους όπως φαίνεται ακολούθως.



Εικόνα 4.13. *Χωρισμός περιοχής ενδιαφέροντος σε πλέγμα περιοχών ενδιαφέροντος*

Έπειτα, η συνάρτηση υπολογίζει το σταθμισμένο άθροισμα (Weighted Sum) των λευκών pixels μέσα σε κάθε μια από τις περιοχές και επιλέγει εκείνη με το μέγιστο, διότι, ιδανικά μιλώντας, εκείνη περιέχει τον δρόμο. Τα βάρη των περιοχών από προεπιλογή είναι σταθερά σε όλες τις περιοχές ίσα με 1.0 και εισάγονται στη συνάρτηση **followRoute** ως 1x3 πίνακας (pattern), στην προκειμένη περίπτωση [1.0, 1.0, 1.0]. Δεδομένου ότι υπάρχει περιττός αριθμός περιοχών μεγαλύτερος ή ίσος του 3 όπως π.χ. 5 στην παρούσα υλοποίηση, από τα στοιχεία του πίνακα το πρώτο αντιστοιχίζεται στην πρώτη, το δεύτερο στη μεσαία και το τρίτο στην τελευταία περιοχή. Κάθε άλλη ενδιάμεση αυτών περιοχή π.χ. η δεύτερη και η τέταρτη λαμβάνουν βάρη που προκύπτουν από την πολυωνυμική παρεμβολή (Polynomial Interpolation) δευτέρου βαθμού των άκρων τους με το αντίστοιχο βήμα. Για παράδειγμα, αν χρησιμοποιηθεί το pattern [1.0, 0.0, 0.5] σε 5 περιοχές τότε στη πρώτη αντιστοιχίζεται το βάρος 1.0, στη τρίτη το βάρος 0.0 και στην τελευταία το βάρος 0.5. Στη δεύτερη αντιστοιχίζεται το βάρος 0.25 και όχι το 0.5 που θα περίμενε κανείς καθώς υπάρχει ύψωση του βάρους στο τετράγωνο για πιο

απότομη πτώση, η οποία αξιοποιείται κατά τη στροφή του οχήματος. Στην τέταρτη περιοχή αντιστοιχίζεται με την ίδια λογική το βάρος 0.0625.



Εικόνα 4.14. *Απεικόνιση βαρών με χρήση του pattern [1.0, 0.0, 0.5]*

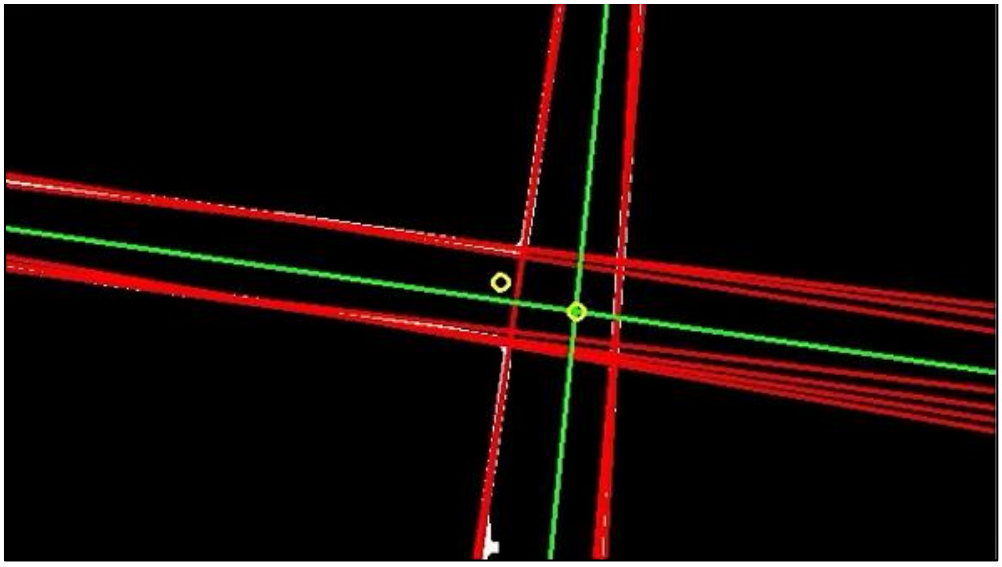
Σε αυτό το σημείο, εφόσον έχει βρεθεί σε ποια περιοχή ανήκει το μέγιστο σταθμισμένο άθροισμα, πρέπει να δοθεί μια εντολή οδήγησης, τέτοια ώστε το όχημα να ακολουθήσει τη συγκεκριμένη περιοχή. Αυτό πρακτικά σημαίνει πως το όχημα θα κινηθεί με σκοπό να φέρει στο κέντρο τη συγκεκριμένη περιοχή δηλαδή για παράδειγμα από πρώτη να γίνει τρίτη. Για να συμβεί αυτό, η συγκεκριμένη συνάρτηση υπολογίζει και στέλνει στον μικροελεγκτή μέσω του ROS κόμβου "/controller" ένα μήνυμα τύπου *geometry_msgs/Twist*. Ο συγκεκριμένος τύπος μηνύματος περιλαμβάνει τόσο τη γραμμική (linear) όσο και τη γωνιακή (angular) ταχύτητα του οχήματος στους τρεις άξονες, καθώς το όχημα εκτελεί σύνθετη κίνηση. Στην παρούσα υλοποίηση λόγο αναφοράς και χρήσης έχουν μόνο οι συνιστώσες ***x*** για τη γραμμική και ***z*** για τη γωνιακή ταχύτητα, ενώ οι υπόλοιπες λαμβάνουν μηδενική τιμή καθώς δεν συμβάλουν στην κίνηση του οχήματος εφόσον το δάπεδο είναι επίπεδο. Η ποσοστιαία τιμή των δύο αυτών συνιστωσών προκύπτει από τη θέση της περιοχής με το μέγιστο σταθμισμένο άθροισμα. Πιο συγκεκριμένα, στις ακραίες περιοχές η τιμή της γραμμικής ταχύτητας πρέπει να είναι μηδενική και η γωνιακή να λαμβάνει μέγιστη τιμή ώστε το όχημα να εκτελεί

μόνο περιστροφική κίνηση, προκειμένου να μην χάσει οπτική επαφή με τον δρόμο. Αντιθέτως, στην κεντρική περιοχή πρέπει να είναι μέγιστη η γραμμική ταχύτητα, ενώ η αντίστοιχη γωνιακή να είναι μηδενική, ώστε το όχημα να εκτελεί μόνο μεταφορική κίνηση διατηρώντας την πορεία του σταθερή. Οι ενδιάμεσες περιοχές λαμβάνουν ενδιάμεσες ποσοστιαίες τιμές και των δύο ειδών κίνησης μέσω της χρήσης γραμμικής παρεμβολής (Linear Interpolation). Οι τύποι οι οποίοι υπολογίζουν τις ποσοστιαίες ταχύτητες με βάση τη θέση της περιοχής με το μέγιστο σταθμισμένο άθροισμα είναι οι ακόλουθοι:

```
linearVelX = routeRoiHGrids*(centerGridIndex/10.0-0.1*abs(centerGridIndex-maxGridIndex))
angularVelZ = (1.0/centerGridIndex)*(centerGridIndex-maxGridIndex)
driveVehicle(linearVelX*0.1, angularVelZ*1)
```

Να σημειωθεί πως η γωνιακή ταχύτητα υπολογίζεται με πρόσημο ώστε το όχημα να στρίβει προς την κατάλληλη πλευρά (δεξιόστροφα για αρνητικές και αριστερόστροφα για θετικές τιμές). Επίσης, να σημειωθεί ότι η συνάρτηση **driveVehicle** αποστέλλει το ROS μήνυμα στον μικροελεγκτή και ότι, θέλοντας η οδήγηση να είναι μόνιμα επιτυχής, δηλαδή στο οπτικό πεδίου του οχήματος να παραμένει πάντα τμήμα του δρόμου, πειραματικές δοκιμές έδειξαν πως πρέπει η μέγιστη γραμμική ταχύτητα να μειωθεί στο 10% της αρχικής. Στο επόμενο υποκεφάλαιο αναλύεται ο τρόπος με τον οποίο ο μικροελεγκτής μεταφράζει τις συγκεκριμένες ποσοστιαίες τιμές σε περιστροφική κίνηση των τροχών του οχήματος.

Όταν ο αλγόριθμος όρασης εντοπίσει πιθανό σταυροδρόμι ακινητοποιεί το όχημα και μεταβαίνει στην κατάσταση Centering Crossroad. Στη συγκεκριμένη κατάσταση σε κάθε καρέ, αρχικά, ομαδοποιεί όλες τις ευθείες σαν σημεία (ρ, θ) σε δύο ομάδες (clusters) με τη χρήση της μεθόδου k-means και εξάγει τα κέντρα τους, δηλαδή δύο ευθείες οι οποίες ιδανικά είναι κάθετες μεταξύ τους. Στη συνέχεια, υπολογίζει τις συντεταγμένες του σημείου τομής των δύο ευθειών, το οποίο αντιπροσωπεύει το κεντρικό σημείο του σταυροδρομιού, και υπολογίζει την απόσταση του σημείου αυτού από το κεντρικό σημείο της εικόνας.

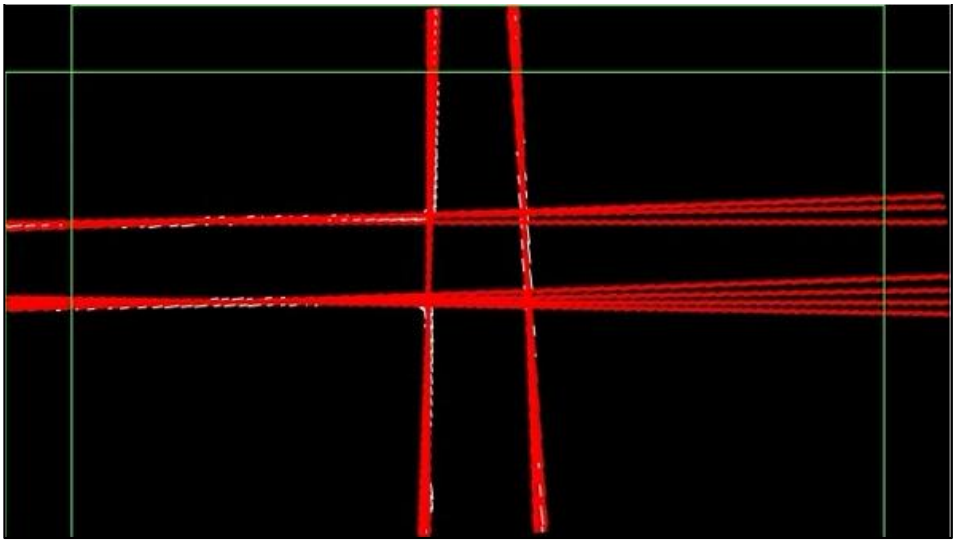


Εικόνα 4.15. *Χωρισμός κόκκινων ευθειών σε δύο ομάδες (πράσινες ευθείες), εύρεση και απεικόνιση σημείου τομής τους και απεικόνιση κέντρου εικόνας*

Αν η απόσταση είναι μικρότερη των 10 pixels τότε ο αλγόριθμος θέτει ως νέο heading τη γωνία της ευθείας η οποία είναι πιο κοντά στις μηδέν μοίρες και μεταβαίνει στην κατάσταση Examining Crossroad, αλλιώς κινεί κατάλληλα και με κοφτές κινήσεις το όχημα ώστε να μειώσει τη συγκεκριμένη απόσταση. Η κίνηση του οχήματος επιτυγχάνεται και πάλι μέσω της συνάρτησης driveVehicle η οποία εκτελεί κινήσεις μόνο για 100 milliseconds εναλλάσσοντας διαδοχικά γραμμική και περιστροφική κίνηση. Η ταχύτητα των κινήσεων εξαρτάται από τη προσημασμένη διαφορά των συντεταγμένων μεταξύ των δύο σημείων, όπου η διαφορά στον ευθύ άξονα ρυθμίζει τη γραμμική ταχύτητα και η διαφορά στον εγκάρσιο την περιστροφική. Να σημειωθεί πως από πειραματικές δοκιμές φάνηκε πως η λογική της εναλλαγής μεταξύ γραμμικής και περιστροφικής κίνησης, και όχι απευθείας σύνθετης, έχει ως αποτέλεσμα την πιο άμεση σύγκλιση της διαδικασίας.

Έχοντας πλέον κεντράρει το σταυροδρόμι, ο αλγόριθμος μεταβαίνει στην κατάσταση Examining Crossroad. Στη συγκεκριμένη κατάσταση καλείται να αναγνωρίσει αν επρόκειτο για σταυροδρόμι, ποιο το είδος του, αν έπρεπε να βρει αυτού του είδους σταυροδρόμι με βάση τον εσωτερικό του χάρτη και προς ποια κατεύθυνση πρέπει να πορευτεί ώστε να μεταβεί στον επόμενο κόμβο. Αρχικά, ο αλγόριθμος, όντας ακινητοποιημένο το όχημα από την προηγούμενη κατάσταση,

περιστρέφει την εικόνα κατά τις μοίρες του τελευταίου heading ώστε να ευθυγραμμίσει το σταυροδρόμι με τους άξονες. Στη συνέχεια, αθροίζει ξεχωριστά στην αριστερή, στην άνω και στη δεξιά πλευρά της εικόνας τα λευκά pixels ώστε να εντοπίσει σε ποιες περιοχές υπάρχει δρόμος με σκοπό να αναγνωρίσει τον τύπο του σταυροδρομιού. Σε αυτό το σημείο διαφαίνεται η χρησιμότητα της προηγούμενης κατάστασης, καθώς, αν δεν ήταν κεντραρισμένη η εικόνα, η περιστροφή της θα οδηγούσε σε λάθος συμπεράσματα για τον τύπο του σταυροδρομιού.



Εικόνα 4.16. *Εύρεση δρόμου στις τρεις πλευρές της εικόνας*

Για να θεωρηθεί πως εμπεριέχεται δρόμος σε μια περιοχή πρέπει το άθροισμα των περιεχόμενων εντός της λευκών pixels να ξεπερνά ένα συγκεκριμένο κατώφλι. Από τη διαδικασία αυτή εξάγεται ένας πίνακας ο οποίος περιέχει τιμές 0 και 1 για το αν η κάθε περιοχή περιέχει ή όχι δρόμο. Για την προηγούμενη εικόνα ο πίνακας που προκύπτει είναι ο [1, 1, 0] καθώς μόνο η δεξιά πλευρά δεν περιέχει δρόμο. Ο συγκεκριμένος πίνακας φανερώνει τον τύπο του σταυροδρομιού και ακολούθως παρουσιάζονται όλοι οι δυνατοί συνδυασμοί ή αλλιώς patterns και σε ποιο τύπο σταυροδρομιού αντιστοιχούν.

Pattern	Τύπος σταυροδρομιού
[0, 0, 0]	Άγνωστος (Αδιέξοδο)
[0, 0, 1]	Δεξιά γωνία
[0, 1, 0]	Άγνωστος (Ευθεία)
[0, 1, 1]	Ευθεία και δεξιά στροφή
[1, 0, 0]	Αριστερή γωνία
[1, 0, 1]	Αριστερή και δεξιά στροφή
[1, 1, 0]	Ευθεία και αριστερή στροφή
[1, 1, 1]	Ευθεία, αριστερή και δεξιά στροφή

Πίνακας 4.2. *Αντιστοίχηση patterns με τύπους σταυροδρομιών*

Αν δεν βρεθεί κάποιος γνωστός τύπος τότε ο αλγόριθμος μεταβαίνει στην κατάσταση Alert, αλλιώς ερευνά αν αυτός που εντόπισε είναι αυτός που έπρεπε να συναντήσει κατά την πλοήγησή του. Χρησιμοποιώντας τον εσωτερικό του χάρτη και με την γνώση για το ποιος ήταν ο προηγούμενος κόμβος και ποιος ο επόμενος, ελέγχει αν στον τρέχοντα κόμβο έπρεπε να εντοπίσει το συγκεκριμένο σχήμα σταυροδρομιού. Αν όχι τότε μεταβαίνει στην κατάσταση Alert, αλλιώς με τα ίδια στοιχεία ψάχνει προς ποια κατεύθυνση πρέπει να πλοηγηθεί ώστε να βρει τον επόμενο κόμβο. Αν είναι ο τελευταίος κόμβος της διαδρομής τότε μεταβαίνει στην κατάσταση Route Done, αλλιώς αντικαθιστά τον τρέχοντα κόμβο με τον επόμενο, καθορίζει το pattern της κατεύθυνσης που θα ακολουθήσει και μεταβαίνει στην κατάσταση Turning Crossroad. Να σημειωθεί πως αν εντοπιστεί αριστερή ή δεξιά γωνία ο αλγόριθμος μεταβαίνει απευθείας στην κατάσταση Turning Crossroad και δεν ασχολείται με τον εσωτερικό χάρτη.

