Σχεδιασμός και κατασκευή έντροχου ρομποτικού εκπαιδευτικού οχήματος

Πολυτεχνείο Κρήτης
Σχολή Μηχανικών Παραγωγής και Διοίκησης

Γρένδας Γεώργιος

Χανιά

Νοέμβριος 2023

Επιτροπή

Δρ. Δοϊτσίδης Ελευθέριος, *Επίκουρος Καθηγητής*	Σχολή Μηχανικών Παραγωγής και Διοίκησης, Πολυτεχνείο Κρήτης
Δρ. Ιψάκης Δημήτριος, *Επίκουρος Καθηγητής*	Σχολή Μηχανικών Παραγωγής και Διοίκησης, Πολυτεχνείο Κρήτης
Δρ. Πιπερίδης Σάββας, *Ε.ΔΙ.Π*	Σχολή Μηχανικών Παραγωγής και Διοίκησης, Πολυτεχνείο Κρήτης

Ευχαριστίες

Με το πέρας της διπλωματικής μου εργασίας, θέλω να εκφράσω τις θερμότερες ευχαριστίες σε όσους συνέβαλαν στην εκπόνηση και ολοκλήρωση της.
Ιδιαίτερα, θα ήθελα να εκφράσω τις ευχαριστίες μου στους επιβλέποντες καθηγητές μου κ. Δοϊτσίδη Ελευθέριο και κ. Πιπερίδη Σάββα για την ανεκτίμητη καθοδήγηση, την εμπιστοσύνη και την υποστήριξη τους, καθόλη την διάρκεια της προσπάθειας μου και την άμεση ανταπόκριση όποτε ήταν αναγκαία η βοήθεια τους.
Επιπλέον, θέλω να ευχαριστήσω τους φίλους μου, που σαν φοιτητές κι αυτοί, γνωρίζοντας τις δυσκολίες και την πίεση, αλληλοστηριζόμασταν σε όλο το ταξίδι μας. Θέλω να ευχαριστήσω τον φίλο μου Αρίσταρχο Τζατζό για όλη την βοήθεια που μου παρήχε, σε όλη την διάρκεια της φοιτητικής μου πορείας και ιδιαίτερα για την βοήθεια του στην ανάπτυξη και κατασκευή του ηλεκτρικού κυκλώματος αυτής της εργασίας.
Τέλος, οφείλω την μεγαλύτερη ευγνωμοσύνη στην οικογένεια μου για την απεριόριστη, άνευ όρων υποστήριξη, ψυχολογική και οικονομική, την ενθάρρυνση που μου έδωσαν και την υπομονή που έδειξαν στην πορεία μου αυτή. Από τα βάθη της καρδιάς μου, σας ευχαριστώ που με στηρίξατε ώστε να μπορέσω να ολοκληρώσω τον στόχο μου.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1

Περίληψη

Η συνεχώς αυξανόμενη εισαγωγή ρομποτικών διατάξεων και διατάξεων STEM, στην εκπαιδευτική διαδικασία έχει δημιουργήσει την ανάγκη ανάπτυξης εύχρηστων διατάξεων με χαμηλό κόστος και αυξημένες δυνατότητες, προκειμένου να χρησιμοποιηθούν ως πλατφόρμες πειραματισμού. Στόχος της παρούσας εργασίας, είναι η ανάπτυξη μιας έντροχης ρομποτικής πλατφόρμας, με χρήση ανοικτών εργαλείων και με μια προσέγγιση που βασίζεται στη τρισδιάστατη σχεδίαση και γρήγορη πρωτοτυποποίηση κατά τη φάση της ανάπτυξης και του σχεδιασμού. Στα πλαίσια της εργασίας πραγματοποιήθηκε έρευνα της τρέχουσας κατάστασης όσον αφορά τις υπάρχουσες εμπορικά διαθέσιμες πλατφόρμες και τις σχετικές τεχνολογίες. Με βάση την πληροφορία που συλλέχθηκε, αναπτύχθηκε ένα ρομποτικό όχημα διαφορικής οδήγησης. Η συγκεκριμένη ρομποτική κατασκευή είχε ώς στόχο το μικρότερο δυνατό μέγεθος της συσκευής, το δυνατότερο χαμηλό κόστος της, καθώς επίσης και η δομοστοιχειωτή (modular) κατασκευή του.Επίσης, βασικό απαιτούμενο της εργασίας αποτελεί η εγκατάσταση του λογισμικού ανοιχτού κώδικα ROS 2 για τον έλεγχο και τη λειτουργία της συσκευής. Η συσκευή αυτή θα χρησιμοποιηθεί για τη διδασκαλία ρομποτικής στους φοιτητές του εργαστηρίου Ρομποτικής και Ευφυών Συστημάτων, της σχολής Μηχανικών Παραγωγής και Διοίκησης, του Πολυτεχνείου Κρήτης.

Κεφάλαιο 2

Εισαγωγή

Όπως η Ρώμη δεν χτίστηκε σε μία μέρα, η ανάπτυξη και εξέλιξη των εκπαιδευτικών ρομπότ διήρκησε γενιές ολόκληρες για να βρεθεί στο επίπεδο που είναι σήμερα. Σύμφωνα με έρευνες, η κατασκευή μηχανικών συναρμογών υπήρχε από την εποχή των αρχαίων Ελλήνων. Τα πρώτα ρομπότ εκπαιδευτικής χρήσης χρονολογούνται περίπου στα μέσα της δεκαετίας του 1940, μετά το τέλος του Δεύτερου Παγκοσμίου Πολέμου, στην Ιαπωνία. Στις Ηνωμένες Πολιτείες της Αμερικής, το ενδιαφέρον και η ζήτηση εκπαιδευτικών ρομπότ αναπτύχθηκαν ραγδαία λόγω του σημαντικού τους ρόλου στην καθημερινή ζωή των ανθρώπων και την οικονομία. Η λογοτεχνία επιστημονικής φαντασίας καθώς και οι ταινίες της εποχής συνέβαλαν επίσης στο να διευρυνθεί η φήμη τους. Έτσι, οι Ιάπωνες κατασκευαστές, βλέποντας την ευκαιρία που ξεδιπλωνόταν, ξεκίνησαν μία ολόκληρη κατασκευαστική βιομηχανία εξαγωγής ρομπότ - παιχνιδιών χαμηλού κόστους στην Αμερική. Το 1954, κατασκευάζεται το πρώτο Αμερικάνικο ρομπότ - παιχνίδι. Την επόμενη χρονιά, η Ιαπωνία ξεκινά την εξαγωγή ρομπότ παιχνιδιών που λειτουργούν με μπαταρία στις Η.Π.Α. Μέχρι τα μέσα του 1950, η Ιαπωνία είχε γίνει πρωτεύων κατασκευαστής ρομπότ - παιχνιδιών στον κόσμο.

Ερχόμενοι στο σήμερα, τα εκπαιδευτικά ρομπότ έχουν αποκτήσει περισσότερες λειτουργίες και χρήσιμοποιούνται σε περισσότερους τομείς από πριν, χάρη στην ραγδαία ανάπτυξη της τεχνολογίας και των αυξανόμενων ανθρώπινων αναγκών. Σήμερα, τα ρομπότ - παιχνίδια χρησιμοποιούνται είτε για εκπαιδευτικούς σκοπούς είτε και για διασκέδαση. Ο τομέας των STEAM (Science, Technology, Engineering, The Arts, and Mathematics) προσελκύει περισσότερο ενδιαφέρον από ποτέ, και γονείς κι εκπαιδευτικοί ψάχνουν συνεχώς καινούριους τρόπους ώστε τα μικρότερα παιδιά να αναμειχθούν με την εκπαίδευση STEAM, καθώς έχει αποδειχτεί αρκετά αποτελεσματικός τρόπος διδασκαλίας. Έρευνες έχουν δείξει ότι τα ρομπότ εκπαιδευτικού τύπου μπορούν να βοηθήσουν στην μετατροπή της μονότονης ψηφιακής μάθησης, σε χειροπιαστή διαδικασία η οποία μπορεί να αυξήσει σημαντικά το ενδιαφέρον και την παραγωγικότητα των παιδιών. Επιπλέον, η ενασχόληση με τα ρομπότ αυτά, τους δίνει την δυνατότητα να εξασκήσουν τα ταλέντα και τις δεξιότητες τους. Μελέτες, ακόμα έχουν δείξει ότι τα παιδιά με αυτισμό αλληλεπιδρούν ευκολότερα με τα ρομπότ και αποκρίνονται πολύ ευκολότερα προς αυτά, λόγω της συνεπούς και ήρεμης φύσης των ρομπότ.

Η ενασχόληση με εκπαιδευτικά ρομπότ όμως, δεν απευθύνεται αποκλειστικά σε παιδιά. Όπως αναφέρθηκε η τεχνολογία σήμερα εξελίσσεται με ραγδαίους ρυθμούς. Έχει βοηθήσει η εργασία να γίνει ευκολότερη, ταχύτερη και επαρκής. Όσον αφορά την ηλικία, δεν είναι χαρακτηριστικό της τεχνολογίας. Απευθύνεται σε μωρά, παιδιά, εφήβους, μαμάδες, μπαμπάδες, μαθητές, εργαζόμενους, ενήλικες, αρχάριους ή μη, με

λίγα λόγια στους πάντες. Γιατί, όμως, ένας ενήλικας να ασχοληθεί με την ρομποτική· Η ρομποτική και η ηλεκτρονική είναι εξαιρετικοί τρόποι εκμάθησης και βελτίωσης πολλών ικανοτήτων, όπως είναι κριτική σκέψη, η λογική, ο συλλογισμός, τα μαθηματικά, η φυσική και ο προγραμματισμός. Τα παραπάνω αποτελούν πολύπλοκα θέματα τα οποία μπορεί να είναι είτε δύσκολα είτε βαρετά για αρχάριους που προσπαθούν να ασχοληθούν μόνοι. Στην περίπτωση αυτή υπάρχουν πολλά διαθέσιμα σετ πρακτικής εκμάθησης για ενήλικες, τα οποία κάνουν την εμπειρία της ενασχόλησης ευκολότερη και πιο διασκεδαστική. Επιπρόσθετα, είναι ένας τρόπος πρόκλησης του εαυτού και διέγερσης του μυαλού, με τόν ίδιο τρόπο όπως είναι τα σταυρόλεξα και τα σουντόκου.

Στόχος της παρούσας εργασίας, είναι η ανάπτυξη μιας έντροχης ρομποτικής πλατφόρμας, με χρήση ανοικτών εργαλείων και με μια προσέγγιση που βασίζεται στη τρισδιάστατη σχεδίαση και γρήγορη πρωτοτυποποίηση κατά τη φάση της ανάπτυξης και του σχεδιασμού. Η δομή της υπόλοιπης εργασίας είναι η ακόλουθη.

Στο κεφάλαιο 3, παρουσιάζεται η έρευνα αγοράς που πραγματοποιήθηκε για την επιλογή των απαραίτητων εργαλείων, εξαρτημάτων και τεχνολογιών για την ανάπτυξη της ρομποτικής πλατφόρμας, καθώς επίσης και αντίστοιχες πλατφόρμες.

Στο κεφάλαιο 4, γίνεται παρουσίαση του μοντέλου της διαφορικής οδήγησης πάνω στο οποίο θα βασίζεται η κίνηση του αυτοκίνητου ρομπότ, του λογισμικού ανοιχτού κώδικα ROS καθώς επίσης και των σχεδίων και των συσκευών που επιλέχθηκαν για τη κατασκευή της ρομποτικής συσκευής. Επιπλέον, στο τέλος παρουσιάζεται η συνολική κατασκευή του ρομπότ.

Στο κεφάλαιο 5, παρουσιάζονται οι δοκιμές που έγιναν για τον έλεγχο της ορθής λειτουργίας της συσκευής.

Τέλος, στο κεφάλαιο 6, παρουσιάζονται παρατηρήσεις πάνω στην συγκεκριμένη εργασία, καθώς επίσης και προτάσεις μελλοντικής εξέλιξης της.

Κεφάλαιο 3

Βιβλιογραφική έρευνα και τρέχουσα κατάσταση

Στο κεφάλαιο αυτό παρουσιάζεται η τρέχουσα κατάσταση στο χώρο της εκπαιδευτικής ρομποτικής και οι σχετικές τεχνολογίες που αφορούν τα βασικά δομικά στοιχεία των εκπαιδευτικών ρομπότ. Παράλληλα γίνεται μια παρουσίαση των εκπαιδευτικών πλατφόρμων της αγοράς από το 2020 και μετά. Για κάθε ένα απο τα αντικείμενα που παρουσιάζονται, παρατίθενται περιγραφή τους και πίνακας με τα χαρακτηριστικά τους. Με τον τρόπο αυτό γίνεται εύκολη η σύγκριση μεταξύ τους, όσων αφορά τις τεχνολογίες και όσον αφορά τις αντίστοιχες εκπαιδευτικές πλατφόρμες, παρέχοντας μία ιδέα ή όραμα για την ανάπτυξη του ρομποτικού οχήματος που είναι το θέμα αυτής της εργασίας.

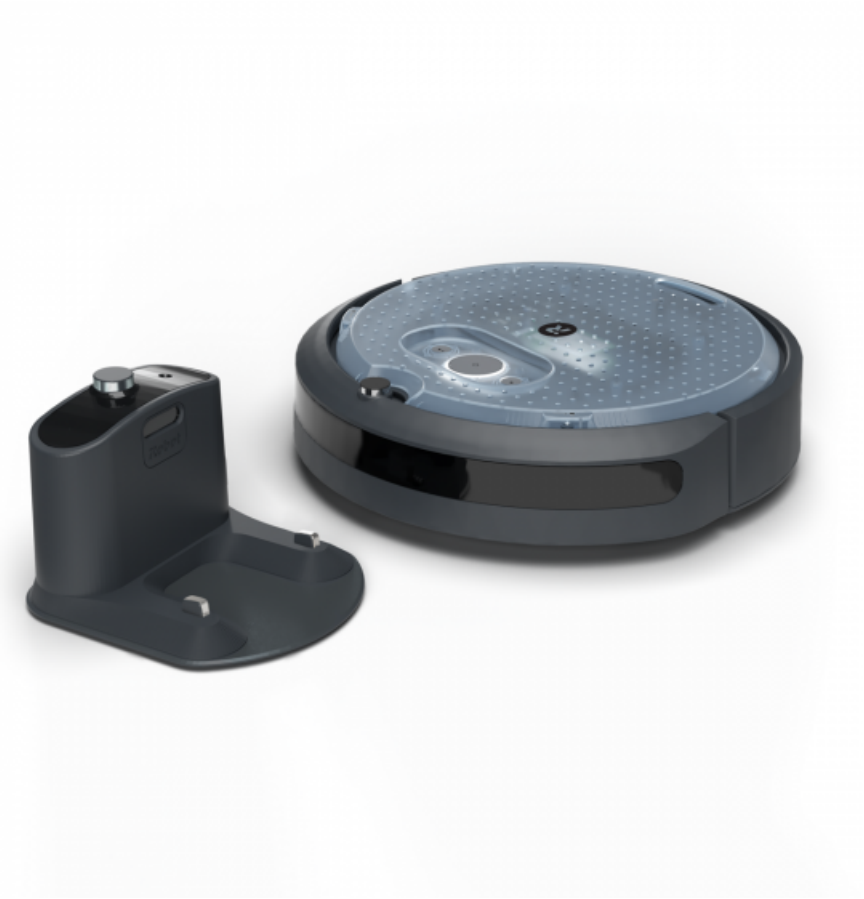
3.1 Εκπαιδευτικές πλατφόρμες

3.1.1 iRobot

Το iRobot® Create® 3 είναι μία προσιτή (τιμή περίπου 280€), φορητή πλατφόρμα για εκπαιδευτικούς, μαθητές και προγραμματιστές. Το ρομπότ περιέχει ένα πλήρες σετ εγκατεστημένων αισθητήρων και ενεργοποιητών που επιτρέπουν την ανάπτυξη και δοκιμή ρομποτικών αλγορίθμων.

Το λειτουργικό του σύστημα είναι εξολοκλήρου βασισμένο στο ROS 2. Όλα τα δεδομένα των αισθητήρων παράγονται μέσω ROS 2 δημοσιεύσεων καθώς οι διακομιστές και συνδρομητές του ROS 2 χρησιμοποιούνται για τον έλεγχο των ενεργοποιητών και την ανταπόκριση στα αιτήματα των χρηστών. Το ρομπότ, επίσης, παρέχει συμπεριφορές μερικής αυτονομίας όπως, προσδεση στην βάση του, ακολούθηση κατά μήκος τοίχων και απόκριση σε εμπόδια. Όλα αυτά μπορούν να πυροδότηθούν ή/και διαμορφωθούν μέσω των ενεργειών και των παραμέτρων του ROS 2.

Σχεδιασμένο για αρχάριους και έμπειρους χρήστες του ROS 2, επιτρέπει ποικιλία προγραμματιστικών μεθόδων. Με τη δυνατότητα WI-FI και Bluetooth σύνδεσης, USB ως Ethernet συνδέσεις, και επιπλέον αποσπόμενο καπάκι για την πρόσδεση φορτίων, δίνει πληθώρα δυνατοτήτων για την διαμόρφωση του.



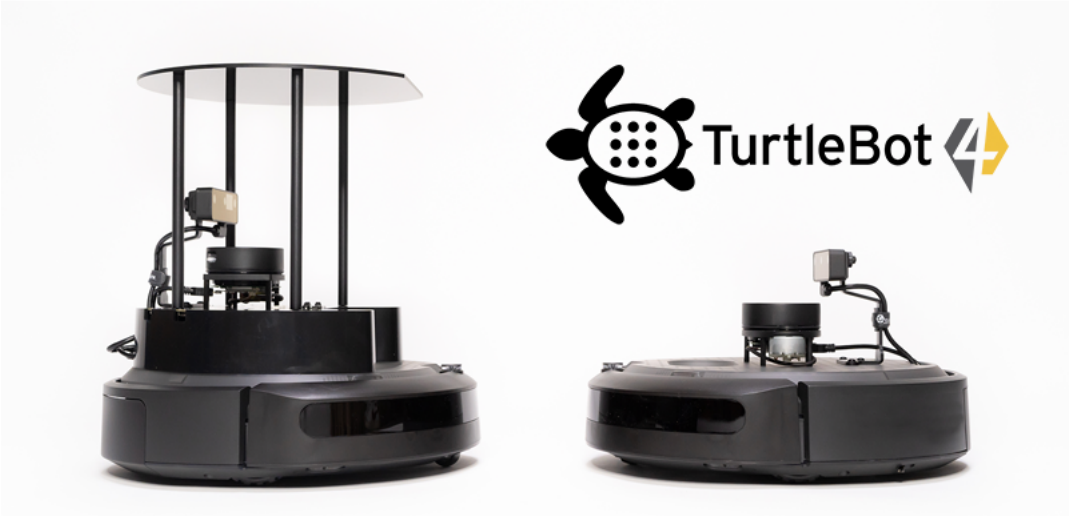
Σχήμα 3.1: iRobot® Create® 3 [1]

Πίνακας 3.1: Χαρακτηριστικά του iRobot® Create3

Διαστάσεις (Μ - Π - Υ)	143 x 407 x 473 mm
Βάρος	3 kg
Αισθητήρας Υπερύθρων - Συνδετήρας	x4
Αισθητήρας Υπερύθρων για εμπόδια	x6
Οπτικός Αισθητήρας Καθοδικής Ροής για οδομετρία	x1
Γυροσκόπιο	x1
Επιταχυνσιόμετρο	x1
Ηχείο	x1
Κωδικοποιητής Τροχών	x2
Προσβάσιμες θύρες Ισχύος και USB	Ναι
Bluetooth	Ναι
Wi-Fi	Ναι
Βάση Στήριξης	Ναι
Λειτουργικό	ROS 2

3.1.2 TurtleBot 4

Το TurtleBot 4 είναι η νέα γενιά της πιο γνωστής ρομποτικής πλατφόρμας με λογισμικού ανοιχτού κώδικα σχεδιασμένη με έμφαση την εκπαίδευση και την έρευνα.



