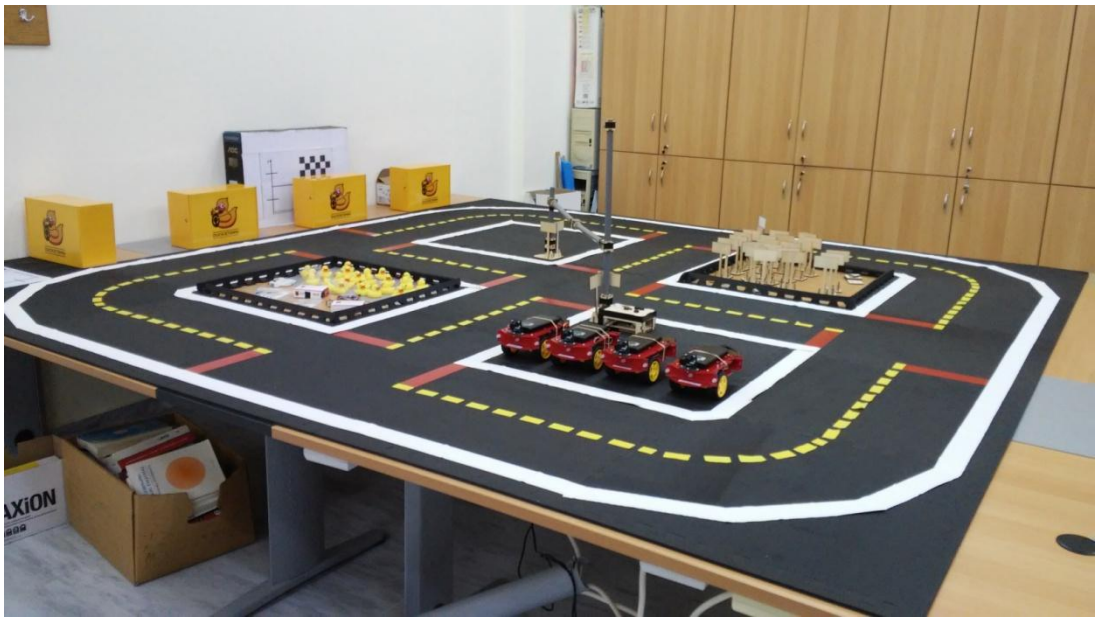
ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

Τμήμα Μηχανικών Πληροφορικής, Υπολογιστών και Τηλεπικοινωνιών

Πρόγραμμα Μεταπτυχιακών Σπουδών στη Ρομποτική

Πτυχιακή Εργασία

Duckietown



Επιβλέπων καθηγητής: Δρ. Βολογιαννίδης Σταύρος

Φοιτητής: Πετρακίδης Αθανάσιος (Α.Μ. 1)

Σέρρες, Δεκέμβριος 2020

Ευχαριστίες

Ο πρώτος που νιώθω την ανάγκη να ευχαριστήσω είναι ο φίλος και κουμπάρος μου Γρηγόρης που στάθηκε η αιτία να εμπλακώ σε αυτό το μεταπτυχιακό πρόγραμμα σπουδών στη Ρομποτική. Εάν δεν ήταν αυτός να με ενθαρρύνει πολύ φοβάμαι ότι εγώ δεν θα άνοιγα από μόνος μου την πόρτα για να περιηγηθώ σε αυτό το συναρπαστικό και μαγικό κόσμο της ρομποτικής φοβούμενος το τίμημα του κόπου που θα έπρεπε να καταβάλω προκειμένου να ανταποκριθώ στις απαιτήσεις των μαθημάτων. Τον ευχαριστώ και για την συμπαράσταση του καθ' όλη τη διάρκεια του προγράμματος , για την ανεκτικότητα που επέδειξε απέναντι στην γκρίνια μου όταν έβρισκα τα "δύσκολα" καθώς επίσης και για τα καφεδάκια που με κερνούσε στο κυλικείο της σχολής πριν το μάθημα.

Το επόμενο ευχαριστώ είναι προς τους καθηγητές του προγράμματος που με τίμησαν με την εμπιστοσύνη τους, τον Κ. Καλόμοιρο Ιωάννη και τον Κ. Βολογιαννίδη Σταύρο που είναι και ο επιβλέπων καθηγητής της πτυχιακής μου εργασίας διότι μου ανέθεσαν αυτό το εξαιρετικά ενδιαφέρον project και με άφησαν εν λευκώ να διαχειριστώ όλο το υλικό του πακέτου Duckietown δίνοντας μου άπλετο χρόνο και χώρο. Τους ευχαριστώ και για την άμεση ανταπόκριση και βοήθεια τους σε κάθε μου αίτημα. Στο σημείο αυτό θα ήταν παράληψη αν δεν ευχαριστούσα και όλους τους υπόλοιπους διδάσκοντες καθηγητές για τις γνώσεις που μπόρεσα να αποκομίσω από αυτούς οφείλοντας να αναγνωρίσω το γεγονός ότι στο χρονικό διάστημα που είχαν στη διάθεση τους προσπάθησαν για το καλύτερο.

Τέλος ένα μεγάλο ευχαριστώ στην γυναίκα μου Γιώτα που και αυτή με τη σειρά της με παρότρυνε να εμπλακώ σε αυτό το πρόγραμμα και με στήριξε ουσιαστικά δείχνοντας υπομονή και ανεκτικότητα και κατά τη διάρκεια παρακολούθησης του προγράμματος αλλά ειδικότερα κατά το διάστημα εκπόνησης της πτυχιακής μου εργασίας. Όσο για την μικρή μου κόρη, Μαριάννα της υπόσχομαι ότι θα αναπληρώσουμε τα παιχνίδια που χάσαμε και με το παραπάνω.

Περίληψη

Η παρούσα εργασία πραγματεύεται την ανάπτυξη μιας πλατφόρμας που γεννήθηκε το 2016 στο Τεχνολογικό Ινστιτούτο της Μασαχουσέτης (MIT), της πλατφόρμας του Duckietown. Η πλατφόρμα αυτή δημιουργήθηκε με σκοπό την εκμάθηση των βασικών αρχών λειτουργίας των αυτόνομων ρομποτικών οχημάτων σε μαθητές - σπουδαστές αλλά και την περαιτέρω διερεύνηση αυτών των αρχών. Θα περιγράψουμε αναλυτικά τους στόχους που υπηρετεί η συγκεκριμένη πλατφόρμα, τα επιστημονικά πεδία τα οποία ακουμπά και αφού αναλύσουμε το θεωρητικό υπόβαθρο στο οποίο βασίζεται θα προχωρήσουμε στην περιγραφή τεχνικών λεπτομερειών που αφορούν την κατασκευή και τη λειτουργία της. Θα περιγράψουμε το εύρος των συμπεριφορών που είναι δυνατόν να αναπτύξουν τα ρομποτικά οχήματα (Duckiebots) μέσα στην πόλη καθώς και όλες τις δυνατές επεκτάσεις αυτής της πλατφόρμας. Τέλος θα εξετάσουμε το αν και κατά πόσο η συγκεκριμένη πλατφόρμα μπορεί να διεισδύσει στην πρωτοβάθμια και στην δευτεροβάθμια εκπαίδευση ως ένα ελκυστικό μέσο για την εκμάθηση κάποιων βασικών αρχών ρομποτικής στα πλαίσια σχετικών μαθημάτων και αν εναρμονίζεται με τους βασικούς σκοπούς του αναλυτικού προγράμματος σπουδών σε κάθε εκπαιδευτική βαθμίδα και τάξη.

Abstract

This thesis deals with the development of a platform born in 2016 at the Massachusetts Institute of Technology (MIT), the Duckietown platform. This platform was created in order to learn the basic principles of operation of autonomous robotic vehicles to students but also to further explore these principles. We will describe in detail the objectives of the specific platform, the scientific fields on which it rests and after analyzing the theoretical background on which it is based, we will proceed to the description of technical details regarding its construction and operation. We will describe the range of behaviors that robotic vehicles (Duckiebots) can develop in the city as well as all possible extensions of this platform. Finally, we will examine whether and to what extent this platform can penetrate primary and secondary education as an attractive means of learning some basic robotics principles in relevant courses and whether it aligns with the main objectives of the curriculum of each educational level and class.

Πίνακας περιεχομένων

Ευρετήριο Εικόνων - Πινάκων

Εισαγωγή

Πάντα με γοήτευαν οι ταινίες επιστημονικής φαντασίας. Ταινίες με εξωγήινους, με ανθρωπόμορφα ρομπότ, με έξυπνες μηχανές. Ταινίες όπως το Avatar, ο εξολοθρευτής, το Interstellar, το Matrix, το Star Wars. Αλλά και σειρές επιστημονικής φαντασίας όπως η Οδύσσεια του Διαστήματος ή αργότερα το Star Trek. Ίσως γιατί πίστευα και πιστεύω πως η φαντασία είναι η ανώτατη νοητική λειτουργία του ανθρώπου. Η φαντασία και όχι η γνώση είναι η κινητήρια δύναμη της εξέλιξης. Το να φανταστούμε, να οραματιστούμε κάτι το οποίο δεν υπάρχει στο παρόν γεννά την ιδέα, το καινούργιο. Η σύλληψη της ιδέας είναι η αρχή. Μπορεί να μην την υλοποιήσουμε εμείς, μπορεί να μην υλοποιηθεί καν στην εποχή που ζούμε. Δεν έχει σημασία. Η ιδέα είναι ο σπόρος και κάποτε όταν οι συνθήκες το επιτρέψουν θα φυτρώσει.

Η αγαπημένη μου σειρά σαν παιδί (δεκαετία του 80...) ήταν ο Ιππότης της ασφάλτου. Ένα μαύρο σπορ, γρήγορο αυτοκίνητο (Pontiac Firebird Trans Am) που μιλούσε μέσα από μία οθόνη στον οδηγό του, μπορούσε να κινηθεί μόνο του, να ανιχνεύσει αντικείμενα στο χώρο, να προβάλει εικόνες από το εξωτερικό περιβάλλον μέσω μιας οθόνης κατασκευασμένο για να υπηρετεί το καλό και το δίκαιο. Στην εποχή μας πολλά από αυτά που βλέπαμε με θαυμασμό και δέος στις ταινίες επιστημονικής φαντασίας διαπιστώνουμε να γίνονται πραγματικότητα σιγά - σιγά...

Στις μέρες μας φαίνεται ότι ανατέλλει μια καινούργια εποχή, αυτή της δημιουργίας αυτόνομων, αυτοκινούμενων κατασκευών που επιτελούν κάποιο έργο ή διεκπεραιώνουν κάποιες λειτουργίες. Μιλάμε για κατασκευές που πρακτικά συνδυάζουν και ενοποιούν επιστήμες όπως η μηχανολογία, η ηλεκτρονική - ηλεκτρολογία και η πληροφορική και που συχνά περιγράφονται ως ρομπότ. Το μεγάλο στοίχημα των ημερών μας είναι ο τρόπος λειτουργίας των παραπάνω ρομποτικών "κατασκευών" να μη βασίζεται σε έναν αλγόριθμο που έχει προνοήσει για όλες τις δυνατές περιπτώσεις που θα συναντήσει σε ένα ελεγχόμενο περιβάλλον αλλά αντίθετα να μπορεί η ίδια η μηχανή να κινηθεί και να αυτενεργήσει σε ένα άγνωστο εξωτερικό περιβάλλον όπως περίπου ο άνθρωπος. Έτσι μπήκαν στο τραπέζι έννοιες όπως μηχανική μάθηση, ρομποτική όραση, ενσωματωμένα συστήματα πραγματικού χρόνου και συστήματα υψηλών επιδόσεων προκειμένου για να είναι σε θέση να "επεξεργαστεί", να "εκτιμήσει" ή να "διαμοιράσει" τα δεδομένα που συλλέγει διαμέσου των αισθητήρων που διαθέτει και τελικά να "αποφασίζει" για τις δικές του ενέργειες λαμβάνοντας υπόψη και τον σκοπό που έχει να επιτελέσει.

Κεφάλαιο 1

Σκοπός - Στόχοι

Mission

" Our mission is to make the world excited about the beauty, the fun, the importance, and the challenges of robotics and AI, through learning experiences that are tangible, accessible, and inclusive".

1.1 Σκοπός

Η βασική αποστολή του Duckietown όπως αυτή περιγράφεται στην δεύτερη σελίδα του εισαγωγικού εγχειριδίου [1] είναι να καταστήσει τον κόσμο ενθουσιασμένο με την ομορφιά, τη διασκέδαση και τις προκλήσεις της ρομποτικής και της τεχνητής νοημοσύνης μέσα από εμπειρικές μαθησιακές διαδικασίες που να είναι απτές, προσβάσιμες σε όλους και περιεκτικές.

Ο παραπάνω σκοπός επιτυγχάνεται σχεδιάζοντας ελεύθερα διαθέσιμες πλατφόρμες ρομποτικής που περιλαμβάνουν τόσο το υλικό όσο και το λογισμικό καθώς και προγράμματα μαθημάτων προσαρμοσμένα σε όλα τα επίπεδα της εκπαίδευσης και προωθώντας αυτές τις πλατφόρμες για χρήση σε ολόκληρο τον κόσμο.

Μέσα από τη χρήση της πλατφόρμας του Duckietown μπορεί να ανακαλύψει κάποιος τον κόσμο της τεχνητής νοημοσύνης και της ρομποτικής μέσα από δεξιότητες και διαδικασίες που έχουν ως σκοπό τη δημιουργία τεχνητών "πλασμάτων" που σκέπτονται και ενεργούν αυτόνομα όπως περίπου οι άνθρωποι. Κατ' επέκταση αντιλαμβάνεται τη μεγάλη σημασία της ρομποτικής που γεννά μια νέα εποχή και που πρόκειται να αλλάξει τον κόσμο μας και να καταλάβει τις δυνατότητες που αυτή μας παρέχει, να συνειδητοποιήσει το τι έχουμε καταφέρει μέχρι τώρα και να οραματιστεί τι απομένει να κάνουμε στο μέλλον.

Το συγκεκριμένο project δίνει μεγάλη έμφαση στο μοντέλο της εμπειρικής μάθησης καθώς περιλαμβάνει την κατασκευή από την αρχή της όλης πλατφόρμας μέχρι το τέλος όπου βλέπουμε τα αυτοκινούμενα οχήματα να κινούνται μέσα στην πόλη που έχουμε κατασκευάσει. Η κεντρική ιδέα δηλαδή που βασίζεται η συγκεκριμένη πλατφόρμα είναι ότι για να μάθεις ρομποτική πρέπει να είσαι σε θέση να αγγίξεις και να κατασκευάσεις ένα ρομπότ. Διαθέτοντας τα χαρακτηριστικά της ελεύθερης προσβασιμότητας και στηριζόμενη σε ανοικτού τύπου λογισμικό η πλατφόρμα του Duckietown προσφέρεται για ακαδημαϊκή μάθηση και διερεύνηση της τεχνητής νοημοσύνης και των οριζόντων που αυτή ανοίγει.

1.2 Στόχοι

Αν θελήσουμε να αναλύσουμε τα οφέλη του βασικού σκοπού του Duckietown project που είναι η παροχή μιας εκπαιδευτικής πλατφόρμας χαμηλού κόστους για τη κατασκευή και λειτουργία αυτόνομων ρομποτικών οχημάτων θα ανακαλύψουμε τους πολλαπλούς στόχους του συγκεκριμένου project τόσο για τους εκπαιδευτές όσο και για τους "αυτοδίδακτους" μαθητές. Καταρχάς η αυτόνομη περιήγηση από μόνη της είναι ένα μεγάλο κεφάλαιο. Η επίτευξη της αυτόνομης κίνησης ακόμα και σε ένα ειδικά διαμορφωμένο περιβάλλον όπως αυτό του Duckietown προϋποθέτει την εξοικείωση και την συνδυαστική εφαρμογή πολλών γνωστικών αντικειμένων πάνω στα οποία βασίζεται αυτή η τεχνίτη νοημοσύνη που "εμφυτεύεται" στα ρομποτικά οχήματα και τα επιτρέπει έχουν κάποιου είδους αυτενέργειας. Συνεπώς όποιος ασχοληθεί με το συγκεκριμένο project είναι αναπόφευκτη η γνωριμία και η ενασχόληση του με τις σύγχρονες αρχιτεκτονικές των αυτόνομων συστημάτων και των εργαλείων που τις υλοποιούν. Το παραπάνω πλαίσιο οριοθετεί και τους στόχους του project οι οποίοι με μια συνοπτική αναφορά είναι οι εξής:

- ✓ Η γνωριμία με **μοντέλα κινηματικής** διαφορικής οδήγησης, με την έννοια της οδομετρίας καθώς και με τη βαθμονόμηση των τροχών.
- ✓ Η εξοικείωση με βασικά αντικείμενα της **ρομποτικής όρασης** όπως μετασχηματισμοί, φίλτρα, ανίχνευση γραμμών, εξαγωγή χαρακτηριστικών, αναγνώριση τοποθεσίας, βαθμονόμηση κάμερας.
- ✓ Η κατανόηση μεθόδων **σχεδιασμού κίνησης** (motion planning) του **ελέγχου** ανατροφοδότησης καθώς και του προγνωστικού ελέγχου.
- ✓ Η ανακάλυψη των μεθοδολογιών **εκτίμησης** της κατάστασης - **πόζας** του οχήματος στο χώρο όπως το φίλτρο Bayes, το εκτεταμένο φίλτρο Kalman καθώς και άλλα μη γραμμικά φίλτρα.
- ✓ Η κατανόηση των **τεχνικών χαρτογράφησης** (mapping) του χώρου βασισμένες στις προσλαμβάνουσες εικόνες από το ρομποτικό όχημα.
- ✓ Η χρήση λογισμικών ανοικτού κώδικα αυτόνομων ρομποτικών συστημάτων όπως το **ROS**, αποθετηρίων κώδικα όπως το **GitHub**, πλατφόρμες ανάπτυξης και εκτέλεσης εφαρμογών όπως το **Docker**, γλωσσών προγραμματισμού όπως η **Python**.

Όλα τα παραπάνω αποτελούν τους βασικούς στόχους που θα πρέπει να καλύψει όποιος εμπλακεί με τη πλατφόρμα του Duckietown. Πέρα από αυτούς όμως η πλατφόρμα μας δίνει τη δυνατότητα να επεκταθούμε, να πειραματιστούμε, να διερευνήσουμε και να εμβαθύνουμε σε πεδία που άπτονται της αντίληψης του αυτοκινούμενου ρομπότ μέσω αλγορίθμων βαθιάς μάθησης είτε σε πεδία που αφορούν τον ταυτόχρονο εντοπισμό του στο χώρο και την χαρτογράφηση του είτε τη διαχείριση της κυκλοφορίας των οχημάτων στην πόλη. Ενδεικτικά αναφέρουμε κάποιους επιπλέον προηγμένους στόχους της πλατφόρμας όπως :

- ✓ Ανίχνευση και εντοπισμός αντικειμένων (Object detection and tracking)
- ✓ Αναγνώριση κειμένου (text recognition)
- ✓ Σημασιολογική κατάτμηση (Semantic segmentation)

- ✓ Δημιουργία γραφήματος για τον ταυτόχρονο εντοπισμό και χαρτογράφηση του χώρου (Graph SLAM)
- ✓ Οπτικά αδρανειακά συστήματα πλοήγησης (Visual Intertial Navigation Systems - VINS)
- ✓ Συντονισμός πολλαπλών οχημάτων - Βελτιστοποίηση της κυκλοφορίας

Βέβαια η ενασχόληση κάποιου με τη συγκεκριμένη πλατφόρμα διευκολύνεται αν προϋπάρχει μία εξοικείωση πάνω σε θέματα που αφορούν τη δικτύωση συσκευών, τον προγραμματισμό καθώς και το Single Board Computer, Raspberry Pi. Έτσι θα καλό θα ήταν να γνωρίζει και να κατανοεί τον τρόπο επικοινωνίας των συσκευών μέσω των πρωτοκόλλων ενσύρματης και ασύρματης δικτύωσης, να διαθέτει κάποια εμπειρία στον προγραμματισμό ειδικά με την γλώσσα προγραμματισμού Python και τέλος να είναι εξοικειωμένος με τον υπολογιστή πλακέτας Raspberry Pi και τις δυνατότητες του. Βέβαια σε καμιά περίπτωση δεν είναι απαγορευτικό να ασχοληθεί κάποιος με τη συγκεκριμένη πλατφόρμα χωρίς τα παραπάνω εφόδια. Απλά θα χρειαστεί περισσότερο χρόνο για να γνωρίσει τα αντικείμενα αυτά ώστε να είναι σε θέση να τα χρησιμοποιήσει έπειτα στην υλοποίηση του project.

Η πλατφόρμα Duckietown είναι ένα "class in a box" που παρέχει σε όσους ασχολούνται με αυτή μία σύγχρονη εμπειρία μάθησης όχι μόνο πάνω στη ρομποτική αλλά ουσιαστικά σε όλους τους τομείς που αυτή στηρίζεται. Έτσι μέσω της ενασχόλησης με τη συγκεκριμένη πλατφόρμα ανακαλύπτει κανείς το μαγικό κόσμο της πληροφορικής μέσα από λογισμικά ανοικτού κώδικα, αποθετήρια, γλώσσες προγραμματισμού, επικοινωνία συσκευών μέσω δικτύων, υπολογιστές πλακέτας για να επεκταθεί μετά σε μοντέλα κινηματικής, βασικές αρχές της ρομποτικής όρασης και μεθόδους σχεδιασμού κίνησης γνωρίζοντας τελικά στην πράξη τον όρο τεχνητή νοημοσύνη. Και φυσικά δύναται να διερευνήσει αν το επιθυμεί προηγμένα θέματα όπως ο εντοπισμός αντικειμένων, η χαρτογράφηση χώρου ή συστήματα πλοήγησης περιδιαβαίνοντας στα μονοπάτια της έρευνας.

Κεφάλαιο 2

Αρχιτεκτονική της αυτόνομης πλοήγησης

Με τον όρο πλοήγηση περιγράφεται η ασφαλής μετάβαση ενός υποκειμένου ή αντικειμένου που διαθέτει την ικανότητα της κίνησης από ένα σημείο σε ένα άλλο. Ο όρος προέρχεται από τη ναυτική ορολογία και αναφέρετε στη διαδικασία της ασφαλούς μετακίνησης των πλοίων από το λιμάνι-αφετηρία στο λιμάνι-προορισμό διαμέσου της θάλασσας. Κατ' επέκταση χρησιμοποιήθηκε για να περιγράψει την οδήγηση ή την καθοδήγηση οποιουδήποτε οχήματος προκειμένου αυτό να φτάσει με ασφάλεια στον προορισμό του.

Ο αυτόνομος κυριολεκτικά είναι αυτός που καθορίζει μόνος του τους νόμους που τον διέπουν ενώ μεταφορικά χρησιμοποιούμε τη λέξη της αυτονομίας ως συνώνυμης της ανεξαρτησίας. Κατ' επέκταση αν μιλάμε για άτομο θα μπορούσαμε να πούμε ότι αυτόνομος, είναι ο αυτόβουλος αυτός δηλαδή που διαθέτει τη δική του ανεξάρτητη θέληση και βούληση και βασίζεται σ' αυτήν για τη λήψη αποφάσεων.

Ο όρος αυτόνομη πλοήγηση περιγράφει μια διαδικασία μετακίνησης από ένα σημείο εκκίνησης σε ένα σημείο τερματισμού κατά την οποία το υποκείμενο ή το αντικείμενο-όχημα τη διεκπεραιώνει μόνο του χωρίς τη καθοδήγηση ή τη συνδρομή κάποιου εξωτερικού παράγοντα. Κλασικά παραδείγματα αυτόνομης πλοήγησης ο τρόπος που μετακινείται ο άνθρωπος από το σπίτι στο χώρο εργασίας ή στο super market ή στο σχολείο ή σε μία άλλη πόλη χωρίς τη χρήση συστημάτων εντοπισμού γεωγραφικής θέσης και χάραξης διαδρομής. Και αν ο άνθρωπος και τα ζώα το κάνουν με επιτυχία, πως θα μπορούσε να κάνει το ίδιο και μία μηχανή; Ένα αυτοκίνητο για παράδειγμα ή ένα ρομπότ;

Στην εποχή μας ακούμε ολοένα και περισσότερο για αυτοκίνητα που διαθέτουν κάποιου είδους αυτονομία. Εταιρίες όπως η Tesla, η Google με το Waymo, η General Motors και η BMW στοχεύουν ή έχουν καταφέρει να κατασκευάσουν αυτοκίνητα που εκτελούν ορισμένες λειτουργίες αυτόνομα, χωρίς την παρέμβαση του οδηγού. Στο σημείο αυτό κρίνουμε σκόπιμο να αναφέρουμε τα επίπεδα της Αυτονομίας όπως αυτά έχουν οριστεί από την SAE (Society of Automotive Engineers) και που περιγράφουν έξι διαφορετικές βαθμίδες αυτονομίας [2].

Εικόνα 1: Τα έξι επίπεδα της αυτονομίας από την SAE

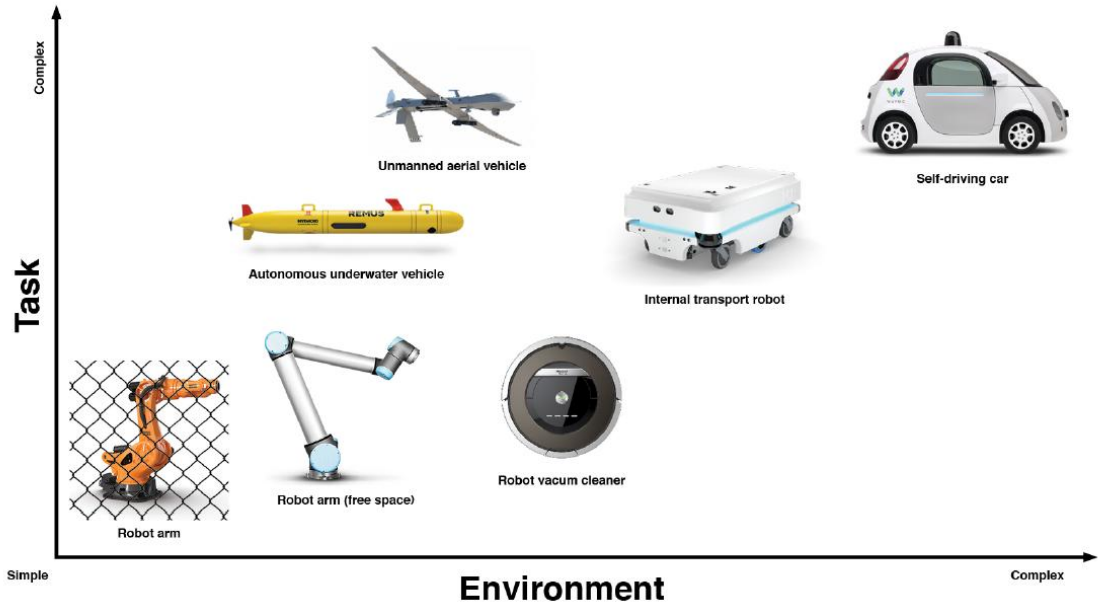
- Επίπεδο 0 : Δεν διαθέτει αυτονομία αλλά περιλαμβάνει την προειδοποίηση εμπρός σύγκρουσης, προειδοποίηση λωρίδας αναχώρησης και παρακολούθηση τυφλού

σημείου. Αυτά τα συστήματα βοηθούν τον οδηγό ο οποίος όμως εξακολουθεί να εκτελεί όλα τα οδηγικά του καθήκοντα.

- Επίπεδο 1 : Στο επίπεδο αυτό συναντούμε τον προσωρινό έλεγχο της κίνησης του αυτοκινήτου όπως το προσαρμοστικό(adaptive) cruise control με βάση το οποίο το αυτοκίνητο διατηρεί μια συγκεκριμένη ταχύτητα αλλά αυτόματα επιβραδύνει όταν πλησιάσει κοντά σε ένα άλλο αυτοκίνητο και το σύστημα παρακολούθησης λωρίδας. Τα χαρακτηριστικά αυτά διευκολύνουν την οδήγηση ειδικά σε λιγότερο απαιτητικά περιβάλλοντα όπως ένας αυτοκινητόδρομος.
- Επίπεδο 2 : Το δεύτερο επίπεδο παρέχει μερική αυτοματοποίηση και απαιτεί δύο ή παραπάνω λειτουργίες του πρώτου επιπέδου να εκτελούνται ταυτόχρονα. Το αυτοκίνητο μπορεί πλέον να επιταχύνει, να στρίψει και να φρενάρει αν για παράδειγμα είναι ενεργό το adaptive cruise control και το σύστημα που τα κρατάει εντός της λωρίδας του δρόμου. Έτσι περιστασιακά μπορεί να οδηγείται από μόνο του όμως ο οδηγός πρέπει να είναι πάντα σε ετοιμότητα για να αναλάβει τον έλεγχο.
- Επίπεδο 3 : Αυτό το επίπεδο διαθέτει υπό όρους αυτοματοποίηση. Το αυτοκίνητο είναι σε θέση να διαχειριστεί τις περισσότερες πτυχές της οδήγησης υπό τις "σωστές" συνθήκες, έτσι ώστε ο οδηγός να μην απαιτείται να παρακολουθεί το περιβάλλον. Το αυτοκίνητο πρέπει να είναι σε θέση να ενημερώνει τον οδηγό πότε πρέπει να αναλάβει τον έλεγχο του αυτοκινήτου.
- Επίπεδο 4 : Το τέταρτο επίπεδο διαθέτει υψηλό αυτοματισμό καθώς απαιτεί το αυτοκίνητο να λειτουργεί σε συγκεκριμένες συνθήκες, όπως ο τύπος δρόμου, ο καιρός και η γεωγραφική περιοχή, χωρίς είσοδο από τον οδηγό. Το αυτοκίνητο μπορεί να οδηγήσει, να επιταχύνει και να φρενάρει, ενώ παρακολουθεί επίσης το περιβάλλον, συμπεριλαμβανομένης της αλλαγής λωρίδας, στροφών και σημάτων σήμανσης σε άλλα οχήματα. Ο οδηγός μπορεί να έχει την επιλογή να ελέγχει το όχημα.
- Επίπεδο 5 : Το αυτό επίπεδο παρέχει πλήρης αυτοματισμό, γεγονός που συνεπάγεται ότι το αυτοκίνητο μπορεί να εκτελεί όλες τις λειτουργίες οδήγησης υπό όλες τις συνθήκες. Η μόνη ανθρώπινη αλληλεπίδραση είναι να εισάγει τον προορισμό. Εάν επιτευχθεί αυτό το επίπεδο αυτονομίας, τα χειριστήρια του οδηγού δηλαδή πεντάλ, φρένα και τιμόνι είναι περιττά.

Τα παραπάνω επίπεδα αυτονομίας περιγράφουν και τα επίπεδα αλληλεπίδρασης οδηγού και αυτοκινήτου καθώς και το πως κατανέμονται οι εργασίες μεταξύ τους προκειμένου να φέρουν εις πέρας την αποστολή τους που είναι η ασφαλής μετακίνηση από το σημείο Α στο σημείο Β. Στο σημείο αυτό να τονίσουμε τα παραπάνω επίπεδα αυτονομίας μπορούν να γενικευτούν και για άλλες ρομποτικές κατασκευές.

Ο βαθμός πολυπλοκότητας της αυτονομίας είναι συνάρτηση δύο βασικών παραγόντων. Του πλήθους των εργασιών με τις οποίες είναι επιφορτισμένη μια ρομποτική κατασκευή και του περιβάλλοντος μέσα στο οποίο θα κληθεί να στις διεκπεραιώσει. Αυτό απεικονίζει και το παρακάτω διάγραμμά (εικόνα 2)



Εικόνα 2: Διάγραμμα πολυπλοκότητας ρομποτικών κατασκευών ως προς το περιβάλλον και τις εργασίες που επιτελούν

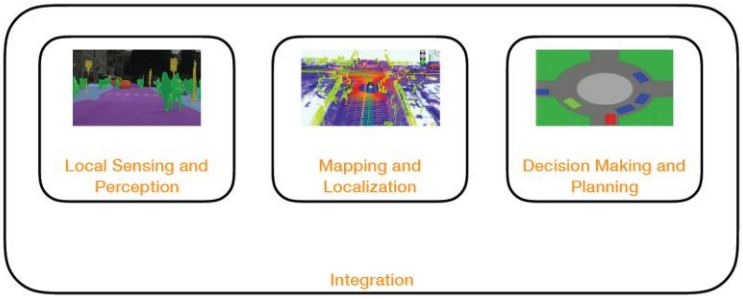
Έτσι για παράδειγμα ένας ρομποτικός βραχίονας που βρίσκεται σε ένα μεταλλικό κλωβό και είναι επιφορτισμένος να μετακινεί αντικείμενα από το σημείο Α στο σημείο Β έχει σαφώς μειωμένη πολυπλοκότητα από έναν βραχίονα που βρίσκεται σε ένα δυναμικό περιβάλλον και πρέπει να ανιχνεύει και τις όποιες αλλαγές λαμβάνουν χώρα σε αυτό πριν ενεργήσει [3].

Εμείς στο παρών project θα δημιουργήσουμε ένα κατάλληλα διαμορφωμένο περιβάλλον (Duckietown) μέσα στο οποίο θα κινούνται ρομποτικά οχήματα (Duckiebots). Το περιβάλλον θα είναι η πόλη η οποία όπως θα περιγράψουμε παρακάτω θα διαθέτει δρόμους, πινακίδες σήμανσης, διασταυρώσεις, φωτεινούς σηματοδότες, κτλ θα είναι δηλαδή μια προσομοίωση μιας πραγματικής πόλης. Στόχος - στοίχημα είναι σε αυτή την πόλη να καταφέρουμε να κατασκευάσουμε οχήματα τα οποία θα κινούνται αυτόνομα σε ένα δυναμικό περιβάλλον αγγίζοντας αυτονομία επιπέδου τέσσερα ή και πέντε. Το εγχείρημα μας γίνεται ακόμα πιο δύσκολο αν αναλογιστούμε ότι ενώ οι διάφορες κατασκευάστριες εταιρείες προκειμένου τα αυτοκίνητα που παράγουν να επιτύχουν κάποιο από τα παραπάνω επίπεδα αυτονομίας τα εξοπλίζουν με πληθώρα αισθητηρίων οργάνων προκειμένου να συλλέγουν πληροφορίες από το εξωτερικό τους περιβάλλον όπως για παράδειγμα αισθητήρες μέτρησης απόστασης (Radar, sonar, LIDAR), GPS, κάμερες, κτλ εμείς θα έχουμε στη διάθεση μας έναν μόνο αισθητήρα, την κάμερα. Οι λειτουργίες των Duckiebots και γενικότερα η συμπεριφορά τους δηλαδή θα βασίζεται μόνο στις εικόνες που θα συλλαμβάνει η κάμερα που διαθέτουν. Για να δούμε πως μπορούμε να επιτύχουμε κάτι τέτοιο θα πρέπει να παραδειγματιστούμε και μιμηθούμε συμπεριφορές από οντότητες που έχουν αυτή την ικανότητα που μπορούν δηλαδή να κινούνται αυτόνομα μέσα σε ένα περιβάλλον βασιζόμενοι κατά κύριο λόγο στις προσλαμβάνουσες εικόνες. Και φυσικά θα επιλέξουμε ως παράδειγμα προς μελέτη τη τελειότερη αλλά και πολυπλοκότερη "μηχανή" που έχει κατασκευαστεί ποτέ δηλαδή τον άνθρωπο.

Αρχικά θα πρέπει να αναλύσουμε και να αποκωδικοποιήσουμε τον τρόπο με τον οποίο λειτουργεί ο άνθρωπος και καταφέρνει σχεδόν πάντα με επιτυχία να φτάνει στον προορισμό του μετακινούμενος μέσα σε ένα ασταθές και μεταβαλλόμενο εξωτερικό περιβάλλον. Τι πρέπει να γνωρίζουμε για να μεταβούμε από το ένα σημείο στο άλλο είτε πεζοί είτε με κάποιο εποχούμενο μέσο; Πως προδιαγράφουμε στο μυαλό μας μια διαδρομή που εμείς θεωρούμε βέλτιστη; Και πως αν λόγω εμποδίων δεν είναι δυνατή η τήρηση αυτής της διαδρομής πως επιλέγουμε μία εναλλακτική; Θα ήταν το ίδιο εύκολο για μας αν ξαφνικά βρισκόμασταν σε ένα άγνωστο σημείο μίας ξένης πόλης και μας έλεγε κάποιος να πάμε στο Δημαρχείο; Μάλλον όχι... Γιατί όμως; Γιατί αρχικά δεν θα γνωρίζαμε ούτε τον χάρτη της πόλης, ούτε τη θέση μας μέσα σ' αυτό το χάρτη, ούτε τη θέση του Δημαρχείου. Και φυσικά αφού θα είχαμε άγνοια των παραπάνω στοιχείων θα αδυνατούσαμε να χαράξουμε μία διαδρομή βασιζόμενοι στο υπάρχον οδικό δίκτυο που θα μας οδηγούσε με ασφάλεια στο Δημαρχείο.

Από το παραπάνω απλό παράδειγμα προκύπτει εύκολα το συμπέρασμα ότι ο σχεδιασμός μιας διαδρομής προϋποθέτει τη γνώση του χάρτη της τοποθεσίας στην οποία θα κινηθούμε και τον εντοπισμό της θέσης μας σε αυτόν. Σε μια πόλη που κατοικούμε χρόνια το να πάμε από το σπίτι μας στο Δημαρχείο φαντάζει και είναι εύκολη υπόθεση γιατί όλα τα παραπάνω είναι γνωστά όπως και η διαδρομή που θα ακολουθήσουμε. Τα ίδια ακριβώς στοιχεία θα πρέπει να έχει ή να αποκτήσει και μία μηχανή για κινηθεί αυτόνομα. Και πως αποκτήσαμε αυτές τις γνώσεις; Είδαμε κάποιο χάρτη και τον μάθαμε "απ' έξω" ; Ίσως αλλά μάλλον όχι. Και αν τώρα είναι πολύ εύκολο να αντλήσεις πληροφορίες μέσα από ψηφιακές διαδικτυακές εφαρμογές χαρτογράφησης παλαιότερα δεν ήταν γιατί πολύ απλά δεν υπήρχαν. Πως λοιπόν αποκτούσαμε γνώση του χάρτη μιας πόλης; Περπατώντας κυρίως ή κάνοντας βόλτες με το αυτοκίνητο ή το μηχανάκι αν είχαμε... Κάνοντας βόλτες λοιπόν βλέπαμε και συγκρατούσαμε ονόματα οδών, χαρακτηριστικά κτήρια, πλατείες, μνημεία, επωνυμίες μαγαζιών. Το μυαλό μας αποθήκευε όλες αυτές τις εικόνες και σιγά - σιγά έχτιζε και διαμόρφωνε το χάρτη της πόλης στην οποία ήμασταν. Μέσα από εμπειρίες και αναπαραστάσεις. Το ερώτημα που προκύπτει είναι: Θα μπορούσε να το κάνει αυτό και μία μηχανή; Και αν ναι, με ποιον τρόπο; Σ' αυτά ακριβώς τα ερωτήματα φιλοδοξεί να απαντήσει και να διερευνήσει η πλατφόρμα Duckietown.

Αν προσπαθήσουμε να βάλουμε σε μία σειρά τις παραπάνω σκέψεις καταλήγουμε στους πυλώνες πάνω στους οποίους στηρίζεται η αυτόνομη πλοήγηση [4] ή οδήγηση.



Εικόνα 3 : Πυλώνες αυτόνομης πλοήγησης

2. 1 Αισθητήριο όργανο και αντίληψη

Καταρχάς χρειαζόμαστε τουλάχιστον ένα αισθητήρα με τον οποίο θα είμαστε σε θέση να αντιλαμβανόμαστε το εξωτερικό περιβάλλον. Ο άνθρωπος για παράδειγμα χρησιμοποιεί κυρίως την όραση για να έχει γνώση του εξωτερικού κόσμου. Ένα ρομπότ θα μπορούσε να έχει μία κάμερα. Το θέμα είναι ενώ ο ανθρώπινος εγκέφαλος εξάγει πολλές πληροφορίες από μία εικόνα εύκολα και γρήγορα, σχεδόν αυτόματα όπως αναγνώριση αντικειμένων, εκτίμηση βάθους αντικειμένων, χρώμα, υφή, κτλ.. για το ρομπότ η προσλαμβάνουσα από την κάμερα εικόνα είναι πίνακες με αριθμούς. Και εδώ έρχεται η συμβολή ενός κλάδου της τεχνητής νοημοσύνης, η ρομποτική ή υπολογιστή όραση για να προσδώσει χαρακτηριστικά αντίληψης για το ρομπότ που προκύπτουν από την επεξεργασία αυτών των πινάκων που αντιπροσωπεύουν εικόνες. Οι τρόποι με τους οποίους επιτυγχάνεται αυτό θα αναφερθούν συνοπτικά στα επόμενα κεφάλαια.

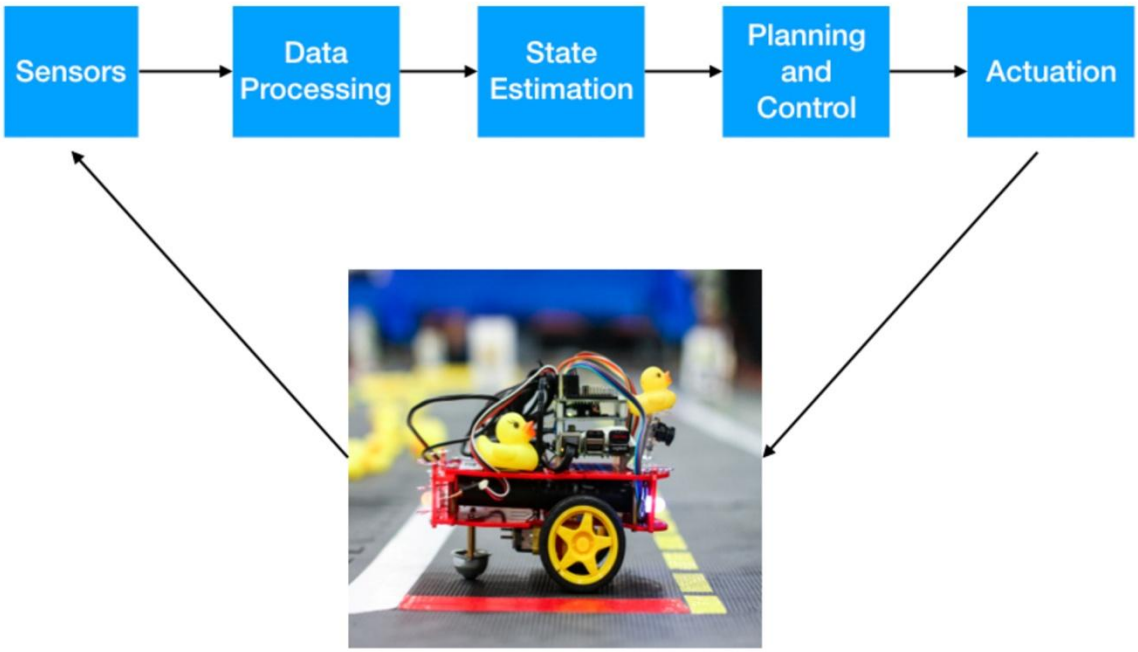
2. 2 Χαρτογράφηση και Εντοπισμός

Απαραίτητη προϋπόθεση για το σχεδιασμό διαδρομής είναι η γνώση του χάρτη και της θέσης μας μέσα σε αυτόν. Η χαρτογράφηση είναι μία τεχνική δημιουργίας ενός συνόλου κωδικοποιημένων και αποθηκευμένων πληροφοριών που απαιτούνται για την αναπαράσταση του περιβάλλοντα χώρου. Μια τεχνική δημιουργίας του χάρτη δηλαδή βασιζόμενοι στην επεξεργασία των δεδομένων που συλλέγουμε από τα αισθητήρια όργανα. Ο εντοπισμός έχει να κάνει μεθόδους και τεχνικές που αφορούν την ακριβή εκτίμηση της θέσης μας μέσα στον χάρτη κάθε τρέχουσα χρονική στιγμή.

2. 3 Σχεδιασμός και έλεγχος κίνησης

Το επόμενο στάδιο μετά τη χαρτογράφηση και τον εντοπισμό και με δεδομένο τον προορισμό είναι η επιλογή της διαδρομής που θα μας οδηγήσει σε αυτόν. Έτσι γνωρίζοντας τη θέση μας μέσα στο χάρτη επιλέγουμε από ένα σύνολο δυνατών διαδρομών τη συντομότερη, ασφαλέστερη και άνευ εμποδίων διαδρομή που θα μας οδηγήσει στο επιθυμητό σημείο. Κατά τη διάρκεια της κίνησης μας διενεργούμε ελέγχους έτσι ώστε αν διαπιστώσουμε ότι παρεκκλίνουμε από τη διαδρομή που σχεδιάσει να επανέλθουμε σε αυτήν ή αν συναντήσουμε απρόσμενα κάποιο εμπόδιο να σχεδιάσουμε εκ νέου μία εναλλακτική διαδρομή που θα μας οδηγήσει στον προορισμό.

Τα παραπάνω στάδια αποτελούν τα δομικά στοιχεία της αρχιτεκτονικής της αυτόνομης πλοήγησης. Για τα έμψυχα όντα οι διεργασίες αυτές γίνονται κάπως αυτόματα αλλά στις μηχανές όμως τα πράγματα είναι διαφορετικά. Απαιτείται μία ακολουθία σταδίων επεξεργασίας των προσλαμβανόντων δεδομένων με εξειδικευμένους αλγορίθμους όπου το αποτέλεσμα του ενός σταδίου θα είναι είσοδος για το επόμενο. Αν θελήσουμε να εστιάσουμε στο συγκεκριμένο project όπου τα Duckiebots φέρουν ως αισθητήριο όργανό μόνο μία κάμερα τότε pipeline της αυτόνομης οδήγησης φαίνεται στην εικόνα που ακολουθεί



Εικόνα 4: Στάδια πλοήγησης του Duckiebot

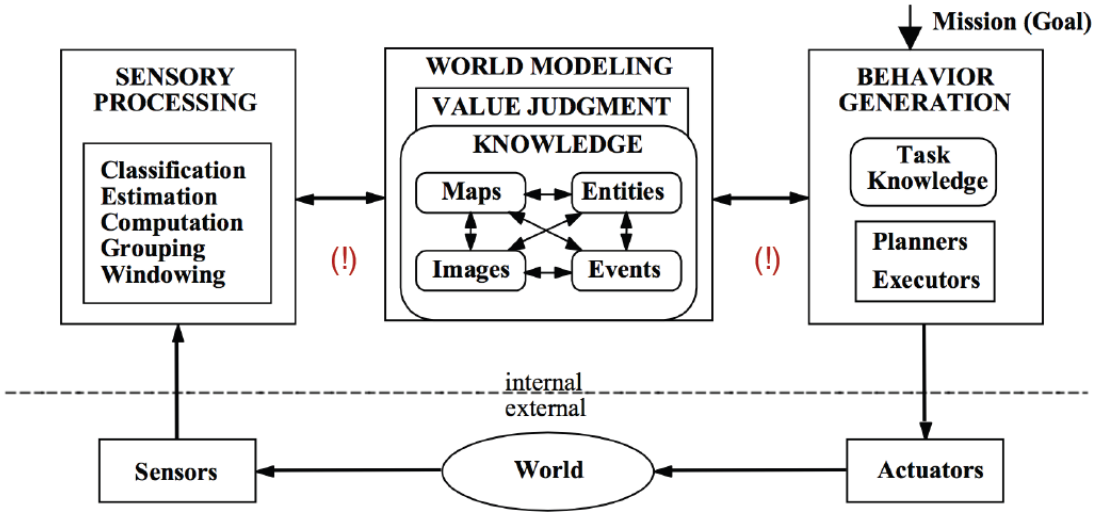
Ο ένας και μοναδικός αισθητήρας που διαθέτει το Duckiebot, η κάμερα λαμβάνει εικόνες από τον περιβάλλον χώρο. Τα δεδομένα που προκύπτουν υφίστανται επεξεργασία ώστε να εξαχθούν σημαντικές και χρήσιμες πληροφορίες για τη θέση του ρομπότ. Η πληροφορία αυτή μεταβιβάζεται στο επόμενο στάδιο όπου λαμβάνοντας υπόψη τον προορισμό του Duckiebot σχεδιάζεται η διαδρομή που πρέπει να ακολουθήσει μέσα στον χάρτη και να δοθούν τελικά οι κατάλληλες εντολές στους ενεργοποιητές ώστε να κινηθεί το ρομπότ προς στην επιθυμητή κατεύθυνση [5].

Γενικότερα έχουν εμφανιστεί αρκετές αρχιτεκτονικές για τη δημιουργία αυτόνομων ρομποτικών κατασκευών. Και η αλήθεια είναι ότι η απόσταση που χωρίζει ένα βιομηχανικό ρομπότ που εκτελεί μία συγκεκριμένη εργασία από ένα αυτόνομα κινούμενο αυτοκίνητο (για παράδειγμα) είναι μεγάλη. Αυτό το μεγάλο κενό έρχονται να καλύψουν οι διάφορες αρχιτεκτονικές που έχουν κατά καιρούς εμφανιστεί όπως το μοντέλο αναφοράς 4D-RCS που αναπτύχθηκε το NIST (National Institute of Standards and Technology) το 2002 που παρέχει μια θεωρητική βάση για το σχεδιασμό, τη μηχανική και την ενσωμάτωση "έξυπνου" λογισμικού σε μη επανδρωμένα οχήματα εδάφους [6].

Βασίζεται και στην αρχιτεκτονική πρακτόρων (agent architecture) που στην επιστήμη των υπολογιστών αυτός ο όρος περιγράφει ένα σχεδιάγραμμα που απεικονίζει πράκτορες λογισμικού και έξυπνα συστήματα ελέγχου και το πως αυτά διασυνδέονται μεταξύ τους. Γενικά ο όρος πράκτορας (agent) χρησιμοποιείται στην επιστήμη των υπολογιστών για να περιγράψει ένα σύστημα υλικού (ρομποτικοί πράκτορες - robotic agents) ή λογισμικού (λογισμικοί πράκτορες - software agents) το οποίο διαθέτει εξειδικευμένες ιδιότητες όπως αυτονομία, κοινωνικότητα, αντιδραστικότητα, προνοητικότητα, κινητικότητα, προσαρμοστικότητα. Κάποιες από αυτές προϋποθέτουν τη δυνατότητα συλλογισμού εκ

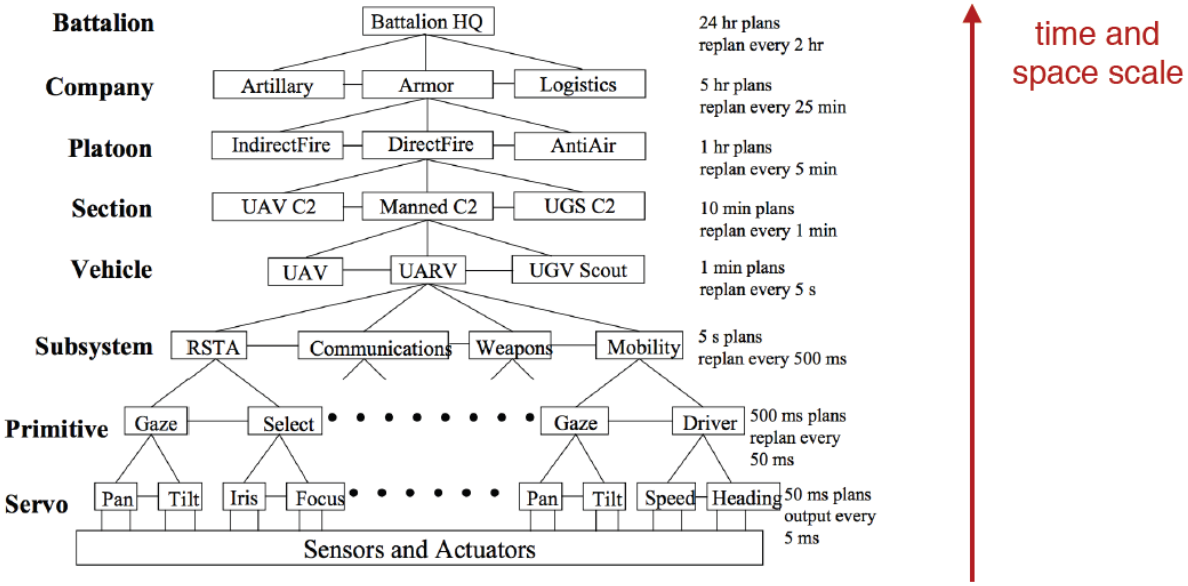
μέρους του πράκτορα. Η λέξη πράκτορας αν και πολλές φορές χρησιμοποιείται αδικαιολόγητα για πολλά σύστημα λογισμικού που αναπτύσσονται παρόλα αυτά η χρήση της δικαιολογείται όταν πρόκειται να περιγράψει μέρη ή για επιμέρους κομμάτια ενός συστήματος τα οποία ενεργούν για λογαριασμό κάποιου χρήστη ή κάποιου άλλου προγράμματος. Δεν υπάρχει ένας ενιαίος ορισμός που να προσδιορίζει επακριβώς τον όρο agent αλλά κατά καιρούς έχουν εμφανιστεί πολλοί ανάλογα με την οπτική με την οποία τους προσεγγίζουμε. Έτσι άλλος ορισμός δίνει έμφαση στην αλληλεπίδραση με το περιβάλλον (Russell και Norvig), άλλος στην αυτονομία τους σε ένα πολύπλοκο περιβάλλον (Maes), άλλος στη συλλογιστική τους ικανότητα (Hayes-Roth) και άλλος στη διαδραστικότητα τους (Coen) [7].

Το μοντέλο αναφοράς 4D/ RCS αναπαριστά σχεδιαστικά ένα μοντέλο ή ένα στιγμιότυπο του κόσμου το οποίο ενημερώνεται με τη χρήση αισθητήρων πραγματικού χρόνου, ελεγκτών και a priori πληροφοριών. Η δράση των πρακτόρων (agents) είναι συνάρτηση της αναπαράστασης του κόσμου όπως αυτή διαμορφώνεται δυναμικά από την επεξεργασία των δεδομένων που λαμβάνουν οι αισθητήρες και φυσικά του επιδιωκόμενου σκοπού. Θα μπορούσαμε να πούμε ότι το παγκόσμιο μοντέλο του κόσμου είναι η καρδιά του συστήματος μας καθώς ενεργεί ως ρυθμιστικό μεταξύ της αντίληψης και της συμπεριφοράς. Η αντίληψη, δηλαδή η κατανόηση χαρακτηριστικών του πραγματικού κόσμου είναι μία πολύπλοκη διαδικασία που περιλαμβάνει διεργασίες όπως ταξινόμηση και ομαδοποίηση χαρακτηριστικών, εντοπισμού, παραθυρικής επεξεργασίας δεδομένων (εφαρμογή φίλτρων) και φυσικά υπολογιστικές πράξεις. Όλα αυτά εφαρμόζονται στα δεδομένα που συλλέγονται από τους αισθητήρες του συστήματος. Η απεικόνιση και μοντελοποίηση του κόσμου μέσα στο σύστημα μας είναι προϊόν αλληλοσυσχέτισης χαρτών, δεδομένων εισόδου, γεγονότων και εικόνων στα πλαίσια μιας γνωσιακής βάσης δεδομένων. Οι καταστάσεις και οι σχέσεις μεταξύ των οντοτήτων, γεγονότων, εικόνων και χαρτών αντιπροσωπεύονται από δείκτες. Οι δείκτες που συνδέουν συμβολικές δομές δεδομένων μεταξύ τους σχηματίζουν συντακτικά, σημασιολογικά, αιτιώδη και περιστατικά δίκτυα και μας επιτρέπουν μέσω του εσωτερικού παγκόσμιου μοντέλου να κατανοήσουμε τον φυσικό κόσμο. Το μοντέλου του κόσμου δεν παραμένει σταθερό και αμετάβλητο αλλά μεταβάλλεται δυναμικά καθώς αλληλεπιδρά τόσο με τις "καινούργιες " κάθε φορά πληροφορίες αντίληψης του εξωτερικού κόσμου όσο και με την παραγόμενη συμπεριφορά του συστήματος μας. Η συμπεριφορά του συστήματος μας καθορίζεται από τους σχεδιαστές που λαμβάνοντας υπόψη τη γνώση που έχουμε για τον εξωτερικό κόσμο και τον επιδιωκόμενο σκοπό του συστήματος δίνουν τις κατάλληλες εντολές στους εκτελεστές ώστε το σύστημα να αλληλεπιδράσει στον πραγματικό κόσμο.



Εικόνα 5 : Βασική εσωτερική δομή του βρόχου ελέγχου του 4D-RCS

Στην πράξη πρόκειται για μία ιεραρχική αρχιτεκτονική η οποία χρησιμοποιεί την ιεραρχική οργάνωση στο χρόνο και στο χώρο για να αντιμετωπίσει την πολυπλοκότητα του εξωτερικού κόσμου. Το παρακάτω σχήμα απεικονίζει ένα υψηλού επιπέδου διάγραμμα της αρχιτεκτονικής του μοντέλου αναφοράς 4D/RCS. Οι εντολές ρέουν από τα υψηλότερα επίπεδα στα χαμηλότερα ενώ οι πληροφορίες που λαμβάνονται από τους αισθητήρες ακολουθούν αντίστροφη πορεία προς τα ανώτερα επίπεδα. Είναι φυσικό πως σε πολλές περιπτώσεις ανταλλάσσεται μεγάλος όγκος πληροφοριών ανάμεσα σε κόμβους του ιδίου επιπέδου ενός υποδέντρου εντολών.

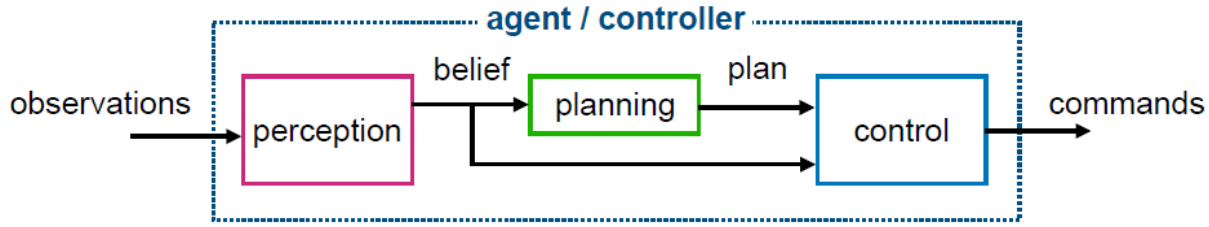


Εικόνα 6 : Υψηλού επιπέδου διάγραμμα μιας τυπικής αρχιτεκτονικής του μοντέλου αναφοράς 4D/RCS

Θέλοντας να κάνουμε μία σύντομη περιγραφή του παραπάνω διαγράμματος από πάνω προς τα κάτω, αρχικά συναντάμε το υψηλότερο επίπεδο, το επίπεδο τάγματος(Battalion). Ονομάζεται έτσι επειδή μπορεί να έχουμε πολλά (160 και πλέον) μη επανδρωμένα οχήματα

διαφορετικού τύπου (όχημα αέρα, εδάφους) κτλ. Ας μη ξεχνάμε ότι το μοντέλο αυτό εμφανίστηκε για την καθοδήγηση μη επανδρωμένων οχημάτων για τον στρατό των Ηνωμένων Πολιτειών της Αμερικής. Το αμέσως χαμηλότερο επίπεδο είναι το επίπεδο της εταιρείας (του λόχου αν θέλουμε να διατηρήσουμε την στρατιωτική ονοματολογία) που περιλαμβάνει μία ομάδα οχημάτων (40) Οι εταιρείες / λόχοι αποτελούνται από διμοιρίες. Αμέσως χαμηλότερα συναντάμε στο επίπεδο της διμοιρίας(platoon) που περιέχει 10 συνήθως οχήματα τα οποία συντονίζονται για τη δημιουργία τακτικών που ορίζει ο λόχος των οποίο ανήκουν. Ακολουθούν τα επίπεδα ενότητας (Section), οχήματος (Vehicle) και υποσυστήματος (subsystem) για να φτάσουμε τελικά στο αρχικό επίπεδο(primitive).Το χαμηλότερο επίπεδο (Servo) περιλαμβάνει τις ομαδοποιημένες εντολές για τους ενεργοποιητές που διαθέτει το σύστημα.

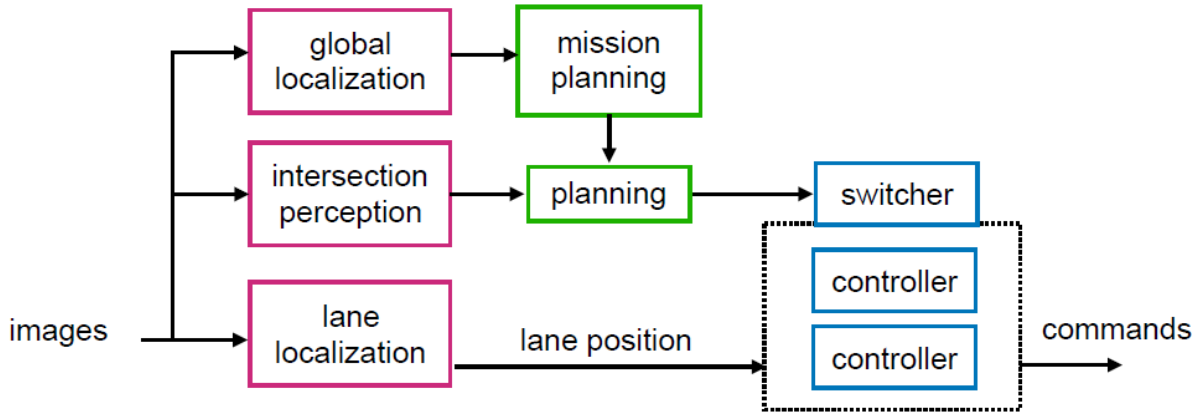
Στο δικό μας project βέβαια δεν έχουμε λόχους και τάγματα έχουμε όμως μη επανδρωμένα οχήματα που θέλουμε να κινούνται αυτόνομα μέσα στην πόλη. Συνεπώς εκείνα που δανειζόμαστε από το μοντέλο αναφοράς 4D/RCS είναι ο βασικός μηχανισμός λειτουργίας του όπως αυτός φαίνεται στο παρακάτω σχήμα.



Εικόνα 7: Μηχανισμός Λειτουργίας της αυτονομίας

Στην γενική του μορφή πρόκειται για ένα μηχανισμό ο οποίος δέχεται παρατηρήσεις, δεδομένα δηλαδή μέσω των αισθητηρίων οργάνων, εξάγει μέσα από πολύπλοκες διαδικασίες χαρακτηριστικά αντίληψης του περιβάλλοντα χώρου στον οποίο αυτό κινείται και χρησιμοποιώντας αυτά τα χαρακτηριστικά σχεδιάζουμε ή επανασχεδιάζουμε τη διαδρομή προκειμένου να φτάσουμε στον προορισμό μας στέλνοντας τις κατάλληλες εντολές στους ενεργοποιητές του συστήματος μας.

Στην περίπτωση μας τώρα οι παρατηρήσεις είναι οι εικόνες που συλλαμβάνει η κάμερα του Duckiebot. Από τις εικόνες αυτές μπορούμε να εξάγουμε τρία βασικά χαρακτηριστικά αντίληψης. Την θέση του Duckiebot μέσα στον χάρτη στης πόλης, να αναγνωρίσουμε τα σημεία τομής των δρόμων, τις διασταυρώσεις δηλαδή και τέλος να έχουμε γνώση της θέσης της γραμμής του δρόμου σε σχέση το Duckiebot. Με βάση αυτά τα χαρακτηριστικά και λαμβάνοντας υπόψη τον τελικό προορισμό σχεδιάζουμε τη διαδρομή που θα πρέπει να ακολουθήσουμε, μεριμνώντας ταυτόχρονα το Duckiebot να κινείται εντός του δρόμου μεταβιβάζοντες τις κατάλληλες εντολές στους ελεγκτές των ενεργοποιητών (servo) που κινούν το όχημα. Πρόκειται για μία διαδικασία που επαναλαμβάνεται όπως είναι φυσικό καθώς κινείται το Duckiebot καθώς η κάμερα λαμβάνει καινούργιες εικόνες βάσει των οποίων γεννιούνται νέα χαρακτηριστικά αντίληψης τα οποία εκτιμώνται εκ νέου για τον έλεγχο ή επανασχεδιασμό της διαδρομής.



Εικόνα 8: Ανάλυση των σταδίων της αυτονομίας

Γενικά υπάρχουν πολλοί τρόποι για να προσεγγίσει και να σχεδιάσει κανείς αρχιτεκτονικές για ρομποτικά οχήματα αυτόνομης πλοήγησης. Σίγουρα όλες περιλαμβάνουν τις έννοιες της αντίληψης, του σχεδιασμού και του ελέγχου. Άλλες δίνουν έμφαση σε ένα ιεραρχικό μοντέλο διαδικασιών ως προς το χρόνο και το χώρο ενώ άλλες εφαρμόζουν τεχνικές σχεδιασμού από πάνω προς τα κάτω (top-down) ή από κάτω προς τα πάνω (bottom-up). Σε κάθε περίπτωση προσπαθούμε να μιμηθούμε τον τρόπο λειτουργίας των έμψυχων υποκειμένων (ζώων - ανθρώπων). Βέβαια στο σημείο αυτό θα πρέπει να τονίσουμε ότι οι μηχανισμοί που διαθέτουν τόσο οι άνθρωποι όσο και τα ζώα και τους επιτρέπουν να πλοηγούνται αυτόνομα είναι αρκετά διαφορετικοί από αυτούς που προσπαθούμε να κατασκευάσουμε για τις μηχανές, περισσότερο πολύπλοκοι και σε κάποιες περιπτώσεις ακόμα προς διερεύνηση.

Κεφάλαιο 3:

Διαμόρφωση της πόλης Duckietown

Η πλατφόρμα του Duckietown περιλαμβάνει από άποψη υλικού δύο βασικά και θεμελιώδη μέρη. Την πόλη (Duckietown) μέσα στην οποία κινούνται τα οχήματα και τα ίδια το οχήματα που αποκαλούνται Duckiebot. Αυτά τα δύο αναπόσπαστα κομμάτια συνθέτουν την πλατφόρμα του Duckietown και την καταστούν λειτουργική.

Η πόλεις αποτελούν αναπόσπαστο κομμάτι του ρομποτικού οικοσυστήματος που ονομάζεται Duckietown. Έχουν σχεδιαστεί με τέτοιο τρόπο ώστε να "στέλνουν" πληροφορίες στα Duckiebots ώστε αυτά να τις λαμβάνουν, να τις επεξεργάζονται και να μπορούν να λειτουργήσουν κατάλληλα μέσα σ' αυτήν. Οι πόλεις αποτελούνται από θεμελιώδης δομικές μονάδες, τα πλακίδια, ώστε να μπορούν να συνδυαστούν ποικιλοτρόπως δημιουργώντας διαφορετικούς κάθε φορά κόσμους θυμίζοντας ή προσομοιάζοντας πραγματικές πόλεις. Στο σημείο αυτό να επισημάνουμε ότι υπάρχουν δύο διαφορετικές περιγραφικές βιβλιοθήκες διαμόρφωσης τόσο των πόλεων όσο και των Duckiebots.

- Η τρέχουσα (2019 version) σταθερή (stable) βιβλιοθήκη διαμόρφωση της πλατφόρμας Duckietown[9]
- Και η πειραματική βιβλιοθήκη διαμόρφωσης της πλατφόρμας [8] που προτείνεται για ανάπτυξη και έρευνα.

Εμείς αρχικά επιλέξαμε στην "επίσημη", σταθερή διαμόρφωση αν και όπως θα δούμε στην πορεία λόγω δυσλειτουργιών και άλλων προβλημάτων καταλήξαμε στην πειραματική. Η διαμόρφωση της πόλης έγινε στηριζόμενοί στις οδηγίες και στις προδιαγραφές της stable βιβλιοθήκης.

Υπάρχουν τρία είδη περιβαλλόντων - πόλεων που μπορούμε να κατασκευάσουμε προκειμένου να ανιχνεύσουμε και να εξερευνήσουμε τις λειτουργίες των Duckiebots.

1. Χωρίς πόλη - μόνο πινακίδες σήμανσης

Στην περίπτωση αύτη δεν κατασκευάζουμε πόλη, απλά τοποθετούμε τις πινακίδες σήμανσης. Εννοείται πως το Duckiebot δεν είναι δυνατό να παρουσιάσει το μεγαλύτερο μέρος της λειτουργικότητας του, μπορούμε όμως να ελέγξουμε την ανίχνευση ετικετών, τη βαθμονόμηση της κάμερας και τις διαδικασίες αναγνώρισης προτύπων του Duckiebot.

2. Πόλη βρόχος

Πρόκειται για την πιο απλή μορφή πόλης που μπορούμε να κατασκευάσουμε με κλειστούς δρόμους που θυμίζουν θηλιά χωρίς διασταυρώσεις. Το μόνο απαιτεί μια τέτοια πόλη είναι το επίπεδο του δαπέδου και η χάραξη του δρόμου. Σε μια τέτοια πόλη μπορούμε να ελέγξουμε το αν και κατά πόσο το Duckiebot είναι σε θέση να εξάγει πληροφορίες από τις εικόνες που λαμβάνει και με βάση αυτές να δημιουργήσει μια "πεποίθηση" για τη θέση και τον

προσανατολισμό του στις λωρίδες του δρόμου. Επίσης μπορούμε να διαπιστώσουμε αν είναι ικανό να διατηρεί μία απόσταση ασφαλείας από τα προπορευόμενα Duckiebots. Και τέλος αν η πόλη διαθέτει και πινακίδες σήμανσης να ελέγξουμε τη δυνατότητα ανίχνευσης των πινακίδων και εκτίμησης του προσανατολισμού του στον χάρτη της πόλης. Αυτό είναι το πακέτο εκκίνησης του Duckietown.

Εικόνα 9 : Πόλη βρόχος

3. Πόλη πλοήγησης

Μία πόλη πλοήγησης περιλαμβάνει δρόμους με διασταυρώσεις, φωτεινούς σηματοδότες και πινακίδες σήμανσης. Σε μια τέτοια πόλη μπορούμε να εξερευνήσουμε πιο σύνθετες -σε σχέση με τη πόλη βρόχο- συμπεριφορές του Duckiebot.

Εικόνα 10 : Πόλη πλοήγησης

4. Robotarium

Με τον όρο Robotarium περιγράφουμε ουσιαστικά πειραματικές Duckietowns σχεδιασμένες έτσι ώστε να ευδοκιμεί η διερεύνηση και η δοκιμή περεταίρω λειτουργιών και δυνατοτήτων των Duckiebots.

Εικόνα 11 : Robotarium

Εμείς είχαμε στη διάθεση μας το πακέτο Classroom Kit (5) το οποίο περιλαμβάνει :

- Πέντε (5) Duckiebots
- Ένα πακέτο για τη δημιουργία μιας πόλης βρόχου
- Ένα πακέτα για τη δημιουργία μιας πόλης πλοήγησης

Με βάση λοιπόν το πακέτο που είχαμε στην κατοχή μας διαμορφώσαμε μία πόλη πλοήγησης και πέντε (5) Duckiebots. Στις ενότητες που ακολουθούν γίνεται λεπτομερή περιγραφή τόσο του τρόπου διαμόρφωσης της πόλης πλοήγησης όσο και της κατασκευής των Duckiebots.

3. 1 Κατασκευή πόλης

Σε μία πόλη (Duckietown) μπορούμε να διακρίνουμε δύο διακριτά επίπεδα : το πάτωμα με τις διαγραμμίσεις του και το επίπεδο των σημάτων.

Το στρώμα του δαπέδου είναι το υπόστρωμα πάνω στο οποίο κινούνται τα Duckiebots, αποτελεί δηλαδή το δρόμο. Ανεξάρτητα από τη γεωμετρία των δρόμων (ευθεία, καμπύλες, διασταυρώσεις), γίνονται δρόμοι δύο λωρίδων-μία για κάθε κατεύθυνση-οδήγησης. Οι λωρίδες αυτές διαμορφώνονται με χρωματιστές ταινίες που κολλούνται πάνω στα πλακίδια με κάποιες προδιαγραφές.

Εκτός από τη χάραξη των δρόμων υπάρχει και το επίπεδο των σημάτων που περιέχει πινακίδες και άλλα λειτουργικά αντικείμενα όπως σηματοδότες και παρατηρητήρια. Τα αντικείμενα αυτά αλληλεπιδρούν με τα Duckiebots και ελέγχουν ή καθοδηγούν την συμπεριφορά των Duckiebots όπως για παράδειγμα οι φωτεινοί σηματοδότες.

Τέλος μπορεί να υπάρχουν και μη λειτουργικά αντικείμενα όπως οι κάτοικοι αυτής τη πόλης (τα παπάκια), κτίρια και άλλα διακοσμητικά αντικείμενα που ομορφαίνουν την πόλη και αυξάνουν την ρεαλιστικότητα της.

3.1.1 Προδιαγραφές

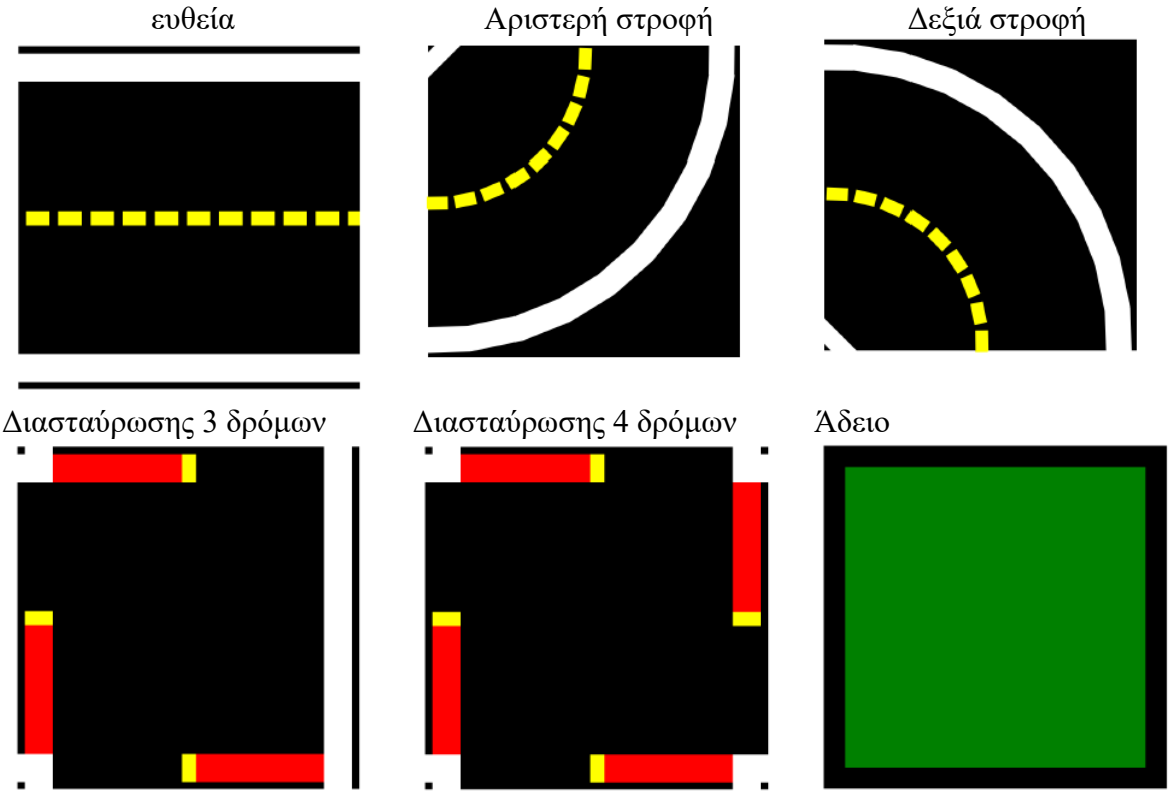
Στο σημείο αυτό θα περιγράψουμε τις προδιαγραφές που πρέπει να πληροί μια Duckietown πόλη. Ως προδιαγραφές ορίζουμε το σύνολο των κανόνων που έχει επαληθευτεί ότι

καταστούν λειτουργική την πόλη από τα Duckiebots. Αυτό πρακτικά σημαίνει ότι αν ακολουθηθούν αυτοί οι κανόνες κατά την οικοδόμηση της πόλης τα Duckiebots θα μπορέσουν να κινηθούν και να λειτουργήσουν αλληλεπιδρώντας μέσα σ' αυτήν ενώ οποιαδήποτε απόκλιση από τις προδιαγραφές αυτόματα μπορεί να είναι αιτία αποτυχίας της λειτουργίας των Duckiebots μέσα σ' αυτή και η οποία τελικά δεν θα θεωρείται Duckietown. Ακόμα και μικρές διαταραχές των προδιαγραφών μπορούν να επηρεάσουν αρνητικά την απόδοση των Duckiebots αν και οι περισσότεροι αλγόριθμοι είναι εύρωστοι στις παραλλαγές.

3.1.2 Επίπεδο 1 - Το επίπεδο του δαπέδου

Το στρώμα του δαπέδου αποτελείται από μαύρα πλακίδια. Κάθε πλακίδιο μπορεί εν δυνάμει να αποτελέσει μία ευθεία ή μία καμπύλη ή μία διασταύρωση ή τέλος έναν κενό, άδειο χώρο. Τα οδικά χαρακτηριστικά διαμορφώνονται πάνω στα πλακίδια με χρήση ειδικών αυτοκόλλητων, χρωματιστών ταινιών με συγκεκριμένο τρόπο και προδιαγραφές [11].

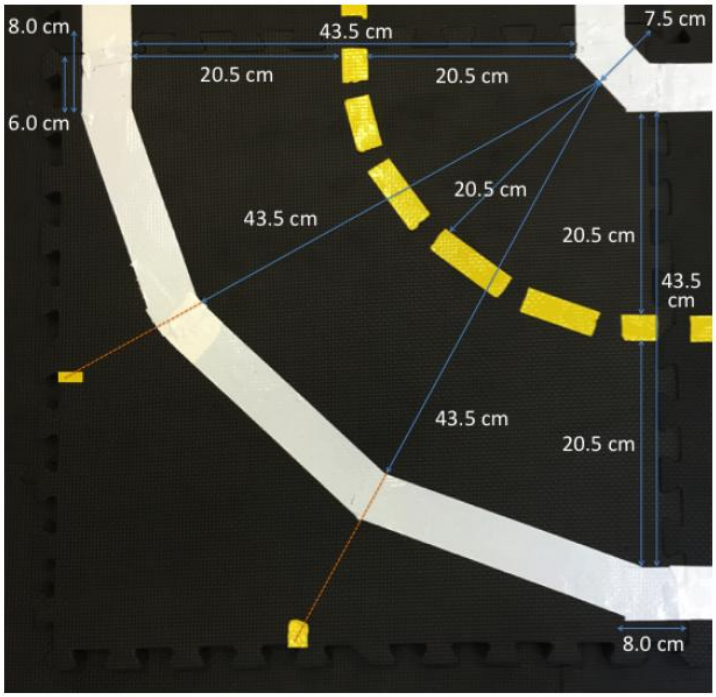
Το κάθε πλακίδιο είναι τετράγωνο διαστάσεων 61εκατοστών x 61εκατοστών και στις άκρες του διαθέτει άκρα (αλληλομανδαλώσεως) κατάλληλα για να μπορεί να συνδεθεί με άλλα. Το πάχος των πλακιδίων δεν είναι τόσο σημαντικό όσο η τραχύτητα της άνω επιφάνειας τους με στόχο την καλή πρόσφυση των Duckiebots πάνω σ' αυτά ώστε να ελαχιστοποιηθεί η ολίσθηση των τροχών.



Πίνακας 1: Τύποι πλακιδίων

Υπάρχουν τρία (3) διαφορετικά χρώματα ταινιών που χρησιμοποιούνται για τη διαμόρφωση του οδικού δικτύου σε μια Duckietown. Το λευκό, το κίτρινο και το κόκκινο. Η λευκή ταινία έχει πλάτος 4,8 εκατοστά (1,88 inches) και χρησιμοποιείται να την χάραξη των δρόμων και τοποθετείται πάντα ενιαία και όχι διακεκομμένη. Τα Duckiebots κινούνται πάντα στη δεξιά πλευρά του δρόμου και προκειμένου να μην συγκρουστούν με άλλα Duckiebots ή άλλα αντικείμενα της Duckietown δεν επιτρέπεται να διασχίσουν ή να αγγίξουν την λευκή ταινία που οριοθετεί ένα δρόμο. Οι δρόμοι σχηματίζονται από δύο παράλληλες λευκές ταινίες οι οποίες απέχουν η μία από την άλλη 43,5 εκατοστά.

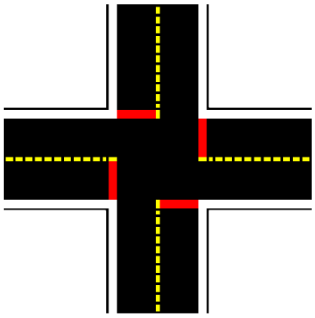
Προκειμένου να σχηματιστούν οι στροφές η λευκή ταινία τεμαχίζεται σε κομμάτια. Συγκεκριμένα για την διαμόρφωση της εξωτερικής πλευράς της στροφής απαιτούνται πέντε (5) κομμάτια λευκής ταινίας ενώ για την εσωτερική αποτελείται από τρία (3) κομμάτια. Οι αποστάσεις και οι προδιαγραφές που πρέπει να πληρούνται στον σχηματισμό των καμπυλών απεικονίζονται στην παρακάτω εικόνα.



Εικόνα 12: Διαγράμμιση στροφής

Η κίτρινη ταινία χρησιμοποιείται για την οριοθέτηση των δύο λωρίδων κυκλοφορίας στον δρόμο. Έχει πάχος περίπου 2,4 εκατοστά (0,94 inches) και τοποθετείται στο κέντρο του δρόμου, διακεκομμένη σε κομμάτια μήκους 5 εκατοστών ενώ κάθε κομμάτι απέχει από το άλλο απόσταση ίση με 2,5 εκατοστά. Στις στροφές όπως φαίνεται και από την παραπάνω εικόνα κάθε κομμάτι της κίτρινης λωρίδας πρέπει να βρίσκεται στο μέσο της καμπύλης, δηλαδή σε απόσταση 20,5 εκατοστών περίπου από το κομμάτι της λευκής ταινίας που είναι τοποθετημένο στο εσωτερικό της στροφής.

Η κόκκινη ταινία χρησιμοποιείται στις διασταυρώσεις. Έχει το ίδιο πλάτος με την άσπρη ταινία δηλαδή, 4,8 εκατοστά (1,88 inches) και τοποθετείται κάθετα στον δρόμο όπως φαίνεται στο παρακάτω σχήμα



Εικόνα 13: Διασταύρωση τεσσάρων κατευθύνσεων

Ένα Duckiebot κινείται μέσα σε μια Duckietown ως εξής: κινείται κατά μήκος της άσπρης λωρίδας μέχρις ότου εμφανιστεί η κόκκινη κάθετη ταινία όπου και σταματάει και περιμένει σήμα συντονισμού. Μόλις δοθεί το κατάλληλο σήμα συντονισμού και το Duckiebot περάσει επιτυχώς από το σημείο τομής των δρόμων (διασταύρωση) επανέρχεται σε κατάσταση κίνησης ακολουθώντας την άσπρη γραμμή του δρόμου.

Υπάρχουν ορισμένοι τοπολογικοί περιορισμοί κατά την κατασκευή του οδικού χάρτη οι οποίοι πρέπει να πληρούνται για την ομαλή κίνηση των Duckiebots και οι οποίοι είναι οι εξής:

1. Μια διασταύρωση δεν μπορεί να είναι δίπλα σε μία στροφή ή σε μια άλλη διασταύρωση
2. Για δύο οποιαδήποτε γειτονικά πλακίδια τα οποία ανήκουν στο οδικό δίκτυο της πόλης (Duckietown) πρέπει να υπάρχει μία εφικτή διαδρομή η οποία να συνδέει το ένα με το άλλο.

Προαιρετικά μπορούμε να χρησιμοποιήσουμε κάποια πλακίδια ως χώρο στάθμευσης των Duckiebots. Υπάρχουν τρεις διαφορετικοί τύποι πλακιδίων που συνθέτουν έναν χώρο στάθμευσης.

1. Πλακίδιο εισόδου στον χώρο στάθμευσης,
2. Πλακίδια χώρου στάθμευσης
3. Πλακίδια πρόσβασης σε σημεία στάθμευσης

Οι κανόνες που πρέπει να διέπουν έναν χώρο στάθμευσης είναι οι ακόλουθοι:

1. Ένα σημείο στάθμευσης έχει μέγεθος όσο ένα πλακίδιο
2. από κάθε σημείο στάθμευσης υπάρχει ένα τέτοιο μονοπάτι που οδηγεί στο πλακίδιο εισόδου στον χώρο στάθμευσης ώστε όταν ένα Duckiebot είναι σταθμευμένο να μην εμποδίζει την πρόσβαση στον χώρο σε άλλα Duckiebots
3. από οποιαδήποτε θέση μέσα στον χώρο στάθμευσης ένα Duckiebot μπορεί να δει τουλάχιστον δύο ορθογώνιες γραμμές ή ένα μια πινακίδα.

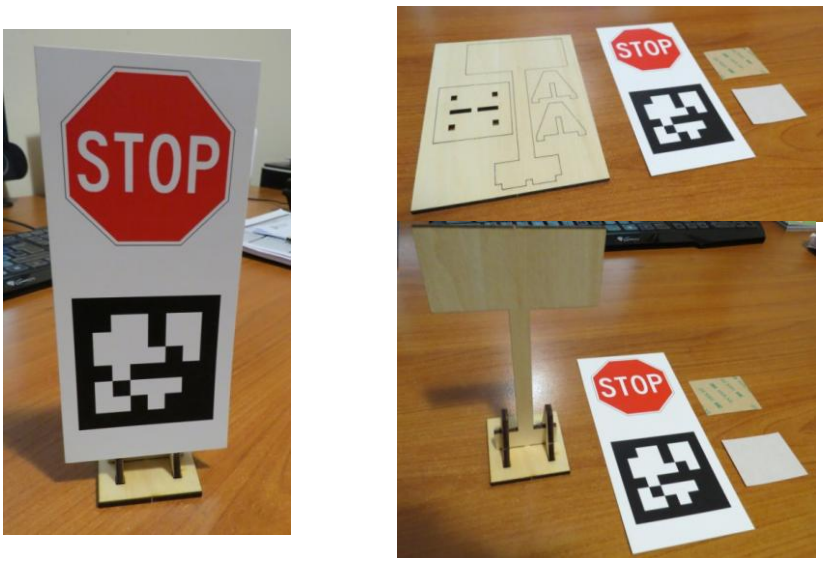
Εμείς στην παρούσα υλοποίηση επιλέξαμε να μην διαμορφώσουμε χώρο στάθμευσης καθώς ακόμα βρίσκεται σε πειραματικό στάδιο. Θα μπορούσε όμως να είναι όμως μια επέκταση της παρούσας εργασίας που να υλοποιηθεί στον μέλλον.

Τέλος επίσης προαιρετικά μπορούμε να διαμορφώσουμε ένα χώρο εκκίνησης για την εισαγωγή ενός νέου Duckiebot στην Duckietown με ελεγχόμενο τρόπο. Το πλακίδιο εκκίνησης θα πρέπει να τοποθετείται δίπλα σε μία καμπύλη (στροφή) ώστε το Duckiebot να "συγχωνευθεί" στην πόλη μόλις ολοκληρωθεί η διαδικασία αρχικοποίησης.

3.1.3 Επίπεδο 2 : Πινακίδες σήμανσης

Οι πινακίδες σήμανσης αποτελούν το δεύτερο επίπεδο του της πόλης Duckietown και χρησιμοποιούνται για να βοηθήσουν τα Duckiebots με τρόπο παρόμοιο όπως βοηθούν οι πινακίδες σήμανσης τους οδηγούς στο οδικό δίκτυο στο οποίο κινούνται [12]. Απεικονίζουν το σήμα οδικής κυκλοφορίας και από κάτω υπάρχει μία ετικέτα AprilTag που είναι στην ουσία ένα είδος κωδικός QR (Quick Response) , δηλαδή ένας δισδιάστατος γραμμωτός κώδικας που μπορεί να εύκολα και γρήγορα να εντοπιστεί από τα Duckiebot.

Υπάρχουν ξύλινοι ορθοστάτες οι οποίοι εύκολα συναρμολογούνται και πάνω σε αυτούς επικολλούνται οι πινακίδες σήμανσης με τη βοήθεια κολλητικής ταινίας διπλής όψης. Ένα σημείο το οποίο χρήζει προσοχής όταν κολλούμε τις πινακίδες πάνω στους ξύλινους ορθοστάτες είναι ότι πρέπει να υπάρχει μία απόσταση τουλάχιστον 0,5 εκατοστών στη πινακίδας από την βάση του ορθοστάτη.



Εικόνα 14 : Διαμόρφωση πινακίδων σήμανσης

Για να είναι συμβατή η σήμανση κυκλοφορίας με τη λειτουργία της πόλης θα πρέπει να πληρούνται οι παρακάτω προδιαγραφές:

- Το κέντρο των σημάτων κυκλοφορίας να απέχει 13 εκατοστά από το στρώμα του δαπέδου.

- Το Apriltag να είναι 6,5 εκατοστά.
- Να υπάρχει ένα λευκό περίγραμμα πάχους περίπου 0,8 εκατοστών γύρω από κάθε πινακίδα
- Οι πινακίδες σήμανσης να είναι τοποθετημένες κάθετα στο έδαφος και η γωνία της πινακίδας με τον δρόμο να είναι 90.
- Η πινακίδα να μην έχει παραμορφώσεις, ρυτίδες ώστε γενικά να είναι ευανάγνωστη.

Τα επιτρεπόμενα σήματα κυκλοφορίας παρουσιάζονται στους παρακάτω πίνακες:

Στοπ	Παραχώρηση προτεραιότητας	Απαγόρευση δεξιάς στροφής	Απαγόρευση αριστερής στροφής
			
Απαγορεύεται η είσοδος	Μονόδρομος δεξιά	Μονόδρομος αριστερά	Διασταύρωση τεσσάρων δρόμων
			

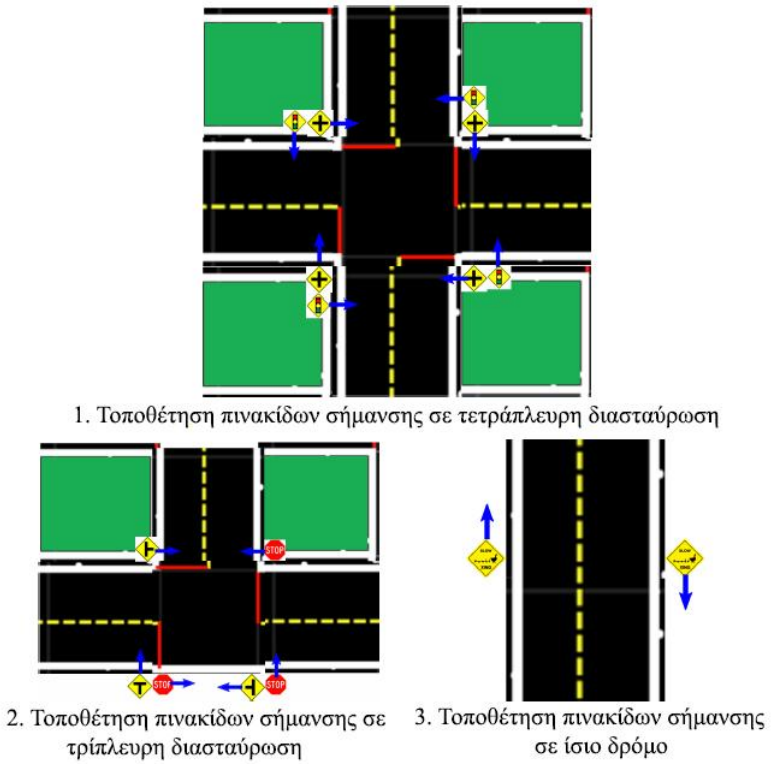
Πίνακας 2: Σήματα Πινακίδων Σήμανσης

Δεξιά -Τ- διασταύρωση	Αριστερή -Τ- διασταύρωση	Τ- διασταύρωση	Πεζοί
			
Φωτεινός σηματοδότης	Αργά - διέλευση παπιών	Παρκινγκ	
			

Πίνακας 3 : Σήματα Πινακίδων Σήμανσης

Τοποθέτηση Πινακίδων Σήμανσης

Οι πινακίδες σήμανσης μπορούν να εμφανίζονται στην απέναντι πλευρά του δρόμου από αυτή που κινείται τα Duckiebots ή στις απέναντι γωνίες προκειμένου να είναι ορατές από αυτό. Αν δεν υπάρχουν πινακίδες κυκλοφορίας τα Duckiebots θεωρούν ότι η κίνηση προς όλες τις κατευθύνσεις είναι δυνατή. Έτσι η πινακίδες τοποθετούνται στις περιπτώσεις που δεν θέλουμε να συμβαίνει αυτό. Τοποθετούνται μόνο σε κενά πλακίδια στα σύνορα του οδικού χάρτη της πόλης και σε καμία περίπτωση δεν πρέπει να επικαλύπτουν τη λευκές λωρίδες του δρόμου. Σε κάθε γραμμή στάσης σε μία διασταύρωση δύο σημεία πρέπει να είναι ευδιάκριτα. Ο τύπος της διασταύρωσης (αν έχει σήμα stop ή φωτεινούς σηματοδότες) και η τοπολογία της. Ειδικά η διασταύρωση τεσσάρων κατευθύνσεων πρέπει να είναι εξοπλισμένη και φωτεινούς σηματοδότες για την ασφαλή διέλευση των Duckiebots από αυτήν.