Κατά την κατάσταση Turning Crossroad για να στρίψει ή να κινηθεί ευθεία το όχημα χρησιμοποιείται η συνάρτηση followRoute με είσοδο συγκεκριμένο πίνακα βαρών ανάλογα με την κατεύθυνση την οποία πρέπει να ακολουθήσει το όχημα. Για παράδειγμα, αν το όχημα πρέπει να στρίψει αριστερά εφαρμόζεται το pattern βαρών [1.0, 0.0, 0.0] ώστε να δοθεί μεγάλη βαρύτητα στις αριστερές περιοχές της εικόνας για συγκεκριμένο αριθμό καρέ και έπειτα μεταβαίνει στην κατάσταση Follow Route. Αντίστοιχα αν το όχημα πρέπει να στρίψει δεξιά εφαρμόζεται το

pattern [0.0, 0.0, 1.0] και για ευθεία πορεία το pattern [0.0, 1.0, 0.0]. Να σημειωθεί πως η εκτέλεση της συγκεκριμένης κατάστασης διαρκεί συγκεκριμένο χρόνο ώστε το όχημα να προλάβει να αφήσει πίσω του το σταυροδρόμι, ώστε όταν επανέλθει η κατάσταση Follow Route να υπάρχει μόνο ένας δρόμος στην εικόνα.

4.5 Το λογισμικό του μικροελεγκτή του οχήματος

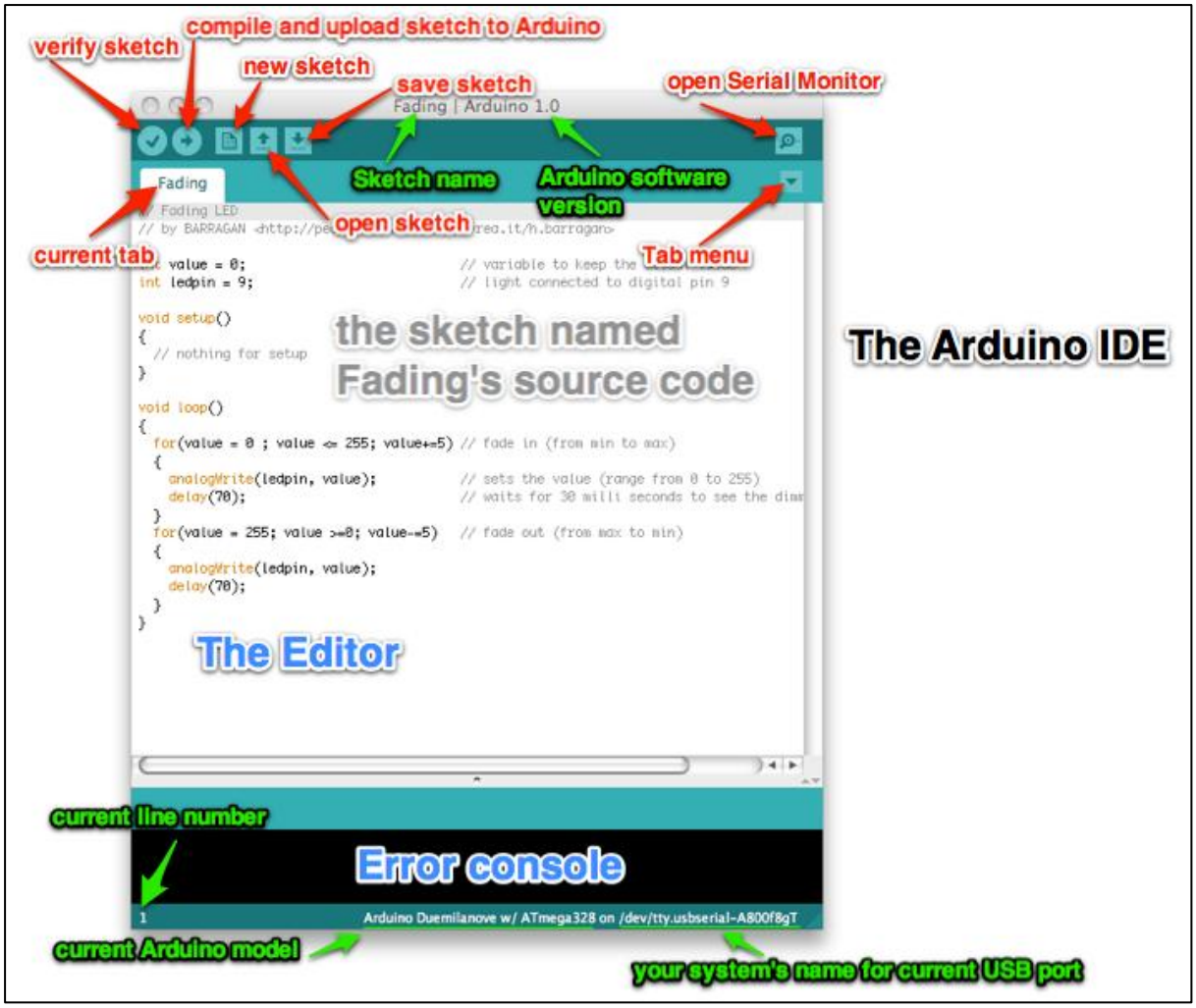
4.5.1 Το ολοκληρωμένο περιβάλλον ανάπτυξης Arduino IDE

Το ολοκληρωμένο περιβάλλον ανάπτυξης Arduino IDE είναι μία εφαρμογή γραμμένη σε Java η οποία λειτουργεί σε πολλές πλατφόρμες και προέρχεται από το IDE των γλωσσών προγραμματισμού Processing και Wiring [10]. Πρόκειται για ένα περιβάλλον εύχρηστο για όσους δεν είναι εξοικειωμένοι με τον προγραμματισμό και διανέμεται δωρεάν από την επίσημη ιστοσελίδα της πλατφόρμας Arduino. Αποτελεί ένα πρόγραμμα επεξεργασίας κώδικα γραμμένου σε C++ με χαρακτηριστικά αντίστοιχα με εκείνα που περιλαμβάνουν άλλα γνωστά IDE καθώς επίσης είναι σε θέση να μεταγλωττίζει και να φορτώνει προγράμματα στην πλακέτα με ένα μόνο κλικ. Δεν υπάρχει συνήθως καμία ανάγκη να υποστούν επεξεργασία τα αρχεία Makefiles ή να τρέξουν τα προγράμματα σε περιβάλλον γραμμής εντολών. Ένα πρόγραμμα ή κώδικας ο οποίος γράφεται για μια Arduino πλακέτα ονομάζεται σκίτσο (sketch).

Οι χρήστες πρέπει μόνο να ορίσουν δύο διαδικασίες για να κάνουν ένα πρόγραμμα κυκλικής εκτέλεσης:

- **setup()** - μία διαδικασία η οποία εκτελείται μία φορά στην αρχή του προγράμματος και αρχικοποιεί ρυθμίσεις και μεταβλητές
- **loop()** - μία διαδικασία η οποία καλείται επαναλαμβανόμενα μετά την setup() μέχρι να απενεργοποιηθεί η πλακέτα

Στην ακόλουθη εικόνα παρουσιάζεται λεπτομερώς και με επεξηγηματικές λεζάντες η μορφή της εφαρμογής Arduino IDE.



Εικόνα 4.17. *Η μορφή της εφαρμογής Arduino IDE. [Πηγή: http://www.hacdc.org/summer-school-2013/]*

Ένα τυπικό πρώτο πρόγραμμα για έναν μικροελεγκτή είναι να αναβοσβήνει απλά έναν ενδείκτη LED. Για το σκοπό αυτό, στο περιβάλλον Arduino ο χρήστης μπορεί να γράψει ένα πρόγραμμα όπως αυτό που φαίνεται στην ακόλουθη εικόνα:

Εικόνα 4.18. *Πρόγραμμα (αριστερά) στο περιβάλλον Arduino (δεξιά) για την αναλαμπή ενός LED [10].*

Το συγκεκριμένο παράδειγμα αναφέρεται σε ένα χαρακτηριστικό το οποίο διαθέτουν οι περισσότερες πλακέτες Arduino, ένα LED και μία αντίσταση, τα οποία συνδέονται μεταξύ του ακροδέκτη 13 και της γείωσης και αποτελούν ένα βολικό χαρακτηριστικό για πολλά απλά τεστ. Ο προηγούμενος κώδικας δεν αναγνωρίζεται απευθείας από έναν κανονικό μεταγλωττιστή C++ ως έγκυρο πρόγραμμα. Όταν ο χρήστης κάνει κλικ στο κουμπί "*Upload*" στο IDE, ένα αντίγραφο του κώδικα γράφεται σε ένα προσωρινό αρχείο με ένα παραπάνω include στην κορυφή του και μία πολύ απλή συνάρτηση main() στο τέλος για να φτιάξει ένα έγκυρο C++ πρόγραμμα.

Το Arduino IDE χρησιμοποιεί την GNU εργαλειοθήκη μαζί με το AVR Libc για να μεταγλωττίζει προγράμματα καθώς και το avrdude για να φορτώνει τα προγράμματα αυτά στην πλακέτα. Δεδομένου ότι η πλατφόρμα Arduino χρησιμοποιεί Atmel μικροελεγκτές, τα περιβάλλοντα ανάπτυξης AVR Studio της Atmel ή η νεότερη έκδοση του Atmel Studio μπορούν επίσης να χρησιμοποιηθούν για την ανάπτυξη λογισμικού για το Arduino.

4.5.2 Το πρόγραμμα του μικροελεγκτή

Όπως αναφέρθηκε προηγουμένως η NodeMCU πλακέτα αποτελεί έναν ROS κόμβο για το λογισμικό της επεξεργαστικής μονάδας. Ο ρόλος του συγκεκριμένου ROS κόμβου είναι η συλλογή μετρήσεων από τους περιφερειακούς αισθητήρες οι οποίοι είναι συνδεδεμένοι σε αυτόν, η αποστολή των μετρήσεων στην επεξεργαστική μονάδα και ο έλεγχος των τροχών του οχήματος.

Στην πλευρά της επεξεργαστικής μονάδας εκτελείται ήδη το πακέτο rosserial_python καθώς είναι απαραίτητο για τη γεφύρωση του μικροελεγκτή ως ROS κόμβου με την επεξεργαστική μονάδα μέσω του σειριακού πρωτοκόλλου UART. Παρόλα αυτά για να επικοινωνήσουν στην ίδια γλώσσα τα δύο συστήματα οφείλει και ο μικροελεγκτής από την πλευρά του να χρησιμοποιήσει μια βιβλιοθήκη η οποία εφαρμόζει το rosserial πρωτόκολλο. Αυτή είναι η **ros_lib** η οποία είναι κατάλληλα υλοποιημένη ώστε να ενσωματώνεται σε Arduino κώδικα και η οποία υποστηρίζει και τον μικροελεγκτή ESP8266. Η συγκεκριμένη βιβλιοθήκη υποστηρίζει λειτουργίες του ROS όπως topics, τύπους ROS μηνυμάτων για τα topics, Publishers και Subscribers, οι οποίες είναι χρήσιμες στην παρούσα υλοποίηση.

Το λογισμικό του μικροελεγκτή είναι απλό, σειριακό και ξεκινά εκτελώντας μια φορά στη διαδικασία setup τις ακόλουθες αρχικοποιήσεις. Αρχικά, αρχικοποιεί ορισμένες παραμέτρους του IMU αισθητήρα κάτι το οποίο δεν υπάρχει λόγος να αναλυθεί στο παρόν κείμενο, καθώς ο συγκεκριμένος αισθητήρας δεν αξιοποιείται στην παρούσα φάση της υλοποίησης. Στη συνέχεια, όντας ROS κόμβος ο μικροελεγκτής εκκινεί προγραμματιστικά τον κόμβο μέσω της σειριακής θύρας ώστε να ξεκινήσει η επικοινωνία με την επεξεργαστική μονάδα και δημιουργεί δύο Publishers και έναν Subscriber. Ο πρώτος Publisher δημιουργεί το topic με όνομα *"/battery_state"* εκπέμποντας σε αυτό ROS μηνύματα τύπου *sensor_msgs/BatteryState*, ενώ ο δεύτερος Publisher δημιουργεί το topic *"/imu"* εκπέμποντας σε αυτό μηνύματα τύπου *sensor_msgs/Imu*. Οι συγκεκριμένοι

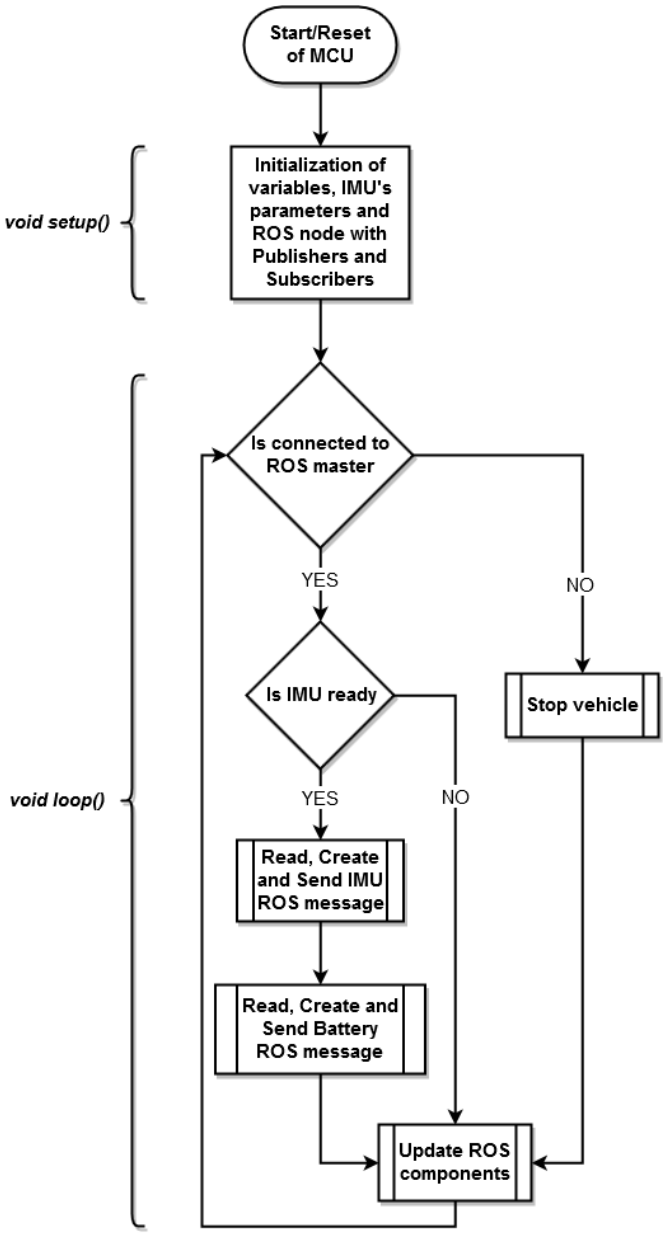
Publishers, όπως δείχνει και το όνομά τους, εκπέμπουν μηνύματα μετρήσεων που αφορούν στην κατάσταση της μπαταρίας του οχήματος και του αισθητήρα IMU. Τέλος, δημιουργεί έναν Subscriber ο οποίος ακούει σε ROS μηνύματα τύπου *geometry_msgs/Twist* τα οποία εκπέμπονται από την επεξεργαστική μονάδα στο topic *"/cmd_vel"*. Το συγκεκριμένο μήνυμα περιλαμβάνει τόσο τη γραμμική ταχύτητα του οχήματος όσο και τη γωνιακή του στους τρεις άξονες.