Σχήμα 3.2: TurtleBot 4 [2]

Υπάρχουν δύο εκδόσεις του TurtleBot 4 όπως φαίνεται και στο Σχήμα 3.2. Το TurtleBot 4 Standard (αριστερά, τιμή περίπου 1,700€) και το TurtleBot 4 Lite (δεξιά, τιμή περίπου 1,100€). Και τα δύο είναι εξοπλισμένα με ένα iRobot® Create3 ως βάση, ένα ισχυρό Raspberry Pi 4 στο οποίο υπάρχει εγκατέστημένο ROS 2, ένας OAK-D spatial AI stereo αισθητήρας εικόνας, ένας αισθητήρας λέιζερ δύο διαστάσεων (2D Lidar) κ.α. Όλα τα εξαρτήματα έχουν ενσωμετωθεί απρόσκοπτα έτσι ώστε να προκύψει μία έτοιμη-για-χρήση ανάπτυξης και εκμάθησης πλατφόρμα.

Τα λειτουργικά χαρακτηριστικά των 2 εκδόσεων παρουσιάζονται στον Πίνακα 3.2.

Πίνακας 3.2: Χαρακτηριστικά των TurtleBot 4 εκδόσεων

341 x 339 x 351 mm	**Διαστάσεις (Μ - Π - Υ)**	341 x 339 x 192 mm
3.9 kg	**Βάρος**	3.3 kg
15 kg	**Μέγιστο Φορτίο**	15 kg
2.5 - 4 ώρες (εξαρτάται και από το φορτίο)	**Διάρκεια Λειτουργίας**	2.5 - 4 ώρες (εξαρτάται και από το φορτίο)
OAK-D-PRO	**Αισθητήρας Εικόνας**	OAK-D-LITE
RPLidar A1	**Αισθητήρας Λέιζερ**	RPLidar A1
Ναι	**Προσβάσιμες θύρες Ισχύος και USB**	Όχι
Ναι	**Οθόνη τεχνολογίας OLED**	Όχι
Ναι	**Βάση Στήριξης**	Όχι
Ναι	**Bluetooth**	Ναι
Ναι	**Wi-Fi**	Ναι
ROS 2	**Λειτουργικό**	ROS 2
Raspberry Pi 4	**Ρομποτικός Ελεγκτής**	Raspberry Pi 4

3.1.3 Thymio

Το Thymio είναι ένα μικρό ρομπότ που θα επιτρέψει την ανακάλυψη του κόσμου της ρομποτικής και της τεχνολογίας, τη μάθηση της γλώσσα των ρομπότ και την μύηση στον προγραμματισμό. Δίνει την δυνατότητα προγραμματισμού του και της πραγματοποίησης πολλών πειράματων. Με το Thymio, οι βασικές αρχές της ρομποτικής και του προγραμματισμού γίνονται έννοιες που ο καθένας μπορεί να ανακαλύψει, ανεξαρτήτως ηλικίας.

Διαθέτει 6 προ-προγραμματισμένες συμπεριφορές όπως η Φιλική (Friendly), η Εξερευνητική (Explorer), η Φοβητσιάρικη (Fearful), η Ερευνητική (Investigator), η Υπάκουη (Obedient) και η Προσεχτική (Attentive). Αυτές είναι πάντα διαθέσιμες στο ρομπότ και επιλέξιμες μέσω των κουμπιών. Μόλις βγει από την συσκευασία του είναι έτοιμο για χρήση και μπορεί άμεσα να γίνει γνωστό πώς αλληλεπιδρά με το περιβάλλον του. Ωστόσο, δίνει τι δυνατότητα προγραμματισμού νέων συμπεριφορών έτσι ώστε να επιτευχθεί ο επιδιώκόμενος στόχος. Το Thymio μπορεί να προγραμματιστεί με δύο διαφορετικούς τρόπους, με οπτικό προγραμματισμό (Visual Programming Language, Blockly, Scratch) και με εισαγωγή εντολών (Aseba Text programming).

Υπάρχουν δύο εκδόσεις του Thymio. Η Thymio Wireless και η Thymio. Η μόνη διαφορά των δύο είναι ότι η wireless έκδοση παρέχει δυνατότητα απομακρισμένου ελέγχου.

Σχήμα 3.3: Thymio [3]

Πίνακας 3.3: Χαρακτηριστικά του Thymio

Διαστάσεις (Μ - Π - Υ)	110 x 112 x 53 mm
Βάρος	270 grams
Επαγωγικά πλήκτρα αφής	x5
Αισθητήρας Υπερύθρων	x9
Θερμόμετρο	x1
Επιταχυνσιόμετρο	x1
Μικρόφωνο	x1
Μονάδα απομακρυσμένου ελέγχου	Ναι (στην περίπτωση touwireless)
LED	x39
Ηχείο	x1

3.2 Ρομποτικοί ελεγκτές

Οι ρομποτικοί ελεγκτές αποτελούν βασικό στοιχείο μιας ρομποτικής συσκευής καθώς αποτελούν το τμήμα του ρομπότ που συλλέγει και διαχειρίζεται τα δεδομένα από τους αισθητήρες του, υλοποιεί το λογισμικό ελέγχου, οδηγεί τους ενεργοποιητές με τις κατάλληλες εντολές δράσης και επικοινωνεί με περιφερειακές συσκευές ή άλλα ρομπότ. Αποτελούνται από έναν ηλεκτρονικό υπολογιστή με ενσωματωμένες συσκευές όπως θύρες επικοινωνίας, κυκλώματα τροφοδοσίας, μετατροπείς αναλογικών σημάτων σε ψηφιακά, κυκλώματα οδήγησης των ενεργοποιητών και συσκευές επικοινωνίας. Οι θύρες επικοινωνίας είναι κατάλληλα διαμορφωμένες ηλεκτρικές επαφές που επιτρέπουν την επικοινωνία του ελεγκτή με τις υπόλοιπες συσκευές του ή με ελεγκτές άλλων ρομπότ. [4]

3.2.1 Raspberry Pi 4

Το Raspberry Pi 4 (Σχήμα 3.4) είναι μια δημοφιλής επιλογή που συνδυάζει επεκτασιμότητα, μικρό μέγεθος, χαμηλή κατανάλωση ενέργειας και χαμηλό κόστος (65 - 150€ ανάλογα με την έκδοση). Χρησιμοποιείται σε μεγάλη ποικιλία εφαρμογών, όπως αυτοσχέδια ρομπότ, παιχνιδομηχανές και κάμερες ασφαλείας. Μπορεί να συνδεθεί με με πλήθος συσκευών όπως, λάμπες, μοτέρ, αισθητήρες καθώς κι άλλες I/O συσκευές.

Σχήμα 3.4: Raspberry Pi 4 [5]

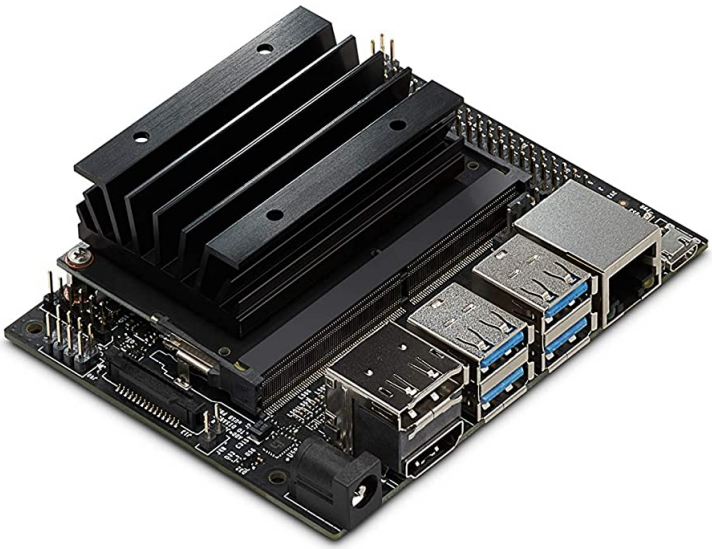
Τα χαρακτηριστικά του Raspberry Pi 4 παρουσιάζονται στον Πίνακα 3.4.

Πίνακας 3.4: Χαρακτηριστικά του Raspberry Pi 4

1	Τετραπύρηνος επεξεργαστής τεχνολογίας ARM Cortex - A72 (64 bit)
2	Εκδόσεις με 1,2,4 ή 8 GB μνήμη RAM
3	Bluetooth 5.0/BLE
4	Gigabit Ethernet
5	2 θύρες USB 3.0 και 2 θύρες USB 2.0
6	40 pin GPIO header
7	2 θύρες micro HDMI
8	USB-C θύρα τροφοδότησης
9	Θύρα σύνδεσης κάμερας
10	Δυνατότητα επέκτασης με κάρτας μνήμης
11	Ασύρματο LAN δύο καναλιών 2.4/5.0 GHz
12	Πολύ μεγάλη ποσότητα πληροφορίας για κάθε πιθανή χρήση του λόγω της δημοτικότητας του
13	Έτος κυκλοφορίας 2019

3.2.2 Nvidia Jetson Nano

Το Nvidia Jetson Nano (Σχήμα 3.5) είναι ένας μικρός, ισχυρός υπολογιστής με μικρό κόστος (~ 100 €) που επιτρέπει την ταυτόχρονη εκτέλεση πολλαπλών νευρωνικών δικτύων παράλληλα, σε εφαρμογές όπως image classification, ανίχνευση αντικειμένων (object detection), διαχωρισμό (segmentation), και επεξεργασία λόγου (speech processing). Προσφέρει νέες δυνατότητες σε εφαρμογές που έχουν να κανουν με εσνωσματωμένα συστήματα, IoT (Internet of Things), καθώς και σε ρομποτικές εφαρμογές. Επίσης, είναι το τέλειο εργαλείο για την ενασχόληση αρχαρίων στον τομέα την τεχνητής νοημοσύνης και την ρομποτική σε πραγματικές συνθήκες.



Σχήμα 3.5: Nvidia Jetson Nano [6]

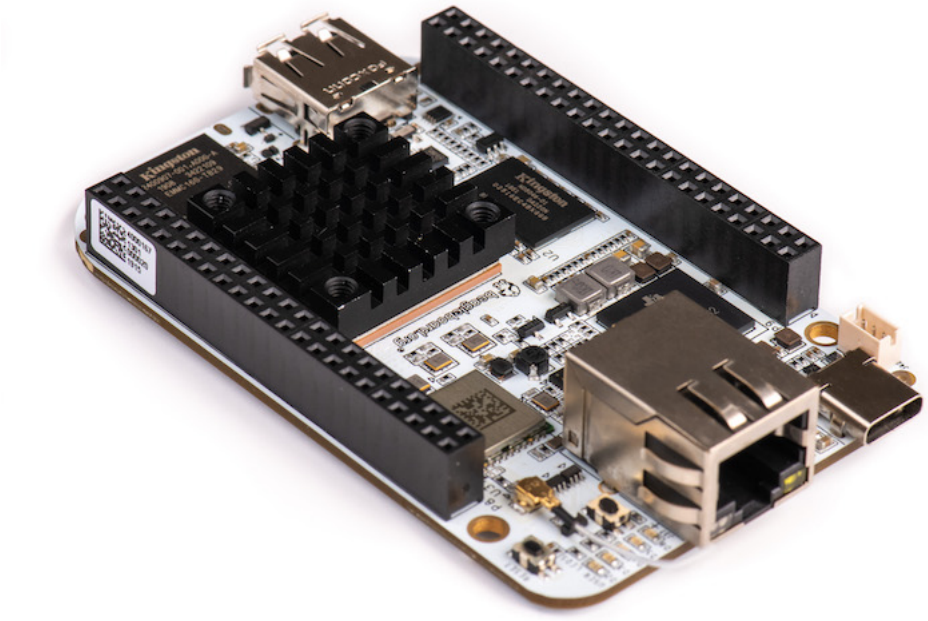
Τα χαρακτηριστικά του Nvidia Jetson Nano παρουσιάζονται στον Πίνακα 3.5.

Πίνακας 3.5: Χαρακτηριστικά του Nvidia Jetson Nano

1	Τετραπύρινος επεξεργαστής τεχνολογίας ARM A57 (64 bit)
2	128-πυρήνων βασισμένη στην Nvidia Maxwell αρχιτεκτονική κάρτα γραφικών
3	Εκδόσεις με 2 ή 4 GB RAM
4	16 GB ενσωματωμένης μνήμης
5	Δυνατότητα επέκτασης με κάρτα μνήμης
6	Gigabit Ethernet
7	40 pin GPIO header
8	Θύρα σύνδεσης κάμερας
9	Θύρα HDMI 2.0
10	Θύρες USB 3.0 και USB 2.0 (το πλήθος εξαρτάται απο την έκδοση)
11	Κωδικοποίηση βίντεο 250 MP/sec
12	Αποκωδικοποίηση βίντεο 500 MP/sec
13	Θύρα τροφοδότησης USB-C και micro USB ή Barrel jack και micro USB ανάλογα με την έκδοση
14	Έτος κυκλοφορίας 2020

3.2.3 Beaglebone AI

Το Beaglebone AI (Σχήμα 3.6) είναι μια επιλογή για όποιον θέλει να εξερευνήσει την τεχνητή νοημοσύνη στην καθημερινότητα. Παρέχει μια σειρά από προεγκατεστημένα εργαλεία με έμφαση στη μηχανική μάθηση (machine learning). Είναι κατάλληλό για χρήση σε αυτοματισμούς σε βιομηχανικό, εμπορικό αλλά και σε οικιακό περιβάλλον και έχει χαμηλό κόστος ($\sim$ 140 €).



Σχήμα 3.6: Beaglebone AI [7]

Τα χαρακτηριστικά του Beaglebone AI παρουσιάζονται στον Πίνακα 3.6

Πίνακας 3.6: Χαρακτηριστικά του Beaglebone AI

1	2 κύριους επεξεργαστές τεχνολογίας ARM Cortex A15 (32 bit)
2	2 Δευτερεύοντες επεξεργαστές τεχνολογίας ARM Cortex M4
3	4 EVEs (Embedded Vision Engines)
4	7 αναλογικά pins
5	72 ψηφιακά pins
6	1 GB μνήμης RAM
7	16 GB ενσωματωμένης μνήμης
8	Δυνατότητα επέκτασης με κάρτα μνήμης
9	Θύρα USB 3.0
10	Bluetooth 4.2/BLE
11	Gigabit Ethernet
12	Ασύρματο LAN δύο καναλιών 2.4/5.0 GHz
13	Θύρα τροφοδότησης USB-C
14	Θύρα micro HDMI
15	Έτος κυκλοφορίας 2019

3.3 Αισθητήρες

Τα αισθητήρια (σενσορς) είναι διατάξεις που μετατρέπουν κάποια γεγονότα ή φυσικά μεγέθη σε ηλεκτρικό σήμα, προκειμένου να γίνει μέτρηση τους.Τα αισθητήρια (σενσορς) διακρίνονται σε δύο βασικές κατηγορίες, τα ενεργά (χρειάζονται εξωτερική διέγερση για τη λήψη της μέτρησης του φυσικού μεγέθους) και τα παθητικά (παρέχουν απευθείας τη μέτρηση του φυσικού μεγέθους με μορφή ηλεκτρικού σήματος, χωρίς την ανάγκη εξωτερικής διέγερσης). Στις ρομποτικές εφαρμογές χρησιμοποιούνται πολλά και διαφορετικά αισθητήρια. Στόχος κατά την ανάπτυξη ενός ρομποτικού συστήματος είναι ή όσο το δυνατόν μεγαλύτερη απλότητα στη συνδεσιμότητα, υλοποίηση και χρήση. Επιθυμητή είναι και η προσέγγιση του αρθρωτού σχεδιασμού [4].

3.3.1 Αισθητήρας (LIDAR (Light Detection and Ranging)

Ο αισθητήρας Lidar χρησιμοποιεί ακτίνες λέιζερ, για την δημιουργία μίας 3D αναπαράστασης του περιβάλλοντος που σαρώνεται. Το Lidar χρησιμοποιείται σε πολλές βιομηχίες, όπως είναι η αυτοκινητοβιομηχανία, η βιομηχανία υποδομών, στην ρομποτική, σε φορτηγά, μη επανδρωμένα σκάφη[1], σε χαρτογραφήσεις κ.α. Επειδή το Lidar χρησιμοποιεί την δική του πηγή φωτός, η τεχνολογία του προσφέρει μεγάλη απόδοση σε διαφορετικές συνθήκες φωτισμού και καιρού. Ένα τυπικό Lidar εκπέμπει παλμούς κυμάτων φωτός στο περιβάλλον. Αυτοί οι παλμοί ανακλώνται σε αντικείμενα του περιβάλλοντος και επιστρέφουν στον αισθητήρα. Το χρόνικο διαστημα που κάθε παλμός κάνει να επιστρέψει στον αισθητήρα χρησιμοποιείται για την μέτρηση της απόστασης που διανύθηκε. Επαναλαμβάνοντας αυτή τη διαδικασία εκατομμύρια φορές το δευτερόλεπτο είναι δυνατή η δημιουργία ενός τρισδιάστατου χάρτη του περιβάλλοντος σε πραγματικό χρόνο [8].