Εικόνα 15: Τοποθέτηση πινακίδων σήμανσης

Σήμανση ονομασίας οδών

Υπάρχει η δυνατότητα ονοματοδοσίας των οδών της πόλης με χρήση ειδικών πινακίδων σήμανσης που πρέπει να πληρούν τις παρακάτω προδιαγραφές [13]:

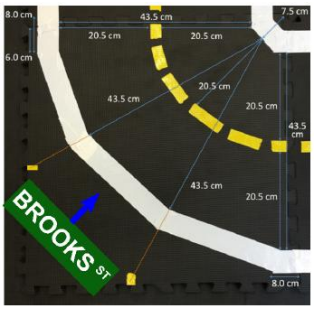
- Γραμματοσειρά : Arial
- Χρώματα : άσπρα γράμματα σε πράσινο φόντο
- Ύψος πινακίδας: 2.1 in

- Πλάτος πινακίδας: 6.1 in + 1.1 in για την ένδειξη ST ή 5.5 In +1.7 in για την ένδειξη AVE
- Αλφάβητο: Αγγλικά κεφαλαία
- Κατεύθυνση κειμένου: Οριζόντια

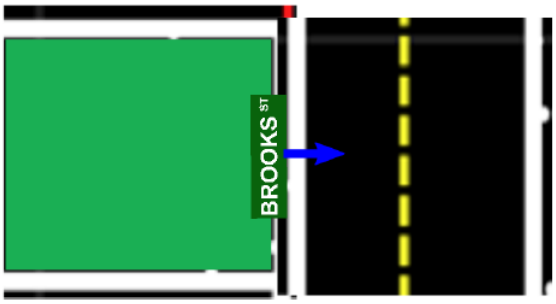
Εικόνα 16: Πινακίδα ονοματοδοσίας οδού

Οι πινακίδες ονοματοδοσίας των οδών πρέπει να τοποθετούνται είτε στην εξωτερική καμπή ενός πλακιδίου στροφής είτε παράλληλα με τον δρόμο και σε κάθε περίπτωση έξω από την επιτρεπόμενη περιοχή οδήγησης όπως φαίνεται στις παρακάτω εικόνες . Τα ονόματα των οδών πρέπει να είναι ορατά στα Duckiebots και από τις δύο πλευρές του δρόμου. Κάθε τμήμα οδού πρέπει να έχει τουλάχιστον ένα όνομα οδικής κυκλοφορίας.

Τοποθέτηση σε στροφή



Τοποθέτηση σε ευθεία



Εικόνα 17: Τοποθέτηση πινακίδων ονομασίας οδών

Τέλος η πόλη μας είναι δυνατό να διαθέτει και φωτεινούς σηματοδότες οι οποίοι θα αναγνωρίζονται από τα Duckiebots και θα ρυθμίζουν την κυκλοφορία στις διασταυρώσεις. Επιπλέον είναι εφικτό να ενσωματώνουν και μία κάμερα η θα λειτουργεί ως παρατηρητήριο και θα εποπτεύει μέρος της πόλης. Περισσότερες πληροφορίες τόσο για τη συναρμολόγηση όσο και για τη λειτουργία τους γι αυτούς δίνονται στο επόμενο κεφάλαιο.

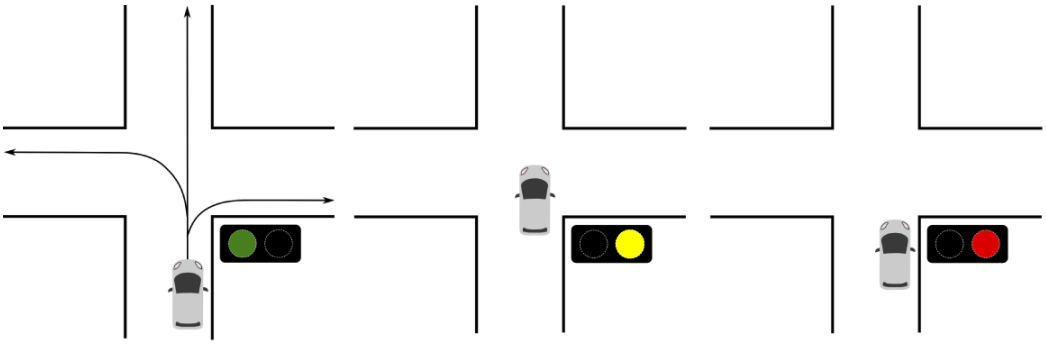
Εικόνα 18: Φωτεινοί σηματοδότες

Κεφάλαιο 4

Φωτεινοί σηματοδότες - Παρατηρητήριο

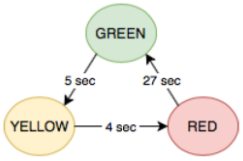
Οι φωτεινοί σηματοδότες χρησιμοποιούνται για την ρύθμιση της κυκλοφορίας στις διασταυρώσεις. Μπορούν να τοποθετηθούν σε διασταυρώσεις τριών ή τεσσάρων δρόμων. Υπό μία έννοια μπορούν να θεωρηθούν ως Duckiebots χωρίς τροχούς, αλλά τοποθετημένα στο δικό τους ειδικά διαμορφωμένο πλαίσιο καθώς αλληλεπιδρούν με το περιβάλλον. Για να αναγνωριστούν οι σηματοδότες από τα διερχόμενα Duckiebots θα πρέπει να υπάρχει κατάλληλή σήμανση στις διασταυρώσεις. Οι φωτεινοί σηματοδότες αποτελούνται από τέσσερα (4) LEDs και τοποθετούνται 20 εκατοστά από το πάνω από το κέντρο της διασταύρωσης των οδών. Η λειτουργίας τους ελέγχεται από ένα Single Board Computer και συγκεκριμένα από το Raspberry Pi το οποίο εδράζεται σε ένα ξύλινο περίβλημα το οποίο είναι τοποθετημένο εκτός της επιτρεπόμενης περιοχής οδήγησης.

Ένα Duckiebot πρέπει να φτάσει σε πλήρη στάση πριν τη γραμμή στάσης όταν το φως του σηματοδότη είναι κόκκινο. Όταν είναι πράσινο τα οχήματα μπορούν να διασχίσουν τη διασταύρωση προς όλες τις κατευθύνσεις. Τέλος όταν η ένδειξη του φωτεινού σηματοδότη είναι κίτρινη τα Duckiebots επιτρέπεται να διασχίζουν τη διασταύρωση αν ήδη έχουν περάσει τη γραμμή στάσης [3].



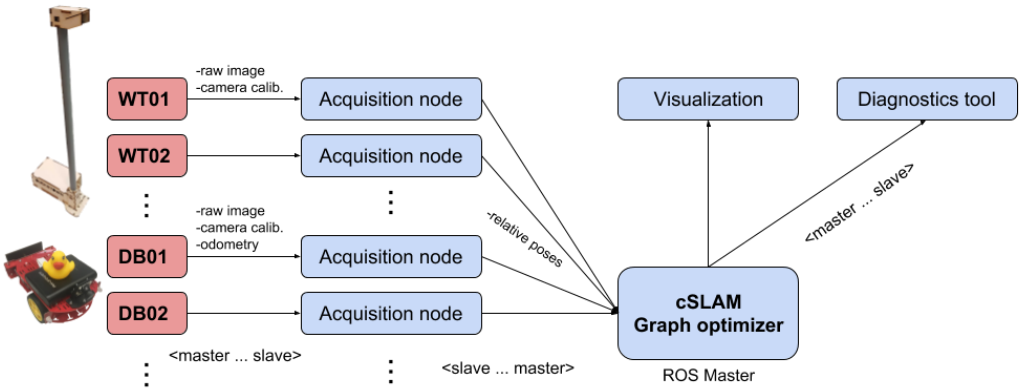
Εικόνα 19: Κανόνες διέλευσης οχημάτων από τους σηματοδότες

Το πολύ σε μία κατεύθυνση μπορεί να έχει πράσινο φως κάθε φορά. Ο χρονοπρογραμματισμός έχει ως εξής: Πράσινο φώς για 5 δευτερόλεπτα, ακολουθεί κίτρινο φως για 4 δευτερόλεπτα και στη συνέχεια κόκκινο φως για 27 $(3 \cdot (5 + 4))$ δευτερόλεπτα. Στη συνέχεια μία από τις άλλες τέσσερις κατευθύνσεις θα είναι ανοιχτή για διέλευση οχημάτων, θα έχει δηλαδή πράσινο φως.



Εικόνα 20: Χρονοπρογραμματισμός Σηματοδοτών

Αν τοποθετήσουμε με ένα ψηλό σημείο αυτής της ξύλινης κατασκευής των φωτεινών σηματοδοτών, σε απόσταση 60 εκατοστών από τα πλακίδια της πόλης, μία κάμερα η οποία θα εποπτεύει μέρος της πόλης τότε έχουμε ένα παρατηρητήριο. Τα παρατηρητήρια λειτουργούν ως κάμερες επιτήρησης μέσα σε μία Duckietown που μπορούν να "δουν" μια περιοχή της πόλης και να παρακολουθήσουν τα Duckiebots που όταν αυτά κινούνται μέσα στο οπτικό πεδίο του παρατηρητηρίου. Ο σκοπός του παρατηρητηρίου είναι ο εντοπισμός της θέσης των Duckiebots μέσα στην πόλη (Duckietown). Αυτό επιτυγχάνεται με τη χρήση των ανιχνευόμενων πινακίδων (Apriltags). Για να είναι εφικτό αυτό κάθε Duckiebot θα πρέπει να είναι εφοδιασμένο με ένα ειδικό πλαίσιο στο οποίο τοποθετείται μία πινακίδα (Apriltag). Έτσι ανιχνεύοντας αυτές τις πινακίδες που φέρουν τα Duckiebots είναι δυνατός ο εντοπισμός τους και ο υπολογισμός των θέσεων τους στον χάρτη της πόλης. Βέβαια αυτό προϋποθέτει την ύπαρξη περισσοτέρων του ενός παρατηρητηρίων οπότε η εκτίμηση των τρεχουσών θέσεων των Duckiebots προκύπτει μέσα από τη συγχώνευση των παρατηρήσεων των πύργων παρακολούθησης μέσα από μια cSLAM αρχιτεκτονική [14].



Εικόνα 21: Διασύνδεση οντοτήτων αρχιτεκτονικής cSLAM

Στον πυρήνα αυτής της υλοποίησης βρίσκεται ένα πρόβλημα βελτιστοποίησης γραφήματος. Οι πύργοι παρακολούθησης παρατηρούν τις ετικέτες AprilTags στο έδαφος, πάνω στα Duckiebots ακόμα τις πινακίδες σήμανσης. Τα Duckiebots "βλέποντας" τις ετικέτες και τις πινακίδες σήμανσης εκτιμούν τις σχετικές τους πόζες. Όλα τα παραπάνω δεδομένα καθώς και οι χρόνοι που λήφθηκαν τα δεδομένα αυτά συνδυάζονται σε ένα πρόβλημα βελτιστοποίησης γραφήματος που τελικά επιλύεται από την g2o βιβλιοθήκη. Πρόκειται για μία δομή ανοικτού κώδικα σε C++ για τη βελτιστοποίηση μη γραμμικών λειτουργιών που βασίζονται σε γραφήματα [6] παρέχοντας λύσεις σε διάφορες παραλλαγές προβλημάτων ταυτόχρονου εντοπισμού και χαρτογράφησης (SLAM) ή ρύθμιση δέσμης (BA).

Όπως φαίνεται και στην παραπάνω εικόνα υπάρχουν πολλές φυσικές οντότητες (Duckiebots και Watchtowers) καθώς Docker containers (θα μιλήσουμε γι' αυτά παρακάτω) που συμμετέχουν, διασυνδέονται και συγκροτούν αυτή την αρχιτεκτονική του cSLAM.

Αυτή βέβαια η υλοποίηση παραπέμπει σε διαμορφώσεις πόλεων τύπου Robotarium και ξεφεύγουν από το συγκεκριμένο εγχείρημα. Θα μπορούσε όμως να αποτελέσει μία ενδιαφέρουσα επέκταση αυτού του έργου στο μέλλον.

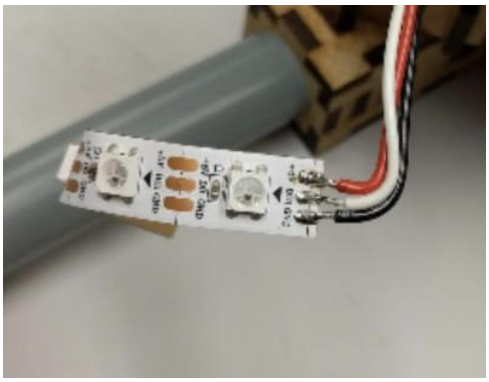
Εμείς με βάση το υλικό που είχαμε στη διάθεση μας μπορούμε να κατασκευάσομε το σύστημα των φωτεινών σηματοδοτών, τοποθετώντας επιπρόσθετα και μία κάμερα που είχαμε στη διάθεση μας [12]. Τα δομικά στοιχεία που είναι απαραίτητα για να συναρμολογηθούν και να λειτουργήσουν τα οι φωτεινοί σηματοδότες είναι τα εξής:

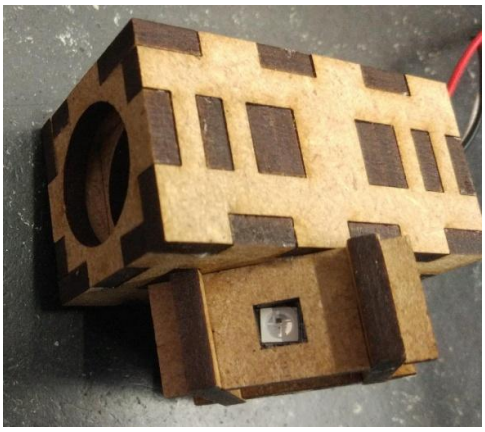
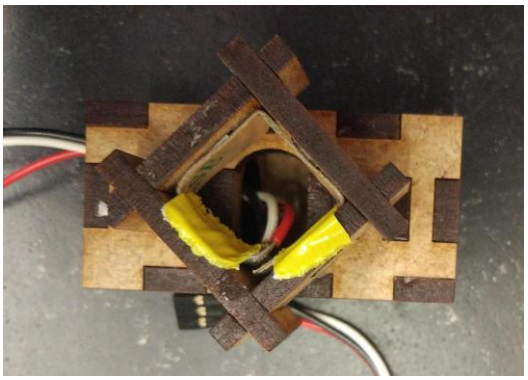
- ✓ Καλώδιο με συγκολλημένη λωρίδα led
- ✓ Raspberry Pi
- ✓ Raspberry Pi shield
- ✓ Καλώδιο usb
- ✓ SD κάρτα
- ✓ Καλώδιο Ethernet
- ✓ Κάμερα
- ✓ Καλωδιοταινία για την σύνδεση της κάμερας με το Raspberry Pi
- ✓ Βάση με το Raspberry Pi
- ✓ Πλαστικοί σωλήνες (1 μεγάλο, 2 μεσαίου μεγέθους και 1 μικρό)
- ✓ M2.5x10 MF πλαστικούς αποστάτες (x8)
- ✓ M2.5x8 πλαστικές βίδες (x4)
- ✓ Βάση συγκράτησης σωλήνα με μεγάλη έδρα
- ✓ Βάση συγκράτησης σωλήνα με μικρή έδρα
- ✓ Εξάρτημα σύνδεσης σωλήνων

Εικόνα 22: Δομικά υλικά Φωτεινών σηματοδοτών

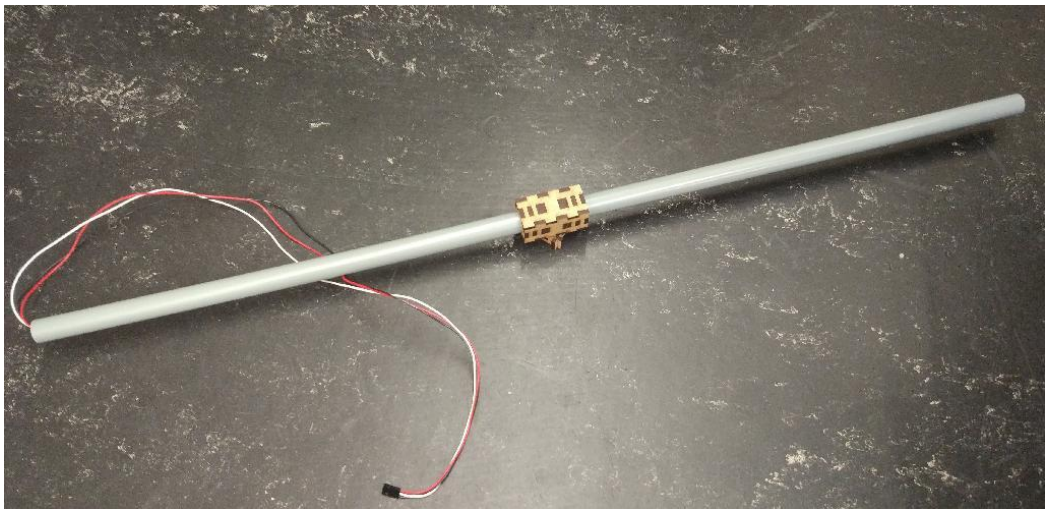
4. 1 Συναρμολόγηση σηματοδοτών

- ↳ Βήμα 1^{o} : Τοποθετούμε προσεκτικά την ταινία led μέσα στο ξύλινο περίβλημα προσέχοντας το εκτεθειμένο μέρος των καλωδίων τροφοδοσίας να μην έρχεται σε επαφή για μειώσουμε την πιθανότατα βραχυκυκλώματος

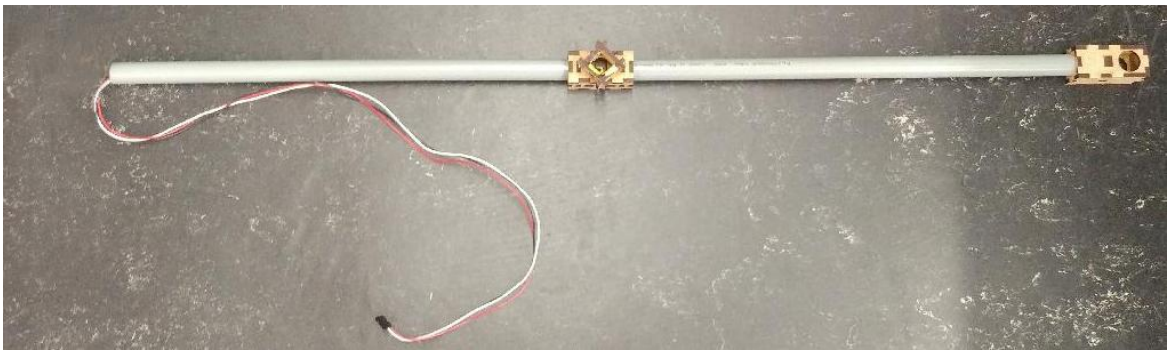


↳ Βήμα 2ο : Συνδέουμε τους δύο πλαστικούς σωλήνες εκατέρωθεν της ξύλινης κατασκευής μεριμνώντας να περάσουμε τα καλώδια τροφοδοσίας των led μέσα από τον ένα σωλήνα.

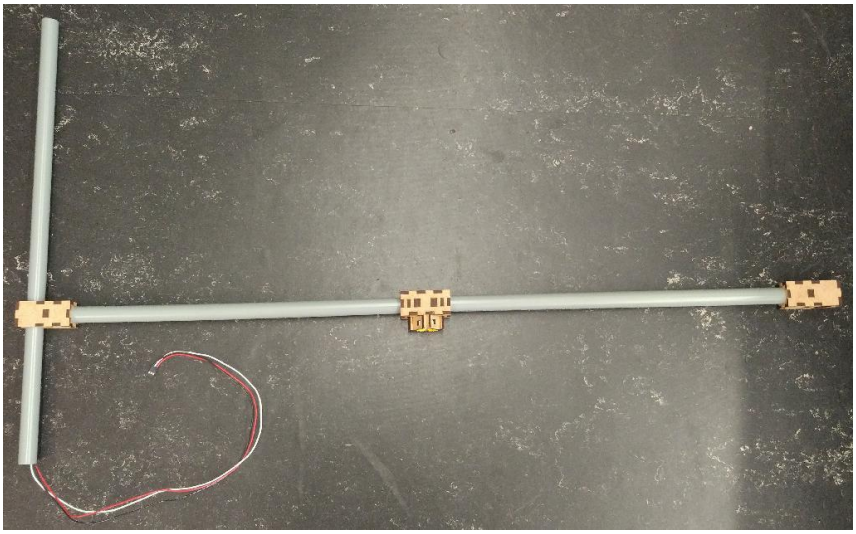


↳ Βήμα 3ο: Συνδέουμε την μία ελεύθερη άκρη του σωλήνα τα ένα ξύλινο συνδετικό εξάρτημα.



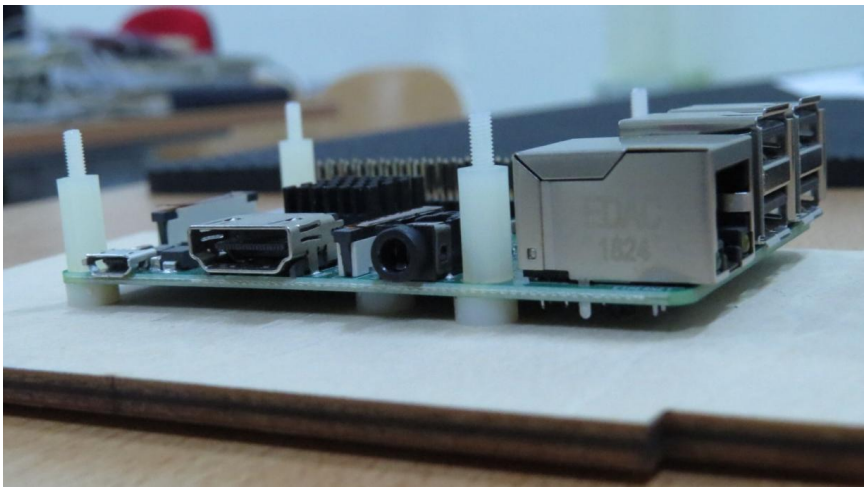
- Βήμα 4ο: Τοποθετούμε στον μεγάλο πλαστικό σωλήνα με την οπή το άλλο συνδετικό ξύλινο εξάρτημα προσέχοντας η οπή του σωλήνα να ευθυγραμμιστεί με την υποδοχή του ξύλινου συνδέσμου για μπορέσουν να περάσουν τα καλώδια τροφοδοσίας των led.

- Βήμα 5ο: Συνδέουμε τους δύο σωλήνες μεταξύ τους φροντίζοντας τα καλώδια τροφοδοσίας των led να περάσουν μέσα από τον σωλήνα και να βγουν στο κάτω μέρος του



- Βήμα 6ο: Τοποθετούμε το μικρό πλαστικό σωλήνα στον ξύλινο σύνδεσμο και στερεώνουμε όλη την κατασκευή στις δύο ξύλινες βάσεις που έχουν ήδη συναρμολογήσει.

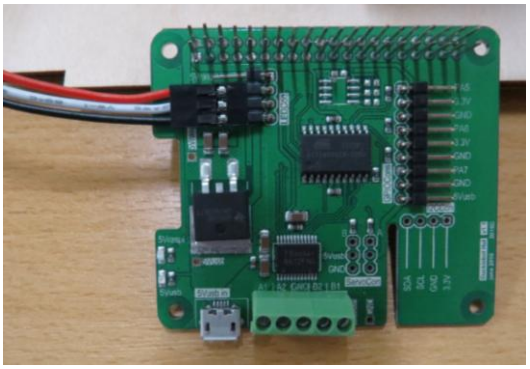
- Βήμα 7ο: Στερεώνουμε το Raspberry Pi πάνω στην ξύλινη έδρα χρησιμοποιώντας και τους αποστάτες ώστε να μην εφάπτεται πάνω σε αυτή.



- Βήμα 8ο : Συνδέουμε την καλωδιοταινίας της κάμερας στην ειδική υποδοχή (MPI CSI) του Raspberry Pi

- Βήμα 9ο : Τοποθετούμε το Raspberry Pi shield πάνω στο Raspberry Pi

- Βήμα 10ο: Συνδέουμε τα καλώδια τροφοδοσίας των Led στο Raspberry Pi shield



- Βήμα 11ο: Συνδέουμε τα usb καλώδια τροφοδοσίας
- Βήμα 12ο : Τοποθετούμε την Micro Sd card στο Raspberry Pi αφού πρώτα την έχουμε αρχικοποιήσει.

4.2 Αρχικοποίηση Κάρτας

Με τον όρο αρχικοποίηση της micro sd card (που τοποθετείται στο Raspberry Pi ως δευτερεύουσα μνήμη), περιγράφουμε τη διαδικασία εγκατάστασης όλου του απαραίτητου λογισμικού στην κάρτα (λειτουργικό συστήματος, Docker, βοηθητικών προγραμμάτων, κτλ) που το καταστούν λειτουργικό. Η αρχικοποίηση είναι μία απλή διαδικασία και γίνεται με τη βοήθεια του φορητού υπολογιστή αφού πρώτα τον διαμορφώσουμε κατάλληλα εγκαθιστώντας σε αυτόν μια σειρά από λογισμικά. Εκτενής αναφορά τόσο στον τρόπο εγκατάστασης όσο και στην χρησιμότητα των λογισμικών αυτών γίνεται στο Κεφάλαιο 7. Θεωρώντας δεδομένα τα όσα αναφέραμε πριν τοποθετούμε την Micro SD Card και έναν usb αναγνώστη καρτών τον οποίο και συνδέουμε στον φορητό υπολογιστή. Έπειτα ανοίγουμε ένα παράθυρο τερματικού και πληκτρολογούμε

$ dts init_sd_card --hostname trafficlights01 --country *GR* --type traffic_light --configuration TL18 --wifi RoboticsLab:xxxxxx

Αναλυτική περιγραφή της παραπάνω εντολής αρχικοποίησης γίνεται στο κεφάλαιο 7. Στο σημείο αυτό απλά θα αναφέρουμε ότι εδώ καθορίζουμε το αναγνωστικό των φωτεινών σηματοδοτών στο δίκτυο σε trafficlights01, το όνομα και τον κωδικό πρόσβασης του ασύρματου τοπικού δικτύου (RoboticsLab) και τον τύπο της αρχικοποίησης της κάρτας που επιθυμούμε (traffic_light).

Αν δεν θέλουμε να δημιουργήσουμε απλά ένα σύστημα φωτεινών σηματοδοτών αλλά ένα παρατηρητήριο τότε η εντολή αρχικοποίησης διαφοροποιείται ως εξής:

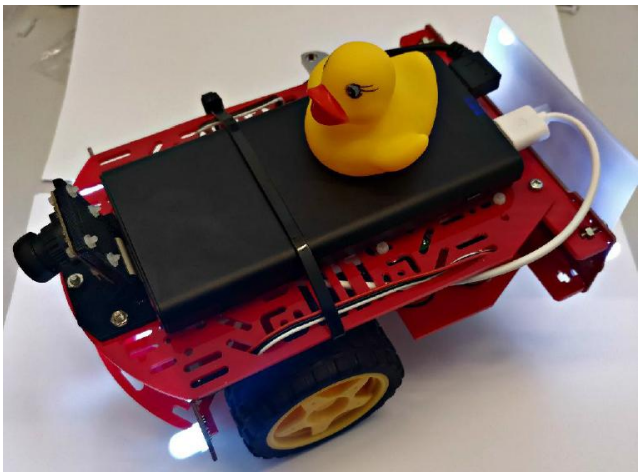
$ dts init_sd_card --hostname watchtower01 --country *GR* --type watchtower --configuration WT18 --wifi RoboticsLab:xxxxxx

Εικόνα 23: Παρατηρητήριο - Σηματοδότες

Κεφάλαιο 5

Κατασκευή του Duckiebot

Τα Duckiebots έχουν θεμελιώδη ρόλο στο project του Duckietown καθώς πρόκειται για τα αυτοκινούμενα ρομπότ τα οποία κινούνται μέσα στην πόλη. Στο κεφάλαιο αυτό θα περιγράψουμε τα υλικά εξαρτήματα από τα οποία αποτελούνται και το πως αυτά συνδέονται μεταξύ τους για να σχηματιστεί τελικά το Duckiebot.



Εικόνα 24: Duckiebot

Αρχικά να τονίσουμε ότι μπορούμε να κατασκευάσουμε το εικονιζόμενο Duckiebot συγκεντρώνοντας τα απαραίτητα υλικά που το απαρτίζουν από μόνοι μας αγοράζοντας τα από διάφορα διαδικτυακά καταστήματα ή μη συμβουλευόμενοι τον σχετικό πίνακα των απαιτούμενων υλικών που μας παρέχει το ηλεκτρονικό εγχειρίδιο The Duckiebot manual. Σε αυτή τη περίπτωση έχουμε αρκετές διαφορετικές διαμορφώσεις οι οποίες ονομάζονται γενικά DB17 (Duckiebot versions 2017). Στιγμιότυπο του πίνακα των συστατικών μερών του Duckiebot εμφανίζεται παρακάτω στο οποίο αναγράφονται και οι τιμές του κάθε υλικού σε δολάρια των Ηνωμένων Πολιτειών της Αμερικής.

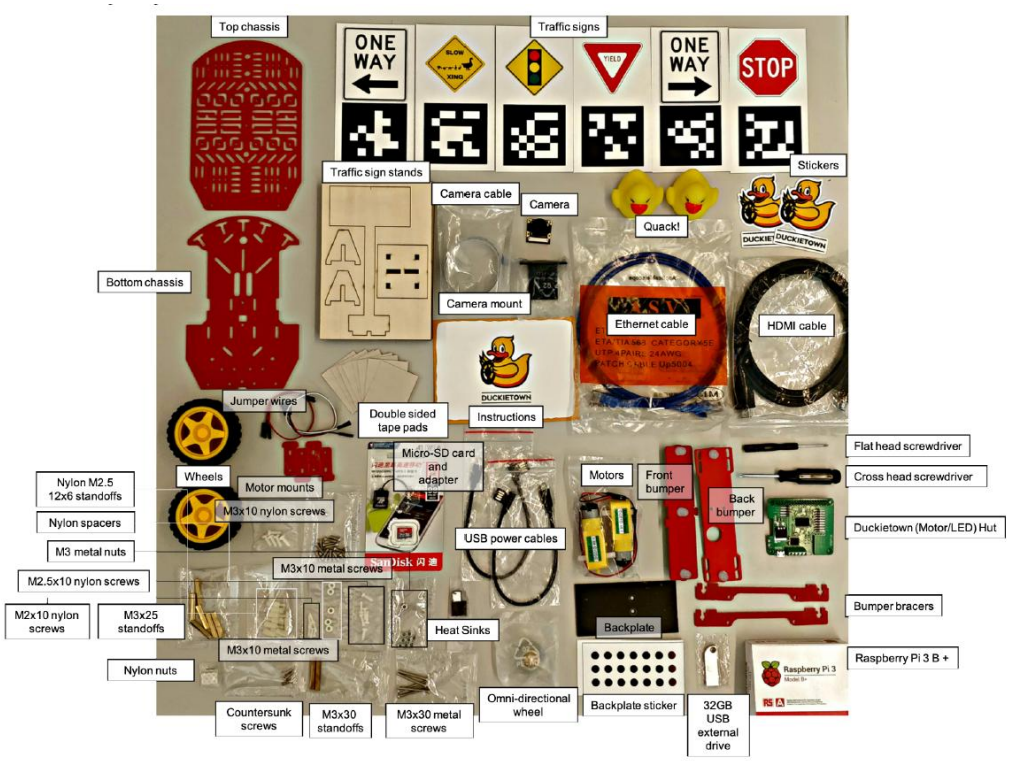
Chassis	USD 20
Camera with 160-FOV Fisheye Lens	USD 39
Camera Mount	USD 4
300mm Camera Cable	USD 2
Raspberry Pi 3 - Model B+	USD 39
Heat Sinks	USD 3
Power supply for Raspberry Pi	USD 7.50
16 GB Class 10 MicroSD Card	USD 10
Mirco SD card reader	USD 6
DC Motor HAT	USD 22.50
2 Stacking Headers	USD 2.50/piece
Battery Pack	USD 25
16 Nylon Standoffs (M2.5 12mm F 6mm M)	USD 0.06/piece
4 Nylon Hex Nuts (M2.5)	USD 0.02/piece
4 Nylon Screws (M2.5x10)	USD 0.05/piece
2 Zip Ties (300x5mm)	USD 9

Εικόνα 25 : Κοστολόγιο υλικών Duckiebot [51]

Στο πακέτο Classroom kit που είχαμε εμείς στη διάθεση μας περιλαμβάνει εκτός των άλλων και πέντε (5) χάρτινα κιβώτια όπου εκεί βρίσκονται συγκεντρωμένα όλα τα απαραίτητα υλικά εξαρτήματα για την κατασκευή του κάθε Duckiebot.

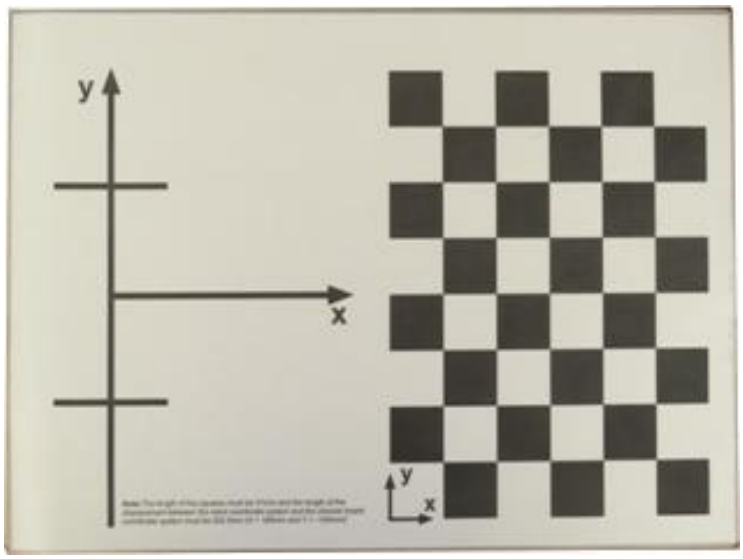
Εικόνα 26 : Κουτί συσκευασίας υλικών Duckiebot

Τα εξαρτήματα αυτά συνθέτουν το Duckiebot version 2018 (DT18) και έχει τις ίδιες λειτουργίες με ένα πλήρως εξοπλισμένο DB17-I Duckiebot. Συγκεκριμένα το παραπάνω πακέτο περιλαμβάνει τα υλικά της εικόνας που ακολουθεί



Εικόνα 27 : Περιεχόμενα συσκευασίας

Επιπλέον περιλαμβάνει και μία μπαταρία η οποία δεν απεικονίζεται στην παραπάνω φωτογραφία. Τέλος κάθε Duckiebot συνοδεύεται και από μία πινακίδα βαθμονόμησης της κάμερας.



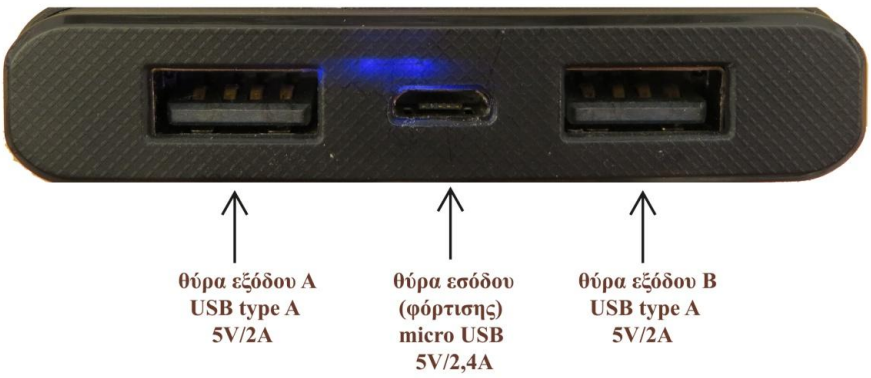
Εικόνα 28 μοτίβο βαθμονόμησης κάμερας

Εικόνα 29 : Μπαταρία

Θα ξεκινήσουμε κάνοντας μία σύντομη περιγραφή των υλικών που περιλαμβάνει το πακέτο DT18 και θα συνεχίσουμε με την συναρμολόγηση αυτών των εξαρτημάτων ώσπου τελικά να διαμορφωθεί το Duckiebot.

- ✓ Μπαταρία (The Duckie power-bank)

Η παροχή ενέργειας στο Duckiebot δίνεται από την μπαταρία (duckiebattery) που απεικονίζεται παραπάνω. Πρόκειται για μία μπαταρία με ονομασία A360 συνολικής χωρητικότητας 10000mAh/37Wh που παρέχει συνεχές ρεύμα τάσης 5Volt και έντασης 2Ampere διαμέσου δύο θυρών USB τύπου A που διαθέτει ενώ η φόρτιση της γίνεται διαμέσου της μιας θύρας micro USB που δέχεται ρεύμα τάσης 5 Volt και έντασης 2,4 Ampere.



Εικόνα 30 : Θύρες USB μπαταρίας

Επιπλέον διαθέτει και τέσσερα (4) Leds τα οποία μας ενημερώνουν για την κατάσταση φόρτισης της μπαταρίας ή αλλιώς για την διαθέσιμη ενέργεια που μπορεί να μας χορηγήσει. Πατώντας το κουμπί στο πλάι τα Leds ενεργοποιούνται με βάση τον παρακάτω πίνακα

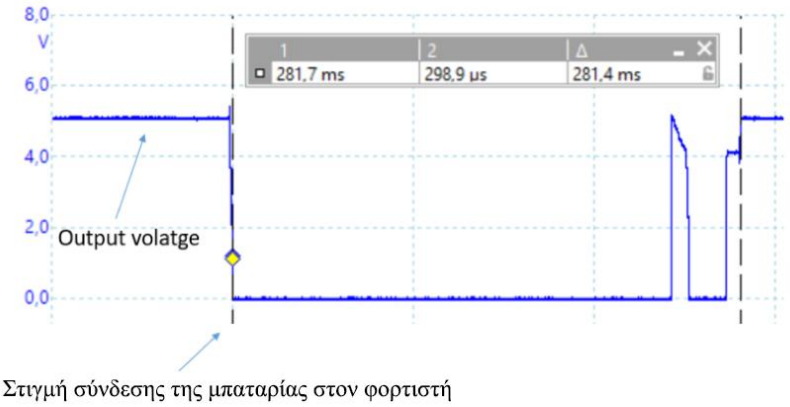
Ενδείξεις φόρτισης Led	
D1	3- 25 %
D2	25 - 50 %
D3	50 - 75 %
D4	75 - 100 %

Πίνακας 4 : Ενδείξεις φόρτισης

Έτσι όταν όλα τα Led είναι αναμμένα η μπαταρία είναι πλήρως φορτισμένη ενώ όταν ανάβουν μόνο τα δύο πρώτα (D1 και D2) περίπου φορτισμένη κατά το ήμισυ. Αν το πρώτο Led αναβοσβήνει (ενώ η μπαταρία δεν βρίσκεται σε διαδικασία φόρτισης) αυτό σημαίνει ότι η μπαταρία έχει "πέσει" κάτω το 3% και συνεπώς χρειάζεται φόρτιση.

Εικόνα 31 : Κουμπί ενεργοποίησης - Ενδείξεις LED

Η μπαταρία φορτίζεται μέσω της micro USB θύρας και κατά τη διάρκεια της φόρτισης αναβοσβήνει ένα από τα τέσσερα Led ενημερώνοντας μας για τον κύκλο φόρτισης στον οποίο βρίσκεται. Εδώ αξίζει να σημειώσουμε ότι όταν η μπαταρία συνδέεται στον φορτιστή η ισχύς των δύο θυρών εξόδου A και B διακόπτεται για περίπου 280ms. Το γεγονός αυτό είναι ικανό να μας προκαλέσει μία ανεπιθύμητη επανεκκίνηση του Raspberry Pi αν επιχειρούμε να φορτίσουμε την μπαταρία ενώ αυτή τροφοδοτεί με ενέργεια το υπολογιστή πλακέτας. Την ίδια ανεπιθύμητη "παρενέργεια" (επανεκκίνηση του Raspberry Pi) θα προκαλέσει και η αποσύνδεση του φορτιστή από την μπαταρία καθώς οι έξοδοι της τελευταίας θα απενεργοποιηθούν για 20ms.



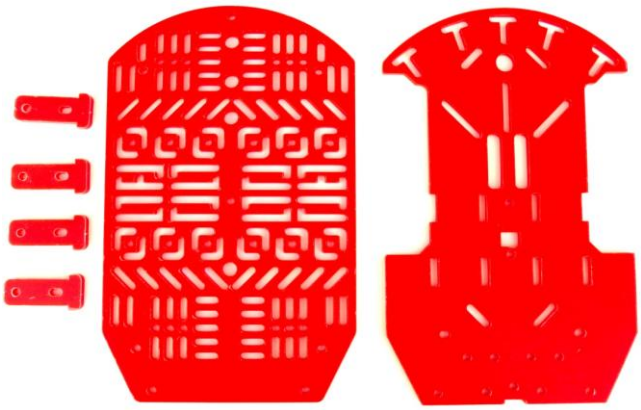
Επιπλέον ενώ η συγκεκριμένη μπαταρία υποστηρίζει την παροχή ενέργειας μέσω των δύο θυρών USB A και B κατά τη διαδικασία της φόρτισης η τάση του ρεύματος των δύο θυρών να πιθανό να "πέσει" κατά 300mV γεγονός το οποίο ενδέχεται να προκαλέσει δυσλειτουργίες στον υπολογιστή πλακέτας. Για όλους τους παραπάνω λόγους είναι μία καλή πρακτική να φορτίζουμε την μπαταρία A360 μόνο όταν αυτή δεν τροφοδοτεί με ενέργεια το Raspberry Pi και το duckieboard.

Τέλος να αναφέρουμε ότι η ενεργοποίηση των δύο USB θυρών εξόδου (παροχής ενέργειας) γίνεται αυτόματα όταν συνδεθεί κάποιο φορτίο ενώ μπορεί να γίνει και χειροκίνητα πατώντας το κουμπί. Η απενεργοποίηση των δύο θυρών A και B γίνεται πάλι αυτόματα όταν διαπιστωθεί ότι δεν υπάρχει η ανάγκη τροφοδότησης μιας συσκευής με ρεύμα έντασης μικρότερο των 100 mA. Το συνδυασμένο ρεύμα εξόδου περιορίζεται στα 2,8 Ampere. Η χωρητικότητα της μπαταρίας είναι 7,4 Ah στα 5 Volt με αποδόσεις που περιγράφονται στον παρακάτω πίνακα:

Ένταση φορτίου	Αποδοτικότητα	Αυτονομία
1 A	91%	6 ώρες και 44 λεπτά
1,5 A	88%	4 ώρες και 33 λεπτά
2 A	85%	3 ώρες και 9 λεπτά
2,5 A	79%	2 ώρες και 21 λεπτά

Πίνακας 5 : Αυτονομία μπαταρίας

- ✓ Σασί (Chassis)



Εικόνα 32 : Σασί Duckiebot

Τα Duckiebots χρησιμοποιούν το επονομαζόμενο Magician Chassis που είναι η τελευταία ρομποτική πλατφόρμα της Dagu. Πρόκειται για ακρυλικές πλάκες με μεγάλη ποικιλία οπών για τη στερέωση αισθητήρων, ελεγκτών, μονάδων παροχής ενέργειας, κτλ. Μπορούν να δεχτούν και δύο τροχούς των 65 mm καθώς και έναν τροχίσκο ή ένα caster όπως στην περίπτωση μας.

- ✓ Raspberry Pi 3 - Model B+

Εικόνα 33 : Υπολογιστής πλακέτας Raspberry Pi 3

Πρόκειται για έναν πλήρη υπολογιστή πλακέτας η οποία έχει το μέγεθος μιας πιστωτικής κάρτας περίπου και διαθέτει τις βασικές διασυνδέσεις ενός τυπικού υπολογιστή (θύρες usb, hdmi, audio out, Ethernet), ενσωματωμένο επεξεργαστή και επεξεργαστή γραφικών, μνήμη RAM, θύρα CSI κάμερας, DSI θύρα οθόνης, κάρτα μνήμης micro sd, καθώς και ακροδέκτες (GPIO) για την σύνδεση ψηφιακών αισθητήρων και συσκευών όπως φαίνεται και στην εικόνα 33.

Το μικρό του μέγεθος και οι δυνατότητες του το καθιστούν ιδανικό για ρομποτικές εφαρμογές. Στον παρακάτω πίνακα αναγράφονται τα βασικά χαρακτηριστικά του Raspberry Pi 3 B+ που φέρουν τα Duckiebots.

Για περισσότερες πληροφορίες τόσο για την ιστορία αυτού του υπολογιστή πλακέτας όσο και τους ακροδέκτες GPIO που διαθέτει θα βρείτε στο παράρτημα Α που ακολουθεί.

Το Raspberry Pi συνοδεύεται από μία κάρτα Micro SD χωρητικότητας 32 GB που αποτελεί το "σκληρό δίσκο" του υπολογιστή πλακέτας καθώς και ένα USB MicroSD Card Reader/Writer που θα μας βοηθήσει να εγκαταστήσουμε στην micro SD κάρτα το λειτουργικό σύστημα και τα απαιτούμενα λογισμικά από το φορητό υπολογιστή. Τέλος επειδή πρέπει να μεριμνήσουμε και για την ψύξη του επεξεργαστή του Raspberry Pi επειδή ο τελευταίος σίγουρα πρόκειται να θερμανθεί κατά τη χρήση του δεδομένου ότι τοποθετούμε και το DC motor hat πάνω στη πλακέτα του Raspberry το πακέτο περιλαμβάνει και ψήκτρες ώστε να τοποθετήσουμε την κατάλληλη πάνω στον επεξεργαστή.

Εικόνα 34 : Κάρτα Micro SD, Αναγνώστης, Ψήκτρες

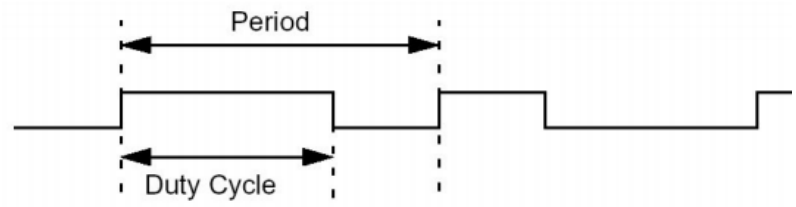
- ✓ Κάμερα (Camera)

Ο μοναδικός αισθητήρας που διαθέτουν τα Duckiebots για να αντιλαμβάνονται το εξωτερικό τους περιβάλλον είναι η κάμερα. Στην προκειμένη περίπτωση τα Duckiebots φέρουν την κάμερα Waveshare Raspberry Pi Camera Module Kit που είναι πλήρως συμβατή με κάθε έκδοση του Raspberry Pi. Η συγκεκριμένη κάμερα παρέχει ένα ευρύτερο οπτικό πεδίο με διαγώνιο 160 μοιρών (ενώ οι συνηθισμένες κάμερες έχουν συνήθως διαγώνιο 72 μοιρών), διαθέτει αισθητήρα OV5647 και μπορεί να συλλαμβάνει εικόνες με μέγιστη ανάλυση 5 megapixels ενώ στο βίντεο με ρυθμό μετάδοσης 30 fps η ανάλυση φτάνει τα 1080p. Είναι εφικτό φυσικά να επιτύχει και μεγαλύτερους ρυθμούς μετάδοσης με τίμημα την μικρότερη ανάλυση(960p:45fps, 720p:60fps,VGA(640x480):90fps, QVGA(320x240)120fps). Η κάμερα συνδέεται με το Raspberry Pi μέσω μια καλωδιοταινίας και στερεώνεται στο σασί του Duckiebot με μία ειδική πλαστική βάση που περιλαμβάνεται στο πακέτο.

Εικόνα 35 : Κάμερα- βάσης κάμερας - Καλωδιοταινίας σύνδεσης

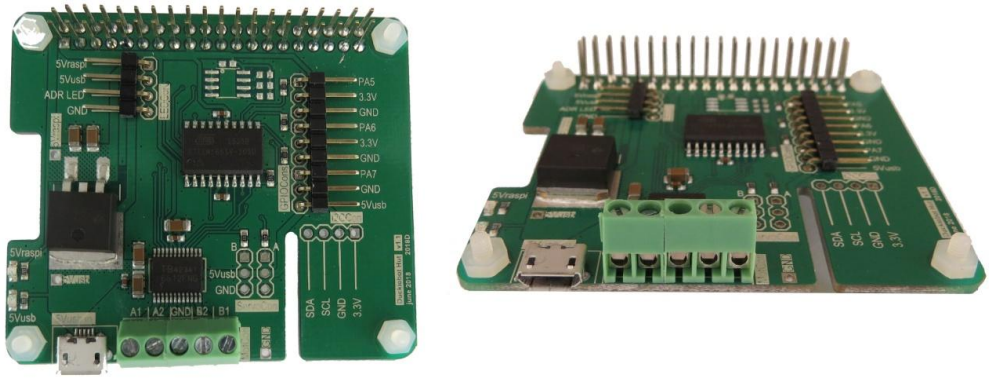
- ✓ DC Motor HAT

Για να ελέγξουμε τους δύο ενεργοποιητές του Duckiebot (που κινούν τους δύο τροχούς του) εφαρμόζουμε την τεχνική της διαμόρφωσης εύρους παλμών (PWM - Pulse Width Modulation). Μία PWM κυματομορφή είναι ουσιαστικά μία περιοδική κυματομορφή η οποία έχει δύο τμήματα. Το τμήμα On στο οποίο η κυματομορφή παίρνει την μέγιστη τιμή της και το τμήμα Off κατά τη διάρκεια του οποίου είναι μηδέν. Το τμήμα On ονομάζεται Duty Cycle και μετριέται είτε σε μονάδες χρόνου (ms,us, κτλ) είτε σε ποσοστό επί της περιόδου. Εφαρμόζοντας μία PWM κυματομορφή στην τροφοδοσία ενός φορτίου επιτυγχάνουμε να ελέγξουμε το ποσοστό της ισχύος που διοχετεύεται στο φορτίο. Έτσι αν το φορτίο είναι ένας κινητήρας αυτό πρακτικά συνεπάγεται έλεγχος των στροφών του κινητήρα.



Εικόνα 36 : Duty Cycle κυματομορφής PWM

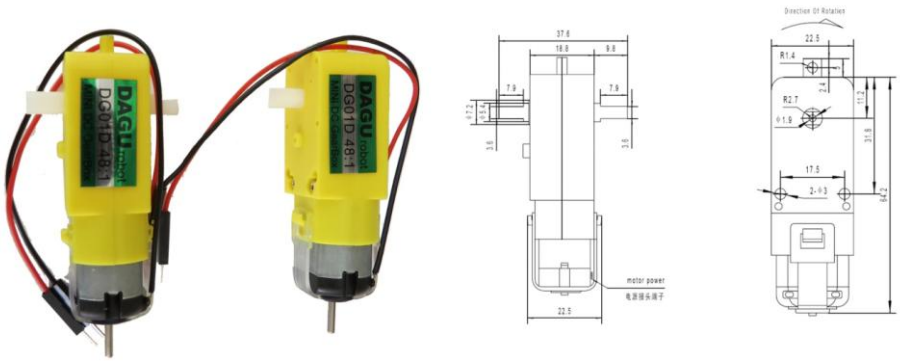
Δεδομένου ότι το Raspberry Pi δεν έχει πολλές ακίδες (pins) PWM χρησιμοποιούμε το Duckiebot hat για οδηγήσουμε τους κινητήρες καθώς αυτό διαθέτει ένα ενσωματωμένο τσιπ οδηγού PWM τόσο για τον έλεγχο της κατεύθυνσης όσο και της ταχύτητας του κινητήρα. Το DC motor Hat τοποθετείται όπως θα δούμε και μετέπειτα πάνω από το Raspberry Pi με βάση το πρωτόκολλο του σειριακό δίαυλου I^2C που χρησιμοποιείται για την σύνδεση περιφερειακών μικρής ταχύτητας σε μητρικές πλακέτες ή ενσωματωμένα συστήματα και διαχειρίζεται τα DC μοτέρ του Duckiebot.



Εικόνα 37 : DC motor Hat

- ✓ Ενεργοποιητές

Τα μοτέρ που κινούν τα Duckiebot έχουν την ονομασία DG01D-A130GearMotor της Dagu. Πρόκειται να δύο μοτέρ που δέχονται συνεχές ρεύμα τάσης 4,5 Volt και συνδέονται στο Duckiebot hat όπως θα δούμε παρακάτω το οποίο και τα οδηγεί μέσω των καλωδίων που φαίνονται στη εικόνα 38.



Εικόνα 38 : DC Κινητήρες τροχών

✓ Τροχοί

Δύο τροχοί διαμέτρου 65 mm (χιλιοστών) με πλάτος πέλματος 27 mm. Αποτελούνται εσωτερικά από πλαστικό ABS ενώ εξωτερικά έχουν καουτσούκ. Το συνολικό βάρος κάθε τροχού είναι 39 γραμμάρια. Διαθέτουν από την μέσα πλευρά μία οπή μήκους 5,3mm και πλάτους 3,66mm για την σύνδεση με το μοτέρ.

Το Duckiebot οδηγείται ελέγχοντας τους τροχούς που βρίσκονται στο εμπρός μέρος και συνδέονται με τους κινητήρες DC. Ωστόσο απαιτεί και παθητική υποστήριξη στο πίσω μέρος. Για το σκοπό αυτό τοποθετείται ένας μεταλλικός πανκατευθυντικός τροχός τύπου caster στο πίσω μέρος του Duckiebot

Εικόνα 39 : Τροχοί - Caster

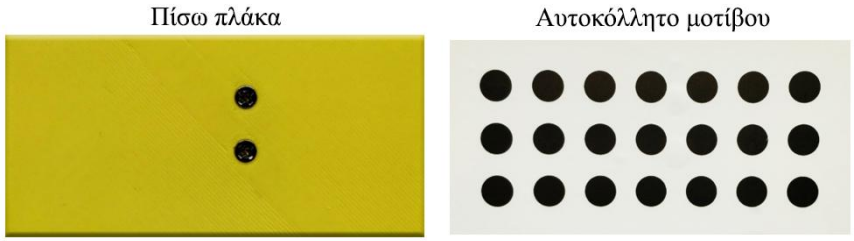
✓ Εμπρός και πίσω προφυλακτήρες

Το Duckiebot εξοπλίζεται με δύο προφυλακτήρες εμπρός και πίσω. Οι προφυλακτήρες αυτοί φέρουν και φωτοδιόδους. Ο εμπρός προφυλακτήρας φέρει τρία (3) Led ενώ ο πίσω δύο (2). Τα led αυτά τροφοδοτούνται με ρεύμα από το Duckiebot hut μέσω των καλωδίων που συνδέονται στο πίσω μέρος των προφυλακτήρων.

Εικόνα 40 : Προφυλακτήρες Duckiebot

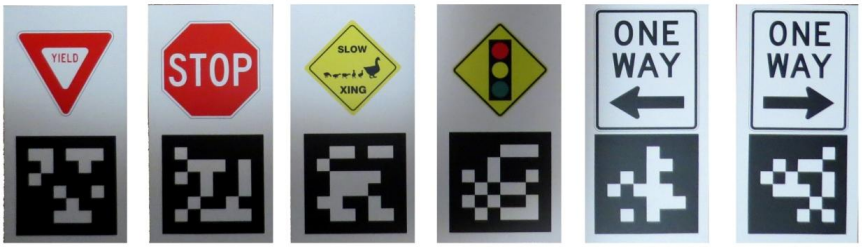
- ✓ Πίσω πλάκα και αυτοκόλλητο μοτίβο

Στον πίσω προφυλακτήρα του Duckiebot στερεώνεται μία πλάκα στην οποία επικολλάται ένα αυτοκόλλητο που απεικονίζει ένα μοτίβο με μαύρους κύκλους σε λευκό φόντο.



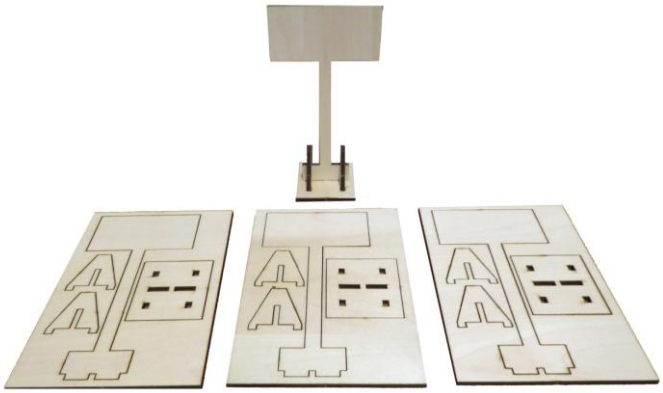
Εικόνα 41: Σύστημα αναγνώρισης Duckiebot

- ✓ Πινακίδες σήμανσης



Εικόνα 42 : Πινακίδες Σήμανσης

- ✓ Ξύλινοι ορθοστάτες



Εικόνα 43 : Κατασκευή ξύλινου ορθοστάτη

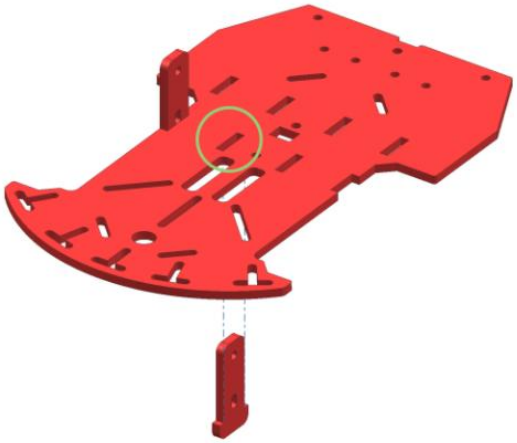
- ✓ Καλώδια

Στο πακέτο θα βρούμε καλώδια τριών διαφορετικών κατηγοριών. Ένα καλώδιο Ethernet για τη σύνδεση του Duckiebot με το laptop σε περίπτωση που δεν μπορεί το τελευταίο να συνδεθεί μέσω wifi στο τοπικό δίκτυο, Jumper wires για τη τροφοδοσία των led που φέρουν οι προφυλακτήρες του Duckiebot με το DC motor hat και τέλος USB καλώδια τροφοδοσίας τόσο για το Raspberry Pi όσο και το DC motor hat με τη μπαταρία που φέρει το Duckiebot.

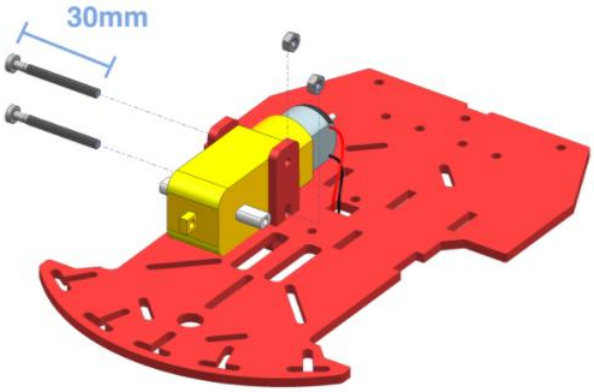
Συναρμολόγηση [15] του Duckiebot (DB18)

↳ Βήμα 1ο : Τοποθέτηση κινητήρων στο σασί

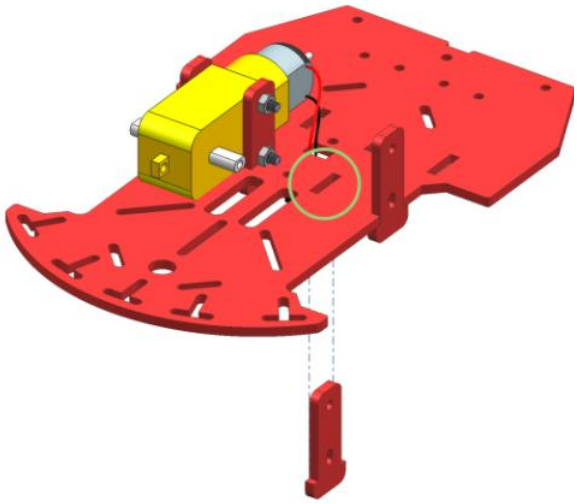
1. Τοποθέτηση βάσων πρώτου κινητήρα



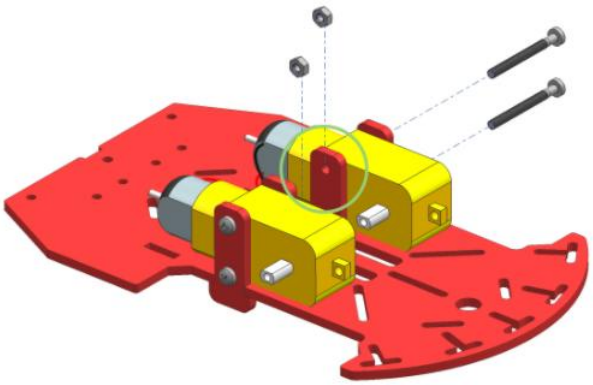
2. Στερέωση πρώτου κινητήρα



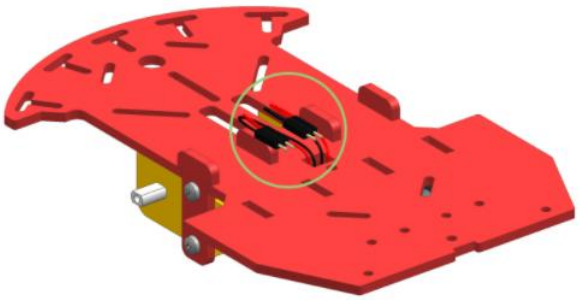
3. Τοποθέτηση βάσων δεύτερου κινητήρα



4. Στερέωση δεύτερου κινητήρα

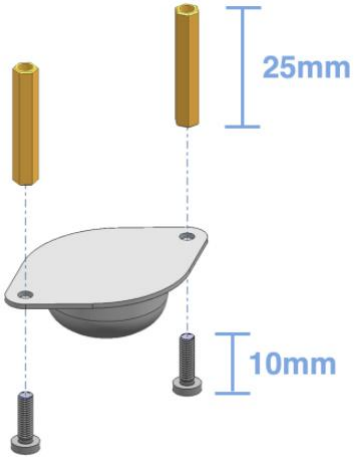


5. Διέλευση καλωδίων κινητήρων μέσα από την κεντρική οπή

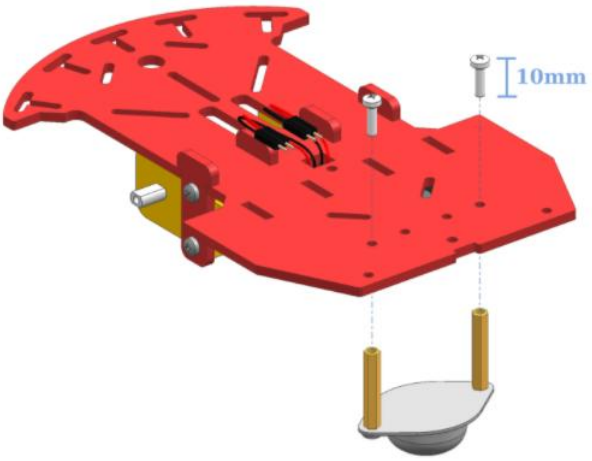


↳ Βήμα 2ο : Τοποθέτηση πανκατεθυντικού τροχού στο σασί

1. Στερέωση αποστατών στον τροχό

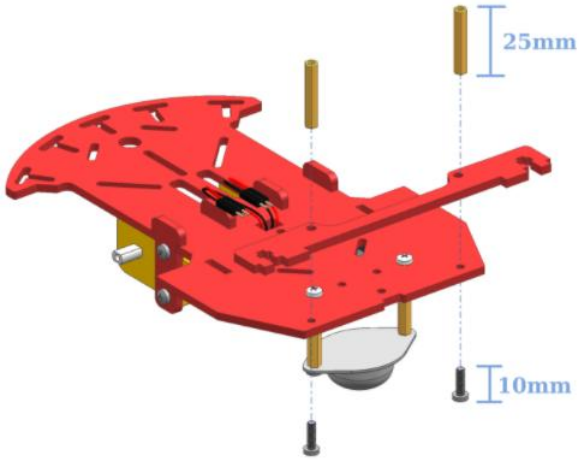


2. Τοποθέτηση του τροχού στο σασί

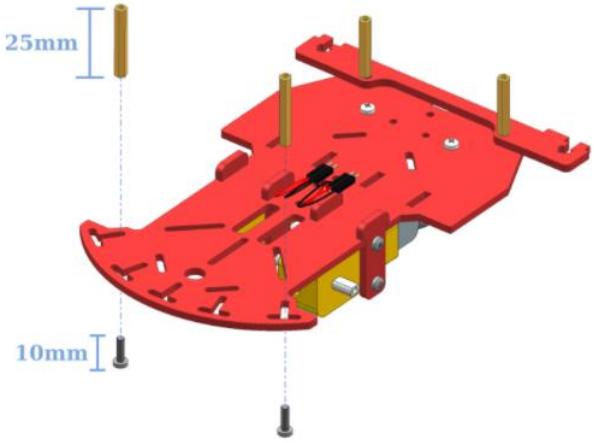


↳ Βήμα 3ο : Τοποθέτηση του πίσω βραχίονα και των αποστατών

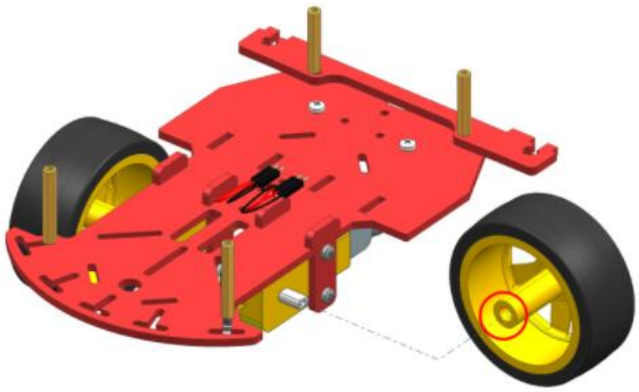
1. Στερέωση μεταλλικών διαχωριστικών και βραχίονα στο πίσω μέρος



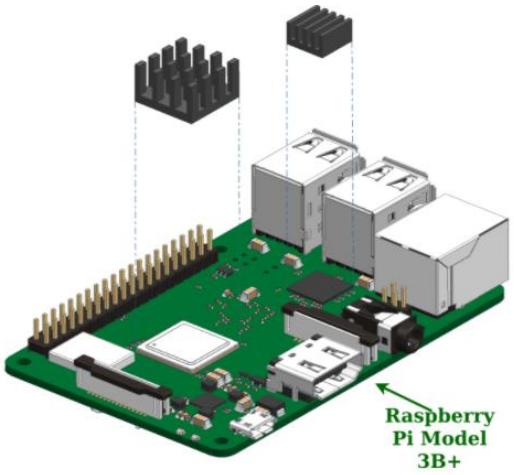
2. Τοποθέτηση μεταλλικών διαχωριστικών στο εμπρός μέρος



↳ Βήμα 4ο : Τοποθέτηση τροχών

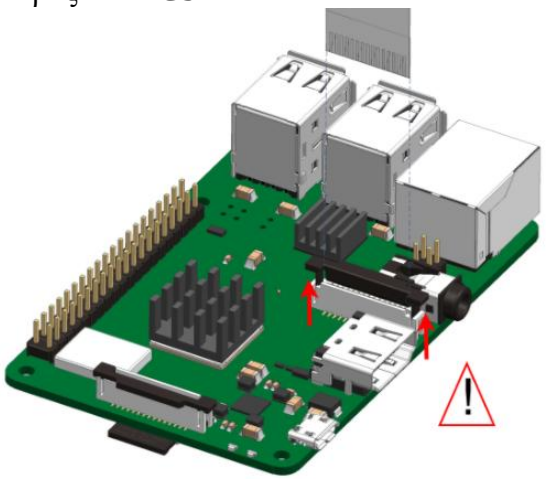


↳ Βήμα 5^{ο} : Τοποθέτηση ψηκτρών στο Raspberry Pi

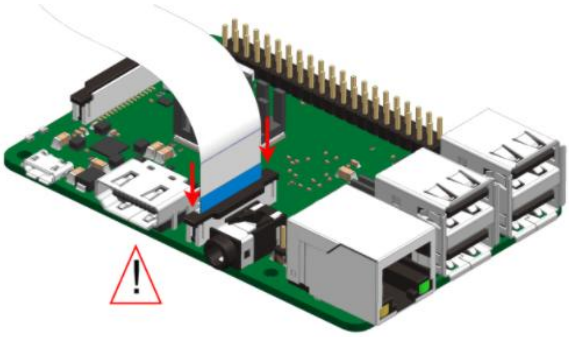


↳ Βήμα 6^{ο} : Τοποθέτηση καλωδιοταινίας κάμερας στο Raspberry Pi

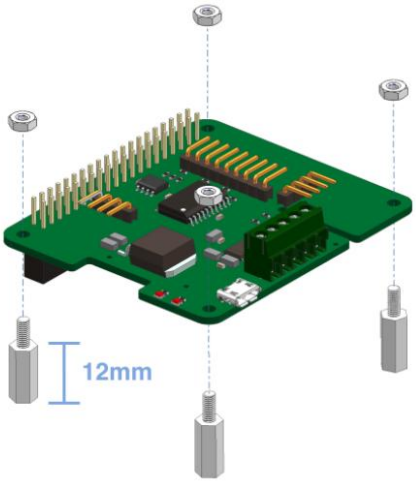
1. Ανύψωση πλαστικού καλύμματος της θύρας MPI CSI



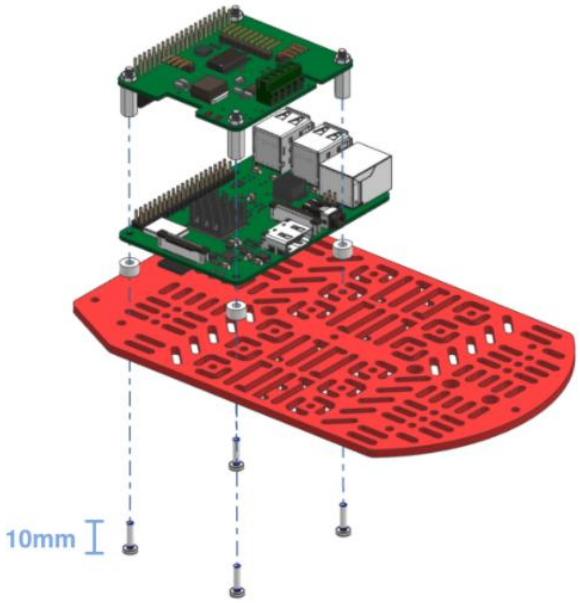
2. Τοποθέτηση καλωδιοταινίας και κατέβασμα καλύμματος της θύρας



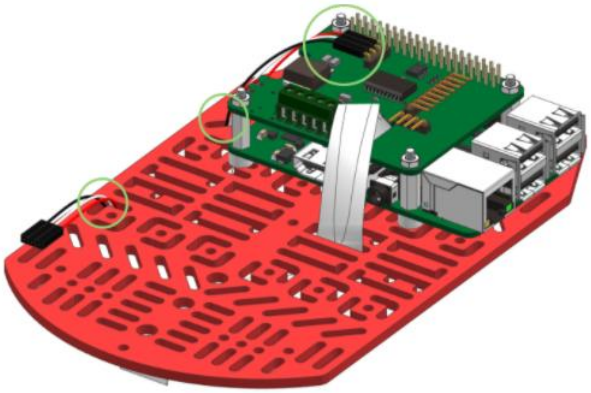
↳ Βήμα 7^{ο}: Προετοιμασία Motor Hut



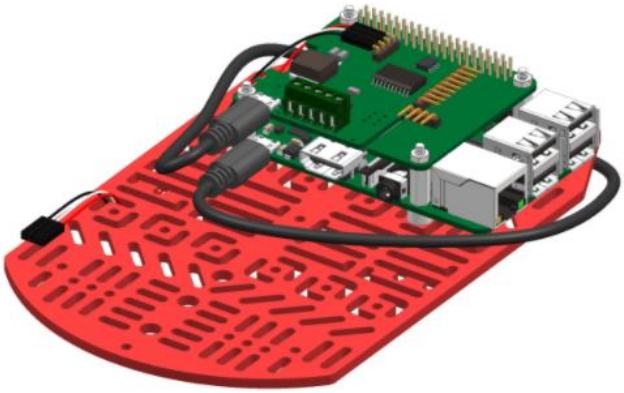
↳ Βήμα 8ο : Τοποθέτηση Duckietown Hut στο Raspberry Pi και στο σασί



↳ Βήμα 9ο : Σύνδεση jumper wires στα pins 5Vusb, ADR LED καιGND του Duckietown Hut

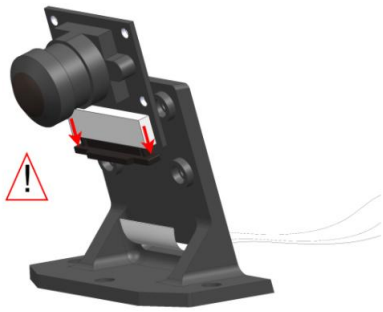


↳ Βήμα 10ο : Σύνδεση καλωδίων τροφοδοσίας



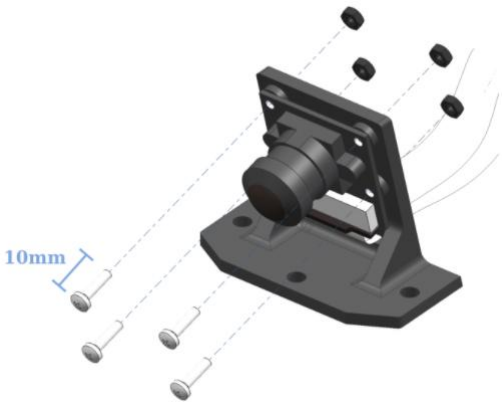
↳ Βήμα 11ο : Σύνδεση κάμερας με καλωδιοταινία και τοποθέτηση στη βάση

1. Άνοιγμα καλύμματος - σύνδεση καλωδιοταινίας

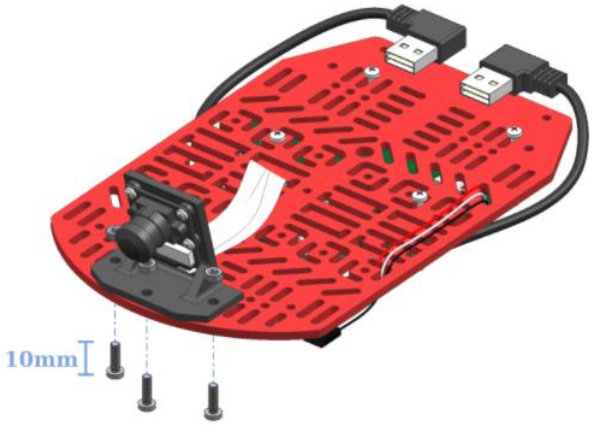


2. Κλείσιμο καλύμματος

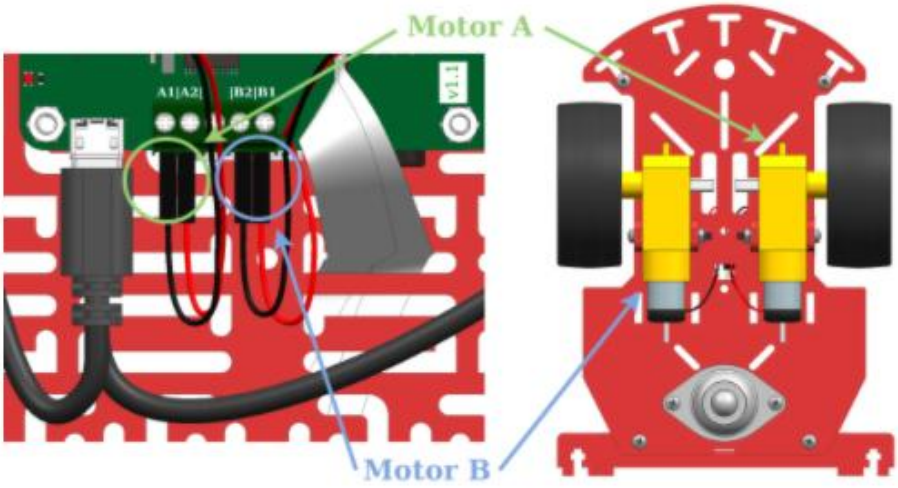
3. Στερέωση κάμερας στη βάση



2. Τοποθέτηση βάσης στο σασί

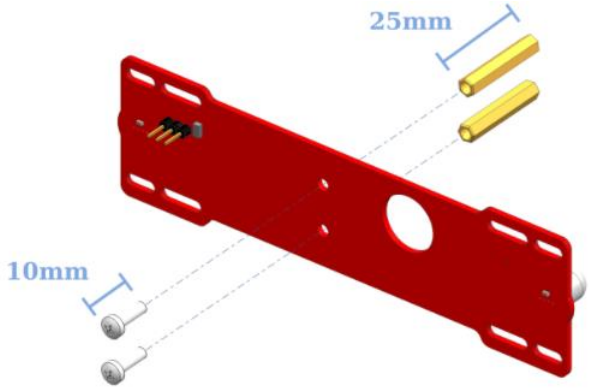


↳ Βήμα 12ο : Σύνδεση καλωδίων των κινητήρων με το DC motor Hut

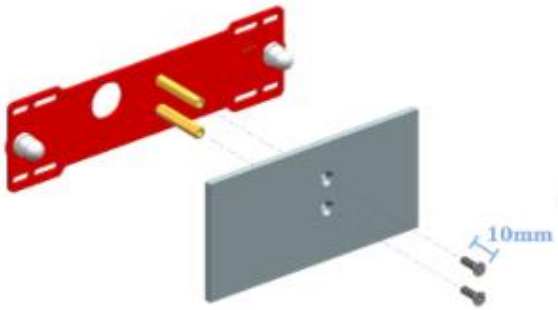


↳ Βήμα 13ο : Προετοιμασία πίσω προφυλακτήρα

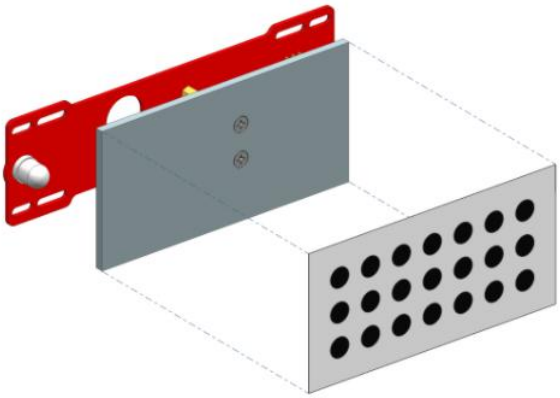
1. Τοποθέτηση μεταλλικών αποστατών



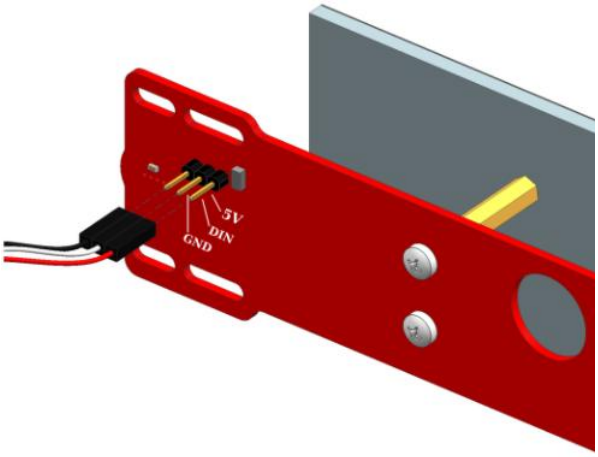
2. Σύνδεση της πλάκας στον πίσω προφυλακτήρα



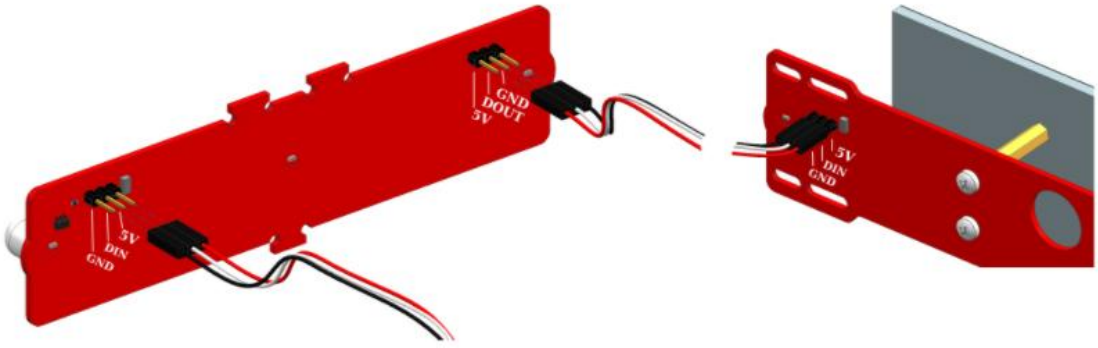
3. Επικόλληση αυτοκόλλητου μοτίβου



4. Σύνδεση jumper wire στα pins



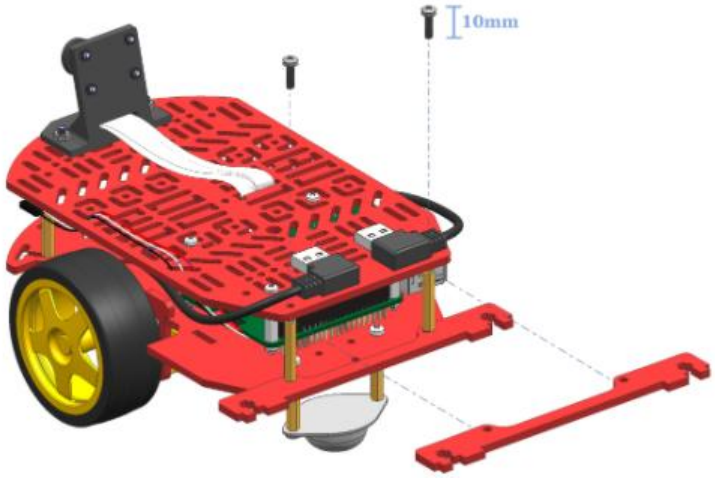
↳ Βήμα 14ο : Σύνδεση το εμπρός και του πίσω προφυλακτήρα μέσω jumper wires



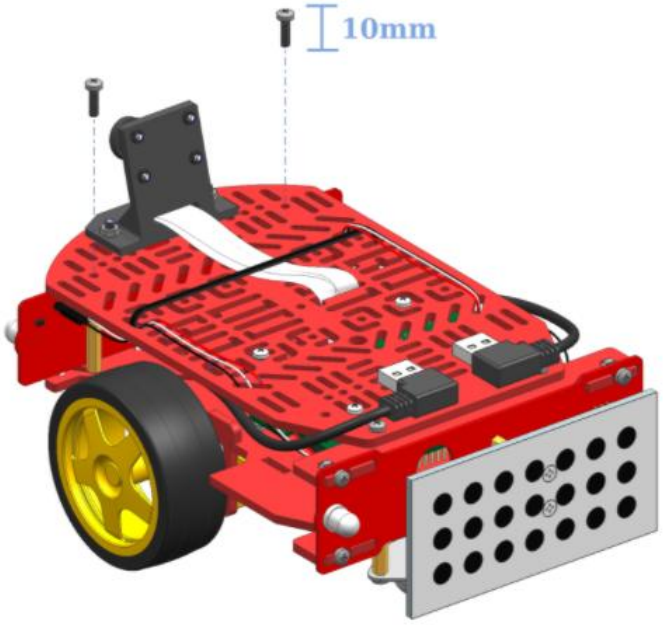
Προσοχή τα καλώδια με το ίδιο χρώμα να συνδέονται στα ίδια pins. Για παράδειγμα:

- κόκκινο --> 5 V
- μαύρο --> DOUT / DIN
- Λευκό --> GND

↳ Βήμα 15ο : Τοποθέτηση του δεύτερου πίσω βραχίονα προφυλακτήρα



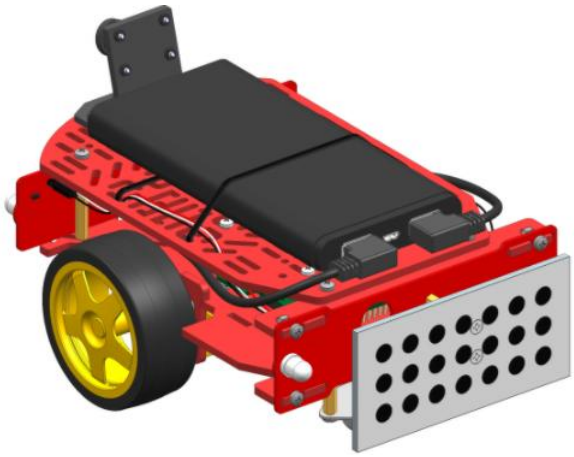
↳ Βήμα 16^{ο} : Τοποθέτηση και στερέωση της πάνω πλάκα του πλαισίου



↳ Βήμα 17^{ο} : Τοποθέτηση της MicroSD κάρτας στο Raspberry Pi

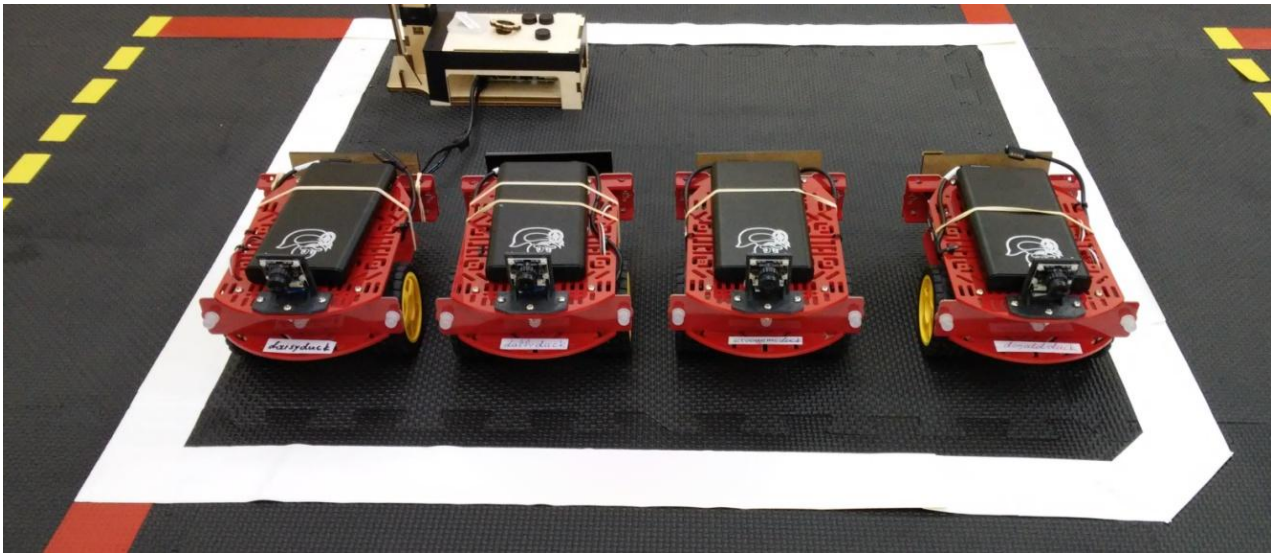
Προσοχή: Η ενέργεια αυτή προϋποθέτει την αρχικοποίηση της κάρτας MicroSD, μια διαδικασία που περιγράφεται στο επόμενο κεφάλαιο.

↳ Βήμα 18^{ο} : Τοποθέτηση μπαταρίας



Παρατήρηση: Καλό είναι την πρώτη φορά που θα συνδεθεί η μπαταρία στο Duckiebot να είναι πλήρως φορτισμένη διότι θα εκκινήσει η διαδικασία αρχικοποίησης του Duckiebot η οποία απαιτεί αρκετό χρόνο και δεν πρέπει να διακοπεί απρόσμενα.

Βήμα 19ο : Τοποθέτηση της πάπιας πάνω στο Duckiebot και είναι έτοιμο!



Εικόνα 44 : Τα τέσσερα Duckiebots που κατασκευάσαμε

Κεφάλαιο 6

Εγκατάσταση λογισμικού

Παράλληλα ή μετά τη συναρμολόγηση του Duckiebot θα πρέπει να εγκαταστήσουμε ένα σύνολο απαραίτητων για τη λειτουργία και τον χειρισμό του λογισμικών στο laptop [16]. Θα ξεκινήσουμε λοιπόν κάνοντας μια αναφορά στα λογισμικά αυτά πριν περάσουμε περιγράφοντας εν συντομία τον τρόπο εγκατάστασής τους και αναδεικνύοντας τη χρησιμότητα του κάθε λογισμικού. Το λειτουργικό περιβάλλον του Duckietown project περιλαμβάνει πέρα από το λειτουργικό σύστημα που θεωρείται προαπαιτούμενο τις εξής πλατφόρμες οι οποίες και πρέπει να εγκατασταθούν στο φορητό υπολογιστικό σύστημα μας :

- Το pip για Python 3
- Το Git και το Git LFS
- Το curl
- Το Docker
- Το Duckietown shell

Το πρώτο βασικό λογισμικό που πρέπει να εγκαταστήσουμε σε ένα οποιοδήποτε υπολογιστικό σύστημα είναι το λειτουργικό σύστημα. Το λειτουργικό σύστημα είναι απαραίτητο για την αποδοτική λειτουργία του υπολογιστή μας καθώς αναγνωρίζει και διαχειρίζεται τους πόρους του συστήματος (ΚΜΕ, κύρια μνήμη, βοηθητική μνήμη, συσκευές εισόδου/εξόδου, κτλ) και παρέχει ένα φιλικό γραφικό περιβάλλον διεπαφής (Graphical User Interface) στον χρήστη για να μπορεί να δίνει εντολές και να χειρίζεται τον υπολογιστή ενεργοποιώντας προγράμματα και εφαρμογές ενώ φροντίζει και για την απρόσκοπτη εκτέλεση αυτών των προγραμμάτων υποστηρίζοντας διαδικασίες πολυπρογραμματισμού, εικονικής μνήμης, κτλ.