Μόλις ολοκληρωθεί η εκτέλεση της διαδικασίας setup σειρά έχει η σειριακά επαναλαμβανομένη εκτέλεση της διαδικασίας loop. Στη συγκεκριμένη διαδικασία, αρχικά, ελέγχεται αν υπάρχει επικοινωνία με τον ROS Master της επεξεργαστικής μονάδας. Αν δεν υπάρχει τότε το όχημα ακινητοποιείται. Αν όμως υπάρχει τότε ελέγχεται αν υπάρχουν έτοιμες μετρήσεις από τον IMU αισθητήρα. Αν υπάρχουν τότε δημιουργείται ένα ROS μήνυμα τύπου *sensor_msgs/Imu* και διαβάζοντας στη συνέχεια την τάση της μπαταρίας δημιουργείται και ένα μήνυμα τύπου *sensor_msgs/BatteryState*. Στο συγκεκριμένο μήνυμα ανανεώνεται μόνο η μέτρηση της τάσης της μπαταρίας, διότι αυτή προβάλλεται στη γραφική διεπαφή του διαχειριστή. Η τιμή της τάσης της μπαταρίας διαβάζεται μέσω του διαιρέτη τάσης μέσω μιας 10-bit αναλογικής εισόδου του μικροελεγκτή. Για να μετασχηματιστεί η μέτρηση στην πραγματική τάση της μπαταρίας χρησιμοποιείται ο τύπος:

```
battery_msg.voltage = ((analogValue/ADC_VALUES)*MAX_MEASURED_VOLTAGE)*5.4844
```

Το μέγεθος ***ADC_VALUES*** δείχνει σε πόσα κβάντα χωρίζεται το εύρος τιμών της αναλογικής εισόδου, όπου για 10-bit είναι 1024 τιμές. Το μέγεθος ***MAX_MEASURED_VOLTAGE*** είναι η πραγματικά μετρούμενη τάση στην τιμή 1024 της αναλογικής εισόδου η οποία είναι 3.15V. Τέλος, η τιμή 5.4844 είναι ο διορθωμένος 5.1667 λόγος διαίρεσης ο οποίος προέκυψε από τις επιλογές των αντιστατών που τον συνιστούν. Τα συγκεκριμένα μηνύματα στη συνέχεια εκπέμπονται στα αντίστοιχά τους topics μέσω των αντίστοιχων Publishers που αναφέρθηκαν στην προηγούμενη παράγραφο. Να σημειωθεί πως η αποστολή πακέτων συγχρονίζεται εσκεμμένα με το πότε ο IMU αισθητήρας έχει έτοιμη

μέτρηση, ώστε να μην σπαταλούνται άσκοπα πόροι στέλνοντας σε κάθε κύκλο της loop την κατάσταση της μπαταρίας η οποία αλλάζει αργά. Μόλις αποσταλούν τα μηνύματα ή αν δεν υπάρχει έτοιμη μέτρηση από τον IMU αισθητήρα τότε το λογισμικό σαν τελευταίο βήμα ελέγχει αν ο Subscriber έχει κάποιο εισερχόμενο μήνυμα από τον ROS Master στο αντίστοιχό του topic. Η σειρά κλήσης των συγκεκριμένων διαδικασιών στον μικροελεγκτή φαίνεται στο διάγραμμα ροής που παρουσιάζεται στην ακόλουθη εικόνα.



Εικόνα 4.19. *Διάγραμμα ροής του προγράμματος του μικροελεγκτή*

Το μήνυμα τύπου *geometry_msgs/Twist* αποτελεί το σημαντικότερο κομμάτι του λογισμικού του μικροελεγκτή και περιλαμβάνει, όπως παρουσιάστηκε σε προηγούμενη παράγραφο, τις ποσοστιαίες τιμές των τριών αξόνων της γραμμικής και της περιστροφικής ταχύτητας του οχήματος. Ο τρόπος με τον οποίο προκύπτουν οι συγκεκριμένες τιμές αναλύθηκε στη σχετική παράγραφο, σε αυτήν την παράγραφο αναλύεται ο τρόπος με τον οποίο μεταφράζονται οι τιμές αυτές σε περιστροφική κίνηση των τροχών του οχήματος. Να σημειωθεί, όπως αναφέρθηκε και στην αντίστοιχη παράγραφο, πως για λόγους απλοποίησης της υλοποίησης θεωρείται πως το όχημα κινείται σε επίπεδο δάπεδο και για αυτόν τον λόγο αξιοποιούνται μόνο οι συνιστώσες ***x*** της γραμμικής και ***z*** της περιστροφικής ταχύτητας. Δεδομένου ότι οι τιμές των ταχυτήτων αυτών είναι ποσοστιαίες και οι μονάδες μέτρησης τους διαφορετικές, είναι απαραίτητη η μετατροπή τους σε μια κοινή μονάδα μέτρησης ώστε να μπορούν να είναι συγκρίσιμα μεγέθη και να γίνονται πράξεις μεταξύ τους. Για τον λόγο αυτόν μετατρέπουμε τις τιμές στη μονάδα ταχύτητας ***m/s*** με τους ακόλουθους μετασχηματισμούς:

```
x = x * MAX_LINEAR_VEL όπου x ≡ velocity_msg.linear.x
z = z * MAX_ROT_VEL * VEHICLE_RADIUS όπου z ≡ velocity_msg.angular.z
```

Το μέγεθος ***MAX_LINEAR_VEL*** δηλώνει τη μέγιστη γραμμική ταχύτητα του οχήματος η οποία είναι 0.1912m/s. Το μέγεθος ***MAX_ROT_VEL*** δηλώνει τη μέγιστη επιτόπια περιστροφική ταχύτητα του οχήματος η οποία είναι 0.5487 rad/s. Να σημειωθεί πως οι μετρήσεις των παραπάνω μεγεθών είναι πειραματικές και προέκυψαν ενώ το όχημα είχε όλο του το βάρος και γεμάτη τη μπαταρία του. Τέλος, το μέγεθος **VEHICLE_RADIUS** δηλώνει την ακτίνα του οχήματος αν θεωρηθεί πως το όχημα όταν περιστρέφεται επιτόπια σχηματίζει έναν κύκλο με κέντρο το κέντρο του και ακτίνα την απόσταση από το κέντρο του μέχρι το εμπρόσθιο μέρος του, δηλαδή το μισό του μήκος, όπου είναι τοποθετημένη η κάμερα και είναι 0.16 m. Εφόσον οι δύο τιμές έχουν μετατραπεί σε m/s απεικονίζονται στο επίπεδο ***(x, z)*** ως σημεία και άρα και ως διανύσματα από την αρχή των αξόνων. Η τιμή της ώθησης του κάθε τροχού του οχήματος προκύπτει

από τη γωνία που σχηματίζει το διάνυσμα των ταχυτήτων με τον άξονα x. Για να υπολογιστεί η εν λόγω γωνία χρησιμοποιείται η συνάρτηση ***rads=atan2(z, x)***, όπου η γωνία υπολογίζεται σε ακτίνια (radians) λαμβάνοντας τιμές στο εύρος (-π, π]. Ακολούθως, παρουσιάζονται οι τύποι ανά περίπτωση από τους οποίους προκύπτουν οι τιμές Duty Cycle για τα PWM σήματα στις εισόδους του οδηγού L298N. Κατά τις μετατροπές χρησιμοποιείται η συνάρτηση ***map()*** η οποία αντιστοιχίζει μια τιμή από ένα εύρος τιμών σε ένα άλλο. Πιο συγκεκριμένα, αντιστοιχίζονται τα ποσοστά των ταχυτήτων στο εύρος τιμών που μπορεί να λάβει το Duty Cycle των PWM ακροδεκτών για να κινηθούν οι τροχοί, δηλαδή πειραματικά το εύρος πρόεκυψε πως είναι από 500 (49%) έως 1023 (100%).

- ***rads = 0 και x ≠ 0***

  in1_PWM = map(x, 0, 1, LOWEST_DC, HIGHEST_DC);

  in2_PWM = 0;

  in3_PWM = 0;

  in4_PWM = in1_PWM;

  Στη συγκεκριμένη περίπτωση η γωνία είναι 0 μοίρες και το όχημα κινείται μόνο με τη γραμμική του ταχύτητα και κατεύθυνση προς τα εμπρός. Για να επιτευχθεί αυτό τίθεται και στους δύο τροχούς η ίδια ταχύτητα και φορά περιστροφής προς τα εμπρός.

- ***rads = π/2***

  in1_PWM = map(z, 0, 1, LOWEST_DC, HIGHEST_DC);

  In2_PWM = 0;

  in3_PWM = in1_PWM;

  in4_PWM = 0;

  Στη συγκεκριμένη περίπτωση η γωνία είναι 90 μοίρες και το όχημα κινείται μόνο με την περιστροφική του ταχύτητα και περιστρέφεται επιτόπια με ωρολογιακή φορά. Για να επιτευχθεί αυτό τίθεται και στους δύο τροχούς η ίδια ταχύτητα αλλά αντίθετη φορά περιστροφής, με τον δεξιό τροχό να περιστρέφεται προς τα εμπρός.

- ***rads = π***

  in1_PWM = 0;

  in2_PWM = map(-x, 0, 1, LOWEST_DC, HIGHEST_DC);

  in3_PWM = in2_PWM;

  in4_PWM = 0;

  Στη συγκεκριμένη περίπτωση η γωνία είναι 180 μοίρες και το όχημα κινείται μόνο με τη γραμμική του ταχύτητα και κατεύθυνση προς τα πίσω. Για να επιτευχθεί αυτό τίθεται και στους δύο τροχούς η ίδια ταχύτητα και φορά περιστροφής προς τα πίσω.

- ***rads = -π/2***

  in1_PWM = 0;

  in2_PWM = map(-z, 0, 1, LOWEST_DC, HIGHEST_DC);

  in3_PWM = 0;

  in4_PWM = in2_PWM;

  Στη συγκεκριμένη περίπτωση η γωνία είναι -90 μοίρες και το όχημα κινείται μόνο με την περιστροφική του ταχύτητα και περιστρέφεται επιτόπια με αντιωρολογιακή φορά. Για να επιτευχθεί αυτό τίθεται και στους δύο τροχούς η ίδια ταχύτητα αλλά αντίθετη φορά περιστροφής, με τον αριστερό τροχό να περιστρέφεται προς τα εμπρός.

- ***0 < rads < π/2***

  in1_PWM = map(x, 0, 1, LOWEST_DC, HIGHEST_DC);

  in2_PWM = 0;

  in3_PWM = 0;

  *in4_PWM = in1_PWM * cos(rads);*

  Στη συγκεκριμένη περίπτωση η γωνία βρίσκεται στο πρώτο τεταρτημόριο και το όχημα κινείται με συνδυασμό γραμμικής και περιστροφικής ταχύτητας και με κατεύθυνση προς τα εμπρός στρίβοντας ελαφρώς αντιωρολογιακά. Για να επιτευχθεί αυτό πρέπει ο δεξιός τροχός να στρέφεται προς τα εμπρός πιο γρήγορα από τον αριστερό, για τον λόγο

αυτόν στον πρώτο τίθεται η γραμμική ταχύτητα και στον δεύτερο τίθεται το μέτρο της προβολής του διανύσματος ταχύτητας στον άξονα x.

- ***π/2 < rads < π***

  in1_PWM = 0;

  *in2_PWM = in3_PWM * cos(rads);*

  in3_PWM = map(-x, 0, 1, LOWEST_DC, HIGHEST_DC);

  in4_PWM = 0;

  Στη συγκεκριμένη περίπτωση η γωνία βρίσκεται στο δεύτερο τεταρτημόριο και το όχημα κινείται με συνδυασμό γραμμικής και περιστροφικής ταχύτητας και με κατεύθυνση προς τα πίσω στρίβοντας ελαφρώς ωρολογιακά. Για να επιτευχθεί αυτό πρέπει ο δεξιός τροχός να στρέφεται προς τα πίσω πιο γρήγορα από τον αριστερό, για τον λόγο αυτόν στον πρώτο τίθεται η γραμμική ταχύτητα και στον δεύτερο τίθεται το μέτρο της προβολής του διανύσματος ταχύτητας στον άξονα x.

- ***-π/2 < rads < 0***

  *in1_PWM = in4_PWM * cos(rads);*

  in2_PWM = 0;

  in3_PWM = 0;

  in4_PWM = map(x, 0, 1, LOWEST_DC, HIGHEST_DC);

  Στη συγκεκριμένη περίπτωση η γωνία βρίσκεται στο τέταρτο τεταρτημόριο και το όχημα κινείται με συνδυασμό γραμμικής και περιστροφικής ταχύτητας και με κατεύθυνση προς τα εμπρός στρίβοντας ελαφρώς ωρολογιακά. Για να επιτευχθεί αυτό πρέπει ο αριστερός τροχός να στρέφεται προς τα εμπρός πιο γρήγορα από τον δεξιό, για τον λόγο αυτόν στον πρώτο τίθεται η γραμμική ταχύτητα και στον δεύτερο τίθεται το μέτρο της προβολής του διανύσματος ταχύτητας στον άξονα x.

- ***-π < rads < -π/2***

  in1_PWM = 0;

  *in2_PWM = in3_PWM * cos(rads);*

in3_PWM = map(-x, 0, 1, LOWEST_DC, HIGHEST_DC);

in4_PWM = 0;

Στη συγκεκριμένη περίπτωση η γωνία βρίσκεται στο τρίτο τεταρτημόριο και το όχημα κινείται με συνδυασμό γραμμικής και περιστροφικής ταχύτητας και με κατεύθυνση προς τα πίσω στρίβοντας ελαφρώς αντιωρολογιακά. Για να επιτευχθεί αυτό πρέπει ο αριστερός τροχός να στρέφεται προς τα πίσω πιο γρήγορα από τον δεξιό, για τον λόγο αυτόν στον πρώτο τίθεται η γραμμική ταχύτητα και στον δεύτερο τίθεται το μέτρο της προβολής του διανύσματος ταχύτητας στον άξονα x.

- ***x = 0 και z = 0***

  in1_PWM = 0;

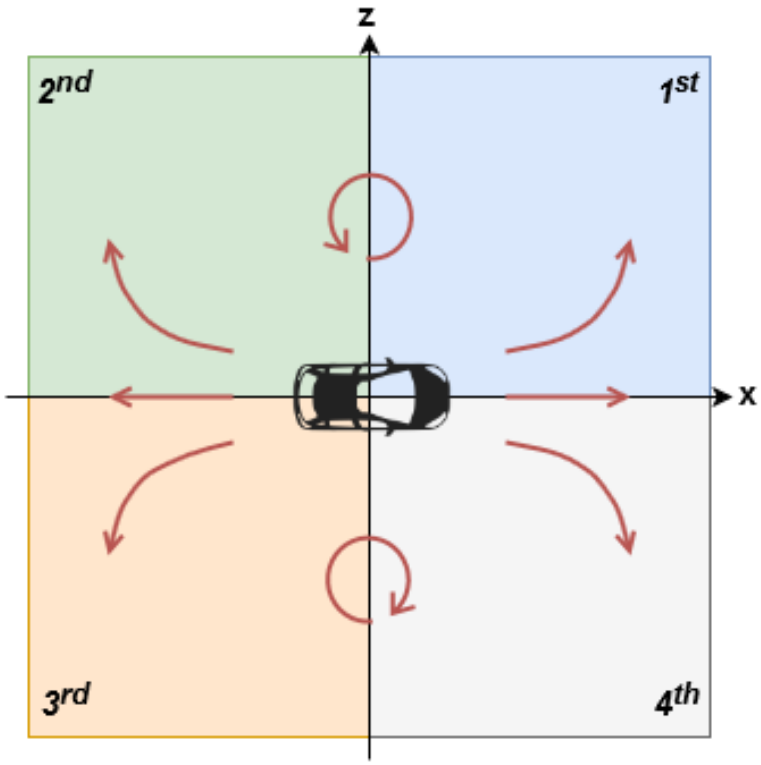
  in2_PWM = 0;

  in3_PWM = 0;

  in4_PWM = 0;

  Στη συγκεκριμένη περίπτωση το όχημα ακινητοποιείται.

Στην ακόλουθη εικόνα παρουσιάζονται οι προαναφερθείσες περιπτώσεις και πιο συγκεκριμένα οι κινήσεις του οχήματος ανάλογα με το σε ποιο τεταρτημόριο ή άξονα του επιπέδου βρίσκεται το ζεύγος **(x, z)**.



Εικόνα 4.20. *Οι κινήσεις του οχήματος ανάλογα με το σε ποιο τεταρτημόριο ή άξονα του επιπέδου βρίσκεται το ζεύγος* ***(x, z)***

4.5.3 Διαδικασία προγραμματισμού του μικροελεγκτή

Στη συγκεκριμένη υλοποίηση δεν χρησιμοποιείται Atmel μικροελεγκτής, αλλά ο ESP8266, παρόλα αυτά η πλατφόρμα Arduino μέσω της μεγάλης κοινότητας της υποστηρίζει τον προγραμματισμό του συγκεκριμένου μικροελεγκτή όπως και πολλών άλλων οικογενειών μικροελεγκτών και μικροεπεξεργαστών σαν να ήταν ένας Atmel. Σε αυτό το σημείο διαφαίνεται το πόσο σημαντικό είναι να χρησιμοποιούνται πλατφόρμες σαν αυτή, οι οποίες έχουν μεγάλη κοινότητα, καθώς υπάρχει ευελιξία στην επιλογή μικροελεγκτών ενώ ο κώδικας τους παραμένει ίδιος.