[1] Drones

Slamtec RPLidar A1

Το RPLidar A1 βασίζεται στην αρχή τριγωνοποίησης εύρους (laser triangulation ranging principle). Ο πυρήνας του περιστρέφεται δεξιόστροφα και πραγματοποιεί σάρωση του περιβάλλοντος προς κάθε κατεύθυνση (360°) και στη συνέχεια μετά από επεξεργασία των δεδομένων παράγεται ένας χάρτης. Ο ρυθμός δειγμάτων ενός Lidar είναι αυτός απο τον οποίο εξαρτάται αν το ρομπότ μπορεί να χαρτογραφήσει γρήγορα και με ακρίβεια. Το RPLidar βελτιώνει το εσωτερικό οπτικό σύστημα σχεδιασμού και αλγορίθμου έτσι ώστε ο ρυθμός δείγματος να είναι έως και 8000 φορές το δευτερόλεπτο. Ο αισθητήρας παρουσιάζεται στο Σχήμα 3.7 ενώ τα χαρακτηριστικά του στον Πίνακα 3.6

Σχήμα 3.7: Slamtec RPLidar A1 [9]

Πίνακας 3.7: Χαρακτηριστικά του Slamtec RPLidar A1

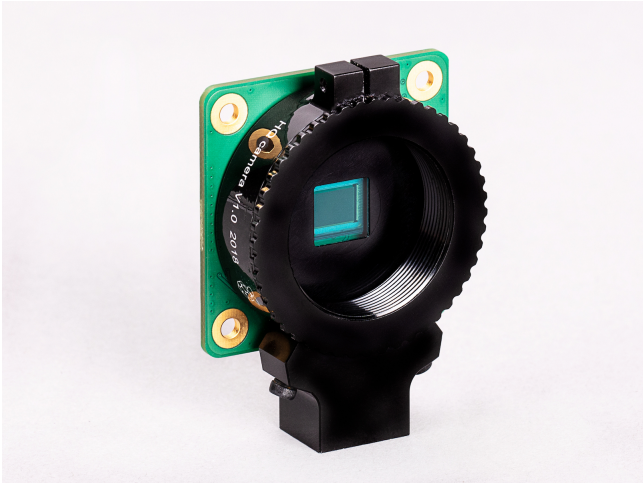
Εύρος ανίχνευσης	12 m
Γωνιακό εύρος	360°
Ρυθμός δειγμάτων	8,000 δείγματα/sec
Γωνιακή ακρίβεια	1°
Ακρίβεια απόστασης	0.2 cm (αν ο ρυθμός σάρωσης είναι ορισμένος στα 5.5Hz, η ακρίβεια είναι 0.2% της πραγματικής απόστασης)
Διαστάσεις	ύψος 55 mm,πλάτος 70.3 mm, βάθος 96.8 mm
Τάση συστήματος	5 V
Ρεύμα συστήματος	100 mA
Κατανάλωση Ισχύος	0.5 W
Συμβατότητα με ROS (Robot Operating System)	-

3.3.2 Αισθητήρας Εικόνας (Image Sensor)

Ο αισθητήρας εικόνας είναι μία ηλεκτρονική συσκευή η οποία μετατρέπει μία οπτική εικόνα σε ηλεκτρονικό σήμα. Χρησιμοποιείται σε ψηφιακές κάμερες και συσκευές απεικόνισης με σκοπό την μετατροπή του φωτός που εισέρχεται στον φακό της συσκευής απεικόνισης σε ψηφιακή εικόνα. Ο αισθητήρας εικόνας χρησιμοποιείται κυρίως σε αυτόνομες ή ενσωματωμένες ψηφιακές κάμερες και συσκευές απεικόνισης. Τυπικά, όταν φως εισέρχεται στον φακό του αισθητήρα, αυτός δέχεται και μετατρέπει το φως σε ηλεκτρονικό σήμα. Το σήμα αυτό αποστέλεται στον επεξεργαστή της συσκευής, ο οποίος μετατρέπει το ηλεκτρονικό σήμα σε ψηφιακή εικόνα. Υπάρχουν δύο κύριοι τύποι αισθητήρων εικόνας οι CCD (Charged Coupled Device) και οι CMOS (Complementary Metal Oxide Semiconductor). Η κύρια διαφορά τον δύο τύπων είναι ότι, οι αισθητήρες CCD παράγουν εικόνες με λιγότερο θόρυβο από ότι οι αισθητήρες CMOS [10].

Raspberry Pi HQ Camera

Η Raspberry Pi HQ Camera (Σχήμα 3.8) είναι απο τις νεότερες συσκευές εικονοληψίας της εταιρείας Raspberry Pi. Παρέχει βελτιωμένες δυνατότητες σε σχέση με τους προκάτοχους της όπως, υψηλότερη ανάλυση και ευαισθησία, κι έχει σχεδιαστεί να λειτουργεί με εναλλάξιμους φακούς τόσο σε C-mount όσο και σε CS-mount. Τα χαρακτηριστικά της συσκευής παρουσιάζονται στον Πίνακα 3.7.



Σχήμα 3.8: Raspberry Pi HQ Camera [5]

Πίνακας 3.8: Χαρακτηριστικά του Raspberry Pi HQ Camera

Αισθητήρας	Sony IMX477R
Ανάλυση	12.3 megapixels
Διαγώνιος φακού	7.9 mm
Μέγεθος pixel	1.55μm x 1.55μm
Συμβατοί τύποι φακών	C-mount/CS-mount
Δυνατότητα εναλλαγής φακών	-

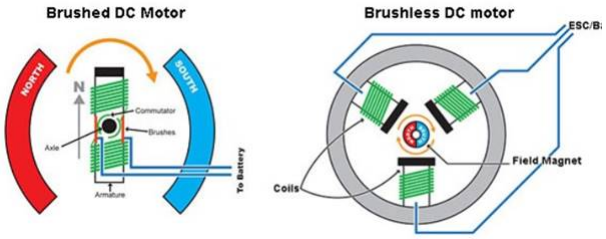
3.4 Κινητήρες

3.4.1 Brushed DC κινητήρες

Ένα από τα απλούστερα είδη κινητήρων, οι Brushed DC κινητήρες είναι πιθανώς ο πιο δημοφιλής τύπος κινητήρα της αγοράς. Μπορούν να βρεθούν σε ποικιλία μεγεθών, από μικροσκοπικοί, για συσκευές μινιατούρες, μέχρι και μεγάλοι, βιομηχανικού τύπου κινητήρες. Χρησιμοποιούνται σε πολλές εφαρμογές όπως είναι οι ανεμιστήρες χειρός, τα μπάζερ κινητών τηλεφώνων, ή και σε τρυπάνια χωρίς καλώδιο ρεύματος. Χρησιμοποιούν βούρτσες επαφής, οι οποίες συνδέονται με έναν μετατροπέα, ο οποίος αλλάζει την κατεύθυνση του ρεύματος. Ουσιαστικά, οι βούρτσες τρίβονται πάνω σε ένα χωρισμένο σε τμήματα, χάλκινο δαχτυλίδι. Αυτό, έχει ως αποτέλεσμα το ρεύμα που διαρέει τα οπλισμένα πηνία να είναι εναλλασσόμενο ακόμα και καθώς ο κινητήρας περιστρέφεται. Είναι φθηνοί στην παραγωγή τους, ελαφροί κι εύκολοι στο να ελεγθούν. Αρκετά αποτελεσματικοί κι έχουν την απαραίτητη ροπή σε χαμηλές ταχύτητες. Το μειονέκτημα τους είναι ότι, όχι μόνο οι ΄βούρτσες' του μετατροπέα παράγουν θόρυβο, αλλά ότι παράγουν και ηλεκτρικό θόρυβο ο οποίος δημιουργεί παρεμβολές σε άλλα κυκλώματα και προκαλούνται προβλήματα στην λειτουργία του συστήματος [11].

3.4.2 Brushless DC κινητήρες

Οι Brushless DC κινητήρες χρησιμοποιούν μόνιμους μαγνήτες στην συναρμογή του ρότορα τους και για τον λόγο αυτό, μηχανικά είναι απλούστεροι απο τους Brushed DC κινητήρες. Στην περίπτωση τους, δεν υπάρχει παραγωγή θορύβων και η εναλλαγή της ροής του ρεύματος για την κίνηση του κινητήρα γίνεται αθόρυβα λόγω της απουσίας των ΄βουρτσών΄. Οι αθόρυβοι αυτοί κινητήρες χρησιμοποιούνται σε ανεμιστήρες υπολογιστών, σκληρούς δίσκους, ηλεκτρικά οχήματα, και σερβομηχανισμούς μεγάλης ακρίβειας. Για τον λόγο αυτό, έχουν μεγάλη ζήτηση στην αγορά των χόμπι για αεροσκάφοι, ελικόπτερα, ραδιοελεγχόμενα οχήματα, βιομηχανικούς σερβομηχανισμούς, υβριδικά οχήματα και σε εφαρμογές οχημάτων εδάφους. Έχουν μεγάλη απόδοση, χρειάζονται λιγότερη συντήρηση, όπως προαναφέρθηκε, παράγουν λιγότερο θόρυβο κι έχουν μεγαλύτερη πυκνότητα δύναμης σε σχέση με τους Brushed DC κινητήρες. Επειδή μπορούν να παράγονται μαζικά, το μειονέκτημα τους είναι ότι για την κατασκευή τους απαιτούνται ειδικοί ρυθμιστές, ελεγκτές και κιβώτια ταχυτήτων το οποίο οδηγεί σε υψηλότερο κόστος [11].



Σχήμα 3.9: Οι δύο τύποι κινητήρων [12]

3.5 Άλλα απαραίτητα εξαρτήματα και τεχνολογίες

3.5.1 Οδόμετρο

Ένα οδόμετρο ή οδογράφος είναι ένα εργαλείο μέτρησης της διανυθείσας απόστασης ενός οχήματος. Η συσκευή μπορεί να είναι ηλεκτρονική-ψηφιακή, μηχανική-αναλογική ή και ένας συνδυασμός και των δύο. Στις κατασκευές αυτοκινήτων ρομπότ, οδόμετρα είναι, τις περισσότερες φορές, εγκατεστημένα στους DC κινητήρες που χρησιμοποιούνται. Για την καταγραφή της απόστασης, τα σύγχρονα οδόμετρα, χρησιμοποιούν τσιπ (ηλεκτρονικά-ψηφιακά) αντί των καλωδίων και γραναζιών που χρησιμοποιούνταν παλαιότερα (μηχανικά-αναλογικά). Χρησιμοποιούν, επίσης, ένα μαγνητικό ή οπτικό αισθητήρα ο οποίος καταμετράει παλμούς του τροχού στον οποίο είναι συνδεδεμένα. Τα δεδομένα αυτά αποθηκεύονται και χρησιμοποιούνται για τον υπολογισμό της διανυθείσας απόστασης του οχήματος. Η χρήση ενός οδομέτρου είναι ο μόνος τρόπος, με τον οποίο ένας χρήστης μπορεί να υπολογίσει την απόσταση που έχει διανύση ένα όχημα, είτε αυτό είναι εκπαιδευτικό έντροχο ρομπότ, είτε αυτοκίνητο.

Κεφάλαιο 4

Σχεδιασμός και Κατασκευή του Ρομποτικού Οχήματος

Στο παρακάτω κεφάλαιο παρουσιάζεται η συνολική σχεδίαση του ρομποτικού οχήματος. Γίνεται μια αναφορά στο αναγκαίο θεωρητικό υπόβαθρο, τα επιλεχθέντα μέρη που προέκυψαν από την έρευνα αγοράς του προηγούμενου κεφαλαίου, τα τμήματα που αναπτύχθηκαν μέσω σχεδιασμού σε περιβάλλον υπολογιστή (CAD) και πρωτοτυποποιήθηκαν με χρήση 3D εκτυπωτή, καθώς επίσης και το σχηματικό σύνδεσης των συσκευών. Η κατάσταση που επικράτησε μετά την πανδημία του Covid-19 επηρέασαν αρκετά την επιλογή των μερών από τα οποία θα αποτελείται η κατασκευή, λόγω ελλείψεων στην αγορά ή μεγάλων χρόνων αναμονής των παραγγελιών. Ωστόσο, παρακάτω φαίνονται τα τελικά επιλεχθέντα τμήματα που αποτελούσαν υποψήφιες επιλογές για την ανάπτυξη του. Στο τέλος του κεφαλαίου παρουσιάζεται το τελικό πρωτότυπο του έντροχου ρομποτικού οχήματος.

4.1 Θεωρητικό Υπόβαθρο

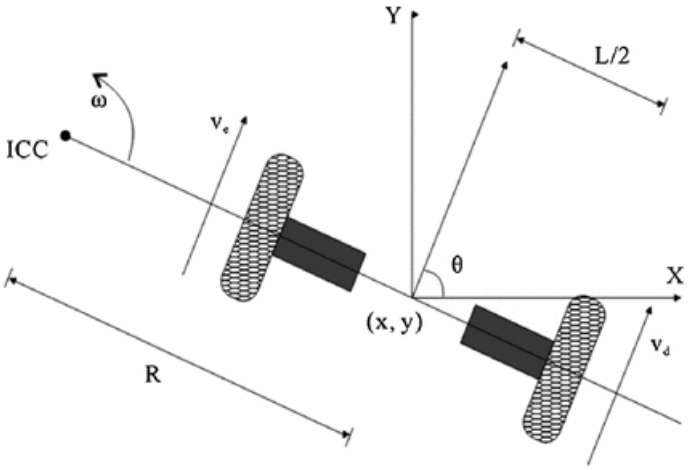
4.1.1 Κινηματικό Μοντέλο Διαφορικής Οδήγησης

Μια από τις δημοφιλέστερες τεχνολογίες για την διαμόρφωση ρομποτικών οχημάτων είναι αυτή της διαφορικής οδήγησης. Ο μηχανισμός αυτός αποτελείται απο δύο κινητήριους τροχούς τοποθετημένους σε κοινό άξονα, όπου ο κάθε τροχός μπορεί να περιστρέφεται ανεξάρτητα είτε μπρος, είτε πίσω. Παρά το γεγονός ότι η ταχύτητα του κάθε τροχού μπορεί να ποικίλλει, για να μπορεί το ρομπότ να εκτελέσει περιστροφική κίνηση, θα πρέπει πάντοτε να περιστρέφεται γύρω απο ένα σημείο το οποίο βρίσκεται κατά μήκος του κοινού άξονα των τροχών του. Το σημείο αυτό ονομάζεται **Στιγμιαίο Κέντρο Στροφής** ή ICC[1] (Σχήμα 4.1). Η επιλογή του συγκεκριμένου μοντέλου στην προκειμένη περίπτωση έγινε λόγω της απλότητας του, της ευκολίας ανάπτυξης των αλγορίθμων για τον έλεγχο του, της δυνατότητας που παρέχει για επιτόπια περιστροφή, όπως επίσης για τον λόγω ότι απαιτεί την χρήση δύο μόνο κινητήρων (έναν για τον εκάστοτε τροχό) για την λειτουργία του [13]. Συγκεκριμένα:

- ICC, το Στιγμιαίο Κέντρο Στροφής,
- ω, η γωνιακή ταχύτητα,

[1]Instantaneous Center of Curvature

- V_e, V_d, η ταχύτητα του αριστερού και του δεξιού τροχού αντίστοιχα,
- R, η απόσταση του ICC από το μέσο της απόστασης μεταξύ των δύο τροχών,
- L, η απόσταση μεταξύ των δύο τροχών,
- (x, y), η θέση του ρομπότ στο σύστημα συντεταγμένων αναφοράς,
- θ, η γωνία που έχει στραφεί το όχημα σε σχέση με το σύστημα συντεταγμένων αναφοράς.



Σχήμα 4.1: Κινηματικό Μοντέλο Διαφορικής Οδήγησης

Μεταβάλοντας της ταχύτητες των δύο τροχών, μπορούν να μεταβληθούν οι τροχές οι οποίες θα ακολουθήσει το ρομπότ. Επειδή η γωνιακή ταχύτητα ω σε σχέση με το ICC πρέπει να είναι η ίδια και για τους δύο τροχούς, μπορούν να γραφούν οι παρακάτω εξισώσεις:

$$\omega(R + \frac{L}{2}) = V_d \tag{4.1}$$

$$\omega(R - \frac{L}{2}) = V_e \tag{4.2}$$

Λύνοντας τις σχέσεις 4.1 και 4.2 ως προς τις μεταβλητές R και ω, προκύπτουν οι παρακάτω εξισώσεις:

$$R = \frac{L}{2}\frac{V_e + V_d}{V_d - V_e} \tag{4.3}$$

$$\omega = \frac{V_d + V_e}{L} \tag{4.4}$$

Υπάρχουν τρείς ενδιαφέρουσες περιπτώσεις που προκύπτουν απο τις παραπάνω εξισώσεις:

- Εάν $V_e = V_d$, τότε το όχημα θα κινείται είτε μπρος (αν είναι θετικές) είτε πίσω (αν είναι αρνητικές) σε ευθεία γραμμή. Η απόσταση R γίνεται άπειρη και ουσιαστικά δεν παρουσιάζεται περιστροφή, δηλαδή η γωνιακή ταχύτητα *omega* είναι ίση με μηδέν.

- Εάν $V_e = -V_d$, τότε η απόσταση R γίνεται μηδέν, και υπάρχει περιστροφή γύρω από το μέσο του άξονα των τροχών και συνεπώς επιτόπια περιστροφή.
- Εάν $V_e = 0$ ή $V_d = 0$, τότε ο ένας τροχός παραμένει ακίνητος και η περιστρογη γίνεται γύρω απο αυτόν. Στην περίπτωση αυτή η απόσταση $R = -L/2$ ή $R = L/2$ αντίστοιχα.

Σημαντική σημείωση είναι ότι ένα ρομπότ διαφορικής οδήγησης δεν μπορεί να κινηθεί στην κατεύθυνση του άξονα των τροχών του. Τα οχήματα διαφορικής οδήγησης παρουσιάζουν ιδιαίτερη ευαισθησία σε μικρές αλλαγές της ταχύτητας του εκάστοτε τροχού. Μικρά σφάλματα στις σχετικές ταχύτητες μεταξύ των τροχών μπορούν να επηρεάσουν την τροχιά του ρομπότ. Επιπλέον, είναι ιδιαίτερα ευαίσθητα σε μικρές μεταβολές της επιφάνειας του εδάφους, και πιθανόν να χρειάζονται επιπλέον τροχούς για υποστήριξη.