Στην περίπτωση μας το λειτουργικό σύστημα που ενδείκνυται ανήκει σε μια ομάδα λειτουργικών συστημάτων ελεύθερου λογισμικού και ανοικτού κώδικα που είναι κατασκευασμένα γύρω από τον πυρήνα του Linux που κυκλοφόρησε για πρώτη φορά το 1991. Συνήθως το Linux είναι "συσκευασμένο" σε διανομές με ονομασίες όπως Fedora ή Ubuntu. Εμείς για τις ανάγκες του project θα χρησιμοποιήσουμε τη διανομή Ubuntu. Αν και η πιο πρόσφατη διανομή είναι η Ubuntu 18.04 LTS (Long Term Service) που θα υποστηρίζεται μέχρι τον Απρίλιο του 2023 προτείνεται να γίνει χρήση της διανομής Ubuntu 16.04 LTS αν και η υποστήριξη της τελειώνει τον Απρίλιο του 2021.

6. 1Διαδικασία Εγκατάστασης Ubuntu

Η διαδικασία εγκατάστασης λειτουργικού συστήματος σε έναν υπολογιστή είναι μια απλή σχετικά διαδικασία που όμως μπορεί να γίνει επικίνδυνη αν την επιχειρήσουμε σε έναν υπολογιστή ο οποίος έχει λειτουργικό, δεδομένα και προγράμματα καθώς αν δεν προσέξουμε μπορεί να διαγραφούν! Ως εκ τούτου μία ενέργεια που μπορεί να αποδειχθεί "σοφή" είναι να

πάρουμε αντίγραφα ασφαλείας όλου του ψηφιακού υλικού που υπάρχει στον υπολογιστή και δεν θα θέλαμε να διαγραφεί. Υπάρχει δυνατότητα εγκατάστασης δύο διαφορετικών λειτουργικών συστημάτων σε έναν υπολογιστή ακόμα κι αν αυτός διαθέτει ένα μόνο σκληρό δίσκο αρκεί να έχουμε δημιουργήσει πριν ξεχωριστό πρωτεύον διαμέρισμα σ' αυτόν. Σ' αυτή τη περίπτωση κατά στην εκκίνηση θα εμφανίζεται ένα μενού επιλογής από το οποίο ο χρήστης θα επιλέγει το λειτουργικό με το οποίο θα εκκινήσει ο υπολογιστής του. Τέλος μπορούμε να αποφύγουμε τη διαδικασία εγκατάστασης "φορτώνοντας" τα Ubuntu με χρήση εικονικής μηχανής επιλογή όμως που δεν προτείνεται για το συγκεκριμένο project. Εμείς θα περιγράψουμε τη διαδικασία εγκατάστασης του λειτουργικού σε Laptop free dos, χωρίς λειτουργικό δηλαδή όπου τα πράγματα είναι σαφώς απλούστερα.

Τα βήματα της εγκατάστασης είναι τα ακόλουθα:

1. Αρχικά πρέπει να κατεβάσουμε από την ιστοσελίδα http://releases.ubuntu.com/ το αρχείο εικόνας (.iso) που περιέχει την διανομή που επιθυμούμε να εγκαταστήσουμε και να το αποθηκεύσουμε στον υπολογιστή μας.
2. Το επόμενο βήμα είναι να δημιουργήσουμε ένα usb flash drive εκκίνησης (bootable) στο οποίο θα αντιγράψουμε και το αρχείο iso που "κατεβάσαμε" στο πρώτο βήμα. Η διαδικασία αυτή μπορεί να γίνει με το πρόγραμμα Rufus (https://rufus.ie/) αν εργαζόμαστε σε υπολογιστή με λειτουργικό σύστημα Windows.
3. Συνδέουμε το usb flash drive στο laptop και επανεκκινούμε το laptop. Αν διαπιστώσουμε ότι η διαδικασία δεν πραγματοποιείται από το usb flash drive κάνουμε εκ νέου επανεκκίνηση, μπαίνουμε στο BIOS (πιέζοντας το κατάλληλο πλήκτρο) και αλλάζουμε τη σειρά των συσκευών που ψάχνει για λειτουργικό ο υπολογιστής μας.

Σε περίπτωση που δεν επιθυμούμε να εγκαταστήσουμε δεύτερο λειτουργικό σύστημα σε υπολογιστή στον οποίο υπάρχει εγκατεστημένο ήδη κάποιο λειτουργικό μπορούμε να εγκαταστήσουμε μία εικονική μηχανή όπως για παράδειγμα το Virtual Box μέσω αυτής της εφαρμογής να φορτώνουμε και να εκτελείται το αρχείο iso που περιέχει τη διανομή Ubuntu που θέλουμε. Υπενθυμίζουμε όμως ότι αυτή η επιλογή δεν προτείνεται για την υλοποίηση του Duckietown και σ' αυτή τη περίπτωση θα πρέπει να μεριμνήσουμε για την υποδικτύωση με το Duckiebot.

Μετά την εγκατάσταση του λειτουργικού συστήματος ακλουθεί η εγκατάσταση ενός πλήθους λογισμικών στο laptop τα οποία και περιγράφουμε παρακάτω:

6. 2 Εγκατάσταση Pip για Python 3

Ως ένα δημοφιλές έργο ανάπτυξης ανοιχτού κώδικα, η Python έχει μια ενεργή υποστηρικτική κοινότητα συντελεστών και χρηστών που κάνουν επίσης το λογισμικό τους διαθέσιμο για χρήση από άλλους προγραμματιστές Python υπό όρους άδειας ανοιχτού κώδικα. Αυτό επιτρέπει στους χρήστες της Python να μοιράζονται και να συνεργάζονται αποτελεσματικά, επωφελούμενοι από λύσεις που έχουν ήδη δημιουργήσει άλλοι προγραμματιστές σε κοινά

προβλήματα, καθώς και ενδεχομένως να συνεισφέρουν τις δικές τους λύσεις στην κοινή ομάδα.

Το pip είναι το προτιμώμενο πρόγραμμα εγκατάστασης. Πρόκειται για ένα εικονικό ημι-απομονωμένο περιβάλλον Python που επιτρέπει την εγκατάσταση πακέτων για χρήση από μια συγκεκριμένη εφαρμογή, αντί να είναι εγκατεστημένο σε όλο το σύστημα. Απλοποιεί τη διαδικασία εγκατάστασης και διαχείρισης πακέτων λογισμικού (packages) όπως αυτά που βρίσκονται στο Python Package Index (PyPI) που είναι ένα αποθετήριο κώδικα της Python. Για εγκαταστήσουμε το προτιμώμενο πρόγραμμα εγκατάστασης της python πληκτρολογούμε στο τερματικό

$ sudo apt install python3-pip

Όταν ολοκληρωθεί η διαδικασία εγκατάστασης μπορούμε να επιβεβαιωθούμε ελέγχοντας την έκδοση (version) γράφοντας

$ pip3 --version

```
sakis@SakisLaptop ~
> $ pip3 --version
pip 9.0.1 from /usr/lib/python3/dist-packages (python 3.6)
```

Είναι καλή πρακτική πριν την παραπάνω διαδικασία να ενημερώσουμε τη λίστα των πακέτων με την εντολή

$ sudo apt update

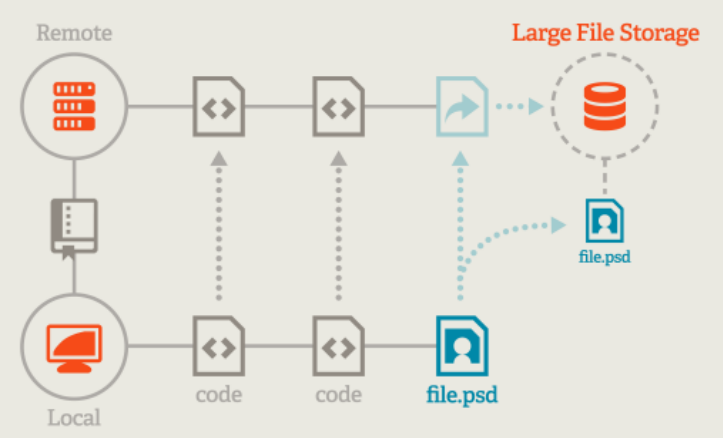
6.3 Εγκατάσταση αποθετηρίου στο GitHub

Το GitHub είναι μια διαδικτυακή πλατφόρμα συνεργασίας και ταυτόχρονα ένα αποθετήριο κώδικα των χρηστών - μελών που είναι εγγεγραμμένοι σε αυτό. Παρέχει τη δυνατότητα της συνεργασίας πάνω κυρίως σε προγραμματιστικά projects στα διάφορα μέλη του ανά τον κόσμο. Τα βασικά δομικά στοιχεία τα οποία βασίζεται η λειτουργία του GitHub είναι τα αποθετήρια (repositories), τα υποκαταστήματα (branches), οι δεσμεύσεις (commits) και τα αιτήματα έλξης (pull request).

Για εγκαταστήσουμε το βασικό Git πρόγραμμα στο laptop μαζί το πακέτο LFS χρησιμοποιούμε την εντολή:

$ sudo apt install git git-lfs

Το πακέτο LFS (Large File Storage) αντικαθιστά αρχεία μεγάλου μεγέθους όπως αρχεία βίντεο, γραφικών, datasets, κτλ με δείκτες κειμένου μέσα στο Git ενώ αποθηκεύει τα περιεχόμενα του αρχείου σε έναν απομακρυσμένο διακομιστή όπως το GitHub.com ή το GitHub Enterprise.



Εικόνα 45 : Διάγραμμα λειτουργίας του Git LFS

Για να εγκαταστήσουμε πρόσθετα βοηθητικά προγράμματα πληκτρολογούμε την εντολή

sudo apt install git-extras

Η επόμενη μας κίνηση είναι η δημιουργία του λογαριασμού σε αυτό δίνοντας ως στοιχεία ταυτοποίησης το mail και το όνομα χρήστη με τις παρακάτω εντολές

$ git config --global user.email "email"

$ git config --global user.name "full name"

Οι παραπάνω πρέπει να εκτελεστούν τόσο στο laptop όσο αργότερα και το Duckiebot.

Οι βασικές ενέργειες που λαμβάνουν χώρα στο αποθετήριο GitHub περιγράφονται στο Παράρτημα Β της παρούσας τεκμηρίωσης.

6. 4 Εγκατάσταση curl

Το curl είναι ένα βοηθητικό πρόγραμμα γραμμής εντολών για τη μεταφορά δεδομένων από ή σε έναν διακομιστή που έχει σχεδιαστεί για να λειτουργεί χωρίς αλληλεπίδραση του χρήστη. Με το curl, μπορούμε να κατεβάσουμε ή να ανεβάσουμε δεδομένα χρησιμοποιώντας ένα από τα υποστηριζόμενα πρωτόκολλα, συμπεριλαμβανομένων των HTTP, HTTPS, SCP, SFTP και FTP. Μας παρέχει μια σειρά από επιλογές για τη διαμεταγωγή αρχείων, το εύρος ζώνης, την υποστήριξη διακομιστή μεσολάβησης, τον έλεγχο ταυτότητας χρήστη και πολλά άλλα. Για να εγκαταστήσουμε αυτό το βοηθητικό πρόγραμμα πληκτρολογούμε

$ sudo apt install curl

ενώ επιβεβαιώνουμε την έκδοση του με την εντολή

$ curl --version

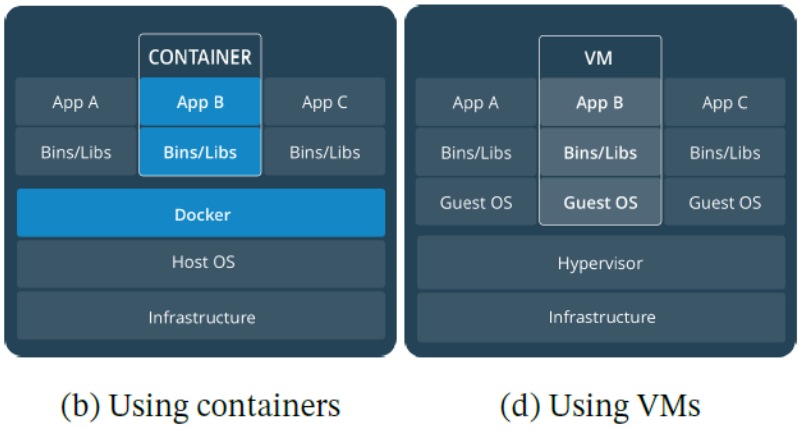
```
sakis@SakisLaptop ~                                                                                    [11:18:00]
> $ curl --version
curl 7.58.0 (x86_64-pc-linux-gnu) libcurl/7.58.0 OpenSSL/1.1.1 zlib/1.2.11 libidn2/2.0.4 libpsl/0.19.1 (+libidn2/2.0.4) nghttp2/1.30.0 librtmp
/2.3
Release-Date: 2018-01-24
Protocols: dict file ftp ftps gopher http https imap imaps ldap ldaps pop3 pop3s rtmp rtsp smb smbs smtp smtps telnet tftp
Features: AsynchDNS IDN IPv6 Largefile GSS-API Kerberos SPNEGO NTLM NTLM_WB SSL libz TLS-SRP HTTP2 UnixSockets HTTPS-proxy PSL
```

6.5 Περιγραφή του Docker

Το Docker είναι μια ανοιχτή πλατφόρμα για την ανάπτυξη, αποστολή και εκτέλεση εφαρμογών. Μας επιτρέπει να διαχωρίσουμε τις εφαρμογές από την υποδομή μας, να διαχειριστούμε το υλικό με τους ίδιους τρόπους όπως διαχειριζόμαστε τις εφαρμογές και αξιοποιώντας τις μεθοδολογίες που να παρέχει η πλατφόρμα του Docker να μειώσουμε σημαντικά τον χρόνο ανάμεσα στη συγγραφή κώδικα και στην εκτέλεση του.

Το Docker παρέχει τη δυνατότητα να συσκευάσουμε και να εκτελέσουμε μια εφαρμογή σε ένα απομονωμένο περιβάλλον που ονομάζεται container (κιβώτιο - δοχείο). Η απομόνωση και η ασφάλεια μας επιτρέπουν να τρέχουμε ταυτόχρονα πολλά κοντέινερ σε έναν συγκεκριμένο κεντρικό υπολογιστή. Η διαδικασία αυτή ονομάζεται containerization. Το containerization είναι μία ελαφριά εναλλακτική λύση για την πλήρη εικονοποίηση μηχανών που περιλαμβάνει την ενθυλάκωση μιας εφαρμογής σε ένα εικονικό δοχείο-κιβώτιο (container) μαζί με το περιβάλλον λειτουργίας της. Αυτό παρέχει πολλά πλεονεκτήματα της φόρτωσης μιας εφαρμογής και στην εκτέλεση της καθώς αυτή μπορεί να τρέξει σε οποιαδήποτε κατάλληλη φυσική μηχανή χωρίς να ανησυχεί για κάθε είδους "εξαρτήσεις". Έτσι η διαδικασία αυτή επιτρέπει την κατανομή του υλικού και του πυρήνα ενός λειτουργικού συστήματος με τέτοιο τρόπο ώστε διαφορετικά κιβώτια (container) να μπορούν να συνυπάρχουν στο ίδιο σύστημα ανεξάρτητα το ένα από το άλλο. Τα προγράμματα που εκτελούνται σε ένα τέτοιο container έχουν πρόσβαση μόνο στους πόρους του συστήματος που απαιτούνται για την εκτέλεση τους και είναι απόλυτα ανεξάρτητα από τις βιβλιοθήκες και τις διαμορφώσεις των άλλων δοχείων (container).Εξαιτίας αυτής της λειτουργίας τους τα Docker containers είναι εξαιρετικά φορητά.

Η διαδικασία του containerization συχνά συγκρίνεται με τις εικονικές μηχανές (VM-Virtual Machine). Η βασική διαφορά τους από αυτές είναι ότι οι εικονικές μηχανές απαιτούν ένα λειτουργικό σύστημα εγκατεστημένο σε έναν κεντρικό (host) υπολογιστή με έναν hypervisor ο οποίος θα δημιουργεί, θα "τρέχει" και θα επιβλέπει την εκτέλεση των εικονικών μηχανών ή αλλιώς όπως ονομάζονται των quest machines. Κάθε quest machine αντιστοιχεί σε ένα διαφορετικό φιλοξενούμενο λειτουργικό σύστημα με τις δικές του βιβλιοθήκες και τον κώδικα εφαρμογής του. Αυτό όμως έχει ως αποτέλεσμα σημαντική υπολογιστική επιβάρυνση. Αν για παράδειγμα θέλουμε να "τρέξει" ένας απλός Ubuntu server σε μια εικονική μηχανή σε έναν υπολογιστή με λειτουργικό σύστημα Ubuntu τότε θα έχουμε τις περισσότερες από τις βιβλιοθήκες του πυρήνα και από τα δυαδικά αρχεία του πυρήνα να εκτελούνται ¨εις διπλούν" και πολλές διεργασίες να αντιγράφονται και να "τρέχουν" και στον κεντρικό υπολογιστή (host) και στον quest. Η διαδικασία containerization από την άλλη πλευρά αξιοποιεί τον υπάρχοντα πυρήνα και το λειτουργικό σύστημα και προσθέτει μόνο τα δυαδικά αρχεία, τις βιβλιοθήκες και τον κώδικα που απαιτείται για την εκτέλεση της συγκεκριμένης εφαρμογής. Αυτό απεικονίζεται και στην παρακάτω εικόνα:



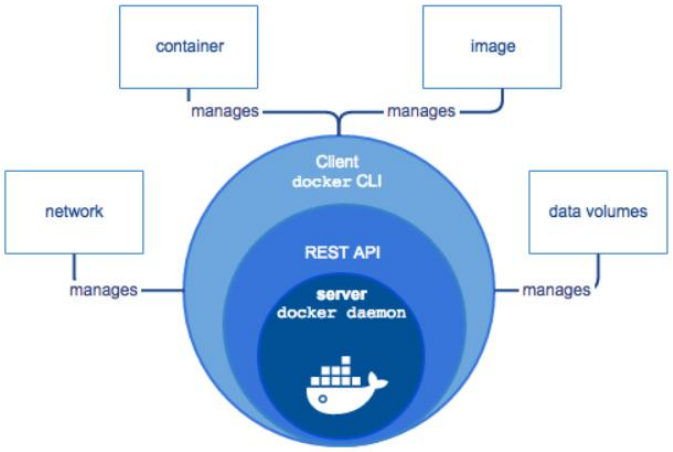
Εικόνα 46 : Διαφορά εκτέλεσης Docker containers και Εικονικών Μηχανών

Τα δοχεία είναι "ελαφριά" επειδή δεν χρειάζονται το πρόσθετο φορτίο ενός hypervisor, αλλά τρέχουν απευθείας μέσα στον πυρήνα του μηχανήματος υποδοχής. Αυτό σημαίνει ότι μπορούμε να εκτελέσουμε περισσότερα κοντέινερ σε ένα δεδομένο συνδυασμό υλικού από ό, τι αν χρησιμοποιούσατε εικονικές μηχανές. Μπορούμε ακόμη να εκτελέσουμε κιβώτια Docker σε μηχανές υποδοχής που είναι πραγματικά εικονικές μηχανές.

Επειδή λοιπόν τα containers δεν χρειάζονται ξεχωριστό λειτουργικό σύστημα απαιτούν λιγότερους υπολογιστικούς πόρους σε σχέση με τις εικονικές μηχανές γεγονός που τους καθιστά ιδανικούς στην περίπτωση θέλει κάποιος να αναπτύξει και να εκτελέσει πολλές ανεξάρτητες υπηρεσίες στην ίδια μηχανή. Τα Duckiebots που πρόκειται να κατασκευάσουμε διαθέτουν έναν υπολογιστή πλακέτας, το Raspberry Pi. Για το μέγεθος και την κατηγορία της, αυτή η αναπτυξιακή πλατφόρμα θεωρείται αρκετά ικανή όπως θα περιγράψουμε και παρακάτω όμως επειδή πρόκειται να επωμιστεί το βάρος της εκτέλεσης πολλών ανεξάρτητων υπηρεσιών κατά τη λειτουργία του Duckiebot η λύση αυτή του containerization είναι επιβεβλημένη.

Η μηχανή του Docker (Docker Engine) είναι μία εφαρμογή πελάτη-διακομιστή με τα παρακάτω βασικά στοιχεία [17]:

- Ένας διακομιστής (server) στον οποίο τρέχει ένα πρόγραμμα long-running που ονομάζεται daemon διεργασία. Long-running ονομάζονται τα προγράμματα που μπορούν να διαχειριστούν περισσότερα από ένα μηνύματα. Ο όρος της μακράς εκτέλεσης δεν έχει την έννοια του χρόνου αλλά της διαχείρισης πολλαπλών μηνυμάτων.
- Ένα REST API το οποίο καθορίζει τις διεπαφές τις οποίες μπορούν να χρησιμοποιήσουν τα προγράμματα για να "μιλήσουν" με τον daemon και να του δώσουν οδηγίες για το τι πρέπει να κάνει.
- Μια διεπαφή γραμμής εντολών (CLI - Command Line Interface)

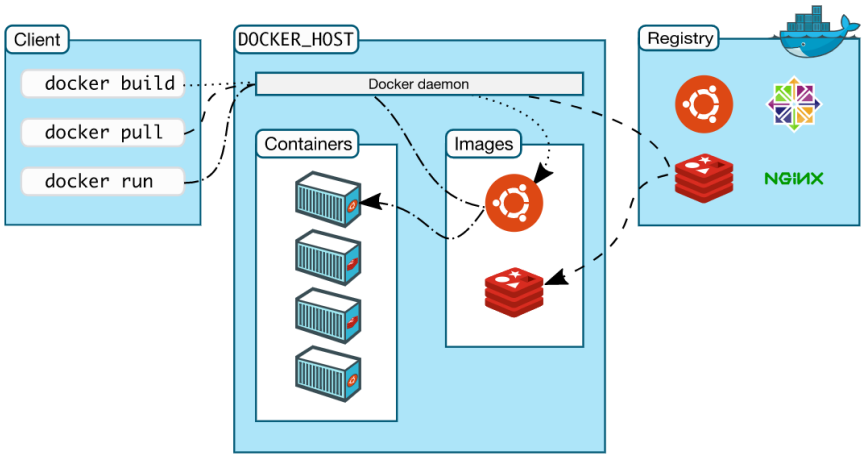


Εικόνα 47 : Βασικά συστατικά Docker Engine

Το CLI χρησιμοποιεί το Docker REST API για τον έλεγχο ή την αλληλεπίδραση με τον Docker daemon μέσω σεναρίου εντολών ή απευθείας εντολών CLI. Πολλές άλλες εφαρμογές Docker χρησιμοποιούν τόσο το API όσο και το CLI. Το πρόγραμμα daemon κατασκευάζει και διαχειρίζεται αντικείμενα Docker όπως εικόνες (Images), κιβώτια (containers), δίκτυα (networks) και τόμους (volumes) όπως φαίνεται και στην παραπάνω εικόνα.

Η αρχιτεκτονική του Docker

Το Docker χρησιμοποιεί μια αρχιτεκτονική πελάτη - εξυπηρετητή (client-server). Ο πελάτης Docker "μιλάει" με το πρόγραμμα daemon το οποίο είναι επιφορτισμένο με τη δόμηση, την εκτέλεση και τη διανομή των containers (κιβωτίων). Οι δύο αυτές εφαρμογές, πελάτης Docker και Docker daemon μπορούν να εκτελεστούν στο ίδιο σύστημα ή να συνδέσουμε το πρόγραμμα πελάτη Docker σε έναν απομακρυσμένο daemon. Οι δύο αυτές εφαρμογές επικοινωνούν χρησιμοποιώντας ένα REST API μέσω θυρών - υποδοχών (sockets) UNIX ή μιας διεπαφής δικτύου.



Εικόνα 48 : Αρχιτεκτονική του Docker

Μια εικόνα Docker (Docker image) είναι μία build-time στατική δομή, ένα αρχείο (όπως για παράδειγμα ένα αρχείο τύπου iso ή zip), ένα πρότυπο μόνο για ανάγνωση με οδηγίες για τη δημιουργία ενός κιβωτίου (container) Docker. Συχνά μια εικόνα βασίζεται σε μια άλλη εικόνα με κάποια επιπλέον προσαρμογή. Για παράδειγμα μπορούμε να δημιουργήσουμε μια εικόνα που βασίζεται στην εικόνα του Ubuntu αλλά να εγκαθιστά στον διακομιστή ιστού Apache και την εφαρμογή μας μαζί με όλες τις λεπτομέρειες διαμόρφωσης που απαιτούνται για την εκτέλεση της.

Οι εικόνες Docker δημιουργούνται από στρώματα. Τα αρχικό, βασικό στρώμα είναι συνήθως μια απογυμνωμένη έκδοση ενός λειτουργικού συστήματος. Για παράδειγμα πολλές από τις Docker εικόνες που τρέχουν στα Duckiebots έχουν ως βάση το rpi-ros-kinetic-base. Μπορούμε να δημιουργήσουμε τις δικές μας εικόνες Docker ή να χρησιμοποιήσουμε μόνο αυτές που έχουν δημιουργηθεί από άλλους και είναι δημοσιευμένες στο μητρώο. Για κατασκευάσουμε την δική μας εικόνα, δημιουργούμε ένα Dockerfile με μια απλή σύνταξη για τον ορισμό των βημάτων που απαιτούνται για τη δημιουργία και εκτέλεση της εικόνας. Κάθε εντολή σε ένα Dockerfile δημιουργεί ένα στρώμα στην εικόνα. Όταν αλλάζουμε το Dockerfile και ανοικοδομούμε την εικόνα, μόνο τα στρώματα που έχουν αλλάξει ξαναχτίζονται. Αυτό ακριβώς είναι που κάνει τις εικόνες τόσο "ελαφριές", μικρές και γρήγορες σε σχέση άλλες τεχνολογίες εικονοποίησης.

Τα Docker containers είναι δομές run-time. Το Docker container δημιουργείται όταν το Docker image εισέρχεται σε φάση εκτέλεσης και συνεπώς είναι κάτι δυναμικό. Μπορούμε δηλαδή να διακόψουμε, να επανεκκινήσουμε ή βάλουμε σε προσωρινή αναστολή τη λειτουργία ενός container. Πολύ απλοϊκά θα μπορούσαμε να παρομοιάσουμε την εικόνα Docker με μία συνταγή μαγειρικής που περιγράφει με σαφήνεια όλα τα υλικά που απαιτούνται αλλά και τις ενέργειες (βήματα) που πρέπει να εκτελεστούν για να παραχθεί ένα συγκεκριμένο γλυκό ή φαγητό. Όταν αυτή τη συνταγή αρχίζει να την εκτελεί ένας μάγειρας ή ζαχαροπλάστης τότε μιλάμε για Docker container.

Το Docker container είναι μια τυποποιημένη μονάδα λογισμικού που πακετάρει τον κώδικα και όλες τις εξαρτήσεις του, όλα δηλαδή τα απαραίτητα για την εκτέλεση του στοιχεία. Μια εικόνα δοχείου Docker (Docker container image) είναι ένα ελαφρύ, αυτόνομο εκτελέσιμο πακέτο λογισμικού που περιλαμβάνει όλα όσα απαιτούνται για την εκτέλεση της εφαρμογής, δηλαδή κώδικα, εργαλεία συστήματος, βιβλιοθήκες και ρυθμίσεις.

Ένα επιπλέον ακόμα πλεονέκτημα των containers εκτός του ότι επιτρέπουν την επαναχρησιμοποίηση των πόρων και του κώδικα είναι ότι πολύ εύκολο να ελέγξουμε ή να τροποποιήσουμε το περιεχόμενο της εφαρμογής.

Στον κόσμο του Docker οι εικόνες οργανώνονται από το όνομα του αποθετηρίου, το όνομα της εικόνας και τις ετικέτες. Όπως και με το GitHub οι εικόνες του Docker αποθηκεύονται σε καταχωρητές εικόνων. Το πιο δημοφιλές μητρώο Docker ονομάζεται DockerHub και είναι αυτό που χρησιμοποιούμε στο Duckietown. Μια εικόνα που είναι αποθηκευμένη στο Dockerhub έχει την εξής μορφοποίηση :

όνομα_αποθετηρίου/όνομα_εικόνας: όνομα_ετικέτας

Έτσι για παράδειγμα η αποθηκευμένη εικόνα duckietown/rpi-kinetic-base:master18 προέρχεται από το αποθετήριο duckietown το όνομα της είναι rpi-kinetic-base και το όνομα της ετικέτας της είναι master18.

Όλες οι εικόνες που σχετίζονται με το Duckietown βρίσκονται στο αποθετήριο του Duckietown. Οι ίδιες οι εικόνες μπορεί να είναι πολύ διαφορετικές και για διάφορες εφαρμογές. Μερικές φορές μια συγκεκριμένη εικόνα μπορεί να έχει αρκετές διαφορετικές εκδόσεις. Αυτές μπορούν να οριστούν με ετικέτες. Για παράδειγμα, η ετικέτα master18 σημαίνει ότι αυτή είναι η εικόνα που θα χρησιμοποιηθεί με την έκδοση DT18 του Duckietown. Δεν είναι απαραίτητο να καθορίσουμε μια ετικέτα. Εάν δεν το κάνετε, το Docker υποθέτει ότι μας ενδιαφέρει η εικόνα με την πιο πρόσφατη ετικέτα.

6.5.1 Εντολές διαχείρισης εικόνων (docker images)

Αν θέλουμε να "κατεβάσουμε" μια καινούργια εικόνα από ένα Docker αποθετήριο στον τοπικό μας υπολογιστή για πρέπει να την "τραβήξουμε". Έτσι για παράδειγμα αν θέλουμε να κατεβάσουμε μια εικόνα Ubuntu 18.04 εκτελούμε την εντολή:

$ docker pull library/ubuntu:18.04

ενώ για δούμε τις εικόνες που έχουμε στον υπολογιστή μας εκτελούμε:

$ docker image list

Για να διαγράψουμε την εικόνα που κατεβάσαμε πριν (Ubuntu 18.04) ώστε να μη δεσμεύει αποθηκευτικό χώρο στο υπολογιστικό μας σύστημα πληκτρολογούμε

$ docker image rm ubuntu:18.04

Μπορούμε επίσης να διαγράψουμε μια εικόνα χρησιμοποιώντας το ID της όπως αυτό εμφανίζεται με την εντολή docker image list. Τέλος μπορούμε να διαγράψουμε πολλές εικόνες που δεν χρησιμοποιούμε πια με την εντολή:

$ docker image prune

Πρέπει ασφαλώς να είμαστε προσεκτικοί ώστε να μη διαγράψουμε εικόνες που χρειαζόμαστε. Στο σημείο αυτό να σημειώσουμε ότι δεν μπορούμε να διαγράψουμε εικόνες που χρησιμοποιεί ένα Docker container. Για το κάνουμε αυτό θα πρέπει να σταματήσουμε την εκτέλεση του κιβωτίου (container) να το αποσύρουμε και έπειτα να διαγράψουμε τις σχετικές με αυτό εικόνες. Τέλος αν θέλουμε να εξετάσουμε τα ενδότερα των εικόνων μπορούμε να εκτελέσουμε τις εντολές

$ docker image history και

$ docker image inspect

για να πάρουμε μια λεπτομερή αναφορά για το τί περιλαμβάνουν.

6.5. 2 Διαχείριση Docker containers

Όταν επιθυμούμε να εκκινήσει ένα container το Docker παίρνει την εικόνα (image) που έχουμε καθορίσει, δημιουργεί ένα σύστημα αρχείων από τα στρώματα της εικόνας συνδέει όλες τις συσκευές και τους καταλόγους που θέλουμε και ρυθμίζει το περιβάλλον εκτέλεσης του container. Όλη αυτή η διαδικασία λαμβάνει χώρα με την εντολή docker run. Εδώ να σημειώσουμε ότι αν δεν εντοπίσει την εικόνα docker σε κάποιο κατάλογο του τοπικού υπολογιστή θα την αναζητήσει και θα την τραβήξει από το αποθετήριο DockeHub. Έτσι για παράδειγμα αν θέλουμε να τρέξουμε την εικόνα Ubuntu εκτελούμε την εντολή:

$ docker run Ubuntu

Στην πράξη βέβαια ο φλοιός (bash) εκτελείται ήδη αφού μέσα από αυτόν δώσαμε την παραπάνω εντολή με αποτέλεσμα το container να εξέλθει αμέσως από τη διαδικασία εκτέλεσης. Αν θέλουμε να το κρατήσουμε ενεργό χρησιμοποιώντας τον διακόπτη -it δημιουργώντας μία διαδραστική συνεδρία. Εκτελώντας την παρακάτω εντολή:

$ docker run -it ubuntu

βλέπουμε στην οθόνη μας ότι βρισκόμαστε στον φλοιό του container και γ' αυτό πλέον εμφανίζεται ο κωδικός αριθμός της διεργασίας του container. Αν δοκιμάσουμε να εκτελέσουμε κάποιες εντολές όπως για παράδειγμα τα δούμε τα περιεχόμενα του καταλόγου στον οποίο βρισκόμαστε εύκολα αντιλαμβανόμαστε ότι βρισκόμαστε στο περιβάλλον του container και όχι στον host υπολογιστή μας.

```
root@938a556e9af6:/# ls
bin  boot  dev  etc  home  lib  lib64  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
```

```
> $ ls
4974_Odhgies   DuckieTown                    Pictures                    Templates
Desktop        examples.desktop             Public                      Videos
Documents      'first communicate with docker'  rpi-duckiebot-simple-python  zoom_amd64.deb
Downloads      Music                        snap                       'Βιβλία Τομέα - Ειδικότητας'
```

Για να ελέγξουμε τα ενεργά containers τα οποία εκτελούνται στον υπολογιστή μας πληκτρολογούμε

$ docker ps

```
> $ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
00dacae45fa2        ubuntu              "/bin/bash"         24 seconds ago      Up 22 seconds                           wizardly_kepler
```

ή εναλλακτικά

$ docker container list

Αν θελήσουμε να τερματίσουμε την εκτέλεση του Ubuntu container πληκτρολογούμε exit εντολή που θα μας επαναφέρει στον φλοιό του υπολογιστή μας. Για να δούμε όλα τα containers τα οποία έτρεξαν στον υπολογιστή μας αρκεί να γράψουμε:

$ docker container list -a

Για να εκκινήσουμε, να σταματήσουμε ή να επανεκκινήσουμε την εκτέλεση ενός container πληκτρολογούμε αντίστοιχα

$ docker container start όνομα_container

$ docker container stop όνομα_container

$ docker container restart όνομα_container

Αν εκτελείτε ένα container στο παρασκήνιο χωρίς να έχουμε προσαρτημένο φλοιό με αυτό ώστε να μπορούμε να αλληλεπιδρούμε μπορούμε να ενεργοποιήσουμε ένα τερματικό συνδεδεμένο με το container που εκτελείται με την εντολή

$ docker attach όνομα_container

Τέλος για να διαγράψουμε containers που δεν χρειαζόμαστε πια γράφουμε

$ docker container rm όνομα_container.

Εμείς βέβαια θα χρειαστεί να εκκινούμε containers με πιο εξελιγμένες επιλογές. Ας δούμε ένα τέτοιο παράδειγμα για να σχολιάσουμε τις επιλογές αυτές:

```
$ docker -H hostname.local run -dit --privileged --name joystick --
network=host -v /data:/data duckietown/rpi-duckiebot-joystick-de-
mo:master18
```

Ο διακόπτης -H δηλώνει την εκτέλεση του container σε κάποιο απομακρυσμένο υπολογιστικό σύστημα, για παράδειγμα στο Duckiebot γι' αυτό ακολουθείται και από το αναγνωριστικό της απομακρυσμένης υπολογιστικής μονάδας. Η επιλογή --privileged παρέχει εκτεταμένα δικαιώματα σ' αυτό το container που πρακτικά περιλαμβάνει την πρόσβαση σε όλες τις συσκευές που είναι συνδεδεμένες. Ακολουθεί το όνομα (--name) του προς εκτέλεση container που στη περίπτωση μας είναι joystick ενώ έπεται η διαδρομή του καταλόγου του host υπολογιστή που ορίζεται ως κατάλογος του container.

6.5. 3 Εγκατάσταση του Docker

Αρχικά θα πρέπει να απεγκαταστήσουμε τυχόν παλαιότερες εκδόσεις του Docker με την ονομασία docker, docker.io ή docker-engine αν αυτά είναι εγκατεστημένα στην υπολογιστική μας μονάδα [18].

$ sudo apt-get remove docker docker-engine docker.io containerd runc

Τα περιεχόμενα του καταλόγου /var /lib/docker /, συμπεριλαμβανομένων εικόνων, κοντέινερ, τόμων και δικτύων, διατηρούνται. Το πακέτο Docker Engine ονομάζεται τώρα docker-ce.

Μπορούμε να εγκαταστήσουμε το Docker Engine με τρεις διαφορετικούς τρόπους ανάλογα με τις ανάγκες μας.

- Οι περισσότεροι χρήστες ρυθμίζουν τα αποθετήρια του Docker και εγκαθιστούν από αυτούς, για ευκολία εγκατάστασης και αναβάθμιση εργασιών. Αυτή είναι και η προτεινόμενη προσέγγιση.
- Ορισμένοι χρήστες κατεβάζουν το πακέτο DEB και το εγκαθιστούν χειροκίνητα και διαχειρίζονται πλήρως τις αναβαθμίσεις χειροκίνητα. Αυτό είναι χρήσιμο στις περιπτώσεις που εγκαθιστούμε το Docker σε συστήματα χωρίς πρόσβαση στο Διαδίκτυο.
- Σε περιβάλλοντα δοκιμών και ανάπτυξης, ορισμένοι χρήστες επιλέγουν να χρησιμοποιούν αυτοματοποιημένα σενάρια ευκολίας για την εγκατάσταση του Docker.

Εμείς θα επιλέξουμε την προτεινόμενη μέθοδο εγκατάστασης. Έτσι πριν εγκαταστήσουμε το Docker Engine για πρώτη φορά στον υπολογιστή μας θα πρέπει να ορίσουμε το αποθετήριο Docker. Στη συνέχεια, μπορούμε να εγκαταστήσουμε και να ενημερώσουμε το Docker από το αποθετήριο.

1. Ενημερώνουμε το ευρετήριο πακέτων και επιτρέπουμε στη διεπαφή διαχείρισης πακέτων apt να χρησιμοποιεί ένα αποθετήριο μέσω https.

   $ sudo apt-get update

   $ sudo apt - get install \
   apt-transport-https \
   ca-certificates \
   curl \
   gnupg-agent \
   software-properties-common

2. Προσθέτουμε το επίσημο κλειδί GPG του Docker

   $ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

   και επιβεβαιώνουμε ότι διαθέτουμε το κλειδί με το δακτυλικό αποτύπωμα 9DC8 5822 9FC7 DD38 854A E2D8 8D81 803C 0EBF CD88 αναζητώντας τους τελευταίους 8 χαρακτήρες του.

```
sakis@SakisLaptop ~
> $ sudo apt-key fingerprint 0EBFCD88
pub   rsa4096 2017-02-22 [SCEA]
      9DC8 5822 9FC7 DD38 854A  E2D8 8D81 803C 0EBF CD88
uid           [ unknown] Docker Release (CE deb) <docker@docker.com>
sub   rsa4096 2017-02-22 [S]


sakis@SakisLaptop ~
> $
```

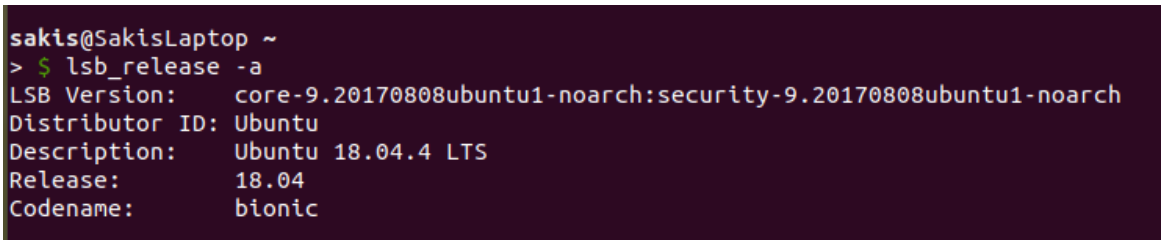
Εικόνα 49 : Επιβεβαίωση κλειδιού GPG

3. Ορίζουμε τον τύπο του αποθετηρίου που θα χρησιμοποιήσουμε. Υπάρχουν τρεις διαφορετικοί τύποι: Το σταθερό (stable) το δοκιμαστικό (test) και το νυχτερινό (nightly). Εμείς θα κάνουμε χρήση του σταθερού αποθετηρίου.

   Πρώτα θα πρέπει να γνωρίζουμε τον κωδικό της έκδοσης της διανομής που έχουμε εγκατεστημένη στον υπολογιστή μας. Αυτό το διαπιστώνουμε με την εντολή lsb_release.
   Η εντολή lsb_release εμφανίζει πληροφορίες σχετικά με τη συγκεκριμένη διανομή Linux που έχουμε στον υπολογιστή μας συμπεριλαμβανομένου του αριθμού και του κωδικού έκδοσης και του αναγνωριστικού διανομέα. Αποτελεί μέρος ενός πακέτου με την ονομασία LSB (Linux Standard Base) core το οποίο δεν είναι εγκατεστημένο κατ' ανάγκη στο λειτουργικό μας σύστημα. Για να το εγκαταστήσουμε στα Ubuntu πληκτρολογούμε:

   $ sudo apt-get update && sudo apt-get install lsb-core

   Για να όλες τις πληροφορίες της εγκατεστημένης μας διανομής γράφουμε :
   lsb_release -a



Εικόνα 50 : Πληροφορίες εγκατεστημένης διανομής Ubuntu στον υπολογιστή μας

   Αφού μας εμφανίσει τον κωδικό της διανομής μας (bionic) τον χρησιμοποιούμε για να ορίσουμε το stable αποθετήριο μας.
   sudo add-apt-repository \
   "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
   bionic \
   stable "

4. Τώρα είμαστε έτοιμοι να εγκαταστήσουμε την τελευταία έκδοση του Docker Engine πληκτρολογώντας

   $ sudo apt-get update
   $ sudo apt-get install docker-ce docker-ce-cli containerd.io

5. Για να διαπιστώσουμε αν η εγκατάσταση του docker engine ήταν επιτυχής θα "τρέξουμε" το docker image hello-world γράφοντας

$ sudo docker run hello-world

```
sakis@SakisLaptop ~
> $ sudo docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

Εικόνα 51 : Μήνυμα επιτυχούς εγκατάστασης του Docker Engine

6. 6 Περιγραφή του ROS

Πριν περιγράψουμε τα επόμενα βήματα εγκατάστασης των απαραίτητων για τη λειτουργία του έργου λογισμικών θα ήταν μεγάλη παράλειψη αν δεν αναφερόμασταν στο ROS. Είπαμε στις προηγούμενες σελίδες ότι πολλές από τις Docker images που "τρέχουν" στα Duckiebots έχουν ως βάση το rpi-ros-kinetic-base. Γενικά όλη η αρχιτεκτονική της πλατφόρμας του Duckietown στηρίζεται στο ROS. Το πρώτο ερώτημα που τίθεται είναι: Τί είναι το ROS; και το δεύτερο: Γιατί επιλέχθηκε το ROS; Ας ξεκινήσουμε από το δεύτερο [19].

Το Duckiebot είναι ένα πολύ απλά στην υλοποίηση του ρομπότ καθώς διαθέτει έναν μόνο αισθητήρα, την κάμερα και δύο ενεργοποιητές, τους κινητήρες των τροχών. Πιθανότατα θα μπορούσαμε να αναπτύξουμε δικό μας κώδικα που θα ορίζει τη συμπεριφορά του μέσα στην πόλη βασιζόμενοι στις εικόνες που λαμβάνουμε από την κάμερα. Θα μπορούσαμε δηλαδή να είχαμε δημιουργήσει ένα πρόγραμμα όπως το παρακάτω:

```
img = get_image_from_camera()
pose = get_pose_from_image(img)
cmd = get_command_from_pose(pose)
run_motors(cmd)
```

Αν έπειτα διαπιστώναμε ότι το Duckiebot συγκρούεται με ένα παπί που έτυχε να διασχίζει το δρόμο θα θέλαμε να προσθέσουμε στον παραπάνω κώδικα τη λειτουργικότητα της ανίχνευσης του duckie ώστε να αποφεύγονται τέτοια ατυχήματα. Έτσι πιθανόν να εμπλουτίζαμε τον παραπάνω κώδικα ως εξής:

```
img = get_image_from_camera()
pose = get_pose_from_image(img)
cmd = get_command_from_pose(pose)

if duckie_detected(img):
    cmd = EMERGENCY_STOP

run_motors(cmd)
```

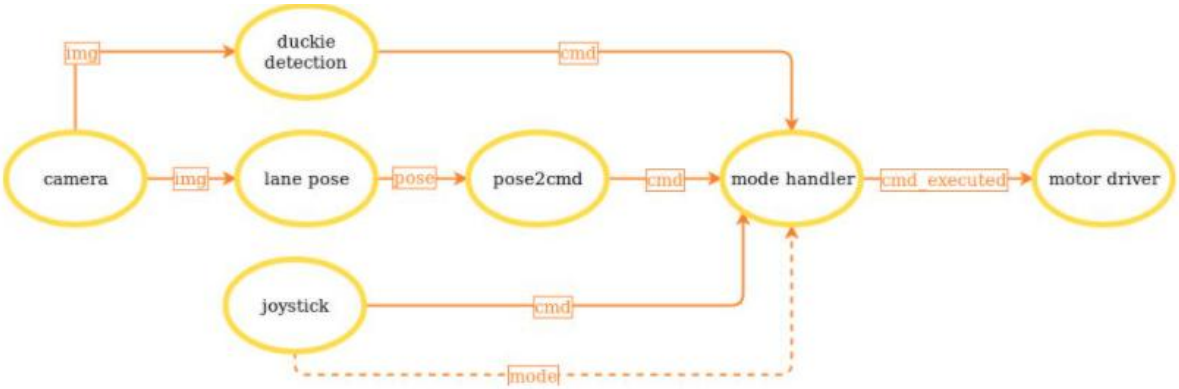
Στη συνέχεια βέβαια αφού το Duckiebot μας δεν έχει ακόμα αυτονομία επιπέδου 5 θα θέλαμε νε προσθέσουμε και χειροκίνητο έλεγχο για κάποιες "δύσκολες" στιγμές κατά τη περιήγηση του μέσα στην πόλη οπότε να ο κώδικας θα γινόταν κάπως έτσι

```
img = get_image_from_camera()
pose = get_pose_from_image(img)
cmd = get_command_from_pose(pose)

if mode == AUTONOMOUS:
    if duckie_detected(img):
        cmd = EMERGENCY_STOP
else:
    cmd = get_command_from_joystick()

run_motors(cmd)
```

Από το παραπάνω παράδειγμα εύκολα γίνεται κατανοητό ότι στην προσπάθεια μας να βελτιώσουμε τη συμπεριφορά του Duckiebot και να το κάνουμε πιο "έξυπνο" όπως για παράδειγμα να περνά μία διασταύρωση μόνο του ή να είναι ικανό να ανιχνεύει ένα άλλο Duckiebot ή μια πινακίδα σήμανσης θα καταλήγαμε σε ένα μακροσκελή και πολύπλοκο κώδικα. Πως θα μπορούσαμε να απλοποιήσουμε λίγο τα πράγματα; Μήπως αν χωρίζαμε το πρόγραμμα μας σε διαφορετικά ανεξάρτητα δομικά στοιχεία όπου για παράδειγμα το ένα κομμάτι θα λαμβάνει εικόνες από την κάμερα, το άλλο θα ανιχνεύει πεζούς, το τρίτο να ελέγχει τους κινητήρες και ούτω καθεξής; Βέβαια όλες αυτές οι προγραμματιστικές οντότητες θα έπρεπε να επικοινωνούν μεταξύ τους και να μπορούν εύκολα να εμπλουτιστούν και επεκταθούν και δημιουργηθούν νέες ώστε να μπορεί αυτή η υλοποίηση να ανταποκριθεί και σε ποιο προηγμένες κατασκευές (ρομπότ) και περισσότερους αισθητήρες και μεγαλύτερο εύρος πιθανών συμπεριφορών.

Παρατηρώντας προσεκτικά το παρακάτω γράφημα διαπιστώνουμε ότι εκτελεί την ίδια εργασία με τον κώδικα μας αλλά είναι δομημένο από ανεξάρτητες οντότητες. Αυτό πρακτικά σημαίνει ότι όχι μόνο μπορούμε να τροποποιήσουμε τον κώδικα μιας συγκεκριμένης προγραμματιστικής μονάδας αλλά είναι εφικτές οι αλλαγές παράλληλα σε πολλές προγραμματιστικές οντότητες του συστήματος. Επίσης είναι δυνατή η παράλληλη εκτέλεση τους συμβάλλοντας στην καλύτερη αξιοποίηση των υπολογιστικών μας πόρων. Έτσι για

παράδειγμα ο αλγόριθμος ακολούθησης λωρίδας μπορεί να εκτελείται παράλληλα με τον αλγόριθμο ανίχνευσης duckie. Αυτός ακριβώς είναι ο τρόπος που λειτουργεί το ROS.



Εικόνα 52 : Διάγραμμα λειτουργίας ROS

Στο ROS κάθε πλαίσιο ονομάζεται κόμβος(node) και κάθε βέλος ονομάζεται θέμα (topic). Είναι φανερό ότι κάθε θέμα φέρει διαφορετικό τύπο *μηνύματος* . Το θέμα img έχει εικόνες που είναι πίνακες αριθμών, ενώ το θέμα pose μπορεί να έχει στοιχεία περιστροφής και μετάφρασης. Το ROS παρέχει πολλούς προκαθορισμένους τύπους μηνυμάτων όπως για παράδειγμα Int, Bool, String, εικόνες, στάσεις, μετρήσεις IMU, κτλ. Μπορούμε επίσης να ορίσουμε τα δικά μας προσαρμοσμένα μηνύματα που συνδυάζουν διαφορετικούς τύπους μηνυμάτων σε ένα.

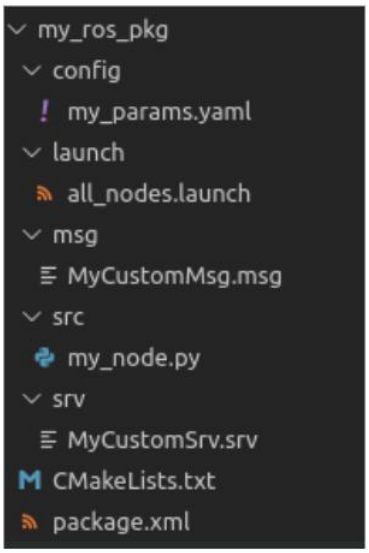
Οι κόμβοι που στέλνουν δεδομένα σε ένα θέμα ονομάζονται εκδότες(publishers) αυτού του θέματος και αυτοί που λαμβάνουν τα δεδομένα και τα χρησιμοποιούν ονομάζονται συνδρομητές (subscriber) αυτού του θέματος. Όπως φαίνεται από το παραπάνω διάγραμμα, ένας κόμβος μπορεί να είναι εκδότης για ένα θέμα και συνδρομητής για άλλο ταυτόχρονα.

Στο παραπάνω γράφημα υπάρχει ένα διακεκομμένο βέλος από τον κόμβο joystick στον κόμβο mode_handler. Αυτό δηλώνει ότι μπορούμε να αλλάξουμε από χειροκίνητη σε αυτόνομη και αντίστροφα τη λειτουργία του Duckiebot χρησιμοποιώντας ένα κουμπί στο (εικονικό) χειριστήριο σας. σε αντίθεση με την αποστολή εικόνων, η οποία είναι μια συνεχής ροή πληροφοριών. Το ROS έχει ένα πλαίσιο ειδικά σχεδιασμένο για μια τέτοια περίπτωση. Αυτό ονομάζεται *υπηρεσία* . Όπως και με τα μηνύματα, μπορείτε επίσης να ορίσετε τις δικές σας υπηρεσίες. Εδώ, ο κόμβος mode_handler προσφέρει μια υπηρεσία και ο κόμβος joystick είναι ο πελάτης αυτής της υπηρεσίας.

Αυτό που διαχειρίζεται τις συνδέσεις μεταξύ κόμβων είναι το ROS Master. Το ROS Master είναι υπεύθυνο για να βοηθά τους μεμονωμένους κόμβους να βρίσκουν ο ένας τον άλλον και να δημιουργούν συνδέσεις μεταξύ τους. Αυτό μπορεί να γίνει επίσης και μέσω δικτύου. Αυτό οφείλεται στο γεγονός ότι ο φορητός υπολογιστής μας είναι συνδεδεμένος με το ROS Master του Duckiebot. Έτσι, χωρίς ίσως να το γνωρίζουμε, κάνουμε ήδη κατανεμημένη ρομποτική. Είναι σημαντικό να έχουμε κατά νου ότι ένας κόμβος μπορεί να διαχειρίζεται μόνο έναν ROS Master κάθε φορά.

Ένα άλλο βασικό στοιχείο του ROS είναι οι παράμετροι για κάθε κόμβο. Όπως θα δούμε και παρακάτω θα χρειαστεί να βαθμονομήσουμε τους τροχούς και την κάμερα του Duckiebot. Τα αποτελέσματα αυτών των διαδικασιών θα αποθηκευτούν σε αρχεία ώστε να μη χάνονται κάθε φορά που απενεργοποιείτε το Duckiebot. Οι παράμετροι αυτοί είναι πολύ χρήσιμοι για τη διαμόρφωση των κόμβων του ROS και συνεπώς για τη "σωστή" συμπεριφορά του robot. Μέσω των παραμέτρων μπορούμε να διορθώσουμε κάποιες "αντιδράσεις" του Duckiebot χωρίς να χρειαστεί να επέμβουμε στον πηγαία κώδικα. Το ROS μας παρέχει μέσω της "εντολής" rosparam πρόσβαση σε μία μεγάλη παλέτα παραμέτρων για διάφορα project ρομποτικής. Είναι εφικτό επίσης να χρησιμοποιήσουμε τις παραμέτρους σε συνδυασμό με υπηρεσίες και να τροποποιήσουμε δυναμικά τη συμπεριφορά των ρομποτικών κατασκευών μας.

Στο ROS, ο κώδικας οργανώνεται με τη μορφή *πακέτων* . Κάθε πακέτο είναι ουσιαστικά μια συλλογή κόμβων που εκτελούν πολύ συγκεκριμένες, σχετικές εργασίες. Τα πακέτα ROS περιέχουν επίσης μηνύματα, υπηρεσίες και προεπιλεγμένα αρχεία διαμόρφωσης παραμέτρων που χρησιμοποιούνται από τους κόμβους. Ένα τυπικό πακέτο ROS μοιάζει όπως φαίνεται στο παρακάτω στιγμιότυπο



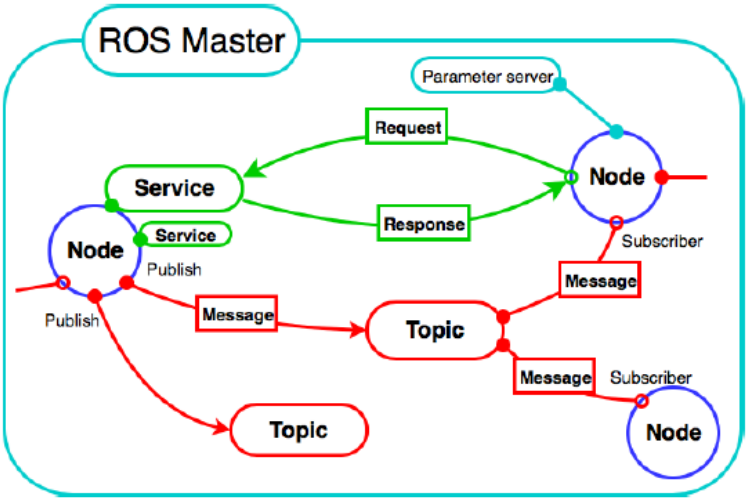
Εικόνα 53: Δομή πακέτου του ROS

Βέβαια αυτό που λαμβάνει χώρα πραγματικά κατά τη λειτουργία των Duckiebots είναι λίγο διαφορετικό όπως θα δούμε παρακάτω.

Κανονικά για χρησιμοποιήσει κάποιος το ROS θα πρέπει να το εγκαταστήσει τον υπολογιστή του. Βέβαια το ROS δεν είναι συμβατό με όλα τα λειτουργικά συστήματα αλλά "τρέχει" σε λειτουργικά συστήματα GNU/Linux. Στην δική μας περίπτωση αυτό δεν είναι απαραίτητο γιατί γίνεται χρήση του ROS μέσω του Docker. Αν παρόλα αυτά θέλουμε να το εγκαταστήσουμε στον υπολογιστή μας για έχουμε τη δυνατότητα να χρησιμοποιήσουμε και άλλα εργαλεία που μας παρέχει όπως τα περιβάλλοντα οπτικών αναπαραστάσεων θα πρέπει να προσέξουμε να εγκαταστήσουμε την ίδια έκδοση με αυτή που "τρέχουν" τα Docker containers. Η εκδόσεις του **Duckietown shell Master19 βασίζεται στο ROS Kinetic** ενώ η

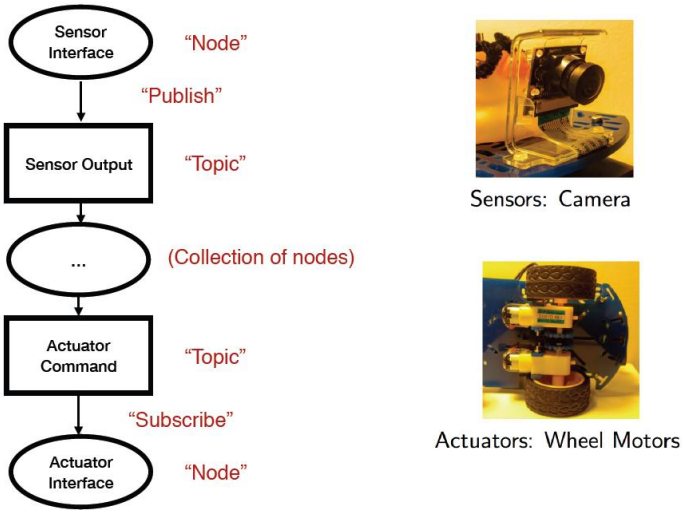
έκδοση daffy κάνει χρήση του ROS Noetic Ninjemys. Η ηλεκτρονική διεύθυνση από την οποία θα μπορούσε να μεταφορτώσουμε τη διανομή του ROS που επιθυμούμε είναι http://wiki.ros.org/ROS/Installation.

Το ROS μπορεί και υλοποιεί διάφορες μορφές επικοινωνίας όπως σύγχρονη τύπου RPC (Remote Procedure Call) πάνω σε υπηρεσίες (services), ασύγχρονη ροή δεδομένων μέσω των θεμάτων (topics) και αποθήκευση δεδομένων σε διακομιστή. Κάνοντας εφικτή την επικοινωνία μεταξύ διαφορετικών μονάδων λογισμικού και διαφορετικών συσκευών μας επιτρέπει να ελέγχουμε τη λειτουργία περισσοτέρων του ενός ρομπότ από ένα σταθερό ή φορητό υπολογιστή όπως στην περίπτωση μας που ελέγχουμε τα Duckiebots από έναν φορητό υπολογιστή. Μάλιστα όπως θα δούμε παρακάτω αυτή η δυνατότητα μας παρέχεται με δύο τρόπους. Είτε μέσω του Duckietown shell είτε μέσω ενός φιλικού και εύχρηστου web interface.



Εικόνα 54 : Κόμβοι, Μηνύματα, Κανάλια του ROS

Πρακτικά η χρήση του ROS μας γλυτώνει από τη συγγραφή πολύπλοκού κώδικα ειδικά στην περίπτωση στη αυτόνομης πλοήγησης των Duckiebots μέσα στην πόλη καθώς μας παρέχει έτοιμο κώδικά για όλες τις διαδικασίες που απαιτούνται για να υλοποιηθεί η αυτόνομη πλοήγηση όπως δημιουργία χαρτών, ανίχνευση λωρίδας, εντοπισμός εμποδίων κτλ. Η παρακάτω εικόνα (55) μας δείχνει το βασικό ρομποτικό pipeline που αρχίζει από τον μοναδικό αισθητήρα που διαθέτουν τα Duckiebots, την κάμερα και ότι αυτή δημοσιεύει και καταλήγει στις εντολές που δέχονται οι ενεργοποιητές του, δηλαδή οι κινητήρες των τροχών.

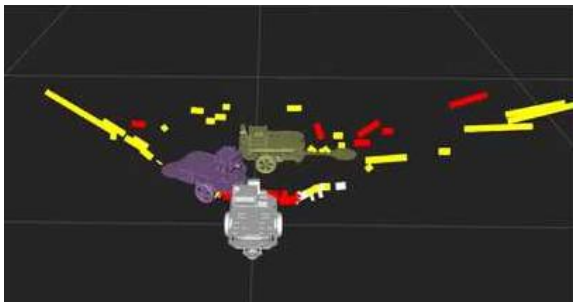


Εικόνα 55: Βασικό ρομποτικό pipeline

Εκτενέστερα θα αναφερθούμε σε αυτό το pipeline στα επόμενα κεφάλαια όπου και θα περιγράψουμε πως επιτυγχάνεται ο τηλεχειρισμός των Duckiebots καθώς και ο τρόπος που κινούνται αυτόνομα ακολουθώντας τη λωρίδα του δρόμου.

Τέλος θα ήταν παράληψη αν δεν κάναμε μία αναφορά στα περιβάλλοντα οπτικών αναπαραστάσεων που διαθέτει το ROS. Συγκεκριμένα διαθέτει:

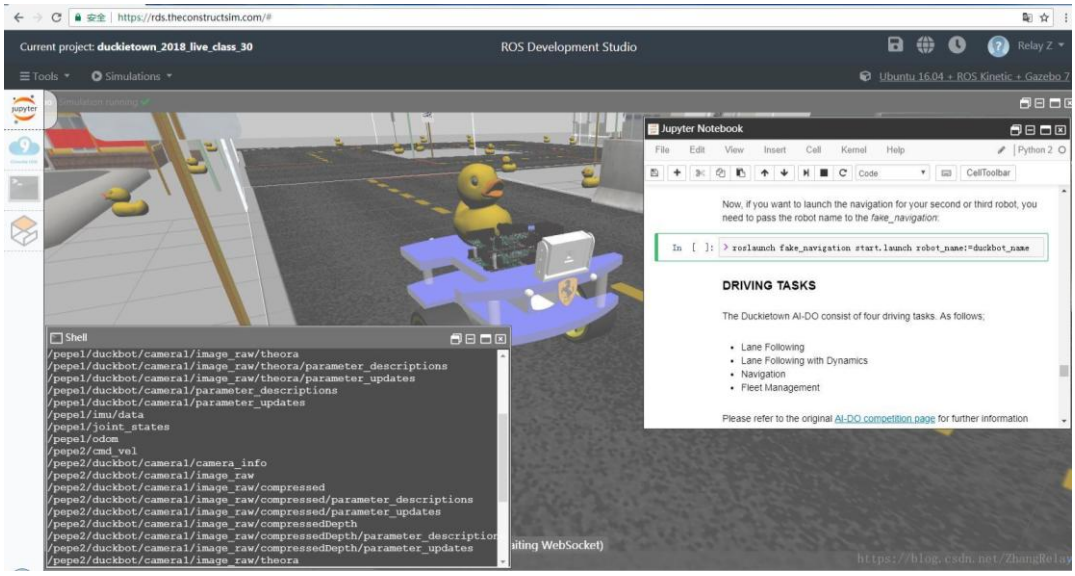
- Το rqt_graph που απεικονίζει γραφικά τη διασύνδεση ανάμεσα σε κόμβους του ROS,
- Το Rviz που είναι ένας τρισδιάστατος οπτικοποιητής για την εμφάνιση δεδομένων όπως : ρομποτικά μοντέλα, δεδομένα τρισδιάστατου μετασχηματισμού του ρομπότ, πληροφοριών κατάστασης του ROS καθώς και μια ποικιλία από δεδομένα διαφορετικών αισθητήρων.



Εικόνα 56: Οπτικοποίηση των Duckiebots στο Rviz

- Το rqt_plot που παρέχει ένα εργαλείο σχεδιασμού με γραφικών παραστάσεις δύο διαστάσεων αριθμητικών τιμών που προέρχονται από topics του ROS
- Το Gazebo, το "βαρύ πυροβολικό" του ROS είναι ένας δυναμικός τρισδιάστατος προσομοιωτής ανοικτού κώδικα που λειτουργεί σε περιβάλλον Linux και φυσικά συνεργάζεται με το ROS. Το Gazebo παρέχει τη δυνατότητα προσομοίωσης φυσικών χαρακτηριστικών και δυνάμεων όπως για παράδειγμα η βαρύτητα, η τριβή, κτλ, διαθέτει μεγάλη σε όγκο βιβλιοθήκη ρομποτικού περιεχομένου καθώς και έναν

μεγάλο αριθμό αισθητηρίων οργάνων και διακρίνεται για το φιλικό και εύχρηστο προγραμματιστικό του περιβάλλον.



Εικόνα 57 : Στιγμιότυπο από τη προσομοίωση Duckiebot στο Gazebo

Περισσότερες πληροφορίες για το ROS μπορούμε να βρούμε στο Παράρτημα της παρούσας τεκμηρίωσης.

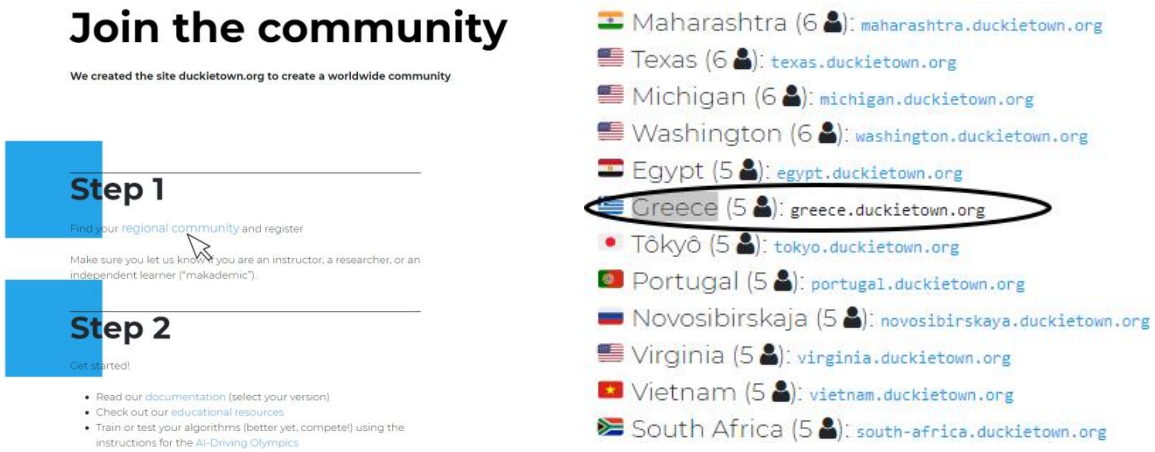
6.7 Δημιουργία λογαριασμού στο Duckietown

Κλείνοντας την παρένθεση της σύντομης περιγραφής του ROS επανερχόμαστε στα προαπαιτούμενα στάδια - βήματα προετοιμασίας της πλατφόρμας του Duckietown από άποψη λογισμικού. Ένα τέτοιο στάδιο είναι και η δημιουργία λογαριασμού στην διαδικτυακή πλατφόρμα του Duckietown ώστε να προμηθευτούμε ένα διακριτικό ελέγχου ταυτότητας (Duckietown token) με βάση το οποίο γίνεται ο έλεγχος της ταυτότητας των συσκευών μας στο δίκτυο Duckietown.

Η εγγραφή είναι μία απλή διαδικασία που γίνεται μέσα από το website:

https://www.duckietown.org

Αρχικά θα πρέπει να επιλέξουμε την κοινότητα στην οποία ανήκουμε με βάση τη χώρα στην οποία κατοικούμε.

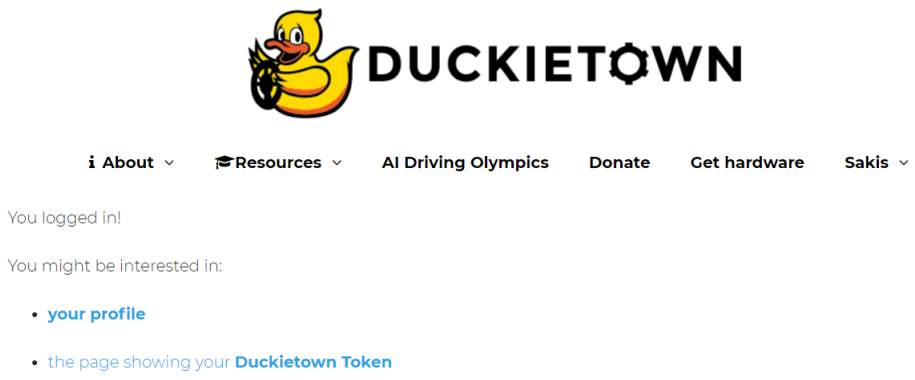


Εικόνα 58 : 1ο Βήμα εγγραφής : Επιλογή χώρας προέλευσης

Έπειτα επιλέγουμε Εγγραφή και συμπληρώνουμε τα ατομικά μας στοιχεία

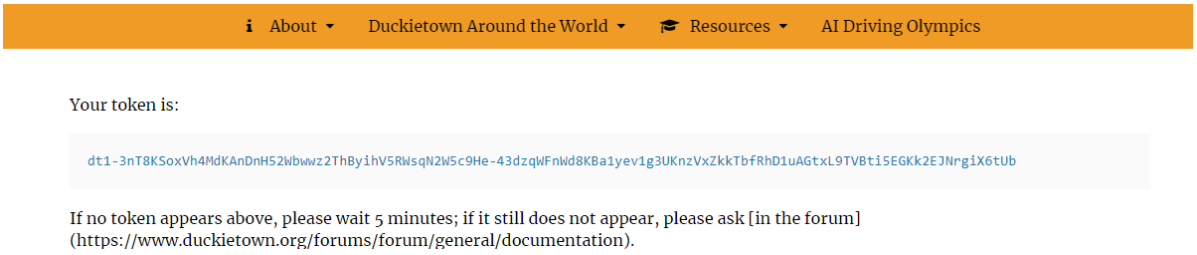
Εικόνα 59 : Στιγμιότυπο από τη διαδικασία εγγραφής στην πλατφόρμα Duckietown

Όταν ολοκληρώσουμε την εγγραφή μας και συνδεθούμε με το όνομα χρήστη και τον κωδικό που έχουμε ορίσει, έχουμε διαθέσιμες δύο βασικές επιλογές: την διαμόρφωση του προφίλ μας και την εμφάνιση του Duckietown Token



Εικόνα 60 : Διαθέσιμες επιλογές μετά την εγγραφή

Ενεργοποιώντας τον δεύτερο σύνδεσμο εμφανίζεται το προσωπικό μας διακριτικό



Εικόνα 61 : Εμφάνιση του προσωπικού μας "κουπονιού"

6.8 Εγκατάσταση Duckietown shell

Το Duckietown Shell είναι ένα βοηθητικό πρόγραμμα βασισμένο στην γλώσσα προγραμματισμού Python εύκολο στη διανομή για το Duckietown project. Ως γνωστόν το μεγαλύτερο μέρος της λειτουργικότητας της πλατφόρμας Duckietown υλοποιείται ως κοντέινερ Docker. Η χρησιμότητα του γραμμένου σε Python φλοιού Duckietown έγκειται στην παροχή μιας φιλικής διεπαφής ώστε ο χρήστης να μην πληκτρολογεί μακροσκελής εντολές (docker run) στη γραμμή εντολών. Για να εγκαταστήσουμε το Duckietown shell πληκτρολογούμε:

$ pip3 install --no cache-dir --user -U duckietown-shell

έπειτα γράφουμε

$ which dts

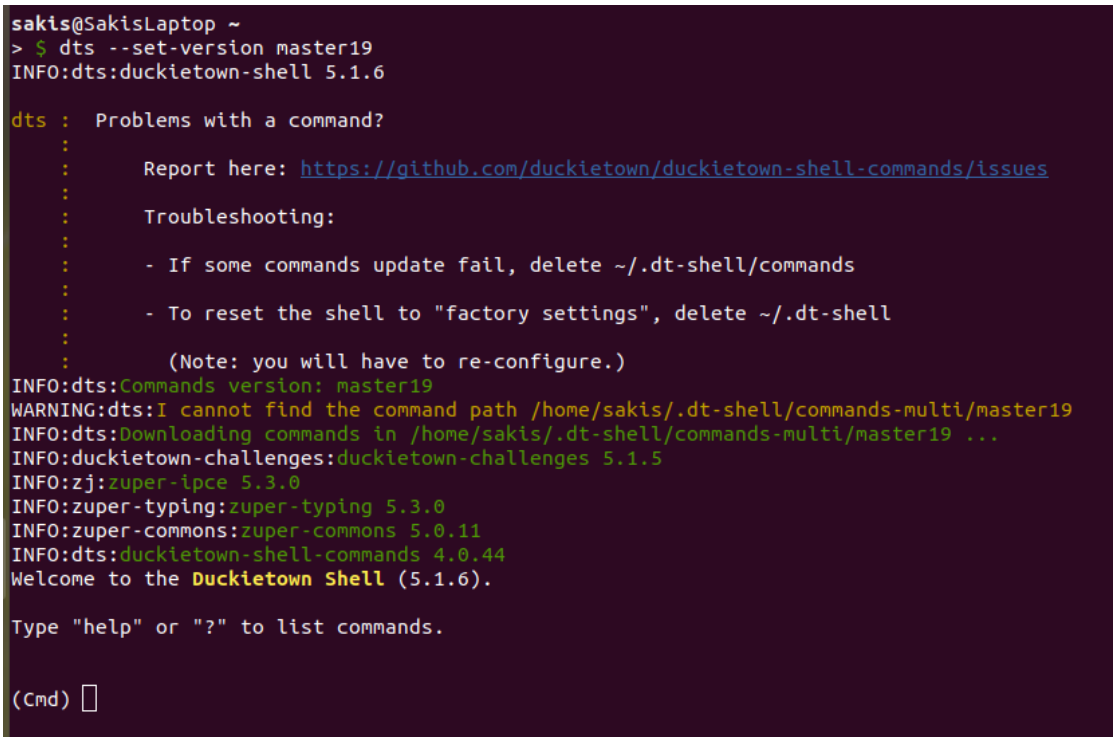
για να διαπιστώσουμε τον κατάλογο που έχει δημιουργηθεί.

```
sakis@SakisLaptop ~
> $ which dts
/home/sakis/.local/bin/dts
```

Η ενεργοποίηση του φλοιού γίνεται γράφοντας

$ dts

Την πρώτη φορά θα πρέπει να ορίσουμε ποιας έκδοσης εντολές πρόκειται να χρησιμοποιήσουμε (daffy ή master19). Εμείς επιλέγουμε master19.



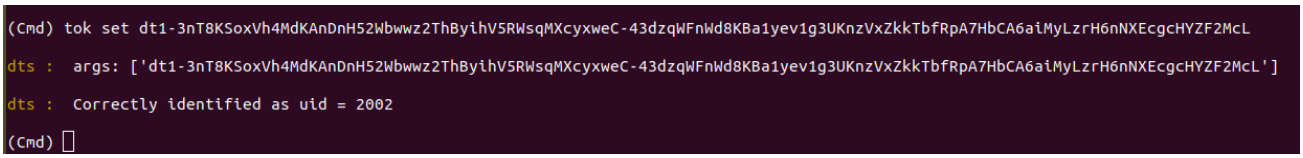
Εικόνα 62 : Στιγμιότυπο επιτυχούς εκκίνησης του Duckietown Shell

Επίσης θα πρέπει να ορίσουμε το διακριτικό ελέγχου ταυτότητας (authentication token) εκτελώντας την εντολή

$ dts tok set

Η παραπάνω εντολή μας οδηγεί το site https://www.duckietown.org/ στο οποίο μετά την επιτυχή μας σύνδεση σε αυτό (με χρήση του username και του password) θα εμφανιστεί το κουπόνι (token) το οποίο και θα πρέπει να αντιγράψουμε. Εάν ήδη το γνωρίζουμε μπορούμε να πληκτρολογήσουμε την εντολή:

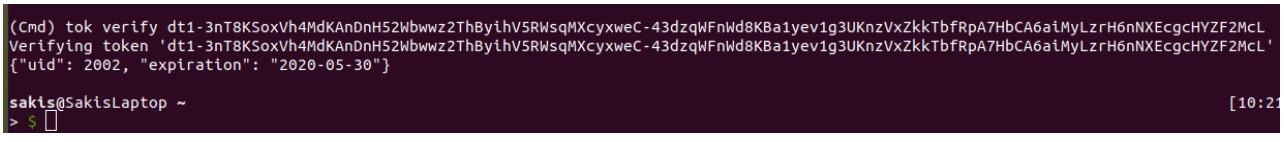
$ dts tok set dt1-YOUR-TOKEN



Εικόνα 63 : Ορισμός του προσωπικού μας "κουπονιού" στο Duckietown shell

Για να επιβεβαιώσουμε ότι το διακριτικό είναι έγκυρο χρησιμοποιούμε την εντολή

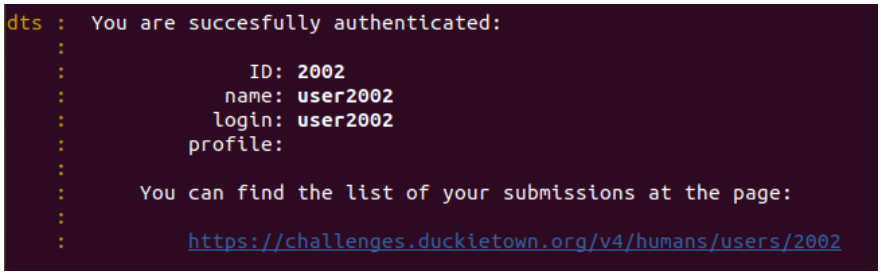
$ dts tok verify dt1-TOKEN-TO-VERIFY



Εικόνα 64 : Στιγμιότυπο επιβεβαίωσης "κουπονιού"

εναλλακτικά μπορούμε να εκτελέσουμε την εντολή

$ dts challenges info



Εικόνα 65 : Στιγμιότυπο εμφάνισης διαπιστευτηρίων

Αν για κάποιο λόγο δεν είναι εφικτή η ενημέρωση των εντολών του duckietown shell διαγράφουμε τον κατάλογο των εντολών ~/.dt-shell/commands

ενώ για να ορίσουμε εκ νέου το ρεπερτόριο των εντολών που θα χρησιμοποιήσουμε (daffy ή maste19) καθώς και το κουπόνι ελέγχου ταυτότητας θα πρέπει να διαγράψουμε τον κατάλογο ~/.dt-shell

Οι βασικές ενέργειες που θα πραγματοποιήσουμε με χρήση εντολών του duckietown shell είναι:

- Εγκατάσταση λογισμικού στις κάρτες sd των Duckiebots ($ dts init_sd_card)
- Η εκκίνηση των γραφικών εργαλείων ελέγχου της κίνησης του Duckiebot του ROS ($ dts start_gui_tools <DUCKIEBOT_NAME>
- Η εκκίνηση της διαδικασίας βαθμονόμησης της κάμερας και των τροχών του Duckiebot
  ($ dts calibrate_Duckiebot <DUCKIEBOT_NAME>

Όλες τις παραπάνω διαδικασίες θα τις περιγράψουμε αναλυτικά στη συνέχεια της τεκμηρίωσης.

6.9 Εγκατάσταση Z shell

Το κέλυφος Z (Z shell, zsh) είναι ένας φλοιός του Unix που μπορεί να χρησιμοποιηθεί σαν ένα αλληλεπιδραστικό κέλυφος χρήστη καθώς και σαν ένας ισχυρός διερμηνέας γραμμής εντολών για προγραμματισμό σεναρίων. Το Zsh μπορεί να θεωρηθεί ένα επεκταμένο κέλυφος Bourne με αρκετές βελτιώσεις περιλαμβάνοντας μεταξύ άλλων χαρακτηριστικά του bash, του ksh και του tcsh.

Για να εγκαταστήσουμε το κέλυφος Z πληκτρολογούμε

$ sudo apt install zsh

Έπειτα αλλάζουμε το χαρακτηριστικό του φλοιού σύνδεσης του χρήστη ορίζοντας το κέλυφος Z.

$ chsh -s /usr/bin/zsh

Μπορούμε να εγκαταστήσουμε το Oh My Zsh το οποίο είναι ένα ανοιχτό, διαμορφωμένο από την κοινότητα του github εργαλείο για την διαμόρφωση του κέλυφους Z.

$ sh -c "$(wget https://raw.githubusercontent.com/robbyrussell/oh-my zsh/master/tools/install.sh -0 -)"

Μπαίνοντας στα περιεχόμενα το αρχείου .zshrc (nano ~/.zshrc) έχουμε τη δυνατότητα να αλλάξουμε το θέμα του Z φλοιού σε bureau. Υπάρχουν και άλλα διαθέσιμα θέματα στην κοινότητα του GitHub (https://github.com/ohmyzsh/ohmyzsh/wiki/themes) όπως το προκαθορισμένο robbyrussell, το af-magic, το afowler και το agnoster).

Ο ορισμός του θέματος γίνεται ορίζοντας μέσα στο αρχείο .zshrc την επιθυμητή τιμή του θέματος στην μεταβλητή ZSH_THEME και αποθηκεύοντας τις αλλαγές.

ZSH_THME="bureau"

τέλος προσθέτουμε και τη γραμμή . ~/.profile στο παραπάνω αρχείο.

Για να μην απαιτείται η εισαγωγή του password με την εντολή sudo ανοίγουμε το αρχείο visudo και αλλάζουμε τη γραμμή

%sudo ALL=(ALL:ALL) ALL

με τη γραμμή

%sudo ALL=(ALL:ALL) NOPASSWD:ALL

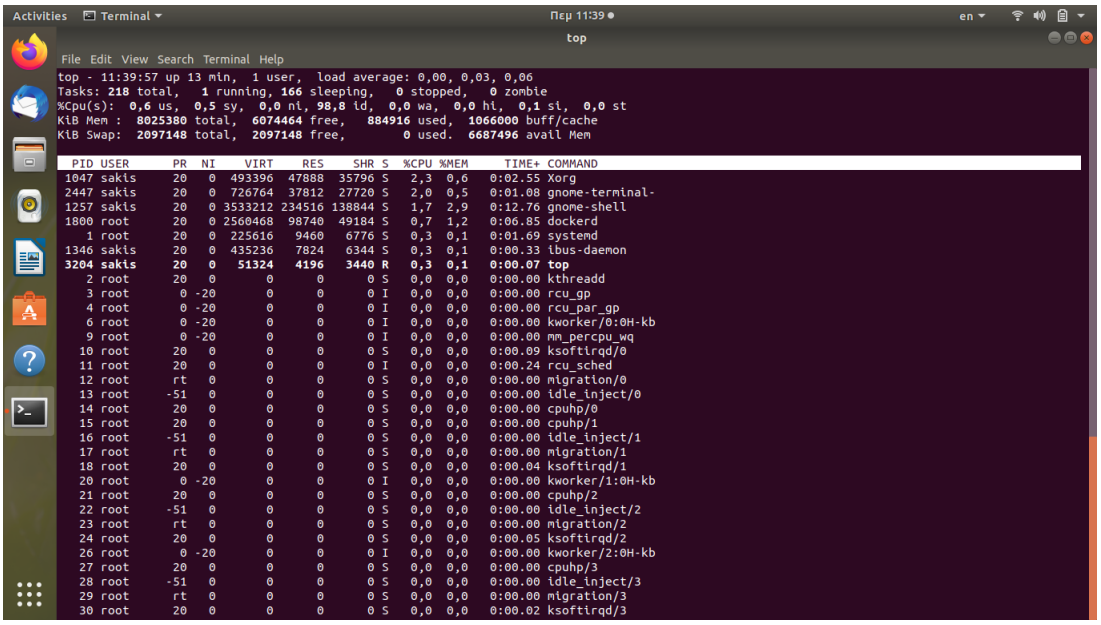
και αποθηκεύουμε τις αλλαγές.

6.10 Εγκατάσταση άλλων χρήσιμων βοηθητικών προγραμμάτων

Υπάρχουν κάποια πολύ χρήσιμα εργαλεία που χρησιμοποιούνται για τη λήψη πληροφοριών σχετικά με το τι συμβαίνει σε ένα σύστημα Linux. Αυτά βοηθητικά προγράμματα μας δείχνουν μέρος της χρήσης και του κορεσμού των πόρων του συστήματος, απεικονίζοντας στην οθόνη ένα στιγμιότυπο συγκεκριμένων ιδιοτήτων σε πραγματικό χρόνο. Και επειδή εκτελούνται σε τη γραμμή εντολών, δίχως την ανάγκη ύπαρξης γραφικής διεπαφής με τον χρήστη, είναι ιδανικά για διακομιστές.

6.10.1 top

Είναι το δημοφιλέστερο βοηθητικό πρόγραμμα που πιθανώς συνοδεύει τη διανομή Ubuntu που έχουμε εγκαταστήσει. Αυτό μας δίνει πληροφορίες σχετικά με την εκτέλεση διεργασιών, το ποσοστό χρήσης του επεξεργαστή, την κατανάλωση μνήμης, τον κωδικό (PID) κάθε διεργασίες, κτλ.

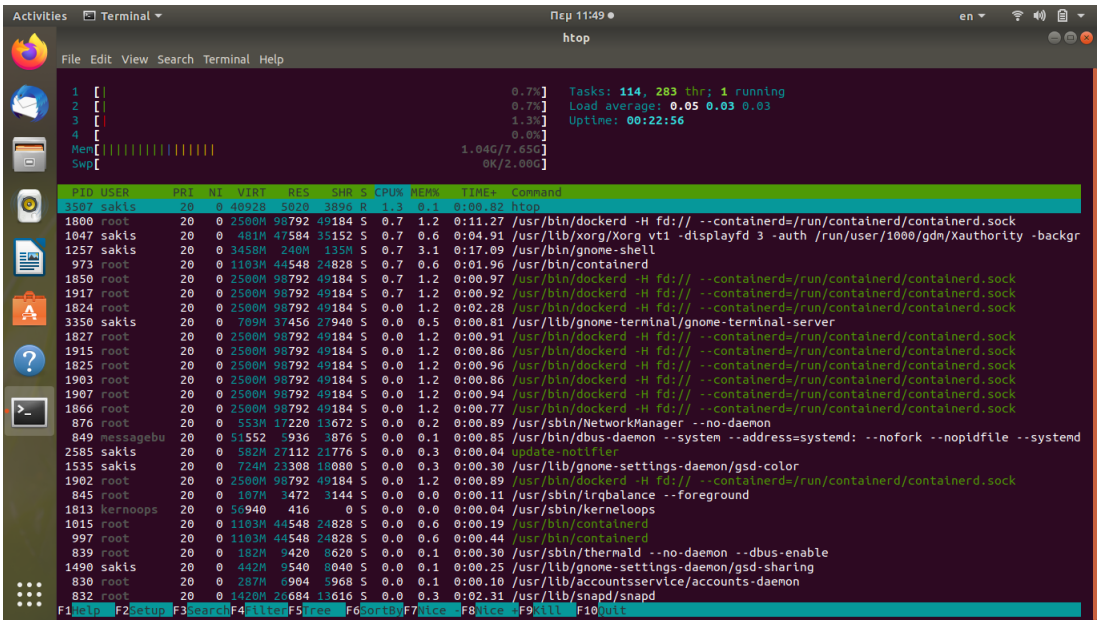


Εικόνα 66 : Στιγμιότυπο εκτέλεσης της εντολής top

- Πατώντας το c εμφανίζει τη γραμμή εντολών
- Πατώντας το V εμφανίζει τις συσχετίσεις ανάμεσα τις εκτελούμενες διεργασίες
- Πατώντας το H εμφανίζει όλα τα threads.

6.10.2 htop

Το htop είναι ένα εναλλακτικό του προηγούμενου προγράμματος εργαλείο το οποίο μας δίνει πληροφορίες σχετικά με την εκτέλεση των διεργασιών με πιο εκτεταμένο τρόπο και με περισσότερες επιλογές εμφανίζοντας με δεδομένα σε πραγματικό χρόνο για την χρήση του κάθε πυρήνα της CPU.



Εικόνα 67 : Στιγμιότυπο εκτέλεσης της εντολής htop

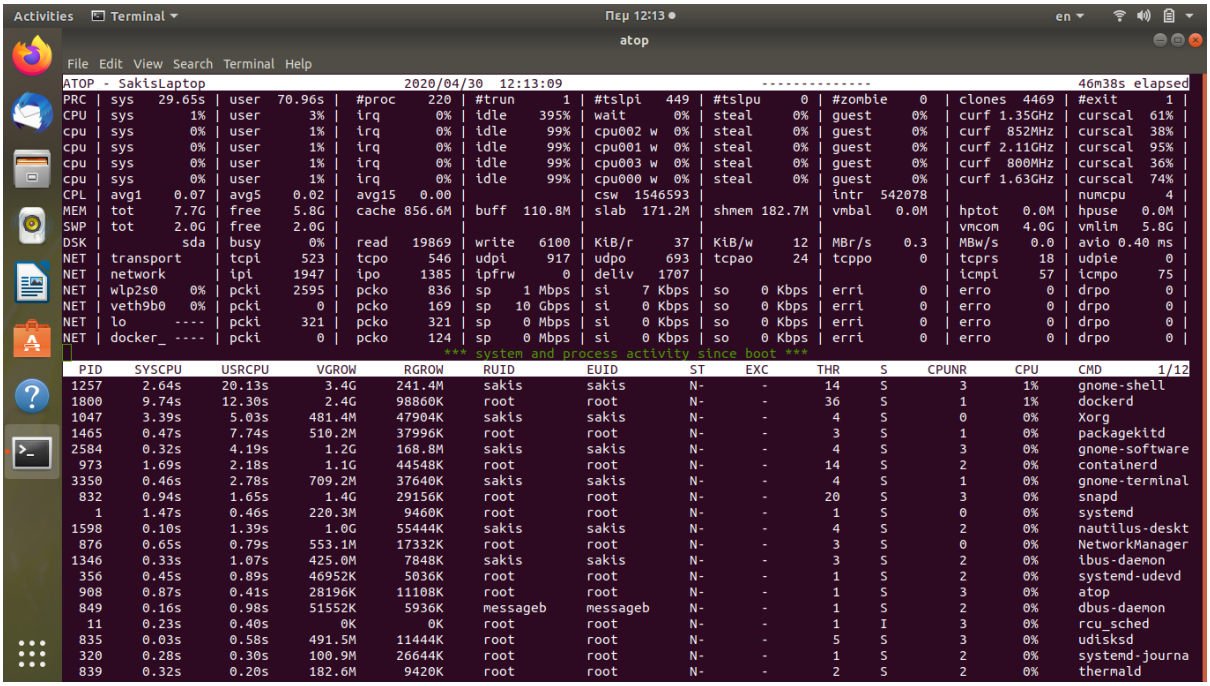
- Επιλέγοντας μια διεργασία με πιέζοντας το πλήκτρο s μας εμφανίζει μια λίστα από τις κλήσεις του συστήματος που πραγματοποιούνται από την επιλεγμένη διεργασία.
- Επιλέγοντας μια διεργασία με πιέζοντας το πλήκτρο l μας εμφανίζει μια λίστα από αρχεία που έγιναν open από την επιλεγμένη διεργασία.
- Πιέζοντας το πλήκτρο M ταξινομεί τις διεργασίες ως προς τη χρήση μνήμης
- Πιέζοντας το πλήκτρο P ταξινομεί τις διεργασίες ως προς τη χρήση της CPU
- Πιέζοντας το πλήκτρο p εμφανίζει ή κάνει απόκρυψη της γραμμής εντολών
- Πιέζοντας το πλήκτρο F5 εμφανίζει τις συσχετίσεις μεταξύ των διεργασιών
- Πιέζοντας το πλήκτρο F2 ρυθμίζουμε αρκετές χρήσιμες επιλογές.

6.10. 3 iotop

Πρόκειται να ένα βοηθητικό πρόγραμμα που εστιάζει στην παρακολούθηση λειτουργιών εισόδου / εξόδου όπως για παράδειγμα η εγγραφή ή η ανάγνωση από τον σκληρό δίσκο.

6.10.4 atop

ακόμα ένα εργαλείο της οικογένειας top για την παρακολούθηση ενός Linux συστήματος το οποίο μας παρέχει περισσότερες πληροφορίες σε σχέση με το top όπως παρακολούθηση της κίνησης του δικτύου, τη χρήση συσκευών εισόδου εξόδου και γενικώς συγχωνεύοντας κατά κάποιον τρόπο τις πληροφορίες των παραπάνω top-like βοηθητικών προγραμμάτων.



Εικόνα 68: Στιγμιότυπο της εκτέλεσης της εντολής atop

Χρήσιμες επιλογές:

- Πιέζοντας το πλήκτρο g μετάβαση σε γενική προβολή
- Πιέζοντας το πλήκτρο m εμφανίζονται στατιστικά τα οποία σχετίζονται με τη μνήμη

- Πιέζοντας το πλήκτρο d εμφανίζονται έξοδοι που σχετίζονται με το σκληρό δίσκο, αν υπάρχουν
- Πιέζοντας το πλήκτρο s εμφανίζονται πολλά στατιστικά τα οποία σχετίζονται με τον χρονοπρογραμματισμό των διεργασιών

Τέλος μας δίνει τη δυνατότητα να αποθηκεύσουμε όλα τα στατιστικά σε συμπιεσμένα δυαδικά αρχεία για μακροχρόνια ανάλυση.

Για να εγκαταστήσουμε τα παραπάνω βοηθητικά προγράμματα, εκτός του top που είναι ενσωματωμένο στη διανομή Ubuntu, προκειμένου να είμαστε σε θέση να εποπτεύουμε το σύστημα μας πληκτρολογούμε :

$ sudo apt install iotop atop htop

Κεφάλαιο 7

Αρχικοποίηση και λειτουργία του Duckiebot

Αφού ολοκληρώσαμε επιτυχώς την εγκατάσταση τόσο των βασικών όσο και των βοηθητικών προγραμμάτων και έχουμε συναρμολογήσει το Duckiebot όπως περιγράψαμε παραπάνω είμαστε πλέον έτοιμοι να εγκαταστήσουμε ό,τι λογισμικό απαιτείται προκειμένου να το καταστήσουμε λειτουργικό. Η εγκατάσταση είναι μία απλή αλλά χρονοβόρα διαδικασία η οποία ολοκληρώνεται σε δύο στάδια: Την εγκατάσταση του λογισμικού στην κάρτα sd του Duckiebot και την διαδικασία αρχικοποίησης που απαιτεί αρκετό χρόνο ώσπου να ολοκληρωθεί.

Το πρώτο στάδιο απαιτεί αρχικά την επιλογή ενός ονόματος που είναι ουσιαστικά το αναγνωριστικό του Duckiebot στο τοπικό μας δίκτυο και γι' αυτό πρέπει να είναι μοναδικό. Για είναι το όνομα του robot έγκυρό θα πρέπει να πληροί τις παρακάτω προδιαγραφές:

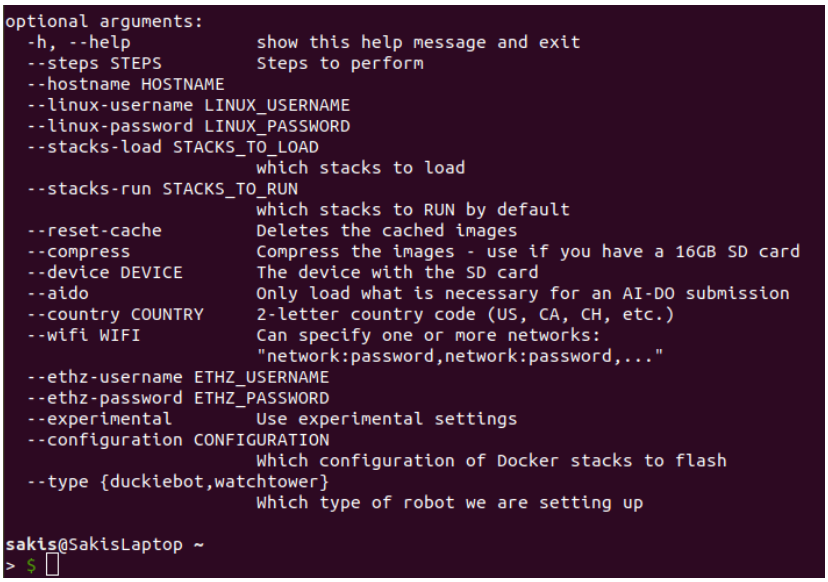
- να περιέχει μόνο πεζά γράμματα και όχι κεφαλαία
- το όνομα να αρχίζει από γράμμα
- να περιέχει μόνο γράμματα, αριθμούς και τον χαρακτήρα της κάτω παύλας (_)

Εμείς αφού όλο το project έχει να κάνει με πάπιες χρησιμοποιήσαμε ως ονόματα των Duckiebots τα ονόματα των χαρακτήρων κινουμένων σχεδίων της Disney της οικογένειας του Donald Duck με μορφή που να ικανοποιούν τις παραπάνω προδιαγραφές για να τονίσουμε τον παιγνιώδη χαρακτήρα του project. Έτσι λοιπόν επιλέξαμε τα παρακάτω ονόματα για τα Duckiebots :

- donaldduck
- daisyduck (αρχικά della)
- daffyduck
- scroogemcduck

Για την εγγραφή όλου του λογισμικού στην κάρτα sd χρησιμοποιούμε την εντολή init_sd_card που μας παρέχει ο duckietown shell αφού προηγουμένως τοποθετήσουμε την κάρτα sd στον usb αναγνώστη καρτών και τον συνδέσουμε σε μια θύρα usb του υπολογιστή μας. Καλό είναι πριν επιχειρήσουμε να εκτελέσουμε την εντολή αρχικοποίησης της κάρτας SD να διαβάσουμε προσεκτικά τις διαθέσιμες επιλογές - παραμέτρους της. Αυτές εμφανίζονται πληκτρολογώντας:

$ dts init_sd_card --help



Εικόνα 69 : Στιγμιότυπο παραμέτρων της εντολής αρχικοποίησης της Micro SD Card.

Οι πιο σημαντικές επιλογές της παραπάνω εντολής είναι οι εξής:

--hostname : Το αναγνωριστικό του robot στο δίκτυο

--linux-username : Προκαθορισμένη τιμή duckιe

--linux-password: Προκαθορισμένη τιμή : quackquack

--wifi : Το όνομα του ασύρματου δικτύου και τον κωδικό του με διάταξη όνομα:κωδικός

--country : Προκαθορισμένη τιμή US

--compress : αν η κάρτα sd έχει χωρητικότητα 16 GB

--type : προκαθορισμένη επιλογή Duckiebot, άλλη επιλογή watchtower (παρατηρητήριο)

Αφού διαβάσουμε προσεκτικά τα παραπάνω είμαστε έτοιμοι να προχωρήσουμε στην αρχικοποίηση της κάρτας sd συνθέτοντας την εντολή:

$ dts init_sd_card --hostname della --country GR --wifi WIND_2.4G_C0E3FF: xxxxxxx,robotics.teicm.gr:xxxxxxxx

Το μόνο που αξίζει να σημειώσουμε στο σημείο αυτό είναι ότι επειδή επιθυμούμε να Duckiebots θα είναι σε θέση να συνδεθούν σε δύο διαφορετικά δίκτυα (του πανεπιστημίου και της οικίας μας) ορίσαμε τα αναγνωριστικά των ασύρματων δικτύων και τους κωδικούς πρόσβασης διαχωρίζοντας τα με κόμμα (,). Εναλλακτικά αν δεν τα είχαμε ορίσει εδώ θα μπορούσαμε μετά το πέρας της διαδικασίας αρχικοποίησης να συνδεθούμε με το Duckiebot (είτε ασύρματα είτε μέσω καλωδίου utp) και να προσθέσουμε το αναγνωριστικό του δεύτερου ασύρματού δικτύου καθώς και τον κωδικό του στο αρχείο /etc/wpa_supplicant/wpa_supplicant.conf που βρίσκεται στο ριζικό κατάλογο του λειτουργικού συστήματος του raspberry pi που φέρει το Duckiebot.

Επίσης αν η κάρτα sd είχε μέγεθος 16 GB θα έπρεπε στην παραπάνω εντολή να προσθέσουμε και την επιλογή --compress για συμπίεση των εγγραφόμενων εικόνων σε αυτήν. Τέλος προσέχουμε κατά τη διαδικασία εκτέλεσης της παραπάνω εντολής να επιλέξουμε το σωστό αποθηκευτικό μέσο στο οποίο θα πραγματοποιηθεί η εγγραφή των δεδομένων (/dev/sdb) λαμβάνοντας υπόψη ότι η παραπάνω διαδικασία διαγράφει όλα τα δεδομένα του αποθηκευτικού μέσου που έχει επιλεγεί.

Αν η διαδικασία ολοκληρωθεί επιτυχώς αποσυνδέουμε τον usb αναγνώστη καρτών από τον υπολογιστή μας και εξάγουμε την κάρτα sd από αυτόν.

7.1 Αρχικοποίησης λειτουργίας Duckiebot

Πριν τοποθετήσουμε την κάρτα sd στο Raspberry pi του Duckiebot πρέπει προηγουμένως να έχουμε φορτίσει στην μπαταρία του Duckiebot καθώς η διαδικασία αρχικοποίησης του Duckiebot θα απαιτήσει αρκετό χρόνο και δεν πρέπει να διακοπεί. Επίσης δεν ενδείκνυται να έχουμε την μπαταρία σε κατάσταση φόρτισης κατά τη διαδικασία αυτή καθώς υπάρχει πιθανότητα η εξωτερική τροφοδοσία να μην παρέχει επαρκές ρεύμα στο Raspberry Pi με αποτέλεσμα αυτό να επανεκκινήσει γεγονός που πιθανότατα θα μας οδηγήσει στην επανάληψης εγγραφής της κάρτας SD.

Τοποθετώντας την κάρτα SD στην ειδική υποδοχή του Raspberry Pi του Duckiebot θα πρέπει να δούμε το πράσινο LED να αναβοσβήνει. Αν δεν συμβαίνει αυτό τότε υπάρχει κάποιο πρόβλημα είτε με την κάρτα SD είτε με το Raspberyy Pi. Μετά από λίγη ώρα το πράσινο και το κόκκινο LED του Raspberry Pi θα αρχίσουν να αναβοσβήνουν εναλλάξ γεγονός που δηλώνει ότι εξάγονται τα απαραίτητα Docker containers. Όταν η χρονοβόρα αυτή διαδικασία ολοκληρωθεί το κόκκινο LED θα σβήσει και το πράσινο να μείνει αναμμένο. Εννοείται όσο η διαδικασία αυτή είναι σε εξέλιξη δεν πρέπει να απενεργοποιήσουμε το Duckiebot.

Για να διαπιστώσουμε αν ολοκληρώθηκε επιτυχώς η διαδικασία αρχικοποίησης του Duckiebot και βρισκόμαστε συνδεδεμένοι στο ίδιο ασύρματο δίκτυο στέλνουμε ένα αίτημα επικοινωνίας σε αυτό μέσω της εντολής ping. Στην περίπτωση μας γράφουμε:

$ ping della.local