Ξεκινώντας από την έκδοση 1.6.4, το Arduino IDE επιτρέπει την εγκατάσταση πακέτων τρίτων στην πλατφόρμα του με τη χρήση του *Boards Manager*. Για την εγκατάσταση των απαραίτητων πακέτων για τον μικροελεγκτή ESP8266 ακολουθούνται με σειρά τα επόμενα βήματα:

1. Εκκινούμε το λογισμικό Arduino IDE

2. Ανοίγουμε το παράθυρο των επιλογών από το μενού "*File* > *Preferences*"
3. Εισάγουμε την υπερσύνδεση "*https://arduino.esp8266.com/stable/package_esp8266com_index.json*" στο "*Additional Board Manager URLs*" πεδίο και κλείνουμε το παράθυρο πατώντας το κουμπί "*OK*"
4. Ανοίγουμε τον *Boards Manager* από το μενού "*Tools* > *Board*" της εφαρμογής
5. Αναζητούμε και εγκαθιστούμε την πλατφόρμα "*esp8266*"
6. Αναζητούμε και εγκαθιστούμε μέσα από το μενού "*Sketch* > *Include Library* > *Manage Library*" τη βιβλιοθήκη "*rosserial*"
7. Επιλέγουμε την πλακέτα "*NodeMCU 1.0 (ESP-12E Module)*" από το ίδιο μενού
8. Συνδέουμε τη NodeMCU πλακέτα σε μια USB θύρα του ηλεκτρονικού υπολογιστή μέσω ενός USB καλωδίου
9. Επιλέγουμε από το μενού "*Tools* > *Port*" τη σειριακή θύρα στην οποία είναι συνδεδεμένη η πλακέτα
10. Τέλος, για να φορτωθεί το εκάστοτε πρόγραμμα στον μικροελεγκτή χρησιμοποιείται το κουμπί "*Upload*"

5. Δοκιμή του συστήματος

5.1 Εισαγωγή

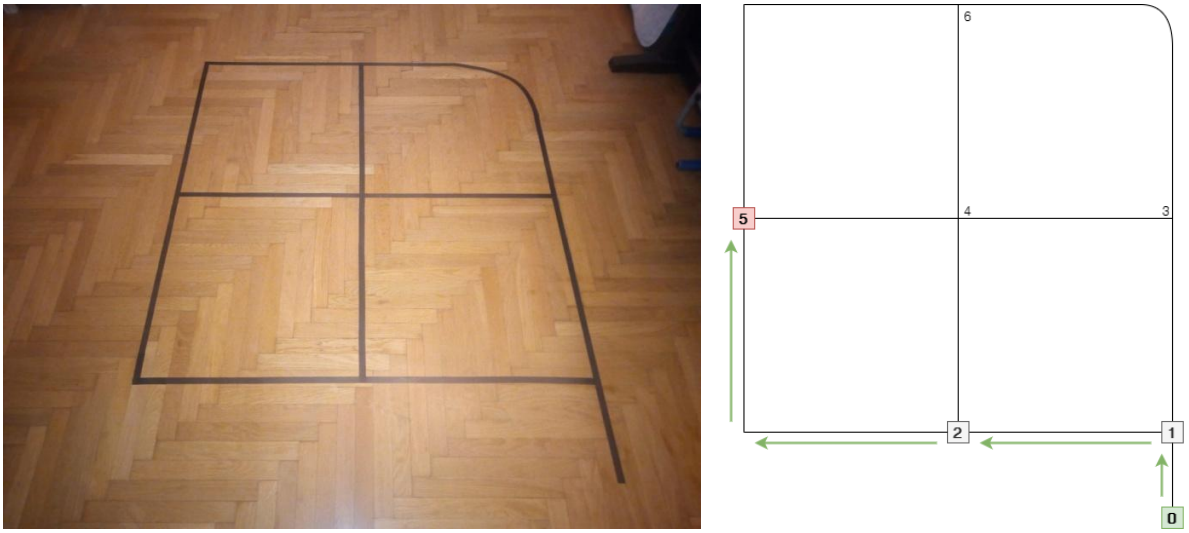
Στο παρόν κεφάλαιο δημιουργείται ένα προς εκτέλεση σενάριο δοκιμής για το υλοποιημένο σύστημα, ώστε να παρουσιαστεί και να ελεγχθεί αν στην ολότητά του είναι λειτουργικό. Μέσα από την εκτέλεση της δοκιμής διαφαίνονται οι αδυναμίες της υλοποίησης και εξάγονται ορισμένα συμπεράσματα για την έκβασή της.

5.2 Σενάριο δοκιμής

Για να φανερωθούν οι αδυναμίες ενός συστήματος πρέπει να δημιουργηθεί ένα σενάριο δοκιμής με μη ευνοϊκές συνθήκες για το σύστημα. Στη συγκεκριμένη εφαρμογή ίσως η πιο μη ευνοϊκή συνθήκη για το σύστημα είναι η έλλειψη φωτισμού στον περιβάλλοντα χώρο. Σε αυτή τη συνθήκη το όχημα θα πρέπει με τον δικό του φωτισμό να πλοηγηθεί και να καταφέρει να φτάσει στο σημείο που θα του υποδείξει ο διαχειριστής. Η συνθήκη χαμηλού φωτισμού εισάγει θόρυβο στους αλγόριθμους όρασης καθώς αλλοιώνει τα χαρακτηριστικά, τα οποία καλούνται εκείνοι να αναγνωρίσουν, όπως για παράδειγμα την αντίθεση μεταξύ δρόμου και δαπέδου. Να σημειωθεί πως για το σενάριο της δοκιμής η μπαταρία του οχήματος είναι πλήρως φορτισμένη, κάτι το οποίο δεν είναι απόλυτα ρεαλιστικό, αλλά σύμβαση, ώστε να μην υπάρχει εναλλαγή στην ένταση του φωτισμού που διαθέτει το όχημα και να μην εμφανιστεί κάποια ανωμαλία κατά την περιστροφή των τροχών.

Μια δοκιμή για την πιστότητα του συστήματος πλοήγησης είναι ο διαχειριστής να επιλέξει μια διαδρομή η οποία θα περιέχει διαφορετικά είδη σταυροδρομιών,

είτε είναι γωνίες είτε κανονικά σταυροδρόμια, ώστε ο αλγόριθμος πλοήγησης να είναι σε θέση να εντοπίσει όλα τα είδη. Για τον λόγο αυτόν από τον ακόλουθο χώρο πλοήγησης επιλέγεται η διαδρομή με αρχή τον κόμβο **0** και τερματισμό τον **5** μέσω των κόμβων **1**, **2** και μιας γωνίας.



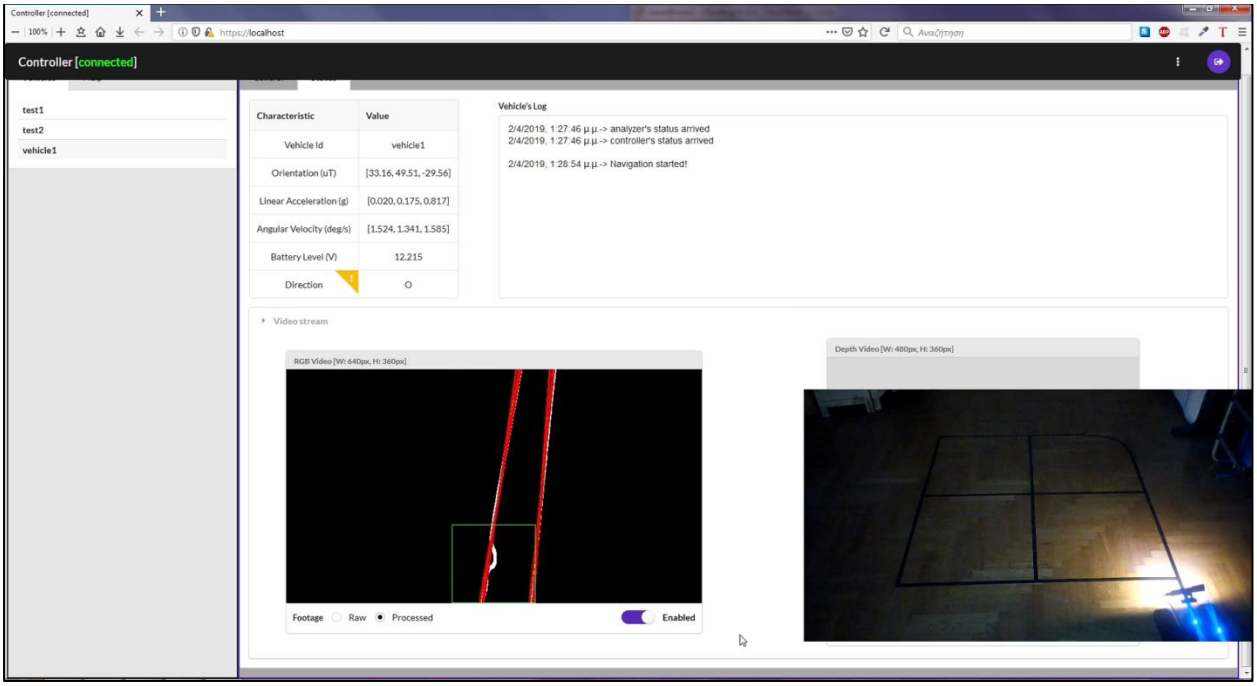
Εικόνα 5.1. *Φυσική διαδρομή (αριστερά) και αντίστοιχος γράφος με επισημασμένη τη δοκιμαστική διαδρομή (δεξιά)*

Προφανώς η συγκεκριμένη δεν θεωρείται η πιο απαιτητική διαδρομή, απλώς για χάρη συντομίας στην περιγραφή επιλέχτηκε η συγκεκριμένη. Εναλλακτικά, μια από τις πιο σύνθετες διαδρομές η οποία έχει εκτελεστεί επιτυχώς, είναι εκείνη η οποία έχει ως εκκίνηση τον κόμβο **0** και τερματισμό τον **5** μέσω των κόμβων **1**, **2**, **4**, **3** και **6**.

5.3 Εκτέλεση δοκιμής

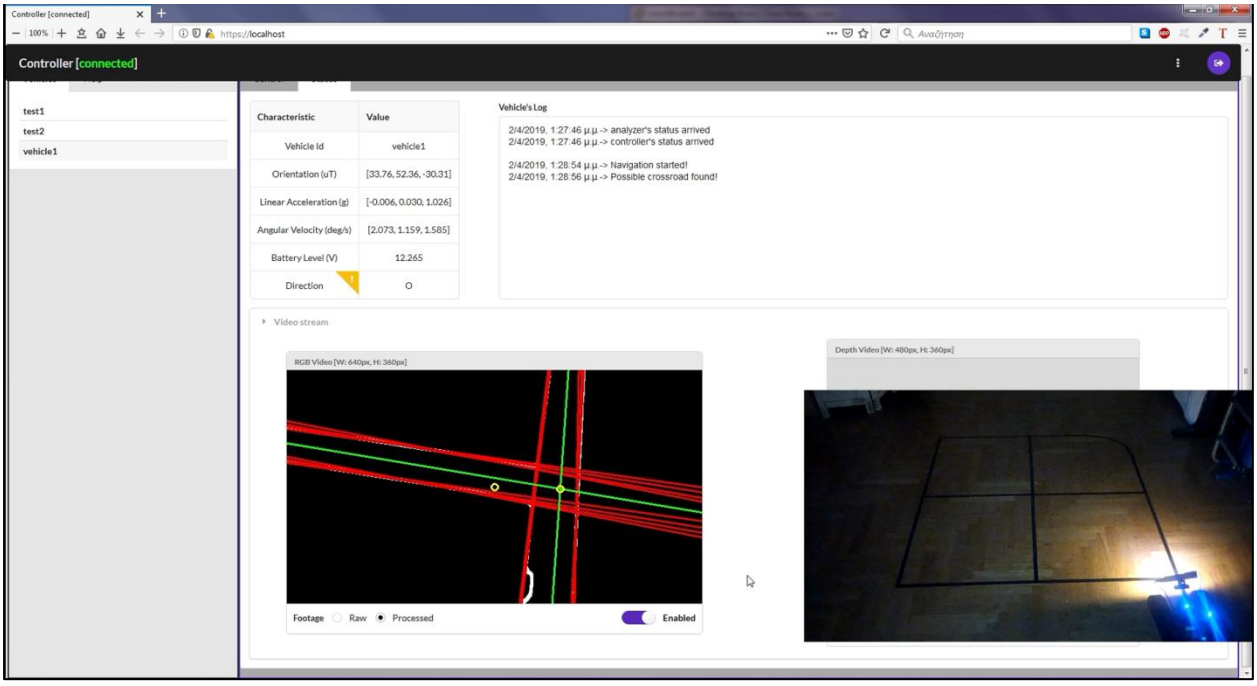
Κατά τη διάρκεια της δοκιμής παρατίθενται πλάνα από την επεξεργασμένη εικόνα την οποία αποστέλλει το όχημα στη διεπαφή του διαχειριστή, συνοδευομένη από τα μηνύματα τα οποία το ίδιο στέλνει κατά τις εναλλαγές των καταστάσεών του. Στα ίδια πλάνα εμπεριέχονται και πλάνα από εξωτερική κάμερα τα οποία δείχνουν τη θέση του οχήματος πάνω στις διαδρομές. Πριν την έναρξη

της διαδρομής το όχημα έχει τοποθετηθεί στον δρόμο μεταξύ του κόμβου **0** και του κόμβου **1** με κατεύθυνση προς τον δεύτερο. Μόλις εκκινήσει η πλοήγηση, η διεπαφή μεταβαίνει αυτόματα στην καρτέλα Status, τα στοιχεία του οχήματος ξεκινούν να ενημερώνονται καθώς και η εικόνα από την κάμερά του. Ως υπενθύμιση να σημειωθεί πως οι κόκκινες γραμμές είναι οι ευθείες που εντοπίζει ο αλγόριθμος όρασης και εκτείνονται από τη μία άκρη της εικόνας στην άλλη, το τετράγωνο με το πράσινο περίγραμμα στο κάτω μέρος της εικόνας προσδιορίζει την περιοχή με το μέγιστο σταθμισμένο άθροισμα, οι πράσινες γραμμές προέρχονται από την ομαδοποίηση των κόκκινων γραμμών σε δύο ομάδες και οι κίτρινοι κύκλοι δείχνουν ο ένας την τομή των πράσινων γραμμών ενώ ο άλλος το κέντρο της εικόνας.



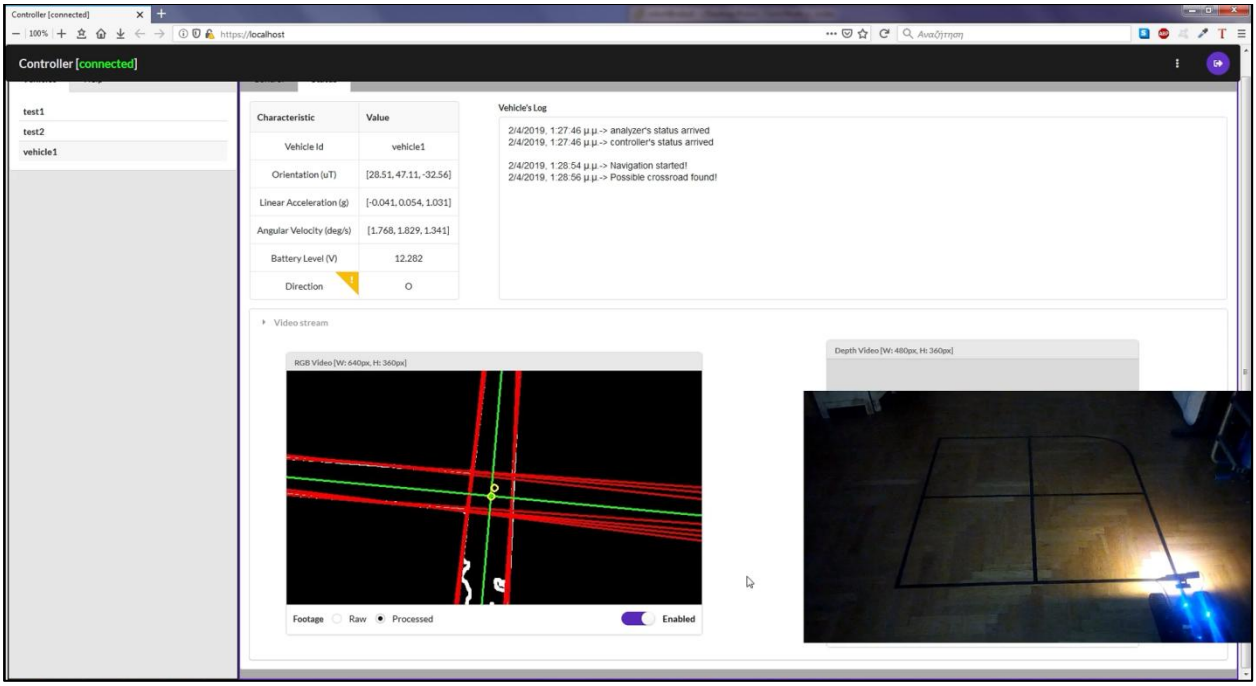
Εικόνα 5.2. *Στιγμιότυπο κατά την εκκίνηση της αυτόνομης πλοήγησης*

Μόλις ο αλγόριθμος όρασης εντοπίσει το πρώτο σταυροδρόμι, ενημερώνει τον διαχειριστή ότι βρήκε πιθανό σταυροδρόμι και μεταβαίνει στην κατάσταση Centering Crossroad.