4.1.2 Ευθύ Κινηματικό Μοντέλο

Με το ευθύ κινηματικό μοντέλο μπορεί να υπολογιστεί η νέα θέση του ρομποτικού οχήματος, έχοντας ως δεδομένο την γωνιακή του ταχύτητα *omega*. Αν το ρομποτικό όχημα βρίσκεται την χρονική στιγμη t στη θέση (x, y) με γωνία στροφής ω, όπως φαίνεται και στο σχήμα 4.1. Τότε, από τις Εξισώσεις 4.3 και 4.4 μπορεί να υπολογιστεί η θέση του ICC όπως φαίνεται παρακάτω:

$$ICC = [x - Rsin\theta, y + Rcos\theta] \tag{4.5}$$

και την χρονική στιγμή $t+\delta t$ το όχημα θα βρίσκεται στην θέση (x', y') υπό γωνία $theta'$ με γωνιακή ταχύτητα ω. Η νέα αυτη θέση υπολογίζεται ως εξής:

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} cos(\omega \cdot \delta t) & -sin(\omega \cdot \delta t) & 0 \\ sin(\omega \cdot \delta t) & cos(\omega \cdot \delta t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x - ICC_x \\ y - ICC_y \\ \theta \end{bmatrix} + \begin{bmatrix} ICC_x \\ ICC_y \\ \omega \cdot \delta t \end{bmatrix} \tag{4.6}$$

4.1.3 Αντίστροφο Κινηματικό Μοντέλο

Στο αντίστροφο κινηματικο μοντέλο λαμβάλει χώρα η αντίστροφη διαδικασία από αυτή του ευθέως κινηματικού. Έχοντας δηλαδή ως δεδομένη νέα την θέση (x, y) του οχήματος, μπορεί να υπολογιστεί η γωνιακή ταχύτητα έτσι ώστε το όχημα να φτάσει σε αυτή την θέση. Γενικά, η θέση ενός κινούμενου ρομπότ με συγκεκριμένη κατευθυνση $\Theta(t)$, δεδομένη ταχύτητα $V(t)$ και υπο γωνία $\theta(t)$ μπορεί να υπολογιστεί ως εξής:

$$x(t) = \int_0^t V(t)cos[\theta(t)]dt \tag{4.7}$$

$$y(t) = \int_0^t V(t)sin[\theta(t)]dt \tag{4.8}$$

$$\Theta(t) = \int_0^t \omega(t)dt \tag{4.9}$$

Στις περιπτώσεις των ρομπότ διαφορικής οδήγησης, η παραπάνω εξισώσεις 4.7, 4.8 και 4.9 γίνονται:

$$x(t) = \frac{1}{2}\int_0^t [V_e(t) + V_d(t)]cos[\theta(t)]dt \tag{4.10}$$

$$y(t) = \frac{1}{2}\int_0^t [V_e(t) + V_d(t)]sin[\theta(t)]dt \tag{4.11}$$

$$\Theta(t) = \frac{1}{L}\int_0^t [V_e(t) - V_d(t)]dt \tag{4.12}$$

Στις ιδιαίτερες περιπτώσεις όπου οι γραμμικες ταχύτητες των τροχών είναι ίσες ή αντίθετες μεταξύ τους (και διάφορες του μηδενός). Οι εξισώσεις κίνησης παίρνουν την παρακάτων μορφή:

- $V_d = V_e = V$, για γραμμική κίνηση:

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x + Vcos(\theta)\delta t \\ y + Vsin(\theta)\delta t \\ \theta \end{bmatrix} \tag{4.13}$$

- $V_d = -V_e = V$, για περιστροφική κίνηση γύρω από τον εαυτό του:

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta + \frac{2V\delta t}{L} \end{bmatrix} \tag{4.14}$$

Από τις δύο αυτές περιπτώσεις προκύπτει μία στρατηγική κίνησης, για τα ρομπότ διαφορικής κίνησης, η οποία αποτελείται απο τα εξής επαναλαμβανόμενα βήματα:

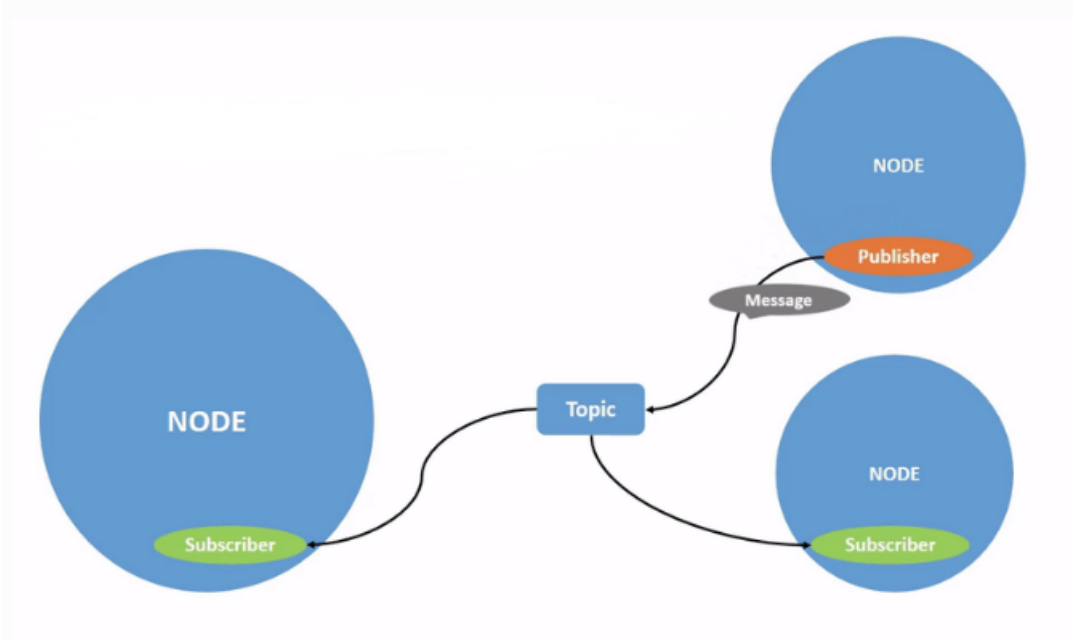
1. Κίνηση σε ευθεία γραμμή (είτε εμπρός είτε πίσω)
2. Επιτόπια περιστροφή

4.2 Robot Operating System (ROS)

Το Robot Operating System (ROS) είναι ένα λογισμικό ανοιχτού κώδικα που παρέχει ενδιάμεσες δομές που ελέγχουν και συντονίζουν τα κατανεμημένα συστήματα και το υλικολογισμικό (firmware), για την ανάπτυξη λογισμικού για ρομπότ. Παρά το γεγονός ότι δεν είναι λειτουργικό σύστημα αλλά ένα σύνολο από δομές, παρέχει χαμηλού επιπέδου έλεγχο του υλισμικού, ένα σύστημα μετάδοσης μηνυμάτων μεταξύ των διεργασιών που εκτελούνται και διαχείριση πακέτων. Παρά την σπουδαιότητα της αντιδραστικότητας και της χαμηλής απόκρισης στον ρομποτικό έλεγχο, το ROS δεν είναι σύστημα ελέγχου πραγματικού χρόνου (Real Time Operating System - ROTS). Η έλλειψη αυτή οδήγησε στην δημιουργία του ROS 2, μία μείζων ενημέρωση του περιβάλλοντος διεπαφής για τον προγραμματισμό εφαρμογών (Application Programming Interface - API) του ROS. Το ROS 2 χρησιμοποιεί σύγχρονες βιβλιοθήκες και τεχνολογίες για τις βασικές λειτουργίες, όπως ακριβώς και το ROS, προσθέτοντας επιπλέον υποστήριξη για αλγορίθμους πραγματικού χρόνου και υλισμικό ενσωματωμένων συστημάτων (empedded systems hardware).

Όσον αφορά τον προγραμματισμό στο περιβάλλον του ROS παρέχετε ευελιξία στην επιλογή της γλώσσας. Αυτό διότι διαθέτει βιβλιοθήκες για C++, Python, Lisp, Matlab, Java και Javascript, οι οποίες μπορούν να επικοινωνούν μεταξύ τους παρά τις διαφορετικές γλώσσες προγραμματισμού. Με την εγκατάσταση του ROS παρέχονται πολλά χρήσιμα εργαλεία όπως ο προσομοιωτής Gazebo, το εργαλείο διαχείρισης θεμάτων rqt και το εργαλείο οπτικοποίησης του χάρτη, των αισθητήρων και των κινητήρων Rviz.

Η φιλοσοφία λειτουργίας του ROS παρουσιάζεται με μία αρχιτεκτονική διαγράμματος κατά την οποία η επεξεργασία λαμβάνει χώρα σε κόμβους οι οποίοι λαμβάνουν, αναρτούν και συνθέτουν δεδομένα αισθητήρων, έλεγχο, κατάσταση, οργάνωση, ενεργοποιητές κι άλλα μηνύματα. Κάθε διεργασία αποτελεί έναν ξεχωριστό κόμβο. Κάθε κόμβος μπορεί να συνδεθεί με άλλους κόμβους μέσω μηνυμάτων που στέλνονται διαμέσω ορισμένων κοινών (για κάποιους κόμβους) θεμάτων που ονομάζονται topics. Αυτό δίνει την δυνατότητα στον χρήστη να επικοινωνεί με τον εκάστοτε κόμβο δημοσιεύοντας πληροφορίες υπό μορφή των μηνυμάτων αυτών. Με τον τρόπο αυτό αναπτύσσεται η αρχιτεκτονική διαγράμματος που αναφέρθηκε στην οποία ανταλλάσσονται μηνύματα απο κόμβο σε κόμβο. Αναπτύσσεται δηλαδη μία σχέση εξάρτησης των κόμβων της μορφής αποστολέα (publisher) - συνδρομητή (subscriber) όπως φαίνεται και στο σχήμα 4.2 [14].



Σχήμα 4.2: Γραφική απεικόνιση ρομποτικής εφαρμογής με τρείς κόμβους. Ο πάνω δεξιά κόμβος έχει τον ρόλο του αποστολέα (publisher) του μηνύματος και οι δύο άλλοι τον ρόλο του συνδρομητή (subscriber). [14]

4.3 Επιλεχθέντα Ηλεκτρονικά και Λοιπά Μέρη

4.3.1 Ρομποτικός Ελεγκτής

Η αρχική επιλογή για τον ρομποτικό ελεγκτή ήταν το Jetson Nano της εταιρίας NVIDIA. Απορρίφθηκε ωστόσο, λόγω έλλειψης των ηλεκτρονικών που υπήρξε μετά την περίοδο του Covid-19. Επόμενη και τελική επιλογή ήταν το Raspberry pi 4 του οργανισμού RaspberryPi, ένας ρομποτικός ελεγκτής που χρησιμοποιείται σε παρόμοιες υλοποιήσεις ακόμη και σήμερα (παρά το ότι είναι κατασκευασμένος το 2019), ικανός για την πραγματοποίηση της συγκεκριμένης εργασίας. Η προτίμηση αυτή έγινε διότι ο συγκεκριμένος ρομποτικός ελεγκτής βρέθηκε διαθέσιμος προς άμεση αγορά, και προς αποφυγή του ρίσκου μη εύρεσης άλλου ελεγκτή λόγω της κατάστασης και των ελλείψεων της αγοράς.

Λειτουργικό Σύστημα

Το λειτουργικό σύστημα που επιλέχθηκε κι εγκαταστάθηκε στον ηλεκτρονικό υπολογιστή που χρησιμοποιήθηκε καθόλη την διάρκεια της εργασίας αλλά και στον Raspberry Pi 4 είναι το Linux. Πιο συγκεκριμένα, χρησιμοποιήθηκε η έκδοση Ubuntu 20.04 η οποία είναι γνωστό ότι έχει δοκιμαστεί και είναι λειτουργική σε συνδυασμό με το λογισμικό ανοιχτού κώδικα ROS 2 και συγκεκριμένα την έκδοση Galactic, η οποία επίσης είναι η έκδοση του ROS 2 που εγκαταστάθηκε. Η επιλογή αυτή έγινε διότι το λειτουργικό Linux είναι λειτουργικό σύστημα ανοιχτού κώδικα, γεγονός το οποίο ήταν απαραίτητο για τις απαιτήσεις της συγκεκριμένης εργασίας, όπως επίσης και λόγω των απαιριόριστων επιλογών που προσφέρει στον χρήστη για την διαμόρφωση του περιβάλλοντος εργασίας. Ακόμα, ένας επιπλέον λόγος είναι το γεγονός ότι δίνει στον χρήστη πολλά εργαλεία, τα οποία σε άλλα λειτουργικά απαιτούν την εγκατάσταση λογισμικών για την παροχή τους, όπως παραδείγματος χάρη τον απομακρυσμένο έλεγχο άλλης συσκευής με το ίδιο λειτουργικό σύστημα, γεγονός το οποίο στην συγκεκριμένη εργασία είναι μεγάλη διευκόλυνση (έλεγχος του Raspberry Pi 4 μέσω του ηλεκτρονικού υπολογιστή εργασίας).

4.3.2 Αισθητήρες

Οι αισθητήρες που επιλέχθηκαν να τοποθετηθούν στην υπό ανάπτυξη συσκευή ήταν ένας αισθητήρας λέιζερ κι ένας αισθητήρας εικόνας. Συγκεκριμένα, ο αισθητήρας λείζερ Slamtec RPlidar A1m8 και ο αισθητήρας εικόνας Raspberry Pi HQ Camera. Ωστόσο, διαπιστώθηκε κατά την πορεία της εργασίας, ότι ο συγκεκριμένος αισθητήρας εικόνας δεν λειτουργεί με το λειτουργικό σύστημα των Ubuntu που επιλέχθηκε να εγκατασταθεί. Είναι σχεδιασμένος έτσι ώστε να λειτουργεί με το λειτουργικό σύστημα Rasbian, δηλαδή το λειτουργικό σύστημα που έχει αναπτύξει η εταιρεία που παράγει τα Raspberry Pi 4 και Raspberry Pi HQ Camera. Αυτό διότι όλες οι βιβλιοθήκες για την χρήση του συγκεκριμένου αισθητήρα είναι αναπτυγμένες για την λειτουργία του στο συγκεκριμένο λειτουργικό. Οπότε, για την χρήση του με το σύστημα Ubuntu θα έπρεπε να αναπτυχθούν οι βιβλιοθήκες για την λειτουργία του αισθητήρα, πράγμα που δεν ήταν αντικείμενο της παρούσας εργασίας. Έτσι ο αισθητήρας που τοποθετήθηκε τελικά είναι μόνο ο αισθητήρας λείζερ Slamtec RPlidar A1m8.

Λειτουργία αισθητήρα λέιζερ

Για την λειτουργία του αισθητήρα λέιζερ, αρχικά έγινε προσπάθεια ανάπτυξης κόμβου ROS 2 ο οποίος θα στέλνει τις μετρήσεις (απόσταση του αντικειμένου και την γωνία στην οποία βρίσκεται αυτό) του αισθητήρα στον ρομποτικό ελεγκτή μέσω ενός ορισμένου topic (θέματος) και αυτός με τη σειρά του θα μπορούσε να τις χρησιμοποιήσει αναλόγως. Ωστόσο, κατά την ανάπτυξη αυτού του κόμβου παρουσιάστηκαν πολλά εμπόδια. Για τον λόγο αυτό, βρέθηκε και τελικά εγκαταστάθηκε έτοιμο πακέτο από το διαδίκτυο το οποίο πραγματοποιεί την παραπάνω λειτουργία. Το πακέτο αυτό είναι το rplidar ros2 με δημιουργό τον χρήστη CreedyNZ του Github. Το παρόν πακέτο επιστρέφει στην οθόνη τις παραπάνω μετρήσεις (μέχρι να τερματιστεί το πρόγραμμα απο το πληκτρολόγιο) που αναφέρθηκαν και επιπλέον δίνει την δυνατότητα για απεικόνιση των συγκεκριμένων μετρήσεων στο πρόγραμμα Rviz 2. Απεικονίζει δηλαδή την σχετική θέση των αντικειμένων που ΄βλέπει΄ ο αισθητήρας σε πραγματικό χρόνο, για 360° σαν να παρατηρούσε ο χρήστης από το πάνω μέρος ενός κουκλόσπιτου. Έπειτα, αναπτύχθηκε κόμβος ROS 2 ο οποίος παίρνει τις μετρήσεις (απόσταση και γωνία του εμποδίου) και τις αποθηκεύει σε αρχείο κειμένου σβήνοντας κάθε φορά τις μετρήσεις του προηγούμενου ελέγχου.

4.3.3 Συσσωρευτές

Για την παροχή ενέργειας στις συσκευές της κατασκευής, αποφασίστηκε ότι η καλύτερη επιλογή ήταν οι συσσωρευτές τύπου 18650 λιθίου-ιόντων (Li-ion). Η επιλογή αυτή έγινε διότι ο τύπος συσσωρευτών αυτός είναι διαθέσιμος στην παγκόσμια αγορά και είναι επαναφορτιζόμενοι. Επιπλέον, προσφέρει σημαντικά πλεονεκτήματα σε σχέση με άλλους τύπους επαναφορτιζόμενων, όπως είναι οι συσσωρευτές νικελίου-καδμίου (NiCad), οι νικελίου-μετάλλου-υδριδίου (NiMH) και οι μολύβδου (PD), οι οποίοι είναι ευρέως διαδεδομένοι σε ανάλογες εφαρμογές ρομποτικής. Αρχικά, οι λιθίου-ιόντων έχουν ανοχές στην ένταση του ρεύματος φόρτισης τους, οι οποίες τους επιτρέπουν και ταχύτερη φόρτιση. Παραδείγματος χάρη αν ένας συσσωρευτής απαιτεί 1A ρεύματος για την φόρτιση του και του δωθούν 10A η φόρτιση απλά θα ολοκληρωθεί σε μικρότερο χρόνο, χωρίς επιπτώσεις στην ακεραιότητα του και χωρίς να υπερθερμανθούν (ωστόσο, ίσως υπάρξει μείωση στον αριθμό κύκλων φόρτισης τους). Αντίθετα, οι υπόλοιποι

τύποι συσσωρευτών που αναφέρθηκαν, σε αντίστοιχη περίπτωση θα καταστρέφονταν. Επιπρόσθετα, είναι ακίνδυνοι απέναντι στους χρήστες καθώς δεν περιέχουν τοξικά υλικά και προσφέρουν ευκολία στην αντικατάσταση τους λόγω του μικρού μεγέθους και βάρους τους.

Στην παρούσα εργασία συγκεκριμένα, χρησιμοποιήθηκαν 6 συσσωρευτές Efest IMR 18650 3000mAh 35A flat top battery των οποίων τα τεχνικά χαρακτηριστικά καλύπτουν επαρκώς τις ανάγκες για ισχύ της υπό κατασκευή συσκευής (ζητούμενο της εργασίας ήταν η συσκευή να μπορεί να καλύψει με μία φόρτιση διάρκεια δύο διδακτικών ωρών).

Σχήμα 4.3: Συσσωρευτής Efest IMR 18650 3000mAh 35A με επίπεδη επιφάνεια [12]

Οι 3 συσσωρευτές χρησιμοποιήθηκαν για την παροχή ισχύος του RPi 4 (15.3W - μέγεθος το οποίο πάρθηκε από τον συνιστώμενο φορτιστή της ίδιας εταιρείας) και του RPlidar (0.5W - μέγεθος που βρέθηκε στην σελίδα του προϊόντος). Αφήνοντας και μία μικρή ανοχή, είναι απαραίτητα περίπου 16W ενέργειας για τη λειτουργία των συγκεκριμένων συσκευών. Οι συσσωρευτές τοποθετούνται σε σειρά οπότε, για την διάταξη που προκύπτει προστίθενται οι τάσεις τους (3.7 V) και συνεπώς η διάταξη παρέχει 11.1V. Αν διαιρεθούν τα απαιτούμενα 16 W με την τάση της διάταξης (11.1V) προκύπτει:

$$\frac{16W}{11.1V} = 1.44A \tag{4.15}$$

Οπότε, είναι απαραίτητα 1.44A ανά ώρα με την συγκεκριμένη διάταξη συσσωρευτών. Επιπλέον, τα 3000 mAh των συσσωρευτών δίνουν ουσιαστικά 3A ανά ώρα. Διαιρώντας την παροχή ρεύματος των συσσωρευτών με την ζητούμενη παροχή ρεύματος των συσκευών προκύπτει:

$$\frac{3}{1.44} = 2.08 \tag{4.16}$$

Συνεπώς, μία πλήρης φόρτιση των συγκεκριμένων 3 συσσωρευτών θα παρέχει ισχύ στις συσκευές (RPi4 και RPlidar) για 2.08 ώρες πριν την ανάγκη για νέα φόρτιση.