```
sakis@SakisLaptop ~
> $ ping della.local
PING della.local (192.168.1.11) 56(84) bytes of data.
64 bytes from della (192.168.1.11): icmp_seq=1 ttl=64 time=106 ms
64 bytes from della (192.168.1.11): icmp_seq=2 ttl=64 time=5.92 ms
64 bytes from della (192.168.1.11): icmp_seq=3 ttl=64 time=7.06 ms
64 bytes from della (192.168.1.11): icmp_seq=4 ttl=64 time=7.48 ms
64 bytes from della (192.168.1.11): icmp_seq=5 ttl=64 time=6.97 ms
64 bytes from della (192.168.1.11): icmp_seq=6 ttl=64 time=6.76 ms
^C
--- della.local ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 9129ms
rtt min/avg/max/mdev = 5.924/23.402/106.214/37.037 ms
```

Εικόνα 70 : Στιγμιότυπο επιτυχούς επικοινωνίας με το Duckiebot

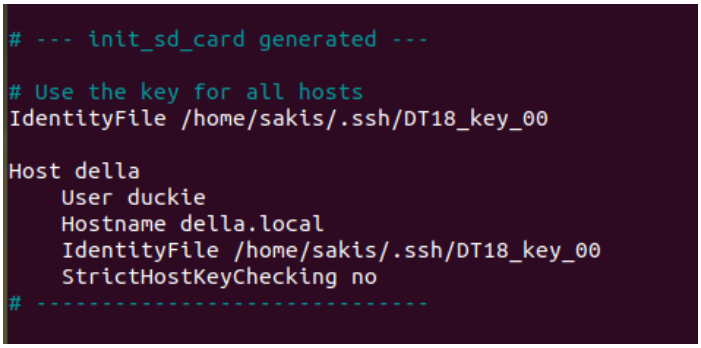
Η παραπάνω εικόνα μας δείχνει την επιτυχημένη προσπάθεια επικοινωνίας με το Duckiebot με το όνομα della καθώς στείλαμε συνολικά 6 πακέτα των 64 bytes το καθένα και παρελήφθησαν όλα χωρίς απώλειες.

Επόμενο στάδιο είναι να επιχειρήσουμε να δημιουργήσουμε ένα κανάλι ασφαλούς σύνδεσης ανάμεσα στον διακομιστή (laptop) και του πελάτη (Duckiebot) μέσω του πρωτοκόλλου ssh (Secure Shell) πληκτρολογώντας :

$ ssh della

Εικόνα 71 : Στιγμιότυπο επιτυχημένης εγκατάστασης απομακρυσμένης διαχείρισης του Duckiebot

Η παραπάνω σύνδεση κανονικά πρέπει να επιτευχθεί χωρίς τη χρήση password. Αν για κάποιο λόγο δεν μπορεί να πραγματοποιηθεί τότε πρέπει να ελέγξουμε το αρχείο ~/.ssh/config αν περιέχει τις ακόλουθες πληροφορίες :



Εικόνα 72 : Στιγμιότυπο περιεχομένου του αρχείου config της απομακρυσμένης διαχείρισης

οι οποίες προστέθηκαν στο αρχείο κατά τη διαδικασία αρχικοποίησης της κάρτας SD. Αν για κάποιο λόγο ενώ υπάρχει το παραπάνω αρχείο και αυτό το περιεχόμενο δεν είναι εφικτή η απομακρυσμένη διαχείριση μπορούμε να δοκιμάσουμε να εκτελέσουμε την εντολή ssh ως εξής:

$ssh duckie@della.local

και σε περίπτωση που μας ζητήσει κωδικό σύνδεσης (password) να δώσουμε το προκαθορισμένο password των Duckiebots που είναι : quackquack

7.1.1 Ασφαλίζοντας το Duckiebot

Από προεπιλογή το κλειδί SSH είναι κοινό για όλα τα Duckiebots. Αυτό πρακτικά σημαίνει ότι όλοι που είναι συνδεδεμένοι στο ίδιο ασύρματο δίκτυο μπορούν να έχουν πρόσβαση σε αυτό. Αν επιθυμούμε να αποτρέψουμε αυτό το γεγονός μπορούμε να αφαιρέσουμε το συγκεκριμένο κλειδί εκτελώντας την εντολή:

$ ssh della rm .ssh/authorized_keys

Μετά από την εκτέλεση της παραπάνω εντολής και της αφαίρεσης του κλειδιού κάθε φορά που θα επιχειρούμε να εγκαταστήσουμε μία ασφαλή σύνδεση με το Duckiebot θα απαιτείται να εισάγουμε τον κωδικό πρόσβασης που ορίσαμε κατά την αρχικοποίηση της κάρτας SD. Υπενθυμίζουμε ότι αν δεν έχουμε ορίσει κωδικό πρόσβασης τότε αυτός θα είναι ο προκαθορισμένος.

7.1.2 Επανεκκίνηση - Τερματισμός Λειτουργίας Duckiebot

Αφού έχουμε αποκαταστήσει το κανάλι απομακρυσμένης διαχείρισης μπορούμε από τον υπολογιστή host να επανεκκινήσουμε στο Duckiebot ή να τερματίσουμε τη λειτουργία του. Για επανεκκίνηση αρκεί να πληκτρολογήσουμε:

duckie@della:~$ sudo reboot

ενώ για να τερματίζουμε στη λειτουργία του γράφουμε

duckie@dell:~$ sudo poweroff

Στο σημείο αυτό να επισημάνουμε ότι θα πρέπει να περιμένουμε περίπου 30 δευτερόλεπτα ώσπου να ολοκληρωθεί η διαδικασία τερματισμού του Duckiebot και μετά να αποσυνδέσουμε τα usb καλώδια του hat και του Raspberry Pi από την μπαταρία. Αν αποσυνδέσουμε τα υκαλώδια τροφοδοσίας νωρίτερα, πριν την ολοκλήρωση του τερματισμού το σύστημα μπορεί να καταστραφεί.

7.1.3 Δικτύωση του Duckiebot

Μετά την επιτυχή σύνδεση και επικοινωνία του Laptop με το Duckiebot γεγονός που πιστοποιεί ότι και οι δύο συσκευές είναι συνδεδεμένες στο ίδιο ασύρματο δίκτυο θα πρέπει να ελέγξουμε αν το Duckiebot έχει πρόσβαση στο διαδίκτυο. Ένας τρόπος για να το πετύχουμε αυτό είναι να επιχειρήσουμε να "κατεβάσουμε" την αρχική σελίδα της google ή απλά να επιχειρήσουμε να στείλουμε ένα αίτημα επικοινωνίας (ping) μέσω του Duckiebot. Για την πρώτη περίπτωση αφού συνδεθούμε μέσω της απομακρυσμένης διαχείρισης στο Duckiebot εκτελούμε

duckie@della:~$ sudo curl google.com

```
duckie@della:~$ sudo curl google.com
<HTML><HEAD><meta http-equiv="content-type" content="text/html;charset=utf-8">
<TITLE>301 Moved</TITLE></HEAD><BODY>
<H1>301 Moved</H1>
The document has moved
<A HREF="http://www.google.com/">here</A>.
</BODY></HTML>
duckie@della:~$
```

Εικόνα 73 : Μεταφόρτωση αρχικής σελίδας της Google στο Duckiebot

ενώ για στη δεύτερη περίπτωση

duckie@della:~$ sudo ping www.google.com

```
duckie@della:~$ sudo ping www.google.com
PING www.google.com (172.217.17.164) 56(84) bytes of data.
64 bytes from sof02s21-in-f164.1e100.net (172.217.17.164): icmp_seq=1 ttl=53 time=40.1 ms
64 bytes from sof02s21-in-f164.1e100.net (172.217.17.164): icmp_seq=2 ttl=53 time=40.7 ms
64 bytes from sof02s21-in-f164.1e100.net (172.217.17.164): icmp_seq=3 ttl=53 time=43.3 ms
64 bytes from sof02s21-in-f164.1e100.net (172.217.17.164): icmp_seq=4 ttl=53 time=42.6 ms
^C
--- www.google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 40.159/41.737/43.376/1.307 ms
duckie@della:~$
```

Εικόνα 74 : Στιγμιότυπο σύνδεσης του Duckiebot με τον διακομιστή της Google

Οι παραπάνω εικόνες φανερώνουν την επιτυχή πρόσβαση του Duckiebot στο Internet. Στην αντίθετη περίπτωση θα πρέπει να ελέγξουμε για την ορθότητα τους το όνομα (SSID) και τον κωδικό του ασύρματου δικτύου έτσι όπως αυτά είναι καταχωρισμένα στο αρχείο /etc/wpa_supplicant/wpa_supplicant.conf. Βέβαια σ' αυτή τη περίπτωση προκύπτει το θέμα με ποιον τον τρόπο μπορούμε να ελέγξουμε τα περιεχόμενα αυτού του αρχείου στο βαθμό που αυτό βρίσκεται στο Raspberry Pi του Duckiebot και εμείς δεν μπορούμε να συνδεθούμε σε αυτό. Σε αυτή τη περίπτωση υπάρχουν δύο τρόποι αντιμετώπισης του προβλήματος.

7.1.4 Τρόποι πρόσβασης στο Duckiebot χωρίς ssh

Ο πρώτος είναι να συνδέσουμε στο Raspberry Pi του Duckiebot μια οθόνη και ένα πληκτρολόγιο αφού κάνουμε login να εξετάσουμε ή και να τροποποιήσουμε τα περιεχόμενα του συγκεκριμένου αρχείου. Υπενθυμίζουμε ότι στο βαθμό που δεν έχουμε αλλάξει τα προκαθορισμένα user name και password κατά τη διαδικασία εγγραφής της κάρτας SD αυτά είναι:

User Name: duckie

Password : quackquack

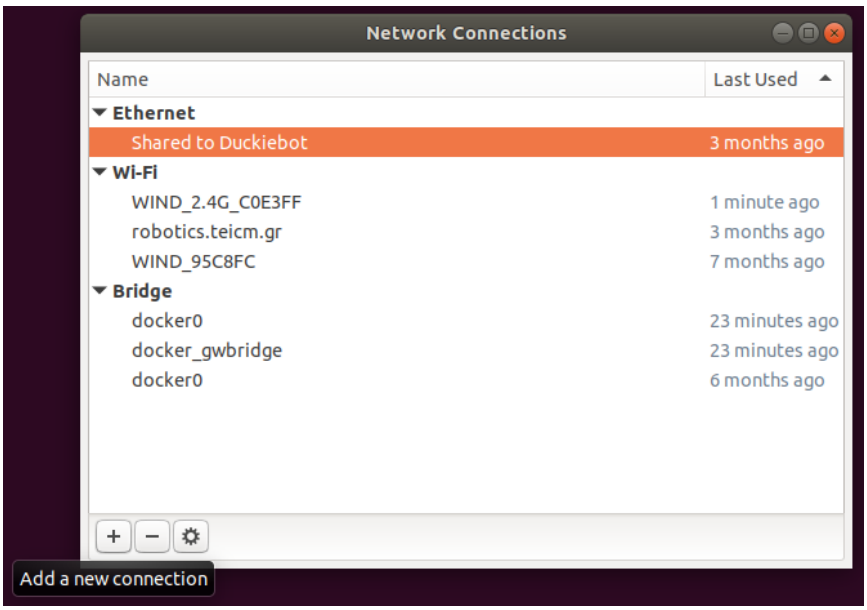
και με αυτά κάνουμε login.

Το μειονέκτημα της παραπάνω διαδικασίας είναι ότι ίσως χρειαστεί να αποσυναρμολογήσουμε μερικώς το Duckiebot προκειμένου να έχουμε πρόσβαση στην θύρα

hdmi για να συνδέσουμε μέσω του κατάλληλου καλωδίου την οθόνη στο Raspberry Pi (αν η οθόνη δεν διαθέτει hdmi είσοδο θα χρειαστούμε και έναν προσαρμογέα hmdi σε VGA ή σε ό,τι άλλη θύρα εισόδου διαθέτει η οθόνη μας)

Ο δεύτερος και ποιο ενδεδειγμένος τρόπος είναι να συνδέσουμε το Duckiebot με το Laptop μέσω καλωδίου ethernet. Ειδικά στις περιπτώσεις όπου το ασύρματο δίκτυο στο οποίο επιχειρεί να συνδεθεί το Duckiebot προστατεύεται επιπρόσθετα με εξειδικευμένα πρωτόκολλα ασφαλείας όπως για παράδειγμα το PEAP (Protected Extensible Protocol) και είναι δύσκολο "περαστούν" αυτά τα διαπιστευτήρια στο Duckiebot μπορούμε να γεφυρώσουμε τη σύνδεση μεταξύ του φορητού υπολογιστή μας (ο οποίος έχει πρόσβαση στο διαδίκτυο) και του Duckiebot. Για υλοποιήσουμε αυτή τη σύνδεση - γέφυρα πρέπει να κάνουμε τις εξής ενέργειες:

1. Συνδέουμε το laptop στο ασύρματο δίκτυο που επιθυμούμε
2. Συνδέουμε το Duckiebot με το Laptop μέσω καλωδίου Ethernet
3. Ανοίγουμε το παράθυρο Network Connections (nm-connection-editor)

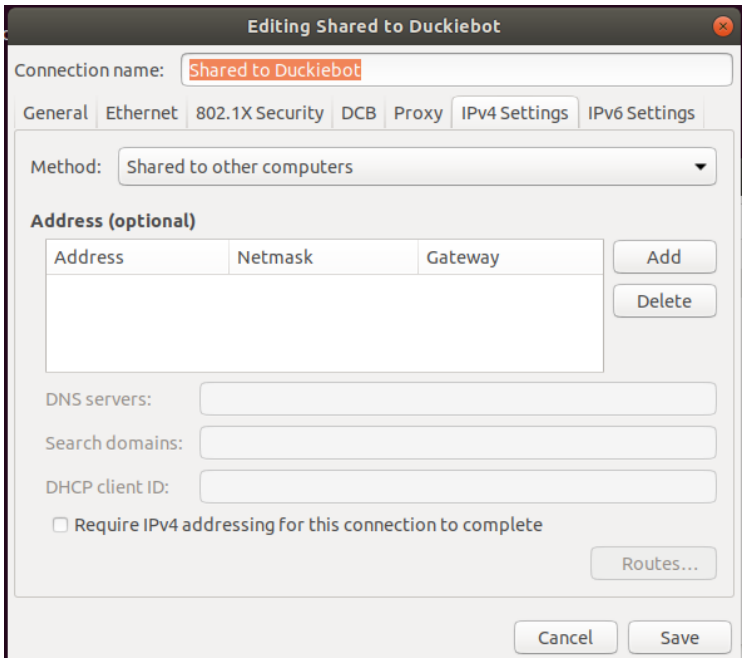


Εικόνα 75 : Δημιουργία Σύνδεσης - Γέφυρας ανάμεσα στο Laptop και το Duckiebot με καλώδιο UTP (βήμα 1)

4. Πατάμε στο πλήκτρο +

5. Το πεδίο Connection Name συμπληρώνουμε : Shared to Duckiebot

6. Στην καρτέλα IPv4 Settings στο πεδίο Method επιλέγουμε Shared to other computers



Εικόνα 76 : Στιγμιότυπο δημιουργίας σύνδεσης Laptop-Duckiebot (βήμα 2)

7. Πατάμε στο κουμπί save

Επίσης είναι εφικτό στο βαθμό που ο φορητός μας υπολογιστής είναι συνδεδεμένος στο διαδίκτυο να αντλούμε docker images μέσω του laptop και να τις στέλνουμε μέσω του πρωτοκόλλου ssh στο Duckiebot με την παρακάτω εντολή:

$ docker save duckietown/image_name | ssh -C hostname docker load

7.2 Ρύθμιση των εργασιών του Docker μέσω του Portainer

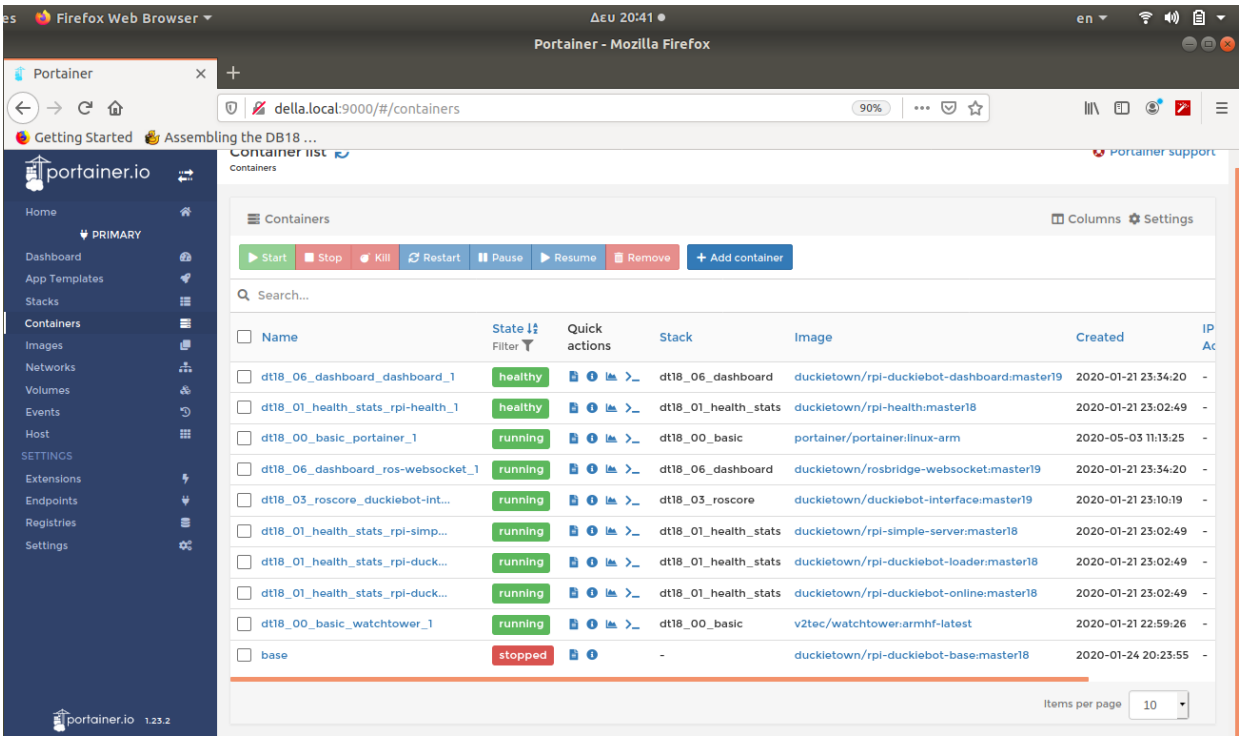
Για την δημιουργία, διαχείριση και συντήρηση του περιβάλλοντος Docker υπάρχει ένα εύχρηστο και φιλικό προς τον προγραμματιστή λογισμικό. Το Portainer αναπτύχθηκε για μας παρέχει μέσα από ένα web interface μία λεπτομερή επισκόπηση των τεκταινόμενων του Docker και να μας επιτρέπει να διαχειριζόμαστε τα κιβώτια (containers), τις εικόνες (images), τα δίκτυα (networks) και τους τόμους (volumes) του Docker. Για μπούμε στο web interface του Portainer αρκεί να εκκινήσουμε έναν φυλλομετρητή (web browser) και να πληκτρολογήσουμε στην γραμμή διεύθυνσης

http://hostname.local:9000/#/containers

δηλαδή στην περίπτωση μας που το όνομα του Duckiebot είναι della πληκτρολογούμε

http://della.local:9000/#/containers

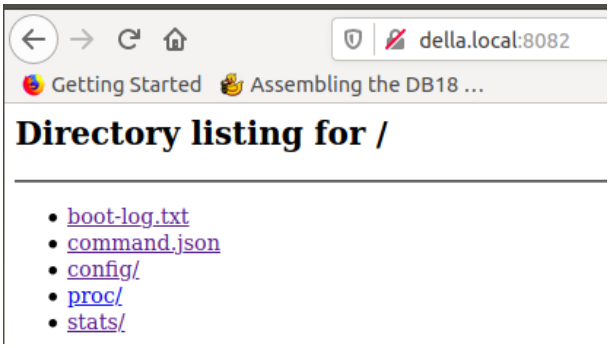
είναι αυτονόητο ότι όλα τα παραπάνω έχουν νόημα μόνο που στο βαθμό ο φορητός υπολογιστής και το Duckiebot είναι συνδεδεμένα στο ίδιο ασύρματο τοπικό δίκτυο.



Εικόνα 77 : Εμφάνιση κατάστασης των Docker Containers στο Duckiebot μέσω του Portainer

Στο σημείο αυτό θα πρέπει να προσέξουμε αν κάποια containers έχουν τη σήμανση "unhealthy". Αν συμβαίνει αυτό θα πρέπει να το διορθώσουμε πριν συνεχίσουμε. Η σήμανση αυτή είναι το αποτέλεσμα ελέγχου που διενεργεί στο υλικό του Duckiebot το container duckietown/rpi-health.

Για να έχουμε πρόσβασή στα αρχεία του Duckiebot μέσω του Portainer επικοινωνούμε με αυτό στη θύρα 8082

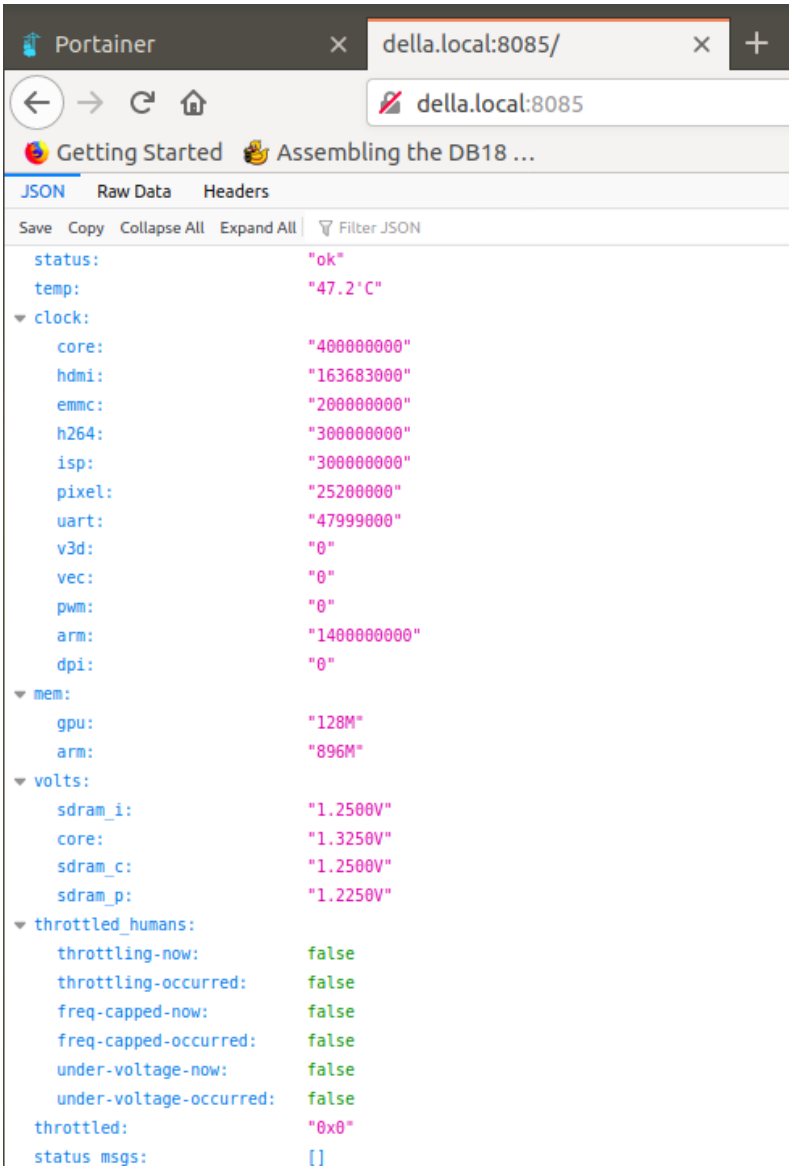


Εικόνα 78 : Πρόσβαση στο σύστημα αρχείων του Duckiebot μέσω του Portainer

ή εναλλακτικά πληκτρολογούμε στη γραμμή εντολών ssh della.local ls /data

Για να δούμε επιπλέον λεπτομερείς πληροφορίες για την κατάσταση του υλικού κάνουμε κλικ στο εικονίδιο που βρίσκεται δεξιά της ένδειξης "unhealthy" ή εναλλακτικά πληκτρολογούμε σε μία άλλη ξεχωριστή καρτέλα του web browser στη γραμμή διεύθυνσης http://hostname.local:8085/ (http://della.local:8085/) οπότε και εμφανίζεται μία λεπτομερής

αναφορά - καταγραφή της κατάστασης λειτουργίας του υλικού του raspberry pi που φέρει το Duckiebot όπως φαίνεται στην εικόνα που ακολουθεί



Εικόνα 79 : Λεπτομερής περιγραφή της κατάστασης του υλικού του Duckiebot μέσω Portainer

7. 3 Έλεγχος λειτουργίας daemon

Για να διαπιστώσουμε αν το long-running πρόγραμμα του Docker, ο daemon, λειτουργεί αποτελεσματικά στο Duckiebot θα δημιουργήσουμε μία απλή εργασία και θα του αναθέσουμε την εκτέλεση της η οποία θα λάβει χώρα στο Duckiebot. Αρχικά θα "κατεβάσουμε" ένα αντίγραφο του πακέτου rpi-Duckiebot-simple-python.git στον φορητό υπολογιστή μας.

$ git clone https://github.com/duckietown/rpi-duckiebot-simple-python.git

έπειτα θα μπούμε στον κατάλογο rpi-Duckiebot-simple-python

$ cd rpi-duckiebot-simple-python

και θα αναθέσουμε στο πρόγραμμα docker daemon να δομήσει ένα container στο Duckiebot με βάση την εικόνα που του στέλνουμε

docker -H della.local build -t my-simple-python-program .

Τέλος θα πρέπει να εκτελέσει το container που δημιούργησε στο Duckiebot

$ docker -H della.local run -it --network=host my-simple-python-program

Το αποτέλεσμα της παραπάνω εκτέλεσης είναι η εμφάνιση ενός μηνύματος ότι εκτελείται στο Duckiebot με το αναγνωριστικό della.

```
sakis@SakisLaptop ~/rpi-duckiebot-simple-python/rpi-duckiebot-simple-python
> $ docker -H della.local run -it --network=host my-simple-python-program
I am executing on the host della

sakis@SakisLaptop ~/rpi-duckiebot-simple-python/rpi-duckiebot-simple-python
> $
```

Εικόνα 80 : Εκτέλεση προγράμματος στο Duckiebot μέσω Docker

Αν θελήσουμε να εξετάσουμε λίγο πιο προσεκτικά τι περιλαμβάνει το πακέτο rpi-Duckiebot-python.git που κατεβάσαμε και χρησιμοποιήσαμε για τον έλεγχο της αποτελεσματικής λειτουργίας του Docker daemon εύκολα διαπιστώνουμε ότι αυτό αποτελείται από δύο βασικά αρχεία:

- το Dockerfile
- το πρόγραμμα hello.py

Ο πηγαίος κώδικας του hello.py εισάγει το Module socket της Python που μας επιτρέπει την πρόσβαση στην BSD socket διεπαφή για να αντλήσουμε έπειτα το αναγνωριστικό όνομα του Duckiebot στο δίκτυο με τη μέθοδο gethostname() και να καταχωρίσουμε στη μεταβλητή hostname και να το εμφανίσουμε στη συνέχεια.

```
1  #!/usr/bin/env python2
2  import socket
3
4  hostname = socket.gethostname()
5  print('I am executing on the host %s' % hostname)
```

Εικόνα 81 : Περιεχόμενο αρχείου Hello.py

Το dockerfile καθορίζει όλες τις απαραίτητες παραμέτρους για την εκτέλεση του προγράμματος hello.py ως container στο raspberry pi του Duckiebot. Δηλαδή αρχικά ορίζει

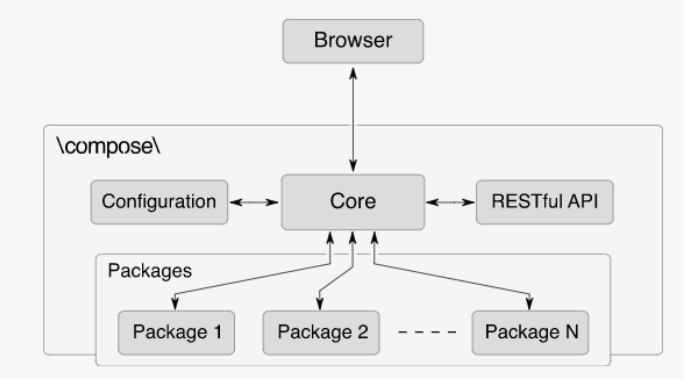
την βασική εικόνα rpi-raspbian, εγκαθιστά όλες τις απαραίτητες "εξαρτήσεις" που απαιτούνται για να εκτελεστεί το πρόγραμμα, αντιγράφει το αρχείο hello.py που περιέχει τον πηγαίο κώδικα στον κατάλογο project, εκχωρεί το δικαίωμα της εκτέλεσης στο αρχείο αυτό και τέλος ορίζει το πρόγραμμα ως προεπιλογή.

```
1  # The base image
2  FROM resin/rpi-raspbian
3
4  ### Installation of dependencies
5  ENV DEBIAN_FRONTEND=noninteractive
6  RUN apt-get update
7  RUN apt-get install -y python
8
9
10 # Installation of our program
11 COPY hello.py /project/hello.py
12 RUN chmod +x /project/hello.py
13
14 # Setting the program as the default
15 CMD /usr/bin/python /project/hello.py
```

Εικόνα 82 : Περιεχόμενο Dockerfile

7.4 Η πλατφόρμα διαχείρισης περιεχομένου \compose\

Ο έλεγχος της λειτουργίας των Duckiebots μπορεί να επιτευχθεί μέσα από μία πλατφόρμα διαχείρισης περιεχομένου (CMS) το \compose\ που παρέχει υπηρεσίες και λειτουργικότητες για web εφαρμογές [21]. Συγκεκριμένα το \compose\ βασίζεται στην ιδέα των εγκατεστημένων πακέτων και το βασικό ενσωματωμένο πακέτο core είναι υπεύθυνο για τη διαχείριση των άλλων πακέτων (third-party packages) καθώς μας επιτρέπει να εγκαταστήσουμε, αν αφαιρέσουμε, να ενημερώσουμε, να ενεργοποιήσουμε και να απενεργοποιήσουμε πακέτα αλλά και μεμονωμένα στοιχεία όπως API End-points ή Σελίδες απευθείας από το πρόγραμμα περιήγησης (browser). Η αρχιτεκτονική του \compose\ περιγράφεται στο παρακάτω διάγραμμα.



Εικόνα 83 : Διάγραμμα αρχιτεκτονικής του \compose

Δεν υπάρχουν πολλά πράγματα που μπορούμε να δούμε ή να κάνουμε με το compose αν δεν έχουμε εγκαταστήσει ή δεν έχουμε δημιουργήσει πακέτα. Το βασικό πακέτο core περιλαμβάνει όλες τις λειτουργίες που απαιτούνται για τη διαχείριση των πρόσθετων πακέτων. Η κατηγορία Configuration παρέχει ένα πακέτο ρυθμίσεων τόσο για το βασικό πακέτο (core) όσο και για τα πρόσθετα πακέτα. Τέλος η ενότητα RESTfulAPI προσφέρει έναν εύκολο τρόπο για τα πακέτα να εξάγουν τελικές εφαρμογές API που να είναι προσβάσιμες μέσω του πρωτοκόλλου HTTP.

Ο προτεινόμενος τρόπος εγκατάστασης του compose είναι χρησιμοποιώντας το Docker. Συνεπώς θα πρέπει να έχουμε εγκαταστήσει το Docker στον υπολογιστή μας πριν ξεκινήσουμε αυτή τη διαδικασία.

- Βήμα 1ο : Να δημιουργήσουμε ένα Docker image (Προαιρετικό)

Αν δεν σχεδιάζουμε να τροποποιήσουμε το \compose\ αλλά ενδιαφερόμαστε να εξελίξουμε μία δική μας εφαρμογή χρησιμοποιώντας το \compose\ μπορούμε να παραλείψουμε αυτό το βήμα και να χρησιμοποιήσουμε μία "έτοιμη" εικόνα (Image) από το DuckerHub. Σε διαφορετική περίπτωση θα πρέπει να δημιουργήσουμε μόνοι μας το Docker image και θα πρέπει να κλωνοποιήσουμε το αποθετήριο τοπικά εκτελώντας τις εντολές:

$ cd ~

$ git clone https://github.com/afdaniele/compose

$ cd ~/compose/

Οι παραπάνω εντολές θα έχουν ως αποτέλεσμα να "τραβήξουν" την νεότερη έκδοση του \compose\ στον κατάλογο compose του home directory. Για να "μπούμε" μέσα στο αποθετήριο που περιλαμβάνει τα Dockerfiles πληκτρολογούμε

$ cd ./docker/master ή

$ cd ./docker/arm32v7/ για υπολογιστές με επεξεργαστή Arm (πχ. Raspberry Pi)

Έπειτα μπορούμε να δημιουργήσουμε το docker image εκτελώντας την παρακάτω εντολή

$ docker build -t afdaniele/compose:latest ./

και πληκτρολογώντας

$ docker images

βλέπουμε σε μία γραμμή την εικόνα Docker με το όνομα : afdaniele/compose

- Βήμα 2ο : Να αντλήσουμε/κατεβάσουμε το Docker image

Αν δεν έχουμε δημιουργήσει την εικόνα Docker με τις παραπάνω εντολές μπορούμε να την "τραβήξουμε" απευθείας από το DockerHub με την εντολή

$ docker pull afdaniele/compose:latest

- Βήμα 3ο : Να εκτελέσουμε το \compose\.

Για να εκκινήσουμε το \compose\ πληκτρολογούμε :

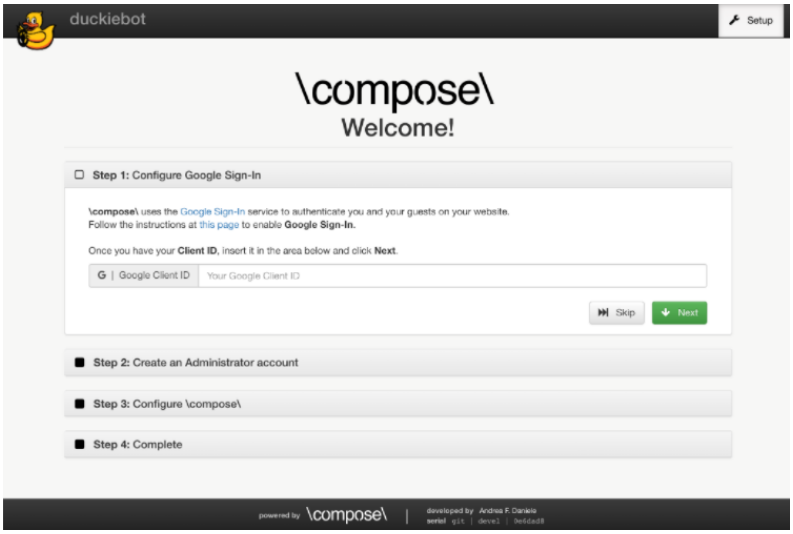
$ docker run -itd -p 80:80 afdaniele/compose:latest

7.4.1 Διαχείριση λειτουργιών του Duckiebot μέσω του Dashboard (\compose\)

Για να ενεργοποιήσουμε το web περιβάλλον του /compose/ αρκεί να εκκινήσουμε έναν φυλλομετρητή και να πληκτρολογήσουμε στη γραμμή διεύθυνσης

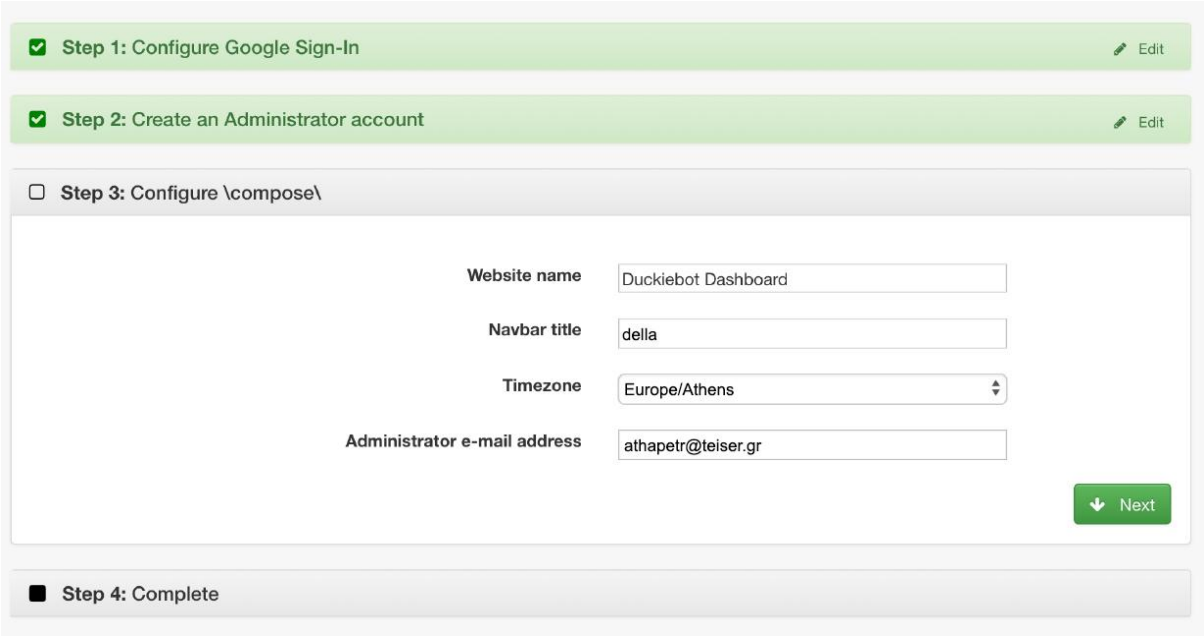
http://hostname_Duckiebot.local δηλαδή στην περίπτωση μας http://donald.local

Την πρώτη φορά που θα θελήσουμε να χρησιμοποιήσουμε το \compose\ θα εμφανιστεί η παρακάτω οθόνη όπου θα πρέπει να το διαμορφώσουμε εισάγοντας κάποιες αναγκαίες πληροφορίες.



Εικόνα 84: Διαμόρφωση του \compose\ (βήμα 1)

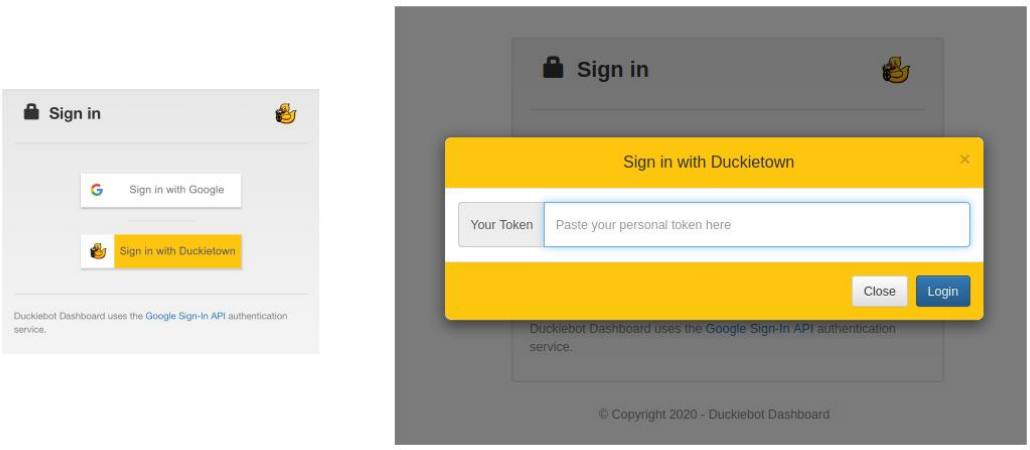
Το /compose/ χρησιμοποιεί λογαριασμούς της Google για να ταυτοποιήσει τους χρήστες που συνδέονται σε αυτό. Εμείς όμως θα χρησιμοποιήσουμε για τη διαδικασία "αυθεντικοποίησης" το κουπόνι που μας προμηθεύει η ιστοσελίδα www.duckietown.org. Συνεπώς παραλείπουμε τα πρώτα δύο βήματα και μεταβαίνουμε στο τρίτο όπου θα πρέπει να εισάγουμε κάποιες πληροφορίες όπως η ώρα ζώνης στην οποία ανήκουμε και ένα email.



Εικόνα 85 : Διαμόρφωση \compose\ (βήμα 3)

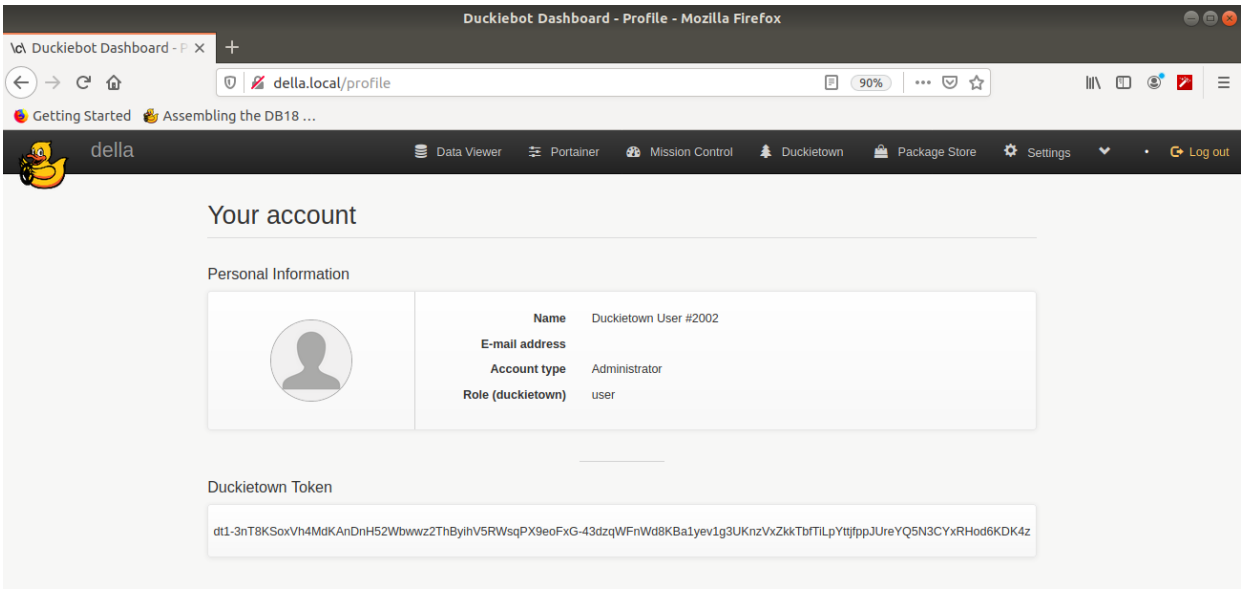
Αυτά τα δεδομένα μπορούν να ενημερωθούν και αργότερα από τις ρυθμίσεις (Settings) του /compose/. Τέλος μεταβαίνουμε στο βήμα της ολοκλήρωσης της διαδικασίας αρχικοποίησης του πίνακα ελέγχου του \compose\ όπου και επιλέγουμε Finish. Τώρα είμαστε έτοιμοι να χρησιμοποιήσουμε το \compose\ για να ελέγξουμε τις βασικές λειτουργίες του Duckiebot. Κατά την είσοδο μας στο περιβάλλον του θα μας ζητήσει να συνδεθούμε είτε χρησιμοποιώντας ως διαπιστευτήρια είτα τα στοιχεία του λογαριασμού της Google είτε αυτά του Duckietown. Εμείς φυσικά θα επιλέξουμε να συνδεθούμε με το token από τη σελίδα Duckietown το οποίο και θα βρούμε στην παρακάτω διεύθυνση :

https://www.duckietown.org/site/your-token



Εικόνα 86 : Σύνδεση στο περιβάλλον του \compose\ μέσω διαπιστευτηρίων Duckietown

Αν η διαδικασία σύνδεσης είναι επιτυχής θα εμφανιστεί η παρακάτω εικόνα

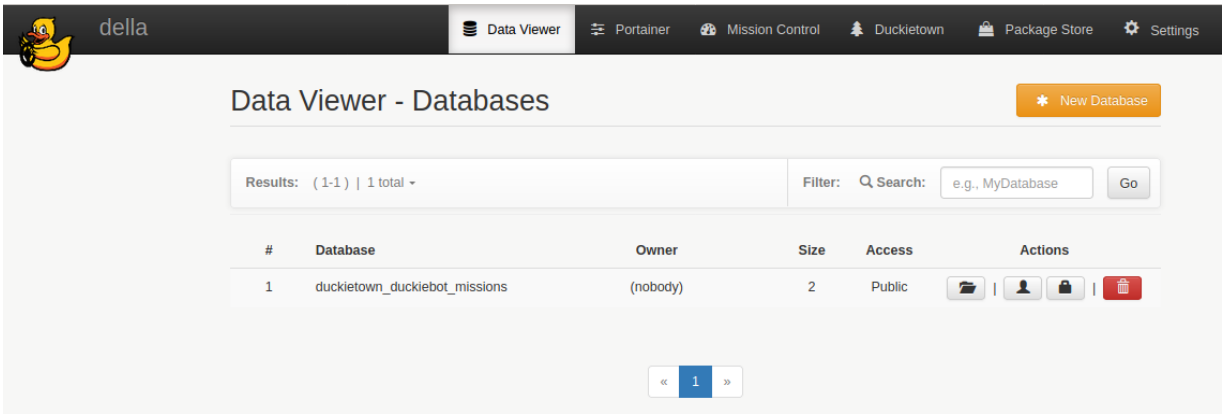


Εικόνα 87 : Επιτυχής σύνδεση στο web-interface του \compose

όπως θα παρατηρήσετε η μπάρα που εμφανίζεται στο πάνω μέρος έχει αριστερά το hostname του Duckiebot (della στην περίπτωση μας) και δεξιότερα ένα μενού με συντομεύσεις επιλογών - σελίδων :

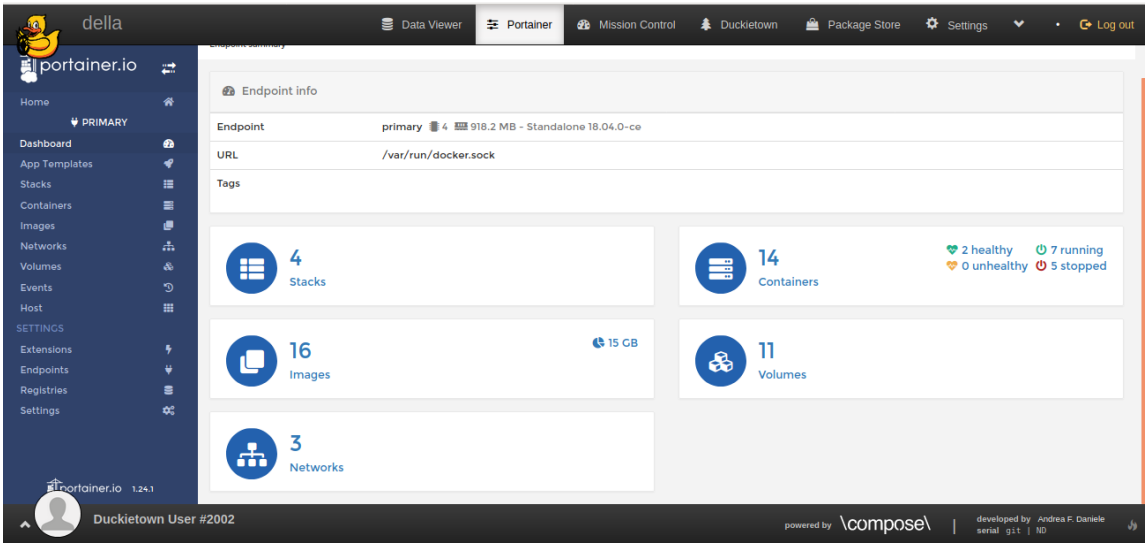
- Data Viewer
- Portainer
- Mission Control
- Duckietown
- Package Store
- Settings
- Log out

Συνοπτικά θα αναφέρουμε ότι επιλέγοντας τη σελίδα Data Viewer θα μας εμφανίζει τη βάση δεδομένων των αποστολών του Duckiebot που έχουμε καταχωρίσει σε αυτήν.



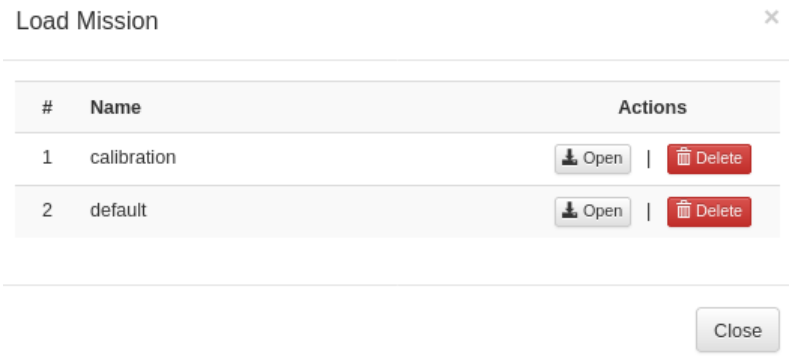
Εικόνα 88 : Καρτέλα με τις διαθέσιμες "αποστολές" του Duckiebot

Με την επιλογή Portainer μας μεταβαίνει στο περιβάλλον του Portainer όπως το γνωρίσαμε παραπάνω.



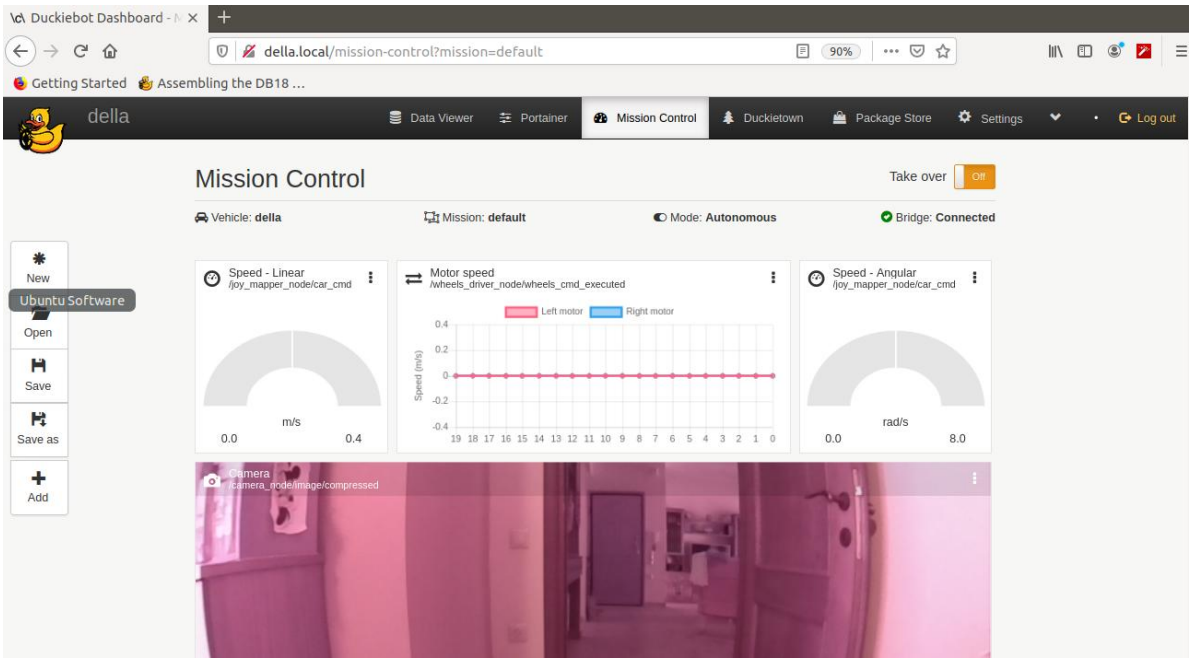
Εικόνα 89 : Μετάβαση στο περιβάλλον του Portainer μέσω \compose

Αν επιλέξουμε Mission Control μεταβαίνουμε σε μία σελίδα που αρχικά καλούμαστε να επιλέξουμε ποια αποστολή θέλουμε να "ανοίξουμε" από δύο διαθέσιμες. Η πρώτη έχει να κάνει με τη διαδικασία βαθμονόμησης του Duckiebot ενώ η δεύτερη είναι η προκαθορισμένη αποστολή που μας δίνει όπως θα δούμε παρακάτω δύο δυνατότητες.



Εικόνα 90: Διαθέσιμες "αποστολές" για να "φορτωθούν" στο Duckiebot

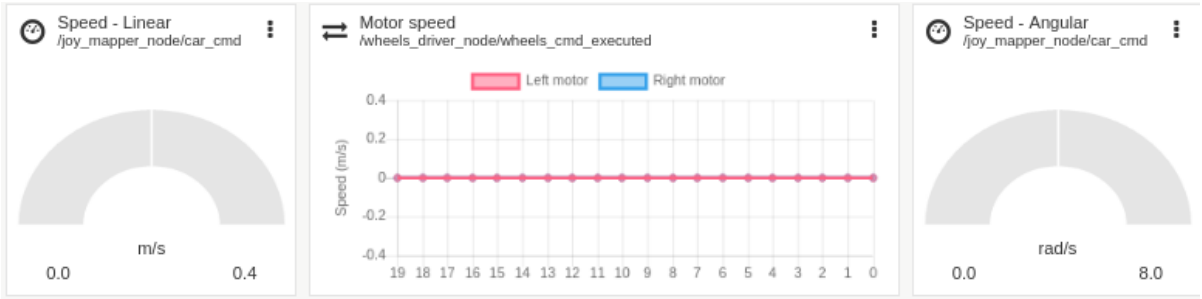
Επιλέγοντας την default αποστολή μεταβαίνουμε στο περιβάλλον της παρακάτω εικόνας



Εικόνα 91 : Στιγμιότυπο από την καρτέλα Mission Control του \compose\ έχοντας επιλέξει την default "αποστολή"

Ξεκινώντας από επάνω βλέπουμε το hostname του Duckiebot - οχήματος (della), την αποστολή που έχουμε επιλέξει (default), την κατάσταση ή τον τρόπο λειτουργίας του Duckiebot (Autonomous) και τέλος την ένδειξη Bridge που μας ενημερώνει για το αν το DC motor hut είναι συνδεδεμένο(connected) ή όχι. Αν η ένδειξη είναι "Closed" αυτό σημαίνει ότι κάτι πήγε στραβά με την εκκίνηση της υπηρεσίας websocket του ROS. Σ' αυτή τη περίπτωση καλό είναι να κάνουμε μία επανεκκίνηση του Raspberry Pi μέσω της απομακρυσμένης διαχείρισης (ssh). Ακριβώς πάνω από αυτή την ένδειξη υπάρχει ένας διακόπτης με την ένδειξη Take over ο οποίος βρίσκεται σε κατάσταση off. Τη λειτουργία αυτού του διακόπτη θα περιγράψουμε στις επόμενες γραμμές.

Ακριβώς από κάτω απεικονίζονται τρεις γραφικές παραστάσεις όπως φαίνονται στο παρακάτω στιγμιότυπο:



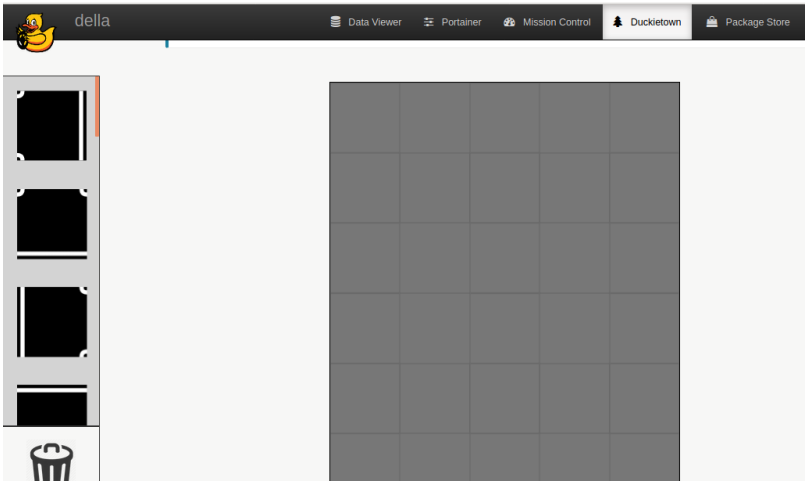
Εικόνα 92: Γραφικές παραστάσεις γραμμικής και γωνιακής ταχύτητας και ταχύτητας των κινητήρων

Η πρώτη από αριστερά αναπαριστά γραφικά τη γραμμική ταχύτητα του Duckiebot μία δεδομένη χρονική στιγμή. Στη μέση συναντούμε μία γραφική παράσταση της ταχύτητας των δύο κινητήρων του Duckiebot και τέλος στα δεξιά απεικονίζεται γραφικά η γωνιακή ταχύτητα με την οποία κινείται το Duckiebot μία συγκεκριμένη χρονική στιγμή. Όλες αυτές

οι ενδείξεις μεταβάλλονται δυναμικά καθώς κινείται το Duckiebot για να μας κατατοπίζουν με γραφικό τρόπο για την ταχύτητα (γραμμική και γωνιακή) του Duckiebot συνεχώς. Ακριβώς από κάτω βλέπουμε την εικόνα που μας στέλνει το Duckiebot μέσω της κάμερας του.

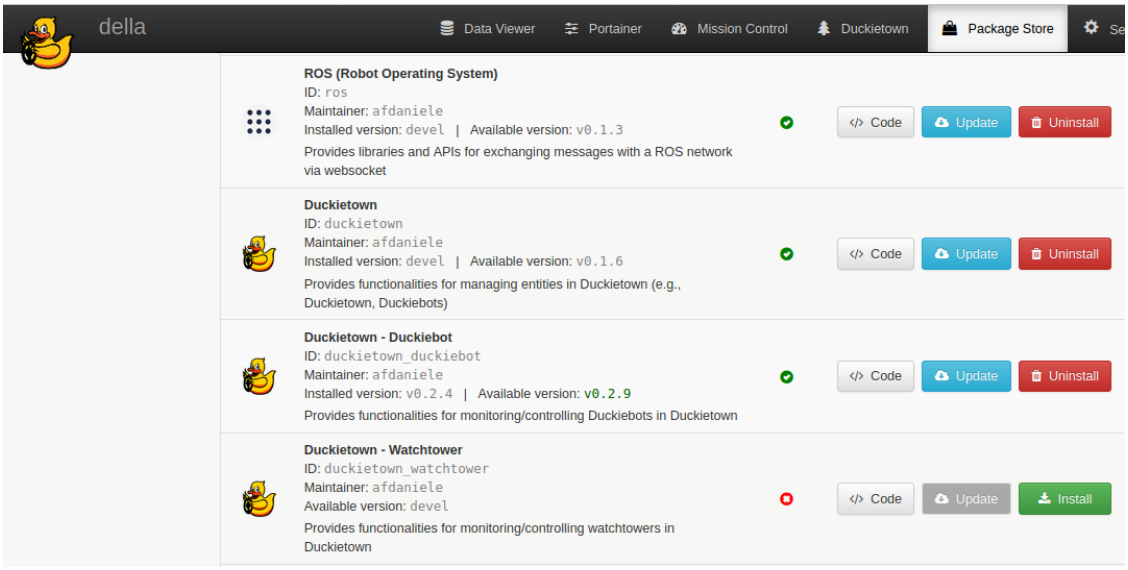
Εικόνα 93 : Ληφθείσα εικόνα από το Duckiebot

Συνεχίζοντας τη γνωριμία μας με το περιβάλλον του \compose\ η επιλογή Duckietown μας ανοίγει μία σελίδα με ένα πειραματικό περιβάλλον για να σχεδιάσουμε την πόλη (Duckietown) προσφέροντας μας μία στήλη με πλακίδια τα οποία τα εναποθέτουμε στο πλέγμα που εμφανίζεται στο κέντρο της σελίδας με την τεχνική drag and drop.



Εικόνα 94: Περιβάλλον σχεδιασμού του χάρτη της πόλης

Η επόμενη επιλογή Package Store μας ενημερώνει για τα πακέτα είναι εγκατεστημένα στο Duckiebot και μας προσφέρει τις δυνατότητες εγκατάστασης, απεγκατάστασης και ενημέρωσης πακέτων μέσα από ένα πολύ εύχρηστο web περιβάλλον.

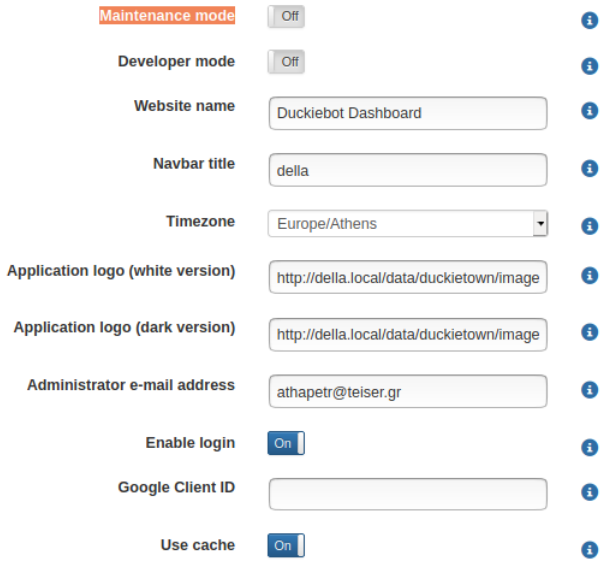


Εικόνα 95: Διαθέσιμα πακέτα με δυνατότητα εγκατάστασης και απεγκατάστασης

Τέλος η τελευταία επιλογή έχει να κάνει με την παραμετροποίηση κάποιων βασικών ρυθμίσεων τόσο του \compose\ στην κατηγορία General όσο και των πακέτων και εφαρμογών που εκτελούνται στο Duckiebot.

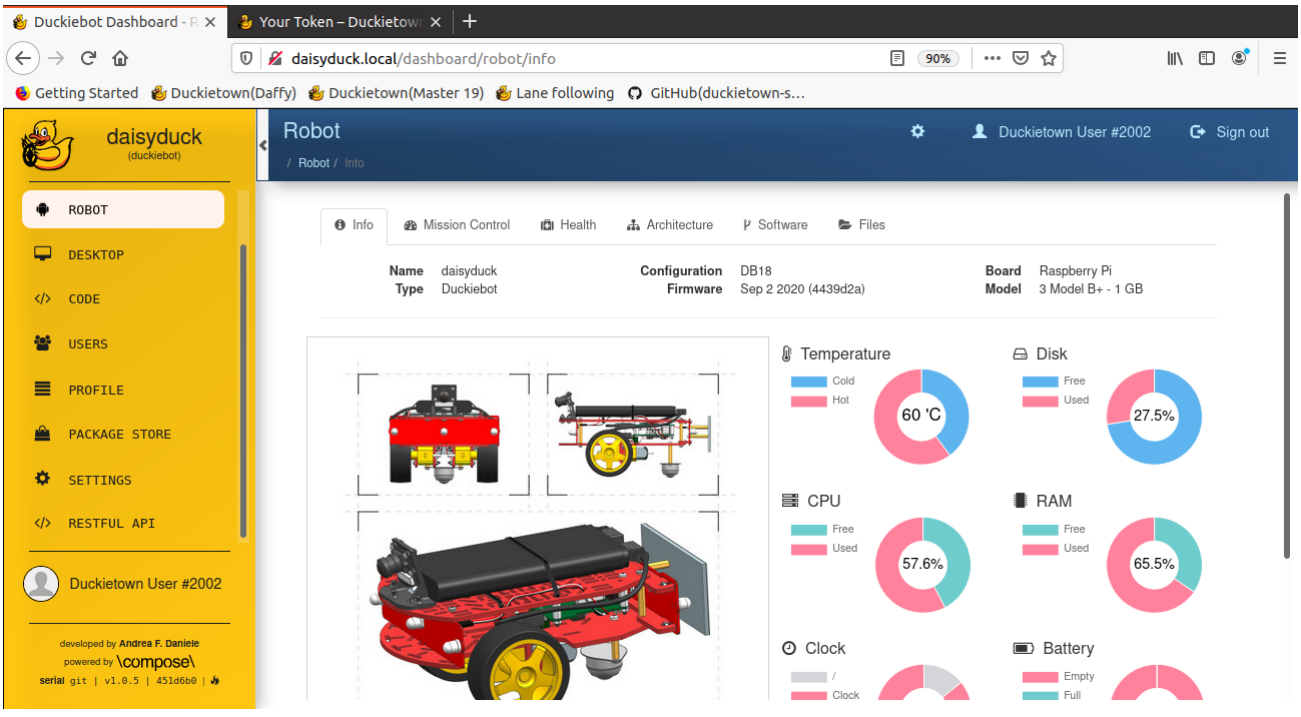
Εικόνα 96: Δυνατότητα παραμετροποίησης πακέτων και που εκτελούνται στο Duckiebot

Ενδεικτικά αναφέρουμε την κατηγορία General όπου μπορούμε να ορίσουμε το όνομα του Website, το όνομα τίτλου που θα εμφανίζεται στη γραμμή περιήγησης, τη ζώνη ώρας. το logo της εφαρμογής, το email του διαχειριστή, κτλ.



Εικόνα 97 : Ρυθμίσεις της κατηγορίας General του \compose

Στο σημείο αυτό να τονίσουμε ότι το περιβάλλον του /compose/ διαφοροποιείται αν ορίσουμε ως έκδοση εντολών του Duckietown shell την daffy και αρχικοποιήσουμε την κάρτα Micro SD που φέρει το Raspberry Pi του Duckiebot με αυτή. Η έκδοση daffy είναι μία πειραματική έκδοση που όμως ενημερώνεται πιο συχνά και τα όποια bags προκύπτουν αντιμετωπίζονται γρηγορότερα σε σχέση με την επίσημη master19 που προτείνεται. Όπως βλέπουμε στην παρακάτω εικόνα η αρχική σελίδα του /compose/ στην έκδοση daffy μετά τη σύνδεση μας με διαπιστευτήρια το κουπόνι (token) που μας παρέχει η πλατφόρμα duckietown.org έχει την εξής μορφή:



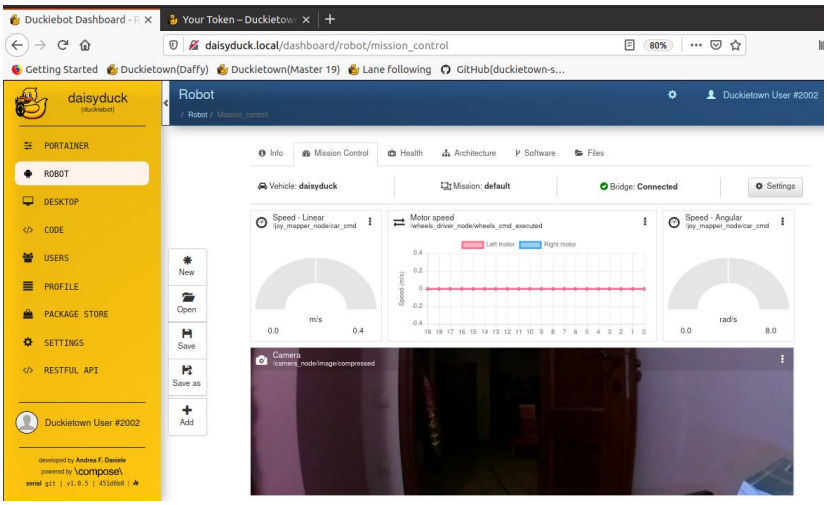
Εικόνα 98 : Καρτέλα πληροφοριών (info) του \compose\ για Duckiebot (daffy έκδοση)

Η καρτέλα info μας παρέχει μία σύνοψη της τρέχουσας κατάστασης των πόρων του Duckiebot εμφανίζοντας πληροφορίες για τη θερμοκρασία λειτουργίας, το ποσοστό χρήσης του και τον χρονισμό του επεξεργαστή, το ποσοστό διαθέσιμου και δεσμευμένου χώρου στη δευτερεύουσα και κύρια μνήμη του καθώς στην κατάσταση της μπαταρίας που διαθέτει.

Το μενού που περιγράψαμε παραπάνω εμφανίζεται με κάποιες διαφοροποιήσεις

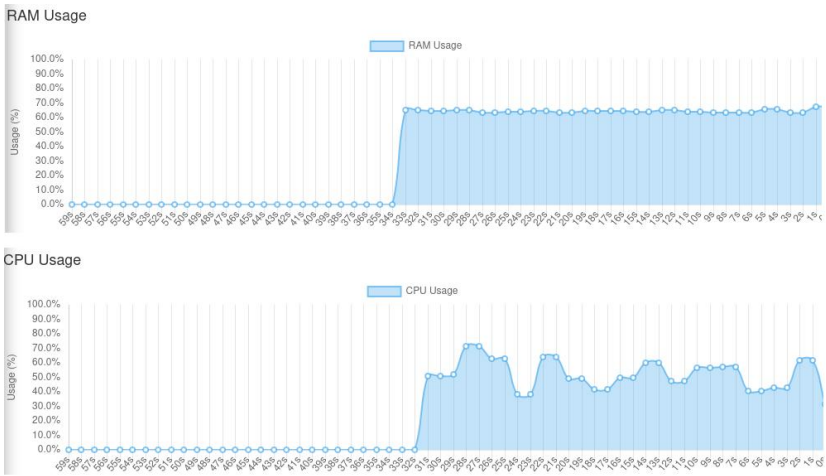
Εικόνα 99 : Οριζόντιο Μενού επιλογών \compose\ (daffy έκδοση)

Το Mission Control έχει την ίδια μορφή και λειτουργικότητα με αυτή που περιγράψαμε παραπάνω όπως φαίνεται και στην παρακάτω εικόνα



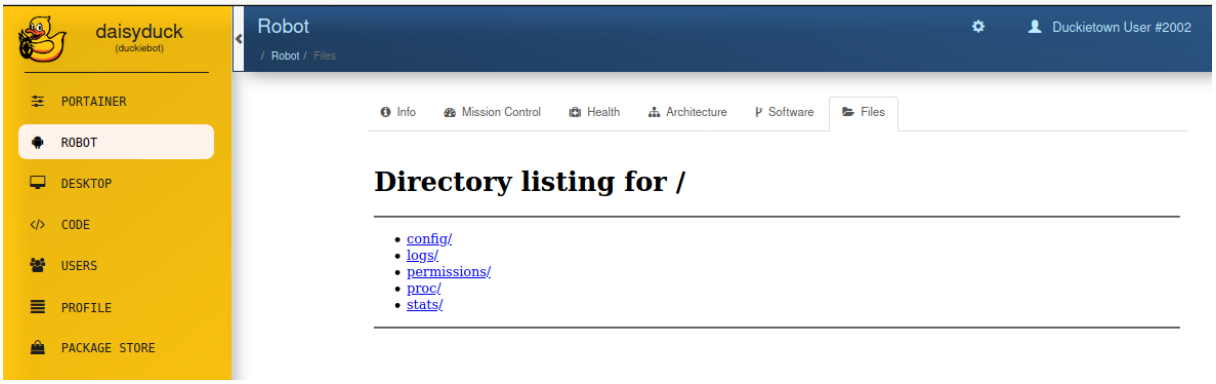
Εικόνα 100 : Καρτέλα Mission Control \compose\ (daffy έκδοση)

Η καρτέλα Health μας ενημερώνει σε πραγματικό χρόνο με γραφικές παραστάσεις για την κατάσταση (ποσοστό χρήσης, θερμοκρασία, τάση) των πόρων του Duckiebot



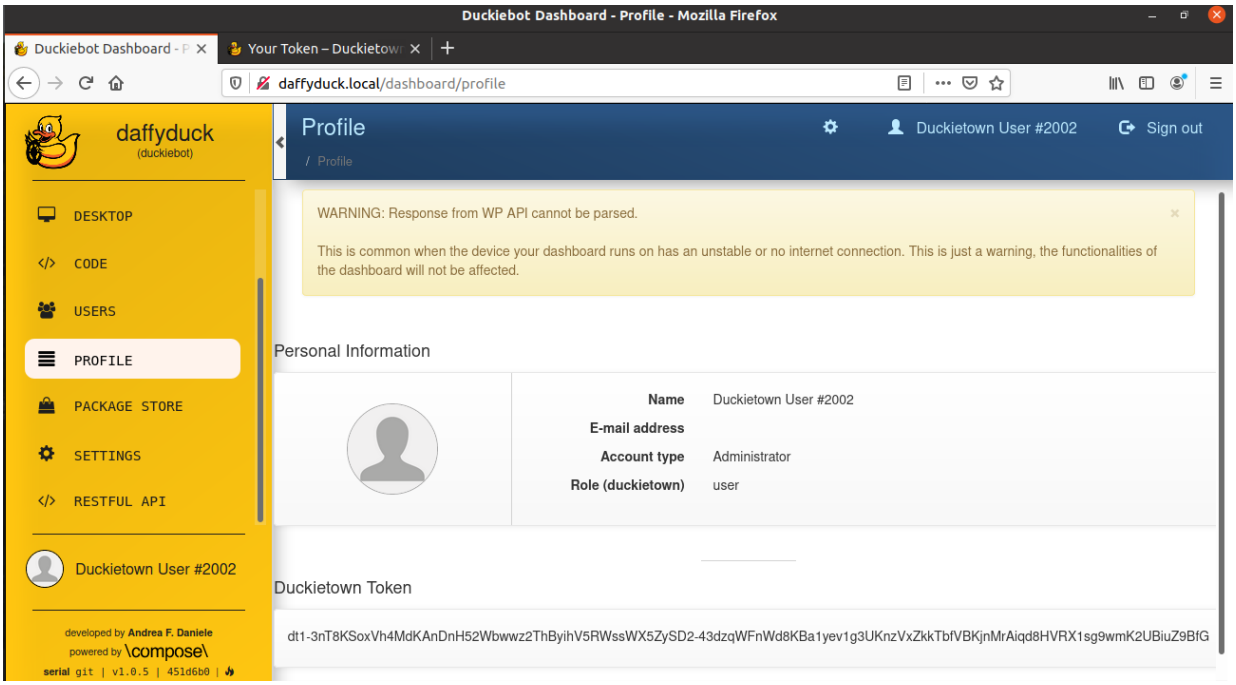
Εικόνα 101 : Καρτέλα Health του \compose\ (daffy έκδοση)

Η καρτέλα Architecture σχεδιάστηκε για απεικονίσει γραφικά τους διασυνδεόμενους κόμβους του ROS που φανερώνουν την αρχιτεκτονική δομή χωρίς όμως ακόμα να είναι διαθέσιμη αυτή η λειτουργικότητα ενώ η καρτέλα files μας παρέχει πρόσβαση στο σύστημα αρχείων του Duckiebot



Εικόνα 102: Πρόσβαση στα αρχεία του Duckiebot μέσω του \compose\ (daffy έκδοση)

Τέλος από το μενού της αριστερής στήλης είναι λειτουργικές οι επιλογές: Package Store με τα εγκατεστημένα αλλά και διαθέσιμα πακέτα που υπάρχουν, η επιλογή Settings με τις ρυθμίσεις που είδαμε και παραπάνω και η επιλογή Profile με τις πληροφορίες του προσωπικού μας λογαριασμού.



Εικόνα 103: Εμφάνιση πληροφοριών του προσωπικού μας προφίλ (daffy έκδοση)

Κεφάλαιο 8 : Λειτουργία κάμερας - Τηλεχειρισμός

8.1 Έλεγχος Λειτουργίας Κάμερας

Το πακέτο camera_driver του ROS διαχειρίζεται όλα όσα σχετίζονται με την κάμερα σε ένα Duckiebot. Έχει ένα μόνο κόμβο, τον κόμβο CameraNode που ανακτά εικόνες από την κάμερα χρησιμοποιώντας το Picamera (ένα πακέτο που μας παρέχει μια διεπαφή γραμμένη σε Python για το χειρισμό της κάμερας στο Raspberry Pi) και στη συνέχεια τις δημοσιεύει. Ο ίδιος κόμβος είναι επίσης υπεύθυνος για τη δημοσίευση πληροφοριών που σχετίζονται με την κάμερα. [22]

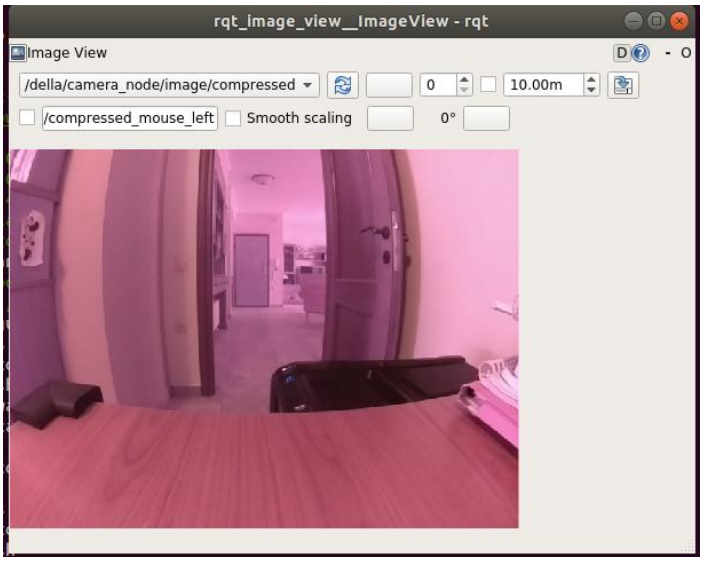
Συγκεκριμένα ο κόμβος CameraNode χειρίζεται τη ροή των εικόνων της κάμερας του Duckiebot αφού αρχικά την έχει αρχικοποιήσει και έπειτα δημοσιεύει πλαίσια εικόνων με καθορισμένη συχνότητα και την σταματά όταν κλείνει. Το Picamera χρησιμοποιείται για το χειρισμό της ροής των εικόνων. Εδώ να σημειώσουμε ότι το αντικείμενο PiCamera πρέπει να χρησιμοποιείται από έναν κόμβο κάθε φορά. Οι παράμετροι διαμόρφωσης μπορούν αν μεταβληθούν δυναμικά όσο ο κόμβος εκτελείται μέσα από το ρεπερτόριο εντολών rosparam.

Πρακτικά για να διαπιστώσουμε αν λειτουργεί κανονικά η κάμερα του Duckiebot που από προεπιλογή στέλνει τη ροή εικόνων που λαμβάνει, μπορούμε να χρησιμοποιήσουμε το Duckietown shell ενεργοποιώντας το container των γραφικών εργαλείων που διαθέτει πληκτρολογώντας σε ένα τερματικό

$ dts start_gui_tools DUCKIEBOT_NAME

και μετά πληκτρολογούμε

$ rqt_image_view

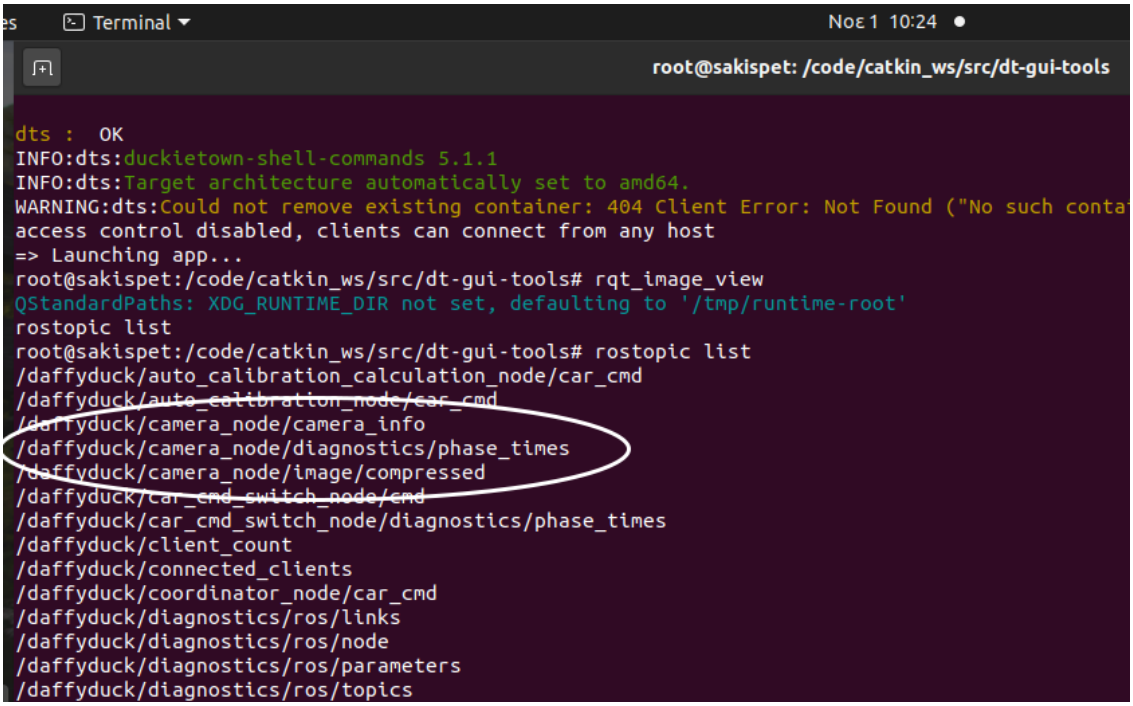


Εικόνα 104 : Ληφθείσα εικόνα από το Duckiebot μέσω του container rqt_image_view

Έπειτα επιλέγουμε το κατάλληλο topic για δούμε τη ροή εικόνων που στέλνει το Duckiebot.

Και μια που κάναμε αναφορά στα topics του ROS μπορούμε να δούμε μία λίστα των καναλιών (topics) που δημοσιεύουν μηνύματα (messages) πληκτρολογώντας :

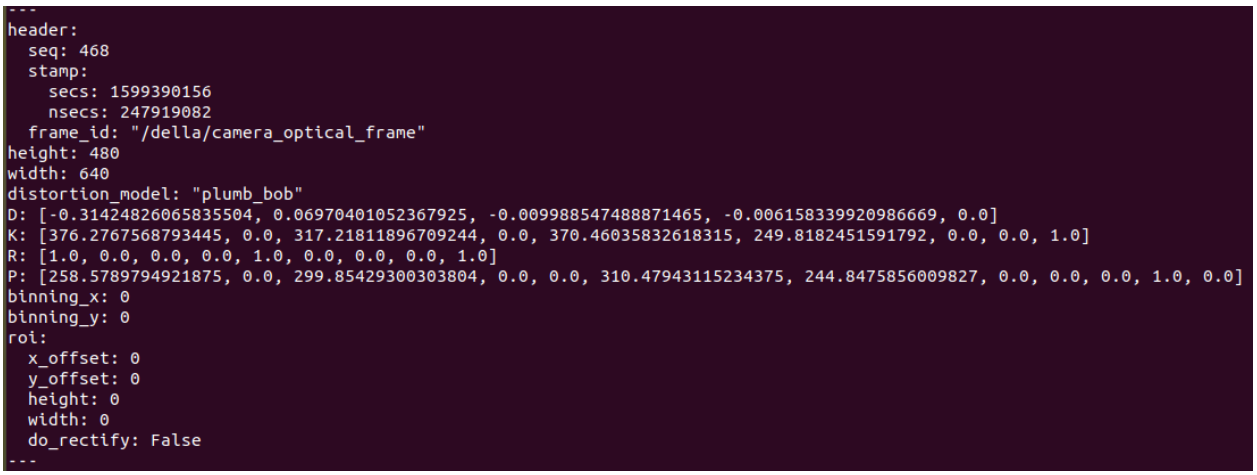
$ rostopic list



Εικόνα 105: Στιγμιότυπο από τα topics του ROS

Εδώ βλέπουμε το topic /daffyduck/camera_node/camera_info που δημοσιεύει πληροφορίες που σχετίζονται με την κάμερα όπως αυτές που φαίνονται στη παρακάτω εικόνα

$ rostopic echo /daffyduck/camera_node/camera_info



Εικόνα 106 : Στιγμιότυπο από πληροφορίες που δημοσιεύει το κανάλι camera_info

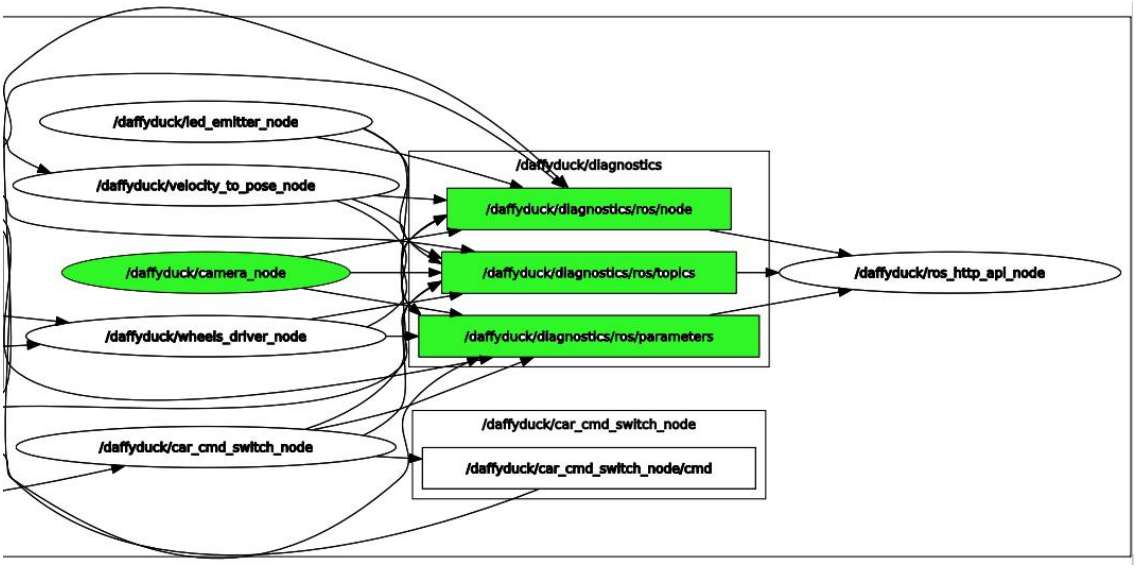
Το topic /daffyduck/camera_node/image/compressed που δημοσιεύει τη ροή των εικόνων όπως την αντιλαμβάνεται η μηχανή δηλαδή μία ακολουθία αριθμών.

$ rostopic echo /daffyduck/camera_node/image/compressed

Εικόνα 107: Η εικόνα όπως την "αντιλαμβάνεται " η μηχανή

Μπορούμε να δούμε τον κόμβο camera_node και μέσα από μία γραφική αναπαράσταση όλων των ενεργών κόμβων του ROS στο Duckiebot πληκτρολογώντας

$ rqt_graph



Εικόνα 108 : Στιγμιότυπο γραφικής αναπαράστασης των κόμβων (nodes) του ROS

Τέλος βλέπουμε εκτός των άλλων τα topics /rosout και /rosout_agg που έχουν ως έργο την εγγραφή των μηνυμάτων σε ένα log αρχείο και την αναδημοσίευση τους. Ακολουθεί ένα στιγμιότυπο της εξόδου του καναλιού /rosout με την εντολή

$ rostopic echo /rosout