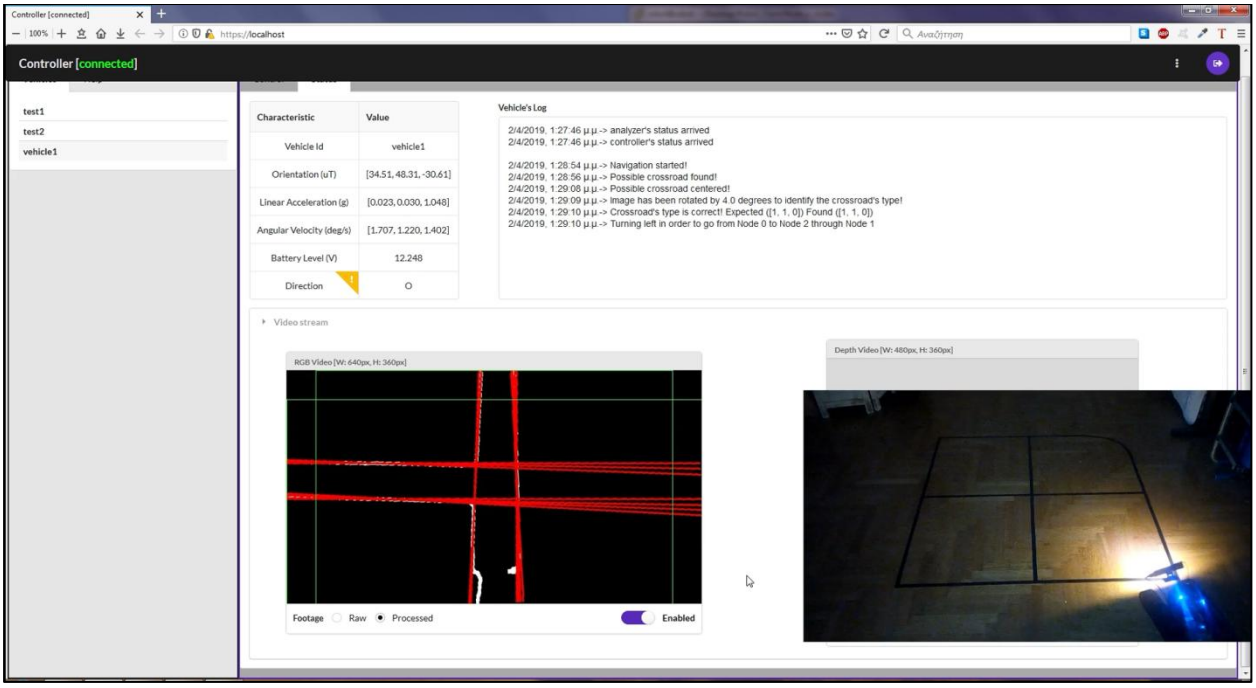
Εικόνα 5.3. *Στιγμιότυπο κατά το οποίο το όχημα έχει εντοπίσει πιθανό σταυροδρόμι και μεταβαίνει στην κατάσταση Centering Crossroad*

Στη συγκεκριμένη κατάσταση το όχημα κινείται όπως έχει περιγραφεί παραπάνω έως ότου ταυτιστούν τα κέντρα του σταυροδρομιού και της εικόνας.



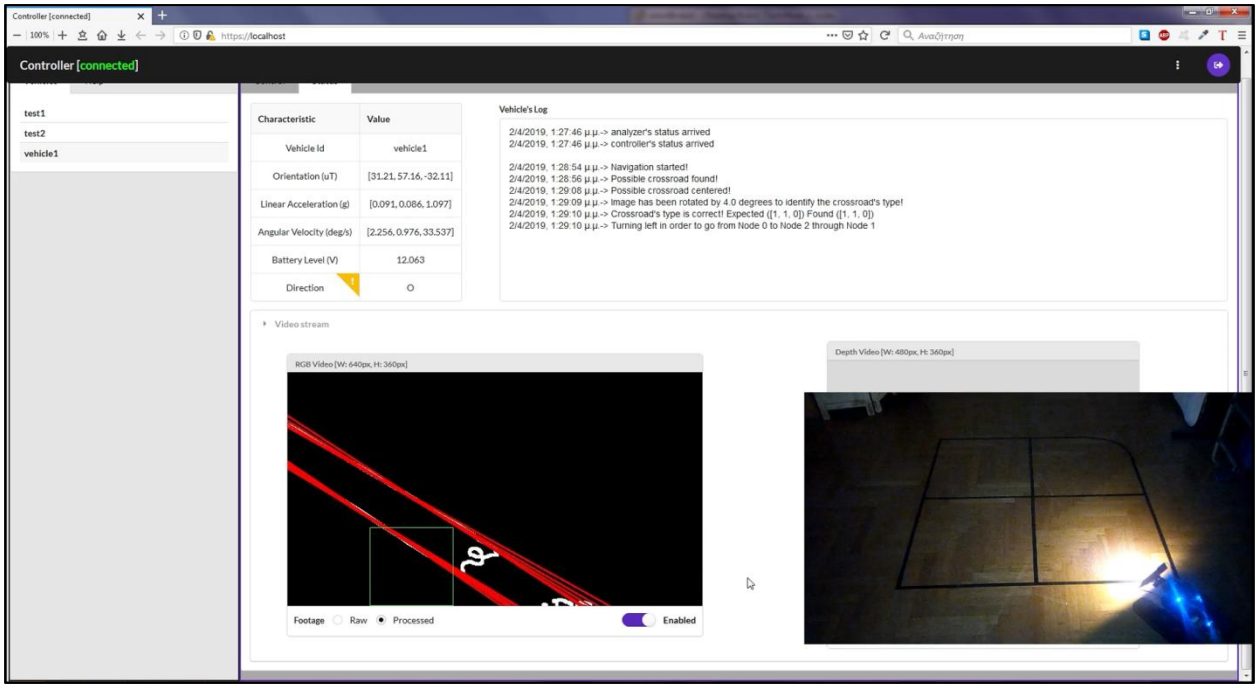
Εικόνα 5.4. *Στιγμιότυπο κατά το οποίο το κέντρο του σταυροδρομιού έχει ταυτιστεί με το κέντρο της εικόνας*

Μόλις συμπέσουν τα δύο κέντρα, ο αλγόριθμος μεταβαίνει στην κατάσταση Examining Crossroad για να εντοπίσει τον τύπο του σταυροδρομιού, εάν έπρεπε να συναντήσει αυτόν τον τύπο και προς ποια κατεύθυνση πρέπει να κινηθεί ώστε να φτάσει στον επόμενο κόμβο. Στο ακόλουθο στιγμιότυπο ο αλγόριθμος εντοπίζει τον σωστό τύπο σταυροδρομιού και ότι ανήκει στον κόμβο **1**, εφόσον γνωρίζει ότι έρχεται από τον **0**, και δίνει εντολή στο όχημα να στρίψει αριστερά ώστε να πλοηγηθεί προς τον κόμβο **2**.



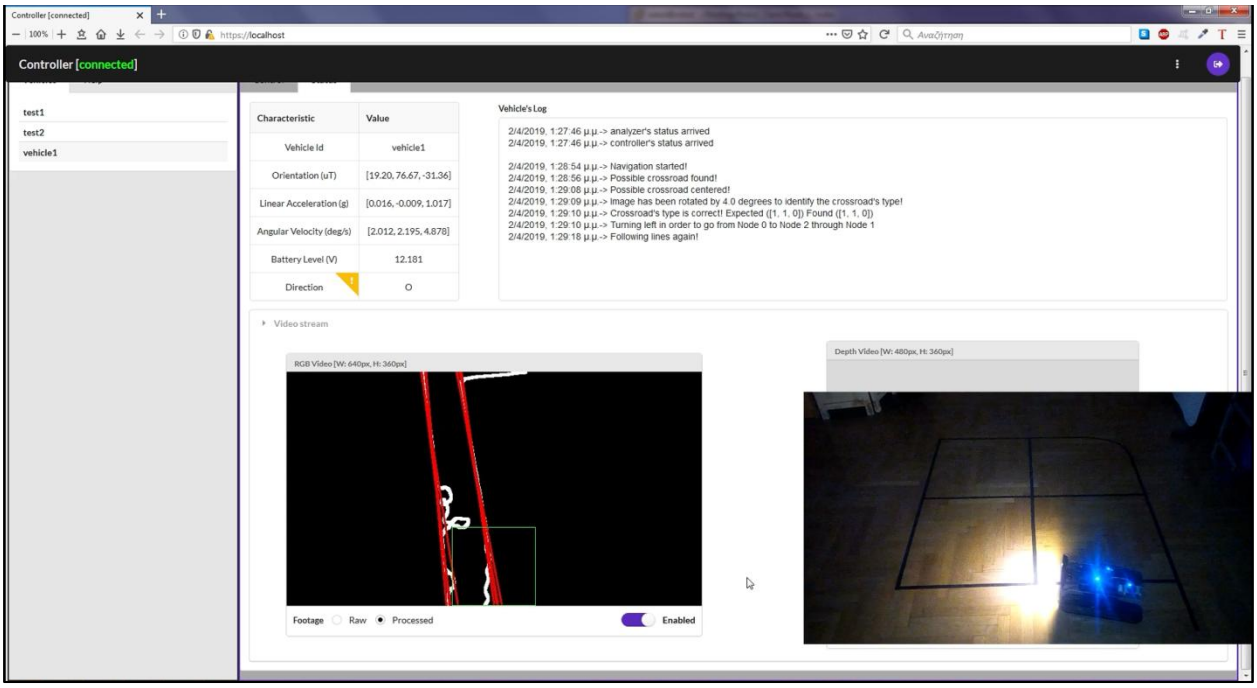
Εικόνα 5.5. *Στιγμιότυπο στο οποίο ο αλγόριθμος εντοπίζει τον τύπο του σταυροδρομιού και την κατεύθυνση στην οποία πρέπει να πλοηγηθεί*

Στο ακόλουθο στιγμιότυπο το όχημα μεταβαίνει για λίγο χρονικό διάστημα στην κατάσταση Turning Crossroad όπου ακολουθεί τον δρόμο με pattern [1.0, 0.0, 0.0] ώστε να στρίψει αριστερά.



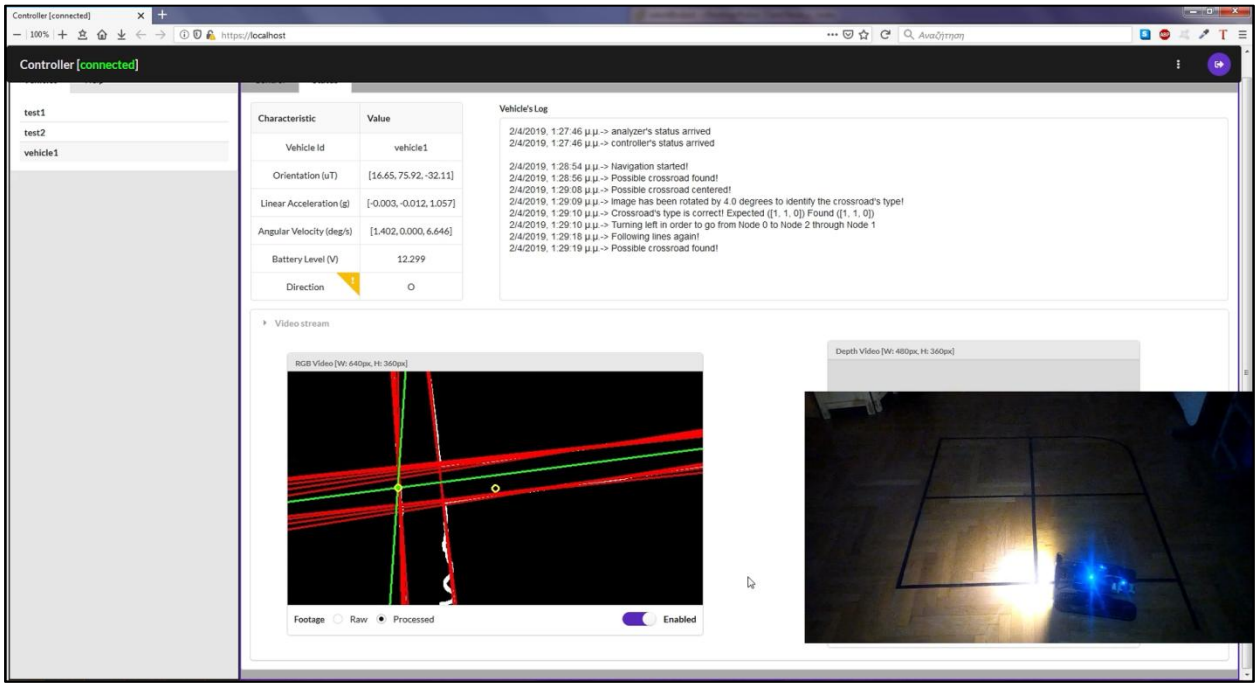
Εικόνα 5.6. *Στιγμιότυπο κατά το οποίο το όχημα βρίσκεται στην κατάσταση Turning Crossroad*

Μετά το πέρας του χρονικού ορίου επιστρέφει στην κατάσταση Following Route, όπου το pattern γίνεται ξανά [1.0, 1.0, 1.0].

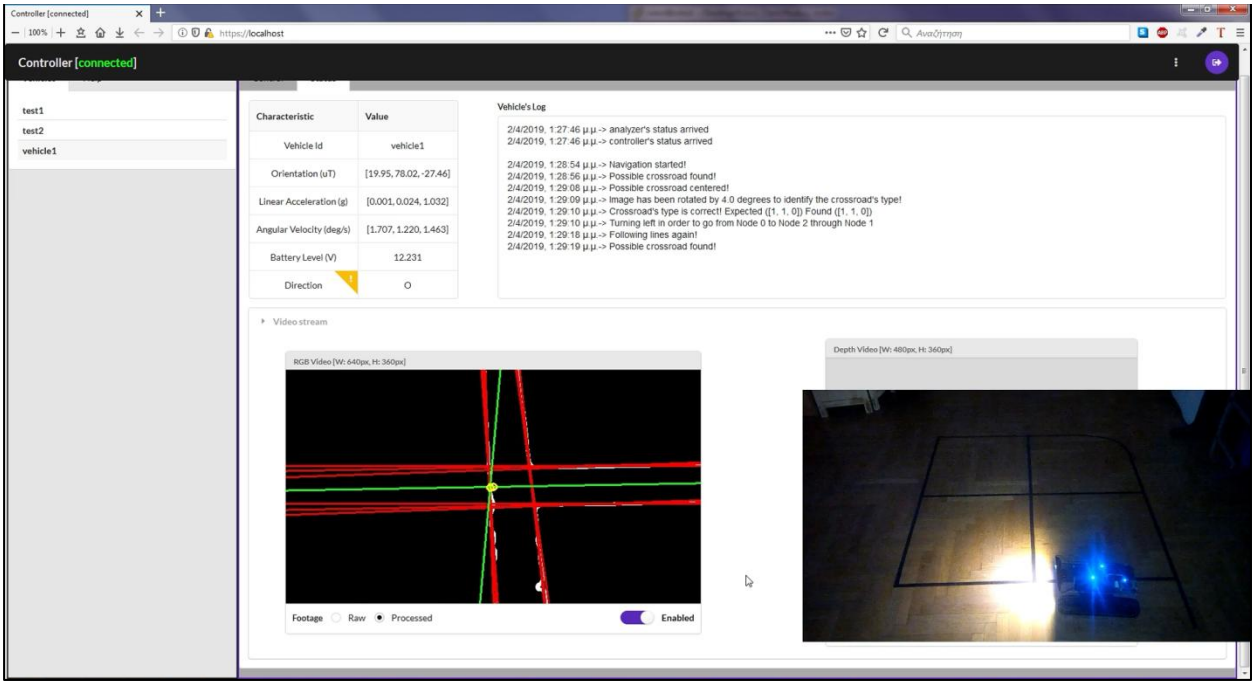


Εικόνα 5.7. *Στιγμιότυπο στο οποίο το όχημα μεταβαίνει στην κατάσταση Following Route*

Στο ακόλουθο στιγμιότυπο ο αλγόριθμος εντοπίζει πιθανό σταυροδρόμι και μεταβαίνει στην κατάσταση Centering Crossroad.