Οι υπόλοιποι 3 συσσωρευτές χρησιμοποιήθηκαν για την παροχή ισχύος στους κινητήρες. Για τους συγκεκριμένους κινητήρες δεν ήταν διαθέσιμα στοιχεία για τη κατανάλωση τους. Συνεπώς, έπρεπε να υπολογιστεί πειραματικά. Επιλέγοντας λοιπόν, την μέγιστη ταχύτητα που μπορούν να αποδώσουν οι κινητήρες (100%), υπολογίστηκε η κατανάλωση για τον έναν εκ των δύο, για δύο διαφορετικά σενάρια. Το πρώτο σενάριο είναι η ελεύθερη κίνηση του άξονα χωρίς εφαρμογή φορτίου. Στο ενδεχόμενο αυτό παρατηρήθηκε ότι για τάση 12.75V, του συσσωρευτή που χρησιμοποιήθηκε στην πειραματική διάταξη, ο κινητήρας κατανάλωνε 0.08A ρεύματος, συνεπώς η κατανάλωση ισχύος είναι:

$$12.75V \cdot 0.08A = 1.02W \tag{4.17}$$

Άρα, 1.02W κατανάλωση από τον εκάστοτε κινητήρα. Δηλαδή, συνολογικά 2.04W και για τους δύο. Με την διάταξη των 3 συσσωρευτών που θα τοποθετηθούν σε σειρά έχουμε:

$$\frac{2.04W}{11.1V} = 0.18A \tag{4.18}$$

και

$$\frac{3}{0.18} = 16.6 \tag{4.19}$$

Επομένως, στο σενάριο όπου οι κινητήρες λειτουργούν στο 100% και δεν έχουν φορτίο στον άξονα τους μία φόρτιση των συσσωρευτών θα μπορεί να τους τροφοδοτεί για 16.6 ώρες.

Στο δεύτερο σενάριο τοποθετήθηκε φορτίο στον άξονα του κινητήρα, τέτοιο ώστε ο άξονας οριακά να μην κινείται. Οπότε, στο ενδεχόμενο αυτό παρατηρήθηκε ότι για τάση 12.6V του συσσωρευτή που χρησιμοποιήθηκε στην πειραματική διάταξη, ο κινητήρας κατανάλωνε 0.7A ρεύματος, συνεπώς η κατανάλωση ισχύος είναι:

$$12.6V \cdot 0.7A = 8.82W \tag{4.20}$$

Άρα, 8.82W κατανάλωση από τον εκάστοτε κινητήρα. Δηλαδή, συνολογικά 17.64W και για τους δύο. Με την διάταξη των 3 συσσωρευτών που θα τοποθετηθούν σε σειρά έχουμε:

$$\frac{17.64W}{11.1V} = 1.59A \tag{4.21}$$

και

$$\frac{3}{1.59} = 1.88 \tag{4.22}$$

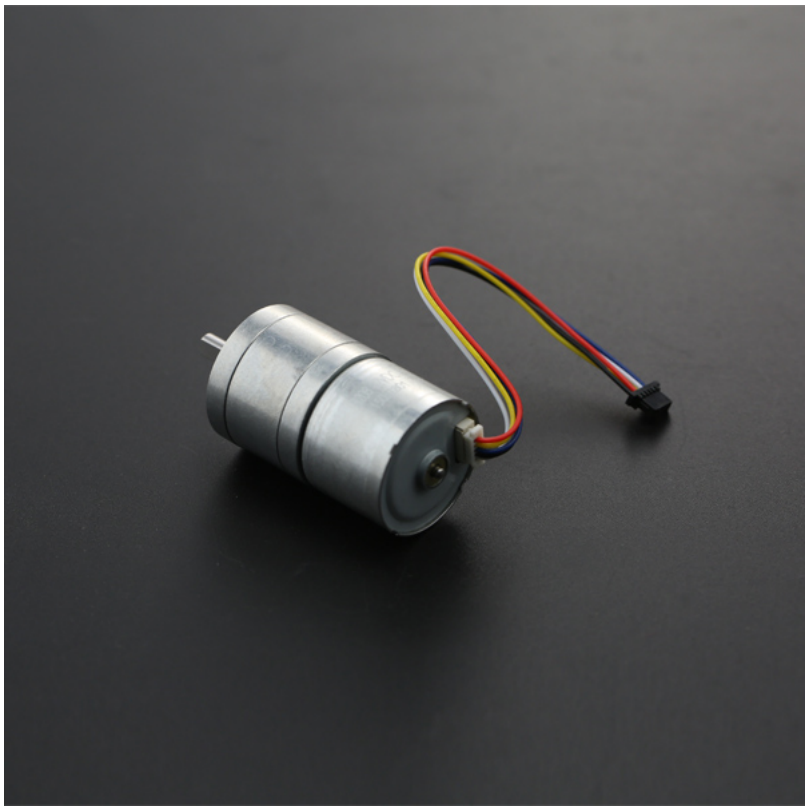
Επομένως, στο σενάριο όπου οι κινητήρες λειτουργούν στο 100% και έχουν φορτίο στον άξονα τους, τέτοιο ώστε να μην μπορεί να κινηθεί η συσκευή, μία φόρτιση των συσσωρευτών θα μπορεί να τους τροφοδοτεί για 1.88 ώρες (Παρατήρηση: στο σενάριο αυτό φαίνεται ότι οι συσσωρευτές δεν θα μπορούν να παρέχουν ισχύ για το ζητούμενο των 2 διδακτικών ωρών. Ωστόσο, το σενάριο αυτό είναι οριακή περίπτωση και η συσκευή δεν πρόκειται να έρθει αντιμέτωπη με αυτήν).

Για την εγκατάσταση των παραπάνω συσσωρευτών στην κατασκευή χρησιμοποιήθηκαν 2 θήκες συσσωρευτών τριών θέσεων, όπως αυτοί που φαίνονται παρακάτω:

Σχήμα 4.4: Θήκες συσσωρευτών [12]

4.3.4 Κινητήρες

Για την κίνηση της κατασκευής επιλέχθηκαν 2 FIT0441 Brushless DC κινητήρες της εταιρείας DFRobot. Η επιλογή του συγκεκριμένου κινητήρα έγινε καθώς έχει προεγκατεστημένο σύστημα οδήγησης (motor driver). Επιπλέον, παρέχει έλεγχο κατεύθυνσης, έλεγχο PWM ταχύτητας και έξοδο ταχύτητας ανάδρασης (speed feedback output). Τα παραπάνω καθιστούν τον έλεγχο των κινητήρων εύκολο, καθώς επίσης ιδανικό και βολικό για την χρήση του σε κινητές ρομποτικές πλατφόρμες.



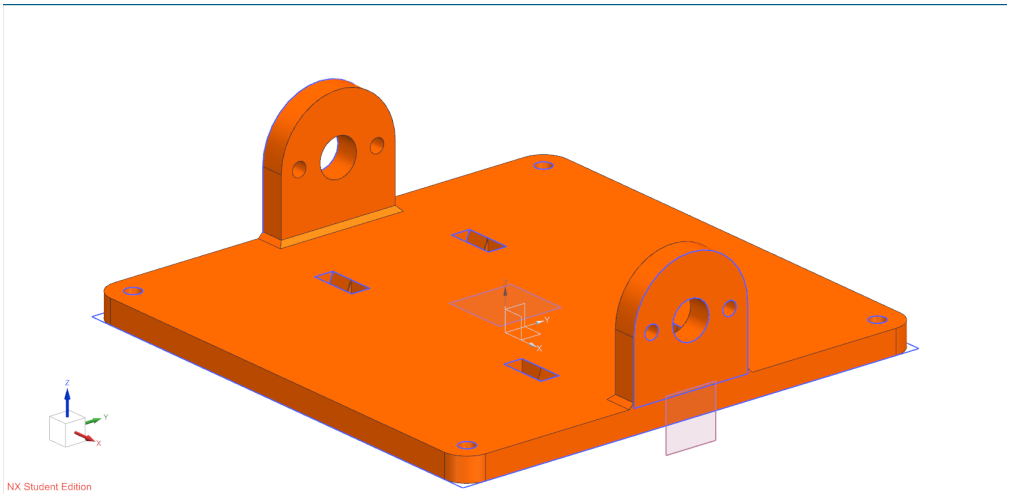
Σχήμα 4.5: DFRobot FIT0441 Brushless DC Κινητήρας [15]

4.4 CAD

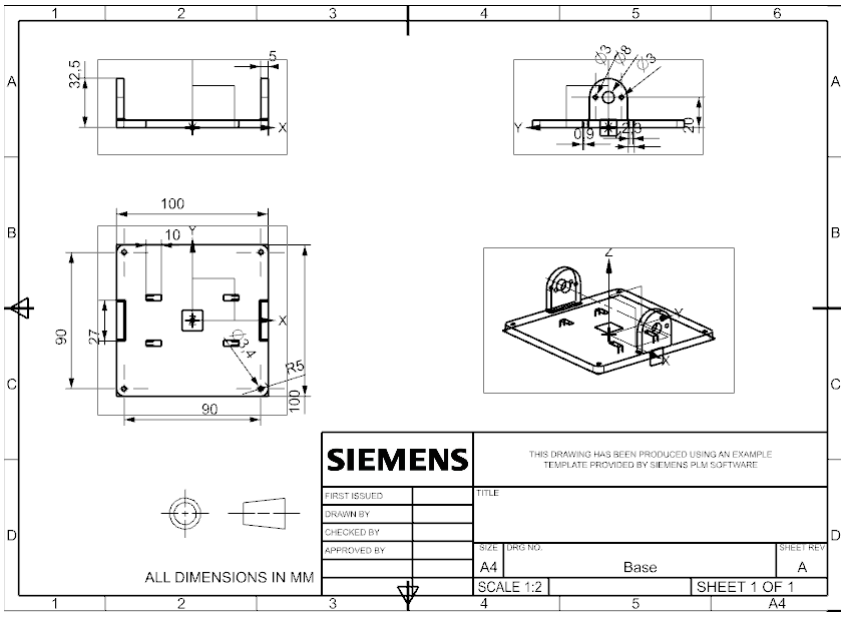
Απαραίτητη για την λειτουργία της πλατφόρμας που αναπτύχθηκε ήταν η χρήση κάποιας μορφής βάσεων, η οποία θα υποστήριζε τα εξαρτήματα που επιλέχθηκαν. Από την πρώτη στιγμή, αποφασίστηκε ότι είναι προτιμότερο, οι βάσεις αυτές να σχεδιαστούν αντί να αγοραστούν, από τις διαθέσιμες προτάσεις που υπάρχουν. Ωστόσο να παραμείνει ο σχεδιασμός τους απλός και τέτοιος ώστε να επιτρέπει την δομοστοιχειωτή (modular) κατασκευή του. Αναπτύχθηκαν λοιπόν, με την βοήθεια του λογισμικού Siemens NX, ορισμένες προτάσεις. Άλλες από αυτές απορρίφθηκαν και διαφοροποιήθηκαν, κι άλλες διατηρήθηκαν ή εξελίχθηκαν. Παρακάτω παρουσιάζεται η τελική μορφή των σχεδίων που αναπτύχθηκαν με βάση την επιλογή των ηλεκτρονικών στοιχείων που παρουσιάστηκαν παραπάνω.

4.4.1 Κάτω Βάση

Η κάτω βάση σκοπό έχει την στήριξη όλου του ρομπότ. Στα δύο υπερυψωμένα, ημικυκλικά τμήματα της θα προσαρτηθούν οι κινητήρες του αυτοκινήτου ρομπότ. Η θέση τους επιλέχθηκε να είναι στην μέση της βάσης για την αποφυγή θεμάτων ανατροπής κατά την λειτουργία του. Η τέσσερις παραλληλόγραμμες σχισμές χρησιμεύουν στην πρόσδεση των στηριγμάτων που σχεδιάστηκαν και παρουσιάζονται παρακάτω (Σχήμα 4.10). Οι σχισμές φέρουν μέχρι το ήμισυ τους τριγωνικές διατάξεις και από τις δύο πλευρές, στις οποίες θα συρθεί η αντίστοιχη διάταξη των στηριγμάτων και θα κουμπώσει στην κάτω βάση. Το λεπτομερές μηχανολογικό σχέδιο της κατασκευής παρουσιάζεται στο Σχήμα 4.7.



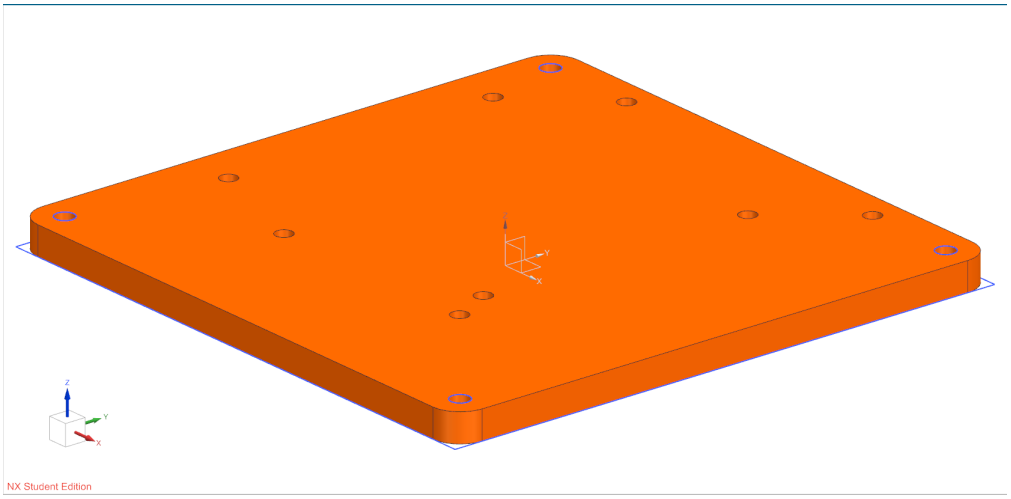
Σχήμα 4.6: Κάτω Βάση



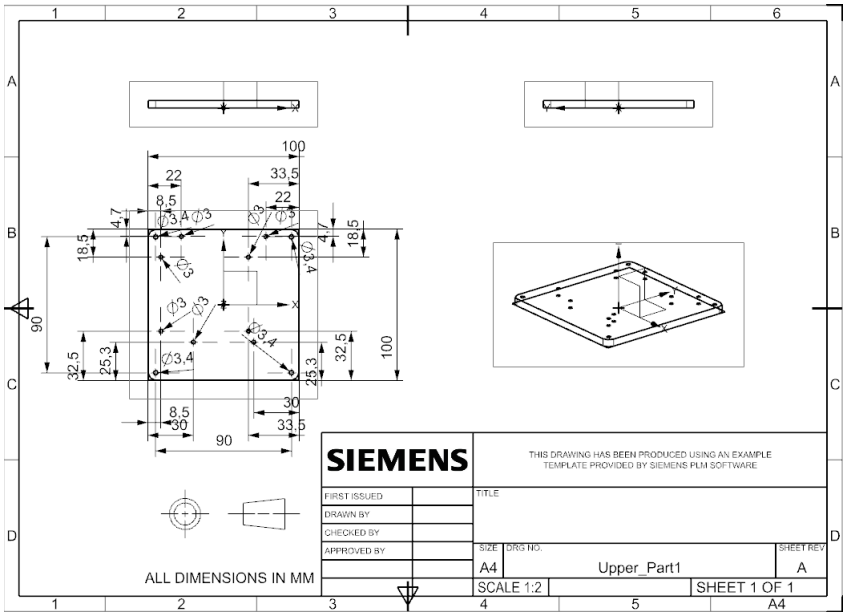
Σχήμα 4.7: Μηχανολογικά Σχέδια Κάτω Βάσης

4.4.2 Πάνω Βάσεις

Ο σχεδιασμός του συγκεκριμένου κομματιού στηρίχθηκε στο γεγονός ότι ήταν απαραίτητη η δομοστοιχειωτή κατασκευή του ρομπότ. Έτσι λοιπόν, η βάση αυτή παρέμεινε απλή στον σχεδιασμό της, τοποθετόντας πάνω της μόνο τις απαραίτητες οπές για την στήριξη του ρομποτικού ελεγκτή και του αισθητήρα λείζερ που επιλέχθηκαν. Ωστόσο, ο σχεδιασμός αυτος δίνει την δυνατότητα στήριξης είτε και των δύο εξαρτημάτων ταυτόχρονα σε ένα μόνο επίπεδο είτε σε δύο ξεχωριστά επίπεδα. Επιπλέον δίνει την δυνατότητα στον χρήστη τοποθέτησης κάποιου τρίτου εξαρτήματος, σε νέο επίπεδο, σε περίπτωση εξέλιξης της παρούσας εργασίας. Με άλλα λόγια, η παρούσα βάση θα αποτελέσει ένα στοιχείο που ανάλογα με τις απαιτήσεις του χρήστη θα μπορεί να του δίνει την δυνατότητα διαμόρφωσης του ρομπότ, όπως αυτός επιθυμεί. Το λεπτομερές μηχανολογικό σχέδιο της κατασκευής παρουσιάζεται στο Σχήμα 4.9.