```
---
header:
  seq: 4
  stamp:
    secs: 1599390158
    nsecs: 640218973
  frame_id: ''
level: 4
name: "/della/cam_info_reader_node"
msg: "Inbound TCP/IP connection failed: connection from sender terminated before handshake\
  \ header received. 0 bytes were received. Please check sender for additional details."
file: "tcpros_base.py"
function: "_tcp_server_callback"
line: 351
topics: [/rosout, /della/camera_node/camera_info, /della/camera_node/image/compressed]
```

Εικόνα 109: Στιγμιότυπο του καναλιού rosout

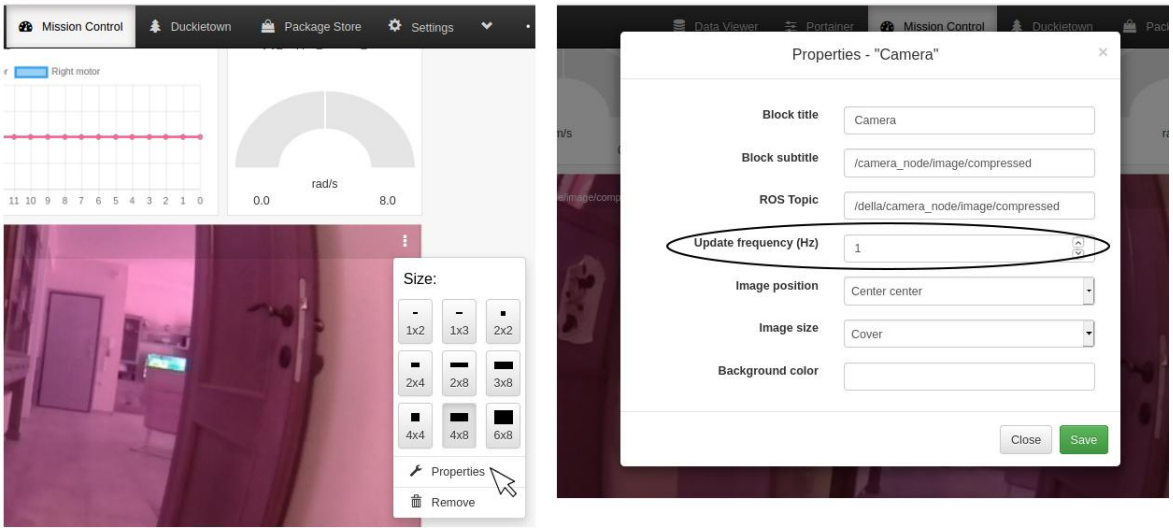
Ένα ενδιαφέρον στοιχείο που μπορούμε να δούμε είναι η συχνότητα αναμετάδοσης της ροής των εικόνων πληκτρολογώντας

$ rostopic hz /daffyduck/camera_node/image/compressed

Εικόνα 110: Εμφάνιση συχνότητας μετάδοσης των εικόνων από το Duckiebot

Έτσι στην περίπτωση μας φαίνεται να αγγίζει τα 20Hz. Φυσικά αυτός ο ρυθμός μετάδοσης μπορεί εύκολα να αυξηθεί ή να μειωθεί.

Αυτή ακριβώς τη συχνότητα του ρυθμού μετάδοσης της ροής των εικόνων μπορούμε να την δούμε και να την μεταβάλλουμε και από το φιλικότερο περιβάλλον του \compose\ Συγκεκριμένα αν παρατηρήσουμε προσεκτικά σε κάθε παράθυρο πάνω δεξιά υπάρχουν τρεις κατακόρυφες τελίτσες. Οι τρεις αυτές τελίτσες σε κάθε παράθυρο μας ανοίγουν ένα αναδυόμενο μενού μέσω του οποίου μπορούμε να αλλάξουμε τις διαστάσεις του παραθύρου και την τιμή σε κάποιες ιδιότητες ανάλογα με το αντικείμενο στο οποίο αναφέρεται το παράθυρο. Έτσι για παράδειγμα αν θέλουμε να μεταβάλλουμε τη συχνότητα με την οποία μας στέλνει το Duckiebot τις εικόνες που λαμβάνει από το εξωτερικό περιβάλλον κάνουμε κλικ στο πάνω αριστερό σημείο του παραθύρου της παραπάνω εικόνας (τελίτσες) και αναδύεται το παρακάτω παράθυρο



Εικόνα 111 : Μεταβολή συχνότητας μετάδοσης εικόνων από το \compose

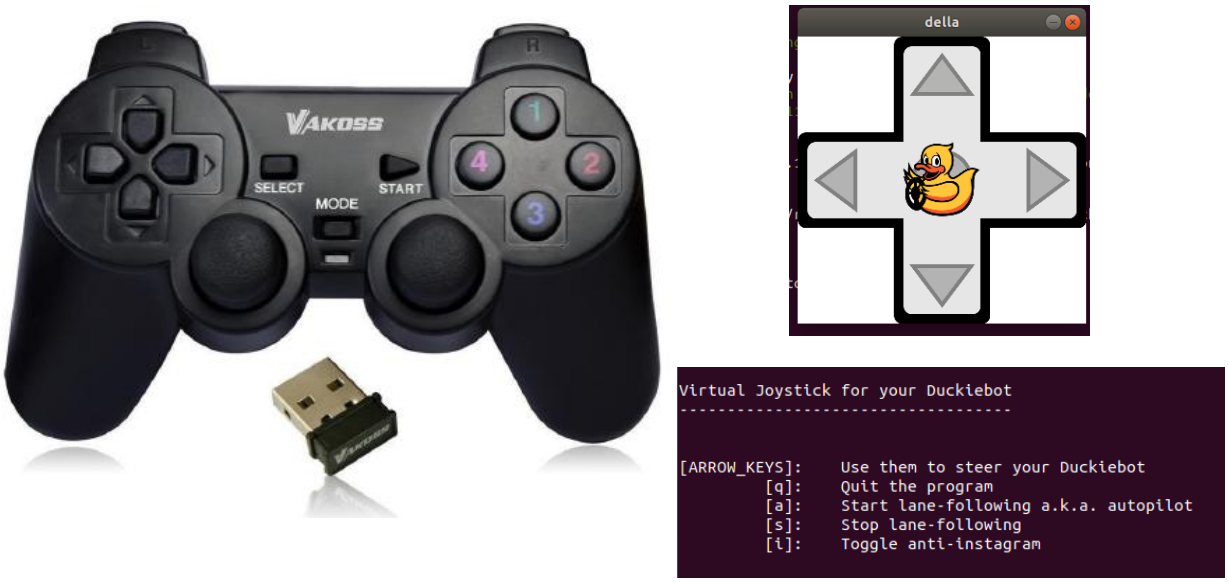
8.2 Τηλεχειρισμός

Η απρόσκοπτη λειτουργία της κάμερας και ο απομακρυσμένος έλεγχος της κίνησης του Duckiebot αποτελούν τις βασικές, θεμελιώδης λειτουργίες πάνω στις οποίες βασίζονται όλες οι υπόλοιπες. Το βασικό πακέτο που επιτρέπει τον τηλεχειρισμό του Duckiebot περιλαμβάνει δύο πακέτα του ROS, το Joystick για να χειρίζεται την είσοδο από ένα Joystick ή έναν προσομοιωτή Joystick και το πακέτο wheel_driver που χειρίζεται την πραγματική εκτέλεση εντολών στους δύο κινητήρες του Duckiebot.

Το πακέτο Joystick περιέχει λογισμικό για λήψη πληροφοριών από το χειριστήριο και την χαρτογράφηση τους σε σχετικά topics που χρησιμοποιούνται από άλλους κόμβους. Το πακέτο περιλαμβάνει δύο βασικούς κόμβους:

- Το κόμβο Joy_node που δημοσιεύει μηνύματα εξόδου του Joystick

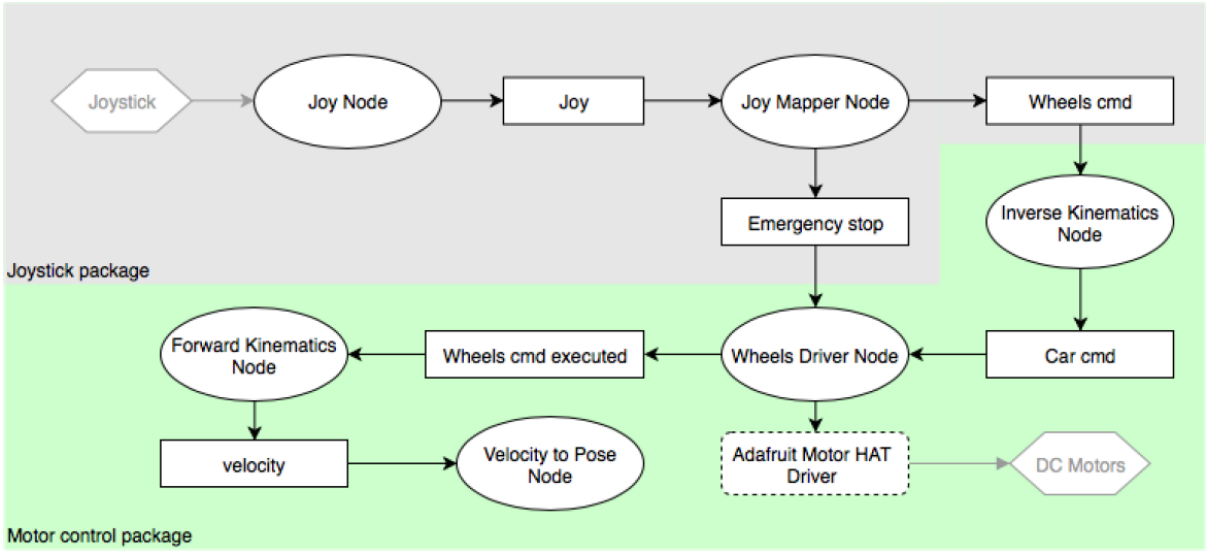
- Το κόμβο joy_mapper_node που χαρτογραφεί (αντιστοιχεί - κατευθύνει) μηνύματα από το Joy Node σε σχετικά topics. Σε ένα συνδεδεμένο στο φορητό υπολογιστή Joystick ο κάθετος άξονας του αριστερού χειριστηρίου αντιστοιχεί στην ταχύτητα ενώ ο οριζόντιος άξονας του δεξιού χειριστηρίου αντιστοιχεί στις εντολές αλλαγής κατεύθυνσης(τιμόνι). Αν δεν υπάρχει συσκευή joystick ο τηλεχειρισμός μπορεί να πραγματοποιηθεί μέσα από ένα γραφικό περιβάλλον που προσομοιώνει το Joystick



Εικόνα 112 : Αριστερά Joystick - Δεξιά προσομοιωτής Joystick

Σε κάθε περίπτωση τα σήματα εισόδου είτε από τα κουμπιά του Joystick είτε από το πληκτρολόγιο αντιστοιχίζονται σε διάφορα θέματα όπως εκκίνηση λειτουργίας ακολούθησης λωρίδας (lane following), τερματισμός της λειτουργίας lane following και μεταβολή του anti-instagram για πιο αξιόπιστες ανιχνεύσεις γραμμής.

Το παρακάτω γράφημα του ROS απεικονίζει συσκευές, κόμβους και μηνύματα κατά την εκκίνηση του προγράμματος ελέγχου της κίνησης του Duckiebot που αποτελεί και την πιο απλή συμπεριφορά του. Οι κόμβοι απεικονίζονται μέσα σε ελλείψεις, τα μηνύματα σε ορθογώνια παραλληλόγραμμα και οι συσκευές μέσα σε εξάγωνα. Το διακεκομμένο πλαίσια απεικονίζει το πακέτο (Adafruit_Motor_Hat_Driver) που οδηγεί τους δύο κινητήρες DC του Duckiebot.



Εικόνα 113 : Γράφημα Συσκευών, Κόμβων και Μηνυμάτων του ROS κατά την εκκίνηση του τηλεχειρισμού.

Το πακέτο Motor control περιλαμβάνει τους παρακάτω κόμβους για έλεγχο χαμηλού επιπέδου των δύο κινητήρων του Duckiebot. Συγκεκριμένα περιλαμβάνει τους κόμβους:

- Kinematics Node
- Wheels Driver Node
- Velocity to Pose Node
- Car Cmd Switch Node

Εκτός από τους παραπάνω κόμβους υπάρχει όπως προαναφέραμε και το πακέτο ελέγχου του Adafruit Motor Hat που τοποθετείται πάνω στο Raspberry Pi και συνδέονται σε αυτό οι δύο DC κινητήρες του Duckiebot τους οποίους και οδηγεί. Το πακέτο αυτό περιλαμβάνει τις εξής βιβλιοθήκες [23]:

- Adafruit_GPIO
- Adafruit_I2C
- Adafruit_MotorHAT
- Adafruit_PWM_Servo_Driver

Στη συνέχεια θα περιγράψουμε συνοπτικά το έργο που έχει επωμιστεί καθένας από τους παραπάνω κόμβους και τον τρόπο με τον οποίο συμβάλει στη διαδικασία τηλεχειρισμού του Duckiebot.

8.2.1 Kinematic Node

Ο κόμβος αυτός χαρτογραφεί τα μηνύματα που δέχεται από διάφορους κόμβους σε εντολές που μπορεί να εκτελεστούν από τους τροχούς του Duckiebot. Ο κόμβος αυτός εκτελεί τόσο τους υπολογισμούς επίλυσης τόσο του ευθέως κινηματικού προβλήματος όσο και του αντίστροφου κινηματικού προβλήματος. Σε προηγούμενες εκδόσεις υπήρχαν ξεχωριστοί κόμβοι (Inverse kinematics Node και Forward Kinematics Node) για τους υπολογισμούς που απαιτούσε κάθε πρόβλημά αλλά λόγω της ομοιότητας και της κοινής χρήσης παραμέτρων συγχωνεύθηκαν σε έναν κόμβο. Ο κόμβος αυτός λαμβάνει υπόψη του τη γεωμετρία του αυτοκινήτου καθώς και έναν αριθμό παραμέτρων για να υπολογίσει τις περιστροφές που πρέπει να εκτελέσουν οι τροχοί προκειμένου το Duckiebot να διαγράψει μία επιθυμητή τροχιά - κίνηση (αντίστροφο κινηματικό πρόβλημα). Στη συνέχεια χρησιμοποιεί αυτές τις "εντολές" που δόθηκαν στους τροχούς για να κάνει μία εκτίμηση της ταχύτητας του (το ευθύ κινηματικό πρόβλημα).

8.2.2 Wheels Driver Node

Ο κόμβος Wheels Driver μεταφράζει τις ταχύτητες περιστροφής κάθε τροχού σε σήματα PWM για τους κινητήρες που αντιστοιχούν σε αυτούς μέσω του προγράμματος οδήγησης Adafruit Motor HAT. Ο κόμβος αυτός δηλαδή είναι σχεδιασμένος για να επικοινωνεί κατευθείαν με το υλικό του Duckiebot. Να επισημάνουμε στο σημείο αυτό ότι αυτός ο κόμβος δεν λαμβάνει ανάδραση από τους κινητήρες των τροχών και συνεπώς δεν γνωρίζει τις πραγματικές τους ταχύτητες. Όταν λαμβάνεται ένα μήνυμα κίνησης του Duckiebot όπως για παράδειγμα περιστροφή και των δύο τροχών δεξιόστροφα ο μήνυμα δημοσιεύεται και εκτελείται. Επί πλέον ο κόμβος "ακούει" και το θέμα (topic) της επείγουσας διακοπής της κίνησης του Duckiebot. Εάν λάβει μήνυμα έκτακτης ανάγκης (stop=true) θα αγνοήσει τυχόν εισερχόμενα μηνύματα ταχύτητας και θα θέσει και τους δύο κινητήρες σε πλήρη παύση. Ο κόμβος θα συνεχίσει την "φυσιολογική" συμπεριφορά του όταν λάβει μήνυμα διακοπής της έκτακτης ανάγκης (stop=false).

8.2.3 Velocity to Pose Node

Ο κόμβος αυτός μπορεί να χρησιμοποιηθεί σε συνδυασμό με τον kinematics κόμβο για την εκτίμηση της τρέχουσας θέσης του Duckiebot. Ο κόμβος κινηματικής δεδομένης της ταχύτητας του Duckiebot θα υπολογίσει τις γραμμικές και γωνιακές ταχύτητες του ρομπότ. Στη συνέχει ο κόμβος Velocity to Pose θα εκτιμήσει την τρέχουσα θέση του ρομπότ από τα ληφθέντα μηνύματα γραμμικής και γωνιακής ταχύτητας.

8.2.4 Car Cmd Switch Node

Το Duckiebot είναι ικανό να υποστηρίξει πολλές διαφορετικές συμπεριφορές. Από απλές όπως είναι ο τηλεχειρισμός του μέχρι πολύπλοκες και πιο προηγμένες όπως η ακολούθηση λωρίδας (lane following) ή ακόμα και η αυτόνομη πλοήγηση μέσα στην πόλη. Ο κόμβος car_cmd_switch_node κάνει δυνατή αυτή την εναλλαγή των τρόπων λειτουργίας του Duckiebot και διασφαλίζει ότι μόνο μία συγκεκριμένη συμπεριφορά μπορεί να έχει το ρομπότ κάθε δεδομένη χρονική στιγμή.

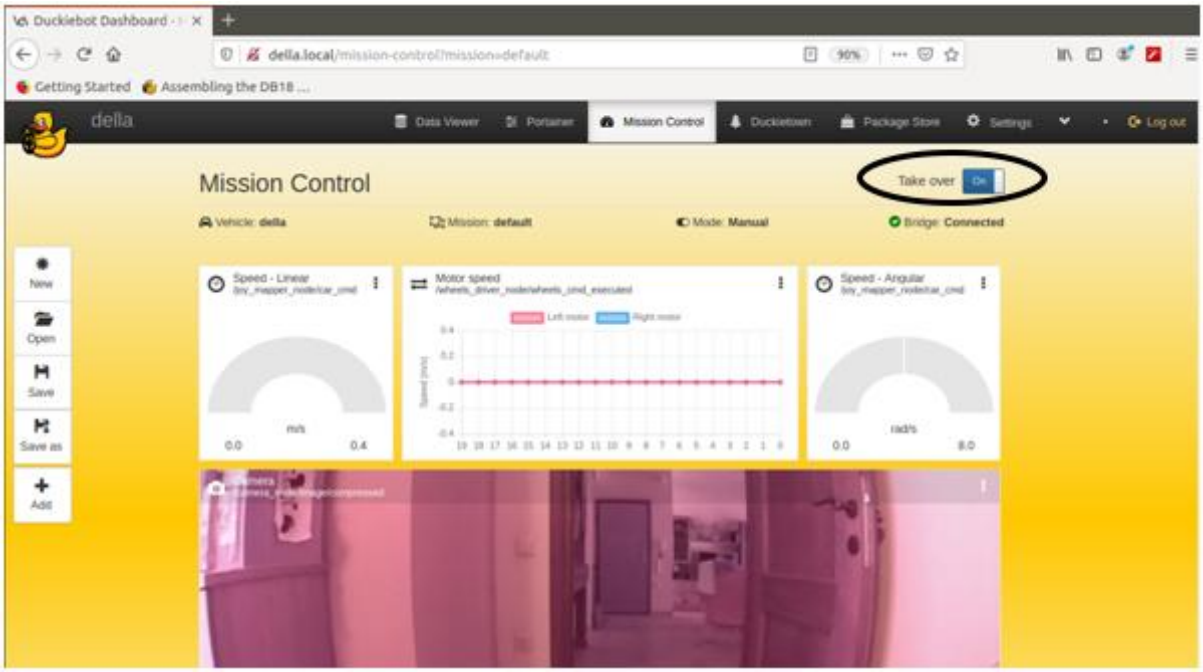
Πρακτικά ο τηλεχειρισμός του Duckiebot είναι εφικτός με δύο τρόπους. Ο ένας είναι μέσω του /compose/ και άλλος μέσω του Duckietown shell.

8.3 Τηλεχειρισμός με το /compose/

Εκκινούμε έναν φυλλομετρητή στον φορητό μας υπολογιστή και πληκτρολογούμε στη γραμμή διεύθυνσης το αναγνωριστικό όνομα του Duckiebot στο τοπικό δίκτυο. Αφού γίνει εφικτή η επικοινωνία μας με το Duckiebot μεταβαίνουμε στην καρτέλα mission control και επιλέγουμε την default αποστολή.

Εικόνα 114 : Επιλογή default "αποστολής"

Έπειτα αλλάζουμε τη θέση του διακόπτη Take Over σε On. Το φόντο της σελίδας γίνεται πράσινο και αν όλα πάνε κατ' ευχήν είμαστε έτοιμοι να ανακτήσουμε τον έλεγχο της κίνησης του Duckiebot. (Προσέχουμε η κατάσταση της ένδειξης Bridge να είναι Connected)

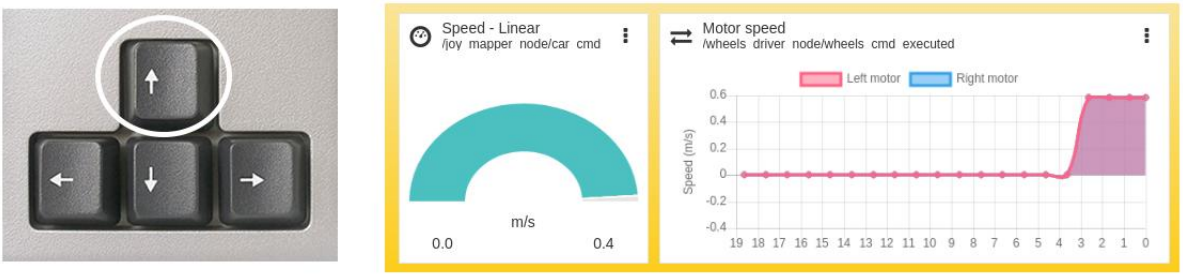


Εικόνα 115 : Αλλαγή κατάστασης λειτουργία Duckiebot (take over)

Είμαστε σε θέση δηλαδή χρησιμοποιώντας τα βελάκια από το πληκτρολόγιο να το κινήσουμε εμπρός ή πίσω ή να το κατευθύνουμε δεξιά ή αριστερά ή και γενικότερα να το τηλε-κατευθύνουμε όπου εμείς επιθυμούμε! Οι παρακάτω εικόνες είναι στιγμιότυπα που δείχνουν γραφικά την επίπτωση που έχει το πάτημα του κάθε βέλους του πληκτρολογίου στην γραμμική και γωνιακή ταχύτητα του Duckiebot.

Κίνηση εμπρός

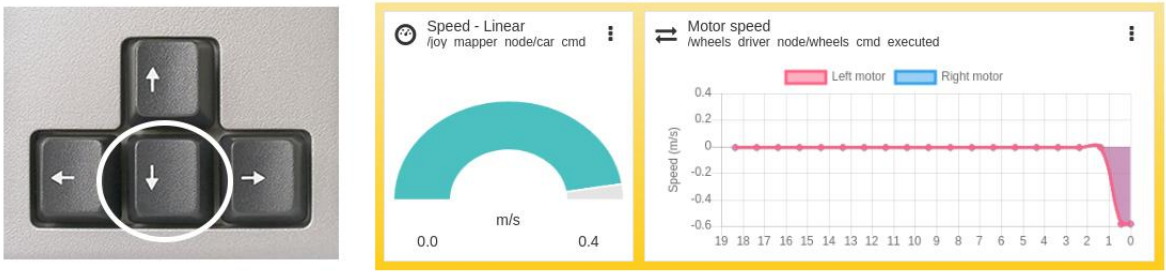
Πατώντας το πάνω βέλος οι δύο κινητήρες του Duckiebot παίρνουν θετικές τιμές όπως φαίνεται και στην γραφική παράσταση της ταχύτητας των motor, συνεπώς κινούνται δεξιόστροφα και το ρομποτικό όχημα κινείται προς τα εμπρός.



Εικόνα 116: Κίνηση εμπρός Duckiebot

Κίνηση όπισθεν

Πατώντας το κάτω βέλος οι δύο κινητήρες του Duckiebot παίρνουν αρνητικές τιμές όπως φαίνεται και στην γραφική παράσταση της ταχύτητας των motor, συνεπώς κινούνται αριστερόστροφα και το ρομποτικό όχημα κινείται προς τα πίσω.



Εικόνα 117 : Κίνηση όπισθεν Duckiebot

Κίνηση δεξιά

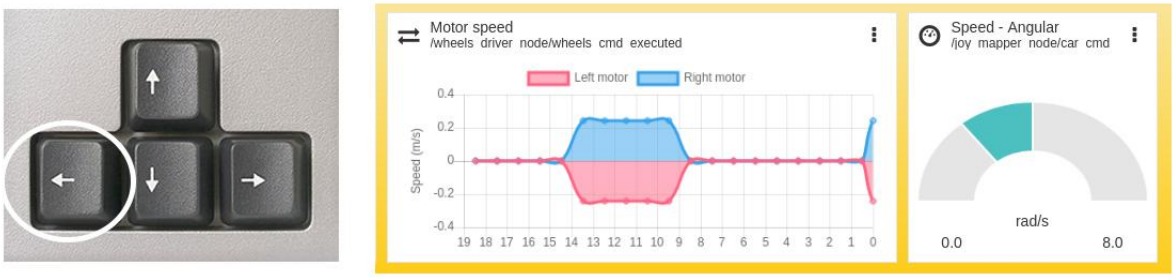
Πατώντας το δεξί βέλος ο αριστερός κινητήρας του Duckiebot παίρνει θετικές τιμές ενώ ο δεξιός αρνητικές όπως φαίνεται και στην γραφική παράσταση της ταχύτητας των motor με αποτέλεσμα ο αριστερός τροχός να κινείται δεξιόστροφα ενώ ο δεξιός τροχός αριστερόστροφα και το ρομποτικό όχημα να στρίβει δεξιά.

Εικόνα 118: Κίνηση δεξιά Duckiebot

Κίνηση αριστερά

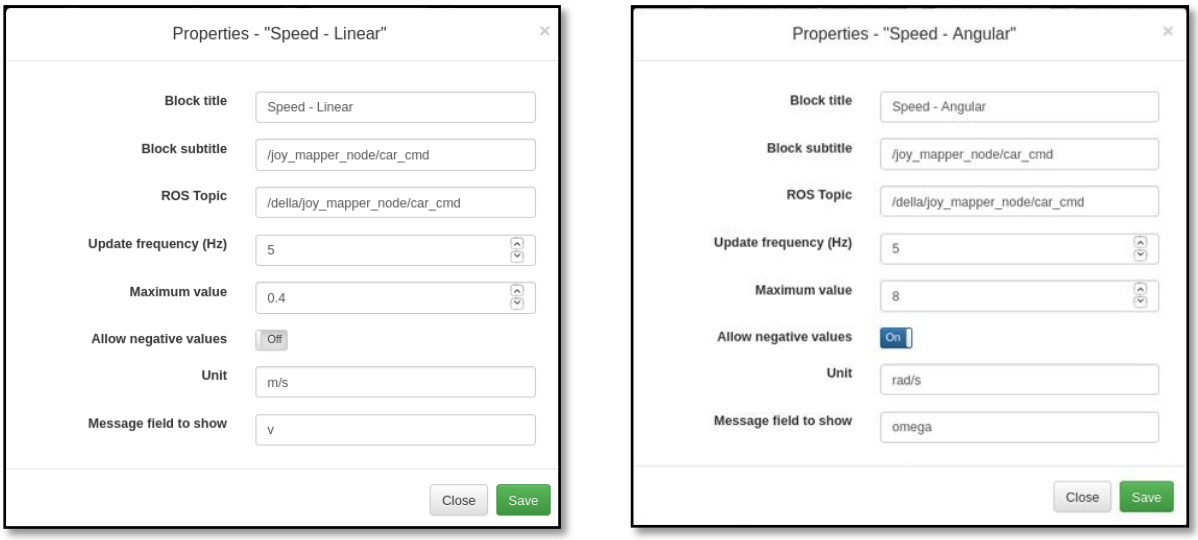
Πατώντας το αριστερό βέλος ο δεξιός κινητήρας του Duckiebot παίρνει θετικές τιμές ενώ ο αριστερός αρνητικές όπως φαίνεται και στην γραφική παράσταση της ταχύτητας των motor

με αποτέλεσμα ο δεξιός τροχός να κινείται δεξιόστροφα ενώ ο αριστερός τροχός αριστερόστροφα και το ρομποτικό όχημα να στρίβει αριστερά.

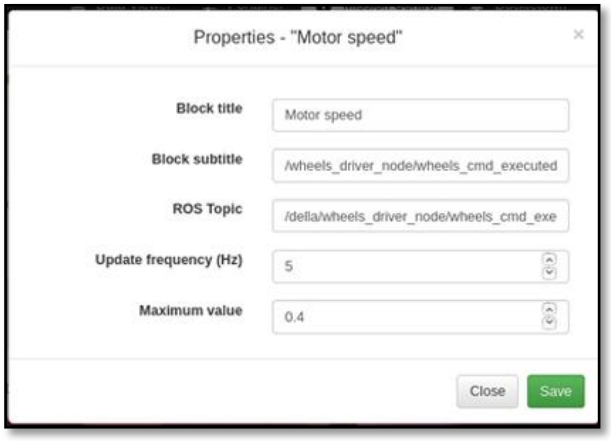


Εικόνα 119: Κίνηση αριστερά Duckiebot

Από τις τρεις κατακόρυφες τελίτσες των παραθύρων Speed Linear, Motor speed και Speed Angural "ανοίγουμε" τις ιδιότητες των αντίστοιχων αντικειμένων και μπορούμε να μεταβάλλουμε δυναμικά τις τιμές των ιδιοτήτων τους όπως τη συχνότητα ενημέρωσης, τη μέγιστη τιμή, την μονάδα μέτρησης και αν θα επιτρέπονται αρνητικές τιμές.



Εικόνα 120 : Ιδιότητες γραμμικής και γωνιακής ταχύτητας



Εικόνα 121 : Ιδιότητες ταχύτητας κινητήρων

8.4 Τηλεχειρισμός με το Duckietown Shell

Ο δεύτερος τρόπος για να ελέγξουμε χειροκίνητα την κίνηση του Duckiebot από απόσταση είτε με τα πλήκτρα - βέλη του πληκτρολογίου είτε με συσκευή τύπου joystick είναι να χρησιμοποιήσουμε τον Duckietown shell. Έτσι αν θέλουμε να κινήσουμε το Duckiebot μέσω joystick αρκεί να πληκτρολογήσουμε την εντολή τερματικού:

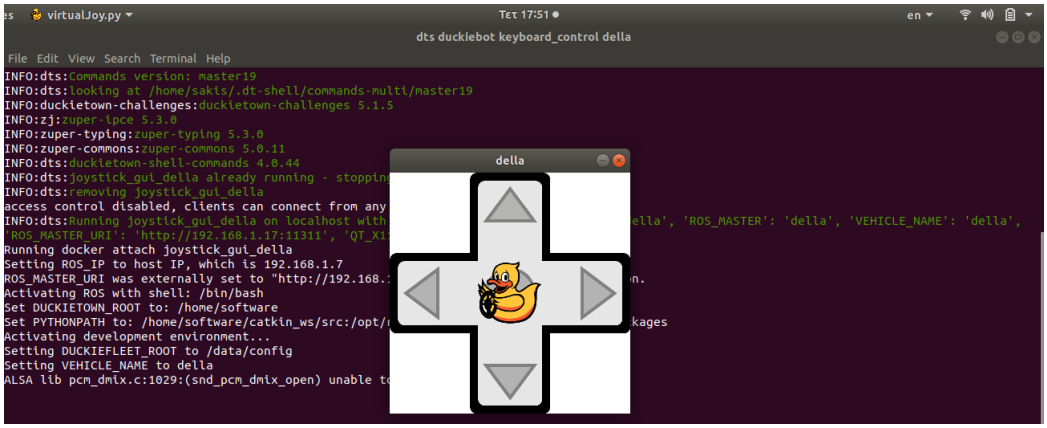
$ dts duckiebot demo --demo_name joystick --duckiebot_name DUCKIEBOT_NAME

ενώ αν θέλουμε να κινήσουμε το Duckiebot όπως με τα πλήκτρα-βέλη του φορητού μας υπολογιστή θα πρέπει να πρέπει να πληκτρολογήσουμε και την εντολή

$ dts duckiebot keyboard_control DUCKIEBOT_NAME

Εικόνα 122: Εντολή εκκίνησης προσομοιωτή Joystick μέσω του Duckietown shell

και θα δούμε να εκκινεί ένα παράθυρο όπως αυτό που φαίνεται στην παρακάτω εικόνα.



Εικόνα 123: Στιγμιότυπο από τον προσομοιωτή Joystick

Αν προσέξουμε κάτω από το αναδυόμενο παράθυρο προσομοίωσης εμφανίζεται και το ακόλουθο μενού πέντε (5) επιλογών:

```
Virtual Joystick for your Duckiebot
-----------------------------------


[ARROW_KEYS]:    Use them to steer your Duckiebot
        [q]:     Quit the program
        [a]:     Start lane-following a.k.a. autopilot
        [s]:     Stop lane-following
        [i]:     Toggle anti-instagram
```

Εικόνα 124: Μενού επιλογών καταστάσεων λειτουργίας Duckiebot

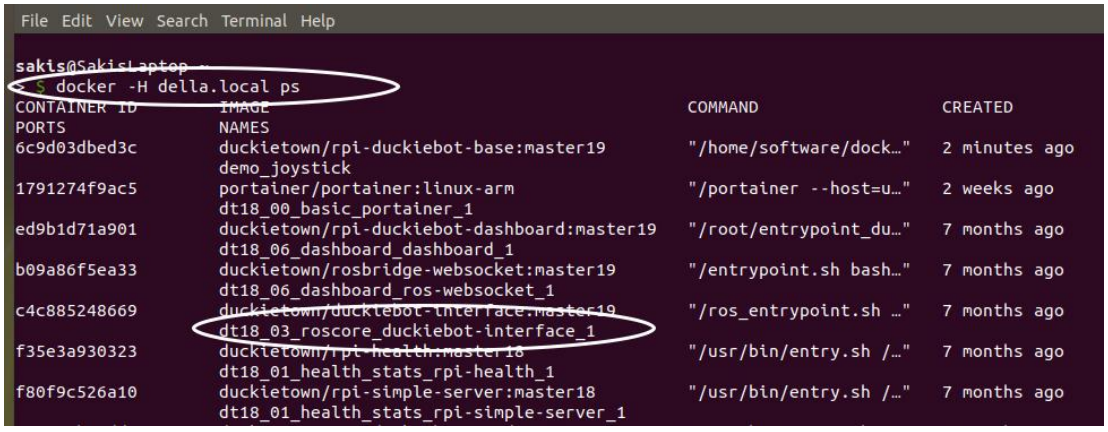
Με τα πλήκτρα-βέλη κατευθύνουμε το Duckiebot χειροκίνητα από το Laptop. Πατώντας το γράμμα [q] τερματίζουμε το πρόγραμμα προσομοίωσης του Duckietown shell. Με το γράμμα [a] του πληκτρολογίου μεταβαίνουμε σε κατάσταση lane-following δηλαδή το Duckiebot να ακολουθεί μόνο του μία γραμμή βασιζόμενο στις προσλαμβάνουσες από την κάμερα του εικόνες, με το πλήκτρο [s] να εξέλθουμε από την προηγούμενη κατάσταση και τέλος με πατώντας το πλήκτρο [i] να εισέλθουμε σε μία διαδικασία βελτίωσης της απόδοσης του ανιχνευτή γραμμής μέσω αλλαγής του φίλτρου χρώματος με βάση την δεδομένη ακτινοβολία του υπάρχοντος φωτισμού.

Σε περίπτωση που το Duckiebot δεν κινείται θα πρέπει να ελέγξουμε αν η υπηρεσία Duckiebot-interface εκτελείται. Αυτό μπορούμε να το διαπιστώσουμε είτε μέσα από το περιβάλλον του Portainer στο οποίο μπορούμε να μεταβούμε και μέσω του \compose\ και να ελέγξουμε να εκτελείτε το container με το όνομα: dt18_03_roscore_Duckiebot-interface_1

Εικόνα 125: Έλεγχος κατάστασης Duckiebot-interface container μέσω του Portainer

Εναλλακτικά μπορούμε να δούμε ποιες είναι οι ενεργές διεργασίες που εκτελούνται στο Duckiebot πληκτρολογώντας την εντολή τερματικού

$ docker -H Duckiebot_Name.local ps



Εικόνα 126: Έλεγχος κατάστασης Duckiebot-interface container μέσω γραμμής εντολών

Στην περίπτωση που διαπιστώσουμε ότι δεν εκτελείται το παραπάνω container θα πρέπει να "κατεβάσουμε" την βασική εικόνα (image) τόσο στο Duckiebot όσο και το Laptop:
$ docker -H DUCKIEBOT_NAME.local pull duckietown/rpi-dukiebot-base:master19
$ docker pull duckietown/rpi-Duckiebot:master19-no-arm

Κεφάλαιο 9

Βαθμονόμηση Κάμερας - Τροχών

9.1 Υπολογιστική όραση (Computer Vision)

Τα δίτροχα οχήματα που ονομάζονται Duckiebots είναι εξοπλισμένα με μία κάμερα. Αυτός είναι ο μοναδικός αισθητήρας που διαθέτουν για να αντιληφθούν τον εξωτερικό περιβάλλον τους και να αλληλεπιδράσουν με αυτό. Είναι λοιπόν σαφές ότι όλες τις πληροφορίες για τον έξω κόσμο τις αντλούν μόνο από τις προσλαμβάνουσες εικόνες που λαμβάνει η κάμερα που διαθέτουν. Ευλόγως κάποιοι θα αναρωτηθούν και τι είδους πληροφορίες μπορεί να εξάγει από μία συλλογή εικόνων ένα μικρό ρομποτικό όχημα; Μπορεί να αναγνωρίσει αντικείμενα, γραμμές ή σύμβολα; Και αν ναι με ποιο τρόπο το επιτυγχάνει; Αυτά τα ερωτήματα άπτονται μιας ειδικής κατηγορίας της τεχνητής νοημοσύνης που ονομάζεται υπολογιστή όραση και θα προσπαθήσουμε να τα απαντήσουμε σε αυτό το κεφάλαιο.

Ο άνθρωπος είναι χωρίς αμφιβολία η πιο "τέλεια μηχανή" που έχει υπάρξει με λειτουργίες πολυσύνθετες και πολύπλοκές που εκπλήσσουν και εντυπωσιάζουν. Ο τρόπος με τον οποίο αντιλαμβάνεται τα διαφορετικού είδους εξωτερικά ερεθίσματα και η ταχύτητα με την οποία επεξεργάζεται τα δεδομένα που λαμβάνει και αντιδρά σ' αυτά είναι ασύλληπτη, ακόμα προς διερεύνηση και σίγουρα προς μίμηση για τον κλάδο της ρομποτικής. Ανάμεσα σ' αυτά η αίσθηση της όρασης είναι ίσως το πιο θαυμαστό από όλα. Αναγνωρίζει αστραπιαία την τρισδιάστατη δομή του κόσμου γύρω του εξάγοντας με μηδενική σχεδόν υστέρηση πληροφορίες για τα αντικείμενα, το βάθος τους, τις υφές τους, το σχήμα τους, τις σκιάσεις. Αναγνωρίζει με ευκολία γνωστά πρόσωπα ανάμεσα σε άγνωστα, αντιλαμβάνεται που ακριβώς είναι αν βρίσκεται σε γνωστό περιβάλλον ακόμα κι αν αυτό έχει κάποιες αλλαγές. Διακρίνει την κίνηση υπολογίζοντας και την ταχύτητα της και κάνει εκτιμήσεις για το χρόνο που απαιτείται για να φτάσει το κινούμενο αντικείμενο ή έμψυχο όν από το ένα σημείο στο άλλο. Όλα αυτά που φαίνονται αυτονόητα σε εμάς απαιτούν στην πραγματικότητα χρονοβόρες και πολυσύνθετες διαδικασίες και αυτό γίνεται αντιληπτό όταν μέρος των παραπάνω ικανοτήτων θέλουμε να τις εμφυτέψουμε στις μηχανές. Αυτός είναι και ο σκοπός της υπολογιστικής όρασης. Η δυνατότητα δηλαδή μία υπολογιστή μηχανή ή ένα ρομπότ να μπορεί να κατανοεί τον φυσικό κόσμο στον οποίο βρίσκεται χρησιμοποιώντας τις εικόνες που λαμβάνει από αυτόν δηλαδή να εξάγει συμπεράσματα επεξεργαζόμενο ουσιαστικά τις μετρήσεις του ανακλώμενου φωτός στα αντικείμενα που το περιβάλλουν.

Για είναι σε θέση το Duckiebot να επιδείξει "έξυπνες" συμπεριφορές όπως να ακολουθεί τη λωρίδα του δρόμου ή να πλοηγείται αυτόνομα μέσα στην πόλη πρέπει να βασιστεί στα δεδομένα που λαμβάνει από τον ένα και μοναδικό αισθητήρα που διαθέτει δηλαδή την κάμερα. Για να είναι σε θέση να εκτιμήσει σωστά τις προσλαμβάνουσες από τον προγραμματικό κόσμο εικόνες και να εξάγει αξιόπιστα συμπεράσματα από αυτόν θα πρέπει να δημιουργηθεί μία σχέση, μία αντιστοίχηση μεταξύ των σημείων του τρισδιάστατου κόσμου και των εικονοστοιχείων (pixel) της δισδιάστατης εικόνας που λαμβάνει η κάμερα. Η

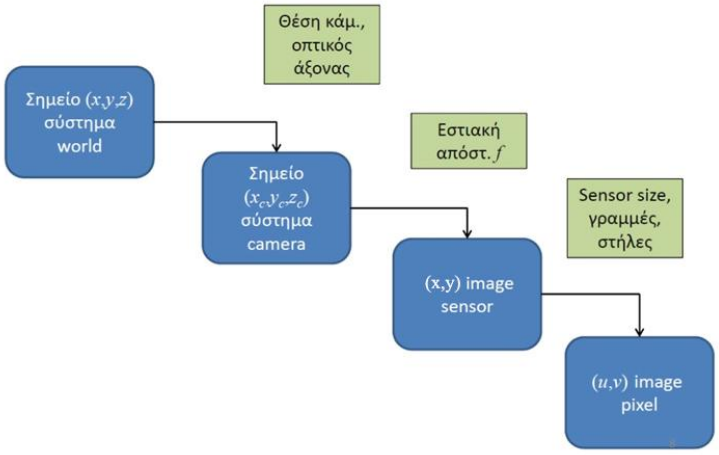
εύρεση μιας τέτοιας σχέσης δεν είναι εύκολη υπόθεση όπως και η εύρεση της αντίθετης σχέσης που θα μας επιτρέψει να αντιστοιχίσουμε τα εικονοστοιχεία της εικόνας με σημεία του πραγματικού κόσμου. Αυτές ακριβώς τις σχέσεις προσπαθεί να προσεγγίσει η διαδικασία της βαθμονόμησης της κάμερας.

Σε μια πραγματική κάμερα οι θέσεις των σημείων σε μια εικόνα αντιστοιχούνται σε pixels, η αρχή των αξόνων είναι στην πάνω αριστερή γωνία της εικόνας, η θέση και ο προσανατολισμός της κάμερας γενικά είναι τυχαίος και υπάρχουν παραμορφώσεις οι οποίες οφείλονται κυρίως στην κάμερα.

Για να μπορούμε να καθορίσουμε τη σχέση μεταξύ των συντεταγμένων ενός σημείου στον τρισδιάστατο χώρο και των συντεταγμένων της προβολής του στην εικόνα, έτσι ώστε να μπορούν να γίνουν μετρήσεις και να βρεθεί η θέση τους στον πραγματικό χώρο, πρέπει να γνωρίζουμε τις παραμέτρους τις κάμερας. Οι παράμετροι χωρίζονται σε ενδογενείς και εξωγενείς. Αυτές που εξαρτώνται μόνο από την ίδια την κάμερα, και δεν αλλάζουν μεταξύ διαφορετικών λήψεων, ονομάζονται ενδογενείς, ενώ εκείνες που εξαρτώνται από τη θέση και τον προσανατολισμό της κάμερας σε κάθε λήψη και που προφανώς μπορούν να αλλάζουν μεταξύ των λήψεων, ονομάζονται εξωγενείς [24]. Οι παράμετροι της κάμερας, κυρίως, εκπροσωπούνται από τον πίνακα της κάμερας και η διαδικασία που ακολουθείται για την εύρεση όλων των παραμέτρων ονομάζεται βαθμονόμηση της κάμερας (camera calibration).

Ένα βασικό κομμάτι της υπολογιστικής όρασης περιλαμβάνει τον μετασχηματισμό των συντεταγμένων ενός σημείου του τρισδιάστατου κόσμου στις συντεταγμένες ενός pixel της δισδιάστατης εικόνας. Ξεκινώντας λοιπόν από ένα σημείο $P_w=(X_w,Y_w,Z_w)^T$ του πραγματικού κόσμου πρέπει να το μετατρέψουμε σε ένα σημείο $P_c=(X_c,Y_c,Z_c)^T$ του συστήματος της κάμερας. Έπειτα το σημείο αυτό να το αντιστοιχίσουμε με ένα σημείο $p(x,y)^T$ του πλάνου της κάμερας (image plane ή image sensor) και τέλος να προσδιορίσουμε το εικονοστοιχείο (pixel) της εικόνας που βρίσκεται το σημείο αυτό.

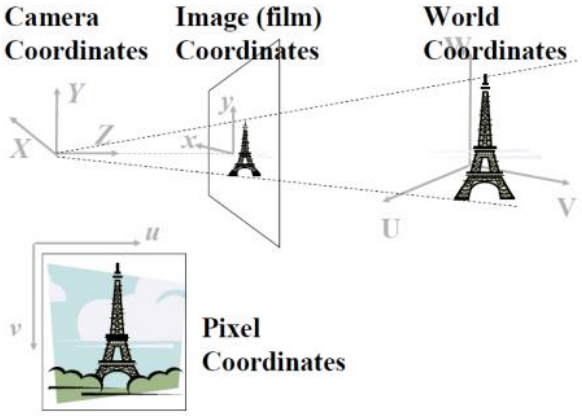
Η παραπάνω διαδικασία περιγράφει τα βήματα της βαθμονόμησης της κάμερας και το παρακάτω σχήματα απεικονίζει γραφικά τα στάδια αυτά.



Εικόνα 127 : Βήματα βαθμονόμησης της κάμερας

Για να περιγραφούν οι μαθηματικές σχέσεις μεταξύ των σημείων στον 3D κόσμο και το 2D επίπεδο προβολής πρέπει αρχικά να ορίσουμε 4 συστήματα συντεταγμένων [29] [28] [27]:

1. Το σύστημα συντεταγμένων κόσμου [X_W, Y_W, Z_W] το οποίο είναι ανεξάρτητο από τη θέση και τις παραμέτρους της κάμερας.
2. Το σύστημα συντεταγμένων της κάμερας [X_C, Y_C, Z_C].
3. Το σύστημα συντεταγμένων εικόνας [x, y], το οποίο αποτελεί την προοπτική προβολή των [X_C, Y_C, Z_C] στο επίπεδο απεικόνισης με σταθερό z = f (εστιακό μήκος).
4. Το σύστημα συντεταγμένων εικόνας σε μονάδες pixel [u, v].



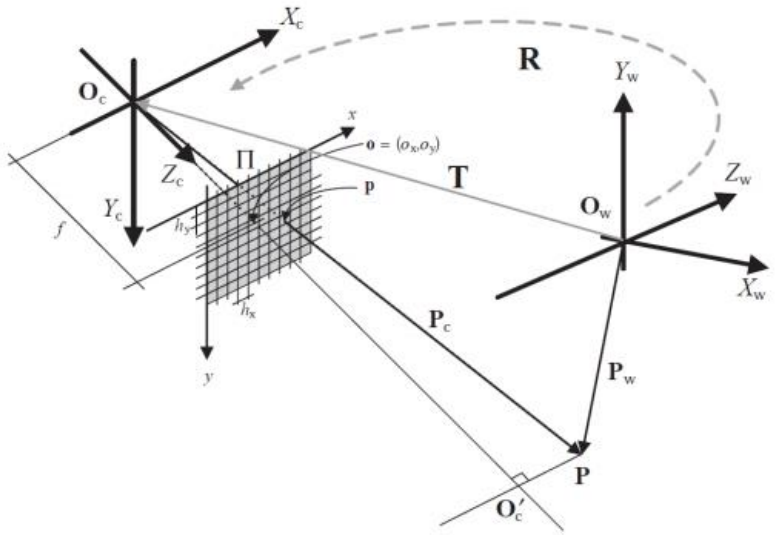
Εικόνα 128:Συστήματα Αναφοράς

Ένα σημείο στο παγκόσμιο σύστημα συντεταγμένων P_w μπορεί να μετασχηματιστεί σε ένα σημείο P_c του συστήματος της κάμερας με τη βοήθεια ενός μετασχηματισμού και μιας περιστροφής όπως φαίνεται στο σχήμα της εικόνας 125 και μπορεί να εκφραστεί ως εξής:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}_{3\times 4} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = [R|\mathbf{t}] \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad \text{(Σχέση 1)}$$

- Ο πίνακας $[R|t]_{3x4}$ ονομάζεται πίνακας εξωγενών παραμέτρων της κάμερας και περιλαμβάνει έναν πίνακα περιστροφής R_{3x3} και ένα 3D διάνυσμα μετατόπισης T. Το διάνυσμα T περιγράφει την αλλαγή θέσης των κέντρων του συστήματος συντεταγμένων της κάμερας O_c και του παγκόσμιου O_w. Ο πίνακας περιστροφής R είναι ορθογώνιος και περιστρέφει τους αντίστοιχους άξονες των δύο πλαισίων αναφοράς ώστε να τους ταυτίσει μεταξύ τους.

$$R = \begin{bmatrix} R_1 \\ R_2 \\ R_3 \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \quad \text{και} \quad T = O_W - O_C = \begin{bmatrix} T_1 \\ T_2 \\ T_3 \end{bmatrix}$$



Εικόνα 129 : Σχηματική αναπαράσταση συστημάτων αναφοράς [29] [27]

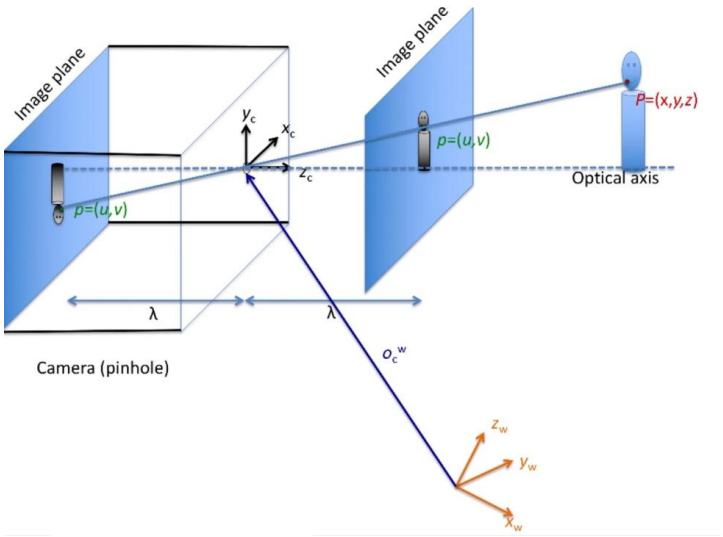
Όπως φαίνεται και στο παραπάνω σχήμα ο πίνακας των εξωγενών παραμέτρων είναι απαραίτητος για την μετάβαση από το σημείο του P_w του πραγματικού κόσμου στο σημείο P_c του συστήματος της κάμερας.

9.1.1 Απεικόνιση μέσω Pinhole κάμερας

Μια κάμερα μπορεί να μοντελοποιηθεί με πολλούς τρόπους, ανάλογα με τον επιθυμητό βαθμό ακρίβειας και την εφαρμογή για την οποία προορίζεται η χρήση της. Το απλούστερο μοντέλο μιας πραγματικής κάμερας περιλαμβάνει μία οπή (pinhole) και μια οθόνη απεικόνισης (επίπεδο απεικόνισης).

Η κάμερα μικρής-οπής (pinhole) αποτελεί μια απλούστευση του μοντέλου κάμερας λεπτού φακού [29]. Το άνοιγμα του λεπτού φακού θεωρείται ότι τείνει στο μηδέν, οπότε όλες οι ακτίνες εξαναγκάζονται να περνούν από το οπτικό κέντρο με αποτέλεσμα να μη διαθλώνται. Το διάφραγμα της κάμερας περιγράφεται ως μια οπή και δε χρησιμοποιούνται φακοί για να εστιάσουν το φως. Η pinhole κάμερα δεν περιλαμβάνει τα σφάλματα που προκαλούνται από τους φακούς όπως για παράδειγμα γεωμετρικές παραμορφώσεις ή θόλωση λόγω λανθασμένης εστίασης αντικειμένων. Τα προβλήματα που θα προκύψουν λόγω των εξιδανικεύσεων του μοντέλου μπορούν ωστόσο να παρακαμφθούν. Αυτό σημαίνει ότι μπορεί να χρησιμοποιηθεί για να περιγράψει με αρκετά καλή ακρίβεια τη λειτουργία της κάμερας στην υπολογιστική όραση. Το μαθηματικό μοντέλο της pinhole κάμερας ονομάζεται μοντέλο προοπτικής προβολής.

Επειδή η οπή βρίσκεται μεταξύ του 3D αντικειμένου και του επιπέδου απεικόνισης κάθε ακτίνα φωτός της σκηνής τη διαπερνά πριν φτάσει στο επίπεδο απεικόνισης στο οποίο δημιουργείται ένα ανεστραμμένο είδωλο του αντικειμένου [27].

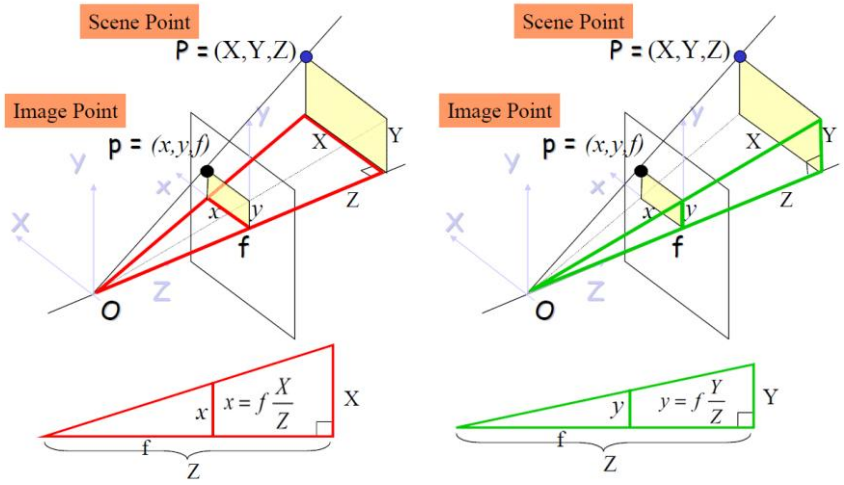


Εικόνα 130: Αναπαράσταση λειτουργίας Pinhole κάμερας

Δοθέντος ενός σημείου $Pc=(X_c,Y_c,Z_c)^T$ του πλαισίου της κάμερας (camera frame) πρέπει να λάβει χώρα ένας μετασχηματισμός που θα μας αντιστοιχίσει το παραπάνω σημείο με ένα σημείο $p=(x,y)^T$ του αισθητήρα της εικόνας. Στην πράξη η εικόνα που δημιουργεί μια ψηφιακή φωτογραφική μηχανή αποδίδεται μέσω pixels με αρχή συνήθως το πάνω-αριστερά pixel της φωτογραφίας. Οι συντεταγμένες της προβολής στην πρώτη περίπτωση ονομάζονται κανονικοποιημένες συντεταγμένες της κάμερας ενώ οι συντεταγμένες της προβολής στην δεύτερη περίπτωση ονομάζονται συντεταγμένες pixel. Για την ανακατασκευή μιας σκηνής είναι απαραίτητη η γνώση των κανονικοποιημένων συντεταγμένων.

Αν επιχειρήσουμε να κάνουμε μια υπολογιστική προσομοίωση της κάμερας θεωρώντας:

- ως κέντρο προβολής το (0,0,0) και επίπεδο προβολής (αισθητήρα εικόνας) z=f
- έστω $P(x_c,y_c,z_c)$ το σημείο στο σύστημα της κάμερας και
- p_c το σημείο κεντρικής προβολής του P στον αισθητήρα της κάμερας με συντεταγμένες x,y



Εικόνα 131 : Μετασχηματισμός σημείου από το πλαίσιο της Σκηνής στο επίπεδο προβολής της εικόνας

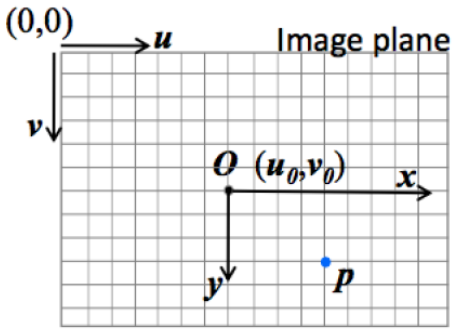
τότε όπως βλέπουμε και στο παραπάνω σχήμα εφαρμόζοντας το θεώρημα των όμοιων τριγώνων έχουμε τις σχέσεις :

$$\frac{x}{f}=\frac{X_C}{Z_C}\Rightarrow x=\frac{f\cdot X_C}{Z_C}\ (\text{Σχέση 2})\ \text{και}\ \frac{y}{f}=\frac{Y_C}{Z_C}\Rightarrow y=\frac{f\cdot Y_C}{Z_C}\ (\text{Σχέση 3})$$

Για να μετατρέψουμε το σημείο $p=(x,y)^T$ του επιπέδου προβολής στο σύστημα συντεταγμένων του pixel της εικόνας πρέπει να λάβουμε υπόψη μας τη μετατόπιση του συστήματος συντεταγμένων των pixel από το οπτικό κέντρο $O=(u_0,v_0)$της κάμερας καθώς επίσης τους συντελεστές κλιμάκωσης k_u και k_v για το μέγεθος των pixel και για τις δύο διαστάσεις έτσι ώστε [3]:

$$u=u_0+k_u x\Rightarrow u=u_0+\frac{k_u f X_C}{Z_C}\ (\text{Σχέση 4})$$

$$v=v_0+k_v y\Rightarrow v=v_0+\frac{k_v f Y_C}{Z_C}\ (\text{Σχέση 5})$$



Εικόνα 132: Από το επίπεδο προβολής στο σύστημα συντεταγμένων των pixel της εικόνας

Κατά τη διαδικασία την αναπαράστασης σημείων που ανήκουν σε σύστημα συντεταγμένων τριών διαστάσεων σε σημεία που ανήκουν σε συστήματα δύο διαστάσεων προκύπτουν μη γραμμικές εξισώσεις που δεν μπορούν να τις εκφράσουμε με τη μορφή πινάκων (λόγω της διαίρεσης με το Z_c). Για το λόγο αυτό "ανακαλύψαμε" τις ομογενείς συντεταγμένες όπου κάνουν δυνατή τη χρήση γραμμικών εξισώσεων [32]. Με τις ομογενείς συντεταγμένες αναπαριστούμε ένα σημείο που ανήκει στο επίπεδο (2D) σε ένα σημείο που ανήκει στον χώρο (3D) προσθέτοντας μία τρίτη "πλασματική" συντεταγμένη. Θεωρούμε ότι μπορούμε να προσδιορίζουμε ένα σημείο του επιπέδου (2D) με τρεις συντεταγμένες (x',y',z') ως εξής:

$$x=\frac{x'}{z'}\ ,\ \ y=\frac{y'}{z'}$$

δηλαδή (x,y)=(x',y',1)=(2x,2y,2)=(kx,ky,k) για κάθε k≠0 (Το k μπορεί να είναι είτε θετικός είτε αρνητικός). Με βάση λοιπόν τις ομογενείς συντεταγμένες οι συντεταγμένες (x,y) του

σημείου p του πλάνου της κάμερας (image plane) μπορούν να εκφραστούν ως εξής με τη μορφή πινάκων:

$$x = f\frac{X}{Z} \quad y = f\frac{Y}{Z} \quad \Leftrightarrow \quad \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Το σημείο p που αντιστοιχεί στο pixel της εικόνας περιγράφεται με ομογενείς συντεταγμένες ως εξής : $p = (u,v,1)^T$. Με βάση όλα τα παραπάνω η εξίσωση προβολής μπορεί να εκφραστεί πλέον με τη μορφή μητρών ως εξής:

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} k_u f & 0 & u_0 \\ 0 & k_v f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} \alpha_u & 0 & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = K \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} \quad \text{(Σχέση 6)}$$

όπου f είναι η εστιακή απόσταση εκφρασμένη σε μονάδες μέτρησης (συνήθως χιλιοστά), $[u_o,v_0]$ είναι το κέντρο του επιπέδου της εικόνας, τα k_u και k_v είναι οι διαστάσεις των pixel και συνεπώς τα α_u και α_v είναι η εστιακή απόσταση σε pixels. Τέλος θεωρούμε ότι ο παράγοντας κλίσης που εκφράζει τη κλίση των αξόνων x - y είναι μηδέν. Ο πίνακας K ονομάζεται μήτρα βαθμονόμησης εγγενών παραμέτρων της κάμερας και οι παράμετροι που περιέχει εξαρτώνται αποκλειστικά από την κατασκευή και τις ρυθμίσεις της κάμερας [25] [26] [32]. Στην παραπάνω εξίσωση παρατηρούμε επίσης ότι $\lambda=Z_c$ το οποίο συμπίπτει με την απόσταση του σημείου από το επίπεδο προβολής.

Συνδυάζοντας τις σχέσεις 1 και 6 καταλήγουμε στην εξίσωση προοπτική προβολής

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K\,[R|\mathbf{t}]_{3\times 4} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad \text{(Σχέση 7)}$$

Ο πίνακας K[R|t]3x4 ονομάζεται πίνακας προβολής M (projection matrix).

Συνεπώς για να αντιστοιχίσουμε τις συντεταγμένες ενός σημείου που ανήκει του παγκόσμιο σύστημα συντεταγμένων με τις συντεταγμένες ενός εικονοστοιχείου της εικόνας που συλλαμβάνει η κάμερα πρέπει πρώτα να έχουμε προσδιορίσει τις παραμέτρους των δύο βασικών πινάκων που λαμβάνουν χώρα σε αυτόν τον μετασχηματισμό, δηλαδή:

- του πίνακα $[R|t]_{3x4}$ των εξωγενών παραμέτρων της κάμερας που περιλαμβάνει έναν πίνακα περιστροφής R_{3x3} και ένα 3D διάνυσμα μετατόπισης T.

- και τον πίνακα K των ενδογενών παραμέτρων της κάμερας δηλαδή :
  - Η παράμετρος της προοπτικής προβολής. Για το μοντέλο της κάμερας οπής αυτή δίνεται από το εστιακό μήκος (focal length) f
  - Οι παράμετροι που μετασχηματίζουν το σύστημα συντεταγμένων της κάμερας [x,y] στο σύστημα συντεταγμένων της εικόνας σε μονάδες pixel [u,v]
  - Οι γεωμετρικές παραμορφώσεις που προκαλούνται από τους φακούς της κάμερας.

Ο προσδιορισμός των παραμέτρων αυτών των πινάκων γίνεται με τη διαδικασία της βαθμονόμησης της κάμερας. Εδώ αξίζει να σημειώσουμε ότι η εξίσωση προοπτικής προβολής δείχνει ότι κάθε σημείο εικόνας είναι η προβολή όλων των άπειρων τρισδιάστατων σημείων που βρίσκονται στην ακτίνα που διέρχεται από το ίδιο σημείο εικόνας και το κέντρο προβολής. Συνεπώς η χρήση μια μόνο κάμερας δεν είναι αρκετή για να εκτιμηθεί η απόσταση από ένα σημείο και απαιτούνται τουλάχιστον δύο κάμερες [32].

9.1.2 Βαθμονόμηση Κάμερας

Το 1987 ο Roger Tsai πρότεινε μια μέθοδο βαθμονόμησης [34] που συνίσταται στη μέτρηση της τρισδιάστατης θέσης των n ($n \geq 6$) σημείων ελέγχου σε έναν τρισδιάστατο στόχο βαθμονόμησης και τις αντίστοιχες συντεταγμένες δύο διαστάσεων που προβάλλονται στην εικόνα. Αυτό το πρόβλημα ονομάζεται "προοπτική από σημεία" ή "στάση κάμερας από αντιστοιχίες 3D σε 2D".

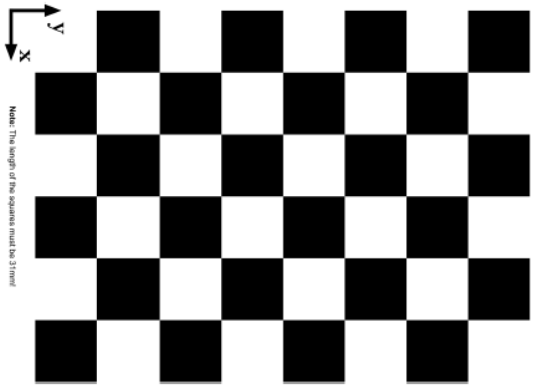
Το πρόβλημα βαθμονόμησης της κάμερας συχνά επιλύεται υπολογίζοντας τις εγγενείς και εξωγενείς παραμέτρους απευθείας από την εξίσωση προοπτικής προβολής ακολουθώντας τα παρακάτω βήματα κάνοντας χρήση ενός 3D μοτίβου (σκακιέρα).

1. Ανίχνευση άκρων
2. Προσαρμογή ευθείας γραμμής στις ανιχνευμένες άκρες
3. Εντοπισμός των σημείων τομής των ευθειών για τον προσδιορισμό των γωνιών της εικόνας
4. Χρησιμοποίηση περισσοτέρων των έξι (6) σημείων (ιδανικά περισσότερα από είκοσι 20)

Η αρχική βαθμονόμηση του Tsai [34] συνήθιζε να λαμβάνει υπόψη δύο διαφορετικά εστιακά μήκη a_u, a_v (το οποίο σήμαινε ότι τα pixels δεν ήταν τετράγωνα) και έναν συντελεστή κλίσης ($K_{12} \neq 0$ στον πίνακα K θεωρώντας τα pixels παραλληλόγραμμα αντί ορθογώνια) για να ληφθούν υπόψη πιθανές εσφαλμένες ευθυγραμμίσεις (μικρές περιστροφές των x,y) μεταξύ του επιπέδου προβολής της εικόνας και του φακού. Σήμερα οι περισσότερες κάμερες είναι καλά κατασκευασμένες και για αυτό το λόγο κάνουμε την παραδοχή ότι $\frac{a_u}{a_v} = 1$και $K_{12}=0$.

Η βαθμονόμηση του Tsai βασιζόταν στον αλγόριθμο DLT [30] ο οποίος απαιτούσε τα σημεία να μην ήταν τοποθετημένα στο ίδιο επίπεδο. Μία εναλλακτική μέθοδος που εφευρέθηκε από τον Zhang [36] και καθιερώθηκε και αποτελεί σήμερα την τυποποιημένη

μέθοδο βαθμονόμησης απαιτεί τη χρήση επίπεδου πλέγματος (που μοιάζει με σκακιέρα) σαν και αυτό της παρακάτω εικόνας και λήψη εικόνων αυτής της πινακίδας από διαφορετικούς προσανατολισμούς.



Εικόνα 133: Πινακίδα βαθμονόμησης κάμερας

Ο σκοπός είναι ο υπολογισμός των παραμέτρων των πινάκων K, R και t ώστε να ικανοποιείται η εξίσωσης της προοπτικής προβολής. Η ακτινική παραμόρφωση παραμελείται για τώρα αλλά είναι δυνατόν να ληφθεί υπόψη. Δεδομένου ότι τα σημεία βρίσκονται στο επίπεδο έχουμε $Z_w=0$. Οπότε από την εξίσωση 7 προκύπτει:

$$\begin{bmatrix} \tilde{u} \\ \tilde{v} \\ \tilde{w} \end{bmatrix} := \lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underbrace{H_{3\times 3}}_{\text{homography}} \cdot \begin{bmatrix} X_w \\ Y_w \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ 1 \end{bmatrix}$$

(Σχέση 8)

όπου h_i είναι η i σειρά του H. Στο σημείο πρέπει να σημειωθεί ότι ο πίνακας H μπορεί να πολλαπλασιαστεί με μια αυθαίρετη μη μηδενική σταθερά c χωρίς να μεταβληθεί ο μετασχηματισμός. Κατά συνέπεια ο πίνακας H έχει 8 βαθμούς ελευθερίας παρόλο που αποτελείται από 9 στοιχεία.

Η μετατροπή από ομογενείς συντεταγμένες σε συντεταγμένες pixel οδηγεί στις σχέσεις:

$$u = \frac{\tilde{u}}{\tilde{w}} = \frac{h_1 P}{h_3 P} \Rightarrow h_1 P = u h_3 P \Rightarrow h_1 P - u h_3 P = 0 \Rightarrow (h_1 - u h_3) P = 0$$

(Σχέση 9)

$$v = \frac{\tilde{v}}{\tilde{w}} = \frac{h_2 P}{h_3 P} \Rightarrow h_2 P = v h_3 P \Rightarrow h_2 P - v h_3 P = 0 \Rightarrow (h_2 - v h_3) P = 0$$

(Σχέση 10)

όπου $P=(X_w, Y_w, 1)^T$.

Από την αναδιάταξη των όρων έχουμε:

$$\begin{matrix}(\mathbf{h}_1 - u\mathbf{h}_3)\cdot\mathbf{P} = 0 \\ (\mathbf{h}_2 - v\mathbf{h}_3)\cdot\mathbf{P} = 0\end{matrix} \Longrightarrow \begin{matrix}\mathbf{P}^\intercal\cdot\mathbf{h}_1^\intercal + 0\cdot\mathbf{h}_2^\intercal - u\mathbf{P}^\intercal\cdot\mathbf{h}_3^\intercal = 0 \\ 0\cdot\mathbf{h}_1^\intercal + \mathbf{P}^\intercal\cdot\mathbf{h}_2^\intercal - v\mathbf{P}^\intercal\cdot\mathbf{h}_3^\intercal = 0\end{matrix}$$

$$\Longrightarrow \begin{bmatrix}\mathbf{P}^\intercal & \mathbf{0}^\intercal & -u\mathbf{P}^\intercal \\ \mathbf{0}^\intercal & \mathbf{P}^\intercal & -v\mathbf{P}^\intercal\end{bmatrix}_{2\times 9}\begin{bmatrix}\mathbf{h}_1^\intercal \\ \mathbf{h}_2^\intercal \\ \mathbf{h}_3^\intercal\end{bmatrix}_{9\times 1} = \begin{bmatrix}0 \\ 0\end{bmatrix}_{2\times 1} \quad \text{(Σχέση 11)}$$

Για n σημεία αυτές οι εξισώσεις μπορούν να εκφραστούν με μήτρες ως εξής

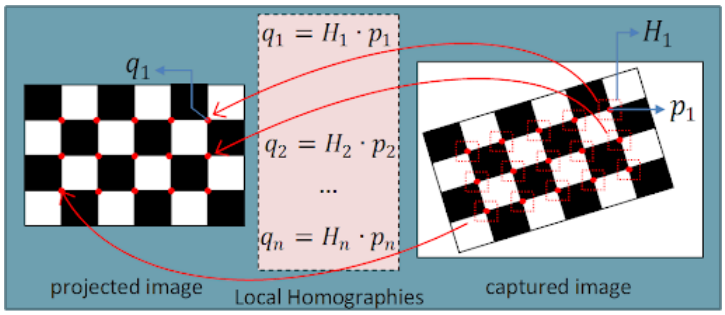
$$\begin{bmatrix}\mathbf{P}_1^\intercal & \mathbf{0}^\intercal & -u_1\mathbf{P}_1^\intercal \\ \mathbf{0}^\intercal & \mathbf{P}_1^\intercal & -v_1\mathbf{P}_1^\intercal \\ \dots & \dots & \dots \\ \mathbf{P}_n^\intercal & \mathbf{0}^\intercal & -u_n\mathbf{P}_n^\intercal \\ \mathbf{0}^\intercal & \mathbf{P}_n^\intercal & -v_n\mathbf{P}_n^\intercal\end{bmatrix}_{2n\times 9}\begin{bmatrix}\mathbf{h}_1^\intercal \\ \mathbf{h}_2^\intercal \\ \mathbf{h}_3^\intercal\end{bmatrix}_{9\times 1} = \begin{bmatrix}0 \\ 0 \\ \vdots \\ 0 \\ 0\end{bmatrix}_{2n\times 1} \Longrightarrow Q\cdot H = 0$$

(Σχέση 12)

Από τα παραπάνω προκύπτει ότι κάθε αντιστοιχία σημείων μας παρέχει 2 επιπλέον εξισώσεις και σε συνδυασμό με το γεγονός ότι ο πίνακας H έχει 8 βαθμούς ελευθερίας καταλήγουμε στο ότι απαιτούνται τουλάχιστον 4 αντιστοιχίες σημείων για τον υπολογισμό του H [3]. Για τη λύση του παραπάνω συστήματος και την εύρεση του διανύσματος h ελαχιστοποιούμε το μέτρο της σχέσης $\|QH\|^2$ με τον περιορισμό $\|H\|^2$=1. Έτσι αφού πραγματοποιήσουμε παραγοντοποίηση ιδιόμορφων τιμών για τον πίνακα Q, η λύση του h προκύπτει ως το ιδιοδιάνυσμα που αντιστοιχεί στην μηδενική ιδιοτιμή του.

Στην πράξη η παραπάνω μέθοδος έχει αποδειχθεί ότι εξαρτάται άμεσα από το κέντρο και την κλίμακα του συστήματος συντεταγμένων του αντιστοιχισμένων σημείων. Η παραπάνω ιδιότητα είναι ανεπιθύμητη καθώς κάνει τον αλγόριθμο που παρουσιάστηκε αρκετά ασταθή. Για να υπερκεράσουμε το παραπάνω πρόβλημα της εξάρτησης του αποτελέσματος από το σύστημα συντεταγμένων των σημείων, πραγματοποιούμε κανονικοποίηση των συντεταγμένων τους [37]. Η κανονικοποίηση ξεκινά μετασχηματίζοντας των σύνολο των σημείων pi σε ένα νέο σύνολο σημείων $\widehat{p_i}$στο οποίο το κέντρο είναι το σημείο $[0,0]^T$ και τα σημεία απέχουν κατά μέσο όρο από αυτό απόσταση $\sqrt{2}$. Ο παραπάνω μετασχηματισμός υλοποιείται με τον πολλαπλασιασμό των ομογενών συντεταγμένων των σημείων του συνόλου p_i με έναν 3×3 πίνακα T. Η παραπάνω διαδικασία εφαρμόζεται και για τα σημεία του συνόλου p_i' μετασχηματίζοντας τα στα $\widehat{p_i}'$ με την χρήση του πίνακα T'. Στην συνέχεια υπολογίζεται ο ομογραφικός πίνακας $\widehat{H}$ για τις κανονικοποιημένες συντεταγμένες. Τέλος ο ζητούμενος πίνακας που απεικονίζει τα σημεία pi στα pi' υπολογίζεται από την παρακάτω σχέση :

$$H = T'^{-1}\widehat{H}T \quad \text{(Σχέση 13)}$$



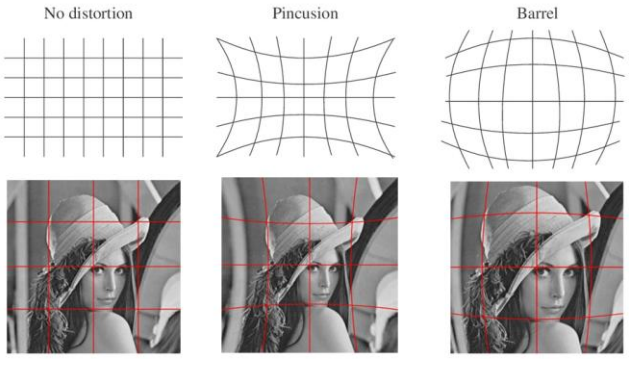
Εικόνα 134 : Διαδικασία βαθμονόμησης κάμερας με την μέθοδο της Ομογραφίας

Το μέσο σφάλμα επαναπροβολής που ονομάζεται επίσης υπολειπόμενο, υπολογίζεται ως μέρος της διαδικασίας βαθμονόμησης. Μετρά την απόσταση (σε εικονοστοιχεία) μεταξύ του παρατηρούμενου σημείου και του τρισδιάστατού σημείου που έχει υποβληθεί εκ νέου στην κάμερα. Το σφάλμα επαναπροβολής δίνει ένα ποσοτικό μέτρο της ακρίβειας της βαθμονόμησης. Ιδανικά θα πρέπει να είναι μηδέν (0) αλλά και μία τιμή μικρότερη του 0,3 συνήθως γίνεται αποδεκτή.

9.1.3 Παραμόρφωση φακού

Ένας άλλος παράγοντας που πρέπει να ληφθεί υπόψη στους παραπάνω μετασχηματισμούς ώστε η αντιστοιχίες των συντεταγμένων των σημείων του τρισδιάστατού χώρου με τις συντεταγμένες των εικονοστοιχείων της δισδιάστατης εικόνας να είναι όσο το δυνατόν πιο ρεαλιστικές και αξιόπιστες είναι παραμόρφωση που ενδέχεται να εισάγει στην εικόνα ο φακός της κάμερας.

Συγκεκριμένα η κάμερα που φέρουν τα Duckiebots διαθέτουν έναν ο φακό (Fisheye-lens) που πράγματι εισάγει κάποια παραμόρφωση στην εικόνα. Ειδικότερα εισάγει τη επονομαζόμενη παραμόρφωση βαρελιού (Barrel distortion) κατά την οποία η μεγέθυνση της εικόνας μειώνεται με την απόσταση από τον οπτικό άξονα Το προφανές αποτέλεσμα είναι η εικόνα να φαίνεται ότι "χαρτογραφείται" γύρω από ένα βαρέλι ή σφαίρα. Αυτό είναι μία συνέπεια των φακών Fisheye στην προσπάθεια τους να χαρτογραφήσουν ένα ευρύ πεδίο της σκηνής. Αντίθετα στην παραμόρφωση Pincushion η μεγέθυνση της εικόνας αυξάνεται με την απόσταση από τον οπτικό άξονα [32]. Σε κάθε περίπτωση αυτή η παραμόρφωση θα πρέπει να προστεθεί στις εξισώσεις προοπτικής προβολής.



Εικόνα 135: Διαφορετικοί τύποι παραμόρφωσης

Το τυπικό μοντέλο ακτινικής παραμόρφωσης είναι ένας μετασχηματισμός από τις ιδανικές συντεταγμένες εικόνας (u, v), δηλαδή χωρίς παραμόρφωση, στις πραγματικές παρατηρήσιμες συντεταγμένες, παραμορφωμένες (u_d, v_d). Η ποσότητα παραμόρφωσης των συντεταγμένων της παρατηρούμενης εικόνας είναι μια μη γραμμική συνάρτηση της ακτινικής τους απόστασης [3]. Για τους περισσότερους φακούς, ένα κυβικό μοντέλο παραμόρφωσης παράγει καλά αποτελέσματα:

$$\begin{bmatrix} u_d \\ v_d \end{bmatrix} = (1 + k_1 r^2 + k_2 r^4) \begin{bmatrix} u - u_0 \\ v - v_0 \end{bmatrix} + \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} \quad \text{(Σχέση 14)}$$

όπου $r^2=(u - u_0)^2 + (u-u_0)^2$, $(u_0,v_0)^T$ είναι η μετατόπιση του συστήματος συντεταγμένων των pixel στο οπτικό κέντρο της κάμερας ενώ τα k_1 και K_2 είναι οι παράμετροι παραμόρφωσης που μπορούν να εκτιμηθούν με τη διαδικασία βαθμονόμησης της κάμερας. Επειδή το μοντέλο παραμόρφωσης δεν είναι γραμμικό, η συνάρτηση προβολής δεν είναι και αυτή γραμμική σε ομογενείς συντεταγμένες. Για το λόγο αυτό την "σπάμε" στα ακόλουθα βήματα:

- Αντιστοίχιση του σημείου P_W που ανήκει στο παγκόσμιο σύστημα συντεταγμένων στο σύστημα αναφοράς της κάμερας P_C εφαρμόζοντας την εξίσωση :

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix}_{3\times4} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = [R|\mathbf{t}] \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

- Προβολή του σημείου στο πλάνο της κάμερας (image plane) για να λάβουμε τις κανονικοποιημένες συντεταγμένες $p=(x,y)^T$

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \frac{X_C}{Z_C} \\ \frac{Y_C}{Z_C} \end{pmatrix} \text{(Σχέση 15)}$$

- Εφαρμογή παραμόρφωσης φακού στο σημείο p για να πάρουμε τις παραμορφωμένες κανονικοποιημένες συντεταγμένες $p_d=(x',y')^T$ με εφαρμογή της εξίσωσης

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = (1 + k_1 r^2 + k_2 r^4) \begin{pmatrix} x \\ y \end{pmatrix} \text{(σχέση 16)}$$

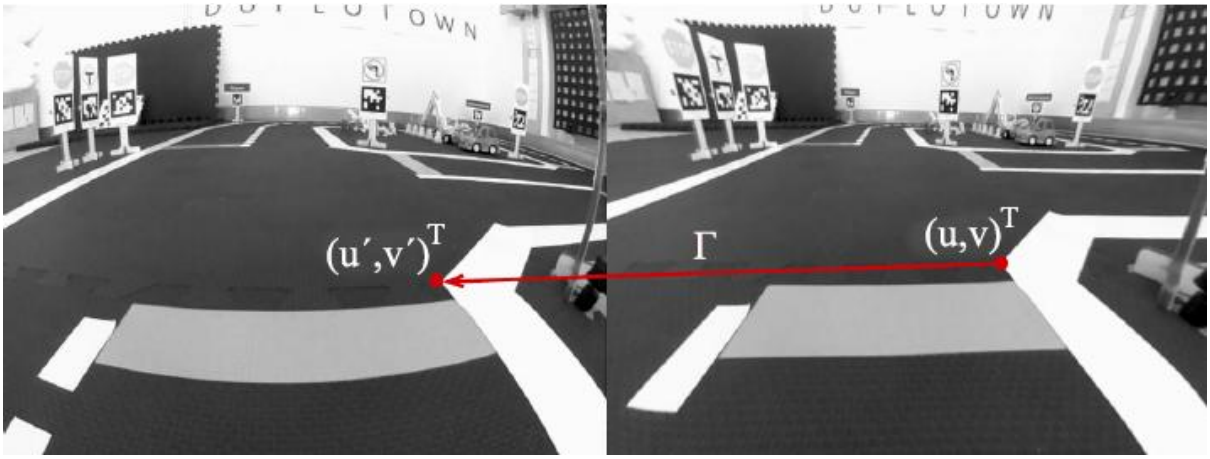
όπου $r^2=x^2+y^2$

- Μετατροπή των παραμορφωμένων κανονικοποιημένων συντεταγμένων pd για να εξάγουμε τις διακριτές συντεταγμένες του εικονοστοιχείου $(u,v)^T$.

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \text{(σχέση 17)}$$

Ενσωματώνοντας στη συνάρτηση προβολής και την στρέβλωση της εικόνας από τον φακό είναι πλέον δυνατή η δημιουργία μια μη παραμορφωμένης εικόνας I_u από την αρχική παραμορφωμένη I_d. Κάθε εικονοστοιχείο της I_u μπορεί να αντιστοιχηθεί στην I_d ως εξής :

$I_u(u,v)=I_d(\Gamma(u,v))$, όπου $\Gamma(u,v)=(u',v')$ η συνάρτηση που αντιστοιχεί συντεταγμένες pixel που ανήκουν στην μη παραμορφωμένη εικόνα με συντεταγμένες pixel που ανήκουν στην εικόνα που έχει υποστεί κάποιας μορφής στρέβλωση. Δεδομένου ότι σημεία Γ(u,v) δεν είναι ακέραιοι θα πρέπει να γίνει παρεμβολή πλησιέστερου γείτονα ή διγραμμική παρεμβολή [38].



Εικόνα 136: Αντιστοίχιση συντεταγμένων pixel ανάμεσα σε παραμορφωμένη και μη εικόνα

Οι τεχνικές βαθμονόμησης όπως η μέθοδος DLT και η Zhang ενώ είναι υπολογιστικά γρήγορες έχουν το μειονέκτημα ότι δεν μπορούν να ενσωματώσουν την παραμόρφωση του φακού στον υπολογισμό στις εξισώσεις της προοπτικής προβολής [38]. Για το λόγο αυτό, για είναι δυνατή δηλαδή η εκτίμηση και των παραμέτρων παραμόρφωσης k_1 και k_2 πρέπει να χρησιμοποιηθεί ένα μη γραμμικό μοντέλο κάμερας. Για το πρόβλημα αυτό χρησιμοποιείται γενικά η μέθοδος βελτιστοποίησης Levenberg-Marquardt καθώς παρέχει γρήγορη σύγκλιση[38]. Οι αρχικές παράμετροι του μη γραμμικού μοντέλου της κάμερας, το οποίο είναι πρόβλημα ελαχιστοποίησης τετραγώνων, πρέπει να οριστούν ως το αποτέλεσμα είτε της μεθόδου DLT είτε της μεθόδου Zhang's από τη στιγμή που η πιθανότητα εύρεσης ενός παγκόσμιου ελάχιστου και όχι μόνο ενός τοπικού ελάχιστου αυξάνεται.

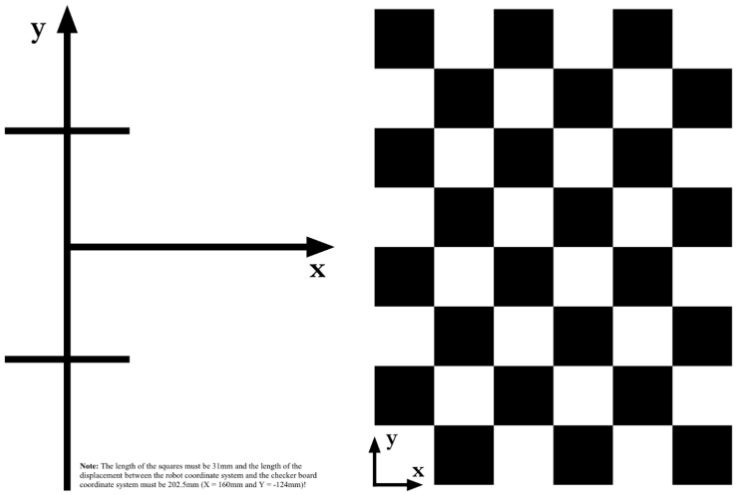
Γενικά ο αλγόριθμος της βαθμονόμησης της κάμερας του Duckiebot περιλαμβάνει τα εξής βήματα:

1. Αρχικοποίηση εγγενών παραμέτρων της κάμερας αγνοώντας τις παραμέτρους παραμόρφωσης
2. Υπολογισμός της αρχικής στάσης - πόζας της κάμερας (εύρεση εξωγενών παραμέτρων) χρησιμοποιώντας την τρέχουσα εκτίμηση των εγγενών παραμέτρων.
3. Ελαχιστοποίηση του σφάλματος επαναπροβολής (βελτιστοποίηση ενδογενών και εξωγενών παραμέτρων καθώς και παραμέτρων παραμόρφωσης) μέσω μη γραμμικής βελτιστοποίησης όπως η μέθοδος Levenberg-Marquardt.

9.2 Βαθμονόμηση της κάμερας του Duckiebot

9.2.1 Βαθμονόμηση εγγενών παραμέτρων της κάμερας του Duckiebot

Επειδή κάθε κάμερα είναι διαφορετική είναι αναγκαίο να καταγράψουμε τις αυτές τις μικρές διαφορές που ενδέχεται να έχει η κατασκευή της και να τις λάβουμε υπόψη στους μετασχηματισμούς που θα ακολουθήσουν ώστε να έχουμε ένα πιο αντιπροσωπευτικό και ποιοτικό αποτέλεσμα. Η βαθμονόμηση των εγγενών παραμέτρων της κάμερας γίνεται με τη χρήση μιας πινακίδας βαθμονόμησης, ενός δηλαδή προκαθορισμένου μοτίβου όπως αυτό που απεικονίζεται στην εικόνα 133.



Εικόνα 137 : Πινακίδα βαθμονόμησης Duckiebot

Η παραπάνω πινακίδα που μοιάζει με σκακιέρα περιέχει τετράγωνα πλευράς 0.013m (=3.1cm) και είναι σε εκτυπωμένη σε χαρτί μεγέθους Α3. Για είναι επιτυχής η διαδικασία βαθμονόμησης θα πρέπει να στερεωθεί σε μία άκαμπτη επιφάνεια η οποία όπως να μπορεί να κινηθεί κατά τη διάρκεια της διαδικασίας και αρχικά τοποθετείται απέναντι από το Duckiebot (εικόνα...)

Η διαδικασία της βαθμονόμησης [40] των εγγενών παραμέτρων της κάμερας πραγματοποιείται εκκινώντας την σχετική υπηρεσία του ROS με τη βοήθεια του Duckietown Shell από το laptop πληκτρολογώντας :

$ dts duckiebot calibrate_intrinsics Duckiebot_Name

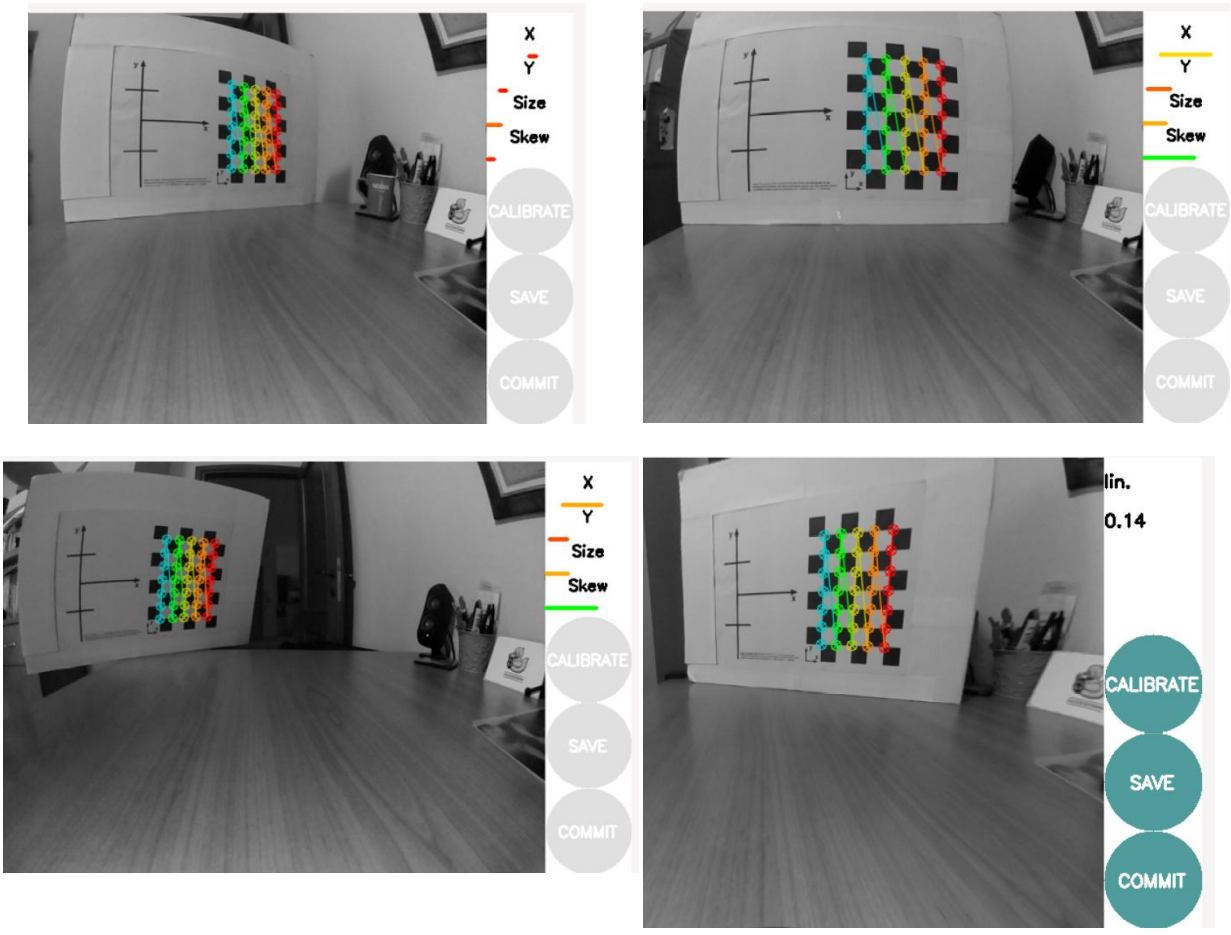
Εικόνα 138 : Στιγμιότυπο από τη διαδικασία βαθμονόμησης των ενδογενών παραμέτρων της κάμερας του Duckiebot

Στο laptop αναδύεται ένα παράθυρο όπως αυτό της παρακάτω εικόνας

Εικόνα 139 : Περιβάλλον βαθμονόμησης ενδογενών παραμέτρων του ROS

Στο γραφικό αυτό περιβάλλον παρατηρούμε στο κέντρο του την προσλαμβάνουσα από την κάμερα του Duckiebot εικόνα. Στο σημείο αυτό να τονίσουμε ότι θα πρέπει να δούμε τις χρωματιστές γραμμές να επικαλύπτουν τη "σκακιέρα" της πινακίδας βαθμονόμησης. Αν αυτό δεν συμβαίνει θα πρέπει να μειώσουμε την απόσταση ανάμεσα στο Duckiebot και του μοτίβου βαθμονόμησης. Επίσης είναι σημαντικό αφού εστιάσουμε στην πινακίδα βαθμονόμησης στρέφοντας το δακτύλιο της κάμερα να μην τον αγγίξουμε ξανά καθώς αυτό θα ακυρώσει τη διαδικασία.

Στη δεξιά στήλη διακρίνουμε τέσσερις μικρές μπάρες με τις ενδείξεις X, Y, Size και Skew. Το X υποδηλώνει το παρατηρούμενο οριζόντιο εύρος, το Y το κατακόρυφο, η ένδειξη Size αναφέρεται στο παρατηρούμενο εύρος κατά τον άξονα Z (εμπρός και πίσω από την κατεύθυνση της κάμερας) ενώ η ένδειξη Skew αφορά τη σχετική κλίση μεταξύ της πινακίδας βαθμονόμησης και της κατεύθυνσης της κάμερας. Οι μπάρες που αντιστοιχούν στις ενδείξεις αυτές δηλώνουν με το χρώμα και το μέγεθος τους το σύνολο των δεδομένων που έχουν συλλεχθεί για κάθε μέγεθος. Μετακινώντας τη πινακίδα βαθμονόμησης ως προς τους τρεις άξονες (X - Y - Z) και αλλάζοντας την κλίση της τα δεδομένα που συλλέγει η εφαρμογή του ROS εμπλουτίζονται και οι μπάρες μεγαλώνουν. Όταν οι παρατηρήσεις που συλλέγονται θεωρηθούν επαρκής από το πρόγραμμα βαθμονόμησης οι μπάρες πρασινίζουν. Όταν όλες γίνουν πράσινες ενεργοποιείται το κουμπί με την ένδειξη CALIBRATE καθώς και τα κουμπιά SAVE και COMMIT. Για να αποθηκευτούν τα αποτελέσματα της διαδικασίας εσωτερικής βαθμονόμησης της κάμερας πατάμε το κουμπί COMMIT.

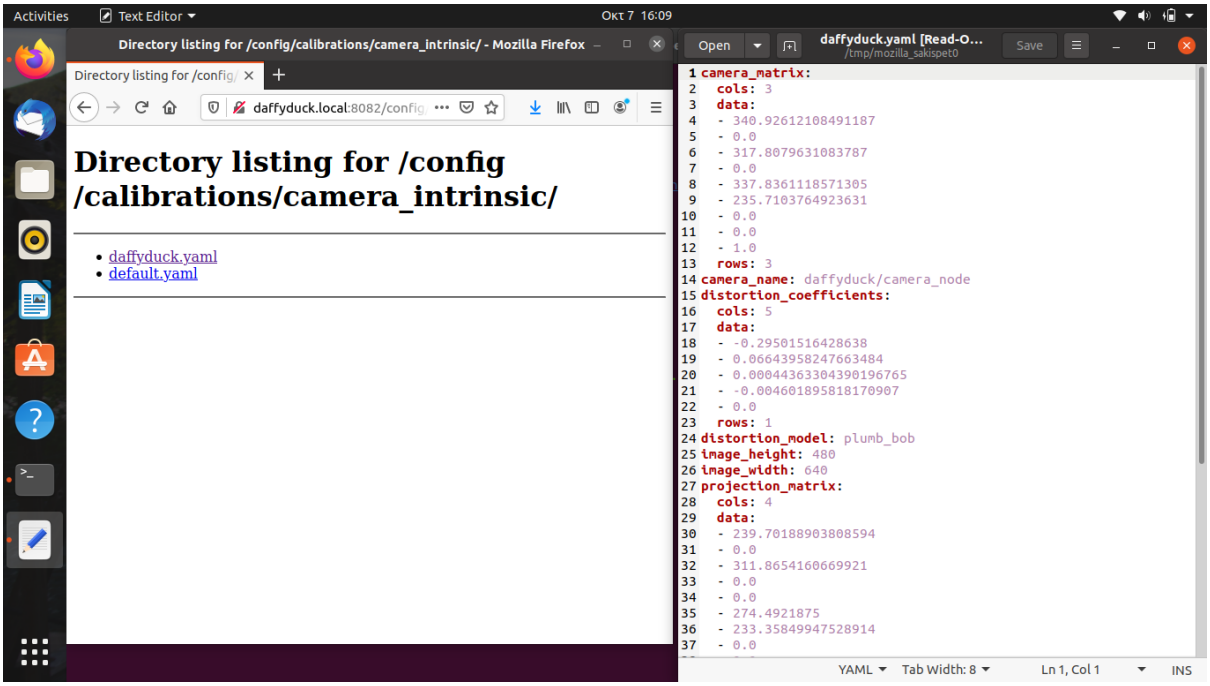


Εικόνα 140 : Στιγμιότυπα από τη διαδικασία βαθμονόμησης (calibration dance)

Τα αποτελέσματα της διαδικασίας της βαθμονόμησης των εγγενών χαρακτηριστικών της κάμερας που ονομάζεται και Calibration dance καθώς η πινακίδα πρέπει να κινηθεί αρκετές φορές μπροστά από το Duckiebot προκειμένου αυτή να ολοκληρωθεί , αποθηκεύονται σε ένα αρχείο στο Duckiebot τύπου yaml που έχει ως όνομα το αναγνωριστικό του Duckiebot στο δίκτυο. Το αρχείο αυτό βρίσκεται στη διαδρομή

/data/config/calibrations/camera_intrinsic/Duckiebot_Name.

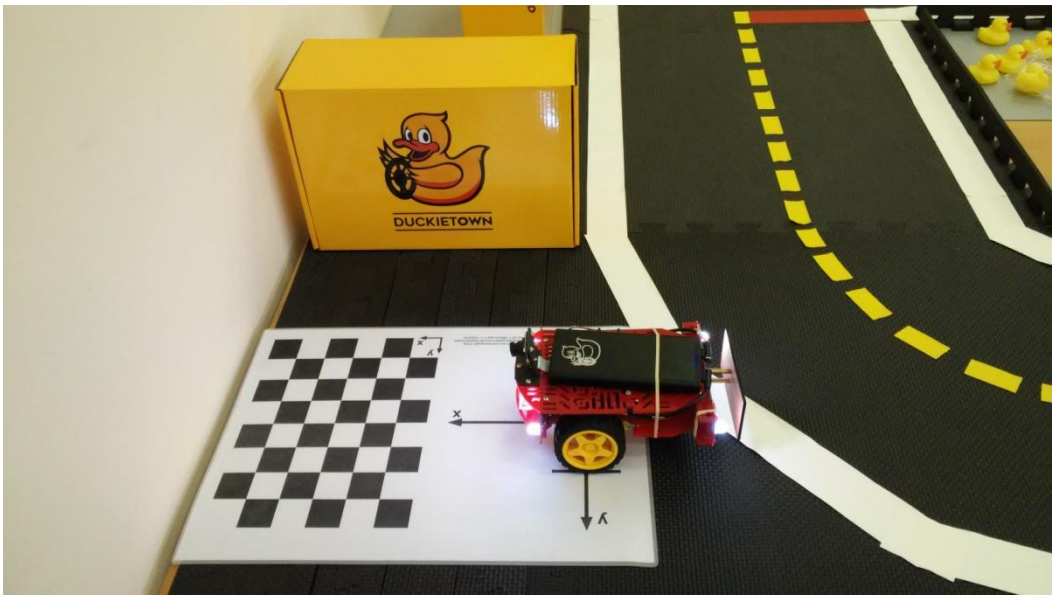
Μπορούμε να προσπελάσουμε το αρχείο είτε μέσω του Partainer είτε μέσω του Dashboard είτε χρησιμοποιώντας την εντολή της απομακρυσμένης διαχείρισης ssh. Η παρακάτω εικόνα απεικονίζει την προσπέλαση του αρχείου εσωτερικής βαθμονόμησης μέσω του Portainer γράφοντας στην γραμμή διεύθυνσης του φυλλομετρητή: http://Duckiebot_Name:8082 οπότε και αποκτούμε πρόσβαση στο σύστημα αρχείων του Duckiebot μέσα από ένα web interface.



Εικόνα 141 : Αρχείο καταγραφής της βαθμονόμησης των ενδογενών παραμέτρων της κάμερας του Duckiebot

9.2.2 Βαθμονόμηση εξωγενών παραμέτρων της κάμερας

Για τη διαδικασία της βαθμονόμησης των εξωγενών παραμέτρων της κάμερας χρησιμοποιούμε πάλι την ίδια πινακίδα - "σκακιέρα" που όμως αυτή τη φορά την τοποθετούμε σε μία επίπεδη επιφάνεια (τραπέζι, πάτωμα, κτλ). Πάνω στην πινακίδα τοποθετούμε το Duckiebot προσέχοντας οι τροχοί του να βρίσκονται ακριβώς επάνω στις δύο γραμμές του άξονα y και η κάμερα να βρίσκεται προς τη μεριά της σκακιέρας (άξονας x) όπως φαίνεται και την εικόνα 142. Εκείνο για το οποίο θα πρέπει να μεριμνήσουμε είναι να μην παρεμβάλλονται άλλα αντικείμενα μέσα στο οπτικό πεδίο της κάμερας, να φροντίσουμε δηλαδή να έχουμε ένα "καθαρό" φόντο (background) γεγονός που θα βοηθήσει στην επιτυχή ολοκλήρωση της διαδικασίας.

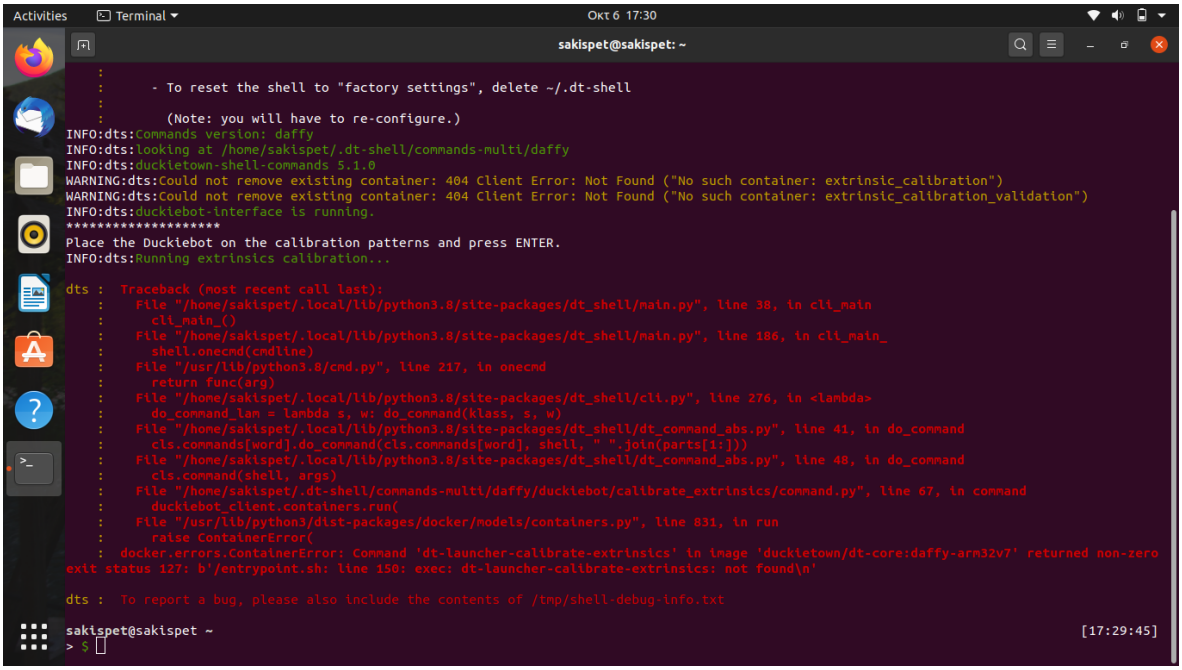


Εικόνα 142: Τοποθέτηση Duckiebot στην πινακίδα για την βαθμονόμηση των εξωγενών παραμέτρων της κάμερας

Έπειτα εκκινούμε τη σχετική υπηρεσία βαθμονόμησης του ROS μέσω του Duckietown shell πληκτρολογώντας στη γραμμή εντολών

$ dts duckiebot calibrate_extrinsics Duckiebot_Name

Στην εκκίνηση του παραπάνω container θα εμφανιστεί ένα μήνυμα το οποίο θα μας προτρέψει να τοποθετήσουμε το Duckiebot πάνω στο μοτίβο βαθμονόμησης και να πατήσουμε Enter. Υπάρχει βέβαια πάντα η περίπτωση κάτι να πάει στραβά και μην μπορεί να εκκινήσει η σχετική υπηρεσία όποτε θα εμφανιστεί ένα μήνυμα όπως αυτό του παρακάτω στιγμιότυπου.

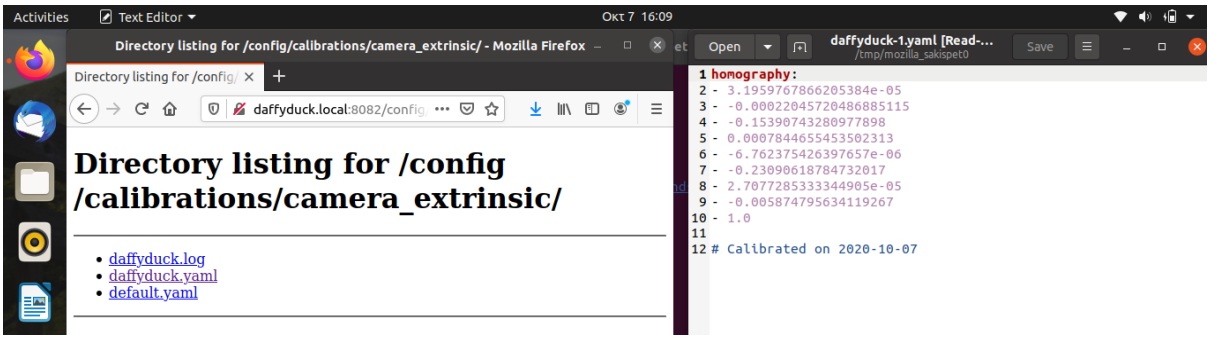


Εικόνα 143 : Σφάλμα κατά την εκκίνηση της υπηρεσίας βαθμονόμησης του ROS

Σε αυτή περίπτωση θα πρέπει να αναβαθμίσουμε την εικόνα (image) του dt-core που του Duckiebot αντλώντας την πιο πρόσφατή έκδοση του από το αποθετήριο. Συνεπώς αφού συνδεθούμε μέσω της απομακρυσμένης διαχείρισης (ssh) στο Duckiebot πληκτρολογούμε

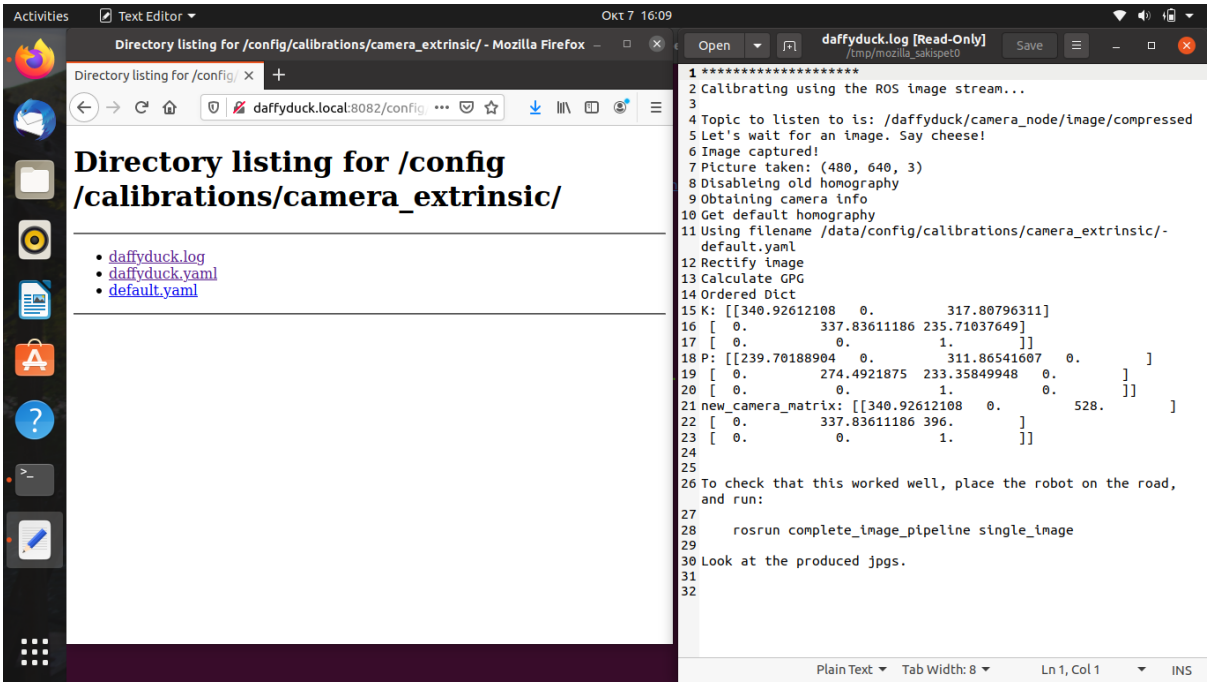
Duckiebot $ sudo docker pull duckietown/dt-core:daffy:arm32v7

Η παραπάνω διαδικασία μπορεί να διαρκέσει κάποια λεπτά. Όταν ολοκληρωθεί επανεκκινούμε το Duckiebot και επιχειρούμε εκ νέου τη διαδικασία της εξωτερικής βαθμονόμησης. Αν όλα πάνε κατ' ευχήν θα εμφανιστεί η λέξη Done που δηλώνει την πετυχημένη ολοκλήρωση της διαδικασίας. Για να επιβεβαιώσουμε το γεγονός αποκτούμε πρόσβαση στο σύστημα αρχείων του Duckiebot και διαπιστώνουμε αν έχει δημιουργηθεί αρχείο Duckiebot_Name.yaml μέσα στον κατάλογο camera_extrinsic.



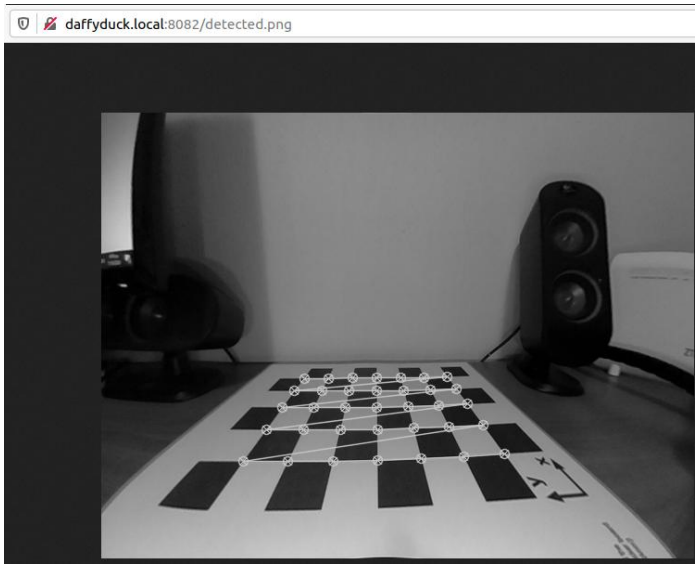
Εικόνα 144 : Αρχείο καταγραφής της βαθμονόμησης των εξωγενών παραμέτρων.

Επίσης στον ίδιο κατάλογο θα πρέπει να έχει δημιουργηθεί και ένα log αρχείο



Εικόνα 145: Αρχείο καταγραφής διαδικασίας βαθμονόμησης

Και τέλος μέσα στον κατάλογο calibrations πρέπει να υπάρχει μία εικόνα png με το όνομα detected όπως αυτή που φαίνεται παρακάτω



Εικόνα 146: Αποθηκευμένη εικόνα από τη διαδικασία βαθμονόμησης εξωγενών παραμέτρων.

9.3 Βαθμονόμηση τροχών

Οι κινητήρες που φέρει το Duckiebot ονομάζονται και κινητήρες ελεγχόμενης τάσης. Αυτό πρακτικά σημαίνει ότι η ταχύτητα κάθε κινητήρα είναι ανάλογη της τάσης η οποία εφαρμόζεται σε αυτόν. Αρχικά θεωρούμε ότι αφού έχουμε τους ίδιους κινητήρες αριστερά και δεξιά εφαρμόζοντας την ίδια τάση θα έχουν ακριβώς την ίδια απόκριση. Στην πράξη όμως κάθε κινητήρας ενδέχεται να ανταποκρίνεται με ελαφρώς διαφορετικό τρόπο δεχόμενος το ίδιο σήμα τάσης. Ομοίως και οι τροχοί που φέρει το Duckiebot μπορεί να φαίνονται πανομοιότυποι αλλά μπορεί να έχουν κάποιες μικρές διαφορές γεγονός που έχει επίπτωση στην κινηματική του συμπεριφορά. Έτσι για τους παραπάνω λόγους όταν οδηγούμε χειροκίνητα το Duckiebot και επιθυμούμε να πηγαίνει για παράδειγμα ευθεία εμπρός αυτό μπορεί να περικλείνει λοξοδρομώντας είτε προς τα αριστερά είτε προς τα δεξιά. Μικρές διαφοροποιήσεις στη λειτουργία των δύο κινητήρων μπορεί να έχουν ως αποτέλεσμα ο αριστερός τροχός να κινείται με διαφορετική ταχύτητα σε σχέση με τον δεξί παρόλο που αμφότεροι έλαβαν το ίδιο σήμα. Ομοίως και μικρές διαφοροποιήσεις στους τροχούς μπορεί να είναι η αιτία που ενώ κάνουν την ίδια περιστροφή να διανύουν διαφορετική απόσταση[3].

Αυτά τα "προβλήματα" μπορούμε να τα αντιμετωπίσουμε με τη διαδικασία βαθμονόμησης των τροχών. Ένα βαθμονομημένο Duckiebot λαμβάνει υπόψη του τις παραπάνω διαφοροποιήσεις και στέλνει διαφορετικά σήματα στον αριστερό και στον δεξιό κινητήρα ώστε το Duckiebot να κινείται στην πράξη σε ευθεία γραμμή.

Οι σχέσεις ανάμεσα στην γραμμική και γωνιακή ταχύτητα του Duckiebot και των ταχυτήτων του αριστερού και του δεξιού κινητήρα δίνονται από τις παρακάτω σχέσεις

$$u_{right} = (g + r) \cdot (u + \frac{1}{2} wl) \qquad \text{και} \qquad u_{left} = (g - r) \cdot (u - \frac{1}{2} wl)$$

όπου u_{right} και u_{left} είναι οι ταχύτητες των δύο κινητήρων, το g ονομάζεται gain το r ονομάζεται trim, u και w είναι επιθυμητή γραμμική και γωνιακή ταχύτητα του Duckiebot και l είναι η απόσταση μεταξύ των τροχών.

Η παράμετρος gain ελέγχει τη μέγιστη ταχύτητα του Duckiebot. Έτσι όταν g >1.0 το όχημα πηγαίνει πιο γρήγορα με την ίδια εντολή ταχύτητας ενώ αν g < 1.0 κινείται πιο αργά.

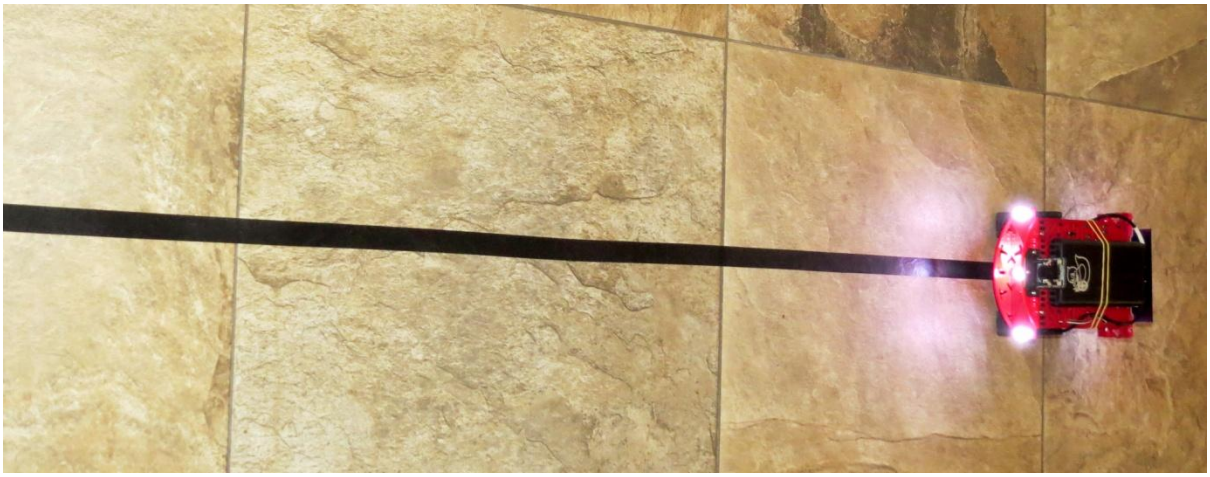
Η παράμετρος trim ελέγχει την ισορροπία μεταξύ των δύο κινητήρων. Έτσι όταν r > 0 ο δεξιός τροχός θα περιστραφεί ελαφρώς περισσότερο σε σχέση με τον αριστερό με την ίδια εντολή ταχύτητας ενώ όταν r < 0 θα συμβεί το αντίστροφο δηλαδή ο αριστερός τροχός θα γυρίσει λίγο περισσότερο από τον δεξιό.

Με αυτές ακριβώς τις παραμέτρους θα "παίξουμε" κατά τη διαδικασία της βαθμονόμησης ώστε το Duckiebot να έχει την επιθυμητή κινηματική συμπεριφορά στις εντολές που δέχεται.

Για να πραγματοποιηθεί η διαδικασία βαθμονόμησης των τροχών απαραίτητη προϋπόθεση είναι το Duckiebot να μπορεί να κινηθεί "χειροκίνητα" κατευθυνόμενο δηλαδή από τον χρήστη - χειριστή είτε μέσω joystick είτε από το πληκτρολόγιο του φορητού υπολογιστή.

9.3.1 Βαθμονομώντας την παράμετρο trim

Πριν την έναρξη της διαδικασίας [41] θα πρέπει να κολλήσουμε σε μία επίπεδη επιφάνεια μία ταινία μήκους τουλάχιστον δύο μέτρων (2m) και να τοποθετήσουμε το Duckiebot στην αρχή της ταινίας προσέχοντας να είναι στο κέντρο της ταινίας όπως φαίνεται στην παρακάτω εικόνα:

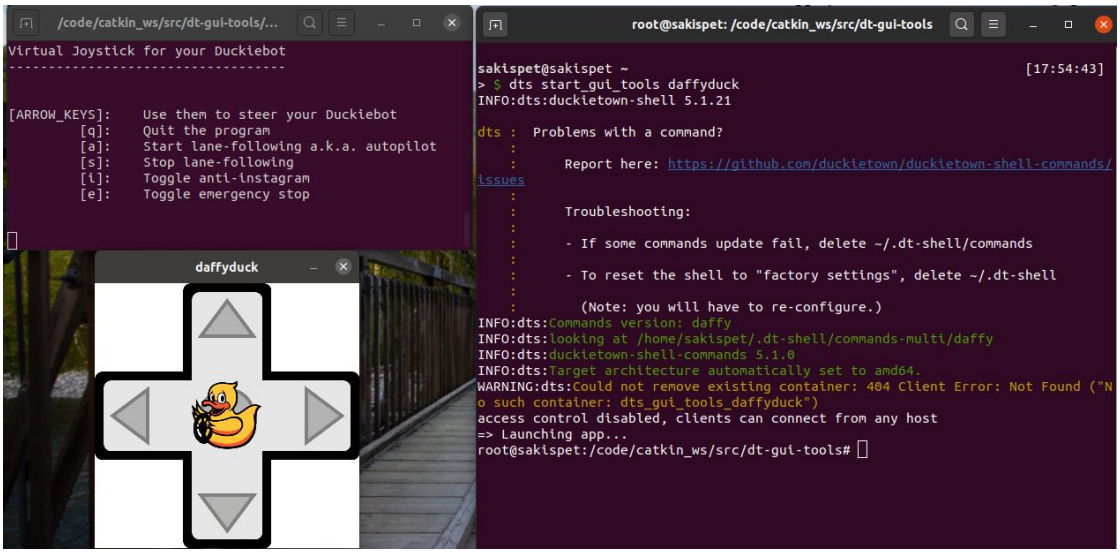


Εικόνα 147 : Τοποθέτηση Duckiebot στο κέντρο της γραμμής.

Από τον φορητό υπολογιστή ανοίγουμε σε ένα παράθυρο γραμμής εντολών την γραφική διεπαφή κίνησης του Duckiebot και σε ένα άλλο ενεργοποιούμε το βασικό container start_gui_tools του duckietown shell για μπορούμε να αλλάξουμε την προκαθορισμένη τιμή

(0.00) της παραμέτρου trim και να αποθηκεύσουμε έπειτα τις αλλαγές. Έτσι πληκτρολογούμε

$ dts start_gui_tools Duckiebot_Name



Εικόνα 148 : Εκκίνηση προσομοιωτή Joystick

Από το Virtual Joystick πατάμε το πάνω βέλος ώστε το Duckiebot να κινηθεί ευθεία εμπρός πάνω στην ταινία που έχουμε κολλήσει για δύο περίπου μέτρα παρατηρώντας αν παρεκκλίνει της πορείας του είτε προς τα αριστερά είτε προς τα δεξιά όπως στην περίπτωση μας. Έπειτα μετράμε το μήκος της απόκλισης του κέντρου του Duckiebot από την ταινία. Αν η απόσταση αυτή είναι μικρότερη από δέκα εκατοστά (10cm) σταματάμε τη διαδικασία βαθμονόμησης της παραμέτρου trim καθώς μία τέτοια απόκλιση είναι εντός των ορίων ανοχής του Duckietown. Στην περίπτωση όμως που απόκλιση υπερβαίνει τα 10 εκατοστά θα πρέπει αν προβούμε σε διορθωτικές ενέργειες αναλόγως της κατεύθυνσης (δεξιά ή αριστερά) και του μεγέθους της απόκλισης.

Εικόνα 149: Μέτρηση απόκλισης μετά από πορεία 2 μέτρων

Στην δική μας περίπτωση όπως φαίνεται και στην παραπάνω εικόνα το Duckiebot είχε μία απόκλιση 20 περίπου εκατοστών δεξιότερα της ταινίας. Συνεπώς έπρεπε να δώσουμε μία θετική δεκαδική τιμή στην παράμετρο trim ώστε ο δεξιός κινητήρας να περιστρέφεται λίγο πιο γρήγορα. Αυτό επιτυγχάνεται πληκτρολογώντας στο παράθυρο της γραμμής εντολών που έχει ενεργοποιηθεί το container start_gui_tools

$ rosparam set /Duckiebot_Name/Kimematics_node/trim 0.1

Αν είχαμε διαπιστώσει απόκλιση προς τα αριστερά θα έπρεπε να δώσουμε αρνητική τιμή στην παράμετρο trim.

Επαναλαμβάνουμε τη διαδικασία κίνησης του Duckiebot πάνω στην ταινία μέχρι να έχουμε την ελάχιστη δυνατή απόκλιση από αυτήν στο τέλος της διαδρομής.

Εικόνα 150 : Απόκλιση Duckiebot από τη γραμμή μετά τη διόρθωση της παραμέτρου trim

9.3. 2 Βαθμονομώντας την παράμετρο gain

Η παράμετρος gain που αφορά την ταχύτητα κίνησης του Duckiebot είναι εξ' ορισμού 1.00 Αν θέλουμε να μεταβάλλουμε (να μειώσουμε ή να αυξήσουμε) την ταχύτητα κίνησης του Duckiebot όταν πιέζουμε κάποιο βέλος του Virtual Joystick μπορούμε να αλλάξουμε την τιμή αυτής της παραμέτρου. Αν την μειώσουμε κάτω από τη μονάδα το όχημα θα κινείται με πιο αργή ταχύτητα ενώ αν την αυξήσουμε το όχημα θα κινείται γρηγορότερα. Για να αλλάξουμε την τιμή της πληκτρολογούμε

$ rosparam set /Duckiebot_Name/kinematics_node/gain gain_value

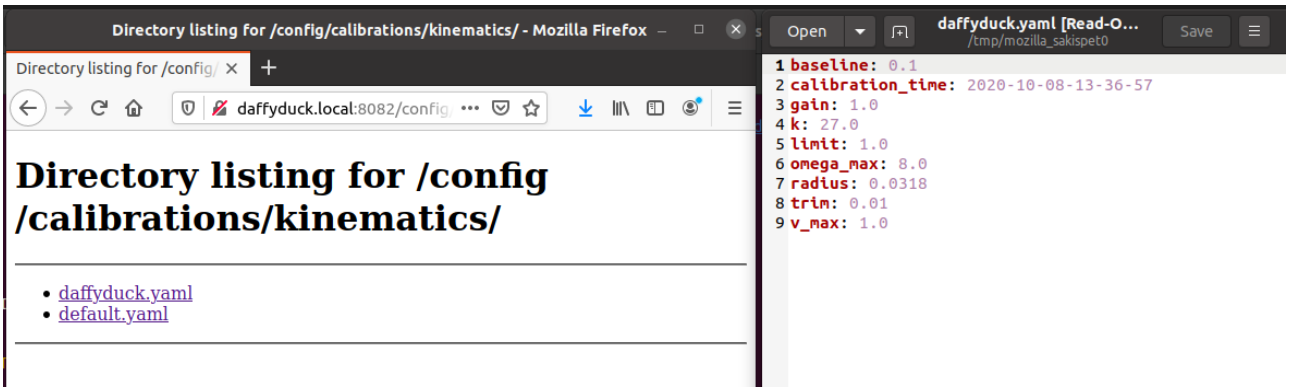
Αφού βαθμονομήσαμε τις δύο παραμέτρους trim και gain πρέπει να αποθηκεύσουμε τις παραπάνω αλλαγές που κάναμε σε ένα αρχείο πληκτρολογώντας

$ rosservice call /Duckiebot_Name/kinematics_node/save_calibration

Η εκτέλεση της παραπάνω εντολής θα έχει ως αποτέλεσμα τη δημιουργία ενός αρχείου με επέκταση yaml και όνομα το αναγνωστικό του Duckiebot στο δίκτυο. Το αρχείο αυτό

βρίσκεται στον κατάλογο /config/calibrations/kinematics στο Duckiebot και μπορούμε να το προσπελάσουμε μέσω του web interface που μας παρέχει το Portainer γράφοντας στη γραμμή διεύθυνσης ενός browser

http://Duckiebot_Name.local:8082



Εικόνα 151 : Αρχείο περιγραφής βαθμονόμησης τροχών

Κεφάλαιο 10

Lane following

Η πιο "απλή" αλλά και η πιο βασική αυτόνομη συμπεριφορά που μπορεί να εμφανίσει το Duckiebot είναι να ακολουθεί τη λωρίδα του δρόμου (lane following) βασιζόμενο στις προσλαμβάνουσες από την κάμερα του εικόνες. Απαραίτητη προϋπόθεση για τη σωστή λειτουργία του αλγορίθμου που υλοποιεί αυτή τη συμπεριφορά είναι τόσο η βαθμονόμηση της κάμερας (η εύρεση δηλαδή των ενδογενών και των εξωγενών παραμέτρων της κάμερας) όσο και η βαθμονόμηση των τροχών του Duckiebot. Πρόκειται ξεκάθαρα για μία εργασία υπολογιστικής όρασης αφού το Duckiebot πρέπει να δημιουργήσει μία σχέση μεταξύ της εικόνας που συνέλαβε η κάμερα και του εξωτερικού κόσμου. Ιδιαίτερα η εύρεση των σημείων της λωρίδας και της θέσης του Duckiebot σε σχέση με αυτά είναι ιδιαίτερα σημαντική για τις εντολές που θα δοθούν στους κινητήρες του προκειμένου να διατηρήσει μία πορεία εντός του οριοθετημένου από τις λωρίδες δρόμου.

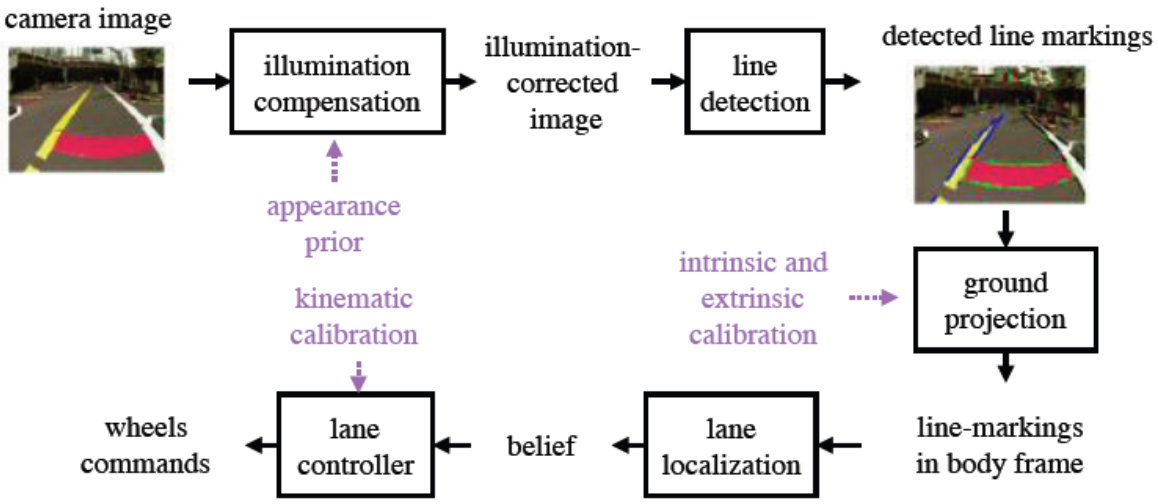
Επίσης μία ακόμα βασική προϋπόθεση για την αυτόνομη πλοήγηση του Duckiebot είναι ο σχεδιασμός σύμφωνα με τις προτεινόμενες προδιαγραφές της πόλης (Duckietown). Συγκεκριμένα είχαμε αναφέρει ότι μία διαμορφωμένη πόλη παρέχει δύο επίπεδα πληροφοριών στο Duckiebot. Το επίπεδο του δρόμου που είναι οριοθετημένο με ταινίες συγκεκριμένου χρώματος και πάχους και το επίπεδο των πινακίδων σήμανσης. Βέβαια για τη λειτουργία Lane Following το Duckiebot λαμβάνει πληροφορίες μόνο από επίπεδο του δρόμου το οποίο πρέπει να πληροί τις προδιαγραφές όπως αυτές περιγράφονται στο Κεφάλαιο 2 προκειμένου το ακριβές χρώμα της λωρίδας και η τοποθέτηση της να μπορούν να αποτελέσουν τη βάση για δημιουργία αντίληψης από το ρομποτικό όχημα.

Η συμπεριφορά αυτή της ακολούθησης της λωρίδας του δρόμου (Lane Following) βασίζεται σε μία ακολουθία βημάτων υπολογιστικής όρασης (computer vision pipeline) που περιλαμβάνει τα έξης στάδια [42]:

- Απόσβεση (αποζημίωση) μεταβλητότητας φωτισμού που αναφέρεται και ως Anti-Instagram στο project.
- Ανίχνευση των σημείων της οδικής χάραξης
- Επαναπροβολή από τον χώρο της εικόνας στο παγκόσμιο πλαίσιο αναφοράς με βάση την ενδογενή και την εξωγενή βαθμονόμηση της κάμερας.
- Εντοπισμός της λωρίδας με χρήση μη παραμετρικού φίλτρου Bayes
- Ελεγκτής λωρίδας (Lane controller)

Στην συνέχεια θα περιγράψουμε αυτά στάδια της υπολογιστικής όρασης για γίνει αντιληπτός ο τρόπος με τον οποίο το Duckiebot κατασκευάσει χαρακτηριστικά αντίληψης και ενεργεί με βάση αυτά ώστε να υπηρετήσει τον σκοπό του που είναι η κίνηση του εντός του χαραγμένου οδικού δικτύου της πόλης. Τα βήματα αυτά απεικονίζονται και στην παρακάτω εικόνα. Στο σημείο αυτό αξίζει να σημειώσουμε ότι το lane following pipeline εκτελείται στον

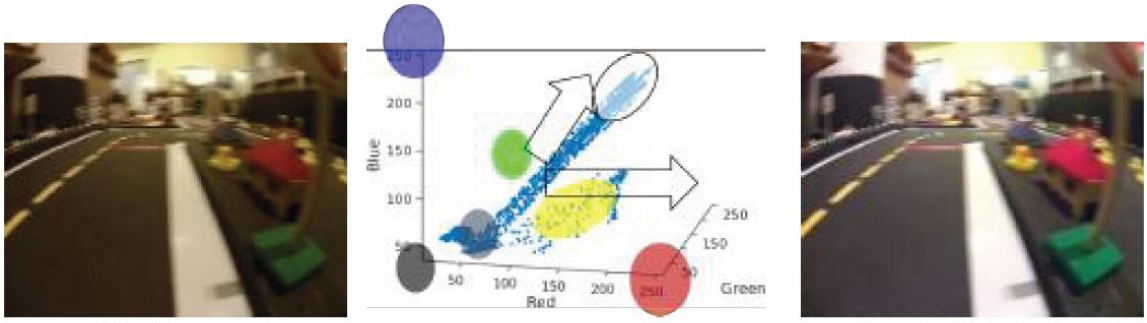
επεξεργαστή του Raspberry Pi που φέρει το Duckiebot με ταχύτητα 10 Hz, οι εικόνες έχουν ανάλυση 320x240 pixels χρονική υστέρηση 110ms.



Εικόνα 152: Ο αγωγός υλοποίησης του Lane Following. Το μωβ κείμενο αντιστοιχεί σε προηγούμενες πληροφορίες.

10.1 Αντιστάθμιση φωτισμού

Η χρησιμοποίηση της κάμερας ως βασικού αισθητήρα για τη δημιουργία αντίληψης έχει γενικά πολλές προκλήσεις. Μία από αυτές είναι η μεταβλητότητα του φωτισμού. Σε αυτό project δεν υπάρχουν συγκεκριμένες απαιτήσεις για τις συνθήκες του φωτισμού που πρέπει να πληρούνται γεγονός που σημαίνει ότι οι αλγόριθμοι που εκτελούνται θα πρέπει να είναι ανεκτικοί και μην επηρεάζονται απέναντι στις αλλαγές των συνθηκών του φωτισμού που ενδέχεται να συμβαίνουν.



Εικόνα 153 : Η εικόνα της κάμερα πριν (α) και μετά (β) την αντιστάθμιση φωτισμού

Για να επιτευχθεί αυτή η ανεκτικότητα γίνεται χρήση του αλγορίθμου ομαδοποίησης k-means (k-means clustering algorithm) πάνω σε ένα δείγμα των ληφθέντων από τον αισθητήρα εικονοστοιχείων προκειμένου να ανιχνεύσουμε τις κύριες συστάδες του δρόμου και να τις αντιστοιχίσουμε με τις αναμενόμενες χρωματικές ομάδας του κόκκινου, του κίτρινου, του λευκού και του γκρι έτσι όπως έχουν διαμορφωθεί από πρότερες

προσλαμβάνουσες εικόνες. Έπειτα εφαρμόζουμε έναν χρωματικό RGB μετασχηματισμό ανάμεσα στις εντοπισμένες συστάδες και τις χρωματικά ισορροπημένες εκδοχές τους. Αυτή η κανονικοποίηση εκφράζει την προηγούμενη πεποίθηση μας για τις συνθήκες φωτισμού και τα χρώματα σήμανσης του δρόμου. Οι μετασχηματισμοί επιτρέπεται να γίνονται σε ξεχωριστά χρωματικά κανάλια με βάση την παρακάτω σχέση:

$$I_{obs}^{(i)} = a_i I_{orig}^{(i)} + b_i + n, \quad i \in \{R, G, B\} \qquad (Σχέση\ 1)$$

όπου n είναι ο Γκουσιανός μετρήσιμος θόρυβος και για την κανονικοποίηση υποθέτουμε ότι οι παράμετροι ai και bi εκφράζουν τις προγενέστερες τιμές του θορύβου. Το αποτέλεσμα αυτής της σχέσης είναι 6x6 γραμμικό σύστημα εξισώσεων:

$$\begin{pmatrix} A_{data} \\ A_{reg} \end{pmatrix} p_I = \begin{pmatrix} b_{data} \\ b_{reg} \end{pmatrix} \quad (Σχέση\ 2)$$

όπου A_{data}, b_{data} είναι δεδομένα που προκύπτουν από το μοντέλο παρατήρησης της σχέσης 1 και A_{reg}, b_{reg} είναι τα αποτελέσματα της κανονικοποίησης. Το διάνυσμα p_I το διάνυσμα των παραμέτρων φωτισμού. Το υπολειπόμενο τετραγωνικό σφάλμα μας δίνει μία εκτίμηση της ποιότητας προσαρμογής η οποία επιτρέπει στο σύστημα να ανιχνεύσει την αστοχία.

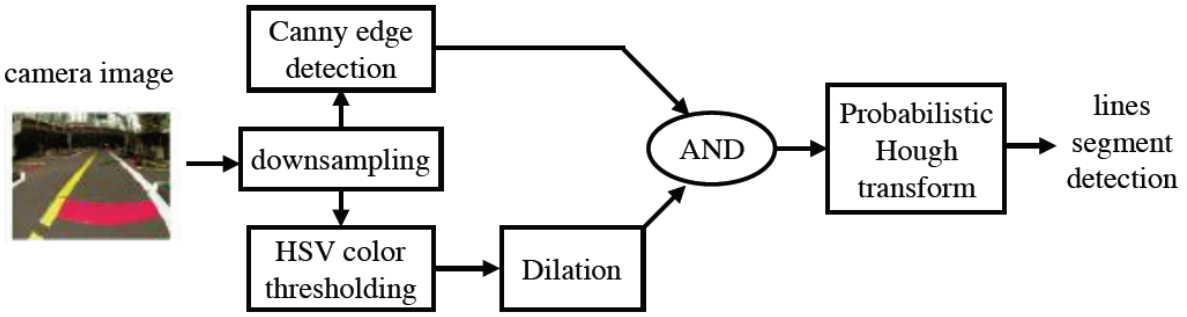
10.2 Ανίχνευση οδικών σημάνσεων

Μετά την αντιστάθμιση φωτισμού, τα προσανατολισμένα τμήματα γραμμής κατά μήκος του περιγράμματος των συνόρων των σημείων της λωρίδας εξάγονται από έναν ανιχνευτή γραμμής. Η ληφθείσα εικόνα αρχικά υπόκειται ένα downsampling ώστε να μειωθεί ο αρχικός όγκος των δεδομένων της προκειμένου να ελαττωθεί το μετέπειτα υπολογιστικό κόστος από την εφαρμογή των φίλτρων σε αυτή. Έπειτα εφαρμόζονται παράλληλα δύο φίλτρα. Ένα φίλτρο ανίχνευσης άκρων Canny και ένα φίλτρο απόχρωσης, κορεσμού και τιμών χρωματικού χώρου προκειμένου να προσδιοριστεί το δεδομένο χρώμα.

Ο αλγόριθμος Canny, αν και είναι ένας σχετικά "παλιός" αλγόριθμος (αναπτύχθηκε το 1986 από τον JohnF. Canny) έχει γίνει ένας πρότυπος αλγόριθμος ανίχνευσης ακμών. Τα βασικά του βήματα είναι [43] :

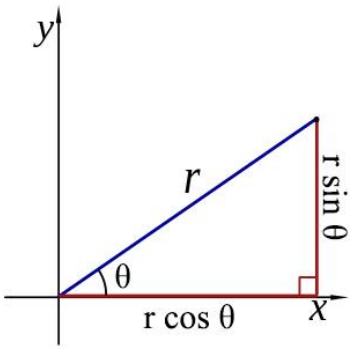
- Smoothing : Εξομάλυνση της εικόνας ώστε να μειωθεί ο θόρυβος
- Finding gradients: Οι ακμές γίνονται εντονότερες εκεί όπου η κλίση γίνεται μεγαλύτερη
- Non-maxima suppression: Μόνο τα μέγιστα μπορούν να σημειωθούν ως ακμές
- Double thresholding : Οι σημαντικές ακμές επιλέγονται με βάση ένα πάνω και ένα κάτω όριο
- Edge tracking by hysteresis: Οι τελικές ακμές καθορίζονται από την αφαίρεση όλων των ακμών που δε συνδέονται σε μια συγκεκριμένη ακμή (strong edge)

Εκτός από φίλτρο χρωματικού ορίου HSV (Hue-Saturation-Value colorspace) εφαρμόζεται και ένα φίλτρο διαστολής το οποίο διευρύνει τα όρια των εικονοστοιχείων, προσθέτει δηλαδή εικονοστοιχεία σε αυτά που ανήκουν εντός του ορίου HSV [42].



Εικόνα 154 : Ο αγωγός του αλγορίθμου εντοπισμού γραμμής

Τα αποτελέσματα των δύο φίλτρων περνάμε μέσα από μία πύλη AND για να αφαιρεθούν οι άκρες που δεν έχουν σωστό χρώμα. Τα μεμονωμένα τμήματα γραμμής εξάγονται χρησιμοποιώντας ένα πιθανοτικό μετασχηματισμό Hough. Η Θεωρητική βάση αυτού του μετασχηματισμού γραμμών [44] είναι πως κάθε σημείο-pixel σε μία δυαδική εικόνα μπορεί να είναι τμήμα κάποιας γραμμής. Μια γραμμή στο χώρο της εικόνας μπορεί να εκφραστεί με δύο μεταβλητές είτε στο Καρτεσιανό σύστημα αξόνων είτε στο Πολικό σύστημα αξόνων.



Εικόνα 155 : Αναπαράσταση γραμμής στο Πολικό σύστημα αξόνων

Ο μετασχηματισμός Hough αξιοποιεί το Πολικό σύστημα αξόνων και διατυπώνει την εξίσωση της γραμμής ως εξής:

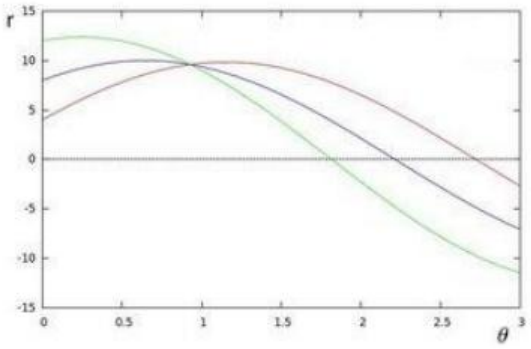
$$y = \left(-\frac{\cos\theta}{\sin\theta}\right)x + \left(\frac{r}{\sin\theta}\right), \quad r = x \cdot \cos\theta + y \cdot \sin\theta$$

Έτσι για κάθε σημείο (x_0,y_0) μπορούμε να ορίσουμε τις πιθανές γραμμές που διέρχονται από αυτό, όπου (r_0, θ) είναι κάθε γραμμή που διέρχεται από το σημείο (x_0,y_0):

$$r_0 = x_0 \cdot \cos\theta + y_0 \cdot \sin\theta$$

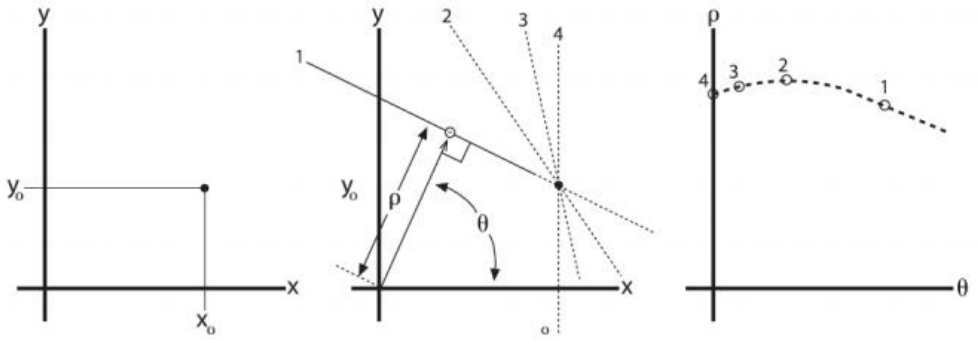
Για κάθε σημείο (x_0,y_0) εάν σχεδιαστούν οι γραμμές που μπορούν να διέρχονται από αυτό σχηματίζεται μια ημιτονοειδής κυματομορφή. Η ίδια διαδικασία γίνεται για κάθε σημείο της

εικόνας. Εάν οι καμπύλες δύο ή περισσότερων σημείων διασταυρώνονται στο πλάνο θ-r σημαίνει πως αυτά τα σημεία ανήκουν στην ίδια γραμμή.



Εικόνα 156: Τρεις καμπύλες που διασταυρώνονται σε ένα σημείο. Οι συντεταγμένες τους είναι οι παράμετροι (θ,r) της γραμμής (x_0,y_0), (x_1,y_1) και (x_2,y_2) στην οποία ανήκουν

Επομένως, η μέθοδος του Μετασχηματισμού Hough Lines ανιχνεύει γραμμές βρίσκοντας τον αριθμό των διασταυρώσεων μεταξύ καμπυλών. Όσες περισσότερες καμπύλες διασταυρώνονται τόσα περισσότερα σημεία έχει η γραμμή που αναπαρίσταται από αυτές τις διασταυρώσεις. Ταυτόχρονα, για να ανιχνευθεί μία γραμμή απαιτείται και μία τιμή κατωφλίου, η οποία θα καθορίζει τον ελάχιστο αριθμό διασταυρώσεων που απαιτούνται για να θεωρηθεί ότι σχηματίζεται μία γραμμή.

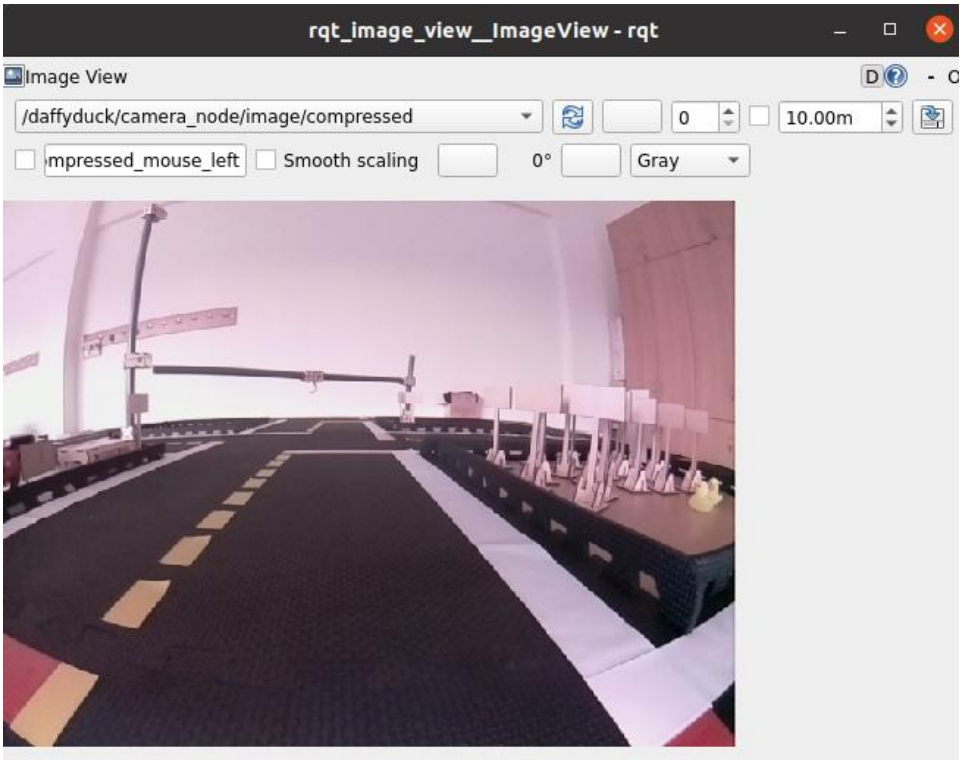


Εικόνα 157: Διαδικασία εύρεσης γραμμών

Ο Πιθανοτικός Μετασχηματισμός υποστηρίζει ότι μπορεί να επιτύχει ανάλογα αποτελέσματα με τον Κλασικό Μετασχηματισμό χρησιμοποιώντας μόνο ένα τμήμα των συνολικών σημείων της εικόνας. Το μέγεθος του δείγματος που θα χρησιμοποιηθεί είναι ανεξάρτητο από τα δεδομένα αλλά εξαρτάται από τη γνώση του ερευνητή για το περιεχόμενο των εικόνων. Όσο πιο άγνωστες είναι οι ιδιότητες των αντικειμένων της εικόνας τόσο μεγαλύτερο πρέπει να είναι το δείγμα των σημείων, ώστε να μην επηρεαστεί η ακρίβεια και η αποτελεσματικότητα του αλγορίθμου.

Ειδικότερα, ο Προοδευτικός Πιθανοτικός Μετασχηματισμός επιχειρεί να μειώσει ακόμη περισσότερο το χρόνο υπολογισμού εκμεταλλευόμενος τη διαφορά μεταξύ του δείγματος

σημείων που χρησιμοποιεί για την ανίχνευση γραμμών και του αριθμού των σημείων που είναι απαραίτητο να «υποστηρίζουν» την ύπαρξη μιας γραμμής.

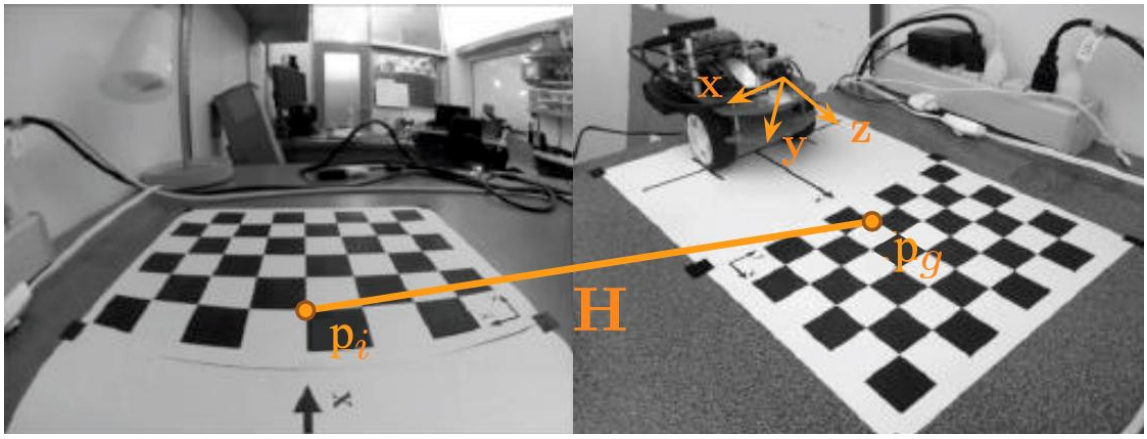


Εικόνα 158 : Ληφθείσα εικόνα από την κάμερα του Duckiebot κατά την κίνηση του σε λειτουργία Lane following

Εικόνα 159 : Οπτικοποιημένο αποτέλεσμα του κόμβου ανίχνευσης ακμών (line_detector_node) του ROS

10.3 Προβολή στο σύστημα αναφοράς του δρόμου

Το επόμενο βήμα είναι η μετατροπή των συντεταγμένων των εικονοστοιχείων της εικόνας (2D) που αντιπροσωπεύουν τμήματα γραμμής σε συντεταγμένες σημείων του πραγματικού τρισδιάστατου κόσμου. Χρησιμοποιώντας την επίπεδη εικόνα της προηγούμενης διαδικασίας είναι εφικτή η δημιουργία ενός χάρτη αντιστοιχίσεων 1 προς 1των σημείων της επίπεδης εικόνας και των σημείων του τρισδιάστατου κόσμου. Η διαδικασία της βαθμονόμησης της κάμερας είναι κομβική για τη φάση αυτή καθώς η εύρεση των ενδογενών παραμέτρων της κάμερας θα διασφαλίσει την μη παραμόρφωση των σημείων της εικόνας ενώ η εξωγενής βαθμονόμηση θα χρησιμοποιηθεί για να χαρτογραφήσει τα σημεία της εικόνας με σημεία του εδάφους. Συγκεκριμένα η εξωγενής βαθμονόμηση που είναι στην ουσία μία ομογραφία αντιστοιχεί σημεία της εικόνας p_i σε σημεία του τρισδιάστατού κόσμου $p_i \approx Hp_g$ όπου $H \in R^{3x3}$. Προσέχουμε ότι τα σημεία p_g ανήκουν στο έδαφος δηλαδή $p_g=(x,y,0)^T$. Η παρακάτω εικόνα απεικονίζει την εξωγενή βαθμονόμηση της κάμερας.



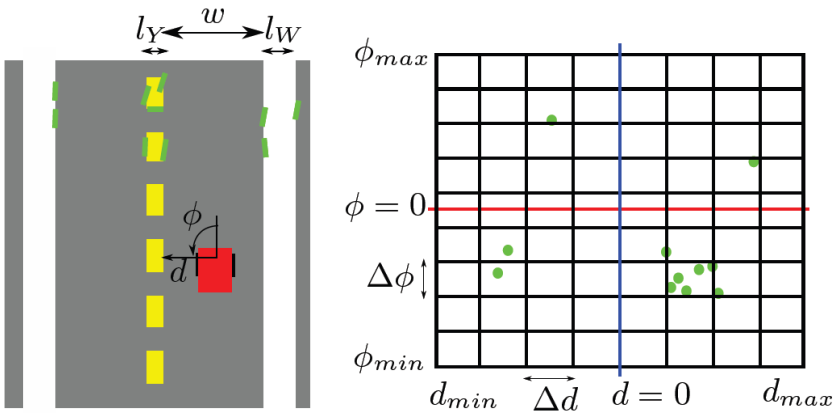
Εικόνα 160 : Εξωγενής βαθμονόμηση κάμερας

Όταν εκτιμηθεί ο πίνακας H είναι δυνατή η χαρτογράφηση από ένα σημείο της εικόνας p_i σε ένα σημείο της εικόνας p_g. Το πλαίσιο συντεταγμένων στη δεξιά εικόνα αντιπροσωπεύει το πλαίσιο συντεταγμένων της κάμερας του Duckiebot.

10.4 Σχετική εκτίμηση Duckiebot σε σχέση με την λωρίδα (Lane-Relative Estimation)

Για να είναι σε θέση το Duckiebot να ακολουθήσει τη λωρίδα του δρόμου θα πρέπει να έχουμε μία εκτίμηση της πλευρικής θέσης και του προσανατολισμού του Duckiebot σε σχέση με τη λωρίδα στο χρόνο t. Μια τυπική παραμετρική προσέγγιση στο πρόβλημα εκτίμησης (όπως για παράδειγμα ένα φίλτρο Kalman που χρησιμοποιεί μια υπόθεση Gauss) πιθανότατα θα αποτύγχανε λόγω της παρουσίας ενός δυνητικά υψηλού ποσοστού ακμών και της μη γραμμικότητας του μοντέλου. Έτσι για το συγκεκριμένο πρόβλημα εκτίμησης χρησιμοποιείται ένα μη γραμμικό, μη παραμετρικό φίλτρο ιστογράμματος όπου μία εκτίμηση της τρέχουσας γραμμικής και γωνιακής ταχύτητας του Duckiebot χρησιμοποιείται στο στάδιο πρόβλεψης.

Η θέση του ρομποτικού οχήματος στη λωρίδα στο χρόνο t αντιπροσωπεύεται [42] από την μειωμένης διάστασης σχέση $x_t \triangleq (d_t, \varphi_t)$, όπου $d_t \in [d_{min}, d_{max}]$ είναι η πλευρική μετατόπιση στη λωρίδα (με d=0 ορίζεται το κέντρο της λωρίδας) και $\varphi_t \in [\varphi_{min}, \varphi_{max}]$ είναι η γωνία σε σχέση με τον κεντρικό άξονα όπως φαίνεται στο παρακάτω σχήμα.



Εικόνα 161 : Lane Filtering.

Από τις προδιαγραφές γνωρίζουμε το πλάτος της λωρίδας (w) και τα πλάτη l_Y και l_W αντίστοιχα και μπορούμε να χρησιμοποιήσουμε αυτές τις πληροφορίες μαζί για να δημιουργήσουν μια μοναδική υπόθεση της πόζας από κάθε τμήμα που ανιχνεύεται από τον ανιχνευτή οδικής σήμανσης.