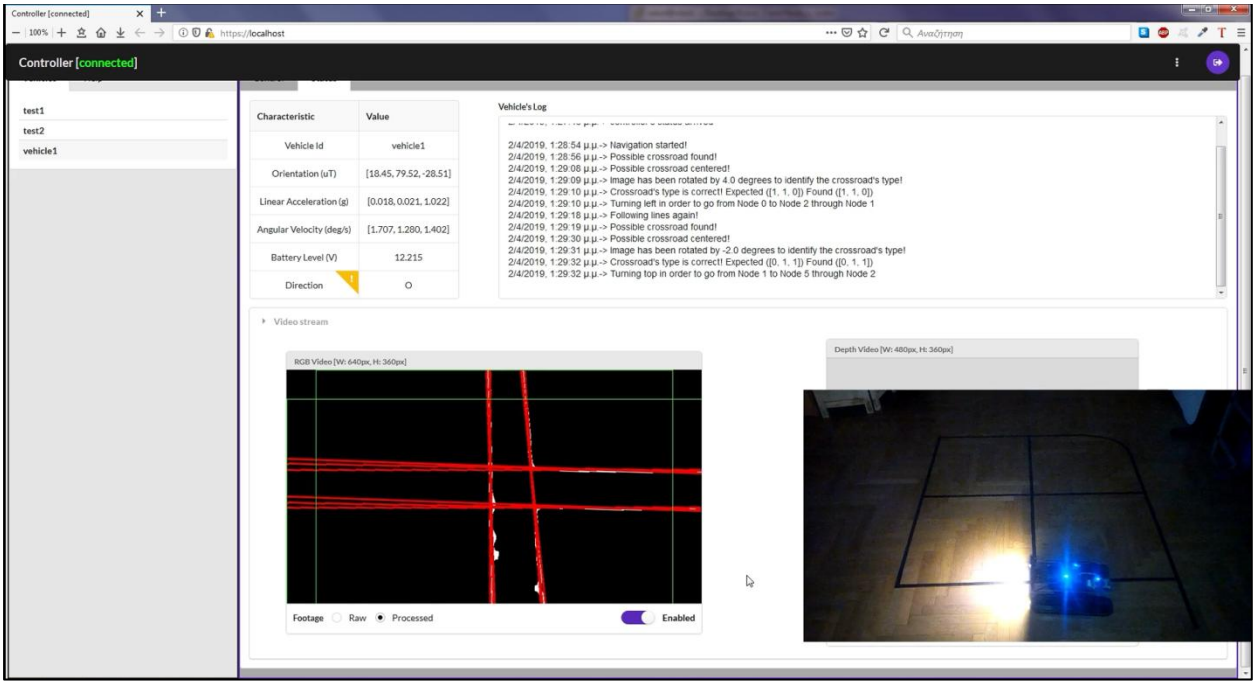


Εικόνα 5.8. *Στιγμιότυπο κατά το οποίο το όχημα έχει εντοπίσει πιθανό σταυροδρόμι και μεταβαίνει στην κατάσταση Centering Crossroad*



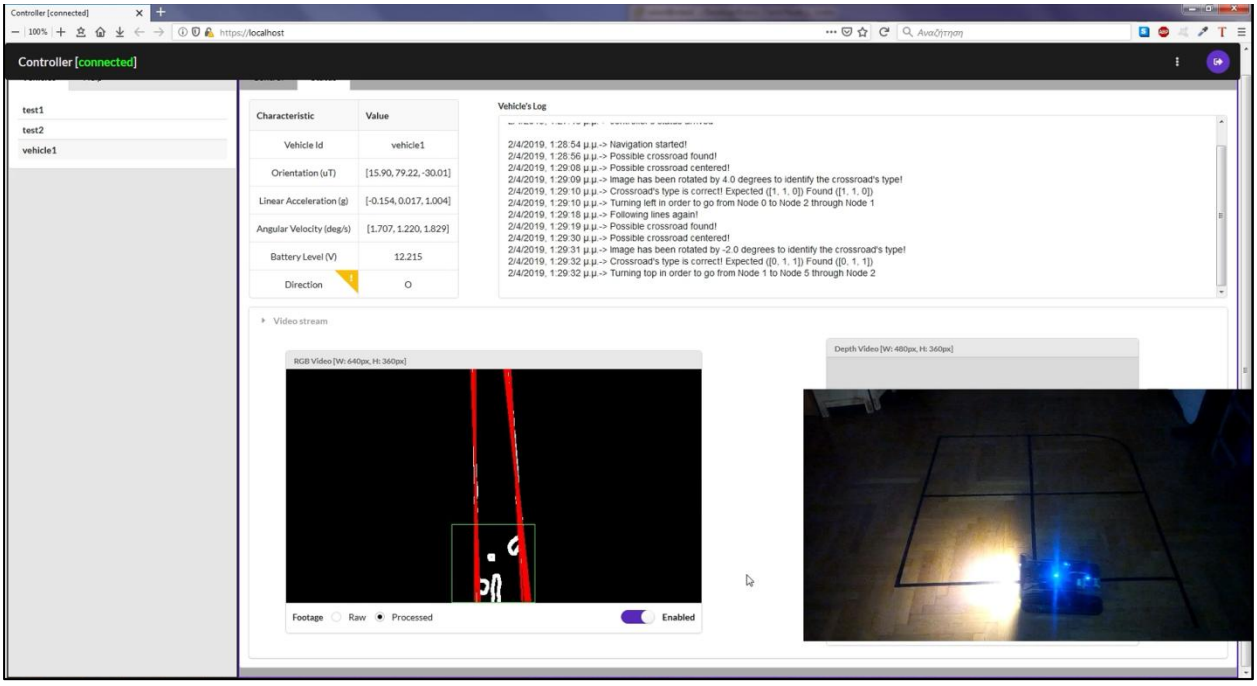
Εικόνα 5.9. *Στιγμιότυπο κατά το οποίο το κέντρο του σταυροδρομιού έχει ταυτιστεί με το κέντρο της εικόνας*

Μόλις συμπέσουν τα δύο κέντρα, ο αλγόριθμος μεταβαίνει στην κατάσταση Examining Crossroad όπου εντοπίζει τον σωστό τύπο σταυροδρομιού και ότι ανήκει στον κόμβο **2**, εφόσον γνωρίζει ότι έρχεται από τον **1**, και δίνει εντολή στο όχημα να πορευτεί ευθεία ώστε να πλοηγηθεί προς τον κόμβο **5**.



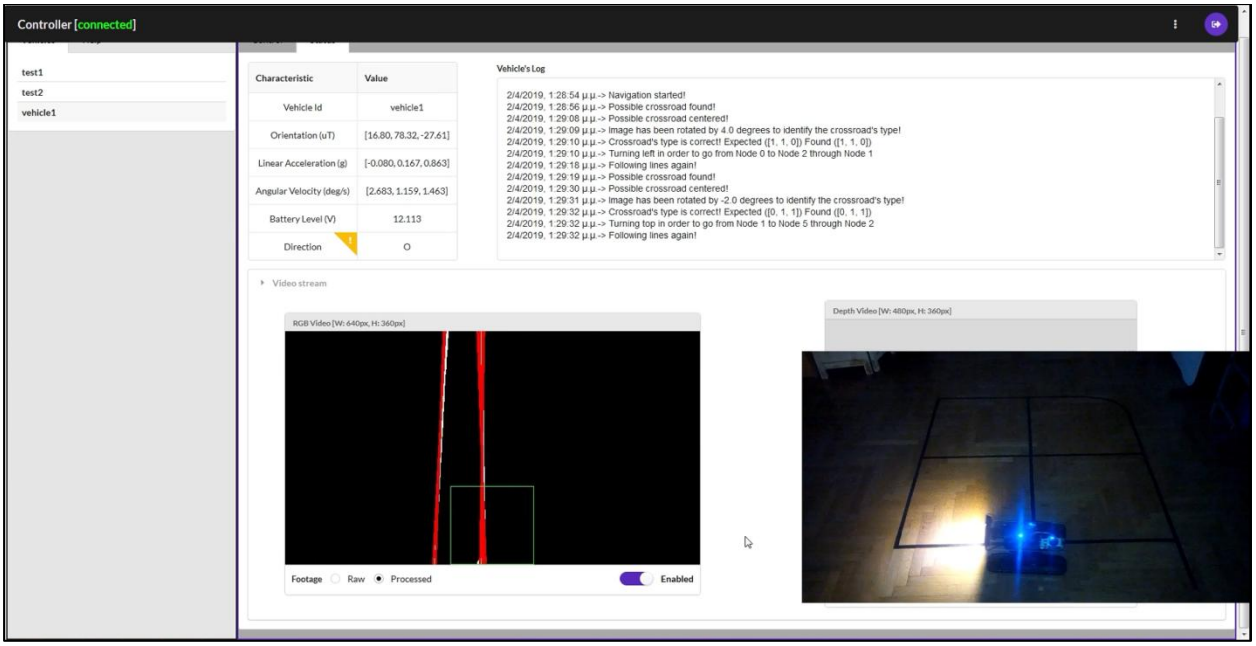
Εικόνα 5.10. *Στιγμιότυπο κατά το οποίο ο αλγόριθμος εντοπίζει τον τύπο του σταυροδρομιού και την κατεύθυνση στην οποία πρέπει να πλοηγηθεί*

Στο ακόλουθο στιγμιότυπο το όχημα μεταβαίνει για λίγο χρονικό διάστημα στην κατάσταση Turning Crossroad όπου ακολουθεί τον δρόμο με pattern [0.0, 1.0, 0.0] ώστε να ακολουθήσει ευθεία πορεία.



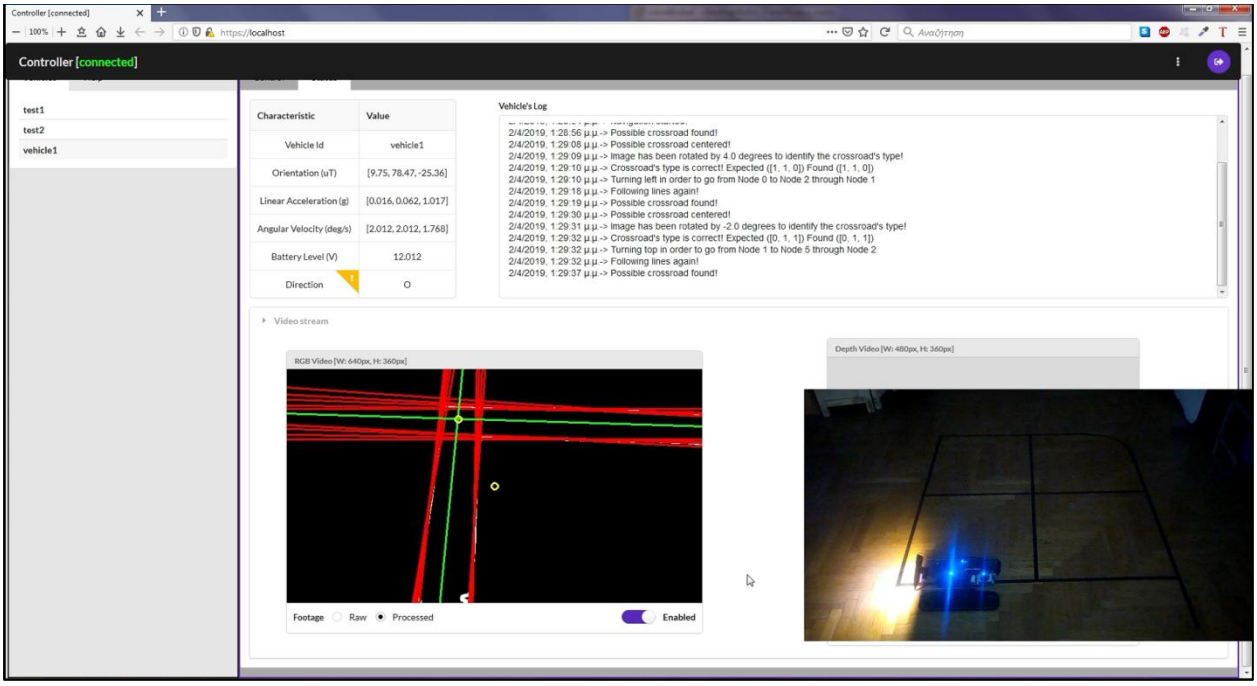
Εικόνα 5.11. *Στιγμιότυπο κατά το οποίο το όχημα βρίσκεται στην κατάσταση Turning Crossroad*

Μετά το πέρας του χρονικού ορίου επιστρέφει στην κατάσταση Following Route, όπου το pattern γίνεται ξανά [1.0, 1.0, 1.0].

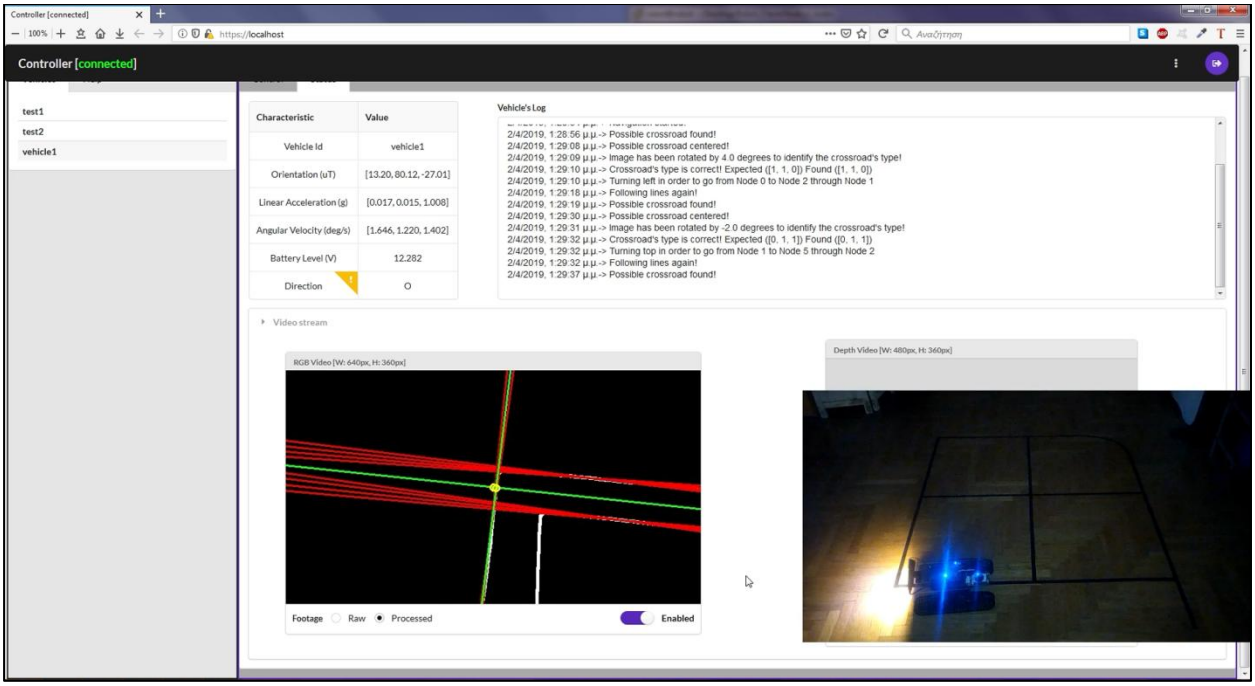


Εικόνα 5.12. *Στιγμιότυπο κατά το οποίο το όχημα μεταβαίνει στην κατάσταση Following Route*

Στο ακόλουθο στιγμιότυπο ο αλγόριθμος εντοπίζει πιθανό σταυροδρόμι και μεταβαίνει στην κατάσταση Centering Crossroad.