Σχήμα 4.8: Πάνω Βάσεις

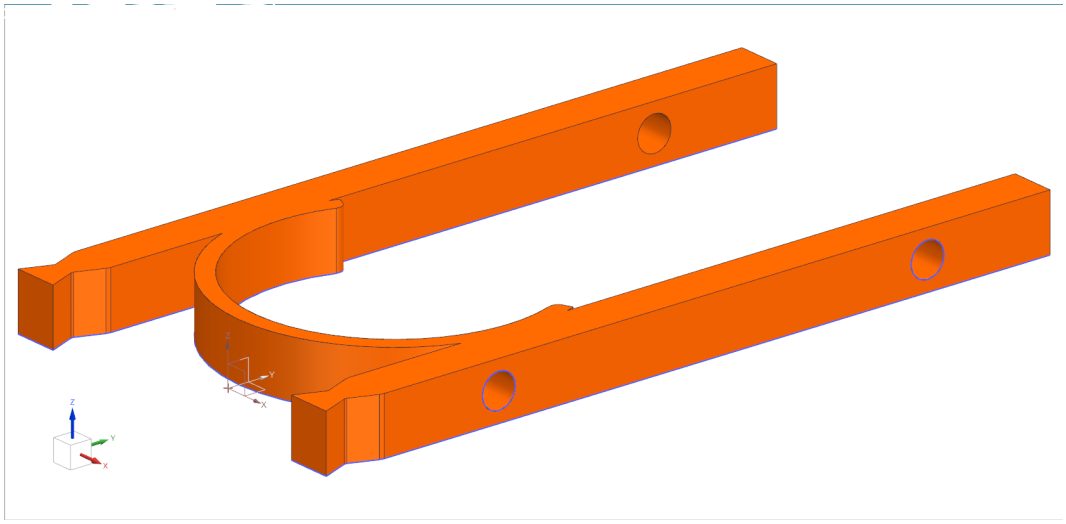


Σχήμα 4.9: Μηχανολογικά Σχέδια Πάνω Βάσεων

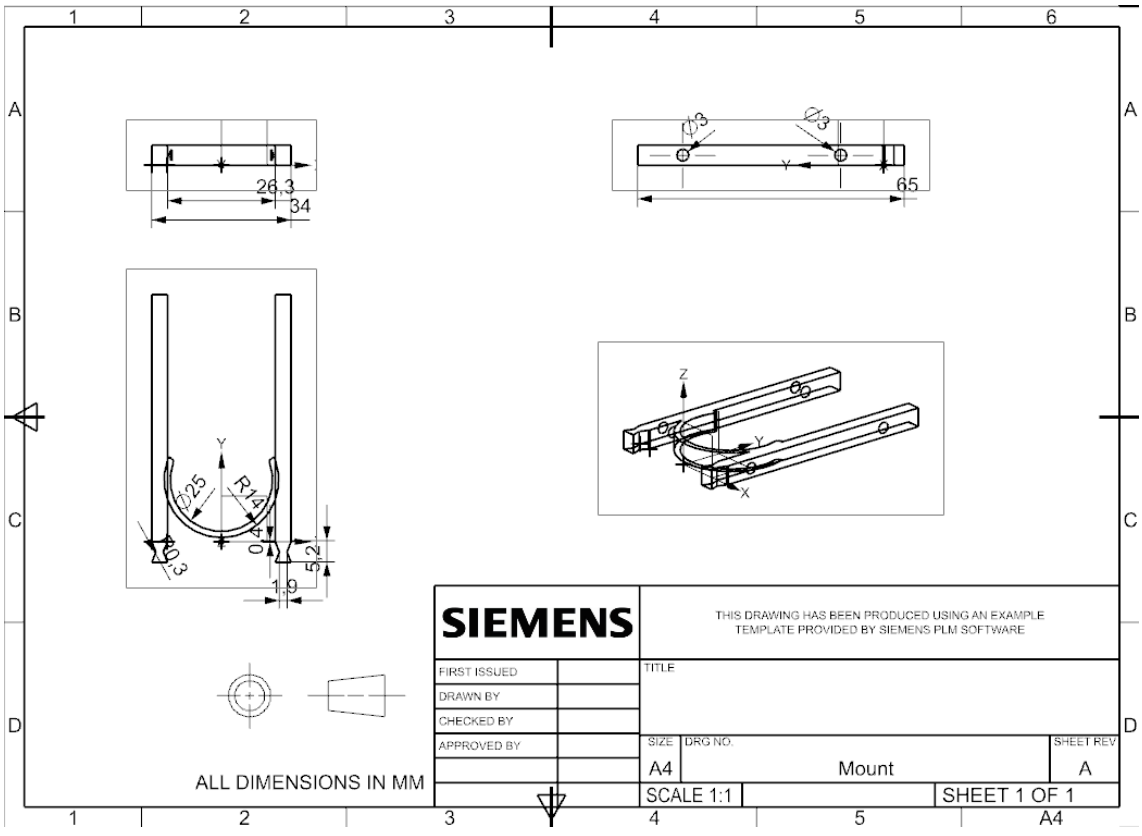
4.4.3 Στηρίγματα

Για την στήριξη των συσσωρευτών που θα τοποθετηθούν στο ρομπότ απόφασίστηκε ότι η καλύτερη προσέγγιση θα ήταν η τοποθέτηση τους στο κάτω μέρος του ρομπότ. Συνεπώς, να συγκρατηθούν πάνω στην κάτω βάση του. Ωστόσο η παράλληλη τοποθέτηση τους πάνω στην κάτω βάση θα απαιτούσε την αύξηση του μεγέθους της, πράγμα το οποίο δεν ήταν επιθυμητό. Για τον λόγο αυτό, η επόμενη επιλογή ήταν να τοποθετηθούν κάθετα πάνω της. Αρχική ιδέα ήταν η τοποθέτηση δοκαριών ως στηριγμάτα πρόσδεσης τους. Τα δοκάρια αυτά όμως, θα έπρεπε να είναι αρκετά μικρά σε πάχος και μακριά σε ύψος, γεγονός που πιθανότατα θα οδηγούσε σε ρίξη τους λόγω του μεγάλου βάρους των συσσωρευτών. Έτσι λοιπόν, προέκυψε το τελικό σχέδιο που φαίνεται στο Σχήμα 4.10. Η ενδιάμεση καμάρα στα δύο δοκάρια δίνει καλύτερη σταθερότητα και αντοχή στα δοκάρια και επιπλέον θα δώσει επιπλέον στήριξη στους κινητήρες οι οποίοι θα τοποθετηθούν ανάμεσα της. Το λεπτομερές μηχανολογικό σχέδιο της κατασκευής

παρουσιάζεται στο Σχήμα 4.11.



Σχήμα 4.10: Στηρίγματα

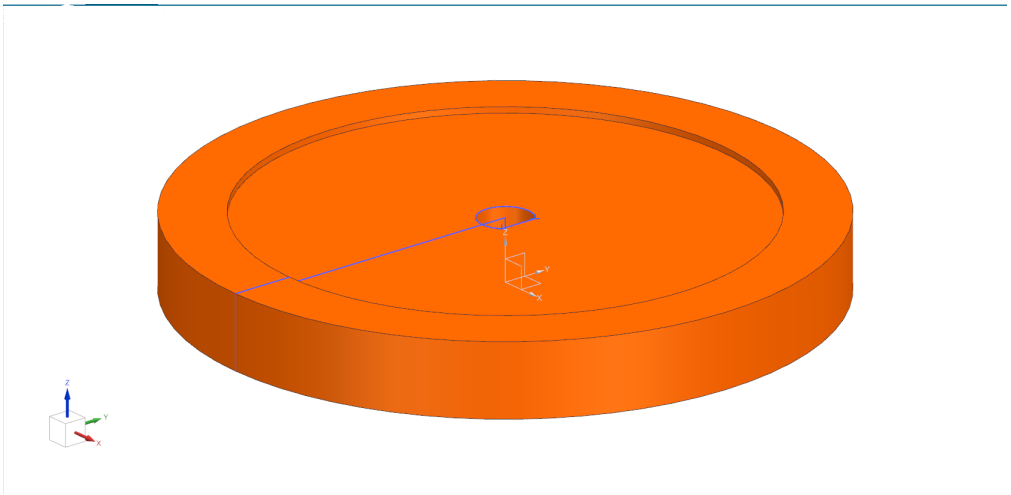


Σχήμα 4.11: Μηχανολογικά Σχέδια Στηριγμάτων

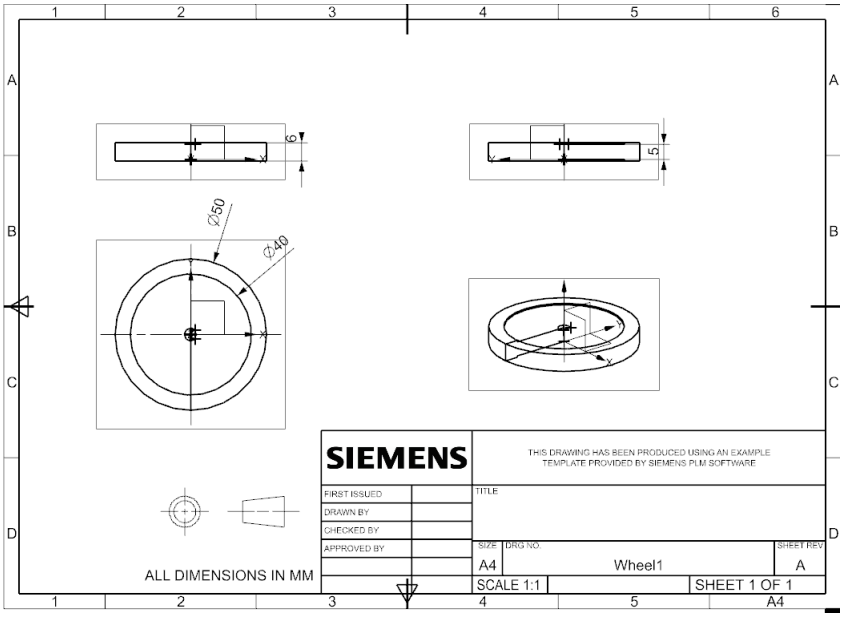
4.4.4 Τροχοί

Αρχική ιδέα για τους τροχούς της κατασκευής ήταν η αγορά έτοιμων του εμπορίου. Ωστόσο, καθ΄όλη τη διάρκεια της πορείας της εργασίας δεν βρέθηκαν τροχοί οι οποίοι

να είναι συμβατοί με τον άξονα του τύπου κινητήρα που επιλέχθηκε ή σε περίπτωση που ήταν συμβατοί δεν ταίριαζαν με τους περιορισμούς του σχεδιασμού του ρομποτικού οχήματος (όπως παραδείγματος χάρη, ότι απαιτείται η απόσταση των 5mm της βάσης του από το πάτωμα). Για τον λόγο αυτό, αποφασίστηκε η σχεδίαση απλών προσωρινών τροχών μέχρι την εύρεση κατάλληλων για την εφαρμογή (Σχήμα 4.12). Το λεπτομερές μηχανολογικό σχέδιο της κατασκευής παρουσιάζεται στο Σχήμα 4.13



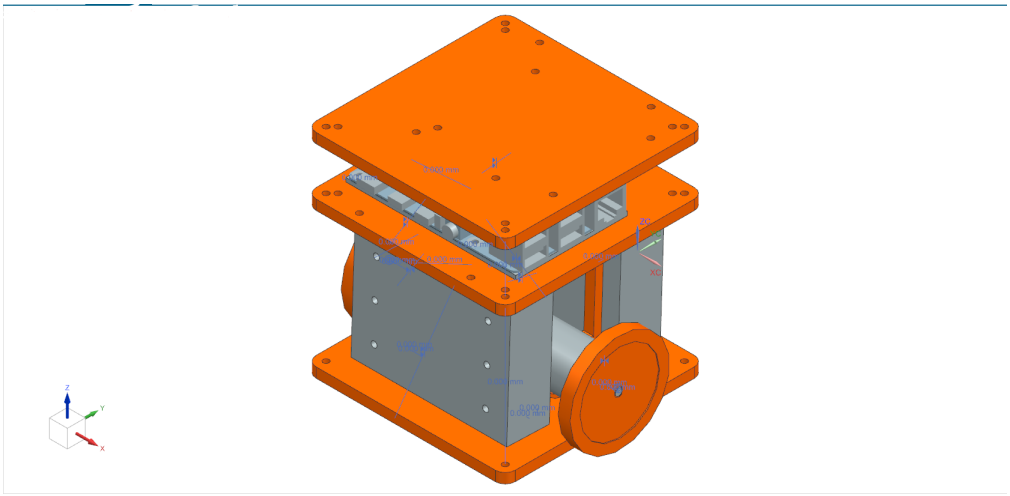
Σχήμα 4.12: Τροχοί



Σχήμα 4.13: Μηχανολογικά Σχέδια Τροχών

4.4.5 Συναρμολόγηση

Η τελική συναρμολόγηση του ρομποτικού οχήματος, στο σχεδιαστικό περιβάλλον, παρουσιάζεται στο Σχήμα 4.14.

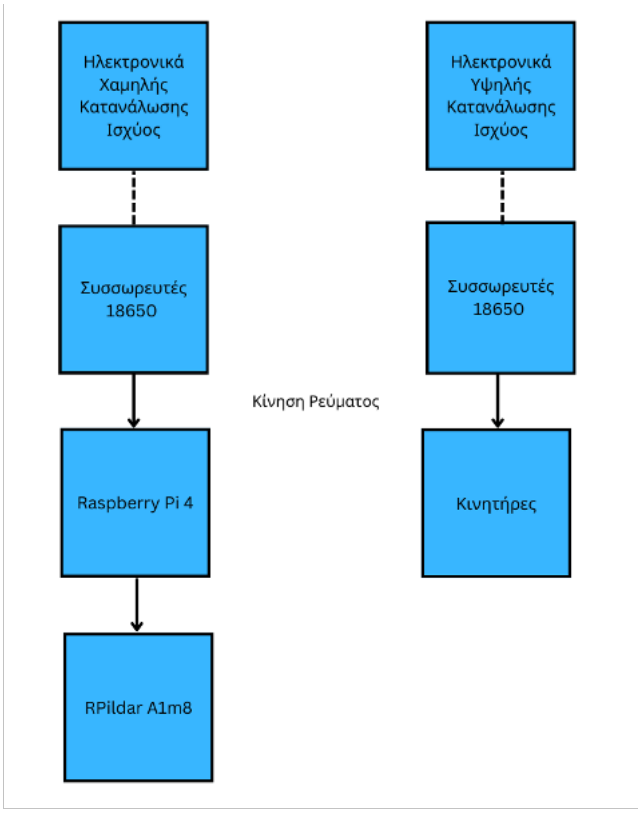


Σχήμα 4.14: Συναρμολόγηση

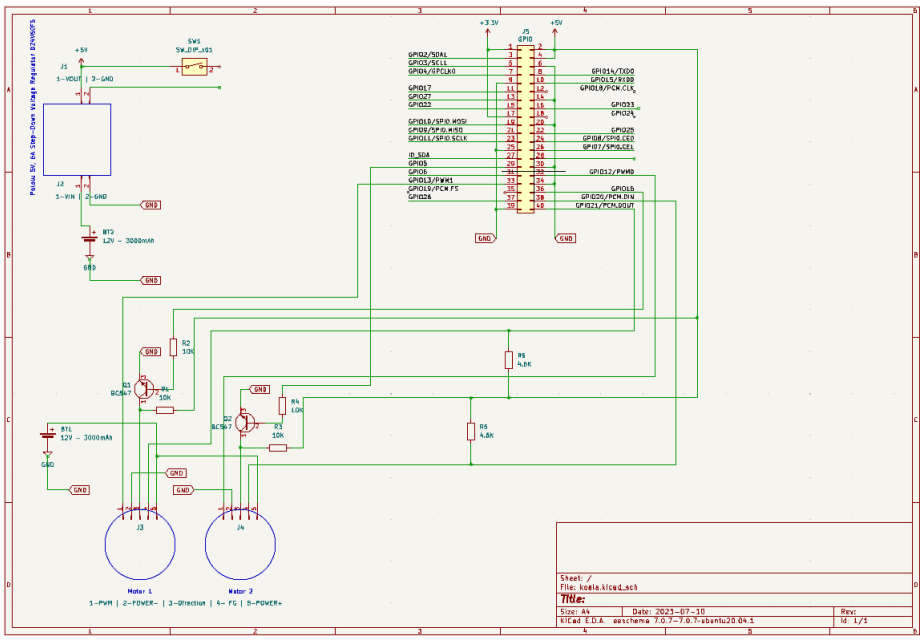
4.5 Σχηματικά και Τελική Συναρμολόγηση

4.5.1 Σχηματικά

Στο σχήμα 4.16 φαίνεται το ηλεκτρικό κύκλωμα που αναπτύχθηκε για την τροφοδοσία των συσκευών και των κινητήρων, όπως επίσης και για την επικοινωνία των κινητήρων με τον ρομποτικό ελεγκτή RPi 4. Όπως φαίνεται, για την σύνδεση των συσσωρευτών με τον RPi 4 έχει τοποθετηθεί ένα τροφοδοτικό (Pololu 5V, 6A Step-Down Voltage Regulator D24V60F5) το οποίο ρίχνει την τάση των συσσωρευτών απο τα 11.1V στα 5V που απαιτεί ο συγκεκριμένος ελεγκτής. Επίσης, έχει τοποθετηθεί ένας διακόπτης on/off στον θετικό πόλο. Για την σύνδεση των κινητήρων με τον ρομποτικό ελεγκτή συνδέθηκαν τα 3 pin σήματος (pins 1, 3, 4) του εκάστοτε κινητήρα (1-PWM, 2-POWER-, 3-Direction, 4-FG, 5-POWER+) με 3 κατάλληλα pin του RPi4 και τα υπόλοιπα 2 (pins 2, 5) χρησιμοποιήθηκαν για την τροφοδοσία του. Τα pin 1 συνδέθηκαν απευθείας με τα pin GPIO012, GPIO013. Τα pin 2 συνδέθηκαν με την γείωση. Τα pin 3 συνδέθηκαν με τα pin GPIO05, GPIO015 αφού πρώτα συνδεθούν τα pin των κινητήρων με το pin των 5V (καθώς τόσα απαιτούν τα pin αυτών) του RPi4. Επιπλέον, ήταν απαραίτητη η χρήση ενός transistor τύπου NPN για την σύνδεση των pin που πλέον έχουν διαφορετική τάση. Τα pin 4 συνδέθηκαν με τα pin GPIO020, GPIO021, αφού κι αυτά πρώτα συνδέθηκαν με το pin των 5V με χρήση ενός pull-up resistor, όπως ήταν ζητούμενο για το συγκεκριμένο pin. Τα pin 5 συνδέθηκαν με τον θετικό πόλο των συσσωρευτών.



Σχήμα 4.15: Απλό σχεδιάγραμμα κατασκευής

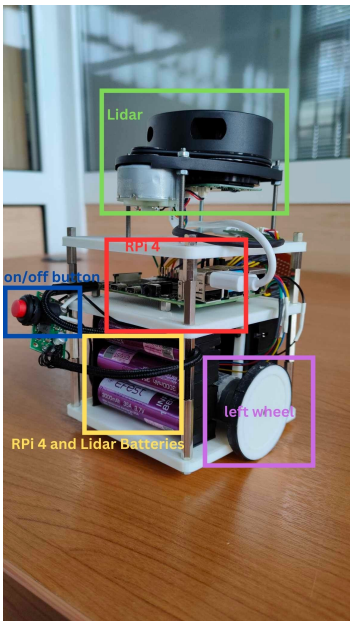


Σχήμα 4.16: Προσωρινό Σχηματικό Σύνδεσης Συσκευών

4.5.2 Τελική Συναρμολόγηση

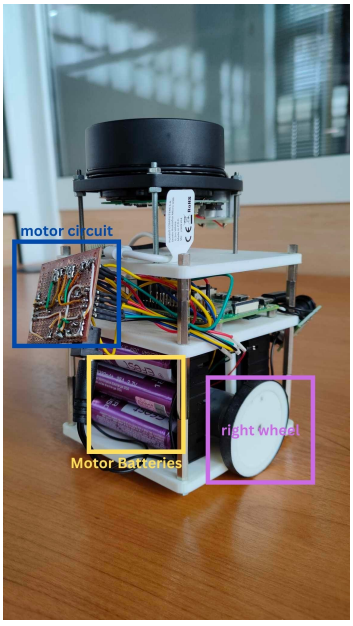
Η συνολική συναρμολόγηση του οχήματος παρουσιάζεται στο Σχήμα 4.17. Συγκεκριμένα διακρίνονται ο αισθητήρας LIDAR (πράσινο πλαίσιο), ο ρομποτικός ελεγκτής RPi 4 (κόκκινο πλαίσιο), ο διακόπτης ON/OFF (μπλε πλαίσιο) καθώς και οι συσσωρευτές

που τροφοδοτούν τον αισθητήρα και το τη μονάδα ελέγχου σε κίτρινο πλαίσιο. Σε μωβ πλαίσιο διακρίνεται ο αριστερός τροχός.



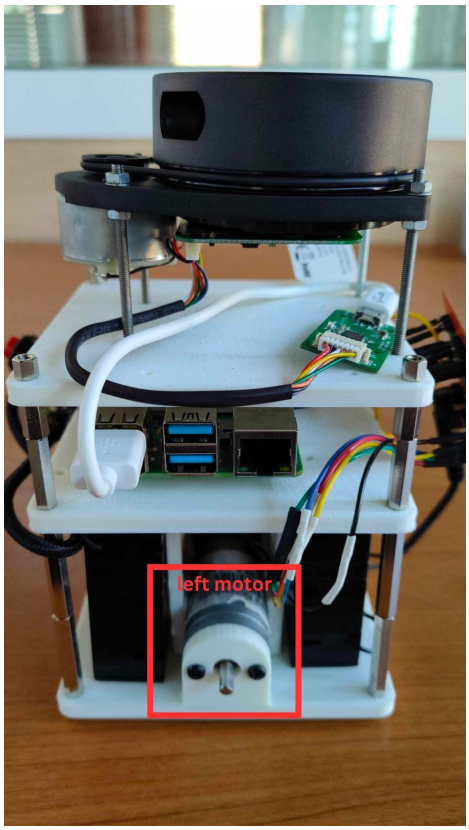
Σχήμα 4.17: Μπρος-Αριστερή όψη ρομποτικού οχήματος

Στο Σχήμα 4.18, διακρίνεται το ηλεκτρικό κύκλωμα που αναπτύχθηκε για την τροφοδοσία των συσκευών και των κινητήρων, η μπαταρίες που τροφοδοτούν τους κινητήρες, καθώς και ο δεξιά τροχός του οχήματος, ενώ στο Σχήμα 4.19, διακρίνεται ο αριστερός κινητήρας.