Σε κάθε βήμα ενημέρωσης κάθε τμήμα από τη λίστα των τμημάτων (πράσινες γραμμές αριστερά) θα "ψηφίσει" για μια δεδομένη πόζα (d,φ) που είναι οι πράσινες κουκκίδες δεξιά. Όλες οι "ψήφοι" χωρίζονται σε ένα ιστόγραμμα (το πλέγμα δεξιά) όπου ολόκληρο το ιστόγραμμα αντιπροσωπεύει την πιθανότητα κάθε πόζας. Όταν όλα τα τμήματα έχουν ψηφίσει το ιστόγραμμα κανονικοποιείται και χρησιμοποιείται για να πραγματοποιηθεί η ενημέρωση της μέτρησης. Στη συνέχεια δίνεται μια εκτίμηση της τρέχουσας πόζας (d,φ) από το φίλτρο ιστογράμματος Bayes.

Το φίλτρο Bayes είναι ένας αναδρομικός αλγόριθμος υπολογισμού πεποιθήσεων ο οποίος στη γενική του μορφή υπολογίζει την πεποίθηση bel(x_t) από την προηγούμενη κατάσταση και από δεδομένα μετρήσεων z_t και ελέγχου u_t. Τα βασικά βήματα του αλγορίθμου είναι δύο:

- Υπολογισμός της πρόβλεψης (prediction) από το ολοκλήρωμα που προκύπτει από τις κατανομές της x_{t-1} και της πιθανότητας ότι ο έλεγχος u_t επηρεάζει την μεταβολή από x_{t-1} σε x_t
- Υπολογισμός της ενημέρωσης της μέτρησης (measurement update) από τον πολλαπλασιασμό της πρόβλεψης με την πιθανότητα ότι η μέτρηση z_t έχει παρατηρηθεί.

10.5 Ελεγκτής Λωρίδας

Μόλις έχουμε μια εκτίμηση της θέσης και του προσανατολισμού του Duckiebot σε σχέση με τη λωρίδα τα χρησιμοποιούμε για να παράγουμε τα σήματα ελέγχου ώστε το Duckiebot να παραμείνει εντός της λωρίδας. Στο σημείο αυτό να σημειώσουμε ότι έχουμε σχεδιάσει τις καμπύλες του δρόμου με τέτοιο τρόπο ώστε το αναλογικό σφάλμα που προκύπτει καθώς το Duckiebot στρίβει να είναι αρκετά μικρό και να εξακολουθεί να βρίσκεται μέσα στη "λεκάνη" σύγκλισης του γραμμικού ελεγκτή παρακολούθησης γεγονός που μετριάζει την ανάγκη για κάποιου είδους παραμετροποίησης της τροχιάς αναφοράς.

Στην πράξη ένας PD ελεγκτής (Proportional - derivative controller) χρησιμοποιείται για τη δημιουργία του σήματος ελέγχου που υπολογίζεται με βάση τη σχέση: $\omega(t)=k_d d(t)+k_\varphi \varphi(t)$, όπου ο k_d και k_φ είναι οι παράμετροι κέρδους (gain parameters). Ο κινηματικός κόμβος του ROS εφαρμόζοντας το αντίστροφο κινηματικό θα υπολογίσει την ταχύτητα κάθε τροχού βασιζόμενος στο ω(t) και την σταθερή γραμμική ταχύτητα u, δίνοντας στο ρομποτικό όχημα εντολή (u, ω_t).

10.6 Lane following στην πράξη

Στο εργαστήριο ερευνών του τμήματος Μηχανικών Πληροφορικής, Υπολογιστών και Τηλεπικοινωνιών του Διεθνούς Πανεπιστήμιου της Ελλάδος που εδρεύει στις Σέρρες και τα τέσσερα Duckiebots πραγματοποίησαν με επιτυχία τη λειτουργία lane following στην πόλη που κατασκευάσαμε. Η διαδικασία είναι απλή, εφόσον βέβαια έχει προηγηθεί η βαθμονόμηση της κάμερας και των τροχών και έχει ως εξής:

Αρχικά πρέπει αν βεβαιωθούμε ότι στα Duckiebots εκτελούνται τα containers που περιλαμβάνουν τους βασικούς κόμβους του Duckietown του ROS δηλαδή τα:

- dt-duckiebot,
- dt-car-interface
- και dt-core.

Αυτό εύκολα μπορούμε να διαπιστώσουμε μέσα από το web intreface του Portainer. Αν για κάποιο λόγο δεν έχουν ξεκινήσει να εκτελούνται θα πρέπει να τα εκκινήσουμε εμείς "χειροκίνητα" πληκτρολογώντας :

$ dts duckiebot demo --demo_name duckiebot-interface --duckiebot_name Duckiebot_Name --package_name duckiebot_interface --image duckietown/dt-duckiebot-interface:daffy-arm32v7

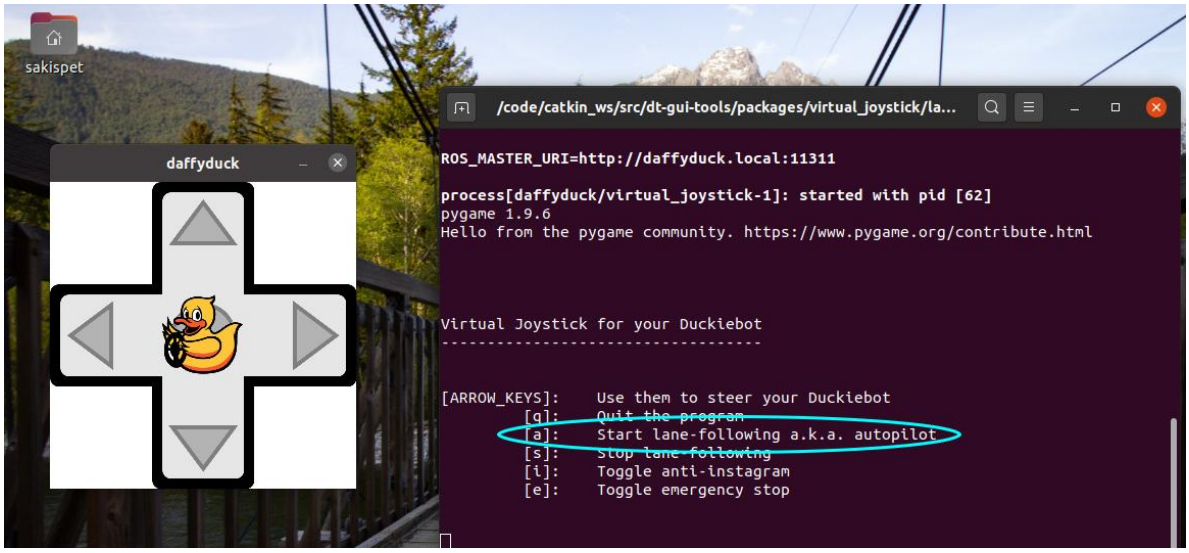
$ dts duckiebot demo --demo_name car-interface --duckiebot_name Duckiebot_name --package_name car_interface --image duckietown/dt-car-interface:daffy-arm32v7

Επόμενα βήμα είναι να εκκινήσουμε το pipeline που υλοποιεί τη λειτουργία lane following πληκτρολογώντας :

$dts duckiebot demo --demo_name lane_following --duckiebot_name Duckiebot_Name --package_name duckietown_demos

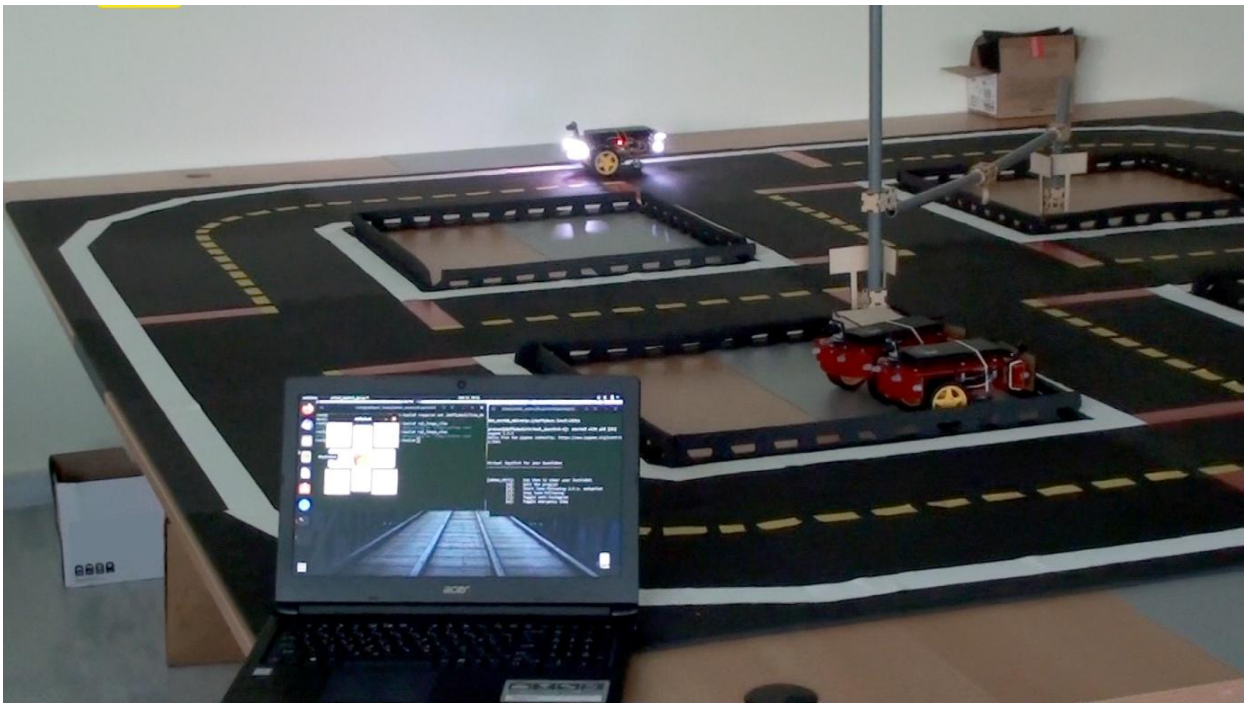
Τέλος εκκινούμε τον προσομοιωτή joystick μέσα από το περιβάλλον του οποίου θα θέσουμε το Duckiebot σε κατάσταση lane following

$ dts duckiebot keyboard_control Duckiebot_Name



Εικόνα 162 : Περιβάλλον προσομοιωτή Joystick

Επιλέγουμε τη λειτουργία Start kane-following a.k.a autopilot πιέζοντας το πλήκτρο [a].



Εικόνα 163 : Στιγμιότυπο από τη λειτουργία Lane Following του Duckiebot daffyduck

Για εξέλθουμε από αυτή τη κατάσταση λειτουργίας πιέζουμε το πλήκτρο [s].

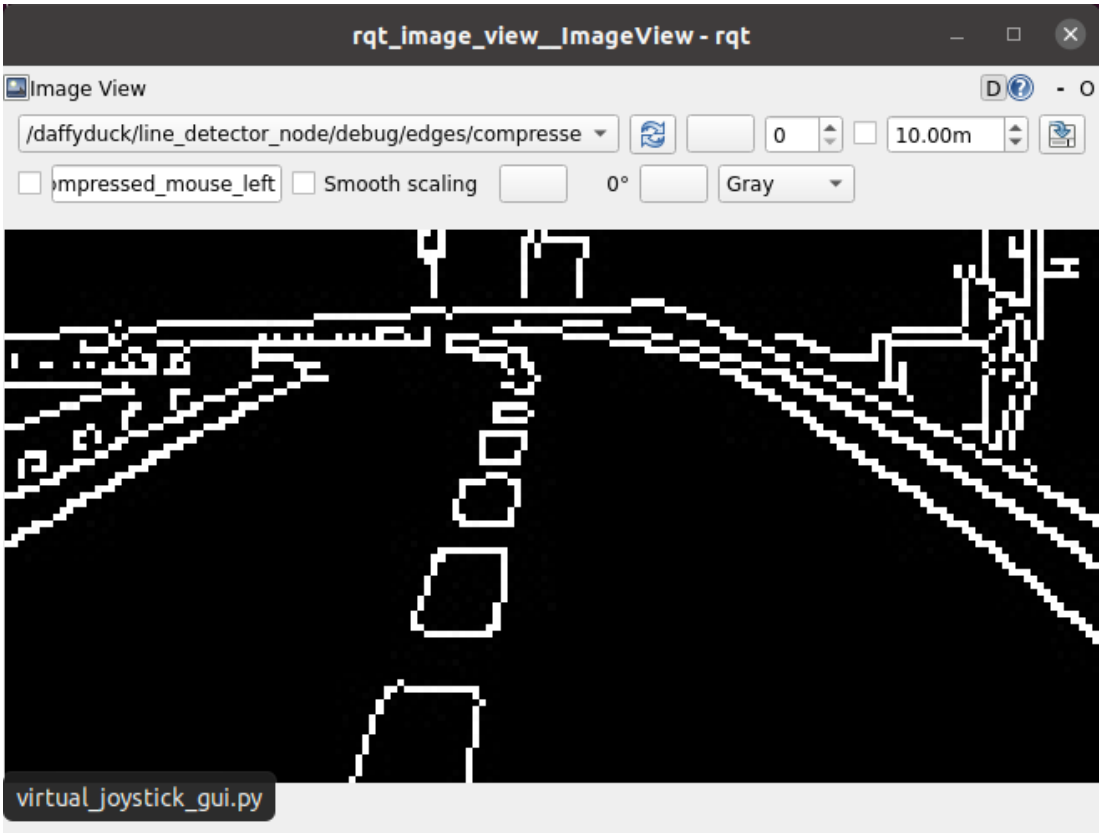
Όσο το Duckiebot βρίσκεται σε λειτουργία lane following υπάρχει η δυνατότητα οπτικοποίησης των ανιχνευμένων τμημάτων των γραμμών σε πραγματικό χρόνο. Αυτό επιτυγχάνεται με την δημοσίευση των topics του κόμβου line_detector_node του ROS. Αρχικά ανοίγουμε το φλοιό επικοινωνίας με το ROS πληκτρολογώντας:

$ dts start_gui_tools Duckiebot_Name

Έπειτα θέτουμε την παράμετρο verbose σε true έτσι ώστε ο κόμβος ανίχνευσης γραμμών να αρχίσει να δημοσιεύει εικόνες με ανιχνευμένες γραμμές

$ rosparam set /Duckiebot_Name/line_detector_node/verbose true

Τέλος ανοίγουμε το περιβάλλον rqt_image_view και επιλέγουμε τον κόμβο line_detector_node για να δούμε τις εικόνες που δημοσιεύει.



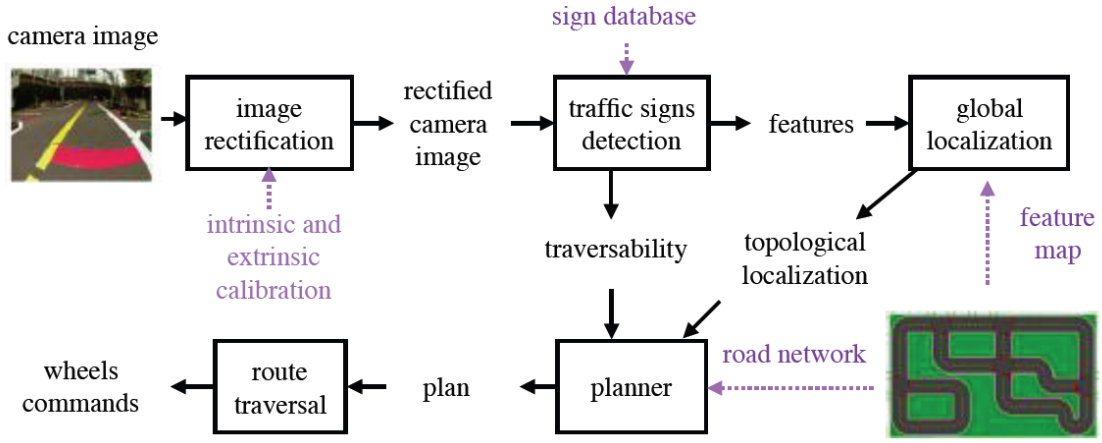
Εικόνα 164 : Εικόνες με ανιχνευμένες γραμμές που δημοσιεύει ο κόμβος lin_detector_node

Κεφάλαιο 11

Προηγμένες λειτουργίες Duckiebot - Επεκτάσεις Duckietown

11.1 Αυτόνομη πλοήγηση

Ο βασικός υπολογιστικός αγωγός που υλοποιεί τη συμπεριφορά lane following λειτουργεί ως εμφωλευμένος βρόχος που επιτρέπει πιο πολύπλοκες και προηγμένες συμπεριφορές όπως αυτή της αυτόνομης πλοήγησης [42].



Εικόνα 165: Πλοήγηση βάσει χάρτη και σχεδιασμό διαδρομών

11.1.1 Επίπεδο σημάτων

Όπως έχουμε αναφέρει η πόλη διαθέτει δύο επίπεδα πληροφοριών στο Duckiebot. Το πρώτο επίπεδο είναι το χαραγμένο οδικό δίκτυο και το δεύτερο επίπεδο είναι οι πινακίδες σήμανσης. Οι πινακίδες σήμανσης αφορούν δύο διαφορετικές κατηγορίες :

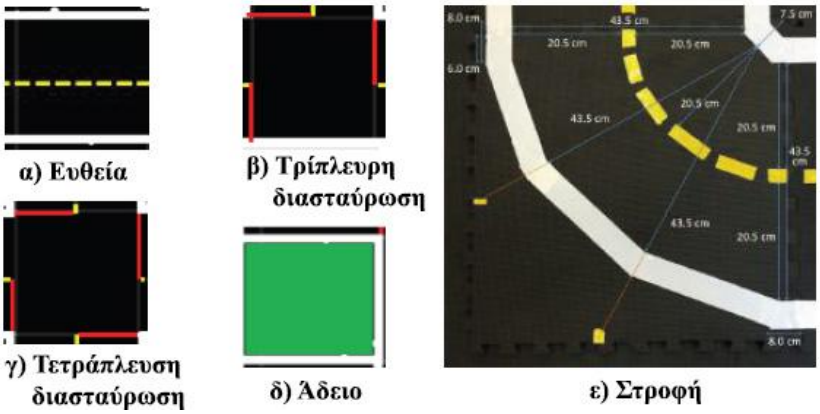
- Πινακίδες σημάτων οδικής κυκλοφορίας
- Πινακίδες ονομασίας οδών

Η πρώτη κατηγορία αφορά όπως είδαμε και στην διαμόρφωση της πόλης τα σήματα κυκλοφορίας που συναντούμε σχεδόν και σε ένα πραγματικό οδικό δίκτυο, όπως υποχρεωτικής στάσης (stop), είδος διασταύρωσης (με σηματοδότες ή όχι), και άλλες σημαντικές πληροφορίες. Αυτές οι πινακίδες περιέχουν όλες τις απαραίτητες πληροφορίες για την υλοποίηση των λειτουργιών του εντοπισμού και της πλοήγησης. Κάθε σύμβολο είναι εξοπλισμένο με ένα AprilTag το οποίο επιτρέπει την ανάπτυξη αλγορίθμων που είναι σε θέση να ανιχνεύουν αυτά τα "σημάδια" που μοιάζουν με τον κώδικα QR. Η τοποθέτηση αυτών των πινακίδων περιορίζεται μόνο σε οκτώ συγκεκριμένες πόζες πάνω σε ένα πλακίδιο του δαπέδου προκειμένου :

a) να βρίσκονται μέσα στο οπτικό πεδίο της κάμερας του Duckiebot
b) να είναι εφικτή η αυτόματη δημιουργία ενός χάρτη μετρικών χαρακτηριστικών.

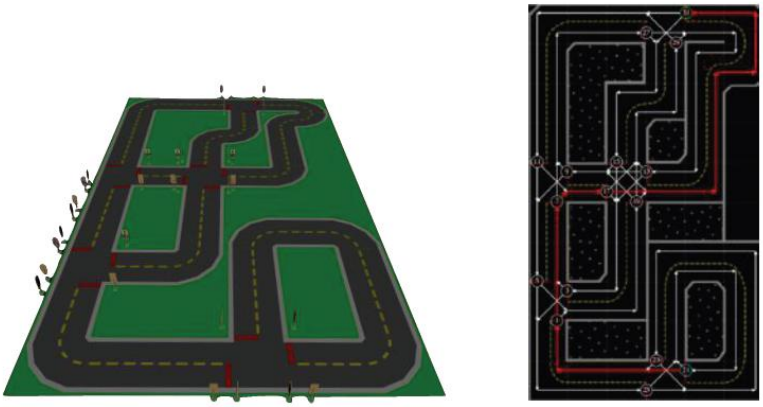
11.1.2 Αναπαράσταση του χάρτη

Δεδομένου ότι οι πόλεις (Duckietowns) αποτελούνται από τετράγωνα πλακίδια όπως αυτά της παρακάτω εικόνας ένας χάρτης είναι δυνατό να προσδιοριστεί από έναν δισδιάστατο πίνακα που κάθε θέση του μπορεί να αναπαριστά ένα τέτοιο πλακίδιο.



Εικόνα 166 : Τύποι πλακιδίων

Επιπρόσθετα μπορούμε να καθορίσουμε την παρουσία, τον τύπο και τη θέση κάθε πινακίδας πάνω σε αυτά τα πλακίδια. Αυτός ο πίνακας μπορεί να χρησιμοποιηθεί για να παράγει αυτόματα δύο διαφορετικές αναπαραστάσεις χαρτών. Ένα χάρτη μετρικών χαρακτηριστικών που χρησιμοποιείται για τη λειτουργία του εντοπισμού και ένα γράφημα τοπολογικού δικτύου που χρησιμοποιείται για το σχεδιασμό της διαδρομής.



Εικόνα 167: Δύο διαφορετικές αναπαραστάσεις χαρτών. Αριστερά: ο χάρτης μετρικών χαρακτηριστικών. Δεξιά: γράφημα τοπολογικού δικτύου

11.1.3 Παγκόσμιος εντοπισμός

Ο "παγκόσμιος" εντοπισμός συνίσταται στην εκτίμηση της θέσης του Duckiebot σε σχέση με το "παγκόσμιο" πλαίσιο αναφοράς, δηλαδή να είναι μπορεί να εκτιμήσει τη θέση και τον προσανατολισμό του στον "κόσμο" στον οποίο κινείται. Η γνώση αυτή είναι απαραίτητη για άλλες βασικές λειτουργίες όπως για παράδειγμα ο σχεδιασμός μιας πορείας προς ένα επιθυμητό προορισμό. Όπως συμβαίνει και με τους ανθρώπους, τα σήματα κυκλοφορίας και τα ονόματα των οδών παρέχουν χρήσιμες πληροφορίες για την επίτευξη του εντοπισμού. Στο

βασικό σχήμα εντοπισμού υποθέτουμε ότι μας παρέχεται ένας ακριβής χάρτης από το σύστημα αυτόματης δημιουργίας χαρτών.

Ωστόσο μία επέκταση της παραπάνω λειτουργίας θα μπορούσε να ήταν η ανακάλυψη του χάρτη της πόλης και ταυτόχρονός εντοπισμός του Duckiebot μέσα σε αυτόν (SLAM). Χρησιμοποιώντας τις ετικέτες Apriltags είμαστε σε θέση να λάβουμε μια σχετική μέτρηση της θέσης της κάμερας σε σχέση με την ετικέτα που εντοπίζεται. Δεδομένου ότι η πόζα της ετικέτας είναι γνωστή κάθε ανίχνευση πινακίδας μπορεί να μας παρέχει μία εκτίμηση της απόλυτης στάσης του Duckiebot. Αυτός ο βασικός αλγόριθμος εξασφαλίζει ακριβή εντοπισμό σε κάθε περίπτωση που ανιχνεύονται ετικέτες. Σε δοκιμές που έγινα το Duckiebot κατάφερε να εκτιμήσει τη θέση στα 2Hz με λάθη εντοπισμού της τάξεως του 0.1m όταν τουλάχιστον τρεις ετικέτες ήταν ορατές σε αυτό.

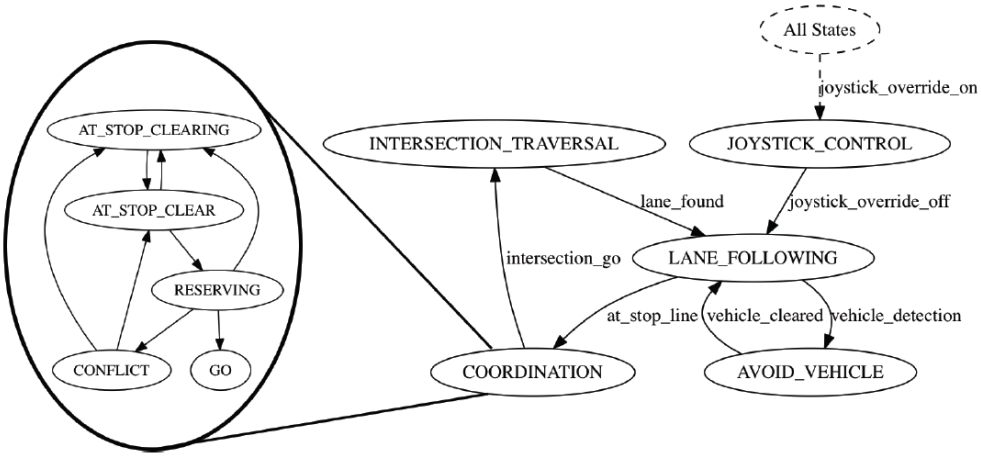
11.1.4 Σχεδιασμός

Η επίτευξη της πλοήγηση από το Duckiebot προϋποθέτει μία αντιστοίχιση του πίνακα των μετρικών χαρακτηριστικών όπως αυτός ορίστηκε από την παραπάνω διαδικασία με το τοπολογικό γράφημα του οδικού δικτύου στο οποίο γίνεται αναζήτηση για την εύρεση εφικτών διαδρομών.

Μόλις το γράφημα παραχθεί και οριστούν οι θέσεις εκκίνησης και προορισμού (στόχου) γίνεται εφαρμογή του αλγορίθμου αναζήτησης διαδρομής A* για να μας παράξει την βέλτιστη διαδρομή διάσχισης του χάρτη προκειμένου να φτάσουμε στον προορισμό μας. Μία ενδιαφέρουσα επέκταση αυτού του σταδίου θα ήταν ο σχεδιασμός των διαδρομών πολλών Duckiebots ταυτόχρονα με την προσομοίωση ενός συστήματος διαχείρισης στόλου.

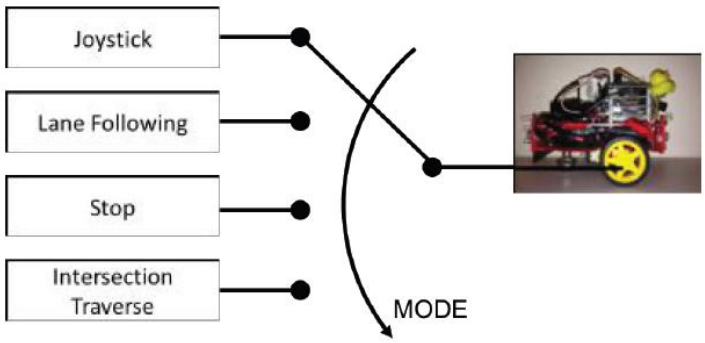
11.1.5 Μηχανή πεπερασμένης κατάστασης (Finite State Machine)

Σε λειτουργία επιπέδου μακροεντολων το Duckiebot ρυθμίζεται μέσω ενός μηχανήματος πεπερασμένης κατάστασης (FSM), μια απλοποιημένη έκδοση του οποίου φαίνεται στο παρακάτω σχήμα



Εικόνα 168: Μια απλοποιημένη εκδοχή του FSM που χρησιμοποιείται για τον έλεγχο του Duckiebot.

Οι μεταβάσεις από την μία κατάσταση λειτουργίας στην άλλη στο FSM δημιουργούνται από ασύγχρονα συμβάντα που βασίζονται στην αντίληψη όπως για παράδειγμα η ανίχνευση γραμμής στάσης. Η λειτουργία έλεγχου του Duckiebot βασίζεται σε ένα υβριδικό σύστημα. Έχει αναπτυχθεί και είναι διαθέσιμοι μια σειρά ελεγκτών που αντιστοιχούν σε διαφορετικές ενέργειες - συμπεριφορές το FSM επιλέγει ποιος ενεργός ελεγκτής θα συνδεθεί με το πρόγραμμα οδήγησης των τροχών.

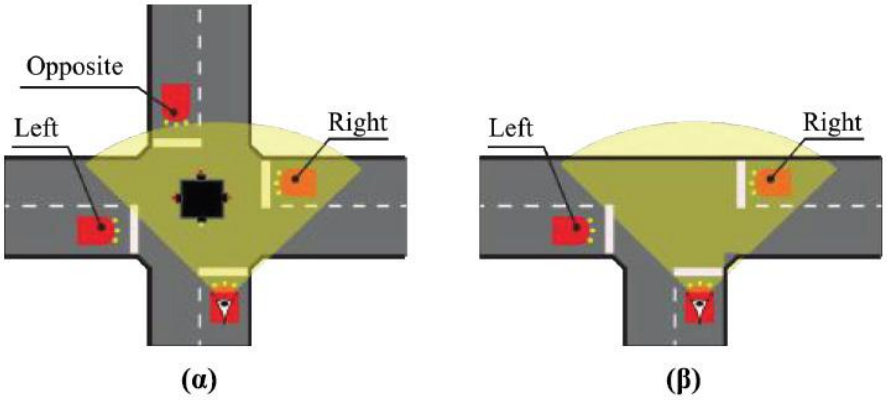


Εικόνα 169: Το υβριδικό σύστημα ελέγχου του Duckiebot

11.1.6 Εκτέλεση διαδρομής

Μετά τον σχεδιασμό της διαδρομής ακολουθεί η εκτέλεση αυτής της διαδρομής από το ρομποτικό όχημα. Που πρακτικά σημαίνει τη σωστή ακολουθία των στροφών σε κάθε διασταύρωση. Έτσι για παράδειγμα αν έχουμε μία ακολουθία εντολών [s, f, s, f, r, f, s, f] όπου :

- s : Πήγαινε ευθεία (go straight)
- f : ακολούθησε τη λωρίδα (follow lane)
- r : στρίψε δεξιά (turn right)
- l : στρίψε αριστερά (turn left)



Εικόνα 170 : Τύποι διασταυρώσεων. Η κίτρινη σκιασμένη περιοχή αντιστοιχεί στο οπτικό πεδίο της κάμερας του Duckiebot.

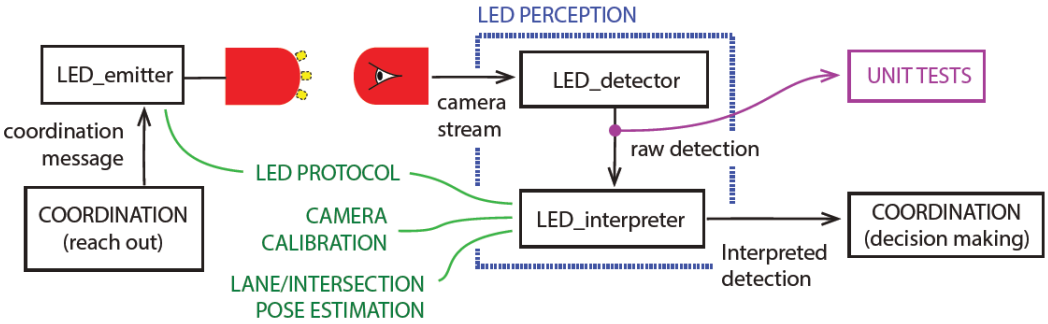
Η διέλευση οποιοσδήποτε διασταύρωσης επιτυγχάνεται με εκτελώντας την εξής ακολουθία καταστάσεων Lane_Following → Intersection_Traversal επαναληπτικά (σε βρόχο δηλαδή) ώσπου να φτάσουμε στον τελικό μας προορισμό (υποθέτοντας σε αυτή τη φάση ότι οι διασταυρώσεις θα είναι ελεύθερες για διέλευση, παρακάμπτοντας δηλαδή τη λειτουργία του συντονισμού (Coordination). Η μετάβαση από τη λειτουργία Lane_Following στη λειτουργία Intersection_Traversal γίνεται όταν το Duckiebot φτάνει σε μία κόκκινη γραμμή στάσης (stop line). Η αντίστροφη μετάβαση (από Intersection_Traversal σε Lane_Following) ενεργοποιείται από την αναφορά σύγκλισης της εκτίμησης λωρίδας όπως αυτή μετράται από ένα όριο εντροπίας.

11. 2 Αλληλεπίδραση Duckiebots

Οι πιο προηγμένες συμπεριφορές περιλαμβάνουν την αλληλεπίδραση τους με άλλα Duckiebots καθώς πλοηγούνται ταυτόχρονα στην πόλη. Σε αυτό το σενάριο τα Duckiebots πρέπει να συντονίζονται μεταξύ τους καθώς μοιράζονται δρόμους και διασταυρώσεις. Στη πόλη έχουμε δύο τύπους διασταυρώσεων [42]. Τις διασταυρώσεις που διαθέτουν φωτεινούς σηματοδότες και τις διασταυρώσεις με πινακίδες STOP. Οι διασταυρώσεις που διαθέτουν σηματοδότες είναι πιο απλή μορφή διευθέτησης καθώς ο συντονισμός βασίζεται στις φωτεινές ενδείξεις των σηματοδοτών. Όταν όπως αυτοί απουσιάζουν από τις διασταυρώσεις και έχουμε μόνο σήματα STOP τότε θα πρέπει να υπάρχει μία επικοινωνία των Duckiebots μεταξύ τους. Πρόκειται για μια αποκεντρωμένη επικοινωνία που βασίζεται στην αντίληψη που μπορούν και αναπτύσσουν τα ρομποτικά οχήματα και στηρίζεται στο πλήθος των αναμμένων LED που φέρουν τα οποία και εκδηλώνουν προσθέσεις κίνησης. Η κατάσταση περιπλέκεται από το γεγονός ότι ενδέχεται σε κάποιες διασταυρώσεις ένα Duckiebot να βρίσκεται εκτός του οπτικού πεδίου κάποιου άλλου Duckiebot όπως για παράδειγμα συμβαίνει στην παραπάνω εικόνα όπου το σταθμευμένο αριστερά Duckiebot είναι εκτός του οπτικού πεδίου του Duckiebot που είναι σταθμευμένο στην κάτω πλευρά της διασταύρωσης.

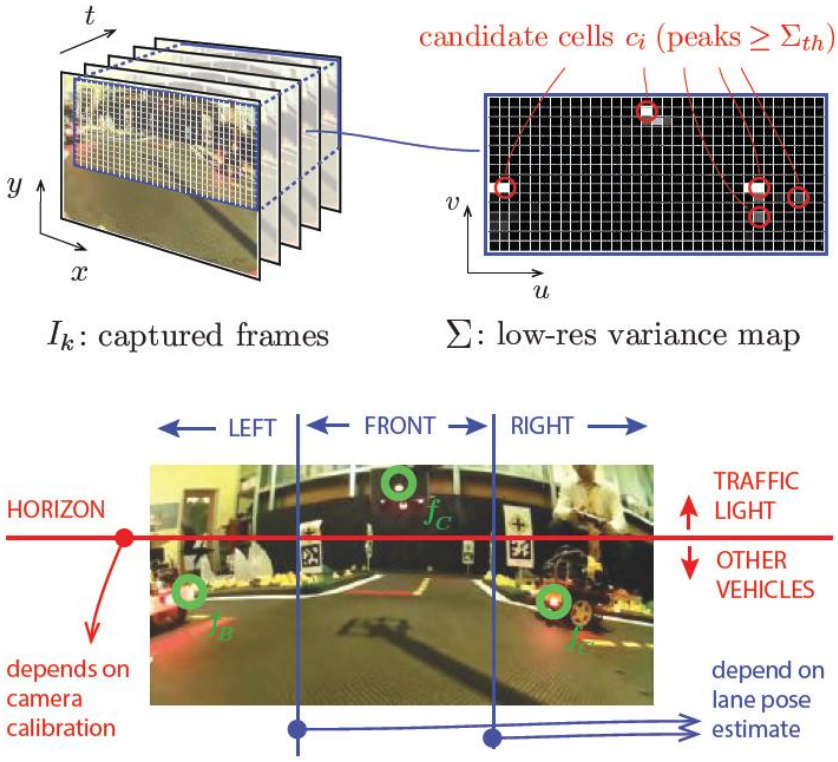
11.2. 1 Ανίχνευση και ερμηνεία των LED

Το σύστημα επικοινωνίας που βασίζεται στα LED διαθέτει δύο βασικά δομικά στοιχεία. Τον ανιχνευτή, ο οποίος συλλαμβάνει τη ροή εικόνων από την κάμερα και καθορίζει θέσεις και συχνότητες όλων των LED που υπάρχουν στην σκηνή και ο διερμηνέας ο οποίος λαμβάνει αυτές τις πληροφορίες και αντιστοιχεί κάθε ανίχνευση με ένα φυσικό αντικείμενο δηλαδή αναγνωρίζει αν η πηγή ανήκει σε όχημα ή σε φωτεινό σηματοδότη. Μια επισκόπηση αυτής της μορφής της επικοινωνίας φαίνεται στην παρακάτω εικόνα



Εικόνα 171 : Γράφημα επικοινωνίας βασισμένη στην ανίχνευση της συχνότητας των LED

Οι λεπτομέρειες του ανιχνευτή LED και του διερμηνέα απεικονίζονται στη παρακάτω εικόνα. Ο διερμηνέας αξιοποιεί την υπόθεση ότι τα LED που ανιχνεύονται πάνω από τον ορίζοντα ανήκουν σε σηματοδότες ενώ αυτά που ανιχνεύονται κάτω από τον ορίζοντα αντιστοιχούν σε Duckiebots.



Εικόνα 172 : Αλγόριθμος επικοινωνίας μέσω LED

Το πάνω μέρος της εικόνας περιγράφει τη λειτουργία του ανιχνευτή. Από την ανιχνευμένη ακολουθία των I_k εικόνων που συλλαμβάνει η κάμερα μια σχετική περιοχή (που οριοθετείται με μπλε χρώμα στην εικόνα) περικόπτεται και υπό-δειγματοληπτείται σύμφωνα με ένα πλέγμα χαμηλής ανάλυσης. Η χρονική διακύμανση της φωτεινότητας υπολογίζεται σε κάθε κελί του πλέγματος παράγοντας έναν χάρτη διακύμανσης (πάνω δεξιά στην εικόνα) του οποίου οι κορυφές καθορίζουν τα υποψήφια κελιά. Οι συχνότητες κορυφής της φωτεινότητας των υποψήφιων κελιών, που μας τις παρήγαγε ένας FFT αλγόριθμος (Fast

Fourier Transform), συγκρίνονται με γνωστές συχνότητες των σημάτων για αντιστοίχιση παράγοντας νέες ανιχνεύσεις. Στο κάτω μέρος της εικόνας απεικονίζει τη λειτουργία του διερμηνέα. Ο διερμηνέας παίρνει αυτές τις ανιχνεύσεις και εφαρμόζει όρια συντεταγμένων εικόνας για να προσδιορίσει την προέλευση των μηνυμάτων, αν προέρχονται δηλαδή από σηματοδότες ή άλλα Duckiebots.

11.2.2 Λειτουργία των σηματοδοτών

Οι φωτεινοί σηματοδότες όπως περιγράψαμε και στο κεφάλαιο 4 "εκπέμπουν" ένα σήμα διέλευσης ("go") προς μια κατεύθυνση και ένα σήμα στάσης ("stop") προς τις άλλες. Τα σήματα κωδικοποιούνται με συχνότητες που καθορίζουν τον ρυθμό που αναβοσβήνουν τα LEDs. Καθώς τα Duckiebots ερμηνεύουν αυτά τα σήματα αποφεύγονται συγκρούσεις και μπλοκαρίσματα στο κέντρο των διασταυρώσεων και διασφαλίζεται η ομαλή ροή της κυκλοφορίας. Όταν ένα Duckiebot φτάσει να γραμμή στάσης αρχίζει να ανιχνεύει τη συχνότητα με την οποία αναβοσβήνουν τα LED στο σηματοδότη και μόλις εντοπίσει τη συχνότητα που αντιστοιχεί στο σήμα "go" αρχίζει αν κινείται μέσω της διασταύρωσης.

11.2.3 Συμπεριφορές συντονισμού - Διασταυρώσεις με πινακίδες STOP

Στις διασταυρώσεις με πινακίδες STOP δίχως σηματοδότες τα Duckiebots πρέπει να επικοινωνούν μεταξύ τους προκειμένου να συντονίσουν τις κινήσεις τους και περάσουν από τις διασταυρώσεις. Αυτή η επικοινωνία γίνεται μέσω των LEDs που διαθέτουν. Γενικά το αποκεντρωμένο πρωτόκολλο "διαπραγμάτευσης" διέλευσης της διασταύρωσης ακολουθεί τους παρακάτω κανόνες:

a) Ένα Duckiebot παραχωρεί προτεραιότητα σε αυτό που βρίσκεται δεξιά του επειδή το τελευταίο (αυτό που βρίσκεται δεξιά) δεν μπορεί να ανιχνεύσει αυτό βρίσκεται αριστερά (εικόνα 170).
b) όταν δύο Duckiebots φτάνουν σε μία διασταύρωση το ένα απέναντι από το άλλο αυτό έφτασε νωρίτερα φεύγει πρώτο
c) αν με βάση την ακρίβεια ανίχνευσης φτάσουν ταυτόχρονα στην διασταύρωση, περιμένουν ένα τυχαίο χρονικό διάστημα πριν την επόμενη προσπάθεια του να διαπραγματευτούν με άλλα Duckiebots.

Η λειτουργία του συντονισμού των Duckiebots περιλαμβάνει μία σειρά καταστάσεων:

- AT STOP CLEARING : Το Duckiebot περιμένει έναν προκαθορισμένο χρόνο για να διασφαλίσει ότι η διασταύρωση είναι ελεύθερη για διέλευση.
- AT STOP CLEAR : Το Duckiebot περιμένει χωρίς άλλα Duckiebots στην διασταύρωση γεγονός που εγγυάται ότι είναι μπορεί να περάσει
- RESERVING : Το Duckiebot σηματοδοτεί μέσω των LED που διαθέτει τη πρόθεση του να διασχίσει την διασταύρωση
- CONFLICT : Δύο Duckiebots προσπάθησαν να "δεσμεύσουν" τη διασταύρωση σηματοδοτώντας ταυτόχρονα ένα σήμα RESERVING
- GO : Το Duckiebot να κινείται διασχίζοντας τη διασταύρωση.

Κεφάλαιο 12

Διάχυση της πλατφόρμας Duckietown στις βαθμίδες εκπαίδευσης

Μετά την περιγραφή του Duckietown project που έλαβε χώρα στα προηγούμενα κεφάλαια θα εξετάσουμε το αν και κατά πόσο είναι κατάλληλο να εισαχθεί σε κάποιες βαθμίδες της πρωτοβάθμιας ή δευτεροβάθμιας εκπαίδευσης ως εργαλείο τόσο για την εκμάθηση βασικών εννοιών της ρομποτικής όσο και για την εξοικείωση των μαθητών με αυτή. Θα ξεκινήσουμε περιγράφοντας συνοπτικά τι προβλέπεται από το αναλυτικό πρόγραμμα σπουδών τόσο για το Δημοτικό όσο και για το Γυμνάσιο και θα κάνουμε μετά μία συνολική κριτική ανάλυση για το αν και σε ποιο βαθμό η συγκεκριμένη πλατφόρμα ενδείκνυται για να καλύψει τους στόχους του αναλυτικού προγράμματος αλλά και αν συνάδει με το γνωστικό επίπεδο των μαθητών.

12.1 Πρωτοβάθμια εκπαίδευση

Αρχίζοντας από την πρωτοβάθμια εκπαίδευση τα αναλυτικά προγράμματα σπουδών που αφορούν τις ΤΠΕ (Τεχνολογίες της Πληροφορικής και των Επικοινωνιών) στο Δημοτικό έχουν ως γενικό σκοπό την καλλιέργεια τέτοιων ικανοτήτων στους μαθητές ώστε να μπορούν να χρησιμοποιούν τις ΤΠΕ με αποτελεσματικό, δημιουργικό και δεοντολογικό τρόπο ώστε να μπορούν να μετέχουν στην κοινωνία της γνώσης ως ενεργοί πολίτες. Εισάγεται η έννοια του Πληροφορικού γραμματισμού (ICT literacy) που περιγράφει την ικανότητα των μαθητών να χρησιμοποιούν τις σύγχρονες ψηφιακές τεχνολογίες, τα εργαλεία επικοινωνίας και τις δικτυακές υπηρεσίες για την προσπέλαση, διαχείριση, ενσωμάτωση, αξιολόγηση, δημιουργία και επικοινωνία πληροφοριών, με στόχο την επίλυση προβλημάτων και, τελικά, τη συμμετοχή τους στη σύγχρονη κοινωνία της γνώσης.

Με βάση λοιπόν τα παραπάνω το προτεινόμενο πλαίσιο ένταξης του πληροφορικού γραμματισμού και των ΤΠΕ στη βασική εκπαίδευση διαρθρώνεται σε τέσσερις αλληλοεξαρτώμενες συνιστώσες [45]:

- **Οι ΤΠΕ ως μαθησιακό-γνωστικό εργαλείο (cognitive tool):** Οι ΤΠΕ διατρέχουν οριζόντια όλα τα αντικείμενα του Προγράμματος Σπουδών και θεωρούνται μέσο υποστήριξης των σύγχρονων παιδαγωγικών προσεγγίσεων, εργαλείο επικοινωνίας, διερευνητικής και συνεργατικής μάθησης, ανάπτυξης της κριτικής σκέψης και της δημιουργικής ικανότητας των μαθητών.
- **Οι ΤΠΕ ως μεθοδολογία επίλυσης προβλημάτων:** Οι μαθητές και οι μαθήτριες εμπλέκονται σε δραστηριότητες επίλυσης προβλημάτων που έχουν ως σκοπό την καλλιέργεια δεξιοτήτων μεθοδολογικού χαρακτήρα (επεξεργασία δεδομένων, μοντελοποίηση λύσεων, δημιουργικότητα και καινοτομία) και δεξιοτήτων υψηλού επιπέδου (διερεύνηση, κριτική και αναλυτική σκέψη, συνθετική ικανότητα, ικανότητες επικοινωνίας και συνεργασίας).

- **Οι ΤΠΕ ως τεχνολογικό εργαλείο:** Οι μαθητές και οι μαθήτριες εξοικειώνονται με τους υπολογιστές και τα σύγχρονα εργαλεία των ΤΠΕ. Ο άξονας αυτός στοχεύει στη συνεχή ανάπτυξη τεχνικών δεξιοτήτων και στην επάρκεια χειρισμού των σύγχρονων περιβαλλόντων των ΤΠΕ (λογισμικά γενικής χρήσης, εκπαιδευτικό λογισμικό, υπηρεσίες Διαδικτύου κ.λπ.).
- **Οι ΤΠΕ ως κοινωνικό φαινόμενο:** Οι μαθητές και οι μαθήτριες γνωρίζουν και αξιολογούν τις εφαρμογές των ΤΠΕ στη σύγχρονη κοινωνία (διοίκηση, εργασία, επιστήμες, εκπαίδευση, ψυχαγωγία, πολιτισμός κ.λπ.). Απώτερος στόχος είναι να αποκτήσουν ευρύτερη **ψηφιακή παιδεία** και να διαμορφώσουν **στάσεις και αξίες**, ώστε να κατανοήσουν το νέο κοινωνικό και πολιτισμικό περιβάλλον που διαμορφώνεται στη σημερινή εποχή.

Η διδασκαλία του Πληροφορικού Γραμματισμού στο Δημοτικό έχει **σαφή εργαστηριακό προσανατολισμό.** Βασικός παράγοντας είναι η **ενεργός συμμετοχή κάθε μαθητή**, η συνεχής αλληλεπίδραση και η συνεργασία με τον διδάσκοντα και, κυρίως, με τους συμμαθητές του.
Το προτεινόμενο πλαίσιο ανάπτυξης των μαθητών στις ΤΠΕ συνίσταται σε τέσσερις διαστάσεις (συνιστώσες) αντίστοιχες με τη διάρθρωση της διδασκαλίας [45]:

- **Τεχνολογική**: Περιλαμβάνει τεχνικές γνώσεις που αφορούν σε θεμελιώδεις έννοιες ΤΠΕ (π.χ. υλικό, λογισμικό, αρχείο, δίκτυο κ.λπ.), και ικανότητες χρήσης βασικών περιβαλλόντων των ΤΠΕ (εκπαιδευτικό λογισμικό, επεξεργασία κειμένου, εννοιολογική χαρτογράφηση, λογισμικό παρουσιάσεων, υπηρεσίες Διαδικτύου κ.λπ.).

- **Γνωστική**: Περιγράφει τις θεμελιώδεις δεξιότητες αξιοποίησης των ΤΠΕ ως εργαλεία έρευνας, δημιουργίας, επικοινωνίας και μάθησης στα πλαίσια όλων των μαθημάτων του Προγράμματος Σπουδών αλλά και της καθημερινής σχολικής ζωής των μαθητών.

- **Επίλυση προβλήματος (problem solving):** Αφορά στην εφαρμογή και ολοκλήρωση των τεχνικών και γνωστικών δεξιοτήτων με στόχο την επίλυση προβλημάτων. Στο επίπεδο, ο άξονας αυτός καταγράφει δεξιότητες δημιουργικότητας, καινοτομίας και αλλαγής στάσεων και κοινωνικών συμπεριφορών για τις ΤΠΕ.

- **Κοινωνικές δεξιότητες**: Οι μαθητές και οι μαθήτριες αναπτύσσουν, επίσης, εκείνες τις κοινωνικές στάσεις και δεξιότητες που διαμορφώνουν τη σύγχρονη ψηφιακή κουλτούρα. Η διάσταση αυτή αφορά σε ζητήματα ηλεκτρονικής ασφάλειας, προστασίας προσωπικών δεδομένων, πληροφορικής ηθικής και δεοντολογίας, σε κώδικες διαχείρισης και αξιοποίησης πληροφοριών από πηγές κ.λπ.).

Κεντρικός στόχος της διδασκαλίας του μαθήματος είναι όλοι οι μαθητές και οι μαθήτριες να αναπτύξουν τις γνώσεις και τις ικανότητες χρήσης των ΤΠΕ μέσα από δραστηριότητες που αφορούν στην αναζήτηση και διαχείριση πληροφοριών, στην επίλυση προβλημάτων και στη λήψη αποφάσεων, στη δημιουργική έκφραση και στην επικοινωνία. Εξίσου σημαντικό είναι, μέσα από κατάλληλες χρήσεις και δραστηριότητες βασισμένες σε ΤΠΕ, οι μαθητές να κατανοήσουν τα όρια και την επίδραση που έχουν οι σύγχρονες τεχνολογίες στα άτομα και στις ομάδες, στις κοινότητες και στην κοινωνία ευρύτερα.

Η ενδεικτική κατανομή των Αξόνων μαθησιακών στόχων ανά Ενότητα και Τάξη παρουσιάζεται στον παρακάτω Πίνακα.

Άξονες μαθησιακών στόχων	**Προτεινόμενες ώρες διδασκαλίας**					
	Α'	**Β'**	**Γ'**	**Δ'**	**Ε'**	**Στ'**
Γνωρίζω, δημιουργώ και εκφράζομαι με ΤΠΕ	**12**	**12**	**10**	**10**	**6**	**6**
• Γνωρίζω και χειρίζομαι τον υπολογιστή	4	4	2	2		
• Δημιουργώ και εκφράζομαι με τη ζωγραφική, τα πολυμέσα και τις παρουσιάσεις	4	4	4	4	3	3
• Δημιουργώ με τον κειμενογράφο	4	4	4	4	3	3
Επικοινωνώ και συνεργάζομαι με ΤΠΕ	**6**	**6**	**6**	**6**	**6**	**6**
• Γνωρίζω το Διαδίκτυο	3	3	3	3	3	3
• Επικοινωνώ και συνεργάζομαι	3	3	3	3	3	3
Διερευνώ, ανακαλύπτω και λύνω προβλήματα με ΤΠΕ	**10**	**10**	**12**	**12**	**16**	**16**
• Μοντελοποιώ με εννοιολογικούς χάρτες	4	4	4	4		
• Λύνω προβλήματα με Υπολογιστικά Φύλλα					4	4
• Προγραμματίζω τον υπολογιστή					4	4
• Υλοποιώ σχέδια εργασίας/έρευνας (project)	6	6	8	8	8	8
Οι ΤΠΕ ως κοινωνικό φαινόμενο	**2**	**2**	**2**	**2**	**2**	**2**
• Οικοδομώ ψηφιακή παιδεία και γραμματισμό	2	2	2	2	2	2
Σύνολο διδακτικών ωρών	**30**	**30**	**30**	**30**	**30**	**30**

Πίνακας 6 : Ενδεικτική κατανομή διδακτικών ωρών ανά άξονα και τάξη.

Με βάση την παραπάνω ενδεικτική κατανομή ωρών διδασκαλίας ανά τάξη και μαθησιακό άξονα αλλά λαμβάνοντας υπόψη και το αναλυτικό πρόγραμμα σπουδών διαπιστώνουμε ότι από την Α' Δημοτικού μέχρι και την Δ' Δημοτικού δεν περιλαμβάνεται η ρομποτική ως προτεινόμενη δραστηριότητα για την υποστήριξη κάποιου μαθησιακού αποτελέσματος που να ανήκει στους άξονες των μαθησιακών στόχων. Και αυτό είναι πολύ λογικό και αναμενόμενο καθώς στην **Α' και Β' Δημοτικού** δίνεται έμφαση στην εξοικείωση των

μαθητών με τις περιφερειακές μονάδες του υπολογιστή και τον χειρισμό τους (ποντίκι, πληκτρολόγιο), στην γνωριμία τους με λογισμικά ζωγραφικής, συγγραφής και επεξεργασίας κειμένου αλλά και με φυλλομετρητές επισκεπτόμενοι προτεινόμενους ιστοτόπους. Γίνεται μία πρώτη γνωριμία των μαθητών με λογισμικά εννοιολογικής χαρτογράφησης και μια πρώτη αναφορά στους κανόνες ασφάλειας του Διαδικτύου και τέλος στα πλαίσια της ενότητας "Υλοποιώ σχέδια εργασίας/έρευνας" αναλαμβάνουν τη διεκπεραίωση μιας εργασίας που στηρίζεται στη χρήση των ψηφιακών αντικείμενων με τα οποία εξοικειώθηκαν όλη τη χρονιά.

Το αναλυτικό πρόγραμμα της **Γ' και Δ' τάξης του Δημοτικού** περιλαμβάνει συνοπτικά την εξοικείωση των μαθητών με το περιβάλλον του λειτουργικού συστήματος του υπολογιστή, την γνωριμία με κάποια εγκατεστημένα προγράμματα του υπολογιστή, μαθαίνουν να οργανώνουν και να διαχειρίζονται τα αρχεία (αποθήκευση, μετονομασία, διαγραφή), αναγνωρίζουν τη διάκριση των μονάδων που συνθέτουν ένα υπολογιστικό σύστημα, γνωρίζουν λογισμικά παρουσιάσεων και πολυμεσικών εφαρμογών και ανακαλύπτουν κάποιες δυνατότητες των επεξεργαστών κειμένου μέσα από προτεινόμενες δραστηριότητες. Έρχονται σε επαφή με κάποιες υπηρεσίες του διαδικτύου (ηλεκτρονικό ταχυδρομείο, ιστολόγια, εκπαιδευτική τηλεόραση, κτλ) και μπλέκονται με την εννοιολογική χαρτογράφηση και εκπονούν ένα project που στηρίζεται στην χρήση των ψηφιακών εργαλείων που γνώρισαν.

Παρά το γεγονός ότι δεν υπάρχει κάποια αναφορά ή έστω συνάφεια κάποιου αντικειμένου των αναλυτικών προγραμμάτων σπουδών της Α', Β' Γ' και Δ' τάξης του Δημοτικού με το δικό μας project παρόλα αυτά η διαδικασία τηλεχειρισμού του Duckiebot από παιδιά αυτής της ηλικίας είναι μία πολύ ενδιαφέρουσα και διασκεδαστική εμπειρία. Αυτό το διαπίστωσα από την μικρή μου κόρη (ηλικίας 6,5 ετών) όταν την προέτρεψα να τηλεκατευθύνει το Duckiebot, μέσω του προσομοιωτή joystick του Duckietown shell. Μάλιστα το γεγονός ότι η κάμερα "στέλνει" τη ροή εικόνων που συλλαμβάνει στον φορητό υπολογιστή μας κάνει την εμπειρία περισσότερο συναρπαστική καθώς επιτρέπει να τον τηλεχειρισμό του και χωρίς να υπάρχει ανάγκη οπτικής επαφής με αυτό. Πάνω σε αυτό το γεγονός θα μπορούσαν να γίνουν ασκήσεις δεξιοτεχνίας όπως για παράδειγμα η απομακρυσμένη οδήγηση του Duckiebot σε ένα συγκεκριμένο σημείο της πίστας. Ο παιγνιώδης χαρακτήρας που προσδίδουν τα παπάκια αρμόζει σε αυτές τις ηλικίες με αποτέλεσμα τα παιδιά να εμπλέκονται μέσα από μία διασκεδαστική εμπειρία στον κόσμο της ρομποτικής και αυτό νομίζω είναι το μεγαλύτερο όφελος που μπορούμε να επιτύχουμε για αυτές τις ηλικίες.

Στις τελευταίες δύο τάξεις του Δημοτικού (Ε' και ΣΤ') τα πράγματα διαφοροποιούνται. Στο αναλυτικό πρόγραμμα της **Ε' Δημοτικού**, στα πλαίσια της ενότητας **Προγραμματίζω τον υπολογιστή** υπάρχει αναφορά στην **εκπαιδευτική ρομποτική** ως εκπαιδευτικό υλικό για την εξοικείωση των μαθητών με τον προγραμματισμό μέσα από ένα παιγνιώδη τρόπο. Έτσι ανάμεσα στα άλλα εκπαιδευτικά υλικά Scratch, EasyLogo, Kodu και Microworlds Pro προτείνεται και η εκπαιδευτική ρομποτική. Προφανώς προτείνεται μία πλατφόρμα ρομποτικής όπου η συμπεριφορά των ρομπότ θα μπορούσε να προγραμματιστεί με ένα από τα παραπάνω προγραμματιστικά περιβάλλοντα (πχ. Scratch). Στα πλαίσια της ενότητας

"Υλοποιώ σχέδια εργασίας/έρευνας (project) με τις ΤΠΕ προτείνεται ανάμεσα στις άλλες ενδεικτικές δραστηριότητες και η Εκπαιδευτική ρομποτική (αν η σχολική μονάδα διαθέτει τον απαιτούμενο υλικοτεχνικό εξοπλισμό).

Υλοποιώ σχέδια εργασίας /έρευνας (project) με τις ΤΠΕ (Ε' Δημοτικού)	
Ενδεικτικές Δραστηριότητες	Εκπαιδευτικό υλικό
3. Εκπαιδευτική ρομποτική Οι μαθητές σε ομάδες των 3-4 ατόμων σχεδιάζουν και οργανώνουν την εργασία τους, διακρίνουν τα μέσα και τα εργαλεία του περιβάλλοντος της εκπαιδευτικής ρομποτικής. Συναρμολογούν το ρομπότ, σχεδιάζουν, υλοποιούν, ελέγχουν και βελτιώνουν απλούς και σύνθετους αλγόριθμους καθοδήγησης του ρομπότ. Ενδεικτικά παραδείγματα, σε συνεργασία και με άλλα μαθήματα, όπως η φυσική και τα μαθηματικά: • κινήσεις του ρομπότ σε ορθογώνιο επίπεδο (ευθύγραμμη, σε γεωμετρικές τροχιές που δημιουργούνται με στροφή υπό γωνία κ.λπ.) • μέτρηση απόστασης με βάση το χρόνο κίνησης του ρομπότ ***Ενδεικτικός διδακτικός χρόνος: 8 ώρες***	Διαθέσιμος εξοπλισμός ρομποτικής

Πίνακας 7 : Απόσπασμα από το Αναλυτικό Πρόγραμμα. Δραστηριότητες Εκπαιδευτικής Ρομποτικής Ε' Δημοτικού

Στην Στ' Δημοτικού συναντούμε πάλι την εκπαιδευτική ρομποτική ως ενδεικτική δραστηριότητα τα πλαίσια της ενότητας "Υλοποιώ σχέδια εργασίας/έρευνας (project) με τις ΤΠΕ. Ο παρακάτω πίνακας περιγράφει αυτή τη δραστηριότητα

Υλοποιώ σχέδια εργασίας /έρευνας (project) με τις ΤΠΕ (ΣΤ' Δημοτικού)	
Δραστηριότητες	Εκπαιδευτικό υλικό
3. Εκπαιδευτική ρομποτική Οι μαθητές, χωρισμένοι σε ομάδες των 3-4 ατόμων, σχεδιάζουν και οργανώνουν την εργασία τους, διακρίνουν τα μέσα και τα εργαλεία του Περιβάλλοντος της εκπαιδευτικής ρομποτικής. Συναρμολογούν το ρομπότ, σχεδιάζουν, υλοποιούν, ελέγχουν και βελτιώνουν τις διαδικασίες καθοδήγησης του ρομπότ. Ενδεικτικά θέματα που προτείνονται είναι: • Μέτρηση απόστασης με βάση την περίμετρο της ρόδας του ρομπότ • υπολογισμός περιμέτρου και εμβαδού επιφάνειας • κίνηση του ρομπότ σε λαβύρινθο ή χώρο με εμπόδια • κίνηση-ανταπόκριση του ρομπότ σε ηχητικές εντολές *Ενδεικτικός διδακτικός χρόνος: 8 ώρες*	Διαθέσιμος εξοπλισμός ρομποτικής

Πίνακας 8 : Απόσπασμα από το Αναλυτικό Πρόγραμμα .Προτεινόμενες Δραστηριότητες Εκπαιδευτικής Ρομποτικής ΣΤ' Δημοτικου

Μια πρώτη εκτίμηση που έχουμε να κάνουμε και για τις δραστηριότητες της εκπαιδευτικής ρομποτικής που προτείνονται τόσο στην Ε' όσο και στην ΣΤ' τάξη είναι ότι ο ενδεικτικός διδακτικός χρόνος κρίνεται οριακός για τις προτεινόμενες δραστηριότητες. Επίσης δεν προτείνεται κάποια συγκεκριμένη ρομποτική πλατφόρμα άρα είναι στην ευχέρεια του εκπαιδευτικού να επιλέξει μία (αν του παρέχει αυτή τη δυνατότητα η σχολικής μονάδα) ή να χρησιμοποιήσει αυτή που ήδη διαθέτει το σχολείο στο οποίο διδάσκει.

Η πλατφόρμα Duckietown θα μπορούσε να υποστηρίξει αρκετές από τις παραπάνω δραστηριότητες. Οι μαθητές χωρισμένοι σε ομάδες θα μπορούσαν να διαμορφώσουν την πόλη (Duckietown) αλλά και να συναρμολογήσουν τα Duckiebots πάντα βέβαια με την καθοδήγηση κάποιου εκπαιδευτικού. Επίσης η μέτρηση της απόστασης με βάση τη διάμετρο των τροχών θα μπορούσε να ήταν ένας εφικτός στόχος όπως και ασκήσεις τηλεχειρισμού του Duckiebot μέσα στην πόλη. Σαφώς όμως δεν είναι μια πλατφόρμα που θα προσφέρει ένα φιλικό περιβάλλον προγραμματισμού. Τόσο οι αλγόριθμοι που εκτελούνται από τα Duckiebots όσο ο τρόπος που επιτυγχάνεται αυτό έχει αυξημένη πολυπλοκότητα και απαιτούν πρότερες γνώσεις που δεν είναι δυνατόν να έχουν μαθητές αυτής της βαθμίδας. Προσφέρεται δηλαδή για να διαπιστώσουν έμπρακτα οι μαθητές το "τι" μπορεί να κάνει ένα ρομποτικό όχημα αυτόνομα αλλά κρύβοντας λεπτομέρειες για το "πως" το επιτυγχάνει. Σίγουρα υπάρχουν άλλες πλατφόρμες με πιο παιγνιώδη εμφάνιση, μεγαλύτερη γκάμα διαθέσιμων αισθητήρων και ενεργοποιητών και πιο φιλικό περιβάλλον προγραμματισμού με εντολές απλές που μπορούν να κατανοήσουν οι μαθητές αυτής της ηλικίας και της εκπαιδευτικής βαθμίδας.

12. 2 Δευτεροβάθμια Εκπαίδευση

12.2. 1 Γυμνάσιο

Το αναλυτικό πρόγραμμα σπουδών του Γυμνάσιου [46] βασίζεται και αυτό στις τέσσερις διαστάσεις (συνιστώσες) που περιγράφηκαν στο αντίστοιχο αναλυτικό πρόγραμμα του Δημοτικού δηλαδή:

- Τεχνολογική
- Γνωστική
- Επίλυση προβλήματος (sovling)
- Κοινωνικές δεξιότητες

Οι ανεπαίσθητες διαφορές στην περιγραφή κάθε συνιστώσας σε σχέση με την πρωτοβάθμια εκπαίδευση είναι ότι στα πλαίσια της διάστασης Επίλυση προβλήματος εισάγεται ο στόχος της ανάπτυξης υπολογιστικής σκέψης. Ως βασική τεχνική διδασκαλίας κυρίως στο εργαστηριακό μέρος του μαθήματος προτείνονται τα σχέδια εργασίας/έρευνας (projects). Τα οποία σε ορισμένες περιπτώσεις είναι δυνατό να συνδυάσουν τη διδασκαλία πολλών θεματικών ενοτήτων της Πληροφορικής μαζί, αλλά και να αξιοποιήσουν διαθεματικές και διεπιστημονικές προσεγγίσεις.

Αρχίζοντας από την **Α' τάξη του Γυμνασίου** η ενδεικτική κατανομή των διδακτικών ωρών ανά μαθησιακό άξονα έχει ως εξής:

Άξονες Προσδοκώμενων Μαθησιακών Αποτελεσμάτων	Προτεινόμενες ώρες διδασκαλίας
Η Πληροφορική στον σύγχρονο κόσμο • Βασικές έννοιες	**8**
Χειρίζομαι και δημιουργώ • Δημιουργώ με τον κειμενογράφο	**18**
Αναζητώ πληροφορίες, επικοινωνώ και συνεργάζομαι • Γνωρίζω το Διαδίκτυο και επικοινωνώ	**12**
Διερευνώ, ανακαλύπτω και λύνω προβλήματα • Προγραμματίζω υπολογιστικές συσκευές και ρομποτικά συστήματα	**14**

Πίνακας 9: Ενδεικτική κατανομή ωρών ανά μαθησιακό άξονα (Α' Γυμνασίου)

Σύμφωνα πάντα με το αναλυτικό πρόγραμμα σπουδών προτείνεται η ενασχόληση με την εκπαιδευτική ρομποτική στα πλαίσια την ενότητας Διερευνώ, ανακαλύπτω και λύνω προβλήματα και σε όλες τις τάξεις του Γυμνασίου . Η αναλυτική περιγραφή των μαθησιακών αποτελεσμάτων, των δραστηριοτήτων και του προτεινόμενου εκπαιδευτικού υλικού αυτής της προτεινόμενης δραστηριότητας περιγράφεται στους παρακάτω πίνακες:

Διερευνώ, ανακαλύπτω και λύνω προβλήματα (Α' Γυμνασίου) Προγραμματίζω υπολογιστικές συσκευές και ρομποτικά συστήματα	
Δραστηριότητες	Εκπαιδευτικό υλικό
Εκπαιδευτική ρομποτική (ως εναλλακτική δραστηριότητα) Οι μαθητές σε ομάδες των 3-4 ατόμων σχεδιάζουν και οργανώνουν την εργασία τους, διακρίνουν τα μέσα και τα εργαλεία του περιβάλλοντος της εκπαιδευτικής ρομποτικής, αναλαμβάνουν ρόλους. Συναρμολογούν το ρομπότ και εξοικειώνονται με το περιβάλλον προγραμματισμού και καθοδήγησης του ρομπότ (εντολές κίνησης, εντολές ελέγχου, εντολές ελέγχου αισθητήρων κ.λπ. Σχεδιάζουν, υλοποιούν, ελέγχουν και βελτιώνουν απλούς και σύνθετους αλγόριθμους καθοδήγησης του ρομπότ. Ενδεικτικά προτείνονται ενέργειες του ρομπότ, όπως: • να διαγράψει ένα τετράγωνο • να ακολουθήσει μια μαύρη γραμμή • να βγει από έναν λαβύρινθο • να παίξει μουσική • να βρει να συλλέξει αντικείμενα	Συστήματα Εκπαιδευτικής Ρομποτικής (Arduino με Scratch, Raspberry Pi με Scratch, κα.)

Πίνακας 10 : Προτεινόμενες Δραστηριότητες Εκπαιδευτικής Ρομποτικής Α' τάξη Γυμνασίου

Στην **Β' τάξη του Γυμνασίου** η ενδεικτική κατανομή των διδακτικών ωρών ανά μαθησιακό άξονα έχει ως εξής:

Άξονες προσδοκώμενων μαθησιακών αποτελεσμάτων	Προτεινόμενες ώρες διδασκαλίας
Η Πληροφορική στο σύγχρονο κόσμο • Βασικές έννοιες	6
Διερευνώ, ανακαλύπτω και λύνω προβλήματα • Προγραμματίζω υπολογιστικές συσκευές και ρομποτικά συστήματα • Λύνω προβλήματα με υπολογιστικά φύλλα	13
Αναζητώ πληροφορίες, επικοινωνώ και συνεργάζομαι • Δημιουργώ και εκφράζομαι με πολυμέσα και παρουσιάσεις • Διερευνώ και συνεργάζομαι μέσω του Διαδικτύου	6

Πίνακας 11: Ενδεικτική κατανομή ωρών βάσει των 3 μαθησιακών αξόνων Β' τάξη Γυμνασίου

Ακολουθεί ο πίνακας που περιγράφει τις προτεινόμενες δραστηριότητες της εκπαιδευτικής ρομποτικής για τους μαθητές της Β' τάξης του Γυμνασίου

Δραστηριότητες	Εκπαιδευτικό υλικό
Εκπαιδευτική ρομποτική Οι μαθητές σε ομάδες των 3-4 ατόμων σχεδιάζουν και οργανώνουν την εργασία τους, διακρίνουν τα μέσα και τα εργαλεία του περιβάλλοντος της εκπαιδευτικής ρομποτικής, αναλαμβάνουν ρόλους. Συναρμολογούν το ρομπότ και εξοικειώνονται με το περιβάλλον προγραμματισμού και καθοδήγησης του ρομπότ (εντολές κίνησης, εντολές ελέγχου, εντολές ελέγχου αισθητήρων κλπ). Σχεδιάζουν, υλοποιούν, ελέγχουν και βελτιώνουν απλούς αλγορίθμους καθοδήγησης του ρομπότ. Ενδεικτικά προτείνονται ενέργειες του ρομπότ όπως: • να διαγράψει ένα τετράγωνο • να ακολουθήσει μια μαύρη γραμμή Ενδεικτικός χρόνος : 5 ώρες	Συστήματα Εκπαιδευτικής Ρομποτικής (Arduino με Scratch, Raspberry Pi με Scratch, κα.)

Πίνακας 12 : Προτεινόμενες Δραστηριότητες Εκπαιδευτικής Ρομποτικής Β' Γυμνασίου

Τέλος στην **Γ' τάξη του Γυμνασίου** η ενδεικτική κατανομή των διδακτικών ωρών ανά μαθησιακό άξονα έχει ως εξής:

Άξονες προσδοκώμενων μαθησιακών αποτελεσμάτων	Προτεινόμενες ώρες διδασκαλίας
Διερευνώ, σχεδιάζω και λύνω προβλήματα • Προγραμματίζω υπολογιστικές συσκευές και ρομποτικά συστήματα	14
Δημιουργώ, παρουσιάζω, επικοινωνώ και συνεργάζομαι • Δημιουργώ έγγραφα και συνεργάζομαι σε διαδικτυακά περιβάλλοντα • Δημιουργώ Παρουσιάσεις	11

Πίνακας 13: Ενδεικτική κατανομή διδακτικών ωρών βάσει 2 μαθησιακών αξόνων Γ' τάξη του Γυμνασίου

Ακολουθεί ο πίνακας που περιγράφει τις προτεινόμενες δραστηριότητες της εκπαιδευτικής ρομποτικής για τους μαθητές της Γ' τάξης του Γυμνασίου

Διερευνώ, σχεδιάζω και λύνω προβλήματα (Γ' Γυμνασίου) Προγραμματίζω υπολογιστικές συσκευές και ρομποτικά συστήματα	
Δραστηριότητες	Εκπαιδευτικό υλικό
Εκπαιδευτική ρομποτική (ως εναλλακτική δραστηριότητα) Οι μαθητές σε ομάδες των 3-4 ατόμων σχεδιάζουν και οργανώνουν την εργασία τους, διακρίνουν τα μέσα και τα εργαλεία του περιβάλλοντος της εκπαιδευτικής ρομποτικής, αναλαμβάνουν ρόλους. Συναρμολογούν το ρομπότ και εξοικειώνονται με το περιβάλλον προγραμματισμού και καθοδήγησης του ρομπότ (εντολές κίνησης, εντολές ελέγχου, εντολές ελέγχου αισθητήρων κ.λπ. Σχεδιάζουν, υλοποιούν, ελέγχουν και βελτιώνουν απλούς και σύνθετους αλγόριθμους καθοδήγησης του ρομπότ. Ενδεικτικά προτείνονται ενέργειες του ρομπότ, όπως: • να διαγράψει ένα τετράγωνο • να ακολουθήσει μια μαύρη γραμμή • να βγει από έναν λαβύρινθο • να παίξει μουσική • να βρει να συλλέξει αντικείμενα	Συστήματα Εκπαιδευτικής Ρομποτικής (Arduino με Scratch, Raspberry Pi με Scratch, κα.)

Πίνακας 14: Προτεινόμενες δραστηριότητες εκπαιδευτικής ρομποτικής για την Γ' τάξη του Γυμνασίου

Κάποιες δραστηριότητες από αυτές που περιγράφονται στο αναλυτικό πρόγραμμα σπουδών τόσο για την Α' τάξη όσο για την Β' και την Γ' τάξη του Γυμνασίου μπορούν ως ένα βαθμό να υλοποιηθούν την πλατφόρμα του Duckietown. Είναι εφικτό δηλαδή οι μαθητές να συναρμολογήσουν ένα Duckiebot, να διαμορφώσουν την πόλη - πίστα, να ελέγξουν την κίνηση του απομακρυσμένα, να υπολογίσουν διανυθέντα απόσταση με βάση τη διάμετρο των τροχών του, να εξοικειωθούν τον τρόπο λειτουργίας της διαφορικής οδήγησης, να δουν τη ροή εικόνων που στέλνει η κάμερα στο φορητό υπολογιστή και να γνωρίσουν ένα διαφορετικό λειτουργικό σύστημα (Ubuntu) και τη γραμμή εντολών του.

Εκείνο όμως που θα αδυνατήσουν να υλοποιήσουν είναι ο προγραμματισμός της συμπεριφοράς του Duckiebot που είναι ένα από τα βασικότερα ζητούμενα στην ρομποτική. Καταρχάς η συμπεριφορά του Duckiebot βασίζεται στην επεξεργασία των δεδομένων του μοναδικού αισθητήρα που διαθέτει, δηλαδή στην κάμερα. Αυτό από μόνο του είναι ένα αποθαρρυντικός παράγοντας για μαθητές αυτής της ηλικίας και αυτής της εκπαιδευτικής βαθμίδας καθώς αυτή η "επεξεργασία" των δεδομένων διεκπεραιώνεται μέσα από μία ακολουθία εφαρμογής πολύπλοκων, υπολογιστικών αλγορίθμων (μετασχηματισμούς, φίλτρα, κτλ) και απαιτούν βασικές γνώσεις υπολογιστικής όρασης. Επίσης το γεγονός ότι ο προγραμματισμός αυτός θα πρέπει να είναι σε μία γλώσσα προγραμματισμού όπως η Python που δεν έχουν διδαχθεί δυσκολεύει ακόμα περισσότερο το όλο εγχείρημα. Αν σε όλα αυτά συνυπολογίσουμε την εμπλοκή του ROS και τον τρόπο εκτέλεσης των αλγορίθμων μέσω Docker containers εύκολα μπορούμε να αντιληφθούμε η συγκεκριμένη πλατφόρμα δεν

ενδείκνυται για τον προγραμματισμό της συμπεριφοράς των ρομποτικών κατασκευών σε αυτή τη βαθμίδα εκπαίδευσης.

Συμπερασματικά η πλατφόρμα του Duckietown μπορεί να δώσει την ευκαιρία στους μαθητές των δύο τελευταίων τάξεων του Δημοτικού αλλά και των δύο πρώτων του Γυμνασίου να έχουν μία πρώτη απτή επαφή και μια εμπειρία με αυτό που ονομάζουμε ρομποτική. Μπορεί να τους βοηθήσει να αντιληφτούν τα βασικά δομικά υλικά ενός ρομποτικού οχήματος όπως κινητήρες, αισθητήρας, υπολογιστής πλακέτας και να εμπλακούν στη διαδικασία συναρμολόγησης του Duckiebot και διαμόρφωσης της πόλης (Duckietown). Τέλος θα μπορούσαν να πάρουν μέρος σε "ασκήσεις" τηλεχειρισμού των Duckiebots με σκοπό να τη πλοήγηση τους μέσα στην πόλη και τη στάθμευση τους σε κάποιο συγκεκριμένο σημείο. Κάπου εκεί όμως σταματάει και ο ρόλος αυτής της πλατφόρμας για μαθητές αυτών των βαθμίδων εκπαίδευσης καθώς η πολυπλοκότητα των αλγορίθμων που εκτελούνται, ο τρόπος που γίνεται αυτό (μέσω Docker containers και ROS), η γλώσσα προγραμματισμού που χρησιμοποιείται και οι γνώσεις που απαιτούνται για να γίνουν κατανοητά πολλά αντικείμενα της υπολογιστικής όρασης υπερβαίνουν τα γνωστικό επίπεδο των μαθητών αυτών των βαθμίδων. Για το λόγο αυτό άλλωστε προτείνονται άλλες πλατφόρμες όπως τα Lego Mindstorms NXT όπου παρέχουν είναι σαφώς πιο εύκολο, φιλικό και πιο απλό τρόπο προγραμματισμού αλλά και ένα μεγαλύτερο εύρος αισθητήρων και ενεργοποιητών για εξοπλίσουν τις ρομποτικές τους κατασκευές ενώ εξαιτίας της φύσης τους είναι πιο ελκυστικά καθώς είναι πιο κοντά σε γνωστές εικόνες και αναπαραστάσεις που έχουν οι μαθητές.

12.2.2 Γενικό Λύκειο

Στο ωρολόγιο πρόγραμμα της **Α' τάξης του Γενικού Λυκείου** συναντούμε το μάθημα **Εφαρμογές Πληροφορικής** το οποίο διδάσκεται 2 ώρες την εβδομάδα. Με βάση τις οδηγίες διδασκαλίας του μαθήματος [47] η διδακτική του βασίζεται στον κοινωνικό εποικοδομισμό και τις σύγχρονες θεωρήσεις για την «επεξεργασία των πληροφοριών». Στο πλαίσιο του μαθήματος ενισχύεται η διερευνητική προσέγγιση, η αυτενέργεια και η συνεργατική μάθηση. Προτείνεται η ευθυγράμμιση με ενεργητικές εκπαιδευτικές τεχνικές και η χρησιμοποίηση αυθεντικών παραδειγμάτων από τον πραγματικό κόσμο. Επίσης προτείνεται η χρήση κατάλληλων διδακτικών σεναρίων τα οποία αποτελούν έναν σαφή και πρακτικό τρόπο προκειμένου να εξειδικευτούν οι γενικές αρχές του Προγράμματος Σπουδών (ΠΣ) και βοηθούν στην οργάνωση της διδασκαλίας κυρίως μέσω των δραστηριοτήτων που δίνονται στους μαθητές. Τέτοια σενάρια, αξιολογημένα ως επαρκή ή βέλτιστα υπάρχουν στην πλατφόρμα "Αίσωπος" (http://aesop.iep.edu.gr/). Τέλος σύμφωνα με το Πρόγραμμα Σπουδών (ΠΣ) το μάθημα «Εφαρμογές Πληροφορικής» έχει σαφή εργαστηριακό προσανατολισμό. Η διδακτέα ύλη περιλαμβάνει τρεις βασικές θεματικές ενότητες.

- Η θεματική ενότητα 2: Προγραμματιστικά περιβάλλοντα - Δημιουργία Εφαρμογών. Έχει σκοπό να επεκτείνει τις βασικές γνώσεις προγραμματισμού που απέκτησαν οι μαθητές στο Δημοτικό και στο Γυμνάσιο και τους δίνει την ευκαιρία να γνωρίσουν

και άλλα προγραμματιστικά περιβάλλοντα μέσα από ενδεικτικές δραστηριότητες ανάπτυξης μικροεφαρμογών.

- Η ενότητα 3: Επικοινωνία και Διαδίκτυο. Έχει ως στόχο οι μαθητές να εμβαθύνουν στις υπηρεσίες του Διαδικτύου και τις Web 2.0 εφαρμογές, να αναγνωρίζουν κώδικα HTML, να μπορούν να τον επεξεργαστούν και να τον ενσωματώσουν σε Διαδικτυακές εφαρμογές.
- Η ενότητα 4: Συνεργασία και ασφάλεια. Έχει σκοπό να εισαγάγει τους μαθητές στη χρήση των εφαρμογών Νέφους που προσφέρονται στο Διαδίκτυο για τη δημιουργία - διαχείριση εγγράφων και τη συνεργασία από απόσταση.

Από τις παραπάνω ενότητες μόνο στα πλαίσια της ενότητας 2 προτείνεται μία δραστηριότητα κατασκευής ρομπότ Lego Mindstorm το οποίο θα προγραμματιστεί με την διαδικτυακή πλατφόρμα του App Inventor όπως φαίνεται και στον αναλυτικό πίνακα που ακολουθεί.

Ενότητα 2: Προγραμματιστικά Περιβάλλοντα - Δημιουργία Εφαρμογών		
Θεματικές Ενότητες	Δραστηριότητες	Εκπαιδευτικό υλικό
7.1 Προγραμματισμός εφαρμογών για φορητές συσκευές	Προγραμματισμός έξυπνων φορητών συσκευών με την υλοποίηση μικροεφαρμογών σε στο App Inventor, Inventor, Alice, Snap!, Blockly, Greenfoot, κ.α. Προτείνεται η υλοποίηση μιας ολοκληρωμένης εφαρμογής υπό τη μορφή Project, όπως: ● εφαρμογή υπολογισμού τελικού αριθμού μορίων σε πανελλαδικές εξετάσεις ● mobile app τουριστικός οδηγός-αξιοθέατα της περιοχής μας ● παιχνίδι λαβύρινθος ● κατασκευή ρομπότ (εφόσον είναι διαθέσιμο σχετικό υλικό) και κίνηση του ρομπότ με το App Inventor, το οποίο θα αποφεύγει εμπόδια και θα κινείται με φωνητική καθοδήγηση.	Μαθησιακά αντικείμενα από το Φωτόδεντρο και τον Αίσωπο App Inventor: Διδασκαλία Προγραμματισμού με Δημιουργία Εφαρμογών για έξυπνες φορητές Συσκευές ●http://photodentro.edu.gr/aggregator/lo/photodentro-aggregatedcontent-8526-8268 ● http://aesop.iep.edu.gr/node/13460
7.2 Αντικειμενοστραφής προγραμματισμός σε 3D περιβάλλον		Εισαγωγή στον αντικειμενοστραφή προγραμματισμό με την βοήθεια παιχνιδιών: Greenfoot ●http://photodentro.edu.gr/aggregator/lo/photodentro-aggregatedcontent-8526-8074 ●http://aesop.iep.edu.gr/node/15856 Καθοδήγηση Lego Mindstorm με τη χρήση του App Inventor ●http://photodentro.edu.gr/aggregator/lo/photodentro-aggregatedcontent-8526-8403 ● http://aesop.iep.edu.gr/node/11425

Πίνακας 15 : Προτεινόμενες Δραστηριότητες της Ενότητας 2 για το μάθημα Εφαρμογές Πληροφορικής

Στην **Β' τάξη του Γενικού Λυκείου** συναντούμε το μάθημα **Εισαγωγή στις αρχές της επιστήμης των Η/Υ** για δύο (2) ώρες την εβδομάδα. Σκοπός του μαθήματος είναι να γνωρίσουν οι μαθητές τομείς και θεμελιώδεις έννοιες της Επιστήμης Υπολογιστών και Πληροφορικής και να αναπτύξουν την αναλυτική και συνθετική τους σκέψη. Η προσέγγιση που ακολουθείται σχετίζεται με θέματα τόσο της Θεωρητικής όσο και της Εφαρμοσμένης Επιστήμης των Υπολογιστών. Με το πρώτο μέρος να καλύπτει θέματα της Θεωρητικής Επιστήμης των Υπολογιστών –από το Πρόβλημα στον Αλγόριθμο και από εκεί στον Προγραμματισμό και τις Εφαρμογές του– και το δεύτερο μέρος με την επισκόπηση βασικών τομέων της Εφαρμοσμένης Επιστήμης των Υπολογιστών. Πιο αναλυτικά:

α) στο πρώτο μέρος που καλύπτονται θέματα της Θεωρητικής Επιστήμης των Υπολογιστών δίνεται βαρύτητα στα ακόλουθα:

- ✓ Έννοια του Προβλήματος
- ✓ Από το πρόβλημα στον αλγόριθμο
- ✓ Ανάπτυξη αλγορίθμων
- ✓ Είδη και τεχνικές προγραμματισμού

β) Στο δεύτερο μέρος γίνεται επισκόπηση δύο βασικών τομέων της Εφαρμοσμένης Επιστήμης των Υπολογιστών (Λειτουργικά Συστήματα, Πληροφοριακά Συστήματα και Δίκτυα και Τεχνητή Νοημοσύνη).

Τέλος στην Γ' τάξη του Γενικού Λυκείου υπάρχει το Πανελλαδικώς εξεταζόμενο μάθημα Πληροφορική (Ανάπτυξη Εφαρμογών σε Προγραμματιστικό Περιβάλλον) για το οποίο προβλέπονται έξι (6) διδακτικές ώρες την εβδομάδα. Σκοπός του μαθήματος είναι οι μαθητές και οι μαθήτριες να αναπτύξουν ικανότητες αναλυτικής και συνθετικής σκέψης, ώστε να επιλύουν προβλήματα, να σχεδιάζουν αλγορίθμους και να δημιουργούν προγράμματα μέσω εφαρμοσμένων προσεγγίσεων μεθοδολογικού χαρακτήρα. Συνδυάζοντας υπάρχουσες βιωματικές και μαθησιακές εμπειρίες να οικοδομούν τη θεμελίωση νέων γνώσεων με χρήση εννοιών, μεθόδων και εργαλείων της Επιστήμης της Πληροφορικής και να αποκτούν το κατάλληλο επιστημονικό υπόβαθρο για την αξιοποίησή τους και σε άλλες επιστήμες. Πιο αναλυτικά η διδακτέα ύλη περιλαμβάνει τέσσερις (4) θεματικές ενότητες:

- **1η Θεματική Ενότητα: Ανάλυση Προβλήματος.** Σκοπός αυτής της ενότητας είναι οι μαθητές να αναπτύξουν ικανότητες αναλυτικής και συστηματικής προσέγγισης στη διαδικασία επίλυσης προβλημάτων.
- **2η Θεματική Ενότητα: Σχεδίαση αλγορίθμων.** Αυτή η ενότητα στοχεύει στην απόκτηση ικανοτήτων μεθοδολογικού χαρακτήρα από τους μαθητές που αφορούν τον σχεδιασμό, την ανάπτυξη και τον έλεγχο αλγορίθμων.
- **3η Θεματική Ενότητα: Υλοποίηση σε προγραμματιστικό περιβάλλον.** Έχει σκοπό οι μαθητές να αποκτήσουν δεξιότητες υλοποίησης αλγορίθμων σε σύγχρονα προγραμματιστικά περιβάλλοντα.
- **4η Θεματική Ενότητα: Αξιολόγηση – Τεκμηρίωση.** Στοχεύει στη απόκτηση ικανοτήτων τεκμηρίωσης και αξιολόγησης των προγραμμάτων που έχουν συντάξει.

Στο σημείο αυτό τα σημειώσουμε ότι στα πλαίσια αυτού του μαθήματος έχει επιλέγει ως γλώσσα προγραμματισμού για την υλοποίηση των αλγορίθμων μία ψευδογλώσσα που κατασκευάστηκε για τις ανάγκες του μαθήματος και ονομάζεται ΓΛΩΣΣΑ και όχι μία υπαρκτή, σύγχρονη γλώσσα προγραμματισμού.

Με βάση λοιπόν τα αναλυτικά προγράμματα σπουδών των μαθημάτων πληροφορικής του Γενικού Λυκείου διαπιστώνουμε ότι δεν υπάρχει αρκετός "χώρος " για μια πλατφόρμα ρομποτικής όπως αυτή του Duckietown. Αναφορά για κατασκευή ρομπότ γίνεται στην Α' τάξη μέσα από μια δραστηριότητα για την υλοποίηση της οποίας προτείνονται τα Lego Maindstorms σε συνδυασμό με το προγραμματιστικό περιβάλλον του App Inventor. Επίσης στο μάθημα της Β΄ τάξης, στο δεύτερο μέρος της ύλης όπου γίνεται λόγος για ερευνητικές εργασίες που έχουν σχέση με την τεχνητή νοημοσύνη προτείνεται ως θέμα: "τα αυτοκίνητα χωρίς οδηγό". Ο βασικός προσανατολισμός όμως των μαθημάτων τόσο της Β' όσο και της Γ' τάξης έχουν ως σκοπό την εξοικείωση των μαθητών με την επίλυση προβλημάτων, την ανάπτυξη αλγοριθμικής σκέψης και τη μεταφορά των αλγορίθμων σε ένα εκπαιδευτικό προγραμματιστικό περιβάλλον.

12.2.3 ΕΠΑ.Λ (Επαγγελματικό Λύκειο)

Στα επαγγελματικά λύκεια οι μαθητές την Α' τάξη διδάσκονται μαθήματα γενικής παιδείας και μαθήματα επιλογής. Στην Β' τάξη καλούνται να επιλέξουν έναν τομέα της αρεσκείας τους και στην Γ' τάξη μία ειδικότητα που να ανήκει στον τομέα που επέλεξαν στην Β' τάξη. Είναι αυτονόητο πως σε κάθε ΕΠΑ.Λ δεν λειτουργούν όλοι οι διαθέσιμοι τομείς και οι αντίστοιχες ειδικότητες. Εμείς θα εξετάσουμε το αναλυτικό πρόγραμμα των μαθημάτων πληροφορικής γενικής παιδείας όσο και των μαθημάτων των τομέων πληροφορικής και ηλεκτρονικών της Β' τάξης και των αντίστοιχων ειδικοτήτων τους της Γ' τάξης για να εντοπίσουμε σκοπούς και στόχους του αναλυτικού προγράμματος που θα μπορούσαν να εκπληρωθούν με την πλατφόρμα Duckietown.

Στην **Α' τάξη** υπάρχει το μάθημα **Πληροφορική** το οποίο διδάσκεται δύο ώρες την εβδομάδα. Ο σκοπός του μαθήματος, οι επιδιωκόμενοι στόχοι και η διδακτέα ύλη [48] είναι η ίδια με αυτή του μαθήματος Εφαρμογές Πληροφορικής της Α' τάξης του Γενικού Λυκείου.

Στην **Β' τάξη** έχουμε το μάθημα γενικής παιδείας **Εισαγωγή τις Αρχές Επιστήμης των Υπολογιστών** για μια (1) ώρα την εβδομάδα το οποίο σύμφωνα με το ΦΕΚ 2010/τ.Β'/16-9-2015 έχει ως σκοπό να βοηθήσει τους μαθητές να γνωρίσουν βασικούς τομείς και θεμελιώδεις έννοιες της Επιστήμης των Υπολογιστών και της Πληροφορικής και να αναπτύξουν την αναλυτική και συνθετική τους σκέψη. Ειδικότερα στα πλαίσια αυτού του μαθήματος οι μαθητές να μπορούν να αποσαφηνίσουν βασικές έννοιες της Επιστήμης των Υπολογιστών, να έρθουν σε επαφή με την έννοια του προβλήματος, τις κατηγορίες των προβλημάτων και τις διαδικασίες επίλυσής τους, να γνωρίσουν βασικούς τύπους και δομές δεδομένων, να διακρίνουν τις βασικές εντολές και δομές που χρησιμοποιούνται σε έναν

αλγόριθμο, να επιλύουν απλά προβλήματα διατυπώνοντας τη λύση σε μορφή αλγορίθμου και τέλος να γνωρίσουν διαφορετικούς αλγορίθμους για την επίλυση απλών προβλημάτων και να προβληματιστούν για τα χαρακτηριστικά τους [48].

Στην Β' τάξη λειτουργεί ο τομέας Πληροφορικής που έχει συνάφεια το αντικείμενο μας. Οι μαθητές παρακολουθούν 23 ώρες μαθημάτων Τομέα και δώδεκα (12) ώρες μαθημάτων γενικής παιδείας. Σύμφωνα με τα οριζόμενα στο Φ.Ε.Κ. 1489/τ. Β'/26-5-2016, από το σχολικό έτος 2017-2018, στον **Τομέα Πληροφορικής** στη Γ' τάξη των Ημερησίων και Εσπερινών ΕΠΑ.Λ., προβλέπεται να λειτουργούν οι εξής ειδικότητες:

- Τεχνικός Εφαρμογών Πληροφορικής
- Τεχνικός Η/Υ και Δικτύων Η/Υ

Σύμφωνα με το Φ.Ε.Κ. 1567, τ.Β', 2-6-2016 τα **μαθήματα Τομέα** που παρακολουθούν οι μαθητές και μαθήτριες του **Τομέα Πληροφορικής** στη Β' τάξη του Ημερησίου ΕΠΑ.Λ. έχουν ως εξής:

Τομέας Πληροφορικής - Β' τάξη		
	Μαθήματα	**Ώρες**
1.	Αρχές Προγραμματισμού Υπολογιστών	1Θ+3Ε
2.	Υλικό και Δίκτυα Υπολογιστών	2Θ+2Ε
3.	Βασικά θέματα Πληροφορικής	2Θ+2Ε
4.	Λειτουργικά Συστήματα και Ασφάλεια Πληροφοριακών Συστημάτων	1Θ+2Ε
5.	Σχεδιασμός και Ανάπτυξη Ιστοτόπων	4Ε
6.	Τεχνικά Θέματα Πωλήσεων	1Θ+2Ε
7.	Αγγλικά Ειδικότητας	1 Θ
Σύνολο		23 ώρες

Πίνακας 16 : Μαθήματα Τομέα Πληροφορικής ΕΠΑ.Λ - Β' Τάξη

Σύμφωνα με το Φ.Ε.Κ. 1426, τ.Β', 26-4-2017, όπως τροποποιήθηκε με το Φ.Ε.Κ. 2072, τ.Β', 15-6-2017 τα **μαθήματα Ειδικότητας** που παρακολουθούν οι μαθητές και μαθήτριες του **Τομέα Πληροφορικής** στη Γ' τάξη του Ημερησίου ΕΠΑ.Λ. έχουν ως εξής:

Ειδικότητα: Τεχνικός Η/Υ και Δικτύων Η/Υ - Γ' τάξη		
	Μαθήματα	Ώρες
1.	Προγραμματισμός Υπολογιστών	3Θ
2.	Δίκτυα Υπολογιστών	3Θ
3.	Προγραμματισμός Υπολογιστών (Εργαστήριο)	2Ε
4.	Δίκτυα Υπολογιστών (Εργαστήριο)	2Ε
5.	Πληροφοριακά Συστήματα σε Επιχειρήσεις και Οργανισμούς	2Θ
6.	Εγκατάσταση, Διαχείριση και Συντήρηση Υπολογιστικών Συστημάτων	4Ε
7.	Ειδικά Θέματα στο Υλικό και στα Δίκτυα Υπολογιστών	4Ε
8.	Τεχνική Υποστήριξη Υπολογιστικών Συστημάτων και Δικτυακών Υποδομών	3Ε
	Σύνολο	23 ώρες

Πίνακας 17 : Μαθήματα Ειδικότητας Τεχνικός Η/Υ και Δικτύων Γ' τάξη ΕΠΑ.Λ

Ειδικότητα: Τεχνικός Εφαρμογών Πληροφορικής - Γ' τάξη		
	Μαθήματα	**Ώρες**
1.	Προγραμματισμός Υπολογιστών	3Θ
2.	Δίκτυα Υπολογιστών	3Θ
3.	Προγραμματισμός Υπολογιστών (Εργαστήριο)	2Ε
4.	Δίκτυα Υπολογιστών (Εργαστήριο)	2Ε
5.	Πληροφοριακά Συστήματα σε Επιχειρήσεις και Οργανισμούς	2Θ
6.	Συστήματα Διαχείρισης Βάσεων Δεδομένων και Εφαρμογές τους στο Διαδίκτυο	4Ε
7.	Ειδικά θέματα στον Προγραμματισμό Υπολογιστών	4Ε
8.	Σχεδιασμός και Ανάπτυξη Διαδικτυακών Εφαρμογών	3Ε
	Σύνολο	23 ώρες

Πίνακας 18 : Μαθήματα Ειδικότητας Τεχνικός Εφαρμογών Πληροφορικής - Γ' τάξη ΕΠΑ.Λ

Μια πρώτη παρατήρηση που έχουμε να κάνουμε είναι η απουσία από το πρόγραμμα σπουδών ενός μαθήματος που να άπτεται ή να συγγενεύει με το αντικείμενο της ρομποτικής. Θεωρούμε ότι αυτό είναι μία παράλειψη καθώς ούτε στα μαθήματα του Τομέα αλλά ούτε και σε αυτά των δύο ειδικοτήτων υπάρχει κάποια σχετικό μάθημα για ένα αντικείμενο που εξάπτει το ενδιαφέρον των μαθητών και προάγει τόσο την εκπαιδευτική όσο και την τεχνική τους κατάρτιση καθώς συνδυάζει γνώσεις και αντικείμενα από πολλές γνωστικές περιοχές. Αυτό βέβαια δε σημαίνει σε καμία περίπτωση ότι αν υπήρχε ένα τέτοιο μάθημα η πλατφόρμα Duckietown θα ήταν κατάλληλη για να το υποστηρίξει. Απλά επειδή στα πλαίσια των μαθημάτων του προγραμματισμού τόσο στην Β' τάξη όσο και στην Γ' τάξη διδάσκεται η γλώσσα προγραμματισμού Python - μια ευρέως διαδομένη, σύγχρονη με πολλές και πλούσιες βιβλιοθήκες γλώσσα - θα ήταν θεμιτό να υπήρχε και κάποιο μάθημα που σε συνδυασμό με μια υπολογιστική πλατφόρμα (πχ Raspberry Pi) η οποία να "προγραμματίζεται" σε Python οι μαθητές να είχαν τη δυνατότητα να υλοποιούν ενδιαφέρουσες κατασκευές - εφαρμογές έτσι ώστε να διαπίστωναν στην πράξη τη χρησιμότητα της γλώσσα προγραμματισμού που διδάσκονται και να είχαν μία πρώτη επαφή με αυτό που ονομάζουμε ενσωματωμένα συστήματα.

Μάθημα Ρομποτική συναντάμε στον Τομέα Ηλεκτρολογίας, Ηλεκτρονικής και Αυτοματισμού. Σύμφωνα με τα οριζόμενα στο Φ.Ε.Κ. 1489/τ. Β'/26-5-2016, από το σχολικό έτος 2017-2018, στον **Τομέα Ηλεκτρολογίας, Ηλεκτρονικής και Αυτοματισμού** στη Γ' τάξη των Ημερησίων και Εσπερινών ΕΠΑ.Λ., προβλέπεται να λειτουργούν οι εξής ειδικότητες:

- Τεχνικός Ηλεκτρονικών και Υπολογιστικών Συστημάτων, Εγκαταστάσεων, Δικτύων και Τηλεπικοινωνιών
- Τεχνικός Ηλεκτρολογικών Συστημάτων, Εγκαταστάσεων και Δικτύων
- Τεχνικός Αυτοματισμού
- Από αυτές τις τρεις (3) ειδικότητες σύμφωνα με το Φ.Ε.Κ. 1426, τ.Β', 26-4-2017 τα **μαθήματα της ειδικότητας** Τεχνικός Ηλεκτρονικών και Υπολογιστικών Συστημάτων, Εγκαταστάσεων, Δικτύων και Τηλεπικοινωνιών

που παρακολουθούν οι μαθητές και μαθήτριες του **Τομέα Ηλεκτρολογίας, Ηλεκτρονικής και Αυτοματισμού** στη Γ' τάξη του Ημερησίου ΕΠΑ.Λ. έχουν ως εξής:

Ειδικότητα: **Τεχνικός Ηλεκτρονικών και Υπολογιστικών Συστημάτων, Εγκαταστάσεων, Δικτύων και Τηλεπικοινωνιών**		
Γ' τάξη		
	Μαθήματα	**Ώρες**
1.	Ψηφιακά Συστήματα	3Θ
2.	Δίκτυα Υπολογιστών	3Θ
3.	Εφαρμοσμένα Ηλεκτρονικά - Κατασκευές	2Ε
4.	Εγκατάσταση και Διαχείριση Δικτύων - Συντήρηση Υπολογιστικών Συστημάτων	3Ε
5.	Συστήματα Ελέγχου και Ασφάλειας	2Ε
6.	Τηλεπικοινωνίες - Τηλεματική	3Θ+2Ε
7.	Ρομποτική	3Ε
8.	Επεξεργασία Σήματος Ήχου και Εικόνας	2Ε
	Σύνολο	23 ώρες

Πίνακας 19 : Μαθήματα ειδικότητας Τεχνικός Ηλεκτρονικών και Υπολογιστικών Συστημάτων, Εγκαταστάσεων, Δικτύων και Τηλεπικοινωνιών Γ' Τάξη ΕΠΑ.Λ

Σκοπός του μαθήματος είναι να εισάγει τους μαθητές στα συστήματα σύγχρονων αυτοματισμών «ευφυούς» τεχνολογίας, που συνδυάζουν την ηλεκτρονική, τον προγραμματισμό και την μηχανολογία. Για να την κάλυψη των διδακτικών στόχων προτείνονται δραστηριότητες με την αναπτυξιακή πλατφόρμα του Arduino ή εναλλακτικά με τον μικροελεγκτή PIC16F877. Το βοηθητικό εκπαιδευτικό υλικό περιλαμβάνει τα ακόλουθα προτεινόμενα συγγράμματα

- Εφαρμογές Arduino - Σεμινάριο Ηλεκτρονικού Τομέα http://users.sch.gr/asal1/material/seminaria/teliko24_1.pdf
- Σεμινάριο Ηλεκτρονικών 3-4-2014 - Πρακτικές εφαρμογές με μικροελεγκτή http://users.sch.gr/asal1/material/seminaria/NEASMYRNH/efarmoges_arduino%20%282%29.pdf