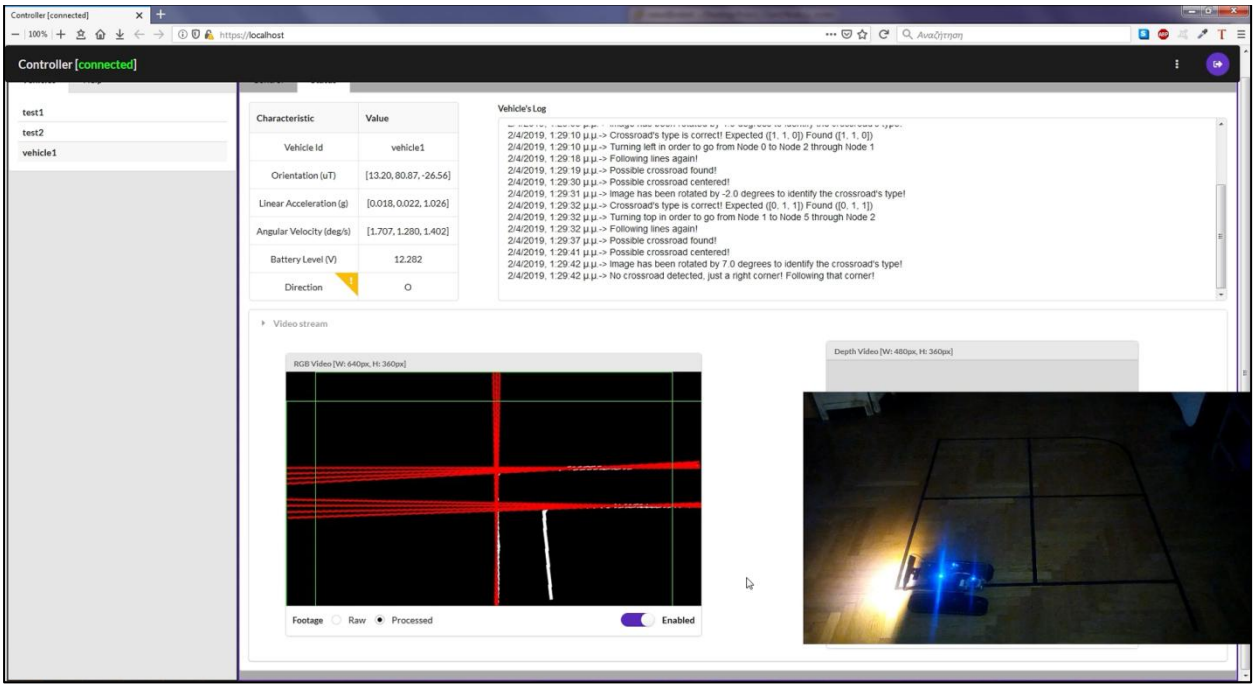


Εικόνα 5.13. *Στιγμιότυπο κατά το οποίο το όχημα έχει εντοπίσει πιθανό σταυροδρόμι και μεταβαίνει στην κατάσταση Centering Crossroad*



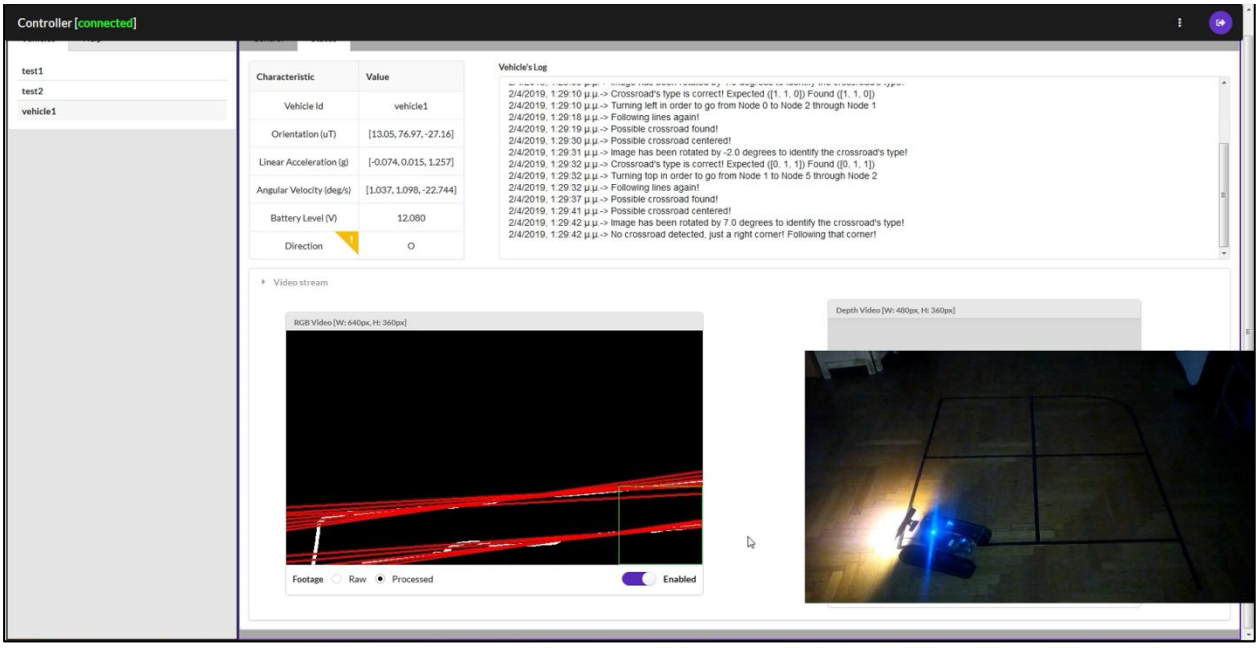
Εικόνα 5.14. *Στιγμιότυπο κατά το οποίο το κέντρο του σταυροδρομιού έχει ταυτιστεί με το κέντρο της εικόνας*

Μόλις συμπέσουν τα δύο κέντρα, ο αλγόριθμος μεταβαίνει στην κατάσταση Examining Crossroad όπου εντοπίζει τον σωστό τύπο σταυροδρομιού ο οποίος είναι δεξιά στροφή και δίνει εντολή στο όχημα να στρίψει δεξιά.



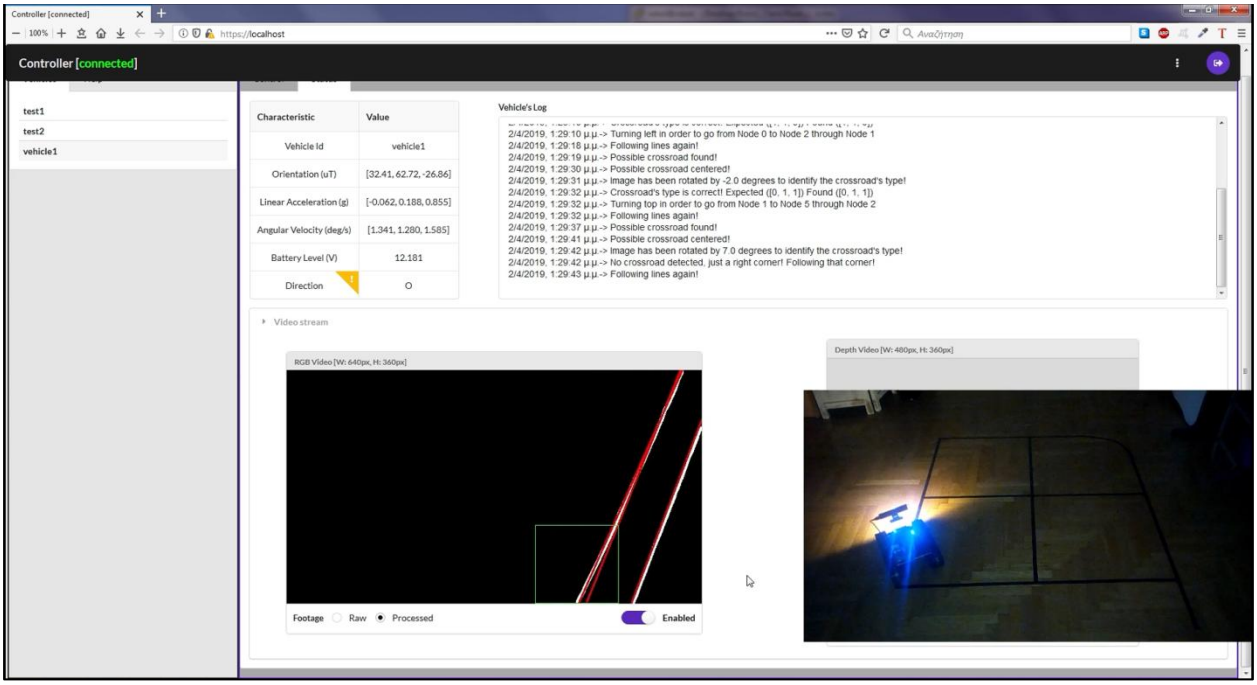
Εικόνα 5.15. *Στιγμιότυπο κατά το οποίο ο αλγόριθμος εντοπίζει τον τύπο του σταυροδρομιού και την κατεύθυνση στην οποία πρέπει να πλοηγηθεί*

Στο ακόλουθο στιγμιότυπο το όχημα μεταβαίνει για λίγο χρονικό διάστημα στην κατάσταση Turning Crossroad όπου ακολουθεί τον δρόμο με pattern [0.0, 0.0, 1.0] ώστε να στρίψει δεξιά.



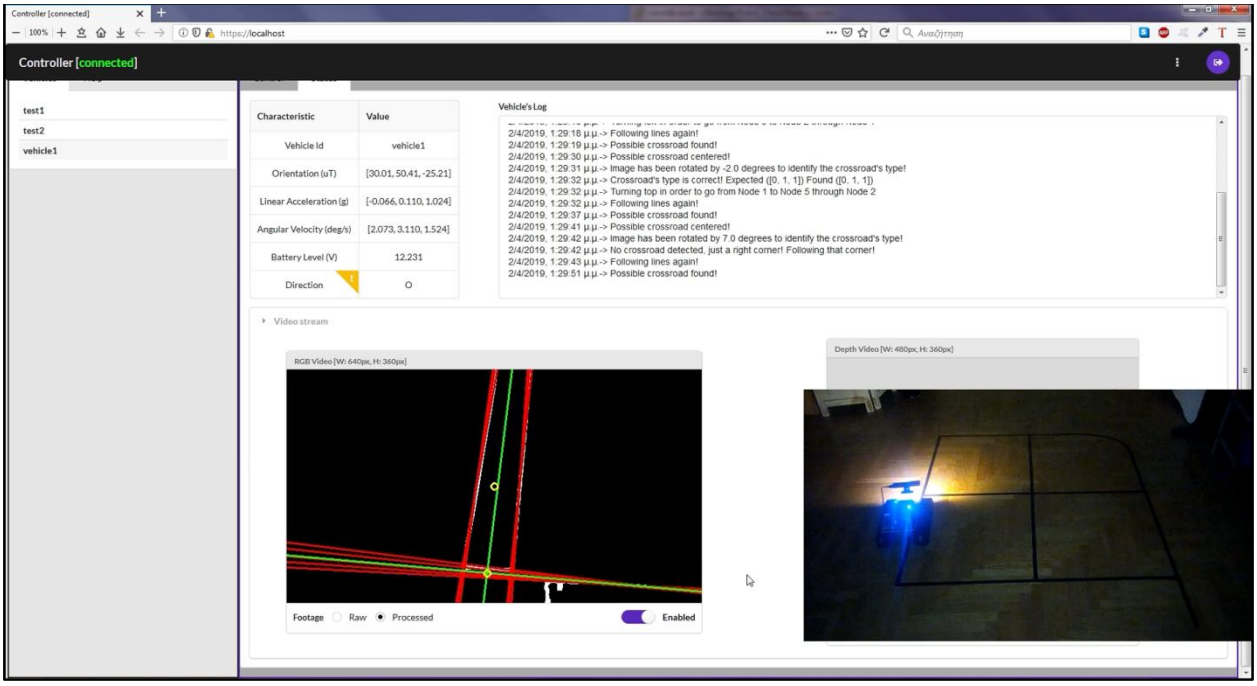
Εικόνα 5.16. *Στιγμιότυπο κατά το οποίο το όχημα βρίσκεται στην κατάσταση Turning Crossroad*

Μετά το πέρας του χρονικού ορίου επιστρέφει στην κατάσταση Following Route, όπου το pattern γίνεται ξανά [1.0, 1.0, 1.0].

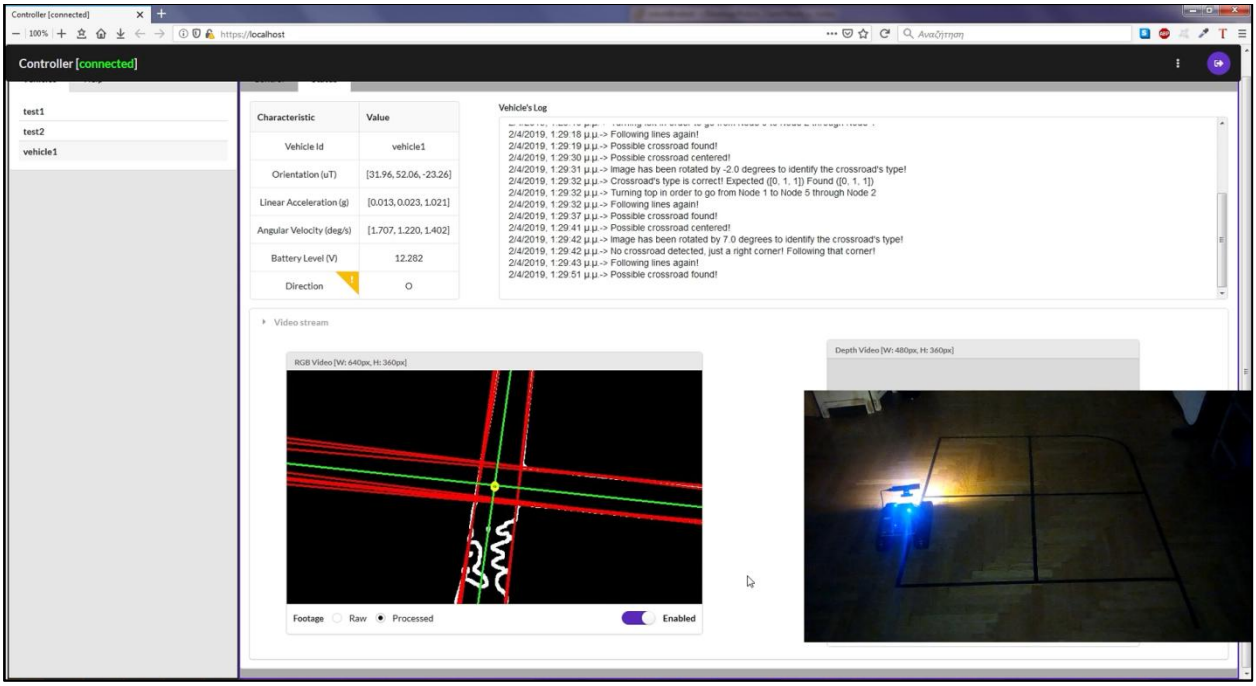


Εικόνα 5.17. *Στιγμιότυπο κατά το οποίο το όχημα μεταβαίνει στην κατάσταση Following Route*

Στο ακόλουθο στιγμιότυπο ο αλγόριθμος εντοπίζει πιθανό σταυροδρόμι και μεταβαίνει στην κατάσταση Centering Crossroad.