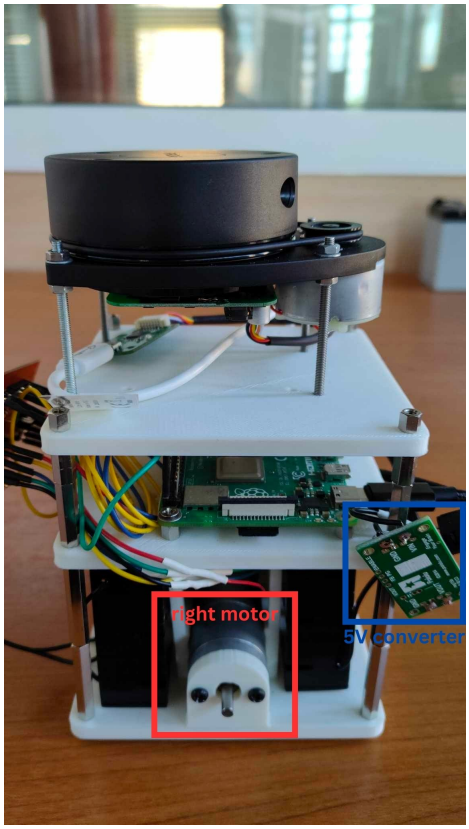


Σχήμα 4.18: Πίσω-Δεξιά όψη ρομποτικού οχήματος

Στο Σχήμα 4.20, διακρίνεται ο δεξιός κινητήρας καθώς και ο μετατροπέας τάσης.



Σχήμα 4.19: Αριστερή όψη ρομποτικού οχήματος



Σχήμα 4.20: Δεξιά όψη ρομποτικού οχήματος

Κεφάλαιο 5

Πειραματικά αποτελέσματα

Στο κεφάλαιο αυτό γίνεται περιγραφή του λογισμικού που αναπτύχθηκε ή χρησιμοποιήθηκε για την λειτουργία του οχήματος (συλλογή δεδομένων μέσω του αισθητήρα lidar και επικοινωνία με τους κινητήρες). Επιπλέον, παρουσιάζονται δεδομένα, εικόνες και γραφήματα που επιβεβαιώνουν την σωστή λειτουργία του με την χρήση του λογισμικού.

5.1 Λογισμικό

5.1.1 Επικοινωνία με τον Αισθητήρα Lidar

Όπως αναφερθηκε στο κεφάλαιο 4, για την επικοινωνία του ρομποτικού ελεγκτή Raspberry pi 4 χρησιμοποιήθηκε το πακέτο rplidar ros2. Το πακέτο αυτό είναι γραμμένο σε γλώσσα C++. Εγκαθιστάται στο περιβάλλον του ROS 2 κι επιτρέπει την συλλογή δεδομένων (απόστασης και γωνίας) τα οποία επιστρέφει στην οθόνη του τερματικού, καθώς επίσης και την οπτικοποίηση των δεδομένων στο Rviz 2 εάν υπάρχει γραφικό περιβάλλον εγκατεστημένο. Για την χρήση του πακέτου αυτού, αφού εγκατασταθεί, πρέπει να γίνουν τα παρακάτω βήματα:

Σε μία κονσόλα γράφονται οι παρακάτω δύο εντολές:

```
source ./install/setup.bash
ros2 run rplidar_ros rplidarNode
```

Σχήμα 5.1: Κώδικας 1ης κονσόλας

Ο κώδικας αυτός ξεκινάει τον κόμβο για την επικοινωνία του Lidar με τον Rpi 4.

Σε 2η κονσόλα γράφονται οι παρακάτω δύο εντολές:

```
source ./install/setup.bash
ros2 run tf2_ros static_transform_publisher 0 0 0 0 0 0  world laser_frame
```

Σχήμα 5.2: Κώδικας 2ης κονσόλας

Ο κώδικας αυτός μετατρέπει τα δεδομένα σε πραγματικό χρόνο έτσι ώστε να μπορούν να τυπωθούν στο περιβάλλον του Rviz 2.

Σε 3η κονσόλα γράφονται οι παρακάτω δύο εντολές:

```
source ./install/setup.bash
rviz2 ./install/rplidar_ros/share/rplidar_ros/rviz/rplidar.rviz
```

Σχήμα 5.3: Κώδικας 3ης κονσόλας

Ο κώδικας αυτός ανοίγει το περιβάλλον του Rviz 2 και οπτικοποιεί τα δεδομένα που συλλέγει ο αισθητήρας (δεν είναι απαραίτητος για να παρθούν τα δεδομένα στην οθόνη, δηλαδή ο κώδικας αυτός μπορεί να παραληφθεί).

Σε 4η κονσόλα γράφονται οι παρακάτω δύο εντολές:

```
source ./install/setup.bash
ros2 run rplidar_ros rplidarNodeClient
```

Σχήμα 5.4: Κώδικας 4ης κονσόλας

Ο κώδικας αυτός επιστρέφει στην 4η κονσόλα τα αποτελέσματα (απόσταση και γωνία) των σαρώσεων που πραγματοποιεί ο αισθητήρας. Για την επεξεργασία τον δεδομένων είναι απαραίτητη η αποθήκευση τους. Για τον λόγο αυτό αναπτύχθηκε ο κώδικας rplidar_subscriber. Αποτελεί κόμβο του ROS 2 και κάνει εγγραφή στο topic 'scan', που είναι το μήνυμα που στέλνει ο κόμβος του πακέτου παραπάνω. Ακολούθως, δημιουργεί σε κάθε νέα σάρωση αρχείο κειμένου και αποθηκεύει τις τιμές των σαρώσεων του αισθητήρα. Για την χρήση αυτού του κόμβου είναι προϋπόθεση να τρέχει αρχικά το πακέτο για την λειτουργία του αισθητήρα όπως περιγράφηκε παραπάνω. Κατόπιν της έναρξης λειτουργίας του πακέτου rplidar ros2, σε νέα κονσόλα γράφονται οι παρακάτω δύο εντολές:

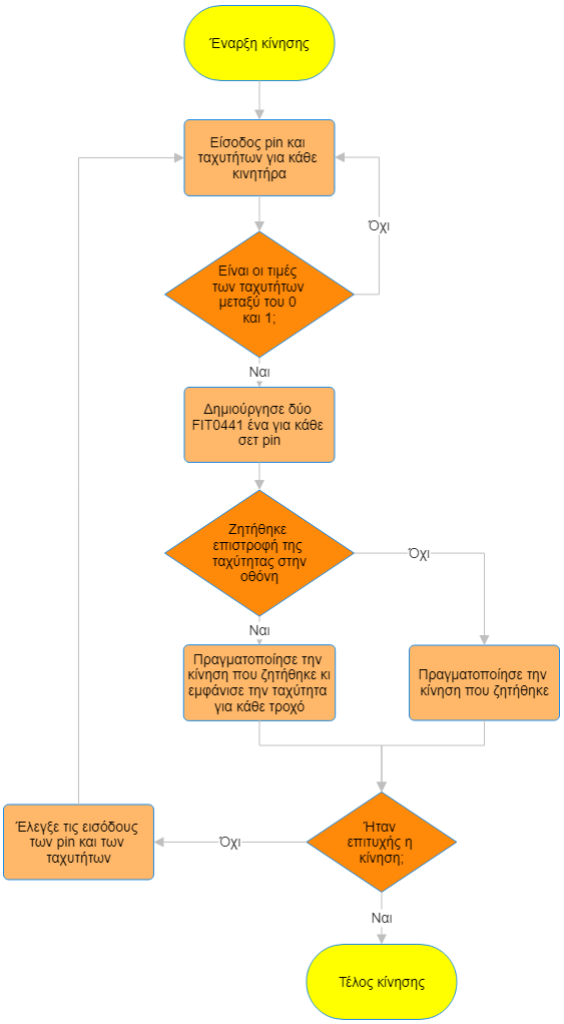
```
. install/setup.bash
ros2 run rplidar_subscriber subscriber
```

Σχήμα 5.5: Αποθήκευση δεδομένων σε αρχείο κειμένου

5.1.2 Επικοινωνία με τους Κινητήρες

Για την επικοινωνία του ρομποτικού ελεγκτή με τους κινητήρες κατασκευάστηκε ο κώδικας FIT0441 ο οποίος αποτελείται από δύο κλάσεις, την FIT0441NoFG που προειδοποιεί ότι δεν γίνεται επιστροφή του σήματος FG (που είναι το σήμα που επιστρέφει ο κινητήρας και είναι απαραίτητο για την οδομετρία) και την κύρια κλάση του κώδικα με όνομα FIT0441 η οποία υλοποιεί την λειτουργία ενός κινητήρα μοντέλου FIT0441. Η κλάση βασίζεται στις συναρτήσεις SmoothedInputDevice και PhaseEnableMotor της Python βιβλιοθήκης gpiozero. Διαχειρίζεται τα σήματα, που στέλνονται κι έρχονται από και προς τον ρομποτικό ελεγκτή με σκοπό την κίνηση του κινητήρα και την οδομετρία, υπολογίζει την ταχύτητα σε rps και δημιουργεί τις συναρτήσεις για τον έλεγχο των κινητήρων (όπως κίνηση με την φορά κίνησης του ρολογιού, αντίστροφη της κίνησης του ρολογιού, ταχύτητας και παύσης λειτουργίας).

Στη συνέχεια κατασκευάστηκε ο κώδικας Koala που περιέχει την ομώνυμη κλάση. Η κλάση αυτή έχει σκοπό τον έλεγχο των κινητήρων. Σε αυτήν ορίζονται τα pin στα οποία είναι συνδεδεμένα οι κινητήρες καθώς και οι επιθυμητές ταχύτητες για τον εκάστοτε κινητήρα. Έπειτα, αν οι τιμές των ταχυτήτων που δόθηκαν είναι σωστές αντιστοιχεί κάθε έναν απο τους κινητήρες σε έναν FIT0441 της προηγούμενης παραγράφου και παρέχει τις συναρτήσεις για την κίνηση του οχήματος (ταυτόχρονη κίνηση και των δύο κινητήρων) και την επιστροφή της ταχύτητας σε rps στην οθόνη. Η διαδικασία που ακολουθείται συνολικά για τη πραγματοποίηση της κίνησης περιγράφεται αναλυτικά στο διάγραμμα ροής που περιγράφεται στο Σχήμα 5.6.



Σχήμα 5.6: Διάγραμμα ροής που περιγράφει την διαδικασία για την πραγματοποίηση κίνησης του ρομποτικού οχήματος

Τέλος, αναπτύχθηκε ένας απλός κώδικας που αποτελεί παράδειγμα χρήσης των παραπάνω δύο με όνομα koala_test.py - simple example. Αν ΄τρέξει΄ ο κώδικας αυτός το όχημα θα κινηθεί για 5 δευτερολεπτα εμπρός, για 5 δευτερόλεπτα πίσω, για 5 δευτερόλεπτα θα εκτελέσει δεξιά στροφή και για 5 δευτερόλεπτα θα εκτελέσει αριστερή στροφή. Πριν την κάθε κίνηση εμφανίζει τον τύπο της κίνησης που εκτελεί και στο τέλος κάθε κίνησης εμφανίζει την ταχύτητα του εκάστοτε τροχού εκείνη την χρονική στιγμή.

5.2 Συλλογή Δεδομένων από τον Αισθητήρα RPlidar

Προκειμένου να εξακριβωθεί η ορθή λειτουργία του αισθητήρα Λιδαρ δοκιμάστηκαν αρκετά διαφορετικά σενάρια λειτουργίας. Στη συνέχεια παρουσιάζονται δεδομένα και 3 περιπτώσεις διατάξεων κατά τις οποίες συλλέχθηκαν δεδομένα απο τον αισθητήρα του ρομποτικού οχήματος. Οι αποτυπώσεις των δεδομένων συμβαδίζουν στον προσανατολισμό με τις εικόνες των διατάξεων.

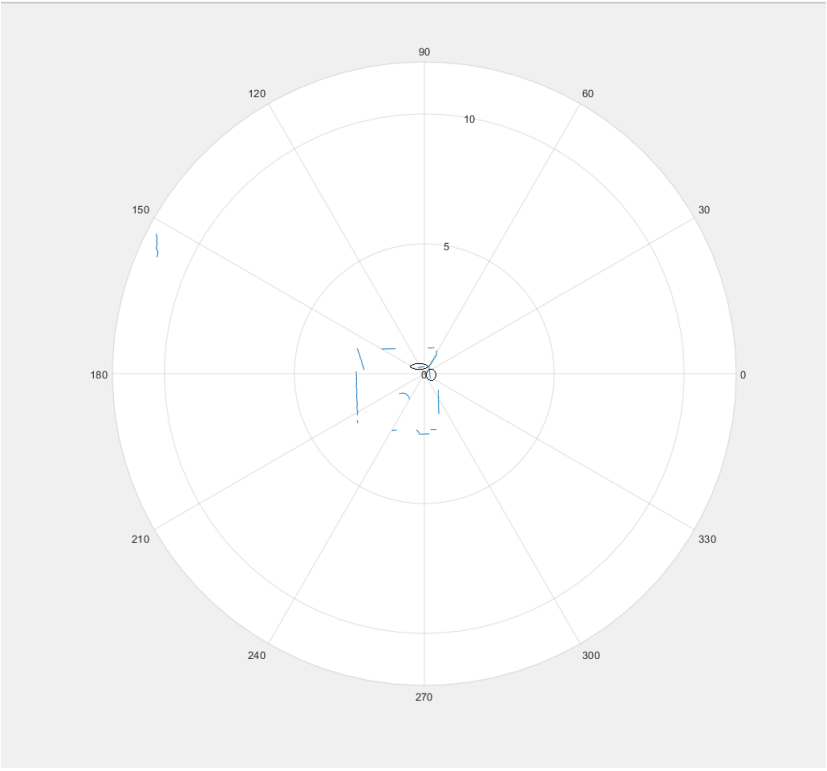
```
Angle: -179.81 degrees, Distance: 2.62 m
Angle: -179.44 degrees, Distance: 2.62 m
Angle: -179.07 degrees, Distance: 2.62 m
Angle: -178.71 degrees, Distance: 2.62 m
Angle: -178.34 degrees, Distance: 2.62 m
Angle: -177.97 degrees, Distance: 2.62 m
Angle: -177.61 degrees, Distance: 2.61 m
Angle: -177.24 degrees, Distance: 2.61 m
Angle: -176.87 degrees, Distance: 2.62 m
Angle: -176.51 degrees, Distance: 2.62 m
Angle: -176.14 degrees, Distance: 2.62 m
Angle: -175.78 degrees, Distance: 2.63 m
Angle: -175.41 degrees, Distance: 2.63 m
Angle: -175.04 degrees, Distance: 2.62 m
Angle: -174.68 degrees, Distance: 2.62 m
Angle: -174.31 degrees, Distance: 2.62 m
Angle: -173.94 degrees, Distance: 2.62 m
Angle: -173.58 degrees, Distance: 2.63 m
Angle: -173.21 degrees, Distance: 2.63 m
Angle: -172.84 degrees, Distance: 2.64 m
Angle: -172.48 degrees, Distance: 2.64 m
Angle: -172.11 degrees, Distance: 2.64 m
Angle: -171.74 degrees, Distance: 2.64 m
Angle: -171.38 degrees, Distance: 2.64 m
Angle: -171.01 degrees, Distance: 2.64 m
Angle: -170.64 degrees, Distance: 2.65 m
Angle: -170.28 degrees, Distance: 2.65 m
Angle: -169.91 degrees, Distance: 2.66 m
Angle: -169.54 degrees, Distance: 2.66 m
Angle: -169.18 degrees, Distance: 2.66 m
Angle: -168.81 degrees, Distance: 2.66 m
```

Σχήμα 5.7: Δεδομένα Σάρωσης

5.2.1 Περίπτωση 1η

Στη 1η περίπτωση το ρομποτικό όχημα έχει τοποθετηθεί ακίνητο και τροφοδοτείται από μια εξωτερική πηγή. Στο χώρο λειτουργίας του έχουν τοποθετηθεί 2 εμπόδια, όπως παρουσιάζονται στο Σχήμα 5.8. Τα δεδομένα, όπως οπτικοποιούνται παρουσιάζονται στο Σχήμα 5.9.

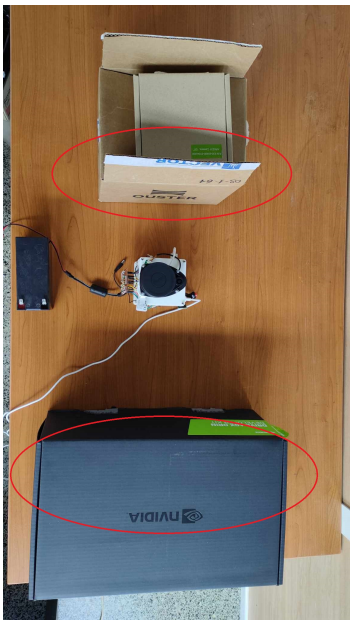
Σχήμα 5.8: Περίπτωση 1η



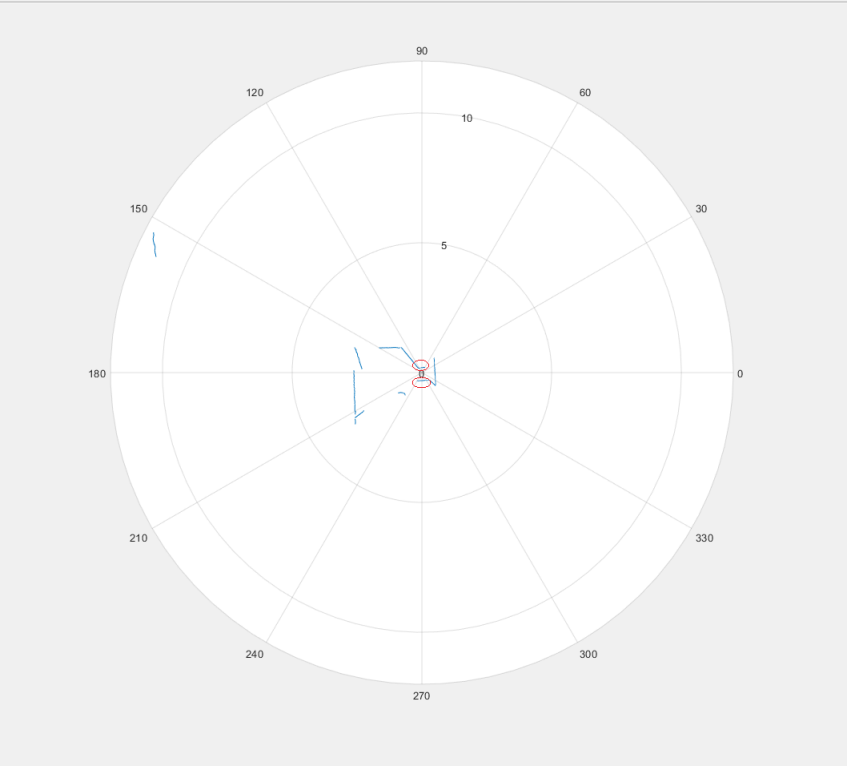
Σχήμα 5.9: Αποτύπωση δεδομένων 1ης περίπτωσης

5.2.2 Περίπτωση 2η

Στη 2η περίπτωση το ρομποτικό όχημα έχει τοποθετηθεί ακίνητο και τροφοδοτείται από μια εξωτερική πηγή. Στο χώρο λειτουργίας του έχουν τοποθετηθεί 2 εμπόδια, όπως παρουσιάζονται στο Σχήμα 5.10. Τα δεδομένα, όπως οπτικοποιούνται παρουσιάζονται στο Σχήμα 5.11.