- Κατασκευάζω και προγραμματίζω με τον Arduino - Αριστείδης Παλιούρας http://robotics-edu.gr/data/arduino/arduino_paliouras.pdf
- Προγραμματίζοντας με τον μικροελεγκτή Arduino http://users.sch.gr/manpoul/docs/arduino/ProgrammingArduino.pdf

και μία σειρά ασκήσεων. Ενδεικτικά αναφέρουμε τις κάτωθι:

Ενδεικτικές δραστηριότητες - ασκήσεις με την πλατφόρμα Arduino	
1. Φωτεινός σηματοδότης ελέγχου	7. Συναγερμός ενεργοποιούμενος με ήχο
2. Ρύθμιση έντασης φωτεινότητας	8. Ανίχνευση απόστασης με αισθητήρα υπερήχων
3. Ενδείκτης LED ενός ψηφίου	9. Έλεγχος Σερβοκινητήρα
4. Ενδείκτης LED τεσσάρων ψηφίων	10. Έλεγχος βηματικού κινητήρα
5. LED MATRIX	11. Ρομποτικός βραχίονας
6. Μετρητής θερμοκρασίας	12. Ρομποτικό όχημα.

Πίνακας 20 : Ενδεικτικές Δραστηριότητες

Συμπερασματικά αν υπήρχε κάπου χώρος στην δευτεροβάθμια εκπαίδευση για τη πλατφόρμα Duckietown υποθέταμε εξ αρχής ότι θα ήταν στα Επαγγελματικά Λύκεια. Κυρίως λόγω της διάρθρωσης του προγράμματος σπουδών δηλαδή την ύπαρξη και λειτουργία Τομέων και ειδικοτήτων σχετικών με το αντικείμενο της συγκεκριμένης πλατφόρμας, των πολλών διδακτικών ωρών που καταλαμβάνουν τα μαθήματα αυτά στο ωρολόγιο πρόγραμμα αλλά και εξαιτίας του εργαστηριακού προσανατολισμού των μαθημάτων. Αν συνυπολογίσουμε στα παραπάνω και την υποδομή που διαθέτουν τα Επαγγελματικά λύκεια για την υποστήριξη των εργαστηριακών μαθημάτων τότε είναι ευνόητη η αρχική μας υπόθεση. Ειδικά στον Τομέα πληροφορικής όπου διδάσκεται και η γλώσσα προγραμματισμού Python θα μπορούσε ίσως περισσότερο από κάθε άλλο τομέα να ενσωματωθεί η πλατφόρμα Duckietown. Όμως κάποια αναμφισβήτητα γεγονότα έρχονται να ανατρέψουν την αρχική μας υπόθεση.

Αρχικά το γεγονός ότι δεν προβλέπεται στο πρόγραμμα σπουδών του Τομέα Πληροφορικής όχι μόνο το μάθημα της ρομποτικής άλλα ούτε καν συγγενή μαθήματα όπως για παράδειγμα τα ενσωματωμένα συστήματα. Συνεπώς δεν υπάρχει κάποιο πλαίσιο με βάση το αναλυτικό πρόγραμμα σπουδών που θα μπορούσε να περιβάλλει ένα τέτοιο project.

Ο δεύτερος λόγος έχει να κάνει με τη δυσκολία του αντικειμένου που πραγματεύεται η πλατφόρμα Duckietown. Ασφαλώς οι μαθητές θα μπορούσαν με τη καθοδήγηση ενός εκπαιδευτικού να διαμορφώσουν την πόλη - πίστα όπως επίσης και να συναρμολογήσουν το Duckiebot. Και ναι, θα μπορούσαν να είχαν γνωρίσει το λειτουργικό σύστημα Ubuntu μέσα από το σχετικό μάθημα της Β' τάξης του τομέα ίσως και να είχαν εξοικειωθεί με κάποιες εντολές της γραμμής εντολών. Και φυσικά η εξοικείωση τους με τη γλώσσα προγραμματισμού Python θα βοηθούσε καθώς ολόκληρος ο φλοιός Duckietown Shell είναι γραμμένος σε Python. Από το σημείο αυτό και μετά όμως αρχίζουν τα δύσκολα. Θα έπρεπε να γνωρίσουν τη πλατφόρμα του Docker και να κατανοήσουν τον τρόπο λειτουργίας της. Να εξοικειωθούν με το ROS και τις έννοιες του και πως υλοποιεί την αρχιτεκτονική την αυτόνομης πλοήγησης μέσω των κόμβων του. Αλλά το δυσκολότερο κομμάτι είναι ο αισθητήρας με τον οποίο το Duckiebot αντιλαμβάνεται κάποια χαρακτηριστικά του εξωτερικού κόσμου καθώς αυτό άπτεται του κλάδου της υπολογιστικής όρασης. Η επεξεργασία των δεδομένων που εξάγει η κάμερα, κακά τα ψέματα, δεν είναι εύκολο αντικείμενο. Αρχικά απαιτείται η διαδικασία βαθμονόμησης της κάμερας όπως περιγράψαμε παραπάνω και έπειτα μία σειρά από μετασχηματισμούς και εφαρμογή φίλτρων προκειμένου να εξαχθεί χρήσιμη πληροφορία. Οι διαδικασίες αυτές (μετασχηματισμοί και φίλτρα) προϋποθέτουν άλλες πρότερες γνώσεις όπως για παράδειγμα τρόπος λειτουργίας της pin-hole κάμερας, συστήματα συντεταγμένων, βασικές γνώσεις μαθηματικών και γραμμικής άλγεβρας, ομογενείς συντεταγμένες, κτλ. Σε όλα αυτά οι μαθητές δεν θα είχαν το γνωστικό υπόβαθρο για να ανταποκριθούν. Μόνο αν γινόταν σκόπιμη απόκρυψη κάποιων λειτουργιών ή περιγραφή τους σε ένα πολύ γενικό, εισαγωγικό, επιφανειακό επίπεδο θα μπορούσε αυτό το project να "σταθεί" στην Δευτεροβάθμια εκπαίδευση. Μια άλλη ενδεχομένως πιο απλή μορφή του Duckietown project με λιγότερες λειτουργικότητες αλλά και πολυπλοκότητα, με πιο απλούς ως προς την επεξεργασία των δεδομένων που παράγουν αισθητήρες ίσως θα μπορούσε να ενσωματωθεί σε ένα αναμορφωμένο πρόγραμμα σπουδών που θα περιλάμβανε και τη ρομποτική.

Αποτίμηση έργου - Συμπεράσματα

Φτάνοντας στο τέλος μιας μακράς διαδρομής ήρθε ώρα του απολογισμού. Τι καταφέραμε να πραγματοποιήσουμε, ποια εμπόδια συναντήσαμε, ποιοι στόχοι έμειναν ανεκπλήρωτοι, τι οφέλη αποκομίσαμε. Αλλά σας ξεκινήσουμε από την αρχή.

Η ανάθεση της παρούσας πτυχιακής εργασίας έγινε περίπου τέλη Ιουνίου του 2019. Τους δύο μήνες του καλοκαιριού συλλέξαμε υλικό για την συγκεκριμένη πλατφόρμα κυρίως από τον επίσημο ιστότοπο της πλατφρόμας Duckietown.org. Υλικό που αφορούσε τις προδιαγραφές διαμόρφωσης της πόλης, οδηγίες συναρμολόγησης του Duckiebot αλλά και παρουσιάσεις που αφορούσαν τα πεδία της ρομποτικής τα οποία εμπλέκονται στο συγκεκριμένο project.

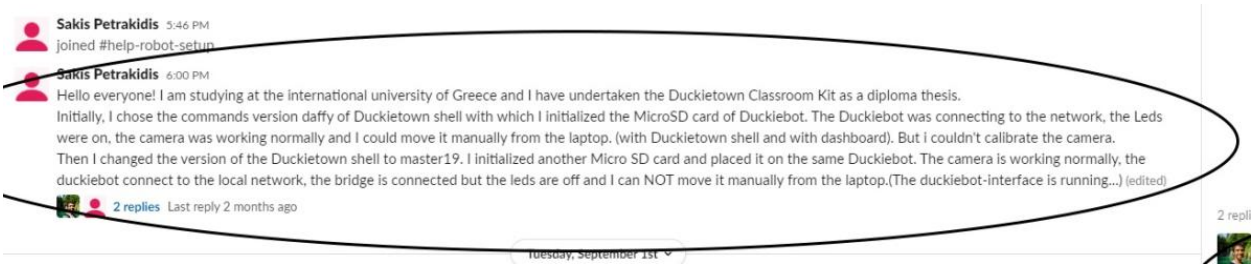
Το Σεπτέμβριο (του 2019) αρχίσαμε να συναρμολογούμε βήμα - βήμα το πρώτο μας Duckiebot παράλληλα με την εγκατάσταση των απαραίτητων λογισμικών στον φορητό μας υπολογιστή. Επιλέξαμε την έκδοση daffy του Duckietown shell , μάλλον από αβλεψία, καθώς η σταθερή (επίσημη) έκδοση είναι η Master 19. Αρχικοποιήσαμε με αυτή την έκδοση (daffy) την πρώτη μας Micro SD κάρτα. Τοποθετώντας την κάρτα στο πρώτο μας Duckiebot με το όνομα donald είδαμε ότι μπορεί να συνδεθεί στο wifi του εργαστηρίου, η κάμερα έστελνε τη ροή εικόνων στο φορητό μας υπολογιστή και λειτουργούσε κανονικά και ο τηλεχειρισμός του δηλαδή μπορούσαμε να το τηλε-κατευθύνουμε από το laptop και ήμασταν πολύ χαρούμενοι γι αυτό. Βέβαια η χαρά δεν κράτησε πολύ καθώς όταν εκκινούσαμε τη διαδικασία βαθμονόμησης της κάμερας εισπράτταμε μηνύματα σφάλματων που καταστούσαν αδύνατη την εκκίνηση της υπηρεσίας του ROS. Μετά την αρχικοποίηση και της δεύτερης Micro SD κάρτας για το δεύτερο Duckiebot που είχαμε κατασκευάσει με το αναγνωριστικό Huey διαπιστώσαμε ότι δεν λειτουργούσε η διαδικασία του τηλεχειρισμού. Έτσι μετά από μία πρώτη επαφή που είχαμε με τον υπεύθυνο καθηγητή Κ. Βολογιαννίδη Σταύρο αποφασίσαμε την αλλαγή την έκδοσης του Duckietown shell σε master19. Παράλληλα είχαμε ξεκινήσει και τη διαμόρφωση της πόλης. Μέχρι τα Χριστούγεννα λοιπόν είχαμε διαμορφώσει την πίστα - πόλη στο εργαστήριο ερευνών, είχαμε κατασκευάσαμε δύο Duckiebots και εγκαταστήσαμε όλα τα απαραίτητα λογισμικά στο φορητό υπολογιστή μας.

Από την αρχή του νέου έτους (2020) επιλέγοντας την έκδοση master19 αρχικοποιήσαμε εκ νέου δύο micro sd κάρτες και αρχίσαμε να δοκιμάζουμε τις λειτουργίες των Duckiebots. Με αυτή την αρχικοποίηση τα Duckiebots μπορούσαν να συνδεθούν ασύρματα στο υπάρχον τοπικό δίκτυο, η κάμερα λειτουργούσε κανονικά στέλνοντας τη ροή εικόνων στο laptop, αλλά τα Duckiebots δεν μπορούσαν να κινηθούν μέσω της λειτουργίας του τηλεχειρισμού από τον φορητό υπολογιστή. Κάναμε όλες τις προτεινόμενες ενέργειες για την επίλυση του προβλήματος (μέχρι και τη διαδικασία Reflash - Microcontroller) δίχως κανένα αποτέλεσμα. Παράλληλα συναρμολογήσαμε και τους φωτεινούς σηματοδότες με το παρατηρητήριο που ενσωματώνουν.

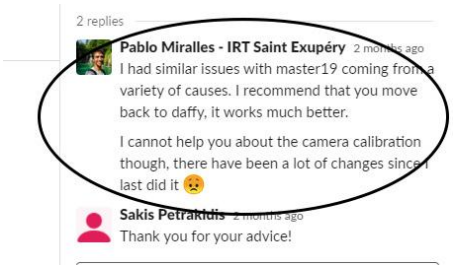
Το Δεκέμβριο του 2019 εμφανίστηκε στην Κίνα ένας ιός που προκλήθηκε από μετάλλαξη του ιού SARS-CoV-2 και αποδείχθηκε εξαιρετικά επικίνδυνος για άτομα με υποκείμενα

νοσήματα με σοβαρές επιπλοκές που έφταναν μέχρι την πνευμονία και το σύνδρομο οξείας αναπνευστικής δυσχέρειας προκαλώντας πολλούς θανάτους. Η εξαιρετικά υψηλή μεταδοτικότητα του ιού COVID-19 οδήγησε την γρήγορη εξάπλωση του και εξελίχτηκε σε πανδημία. Εξαιτίας των παραπάνω γεγονότων αποφασίστηκε στη χώρα μας γενικό lockdown τον Μάρτιο και τον Απρίλιο του 2020 στερώντας μας την πρόσβαση από το εργαστήριο ερευνών του τμήματος Μηχανικών Πληροφορικής, Υπολογιστών και Τηλεπικοινωνιών. Στο διάστημα αυτό ξεκίνησε η συγγραφή της παρούσας τεκμηρίωσης.

Μετά τη λήξη του lockdown συναρμολογήσαμε ακόμα δύο Duckiebot αλλά και "σπαταλήσαμε" ένα μεγάλο χρονικό διάστημα αναλύοντας και συγκρίνοντας τα docker containers που εκτελούνται στο Duckiebot donald (που αρχικοποιήθηκε με την έκδοση daffy και μπορούσε να κινηθεί) και σε αυτά που εκκινούσαν στον Duckiebot dewey (που αρχικοποιήθηκε με την έκδοση master19 και δεν λειτουργούσε ο τηλεχειρισμός). Φτάσαμε μέσα καλοκαιριού του 2020 να αναλύουμε μακροσκελή αρχεία εκκίνησης γραμμένα σε Python δυστυχώς χωρίς αποτέλεσμα. Έτσι φτάσαμε σε αδιέξοδο. Στο σημείο αυτό εκθέσαμε το πρόβλημα μας στην κοινότητα του Duckietown μέσω του slack.



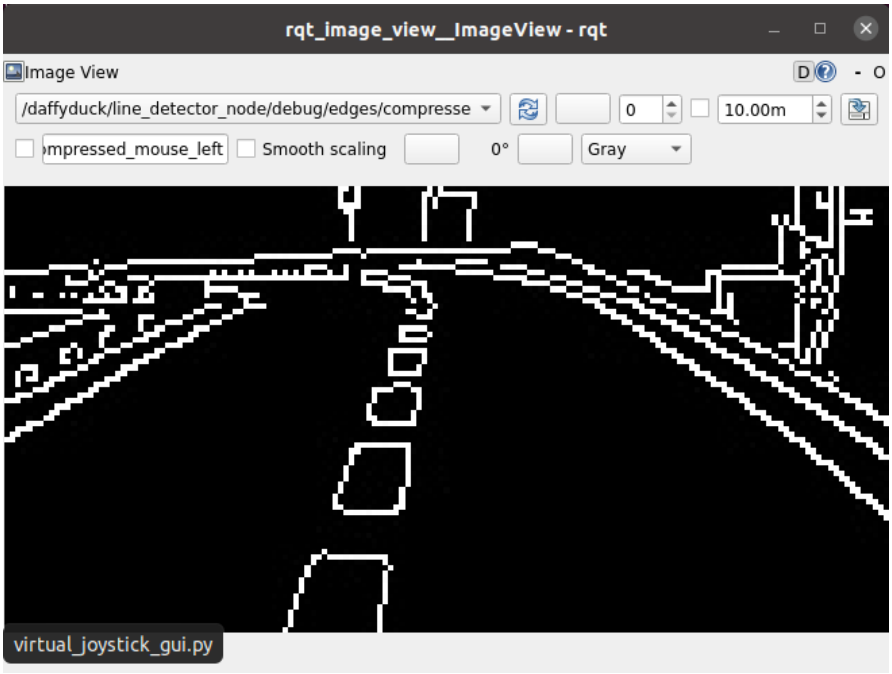
Εκεί κάποιοι πιο "έμπειροι" πάνω στο project από εμάς, μας έδωσαν πολύτιμές συμβουλές. Η βασικότερη από αυτές ήταν να μεταβούμε στην πειραματική έκδοση daffy του Duckietown shell που ενημερώνεται πιο συχνά και λειτουργεί καλύτερα.



Όταν επιχειρήσαμε να το κάνουμε αυτό μας περίμενε ακόμα μια δυσάρεστη έκπληξη καθώς μηνύματα σφάλματος καθιστούσαν αδύνατη στην εκκίνηση του dts. Έτσι αποφασίσαμε μετά από μία σύσκεψη με τον υπεύθυνο επιβλέπων καθηγητή να κάνουμε μία καθαρή εγκατάσταση από την αρχή όλων λογισμικών στο laptop συμπεριλαμβανομένου και του λειτουργικού συστήματος ακολουθώντας βήμα προς βήμα τις οδηγίες της πειραματικής έκδοσης (https://docs.duckietown.org/daffy/). Έτσι επιλέξαμε ως λειτουργικό σύστημα το **Ubuntu 20.04** και φυσικά ως έκδοση του **Duckietown Shell την daffy** και ως **Robot configuration** την **DB18**. Όλες αυτές οι ενέργειες όμως είχαν ως αποτέλεσμα μία πρόσθετη χρονική επιβάρυνση καθώς έπρεπε εκ νέου να αρχικοποιήσουμε το Duckiebots, να

ελέγξουμε τις βασικές τους λειτουργίες, να βαθμονομήσουμε την κάμερα και τους τροχούς, και ελέγξουμε στην πράξη τις προηγμένες συμπεριφορές τους.

Τον Σεπτέμβριο του 2020 ολοκληρώσαμε όλες τις προαπαιτούμενες ενέργειες στο Duckiebot daffyduck όπως, αρχικοποίηση κάρτας, έλεγχος σύνδεσης στο τοπικό δίκτυο, λειτουργία κάμερας, τηλεχειρισμός, βαθμονόμηση κάμερας και βαθμονόμηση τροχών. Και στις 12 Οκτωβρίου του 2020 το Duckiebot daffyduck πραγματοποίησε με επιτυχία τη συμπεριφορά Lane following στο εργαστήριο ερευνών του τμήματος Μηχανικών Πληροφορικής, Υπολογιστών και Τηλεπικοινωνιών.



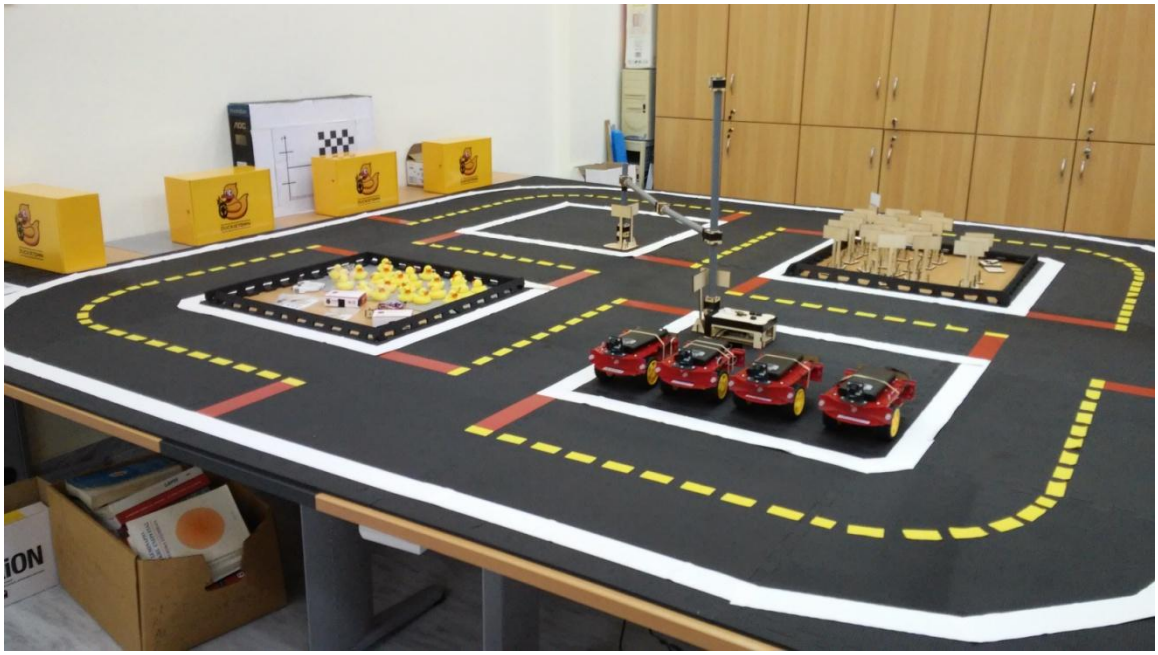
Εικόνα 173 : Ανίχνευση ακμών κατά τη λειτουργία lane following του Duckiebot

Δυστυχώς τον Νοέμβριο του 2020 αποφασίστηκε λόγω της έξαρσης του δεύτερου κύματος της πανδημίας και νέο lockdown το διαρκεί μέχρι και σήμερα που γράφονται αυτές οι γραμμές (1η Δεκεμβρίου 2020). Έτσι δεν στάθηκε δυνατό να ελέγξουμε στην πράξη τις υπόλοιπες προηγμένες συμπεριφορές του Duckiebot, ούτε κατορθώσαμε να δοκιμάσουμε τους φωτεινούς σηματοδότες στην πράξη. Καταφέραμε όμως αφού βαθμονομήσαμε τις κάμερες και τους τροχούς τους να θέσουμε σε λειτουργία Lane Following στην πόλη που κατασκευάσαμε και τα άλλα τρία (3) Duckiebots.

Αναγνωριστικό Duckiebot	Σύνδεση στο wifi	Λειτουργία κάμερας	Λειτουργία Τηλεχειρισμού	Βαθμονόμηση Κάμερα-Τροχών	Λειτουργία Lane-Following
daffyduck	✓	✓	✓	✓	✓
daisyduck	✓	✓	✓	✓	✓
donaldduck	✓	✓	✓	✓	✓
scroogemcduck	✓	✓	✓	✓	✓

Πίνακας 21: Πίνακας λειτουργιών Duckiebots

Σε κάθε περίπτωση κάναμε τα πρώτα βασικά βήματα, αποκτήσαμε πολύτιμες γνώσεις και εμπειρίες και θέσαμε τις βάσεις μιας πλατφόρμας που ενδείκνυται για έρευνα, ανακάλυψη και διερεύνηση πάνω σε αυτό που ονομάζουμε ρομποτική. Όλα αυτά που περιγράφει το κεφάλαιο 11 μπορούν να αποτελέσουν μία ενδιαφέρουσα επέκταση αυτού του project.



Εικόνα 174 : Η διαμορφωμένη πόλη και τα 4 Duckiebots στο Εργαστήριο Ερευνών

Η διδασκαλία και η εκμάθηση της Ρομποτικής είναι δύσκολη διότι είναι μία διεπιστημονική και ταχέως εξελισσόμενη "επιστήμη" η οποία ενσωματώνει υλικό και λογισμικό. Βασίζεται στην σύζευξη γνώσεων και αρχών της μηχανικής και της πληροφορικής ακουμπώντας όμως και σε άλλα επιστημονικά πεδία. Η συγκεκριμένη πλατφόρμα που αναπτύξαμε, είναι ευέλικτή, πολυμορφική και χωρίς να έχει υψηλό κόστος παρέχει ένα περιβάλλον μέσα στο οποίο μπορεί να ευδοκιμήσει η έμπρακτη εξοικείωση με τις βασικές αρχές της ρομποτικής. Δίνει τη δυνατότητα σε μαθητές ή φοιτητές να μάθουν πως να συναρμολογούν, να χειρίζονται και να προγραμματίζουν ένα αυτοκινούμενο όχημα (Duckiebot) σε ένα αστικό περιβάλλον (Duckietown) καθώς και να προσπαθούν να εφαρμόζουν νέες λειτουργικότητες και προηγμένες συμπεριφορές καθώς παρέχει ένα μεγάλο εύρος επεκτάσεων - προκλήσεων που αγγίζει το όρια της έρευνας. Στηρίζεται σε λογισμικά ανοικτού κώδικα τα οποία είναι διαθέσιμα σε αποθετήρια στο GitHub που σημαίνει ότι η πρόσβαση σε αυτά είτε για μελέτη, είτε για προσθήκες είτε για βελτιώσεις είναι ελεύθερη. Γενικά είναι μια πλατφόρμα που προάγει τη συνεργασία μέσα από ομάδες εργασίας και διαδικτυακές κοινότητες. Ιδανικά οι μαθητές ή φοιτητές που εμπλέκονται σε αυτό το project σχηματίζουν ομάδες και αποδέχονται κάποιο ρόλο μέσα σε αυτήν. Μαθαίνουν να επικοινωνούν, να συνεργάζονται και να πειραματίζονται όχι μόνο με τα υπόλοιπα μέλη της ομάδας αλλά και μέσω των κοινοτήτων του Duckietown και με άλλους σπουδαστές ή καθηγητές που ασχολούνται με τη συγκεκριμένη πλατφόρμα σε ολόκληρο τον κόσμο αποκομίζοντας τα μέγιστα δυνατά διδακτικά οφέλη σύμφωνα και με τις σύγχρονες θεωρίες μάθησης.

Η αντίληψη των Duckiebots , η αναγνώριση των χαρακτηριστικών του εξωτερικού κόσμου δηλαδή, βασίζεται στις προσλαμβάνουσες εικόνες της κάμερας, του μοναδικού αισθητήρα που διαθέτουν τα ρομποτικά αυτά οχήματα. Συνεπώς είναι μια πλατφόρμα που εστιάζει στον τομέα της υπολογιστικής όρασης. Το γεγονός αυτό την κάνει δυσπρόσιτη για μαθητές της πρωτοβάθμιας και δευτεροβάθμιας εκπαίδευσης καθώς απαιτεί πρότερες γνώσεις που δεν έχουν διδαχθεί και ενδέχεται να μην καταστεί εφικτή η εξοικείωση και η κατανόηση εννοιών βασικών για τη λειτουργία και τη συμπεριφορά των ρομποτικών οχημάτων. Ως εκ τούτου είναι ίσως κατάλληλη για μια πρώτη γνωριμία - επαφή των μαθητών της πρωτοβάθμιας αλλά και δευτεροβάθμιας εκπαίδευσης με τον κόσμο της ρομποτικής εξαιτίας του παιγνιώδη και ελκυστικού περιβάλλοντος που διαθέτει αλλά σίγουρα δεν ενδείκνυται για μαθητές αυτών των βαθμίδων για περεταίρω εμβάθυνση και μύηση του τρόπου με τον οποίο υλοποιεί τις προηγμένες αυτόνομες συμπεριφορές.

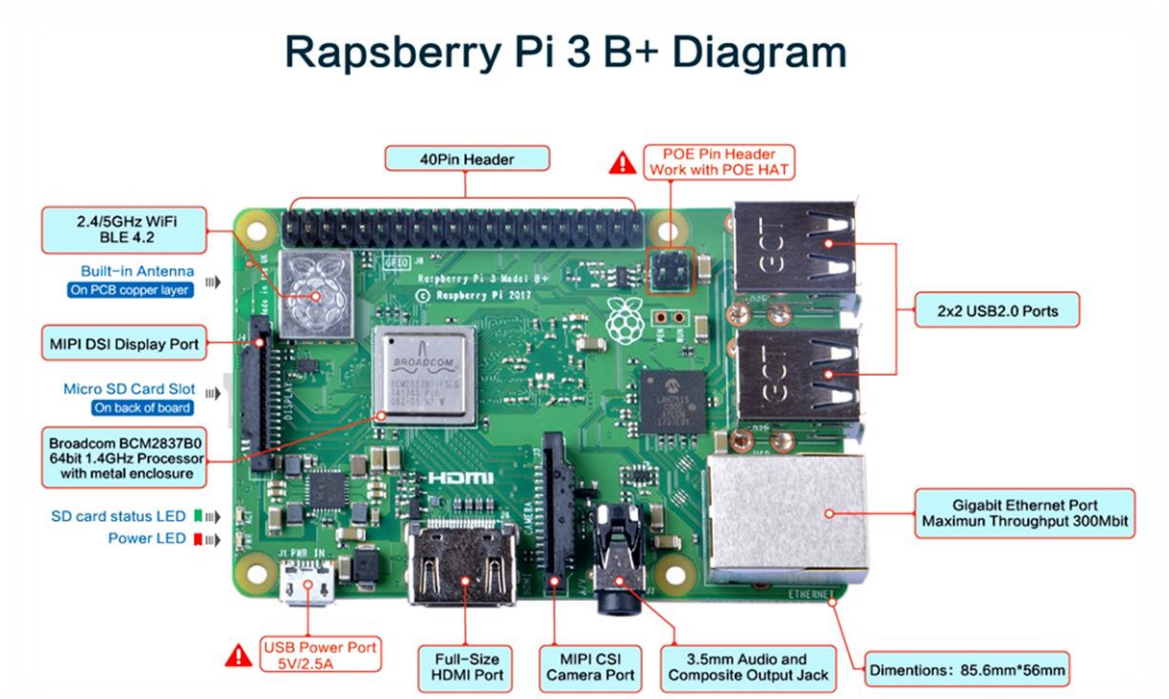
Με την σημερινή μορφή και τις λειτουργικότητες που ενσωματώνει είναι μία κατάλληλη πλατφόρμα για την τριτοβάθμια εκπαίδευση για την υποστήριξη μαθημάτων προπτυχιακού αλλά και μεταπτυχιακού επιπέδου πάνω σε αντικείμενα που άπτονται της ρομποτικής όρασης, τις αρχιτεκτονικές αυτόνομης πλοήγησης, τους αλγόριθμους ταυτόχρονης χαρτογράφησης και εντοπισμού (SLAM) ή αλγόριθμους βελτιστοποίησης κυκλοφορίας που μπορεί να επεκταθεί ακόμα και στην έρευνα πάντα μέσα από ομαδοσυνεργατικές διαδικασίες και πρακτικές.

Παραρτήματα

Παράρτημα Α

Συνοπτική περιγραφή του Raspberry Pi

Εμφανίστηκε για πρώτη φορά το Φεβρουάριο του 2012 στο Ηνωμένο Βασίλειο από το Raspberry Pi Foundation (μια μη κερδοσκοπική εταιρεία η οποία ιδρύθηκε το 2009) σε συνεργασία με το πανεπιστήμιο Cambridge με βασικό σκοπό να προωθήσει την διδασκαλία στις αρχές επιστήμης των υπολογιστών και του προγραμματισμού στα σχολεία. Το όνομα του προέκυψε από την παράδοση που υπήρχε παλιά στην ονοματολογία των μικροϋπολογιστών να παίρνουν ονόματα φρούτων όπως για παράδειγμα η Tangerine Computer Systems, η Apricot Computers και η Acorn. Η κατάληψη Pi είναι εξαιτίας της αρχικής πρόθεσης της εταιρείας να δημιουργήσει έναν υπολογιστή ο οποίος θα έτρεχε προγράμματα γραμμένα στη γλώσσα προγραμματισμού Python. Με την πάροδο των ετών και την ανάπτυξη της τεχνολογίας εξελίχτηκε και αυτό.



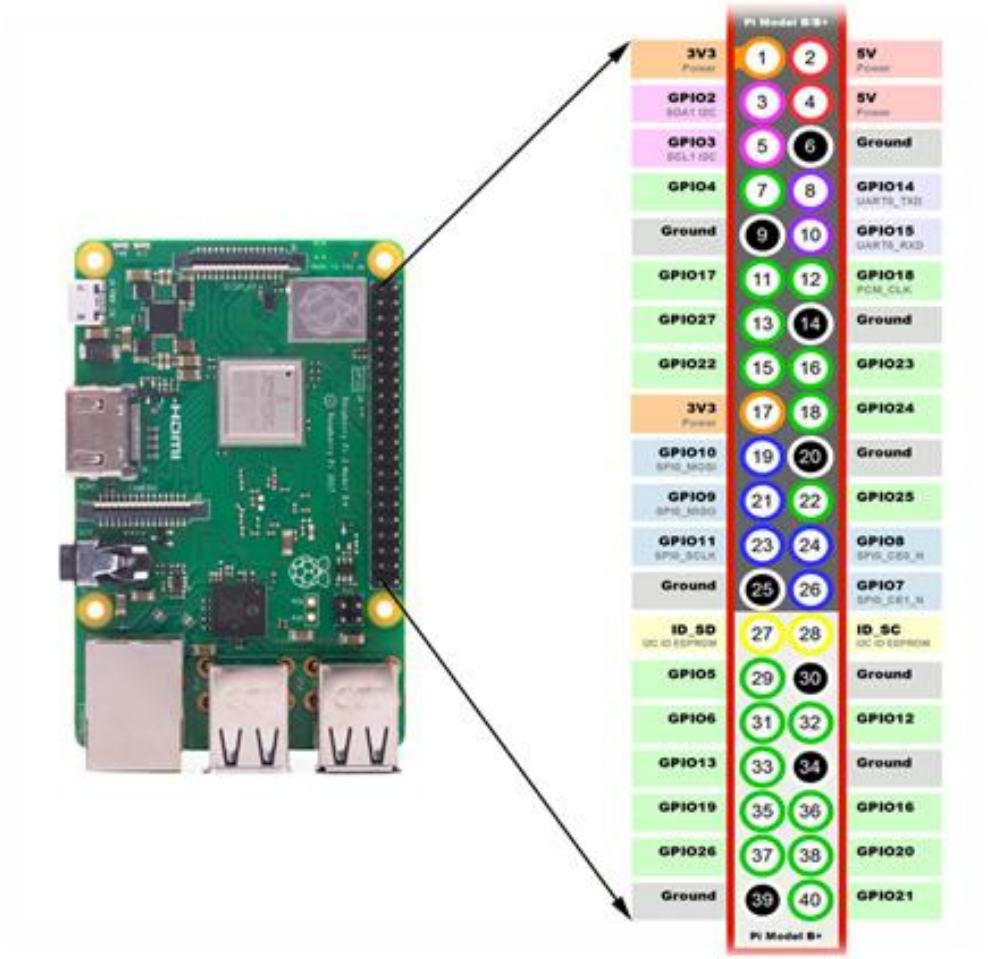
Εικόνα 175: Μητρική πλακέτα Raspberry Pi

.

Εμείς στην παρούσα εργασία χρησιμοποιήσαμε το Raspberry Pi 3 Model B+ και ο πίνακας παρουσιάζει τα βασικά χαρακτηριστικά του.

Πίνακας Τεχνικών Προδιαγραφών	
Επεξεργαστής	Broadcom BCM2837B0, Cortex-A53 64-bit SoC @ 1.4GHz
Μνήμη	1GB LPDDR2 SDRAM
Συνδεσιμότητα	2.4 GHz and 5 GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2BLE Gigabit Ethernet over USB 2.0 (maximum throughput 300Mbps) 4xUSB 2.0 ports
Προσβασιμότητα	Extended 40-pin GPIO header
Ήχος και Βίντεο	1xfull size HDM I MIPI DSI display port MIPI CSI camera port 4 pole stereo output and composite video port
Πολυμέσα	H.264, MPEG-4 decode (1080p30); H.264 encode (1080p30); OpenGL ES 1.1, 2.0 graphics
Κάρτα αποθήκευσης SD	Micro SD για φόρτωση λειτουργικού συστήματος και αποθήκευση δεδομένων.
Τροφοδοσία	5V/2.5 A DC μέσω της micro USB σύνδεσης 5V DC μέσω της GPIO Μέσω θύρας Ethernet - απαιτεί ξεχωριστή συσκευή PoE HAT
Συνθήκες Περιβάλλοντος	Θερμοκρασία λειτουργίας από 0-50^{o} C

Θα ήταν παράληψη αν δεν κάναμε ιδιαίτερη αναφορά και στους ακροδέκτες GPIO που διαθέτει το Raspberry Pi. Οι ακροδέκτες επιτρέπουν την επικοινωνία του Raspberry Pi με μια πληθώρα συσκευών και αισθητήρων ώστε να είναι σε θέση να λαμβάνει σήματα από τον πραγματικό κόσμο, να τα επεξεργάζεται με βάση το έργο που θέλουμε να επιτελεί το ενσωματωμένο σύστημα στο οποίο περιέχεται και να παράγει στις αντίστοιχες εξόδους. Αυτοί οι ακροδέκτες είναι που διαφοροποιούν το Raspberry Pi από έναν απλό υπολογιστή καθώς του επιτρέπουν την αλληλεπίδραση και το εξωτερικό περιβάλλον και το καθιστούν κατάλληλο για τη χρήση του σε ενσωματωμένα συστήματα.
Όπως βλέπουμε και στην Εικόνα 176 το Raspberry Pi 3 B+ διαθέτει συνολικά 40 ακροδέκτες GPIO οι οποίοι αριθμούνται φυσικά με βάση τη θέση τους στην πλακέτα αρχίζοντας από πάνω αριστερά και οι οποίοι διακρίνονται σε ακροδέκτες εισόδου / εξόδου γενικής χρήσης, παροχής τάσης (2 των 3.3 Volt και 2 των 5 Volt), γείωσης, ασύγχρονης επικοινωνίας UART, σειριακής επικοινωνίας SPI και I^2C.



Εικόνα 176 : Ακροδέκτες GPIO του Raspberry Pi 3 B+

Παράρτημα Β

Βασικές ενέργειες στο GitHub

1) Αντίγραφο αποθετηρίου (Fork repository)

Πρόκειται για τη δημιουργία αντίγραφου του αποθετηρίου μας ώστε να μπορούμε ελεύθερα να κάνουμε αλλαγές και να πειραματιζόμαστε σε αυτό (το αντίγραφο) χωρίς να υπάρχει ο κίνδυνος αυτές οι αλλαγές να επηρεάσουν το αρχικό, βασικό μας έργο. Η διαδικασία δημιουργίας αντιγράφου γίνεται από την ιστοσελίδα του GitHub ενεργοποιώντας την αντίστοιχη επιλογή 

στο πάνω δεξί μέρος της σελίδας.

2) Κλωνοποίηση αποθετηρίου

Όταν δημιουργούμε ένα αποθετήριο στο GitHub υπάρχει ως απομακρυσμένο αποθετήριο. Υπάρχει η δυνατότητα να κλωνοποιήσουμε το απομακρυσμένο αυτό αποθετήριο δημιουργώντας ένα αντίγραφο τοπικά, στον υπολογιστή μας. Έτσι μπορούμε να δουλεύουμε με το τοπικό αποθετήριο και να ενημερώνετε και το απομακρυσμένο μέσω της διαδικασίας του συγχρονισμού. Αυτή η δημιουργία τοπικού αντιγράφου μπορεί να γίνει με την εντολή:

$ git clone git@github.com: USERNAME/REPOSITORY.git

3) Δημιουργία branch

Στο Github έχουμε τη δυνατότητα να δημιουργήσουμε έναν κλάδο (branch) για να απομονώσουμε τις κάποιες εργασίες ανάπτυξης χωρίς να επηρεάσουμε άλλους κλάδους του αποθετηρίου. Κάθε αποθετήριο έχει ένα προεπιλεγμένο κλάδο (ή υποκατάστημα σε ελεύθερη μετάφραση) και μπορεί να έχει πολλούς άλλους κλάδους. Υπάρχει η δυνατότητα συγχώνευσης των κλάδων χρησιμοποιώντας ένα αίτημα έλξης (Pull request). Στη γραμμή εντολών η δημιουργία branch γίνεται με την εντολή:

$ git checkout -b branch-name

ενώ για να δούμε τους κλάδους στους οποίους δουλεύουμε μπορούμε να χρησιμοποιήσουμε τις εντολές:

$ git branch

$ git status

Η δεύτερη εντολή μας παρέχει περισσότερες πληροφορίες σχετικά με το ποια αρχεία έχουμε τροποποιήσει τα οποία είναι εν δυνάμει αρχεία προς δέσμευση.

4) Δέσμευση και ώθηση αλλαγών

Μετά την επεξεργασία ορισμένων αρχείων στο τοπικό αποθετήριο του υπολογιστή μας πρέπει να μεταφέρουμε τις αλλαγές στο απομακρυσμένο αποθετήριο. Για να γίνει αυτό

πρέπει πρώτα να κάνουμε έναν έλεγχο ώστε να βεβαιωθούμε ποια αρχεία έχουμε τροποποιήσει με την εντολή

$ git status

Τα αρχεία τα οποία έχουν αλλαγές εμφανίζονται με κόκκινο χρώμα. Πρόκειται για αρχεία που έχουν δεχθεί τροποποίηση αλλά δεν τα έχουμε προσθέσει για μελλοντική δέσμευση. Όταν επιθυμούμε να προωθήσουμε όλες τις αλλαγές που έχουμε κάνει ώστε αυτές να προστεθούν στη δέσμευση μας εκτελούμε την εντολή

$ git add --all

ενώ αν επιθυμούμε να προσθέσουμε μόνο μεμονωμένα αρχεία τότε θα πρέπει να καθορίσουμε το "μονοπάτι" που βρίσκεται το αρχείο μας όπως παρακάτω

$ git add file-path

η επόμενη ενέργεια είναι να προσθέσουμε ένα μήνυμα δέσμευσης ενημερώνοντας τους συνεργάτες μας για τις αλλαγές που έχουμε πραγματοποιήσει με την εντολή

$ git commit -m "commut-message"

Αν όλα τα παραπάνω κύλησαν ομαλά χωρίς προβλήματα είμαστε έτοιμοι να ωθήσουμε τις αλλαγές στον κλάδο εκτελώντας στη γραμμή εντολών την εντολή

$ git push origin branch-name

5) Ανάκτηση νέων braches

Αν νέα υποκαταστήματα (braches) έχουν ωθηθεί πρόσφατα στο αποθετήριο και δεν τα έχουμε διαθέσιμα μπορούμε να ανακτήσουμε την καινούργια εργασία των άλλων συνεργατών. Η διαδικασία της ανάκτησης (fetching) από ένα αποθετήριο αντλεί όλα τα νέα απομακρυσμένης παρακολούθησης υποκαταστήματα και τις ετικέτες τους χωρίς να συγχωνεύει αυτές τις αλλαγές στα δικά μας υποκαταστήματα (braches). Αν διαθέτουμε ένα τοπικό αποθετήριο με τη χρήση ενός απομακρυσμένου URL για να αντλήσουμε όλες τις καινούργιες πληροφορίες για το επιθυμητό έργο πληκτρολογούμε στη γραμμή εντολών

$ git fetch remotename

6) Συγχώνευση αλλαγών στο τοπικό branch

Η επιλογή της συγχώνευσης συνδυάζει τις τοπικές αλλαγές με τι αλλαγές που έγιναν από άλλους συνεργάτες. Συνήθως συγχωνεύουμε ένα υποκατάστημα απομακρυσμένης παρακολούθησης (δηλαδή ένα υποκατάστημα που βρίσκεται στο απομακρυσμένο αποθετήριο) με το τοπικό μας υποκατάστημα. Αυτή η διαδικασία πραγματοποιείται με την εντολή:

$ git merge remotename/branchname

7) Διαγραφή υποκαταστήματος

Για να διαγράψουμε ένα τοπικό υποκατάστημα (branch) πληκτρολογούμε την εντολή

$ git branch -d branch-name

εννοείται δεν είναι εφικτό να διαγράψουμε υποκατάστημα στο οποίο βρισκόμαστε μέσα (είναι σαν να θέλουμε να γκρεμίσουμε ένα σπίτι από το οποίο δεν έχουμε βγει έξω...). Ενώ τέλος με την εντολή

$ git push origin --delete branch-name

διαγράφουμε ένα απομακρυσμένο υποκατάστημα.

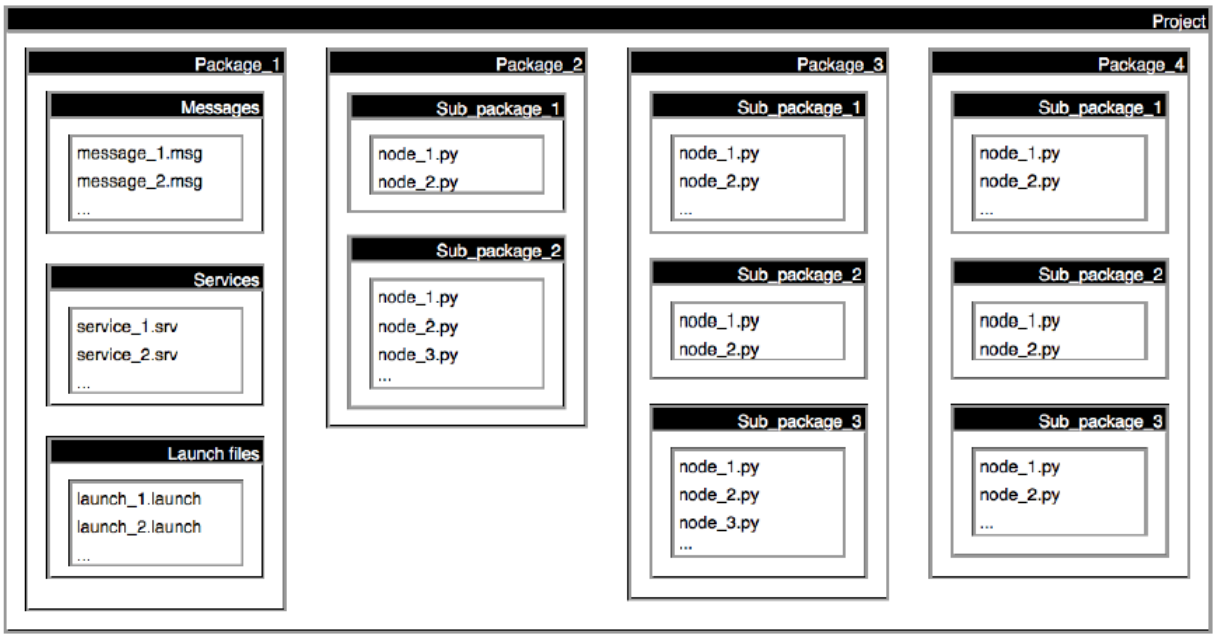
Παράρτημα Γ

Συνοπτική Περιγραφή του ROS

Το Robot Operating System αναπτύχθηκε αρχικά το 2007 στο Stanford Artificial Intelligence Laboratory και από το 2013 διοικείται από το Open Source Robotics Foundation και χρησιμοποιείται από πολλές εταιρείες και ιδρύματα.

Το ROS είναι μια πλατφόρμα ανοικτού κώδικα προσφέροντας μία συλλογή βιβλιοθηκών και εργαλείων που συντελούν στη δημιουργία ρομποτικών εφαρμογών. Αν και δεν αποτελεί λειτουργικό σύστημα παρέχει υπηρεσίες για έλεγχο χαμηλού επιπέδου σε συσκευές όπως μετάδοση μηνυμάτων και διαχείριση πακέτων. "Τρέχει" σε ένα υπάρχον λειτουργικό σύστημα όπως το GNU/Linux και οι διεργασίες που βασίζονται σε αυτό αναπαριστώνται σε μία αρχιτεκτονική γραφημάτων όπου η επεξεργασία λαμβάνει χώρα στους κόμβους (nodes) οι οποίοι μπορούν να λαμβάνουν και να δημοσιεύουν μηνύματα αισθητήρων, ελέγχου, κατάστασης, σχεδιασμού ή ενεργοποίησης. Η μικρή χρονική καθυστέρηση απόκρισης είναι συχνά κρίσιμη για τον έλεγχο και τη λειτουργία ενός ρομπότ. Ωστόσο το ROS δεν είναι ένα λειτουργικό πραγματικού χρόνου αλλά είναι δυνατό να ενσωματωθεί σε μονάδες που παρέχουν έναν εγγυημένο χρόνο απόκρισης όπως για παράδειγμα το Arduino. Στο δικό μας project όπου ο μοναδικός αισθητήρας που διαθέτουν τα Duckiebots είναι μία κάμερα δεν υπάρχει έντονα η ανάγκη για απόκριση σε πραγματικό χρόνο όποτε δεν αποτελεί πρόβλημα ότι το ROS δεν είναι εφικτό να μας εγγυηθεί τέτοιους χρόνους απόκρισης.

Η εξοικείωση με το ROS προϋποθέτει την γνωριμία με κάποιες βασικές και θεμελιώδης για τον τρόπο λειτουργίας του έννοιες που αφορούν είτε το επίπεδο του συστήματος των αρχείων του είτε το επίπεδο του γραφήματος των υπολογιστικών διαδικασιών.



Εικόνα 177: Λογική οργάνωση του συστήματος αρχείων του ROS

Οι βασικές αυτές έννοιες είναι οι εξής:

- **Πακέτα (packages) :** Ένα έργο στο ROS οργανώνεται σε πακέτα όπου κάθε πακέτο μπορεί να περιλαμβάνει διεργασίες χρόνου εκτέλεσης, βιβλιοθήκες του ROS, σύνολα δεδομένων, αρχεία διαμόρφωσης, λογισμικά τρίτων ή οτιδήποτε άλλο μπορεί να αποτελεί μία χρήσιμη λειτουργική μονάδα [8]. Στόχος αυτής της οργάνωσης σε πακέτα είναι η παροχή μιας εύχρηστης λειτουργικότητας και μιας εύκολης επαναχρησιμοποίησης του λογισμικού. Τα πακέτα δημιουργούνται εύκολα είτε "με το χέρι" είτε με εργαλεία όπως το catkin_create_pkg.
- **Κόμβοι (Nodes):** Πρόκειται για διεργασίες που πραγματοποιούν υπολογισμούς. Το ROS είναι σχεδιασμένο ώστε να είναι αρθρωτό. Ένα ρομποτικό σύστημα συνήθως περιλαμβάνει πολλούς κόμβους. Για παράδειγμα ένας κόμβος μπορεί να ελέγχει έναν αισθητήρα, ένας άλλος να σχεδιάζει τη τροχιά της κίνησης και άλλος να ελέγχει τους κινητήρες των τροχών. Ο κάθε κόμβος περιγράφεται να τη χρήση βιβλιοθηκών του ROS όπως rospy και roscpp.
- **Μηνύματα (messages) :** Οι κόμβοι επικοινωνούν μεταξύ τους ανταλλάσοντας μηνύματα. Το ROS χρησιμοποιεί μια απλοποιημένη γλώσσα περιγραφής μηνυμάτων (messages) για την αναπαράσταση των τιμών δεδομένων που δημοσιεύουν οι κόμβοι. Αυτός ο τρόπος περιγραφής διευκολύνει τα εργαλεία του ROS να δημιουργούν αυτόματα πηγαίο κώδικα για κάθε τύπο μηνύματος. Οι περιγραφές των μηνυμάτων αποθηκεύονται σε αρχεία με επέκταση .msg στον υποκατάλογο msg ενός πακέτου ROS [9].
- **Θέματα (Topics)** : Τα μηνύματα δρομολογούνται μέσω ενός συστήματος μεταφοράς με σημασιολογία δημοσίευσης/εγγραφής. Ένας κόμβος στέλνει ένα μήνυμα δημοσιεύοντας το σε ένα συγκεκριμένο κανάλι. Το κανάλι αυτό διαθέτει ένα όνομα ώστε να αναγνωρίζεται το περιεχόμενο του μηνύματος. Ένας κόμβος τον οποίο αφορούν συγκεκριμένου είδους δεδομένα θα εγγραφεί το αντίστοιχό θέμα-κανάλι. Είναι δυνατόν να υπάρχουν ταυτόχρονα πολλοί κόμβοι που δημοσιεύουν σε ένα topic. Επίσης μπορεί ο ίδιος κόμβος να δημοσιεύει ή να εγγεγραμμένος σε πολλά κανάλια.
- **Υπηρεσίες (services)** : Το μοντέλο δημοσίευσης / εγγραφής είναι ένα πολύ ευέλικτο πρότυπο επικοινωνίας αλλά η μεταφορά μονής κατεύθυνσης δεν είναι κατάλληλη για αλληλεπιδράσεις του τύπου ερώτηση / απάντηση οι οποίες απαιτούνται σε ένα κατανεμημένο σύστημα. Αυτού του τύπου οι αλληλεπιδράσεις ικανοποιούνται μέσω των υπηρεσιών. Μια υπηρεσία καθορίζεται από ένα ζευγάρι μηνυμάτων: ένα για το αίτημα ένα για την απάντηση. Ένας κόμβος παροχής ROS προσφέρει μια υπηρεσία με κάποιο όνομα και ένας πελάτης μέσω της υπηρεσίας αυτής στέλνει μήνυμα ερώτησης και αναμένει μήνυμα απάντησης. Οι βιβλιοθήκες πελατών παρουσιάζουν συνήθως αυτή την αλληλεπίδραση στον προγραμματιστή σαν να ήταν κλήση απομακρυσμένης διαδικασίας. Οι υπηρεσίες ορίζονται χρησιμοποιώντας αρχεία srv, τα οποία συγκεντρώνονται στον πηγαίο κώδικα από μια βιβλιοθήκη πελατών ROS [10]

- **ROS Master :** Παρέχει πληροφορίες εύρεσης μεταξύ των κόμβων λειτουργώντας όπως ένας DNS διακομιστής. Οι κόμβοι που έχουν εγγραφεί σε ένα θέμα θα αναζητήσουν σύνδεση από κόμβους που δημοσιεύουν σε αυτό το θέμα.

Κλείνοντας αυτή τη σύντομη αναφορά μας στο ROS να επισημάνουμε ότι στόχος του είναι να υποστηρίξει την επαναχρησιμοποίηση κώδικα στην ερεύνα και στην ανάπτυξη της ρομποτικής και όχι απλά να προσφέρει περισσότερες λειτουργίες και δυνατότητες. Το ROS διαθέτει ένα κατανεμημένο πλαίσιο διαδικασιών μέσω των κόμβων (nodes) το οποίο επιτρέπει την δημιουργία ξεχωριστών εκτελέσιμων προγραμμάτων καθώς και τη σύνδεση τους κατά το χρόνο εκτέλεσης. Αυτές οι διαδικασίες μπορούν να ομαδοποιηθούν σε Packages και Stacks. Τέλος υποστηρίζει να "κοινό" αποθετήριο κώδικα που επιτρέπει τη κοινή συνεργασία πολλών ομάδων [12] γεγονός που προάγει την έρευνα και συμβάλει στην ταχύτερη επέκταση, βελτίωση και εμπλουτισμό των λειτουργιών του.

Παράρτημα Δ

Κινηματική μοντελοποίηση του Duckiebot

Ο προσδιορισμός ενός μαθηματικού κινηματικού μοντέλου εφαρμογής στο Duckiebot είναι σημαντικός για τόσο για την κατανόηση της συμπεριφοράς του όσο και για τον σχεδιασμό ενός ελεγκτή ώστε να δύναται να αποκτήσει δυναμικά τις επιθυμητές συμπεριφορές και επιδόσεις [50].

Υπάρχουν πολλοί τρόποι για να προσδώσουμε το χαρακτηριστικό της κίνησης σε ένα ρομπότ. Ο συχνότερος μηχανισμός κίνησης είναι οι τροχοί, μιας και προσφέρουν ευκολία στο χειρισμό τους αλλά και απλότητα στη συνολική κατασκευή του. Για να κινηθεί ένα ρομπότ με τροχούς υπάρχουν δύο τρόποι: Είτε θα έχει ταυτόχρονη κίνηση και των δυο τροχών είτε θα μπορεί να κινεί ανεξάρτητα τον καθένα. Στην πρώτη περίπτωση χρειαζόμαστε ένα επιπλέον σύστημα για τη στρέψη των τροχών δηλαδή την αλλαγή της κατεύθυνσης του οχήματος, ενώ στη δεύτερη περίπτωση η αλλαγή της κατεύθυνσης επιτυγχάνεται με αυξομείωση της ταχύτητας ή αλλαγή της κατεύθυνσης περιστροφής των τροχών.

Ο δεύτερος τρόπος ονομάζεται διαφορική οδήγηση (differential drive) και είναι ένας από τους πιο ευρέως διαδεδομένους, απλούς και αξιόπιστους τρόπους κίνησης ρομποτικών κατασκευών ειδικά μικρού μεγέθους και είναι αυτός που έχει επιλεγεί για την κίνηση των Duckiebots. Η διαφορική οδήγηση χαρακτηρίζεται από την ανεξάρτητη κίνηση των δυο τροχών που όμως έχουν κοινό και σταθερό άξονα περιστροφής. Τα οχήματα διαφορικής οδήγησης έχουν δυο βαθμούς ελευθερίας μιας και η κίνηση τους γίνεται μόνο στο επίπεδο.

Το Duckiebot χρησιμοποιεί έναν ενεργοποιητή (κινητήρα DC) για κάθε τροχό. Εφαρμόζοντες διαφορετικός ροπές στους τροχούς το Duckiebot μπορεί να στρίψει δεξιά ή αριστερά, να κινηθεί ευθεία ή να παραμείνει ακίνητο. Στην ενότητα αυτή θα αναλύσουμε το μοντέλο διαφορικής κίνησης ενός ρομπότ. Το μοντέλο αυτό θα λαμβάνει τάση ως είσοδο (για τους κινητήρες των τροχών) και παράγει μία διαμόρφωση ή μία πόζα ως έξοδο. Η πόζα περιγράφει τη θέση και τον προσανατολισμό του Duckiebot σε ένα παγκόσμιο πλαίσιο αναφοράς.

Μπορούμε να ακολουθήσουμε πολλές μεθόδους για να περιγράψουμε το μοντέλο του Duckiebot όπως η μηχανική Lagrangian ή οι εξισώσεις Newton-Euler που περιγράφουν συνδυασμένη δυναμική μετάφρασης και περιστροφής ενός άκαμπτου σώματος. Εμείς θα επιλέξουμε τη δεύτερη καθώς αναμφισβήτητα μας παρέχει μία σαφέστερη φυσική εικόνα.

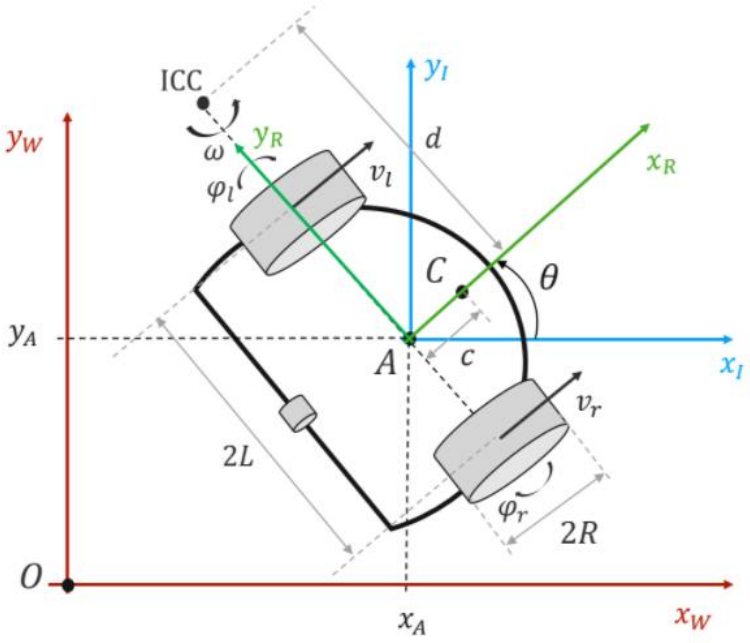
Προκαταρκτικά

Αρχικά είναι σημαντικό να σημειωθεί ότι θα περιορίσουμε την τρέχουσα ανάλυση στις δύο διαστάσεις δηλαδή στο επίπεδο. Για να περιγράψουμε την κινηματική συμπεριφορά του Duckiebot θα χρησιμοποιήσουμε τρία πλαίσια αναφοράς:

- Ένα "παγκόσμιο" πλαίσιο αναφοράς του οποίου την αρχή την συμβολίζουμε με το σημείο Ο. Οι μεταβλητές που θα αναφέρονται σε αυτό θα έχουν την ένδειξη W.
- Ένα "αδρανειακό" πλαίσιο, δηλαδή ένα σταθερό σύστημα αναφοράς παράλληλο με το "παγκόσμιο" που εκτείνεται στο επίπεδο στο οποίο κινείται το Duckiebot. Θα δηλώσουμε τους άξονες του ως $\{X_I,Y_I\}$ και θα έχει ως αρχή το σημείο $A=(x_A,y_A)$ που βρίσκεται στη μέση του νοητού άξονα που ενώνει τους τροχούς του Duckiebot. Οι μεταβλητές αυτού του πλαισίου αναφοράς θα έχουν την ένδειξη I.
- Ένα τοπικό πλαίσιο αναφοράς σταθερό πάνω στο σώμα του ρομπότ με αρχή το σημείο Α, το ίδιο σημείο με το αδρανειακό πλαίσιο. Σε αυτό όμως το σύστημα ο άξονας x δείχνει προς στην κατεύθυνση του μπροστινού μέρους του ρομπότ και ο άξονα y βρίσκεται κατά μήκος του άξονα μεταξύ των τροχών έτσι ώστε να σχηματιστεί ένα σύστημα αναφοράς δεξιού χεριού. Δηλώνουμε το πλαίσιο αναφοράς του robot με $\{X_R,Y_R\}$. Οι μεταβλητές αυτού του συστήματος θα έχουν την ένδειξη R.

θεωρούμε ότι:

- οι τροχοί έχουν ακτίνα R.
- Το ρομπότ θεωρείται ότι είναι άκαμπτο σώμα, συμμετρικό
- ο άξονας X_r συμπίπτει με τον άξονα συμμετρίας
- οι τροχοί θεωρούνται πανομοιότυποι και ότι ισαπέχουν απόσταση L από το κέντρο του άξονα Α.
- Το κέντρο της μάζας Cw=(xC,yC) του ρομπότ είναι πάνω στον άξονα x_r σε απόσταση c από το σημείο Α.
- Ο άξονα Xr σχηματίζει μία γωνία προσανατολισμού θ με το τοπικό οριζόντιο πλαίσιο αναφοράς.



Εικόνα 178: Σχήμα μοντελοποίησης ρομπότ διαφορικής οδήγησης

Μετασχηματισμοί πλαισίων αναφοράς

Έστω $x^I=[x^I,y^I]$ ένα διάνυσμα που αντιπροσωπεύει το αδρανειακό πλαίσιο αναφοράς και $X^I=[x^I,y^I,1]^T$ είναι η επαυξημένη έκδοση του. Είναι δυνατή η έκφραση ενός τέτοιου διανύσματος στο παγκόσμιο πλαίσιο αναφοράς μέσω μιας μετατόπισης:

$$\mathbf{X^W=TX^I} \quad \text{(σχέση 1)}$$

όπου T ένας πίνακας μετατόπισης ορισμένος ως εξής:

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & x_A \\ 0 & 1 & y_A \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^I \\ y^I \\ 1 \end{bmatrix}$$

Στον ευκλείδιο χώρο κάθε μετασχηματισμός διατηρεί του κανόνες ισομετρίας. Έτσι για παράδειγμα μια ταχύτητα που εκφράζεται στο αδρανειακό πλαίσιο ια έχει το ίδιο μέγεθος με αυτήν που εκφράζεται στο παγκόσμιο σύστημα αναφοράς.

Έστω $x^R=[x^R,y^R]^T$ ένα διάνυσμα του τοπικού πλαισίου αναφοράς του robot και $X^R=[x^R,y^R,1]^T$ μία επαυξημένη μορφή του. Είναι εφικτό τα εκφράσουμε ένα τέτοιο διάνυσμα στο αδρανειακό σύστημα αναφοράς μέσω μιας περιστροφής:

$$\mathbf{X^I=R}(\theta)\mathbf{X^R} \quad \text{(σχέση 2)}$$

όπου R(θ) ένας ορθογώνιος πίνακας περιστροφής :

$$\mathbf{R}(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{(σχέση 3)}$$

Στο σημείο αυτό να σημειώσουμε ότι ισχύει : $R^T(\theta)R(\theta)=R(\theta)R^T(\theta)=I$ και συνεπώς :

$$\mathbf{R^T}(\theta)=\mathbf{R^{-1}}(\theta)$$

Ένα επακόλουθο των σχέσεων 1 και 2 είναι η μετατόπιση και η περιστροφή να μπορούν να συνδυαστούν σε έναν μόνο μετασχηματισμό ως εξής: $\mathbf{X^W=TX^I=TR(\theta)X^R}$ όπου :

$$\mathbf{TR}(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & x_A \\ sin\theta & \cos\theta & y_A \\ 0 & 0 & 1 \end{bmatrix} \quad \text{(σχέση 4)}$$

Δυναμική

Κινηματική είναι ο κλάδος της μηχανικής που μελετά την κίνηση των στερεών σωμάτων χωρίς να λαμβάνει υπόψη τη μάζα τους ή τις δυνάμεις που προκαλούν αυτή την κίνηση. Όταν αναφερόμαστε σε κινητές ρομποτικές κατασκευές τότε ο όρος κινηματική ανάλυση περιγράφει τη γεωμετρία της κίνησης αυτών κατασκευών ως προς ένα σταθερό σύστημα

συντεταγμένων χωρίς να γίνεται αναφορά στις δυνάμεις-ροπές που προκαλούν την κίνηση τους. Η δυναμική μοντελοποίηση όμως λαμβάνει υπόψη της την πραγματική διαμόρφωση του ρομποτικού συστήματος. Η κίνηση της μάζας του ρομπότ είναι αποτέλεσμα της ισορροπίας των εξωτερικών δυνάμεων και των ροπών με αδράνεια. Είναι δυνατό να χρησιμοποιηθούν δύο διαφορετικές προσεγγίσεις για να εξαχθούν οι εξισώσεις περιγραφής της κίνησης του Ρομπότ. Η προσέγγιση Lagrangian που βασίζεται σε ενεργειακές εκτιμήσεις και η προσέγγιση Newtonian που στηρίζεται στην ισορροπία γενικευμένων δυνάμεων. Εμείς επιλέγουμε να ακολουθήσουμε την προσέγγιση Newtonian που μας προσφέρει μία πιο φυσική περιγραφή του προβλήματος. Προφανώς και οι δύο μέθοδοι οδηγούν στα ίδια αποτελέσματα όταν γίνονται οι ίδιες υποθέσεις.

Στο σημείο αυτό κρίνεται σκόπιμο να συμφωνήσουμε σε κάποιους συμβολισμούς οι οποίοι πρόκειται να χρησιμοποιηθούν στις κινηματικές εξισώσεις:

- ✓ C^r =(c,0) : είναι το κέντρο μάζας του ρομπότ
- ✓ (u_u, u_w) : Διαμήκεις και πλευρικές ταχύτητες του πλαισίου αναφοράς C του ρομπότ
- ✓ (Fu_R, Fu_L) : Διαμήκεις δυνάμεις που ασκούνται στο όχημα από τον δεξιό και αριστερό τροχό
- ✓ (Fw_R, Fw_L) : Πλευρικές δυνάμεις που ασκούνται στο όχημα από τον δεξιό και αριστερό τροχό
- ✓ (T_R, T_L) : Ροπές που δρουν στον δεξιό και αριστερό τροχό
- ✓ $\theta, \omega = \dot{\theta}$: Προσανατολισμός οχήματος και γωνιακή ταχύτητα
- ✓ M : Μάζα του οχήματος

Καλό είναι επίσης να θυμηθούμε και ορισμένα στοιχεία της κινητικής των πολικών συντεταγμένων. Έστω r(t) ένα σημείο του χώρου στο αδρανειακό πλαίσιο αναφοράς σε απόσταση r(t) από το σημείο A. Τότε ισχύουν οι σχέσεις με βάση τον τύπο Euler :

$$\begin{aligned}
\mathbf{r}(t) &= r(t)e^{j\theta(t)} \\
\dot{\mathbf{r}}(t) &= v_u(t)e^{j\theta(t)} + v_w(t)e^{j(\theta(t)+\frac{\pi}{2})} \\
\ddot{\mathbf{r}}(t) &= a_u(t)e^{j\theta(t)} + a_w(t)e^{j(\theta(t)+\frac{\pi}{2})},
\end{aligned} \quad \text{(σχέσεις 5)}$$

με

$$\begin{aligned}
v_u(t) &= \dot{r}(t) \\
v_w(t) &= r(t)\dot{\theta}(t) \\
a_u(t) &= \dot{v}_u - v_w\dot{\theta}(t) = \ddot{r}(t) - r\dot{\theta}^2(t) \\
a_w(t) &= \dot{v}_w + v_u\dot{\theta}(t) = 2\dot{r}(t)\dot{\theta}(t) + r(t)\ddot{\theta}(t).
\end{aligned} \quad \text{(σχέσεις 6)}$$

Έχοντας τις εξισώσεις 5 και 6 στο μυαλό μας είναι χρήσιμο να σημειωθεί για μελλοντική χρήση ότι ο προσδιορισμός της θέσης το κέντρου της μάζας C στο αδρανειακό πλαίσιο είναι

$$\begin{cases} x_C^W(t) &= x_A(t) + r(t)\cos\theta(t) \\ y_C^W(t) &= y_A(t) + r(t)\sin\theta(t) \end{cases}, \quad \text{(σχέση 7)}$$

και γι' αυτό

$$\begin{cases} \dot{x}_A^I(t) &= x_C^W(t) - v_u(t)\cos\theta(t) + v_w(t)\sin\theta(t) \\ \dot{y}_A^I(t) &= y_C^W(t) - v_u(t)\sin\theta(t) - v_w(t)\cos\theta(t) \end{cases}. \quad \text{(σχέση 8)}$$

Στο διάγραμμα ελεύθερου σώματος που απεικονίζεται στην εικόνα 178 θεωρούμε ότι οι μόνες δυνάμεις που δρουν στο ρομπότ είναι αυτές που εφαρμόζονται από τους τροχούς στο πλαίσιο του οχήματος. Ο τρίτος παθητικός πανκατευθυντικός τροχός δεν λαμβάνεται υπόψη.

Ισορροπία δυνάμεων

Παράγουμε το δυναμικό μοντέλο επιβάλλοντας την ταυτόχρονη ισορροπία δυνάμεων κατά μήκος των διαμηκών και πλευρικών κατευθύνσεων στο πλαίσιο του ρομπότ με τις αντίστοιχες αδρανειακές δυνάμεις και των στιγμών γύρω από τον κατακόρυφο άξονα που διέρχεται από το κέντρο του μάζα του ρομποτικού οχήματος.

$$\begin{aligned} Ma_u(t) &= F_{u_L} + F_{u_R} \\ Ma_w(t) &= F_{w_L} + F_{w_R} \\ \ddot{\theta}(t) &= \frac{L}{J}(F_{u_R} - F_{u_L}) + \frac{c}{J}(F_{w_R} + F_{w_L}). \end{aligned} \quad \text{(σχέση 9)}$$

Αντικαθιστώντας την σχέση 6 στην σχέση 9 οι εξισώσεις ισορροπίας εκφράζονται ως επιταχύνσεις του κέντρο μάζας στο πλαίσιο του ρομπότ.

$$\begin{aligned} \dot{v}_u(t) &= v_w(t)\dot{\theta}(t) + \frac{F_{u_L} + F_{u_R}}{M} \\ \dot{v}_w(t) &= -v_u(t)\dot{\theta}(t) + \frac{F_{w_L} + F_{w_R}}{M} \\ \ddot{\theta}(t) &= \frac{L}{J}(F_{u_R} - F_{u_L}) - \frac{c}{J}(F_{w_R} + F_{w_L}). \end{aligned} \quad \text{(σχέσεις 10, 11 και 12)}$$

Αυτό είναι ένα γενικό δυναμικό μοντέλο (με την έννοια των μη κινητικών περιορισμών) ενός διαφορικού ρομπότ κίνησης σύμφωνα με τις γεωμετρικές παραδοχές που αναφέρονται παραπάνω. Σημειώνεται ότι είναι ένα συζευγμένο και μη γραμμικό μοντέλο, όχι ακριβώς το καλύτερο σενάριο (μας αρέσουν τα γραμμικά πράγματα, επειδή υπάρχουν πολλά εργαλεία για να τα χειριστούμε). Όταν χρησιμοποιούμε το παραπάνω γενικό δυναμικό μοντέλο, είναι λογικό να το συσχετίζουμε και το γενικό κινηματικό μοντέλο, δεδομένου ότι:

$$\dot{x}_A(t) = v_u(t)\cos\theta(t) - (v_w(t) - c\dot{\theta})\sin\theta(t)$$
$$\dot{y}_A(t) = v_u(t)\sin\theta(t) + (v_w(t) - c\dot{\theta})\sin\theta(t). \text{ (σχέση 13)}$$

Τα παραπάνω (σχέσεις 23) μπορούν να ληφθούν υπενθυμίζοντας από οι μετασχηματισμοί είναι ισομετρικοί μετασχηματισμοί και από την άλλη πλευρά ότι :

$$\begin{cases} x_A(t) &= x_C^W(t) - c\cos\theta(t) \\ y_A(t) &= y_C^W(t) - c\sin\theta(t) \end{cases}$$
$$\begin{cases} x_C^I(t) &= v_u(t)\cos\theta(t) - v_w(t)\sin\theta(t) \\ y_C^I(t) &= v_u(t)\sin\theta(t) + v_w(t)\cos\theta(t) \end{cases} \text{ (σχέσεις 14,15)}$$

Η εξίσωση που περιγράφεται στην σχέση 13 μπορεί να διατυπωθεί εκ νέου ως εξής :

$$V_A^I = R(\theta)v_A^R \text{ όπου } v_A^R = [u_u(t), u_w(t) - c\dot{\theta}(t)]^T \text{ (σχέση 16)}$$

Προκειμένου να απλοποιήσουμε το μοντέλο θα προχωρήσουμε στην επιβολή ορισμένων κινηματικών περιορισμών.

Κινηματική

Σε αυτήν την ενότητα θα προσδιορίσουμε το κινηματικό μοντέλο μιας κινητής πλατφόρμας διαφορικής κίνησης. Για γίνει εφικτό αυτό πρέπει να κάνοντας δύο (2) παραδοχές - υποθέσεις για τους κινηματικούς περιορισμούς.

Η πρώτη υπόθεση είναι **δεν υπάρχει το φαινόμενο της πλευρικής ολίσθησης** δηλαδή το ρομπότ δεν μπορεί να κινηθεί προς τα πλάγια αλλά μόνο προς την κατεύθυνση της κίνησης. Αυτό πρακτικά σημαίνει ότι η πλευρική ταχύτητα στο πλαίσιο του ρομπότ είναι μηδέν. $\dot{y}_A^r = 0$ (σχέση 17). Αντιστρέφοντας τη σχέση 2 αυτός ο περιορισμός μπορεί να εκφραστεί μέσω των αδρανειακών μεταβλητών πλαισίου δίνοντας :

$$\dot{y}_A(t)cos\theta(t) - \dot{x}_A(t)sin\theta(t) = 0. \text{ (σχέση 18)}$$

Αν συνδυάσουμε την παραπάνω σχέση με την 8 έχουμε:

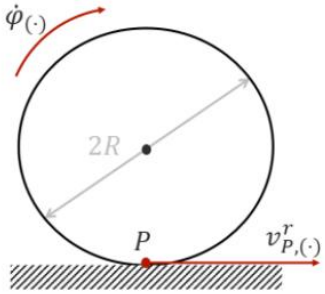
$$u_w(t) = \dot{y}_C^I(t)cos\theta(t) - \dot{x}_C^I(t)sin\theta(t), \quad \text{(σχέση 19)}$$

και υπενθυμίζοντας ότι $C^R = (c, 0)$ οδηγούμαστε στη σχέση

$$\dot{y}_C^I(t)\cos\theta(t) - \dot{x}_C^I(t)sin\theta(t) = c\dot{\theta}(t). \text{ (σχέση 20)}$$

Ως εκ τούτου αποκτούμε την ισχυρότερη έκφραση αυτού του περιορισμού : $v_w(t) = c\dot{\theta}(t)$, και ως εκ τούτου καταλήγουμε στη σχέση $\dot{u}_w(t) = c\ddot{\theta}(t)$. (σχέση 21).

Η δεύτερη παραδοχή που κάνουμε είναι αυτή της **καθαρής κύλισης** ότι δηλαδή οι τροχοί δεν γλιστρούν ποτέ.



Εικόνα 179: Κινηματικός προσδιορισμός καθαρής κύλισης

Υπενθυμίζοντας ότι R είναι η ακτίνα των δύο τροχών είναι ίδια και θέτοντας $\dot{\varphi}_l, \dot{\varphi}_r$ τις γωνιακές ταχύτητες του αριστερού και δεξιού τροχού αντίστοιχα, η ταχύτητα του σημείου επαφής εδάφους P στο πλαίσιο ρομπότ δίνεται από τη σχέση

$$\begin{cases} v_{P,r} &= R\dot{\varphi}_r \\ v_{P,l} &= R\dot{\varphi}_l \end{cases}. \quad \text{(σχέση 22)}$$

Μια άλλη αξιοσημείωτη συνέπεια αυτής της παραδοχής είναι ότι, πάντα στο πλαίσιο του ρομπότ, η πλήρης ισχύς του κινητήρα μπορεί να μεταφραστεί σε μια προωθητική δύναμη για το όχημα κατά τη διαμήκη κατεύθυνση. Ή, πιο απλά μπορούμε να το γράψουμε:

$$F_{u,(\cdot)}(t)R = \tau_{(\cdot)}(t), \text{(σχέση 23)}$$

όπου $\tau_{(\cdot)}(t)$ είναι η ροπή που ασκείται από κάθε κινητήρα στον τροχό του $(\cdot) = l, r$.

Κινηματικό μοντέλο ρομπότ διαφορικής κίνησης

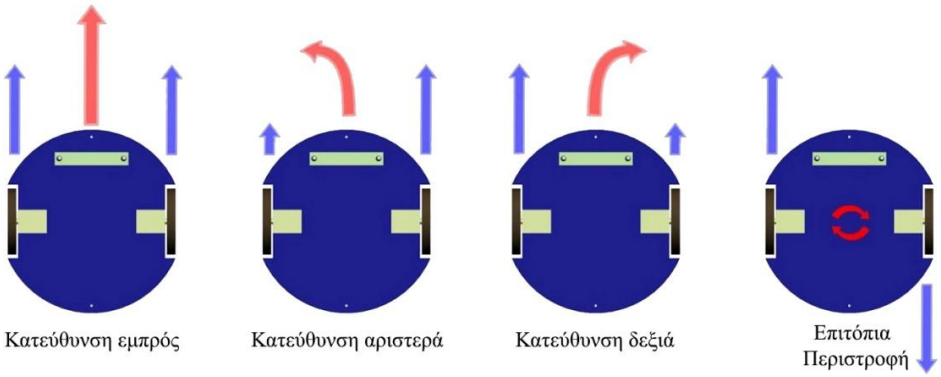
Σε ένα ρομπότ διαφορικής κίνησης, ο έλεγχος των τροχών σε διαφορετικές ταχύτητες δημιουργεί κυλιόμενη κίνηση ρυθμού $\omega = \dot{\theta}$. Σε ένα περιστρεφόμενο πεδίο υπάρχει πάντα ένα σταθερό σημείο, το *κέντρο της στιγμιαίας καμπυλότητας* (ICC) και όλα τα σημεία σε απόσταση d από αυτό θα έχουν μια ταχύτητα που δίνεται από το ωd , και κατεύθυνση ορθογώνια προς εκείνη της γραμμής που συνδέει το ICC και τους τροχούς. Για το λόγο αυτό παρατηρώντας την εικόνα 178 μπορούμε να γράψουμε :

$$\begin{cases} \dot{\theta}(d - L) &= v_l \\ \dot{\theta}(d + L) &= v_r \end{cases}, \quad \text{(σχέση 24) , από την οποία :}$$

$$\begin{cases} d &= L\frac{v_r + v_l}{v_r - v_l} \\ \dot{\theta} &= \frac{v_r - v_l}{2L} \end{cases}. \quad \text{(σχέση 25)}$$

Μερικές παρατηρήσεις που προέρχονται από την παραπάνω σχέση:

- Αν $u_r = u_l$ το ρομπότ δεν στρίβει ($\dot{\theta} = 0$) και ως τούτου το ICC δεν ορίζεται
- Αν $u_r = -u_l$, το ρομπότ περιστρέφεται γύρω από τον εαυτό του και συνεπώς d=0 και $ICC \equiv A$.
- Αν $u_r = 0$ ή $u_l = 0$ η περιστροφή συμβαίνει γύρω από τον δεξιό στην πρώτη περίπτωση, αριστερό στην δεύτερη τροχό και d=2L (d=L).



Εικόνα 180 : Λειτουργία διαφορικής οδήγησης

Σε κάθε περίπτωση ένα ρομπότ διαφορικής κίνησης δεν μπορεί να κινηθεί προς την κατεύθυνση του ICC, είναι μια ιδιαιτερότητα του συγκεκριμένου τρόπου κίνησης. Υπενθυμίζοντας την υπόθεση της μη πλάγιας κίνησης ολίσθησης και τον περιορισμό της καθαρής κύλισης και παρατηρώντας ότι η ταχύτητα μετασχηματισμού του A στο σύστημα αναφοράς του ρομπότ είναι $u_A = \dot{\theta} d = (u_r + u_l)/2$ μπορούμε να γράψουμε:

$$\begin{cases} \dot{x}_A^R &= R(\dot{\varphi}_R + \dot{\varphi}_L)/2 \\ \dot{y}_A^R &= 0 \\ \dot{\theta} &= \omega = R(\dot{\varphi}_R - \dot{\varphi}_L)/(2L) \end{cases}, \quad \textit{(σχέση 26)}$$

που σε πιο συμπαγή απόδοση η απλοποιημένη κινηματική εξίσωση στο πλαίσιο ρομπότ είναι:

$$\begin{bmatrix} \dot{x}_A^R \\ \dot{y}_A^R \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{R}{2} & \frac{R}{2} \\ 0 & 0 \\ \frac{R}{2L} & -\frac{R}{2L} \end{bmatrix} \begin{bmatrix} \dot{\varphi}_R \\ \dot{\varphi}_L \end{bmatrix}. \quad \text{(σχέση 27)}$$

Τέλος χρησιμοποιώντας τον πίνακα περιστροφής της σχέσης (3) μπορούμε να αναδιαμορφώσουμε την παραπάνω σχέση στο αδρανειακό πλαίσιο αναφοράς ως εξής:

$$\dot{\mathbf{q}}^I = \mathbf{R}(\theta) \begin{bmatrix} \dot{x}_A^r \\ \dot{y}_A^r \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{R}{2}\cos\theta & \frac{R}{2}\cos\theta \\ \frac{R}{2}\sin\theta & \frac{R}{2}\sin\theta \\ \frac{R}{2L} & -\frac{R}{2L} \end{bmatrix} \begin{bmatrix} \dot{\varphi}_R \\ \dot{\varphi}_L \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_A \\ \omega \end{bmatrix}.$$

(σχέση 28)

Απλοποιημένο δυναμικό μοντέλο

Εφαρμόζοντας τους σχηματισμούς κινηματικών περιορισμών που προέρχονται παραπάνω, δηλαδή, τη μη πλευρική ολίσθηση (σχέση 21) και την καθαρή κύλιση (23) στο γενικό δυναμικό μοντέλο, είναι εύκολο να συμπεράνουμε:

$$\begin{aligned} \dot{v}_u(t) &= c\dot{\theta}^2(t) + \frac{1}{RM}(\tau_R(t) + \tau_L(t)) \\ \dot{v}_w(t) &= c\dot{\theta}(t) \\ \ddot{\theta} &= -\frac{Mc}{Mc^2 + J}\dot{\theta}(t)v_u(t) + \frac{L}{R(Mc^2 + J)}(\tau_R(t) - \tau_L(t)) \end{aligned}$$

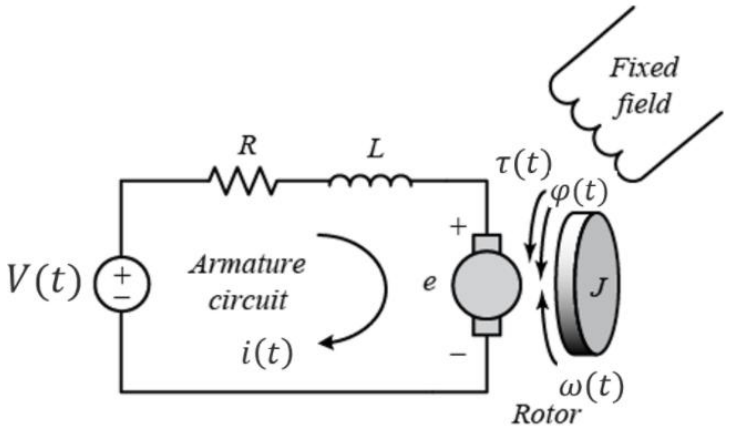
(σχέση 30)

Δυναμικό μοντέλο κινητήρα DC

Οι εξισώσεις που διέπουν τη συμπεριφορά ενός κινητήρα DC καθοδηγούνται από μια *τάση οπλισμού* εισόδου : V(t)

$$\begin{aligned} V(t) &= Ri(t) + L\frac{di}{dt} + e(t) \\ e(t) &= K_b\dot{\varphi}(t) \\ \tau_m(t) &= K_t i(t) \\ \tau(t) &= N\tau_m(t), \end{aligned}$$

όπου (K_b,K_t) είναι οι σταθερές του emf (Electromagnetic field - Ηλεκτρομαγνητικού πεδίου) και της ροπής αντίστοιχα και N είναι η σχέση μετάδοσης (στο Duckiebot N=1).



Εικόνα 181: Διάγραμμα DC κινητήρα

Έχοντας σχέση μεταξύ της εφαρμοζόμενης τάσης και ροπής, εκτός από τα δυναμικά και κινηματικά μοντέλα ενός ρομπότ διαφορικής κίνησης, μας επιτρέπει να προσδιορίσουμε όλες τις πιθανές μεταβλητές κατάστασης ενδιαφέροντος. Οι διαταραχές ροπής που δρουν στους τροχούς, όπως τα φαινόμενα τριβής, μπορούν να μοντελοποιηθούν ως πρόσθετοι όροι στις εξισώσεις του DC κινητήρα.

Εκείνο που έχουμε καταφέρει με τις παραπάνω κινηματικές εξισώσεις είναι να έχουμε τώρα μια ακολουθία περιγραφικών εργαλείων που λαμβάνουν ως είσοδο το σήμα τάσης που αποστέλλεται από τον ελεγκτή και παράγουν ως έξοδο οποιεσδήποτε από τις μεταβλητές κατάστασης, π.χ. τη θέση, την ταχύτητα και τον προσανατολισμό του ρομπότ σε σχέση με ένα σταθερό αδρανειακό πλαίσιο.

Στο σημείο αυτό θα πρέπει να επισημάνουμε δύο σημαντικές παρατηρήσεις. Η πρώτη είναι ότι παρόλο που για την δημιουργία του μοντέλου κινηματικής (διαφορικής οδήγησης) του Duckiebot έγινα πολλές υποθέσεις απλοποίησης όπως άκαμπτη κίνηση του σώματος, συμμετρία, καθαρή κύλιση και μη πλευρική ολίσθηση εντούτοις το μοντέλο δεν είναι γραμμικό.

Η δεύτερη παρατήρηση έχει να κάνει με το γεγονός ότι το παραπάνω μοντέλο προϋποθέτει τη γνώση ορισμένων σταθερών που χαρακτηρίζουν τη γεωμετρία, τα υλικά και τους κινητήρες DC του ρομπότ. Χωρίς τη γνώση αυτών των σταθερών το μοντέλο θα μπορούσε να είναι εντελώς άκυρο. Ο προσδιορισμός αυτών των παραμέτρων με τρόπο που βασίζεται στη μέτρηση, δηλαδή, η «αναγνώριση συστήματος» του ρομπότ, υπόκειται στον κλάδο της *οδομετρίας*.

Πηγές - Βιβλιογραφία

[1] Andrea Censi, Liam Paull. The Duckietown Project Διαθέσιμο στην ηλεκτρονική διεύθυνση: https://www.duckietown.org/instructors/classes/educational-resources

[2] SAE International Releases Updated Visual Chart for Its "Levels of Driving Automation" Standard for Self-Driving Vehicles. Διαθέσιμο στην ηλεκτρονική διεύθυνση : https://www.sae.org/news/press-room/2018/12/sae-international-releases-updated-visual-chart-for-its-%E2%80%9Clevels-of-driving-automation%E2%80%9D-standard-for-self-driving-vehicles

[3] Duckietown - An almost off-the-shelf platform for autonomous vehicle, Rasmus Lunding Henriksen, Aarhus University, June 2018

[4] Intro to Autonomous Vehicles, Liam Paull (Univerite de Montreal), Σεπτέμβριος 2018. Διαθέσιμο στην ηλεκτρονική διεύθυνση: https://github.com/duckietown/lectures/blob/master/2018/pdfs/autonomous_vehicles.pdf

[5] A Technical Overview of Autonomy, Liam Paull (Universite de Montreal), Σεπτέμβριος 2016. Διαθέσιμο στην ηλεκτρονική διεύθυνση : https://github.com/duckietown/lectures/blob/master/1_ideal/05_AV_intro/autonomy_overview.pdf

[6] 4D-RCS Reference Model Architecture. Διαθέσιμο στην ηλεκτρονική Διεύθυνση https://en.wikipedia.org/wiki/4D-RCS_Reference_Model_Architecture

[7] Ι. Βλαχάβας, Π. Κεφαλάς, Ν. Βασιλειάδης, Φ. Κόκκορας, Η. Σακελλαρίου. Τεχνητή Νοημοσύνη - Β' Έκδοση. Ευφυείς Πράκτορες . Διαθέσιμο στην ηλεκτρονική διεύθυνση: http://aibook.csd.auth.gr/include/slides/Chap27.pdf

[8] Duckietown.org, The Duckietown library (experimental version). Διαθέσιμο στην ηλεκτρονική διεύθυνση: https://docs.duckietown.org/daffy/

[9] Duckietown.org, The Duckietown library (stable branch - 2019 version). Διαθέσιμο την ηλεκτρονική διεύθυνση: https://docs.duckietown.org/DT19/

[10] Duckietown.org, Educational Resources. Διαθέσιμο στην ηλεκτρονική διεύθυνση: https://www.duckietown.org/instructors/classes/educational-resources

[11] Duckietown.org, Operation manual. Διαθέσιμο στην διεύθυνση : https://docs.duckietown.org/DT19/opmanual_duckietown/out.pdf

[12] Duckietown.org , Traffic Lights Assembly. Διαθέσιμο στην ηλεκτρονική διεύθυνση: https://docs.duckietown.org/daffy/opmanual_duckietown/out/traffic_light_assembly.html

[13] Duckietown.org Traffic Signs Assembly. Διαθέσιμο στην ηλεκτρονική διεύθυνση: https://docs.duckietown.org/daffy/opmanual_duckietown/out/dt_ops_city_traffic_signs.html

[14] Duckietown.org, AMOD18 cSLAM. Διαθέσιμο στην ηλεκτρονική διεύθυνση: https://docs.duckietown.org/DT18/opmanual_duckiebot/out/demo_cslam.html

[15] Duckietown.org, Assembling the DB18 Duckiebot. Διαθέσιμο στην ηλεκτρονική διεύθυνση:https://docs.duckietown.org/daffy/opmanual_duckiebot/out/assembling_duckiebot_db18.html

[16] Duckietown.org, Laptop Setup. Διαθέσιμο στην ηλεκτρονική διεύθυνση: https://docs.duckietown.org/daffy/opmanual_duckiebot/out/laptop_setup.html

[17] Docker docs, Docker overview. Διαθέσιμο στην ηλεκτρονική διεύθυνση: https://docs.docker.com/get-started/overview/

[18] Docker docs, Install Docker Engine on Ubuntu. Διαθέσιμο στην ηλεκτρονική διεύθυνση: https://docs.docker.com/engine/install/ubuntu/

[19] Duckietown.org, Introduction to ROS. Διαθέσιμο στην ηλεκτρονική διεύθυνση: https://docs.duckietown.org/daffy/duckietown-robotics-development/out/ros_intro.html

[20] Linux top-tools: performance. Διαθέσιμο στην ηλεκτρονική διεύθυνση: http://rrowniak.com/performance/linux-top-tools-performance-measurements/

[21] \compose\ - A lightweight web-based CMS.Διαθέσιμο στην ηλεκτρονική διεύθυνση: http://compose.afdaniele.com/docs/latest/index#introduction

[22] Duckietown API Documentation, Camera_driver package. Διαθέσιμο την ηλεκτρονική διεύθυνση: http://rosapi.duckietown.p-petrov.com/repositories/dt-duckiebot-interface/docs/source/packages/camera_driver.html

[23] Duckietown API Documentation, adafruit_drivers package. Διαθέσιμο την ηλεκτρονική διεύθυνση: http://rosapi.duckietown.p-petrov.com/repositories/dt-duckiebot-interface/docs/source/packages/adafruit_drivers.html

[24] Θεοφάνης Β. Σβίγγος, Βαθμονόμηση Κάμερας με τη Χρήση των Σημείων Φυγής στις Τρεις Διαστάσεις, Διπλωματική Εργασία, Σχολή Θετικών επιστημών- Τμήμα Πληροφορικής και Τηλεπικοινωνιών, Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών, Ιανουάριος 2013.

[25] Richard Szeliski. Computer vision: algorithms and applications. Springer Science & Business Media, 2010

[26] Peter Corke. Robotics, Vision and Control: Fundamental Algorithms In MATLAB R Second, Completely Revised, volume 118. Springer, 2017

[27] Cyganek, B. & Siebert, J.. An Intro to 3D Computer Vision Techniques and Algorithms. Wiley, 2009.

[28] CSE486 Computer Vision I,Lecture 12 camera projection. Διαθέσιμο στην ηλεκτρονική διεύθυνση : http://www.cse.psu.edu/~rcollins/CSE486/

[29] Κανελλάκης Χριστόφορος του Ιωάννη, Κυρίτσης Γεώργιος του Φραγκούλη-Σωτήρη, Ανάπτυξη διάταξης βιομηχανικής όρασης για προσδιορισμό θέσης και προσανατολισμού κινούμενων αντικειμένων και οδήγηση ρομποτικού βραχίονα, Τμήμα Ηλεκτρολόγων Μηχανικών και Τεχνολογίας Υπολογιστών της Πολυτεχνικής Σχολής του Πανεπιστημίου Πατρών, Οκτώβριος 2014

[30] Richard Hartley and Andrew Zisserman. Multiple view geometry in computer vision. Cambridge university press, 2003

[31] Janne Heikkila and Olli Silven. A four-step camera calibration procedure with implicit image correction. In Computer Vision and Pattern Recognition,1997. Proceedings., 1997 IEEE Computer Society Conference on, IEEE, 1997

[32] Roland Siegwart, Illah Reza Nourbakhsh, and Davide Scaramuzza. Introduction to autonomous mobile robots. MIT press, 2011

[33] Zhengyou Zhang. A flexible new technique for camera calibration. IEEE Transactions on pattern analysis and machine intelligence, 2000

[34] Roger Tsai. A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. IEEE Journal on Robotics and Automation, 1987

[35] ETH Zurich University of Zurich Davide Scaramuzza. Vision algorithms for mobile robotics, May 2018. http://rpg.ifi.uzh.ch/teaching.html.

[36] Zhengyou Zhang. A flexible new technique for camera calibration. IEEE Transactions on pattern analysis and machine intelligence, 2000.

[37] Αθανασιάδης Ιωάννης, Τρισδιάστατη Ανακατασκευή για την εκτίμηση του όγκου φαγητού, Διπλωματική εργασία, Τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών, Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης, 2018.

[38] Wikipedia. Bilinear interpolation, May 2018. Available at https://en.wikipedia.org/wiki/Bilinear_interpolation.

[39] Janne Heikkila and Olli Silven. A four-step camera calibration procedure with implicit image correction. In Computer Vision and Pattern Recognition,1997. Proceedings., 1997.

[40] Duckietown.org, Camera calibration and validation. Διαθέσιμο στην ηλεκτρονική διεύθυνση: https://docs.duckietown.org/daffy/opmanual_duckiebot/out/camera_calib.html

[41] Duckietown.org, Wheel calibration. Διαθέσιμο στην ηλεκτρονική διεύθυνση: https://docs.duckietown.org/daffy/opmanual_duckiebot/out/wheel_calibration.html

[42] Liam Paull, Jacopo Tani, Heejin Ahn, Javier Alonso-Mora, Luca Carlone, Michal Cap, Yu Fan Chen, Changhyun Choi, Jeff Dusek, Yajun Fang, Daniel Hoehener, Shih-Yuan Liu, Michael Novitzky, Igor Franzoni Okuyama, Jason Pazis, Guy Rosman, Valerio Varricchio, Hsueh-Cheng Wang, Dmitry Yershov, Hang Zhao,Michael Benjamin, Christopher Carr, Maria Zuber, Sertac Karaman, Emilio Frazzoli, Domitilla Del Vecchio,Daniela Rus, Jonathan How, John Leonard, Andrea Censi, Duckietown: an Open, Inexpensive and Flexible Platform for Autonomy Education and Research, 2017.

[43] Δήμητρα Ν. Πάνου, Αφαίρεση περιθωρίου και διόρθωση παραμόρφωσης σε έγγραφα από κάμερα, Πτυχιακή Εργασία, Τμήμα Πληροφορικής και τηλεπικοινωνιών, Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών, 2016

[44] Γκαγκάρας Σπυρίδων, Τερζάκης Παναγιώτης, Εφαρμογές Ρομποτικής Όρασης, Πτυχιακή εργασία, Σχολή Μηχανικών, Τμήμα Βιομηχανικής Σχεδίασης και Παραγωγής, Πανεπιστήμιο Δυτικής Αττικής, 2018

[45] Οδηγίες για τη διαχείριση της ύλης για το μάθημα "Τεχνολογίες Πληροφορίας και Επικοινωνιών (Τ.Π.Ε.)" του Δημοτικού σχολείου για το σχολικό έτος 2020-21, Γραφείο Επιστημονικών Μονάδων Β' κύκλου, Ινστιτούτο Εκπαιδευτικής Πολιτικής, Υπουργείο Παιδείας και Θρησκευμάτων.

[46] |Διδακτέα ύλη, διδακτικό υλικό και οδηγίες για τη διδασκαλία του μαθήματος "Πληροφορική" των Α', Β' και Γ' τάξεων των ημερησίων Γυμνασίων για το σχ. έτος 2020-2021, Ινστιτούτο Εκπαιδευτικής Πολιτικής, Υπουργείο Παιδείας και Θρησκευμάτων.

[47] Διδακτέα ύλη, επιπρόσθετο διδακτικό υλικό και οδηγίες για τη διδασκαλία του μαθήματος "Εφαρμογές Πληροφορικής" της Α' τάξης του Ημερησίου και Εσπερινού Γενικού Λυκείου για το σχ. έτος 2020-21, Ινστιτούτο Εκπαιδευτικής Πολιτικής, Υπουργείο Παιδείας και Θρησκευμάτων

[48] Διδακτέα Ύλη και Οδηγίες διδασκαλίας των μαθημάτων Γενικής Παιδείας των Α', Β' και Γ' τάξεων Ημερησίου και Εσπερινού ΕΠΑ.Λ σχ. έτους 2020-2021.

[49] Ύλη και οδηγίες διδασκαλίας των Τεχνολογικών-Επαγγελματικών μαθημάτων του Τομέα Πληροφορικής της Β΄ τάξης Ημερήσιου και Εσπερινού ΕΠΑ.Λ., των μαθημάτων ειδικότητας των Ειδικοτήτων του Τομέα Πληροφορικής της Γ΄ τάξης Ημερήσιου και Εσπερινού ΕΠΑ.Λ. για το σχολικό έτος 2020-21

[50] Duckietown.org, Duckiebot modeling. Διαθέσιμο στην ηλεκτρονική διεύθυνση : https://docs.duckietown.org/DT19/learning_materials/out/duckiebot_modeling.html

[51] Duckietown.org, Assembling the DB17 Duckiebot. Διαθέσιμο στην ηλεκτρονική διεύθυνση: https://docs.duckietown.org/DT19/opmanual_duckiebot/out/get_db_hw.html