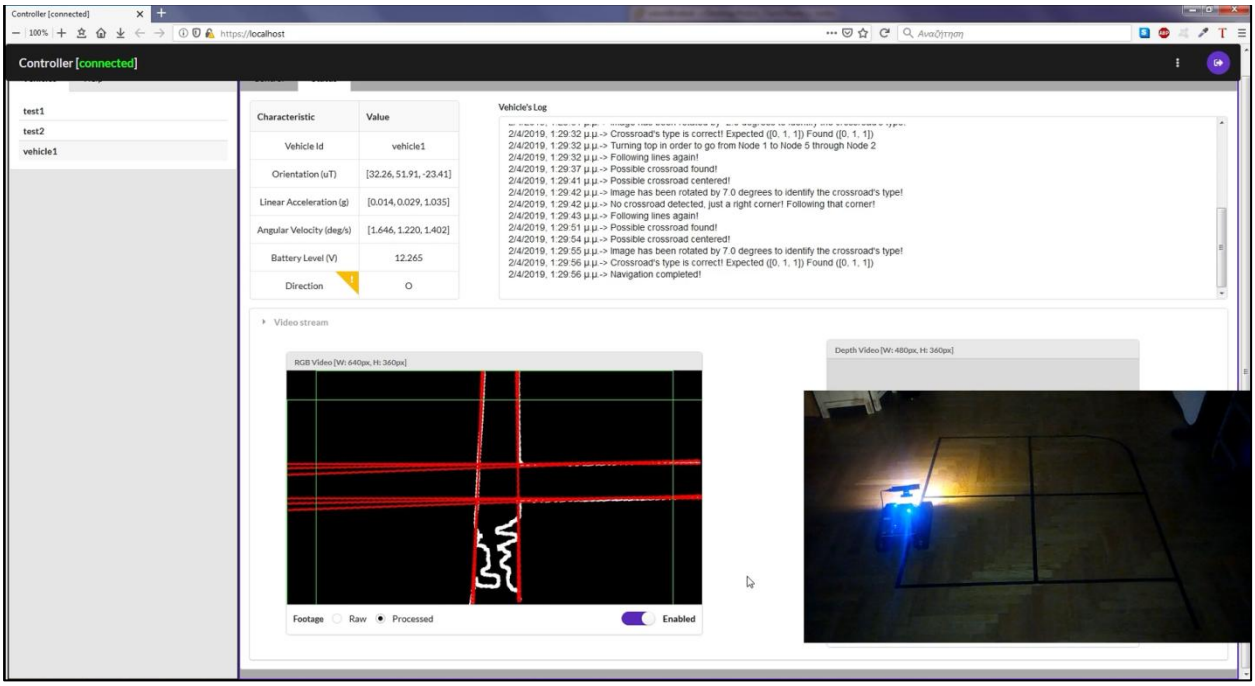


Εικόνα 5.18. *Στιγμιότυπο κατά το οποίο το όχημα έχει εντοπίσει πιθανό σταυροδρόμι και μεταβαίνει στην κατάσταση Centering Crossroad*



Εικόνα 5.19. *Στιγμιότυπο κατά το οποίο το κέντρο του σταυροδρομιού έχει ταυτιστεί με το κέντρο της εικόνας*

Μόλις συμπέσουν τα δύο κέντρα, ο αλγόριθμος μεταβαίνει στην κατάσταση Examining Crossroad όπου εντοπίζει τον σωστό τύπο σταυροδρομιού και ότι ανήκει στον κόμβο **5**, εφόσον γνωρίζει ότι έρχεται από τον **2**, και εφόσον είναι ο τελευταίος κόμβος της διαδρομής μεταβαίνει στην κατάσταση Route Done ενημερώνοντας παράλληλα και τον διαχειριστή.



Εικόνα 5.20. *Στιγμιότυπο κατά το οποίο ο αλγόριθμος εντοπίζει τον τύπο του σταυροδρομιού και ότι είναι ο τελευταίος κόμβος της διαδρομής*

5.4 Αποτελέσματα δοκιμής

Συμπερασματικά, η εκτέλεση της δοκιμής ήταν επιτυχής ακόμα και με την έκθεση του συστήματος σε δυσμενείς συνθήκες, όπως αυτή του χαμηλού περιβάλλοντα φωτισμού. Παρόλη την επιτυχία της δοκιμής, διαφάνηκαν ορισμένες πτυχές του αλγορίθμου όρασης οι οποίες χρειάζονται βελτίωση και επιπλέον έλεγχο. Όπως έχει ειπωθεί, οι αλγόριθμοι όρασης και πλοήγησης έχουν υλοποιηθεί στο πλαίσιο της πρωτοτυποποίησης του συστήματος και είναι κατάλληλοι για συγκεκριμένο σενάριο χρήσης του συστήματος και όχι για γενικευμένα σενάρια.

Αρχικά, στα περισσότερα στιγμιότυπα και ειδικά στις Εικόνες 5.19 και 5.20 υπάρχει αυξημένος θόρυβος (κατσαρές λευκές καμπύλες) ο οποίος εισάγεται στον αλγόριθμο όρασης λόγω της έλλειψης επαρκούς περιβάλλοντα φωτισμού. Παρόλα αυτά, ο αλγόριθμος ανίχνευσης γραμμών δεν επηρεάζεται από τον συγκεκριμένο θόρυβο, καθώς οι γραμμές είναι καμπυλοειδείς, αλλά και στα σημεία που υπάρχουν όντως ευθείες γραμμές είναι μικρού μήκους και ο αλγόριθμος τις αγνοεί. Παρόλο που δεν φαίνεται να υπάρχει κάποιο πρόβλημα, στην περίπτωση που ο συγκεκριμένος θόρυβος ήταν εκτός του δρόμου ίσως υπήρχε πρόβλημα κατά τον εντοπισμό του τύπου του σταυροδρομιού ή ακόμα και ίσως να αναγνωριζόταν κάποια ευθεία. Για τον λόγο αυτόν χρειάζεται καλύτερο φιλτράρισμα της εικόνας στις συνθήκες χαμηλού φωτισμού, αλλά και πιο συνθέτη λογική για το φιλτράρισμα των γραμμών οι οποίες εντοπίζονται.

Στη συνέχεια, στις Εικόνες 5.14, 5.15 και 5.18 διαφαίνεται το πρόβλημα της μη ανίχνευσης ορισμένων ευθειών από τον αρμόδιο αλγόριθμο. Πιο συγκεκριμένα, στις Εικόνες 5.14 και 5.15 ενώ είναι εμφανές το δεξί περίγραμμα του δρόμου ο αλγόριθμος δεν έχει εντοπίσει κάποια ευθεία πάνω σε αυτόν. Στην παρούσα δοκιμή η αναγνώριση επετεύχθη χρησιμοποιώντας το αριστερό περίγραμμα του δρόμου, όμως σε άλλες δοκιμές και ανεξαρτήτως περιβάλλοντα φωτισμού έχουν υπάρξει σφάλματα κατά την πλοήγηση. Στην Εικόνα 5.18 το σταυροδρόμι αναγνωρίζεται οριακά λίγο πριν εξαφανιστεί από την εικόνα διότι μέχρι εκείνη τη

στιγμή δεν είχε εντοπιστεί, ενώ ήταν πάλι εμφανής, η δεξιά στροφή. Για την αποφυγή αυτού του προβλήματος οφείλουν να γίνουν περισσότερες δοκιμές και να ρυθμιστούν καταλληλότερα οι παράμετροι του αλγορίθμου ανίχνευσης γραμμών.

Τέλος, στις Εικόνες 5.8 και 5.9 ενώ έχουν εντοπιστεί ορθά οι ευθείες, η εγκάρσια πράσινη γραμμή είναι η συμμετρική ως προς τον κάθετο άξονα αυτής που αναμενόταν με βάση τις γωνίες των εγκάρσιων κόκκινων ευθειών. Αυτό το σφάλμα συμβαίνει σε τιμές γωνιών μικρότερες των 10 μοιρών και αφορά μόνο στην απεικόνιση των συγκεκριμένων γραμμών, καθώς όπως φαίνεται στην Εικόνα 5.10 υπάρχει κανονική περιστροφή της εικόνας για τον εντοπισμό του τύπου του σταυροδρομιού.

6. Βελτιώσεις και Συμπεράσματα

6.1 Εισαγωγή

Η παρούσα εργασία αποτελεί μία ολοκληρωμένη προσπάθεια ανάπτυξης ενός ολοκληρωμένου συστήματος παρακολούθησης και αυτόνομης οδήγησης οχημάτων εξ' ολοκλήρου με χρήση τεχνικών ανάλυσης εικόνας λαμβάνοντας υπόψη τις ιδιαίτερες συνθήκες που επικρατούν στους χώρους στους οποίους θα μπορούσε να εφαρμοστεί. Στο πλαίσιο, ωστόσο, του εγχειρήματος αυτού ήταν απαραίτητο να γίνουν κάποιες απλοποιήσεις και εκ των προτέρων θεωρήσεις, με αποτέλεσμα το αναπτυχθέν σύστημα να επιδέχεται βελτιώσεων. Στο παρόν κεφάλαιο παρουσιάζονται, αρχικά, μερικές από τις βελτιώσεις τις οποίες επιδέχεται το σύστημα και, στη συνέχεια εξάγονται τα τελικά συμπεράσματα για την πορεία και την έκβαση της παρούσας εργασίας.

6.2 Βελτιώσεις συστήματος

Η βασικότερη βελτίωση του συστήματος συνίσταται στη βελτίωση των αλγόριθμων όρασης και πλοήγησης του οχήματος. Οι συγκεκριμένοι υλοποιήθηκαν με αυτόν τον τρόπο για λόγους πρωτοτυποποίησης του συστήματος ώστε να μπορούν να δουλεύουν αρκούντως καλά στις θεωρήσεις που έγιναν για τον τρόπο λειτουργίας του. Για παράδειγμα, δεν υποστηρίζουν την πραγματική ταυτοποίηση του κόμβου τον οποίον συναντούν, η οποία θα μπορούσε να γίνει με αναγνώριση κάποιας μοναδικής για κάθε κόμβο σήμανσης, αλλά στηρίζονται στη γνώση του προηγούμενου κόμβου. Ένα ακόμα αρνητικό είναι ο τρόπος με τον οποίο στρίβει το όχημα ο οποίος ναι μεν δουλεύει επαρκώς καλά, αλλά στηρίζεται στην τυφλή οδήγηση για συγκεκριμένο χρονικό διάστημα

ούτως ώστε να έχουν εξαφανιστεί οι υπόλοιποι δρόμοι από την εικόνα. Τέλος, η στήριξη των αλγορίθμων στην εύρεση ευθειών για τη λήψη αποφάσεων είναι εξίσου σημαντικό μειονέκτημα, διότι δεν εντοπίζονται πάντοτε ευθείες και σε ορισμένες περιπτώσεις και συνθήκες λαμβάνονται λανθασμένες αποφάσεις.

Περαιτέρω βελτίωση του συστήματος αποτελεί εκείνη της ενεργειακής αυτονομίας του οχήματος. Το όχημα όπως αναπτύχθηκε περιλαμβάνει μία βαριά επαναφορτιζόμενη μπαταρία τύπου Lead Acid, η οποία υπό κανονικές συνθήκες λειτουργίας και χωρίς επαναφόρτιση διαρκεί κατά μέγιστο 2 ώρες. Είναι προφανές ότι η μπαταρία αυτή πρέπει με κάποιο τρόπο να επαναφορτίζεται αυτόματα, καθώς είναι πρακτικά ασύμφορη η χειρωνακτική επαναφόρτισή της, ακόμη και στην περίπτωση χρήσης μπαταριών ίδιου ή διαφορετικού τύπου με μεγαλύτερη διάρκεια ζωής. Μια πιθανή λύση σε αυτό το πρόβλημα θα ήταν, όταν η στάθμη της μπαταρίας του οχήματος είναι χαμηλή, να ενημερώνεται ο διαχειριστής ώστε να οδηγεί ο ίδιος μέσω της εφαρμογής το όχημα σε κάποιον κοντινό διαθέσιμο σταθμό φόρτισης. Προφανώς, θα πρέπει παράλληλα να υλοποιηθεί και ο συγκεκριμένος σταθμός φόρτισης στον οποίο θα μεταβαίνει το όχημα και θα φορτίζεται αυτόματα.

Όσον αφορά στην κατασκευή του οχήματος, η συγκεκριμένη έγινε με γνώμονα το χαμηλότερο δυνατό κόστος και ότι το όχημα απλώς θα κινείται ακολουθώντας δρόμους, ενώ στην περίπτωση που χρησιμοποιηθεί σε ξενοδοχεία ή νοσοκομεία θα πρέπει παράλληλα να είναι σε θέση να κουβαλά και φορτίο. Σε αυτή την περίπτωση προφανώς ο όγκος του θα είναι διαφορετικός και η κατασκευή του επίσης, αλλά η πλοήγηση του θα πρέπει να στηρίζεται στις ίδιες αρχές.

Επιπρόσθετα, είναι επιβεβλημένο τόσο το παρόν σύστημα όσο και οποιαδήποτε περαιτέρω διαμόρφωσή του να δοκιμαστούν σε πραγματικές συνθήκες πραγματικών χώρων όπως νοσοκομεία και ξενοδοχεία, διότι όπως επισημάνθηκε αρκετές φορές το αναπτυχθέν σύστημα εφαρμόστηκε σε εργαστηριακές συνθήκες. Η εφαρμογή του συστήματος σε πραγματικές συνθήκες θα υποδείξει με σαφήνεια τα σημεία που επιδέχονται βελτίωσης στο παρόν σύστημα.

Τέλος, στη διεπαφή του διαχειριστή θα μπορούσαν να συμπεριληφθούν περισσότερα χαρακτηριστικά, ανάλογα με τις ανάγκες ή τις επιθυμίες των χρηστών του συστήματος, όπως για παράδειγμα καλύτερη χαρτογράφηση του χώρου και ρύθμιση των διαδρομών χωρίς τη χρήση πινάκων γειτνίασης.

6.3 Συμπεράσματα

Έχοντας εκτελέσει τον έλεγχο της λειτουργίας του συστήματος μπορεί να συναχθεί το συμπέρασμα ότι υπό τις δεδομένες συνθήκες η όραση και η πλοήγηση του οχήματος λειτουργεί σχεδόν απρόσκοπτα και κατά τον αναμενόμενο τρόπο. Συγκεκριμένα, ακόμα και σε συνθήκες χαμηλού φωτισμού το όχημα είναι σε θέση να φέρει σε πέρας τις ζητούμενες διαδρομές. Επιπλέον, ο διαχειριστής του συστήματος μπορεί με ασφαλή τρόπο να ελέγχει τα οχήματά του και να βλέπει ανά πάσα στιγμή την κατάσταση του συστήματος και των οχημάτων. Τέλος, ο εξυπηρετητής ταυτοποιεί απρόσκοπτα όλους τους χρήστες του και στη συνέχεια προωθεί τα μηνύματα μεταξύ αυτών.

Ωστόσο, πρέπει να ληφθεί υπόψη το γεγονός ότι το θεωρούμενο σενάριο είναι απλουστευμένο και παρουσιάζει πολλούς περιορισμούς, με κύριο αυτόν των αλγορίθμων όρασης και πλοήγησης. Για τον λόγο αυτόν κρίνεται αναγκαία η περαιτέρω ανάπτυξη του παρόντος συστήματος παρακολούθησης, με γνώμονα τη διόρθωση και τη γενίκευση των αλγορίθμων όρασης και πλοήγησης των οχημάτων ώστε να μην δουλεύουν αποκλειστικά με την αναγνώριση ευθειών. Κλείνοντας, το συμπέρασμα το οποίο μπορεί να εξαχθεί είναι πως το αναπτυχθέν σύστημα της παρούσας μελέτης αποτελεί μία σημαντική βάση για την ανάπτυξη πολυπλοκότερων συστημάτων παρακολούθησης και αυτόνομης οδήγησης οχημάτων εξ' ολοκλήρου με χρήση τεχνικών ανάλυσης εικόνας.

Βιβλιογραφία

[1] Colleen C., «Introducing the Intel® RealSense™ R200 Camera (world facing),» [Ηλεκτρονικό]. Available: https://software.intel.com/en-us/articles/realsense-r200-camera.

[2] Wikipedia, «Robot Operating System,» [Ηλεκτρονικό]. Available: https://en.m.wikipedia.org/wiki/Robot_Operating_System.

[3] H. C. R. J. T. L. YoonSeok Pyo, «ROS Robot Programming,» 22 December 2017. [Ηλεκτρονικό]. Available: https://community.robotsource.org/t/download-the-ros-robot-programming-book-for-free/51.

[4] ROS Wiki, «ROS Concepts,» [Ηλεκτρονικό]. Available: http://wiki.ros.org/ROS/Concepts.

[5] ROS Wiki, «realsense_camera,» [Ηλεκτρονικό]. Available: http://wiki.ros.org/realsense_camera.

[6] ROS Wiki, «Ubuntu install of ROS Kinetic,» [Ηλεκτρονικό]. Available: http://wiki.ros.org/kinetic/Installation/Ubuntu.

[7] jadintredup, «Setting up the UP Board,» [Ηλεκτρονικό]. Available: http://www.daslhub.org/unlv/wiki/doku.php?id=up_board_setup.

[8] «Getting started with the Intel RealSense Robotics Development Kit (RDK),» [Ηλεκτρονικό]. Available: https://idorobotics.com/2016/10/31/getting-started-with-the-intel-realsense-robotics-development-kit-rdk/.

[9] ROS Wiki, «Arduino IDE Setup,» [Ηλεκτρονικό]. Available: https://wiki.ros.org/rosserial_arduino/Tutorials/Arduino%20IDE%20Setup.

[10] Wikipedia, «Arduino,» [Ηλεκτρονικό]. Available: https://el.wikipedia.org/wiki/Arduino.