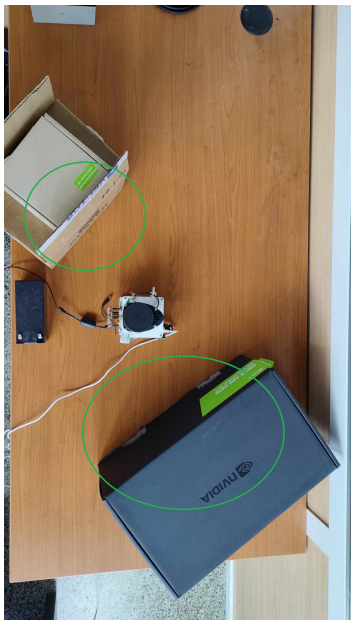
Σχήμα 5.10: Περίπτωση 2η



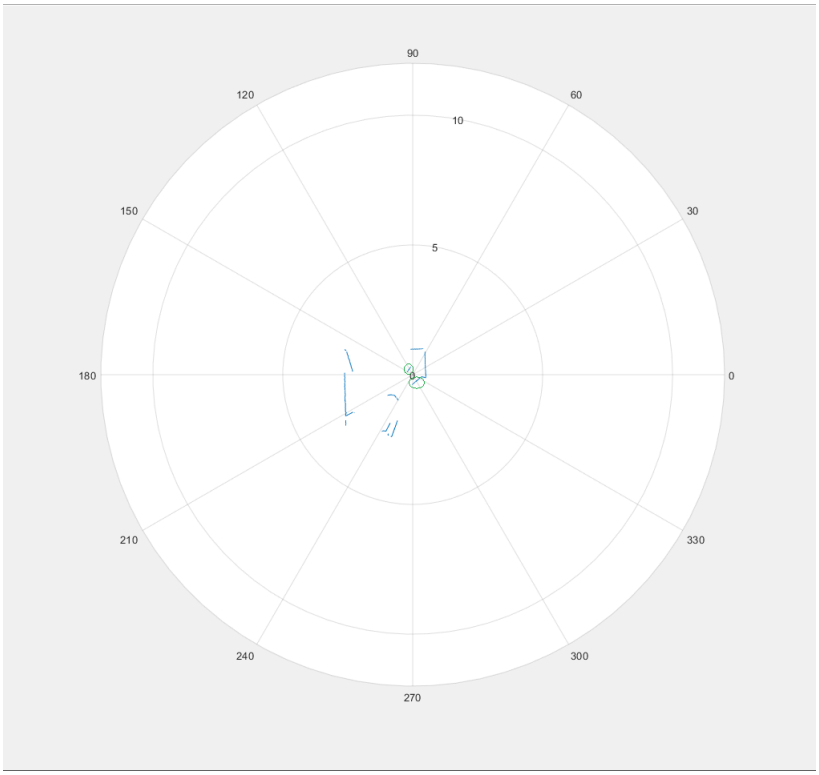
Σχήμα 5.11: Αποτύπωση δεδομένων 2ης περίπτωσης

5.2.3 Περίπτωση 3η

Στη 2η περίπτωση το ρομποτικό όχημα έχει τοποθετηθεί ακίνητο και τροφοδοτείται από μια εξωτερική πηγή. Στο χώρο λειτουργίας του έχουν τοποθετηθεί 2 εμπόδια, όπως παρουσιάζονται στο Σχήμα 5.12. Τα δεδομένα, όπως οπτικοποιούνται παρουσιάζονται στο Σχήμα 5.13.



Σχήμα 5.12: Περίπτωση 3η

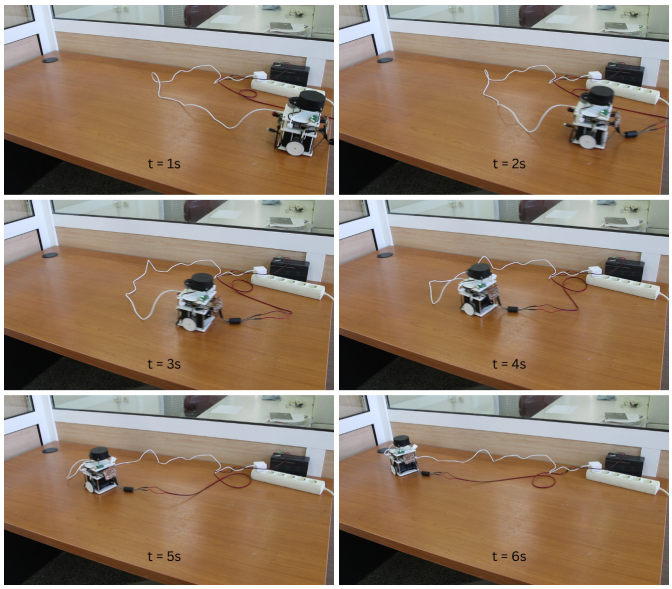


Σχήμα 5.13: Αποτύπωση δεδομένων 3ης περίπτωσης

5.3 Πειραματική ταυτοποίηση κίνησης

5.3.1 Εμπρός Κίνηση

Στην εικόνα 5.14 παρουσιάζονται 6 στιγμιότυπα από την δοκιμή κίνησης του ρομποτικού οχήματος σε προς τα εμπρός κίνηση. Στην περίπτωση αυτή, το όχημα τοποθετήθηκε (κοιτώντας προς τα αριστερά) στην δεξιά πλευρά του γραφείου και ζητούμενο ήταν η



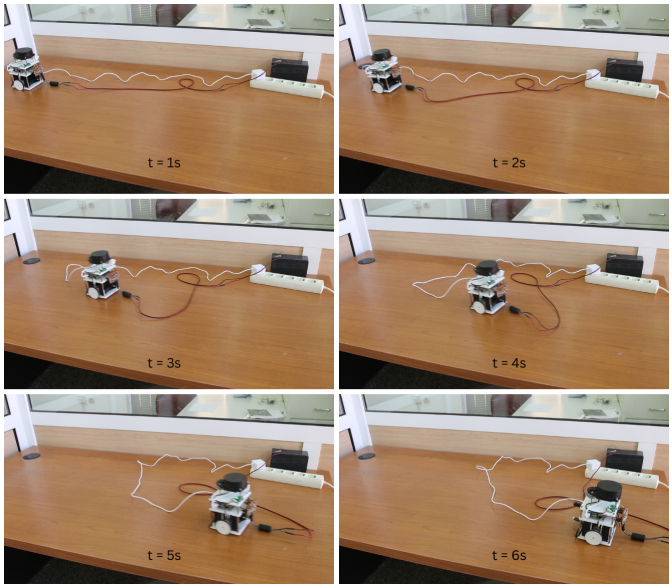
Σχήμα 5.14: Εμπρός Κίνηση

ευθεία του κίνηση μέχρι την απέναντι πλευρά αυτού.

Σημείωση: Η κλίση που φαίνεται να υπάρχει προς το τέλος της τροχιάς κίνησης του ρομποτικού οχήματος ευθύνεται στο τέντωμα του καλωδίου.

5.3.2 Πίσω Κίνηση

Στην εικόνα 5.15 παρουσιάζονται 6 στιγμιότυπα από την δοκιμή κίνησης του ρομποτικού οχήματος σε προς τα πίσω κίνηση.

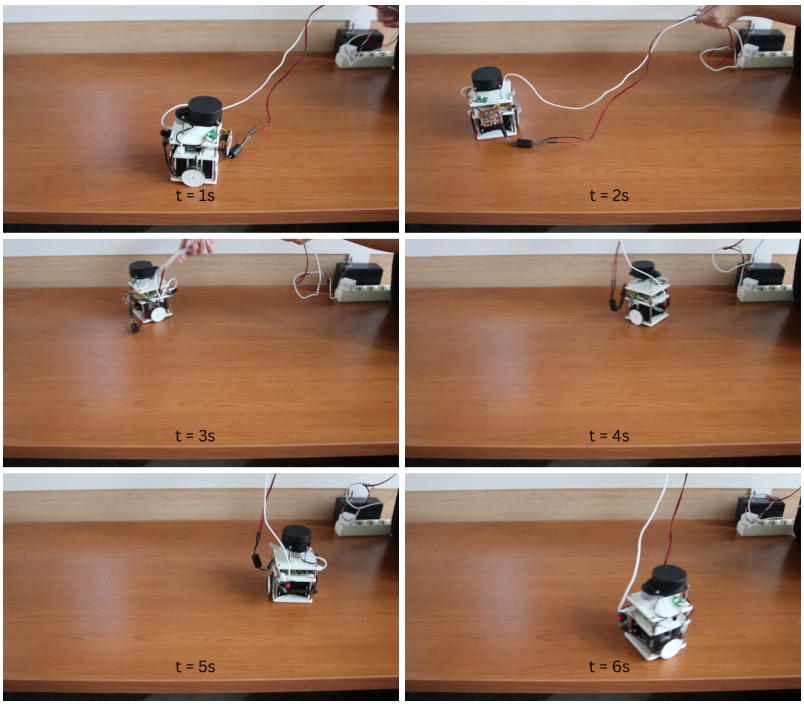


Σχήμα 5.15: Πίσω Κίνηση

Στην περίπτωση αυτή, το όχημα τοποθετήθηκε (κοιτώντας προς τα αριστερά) στην αριστερή πλευρά του γραφείου και ζητούμενο ήταν η ευθεία του κίνηση μέχρι την απέναντι πλευρά αυτού.

5.3.3 Δεξιά Στροφή

Στην εικόνα 5.16 παρουσιάζονται 6 στιγμιότυπα από την δοκιμή του ρομπότ για την πραγματοποίηση δεξιάς στροφής. Στην περίπτωση αυτή, το όχημα τοποθετήθηκε (κοιτώντας προς τα μπροστά) στο μπροστά μέρος του γραφείου και ζητούμενο ήταν η πραγματοποίηση μιας ολοκληρωμένης δεξιάς στροφής.



Σχήμα 5.16: Δεξιά Στροφή

5.3.4 Δεδομένα και Διαγράμματα Κινήσεων

Ακολουθούν δεδομένα και διαγράμματα με τις θέσεις που ακολούθησε το όχημα σε περίπτωση κίνησης σε ευθεία γραμμή και σε περίπτωση στροφής. Η ταχύτητα που επιστρέφεται, για κάθε κινητήρα, από τον κώδικα για τον έλεγχο τον κινητήρων έχει μονάδα μέτρησης το rps (round per second) και εμφανίζονται στην οθόνη όπως φαίνεται παρακάτω.

```
Right:  1.6670630607941488  Left:  1.651045270062811
Right:  1.7012958259250504  Left:  1.7657925679819049
Right:  1.6652312040280184  Left:  1.6201764838379542
Right:  1.6605589321663425  Left:  1.6595113190381898
Right:  1.6145263183454281  Left:  1.5944475726765435
Right:  1.7331097208850477  Left:  1.6605306411130158
Right:  1.7421669130803523  Left:  1.6074155344852852
Right:  1.7460303392311993  Left:  1.6350637963814867
Right:  1.613406633611275  Left:  1.6859495106612699
```

Σχήμα 5.17: Δεδομένα κατά την κίνηση σε ευθεία γραμμή

```
Right:  1.369314984460436  Left:  2.0977360313458004
Right:  1.3780331990466883  Left:  2.1075937483536458
Right:  1.3947746001471226  Left:  1.8351848513957074
Right:  1.322836468135923  Left:  2.1515325083495194
Right:  1.4183752483461072  Left:  1.7032486638478417
Right:  1.6971779869574795  Left:  2.1503757653726097
Right:  1.371870263476851  Left:  1.6766610411188854
Right:  1.4122260352348264  Left:  2.1154859779876176
```

Σχήμα 5.18: Δεδομένα κατά την εκτέλεση στροφής

Ωστόσο, τα δεδομένα αυτά δεν είναι χρήσιμα σε αυτή την μορφή. Για το λόγο αυτό, μετατράπηκαν αρχικά σε m/s πολλαπλασιάζοντας τις παραπάνω τιμές με 2πr όπου r=25mm κι έπειτα διαιρώντας με ακέραιες χρονικές στιγμές (t=1,2,3...) μετατράπηκαν σε διανυθείσα απόσταση του εκάστοτε τροχού. Οι παραπάνω υπολογισμοί πραγματοποιήθηκαν στο EXCEL και τα τελικά δεδομένα φαίνονται αμέσως μετά.

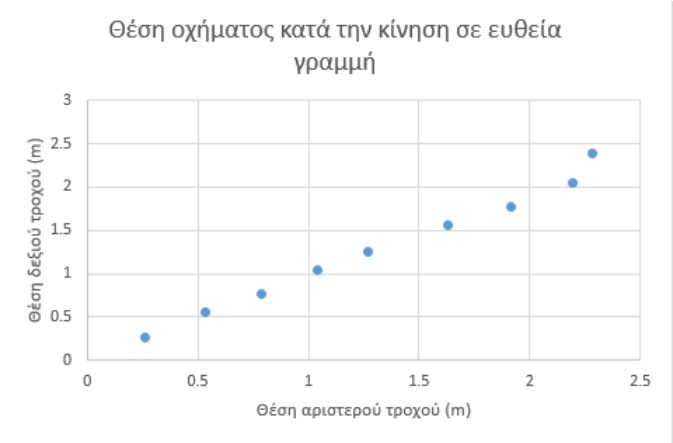
0.261862	0.259346
0.534478	0.55474
0.784722	0.76349
1.04336	1.042702
1.268046	1.252276
1.633417	1.565013
1.915613	1.767446
2.194126	2.054682
2.2809	2.383455

Σχήμα 5.19: Επεξεργασμένα δεδομένα κατά την κίνηση σε ευθεία γραμμή

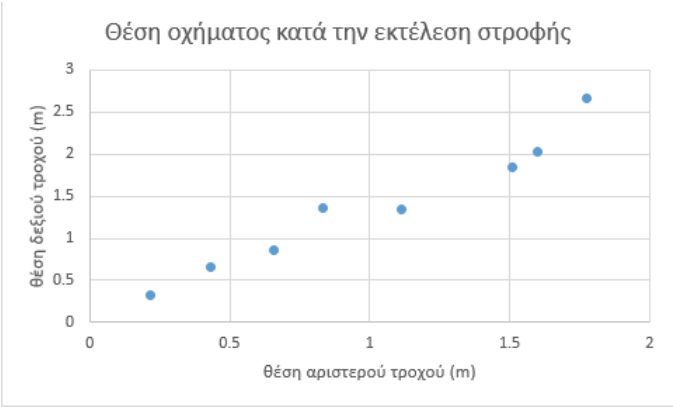
0.215091	0.329512
0.432922	0.66212
0.657272	0.86481
0.831163	1.351848
1.113989	1.337728
1.599553	2.026681
1.50845	1.843585
1.774656	2.658398

Σχήμα 5.20: Επεξεργασμένα δεδομένα κατά την εκτέλεση στροφής

Όπως φαίνεται στην εικόνα 5.19 οι τιμές των δύο τροχών είναι πολύ κοντά μεταξύ τους. Το γεγονός αυτό, επιβεβαιώνει την πραγματοποίηση ευθείας κίνησης. Αντιθέτως, στην εικόνα 5.20 οι τιμές του αριστερού τροχού είναι αισθητά μεγαλύτερες, γεγονός που οδηγεί στην πραγματοποίηση στροφής του οχήματος.



Σχήμα 5.21: Θέση οχήματος κατά την κίνηση σε ευθεία γραμμή



Σχήμα 5.22: Θέση οχήματος κατά την εκτέλεση στροφής

Κεφάλαιο 6

Παρατηρήσεις - Προτάσεις

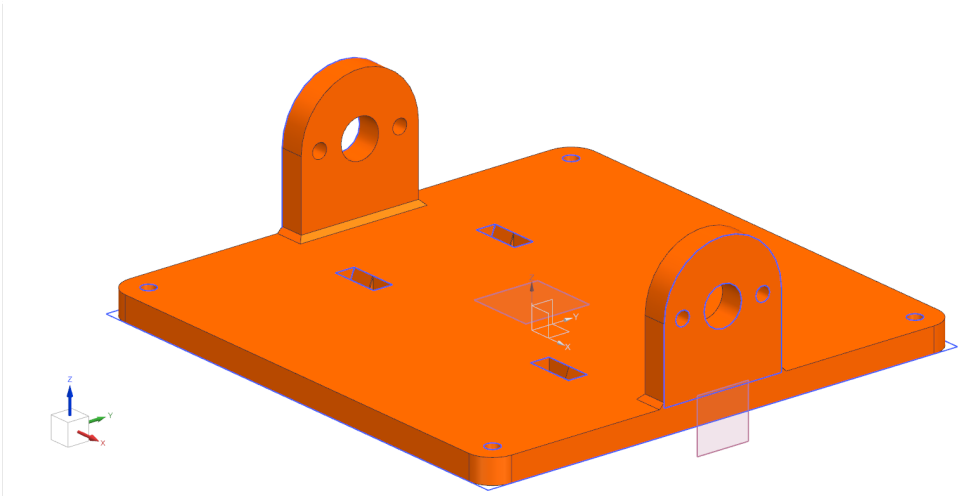
6.1 Ρομποτικός Ελεγκτής

Ο RPi4 αν και ικανότατος ρομποτικός ελεγκτής και ευρέως γνωστός για τις ανάλογες εφαρμογές του στην ρομποτική, όπως επίσης και σε άλλους τομείς δεν ήταν τελικά η ιδανικότερη λύση στην παρούσα εργασία. Το συμπέρασμα αυτό προήλθε από το γεγονός ότι ήταν από την αρχή ζητούμενο η εγκατάσταση στην υπό κατασκευή του λογισμικού ανοιχτού κώδικα ROS 2, το οποίο είναι δυνατό μόνο στην περίπτωση εγκατάστασης λογισμικού Ubuntu. Κατά την διάρκεια της ανάπτυξης της εργασίας με την εγκατάσταση των Ubuntu και ROS 2 προέκυπταν διάφορα, σε άλλες περιπτώσεις επιλύσιμα και σε άλλες απροσπέλαστα ζητήματα: Δεν μπορούσαν να επικοινωνήσουν ο RPi4 με τον αισθητήρα RPlidar, οι βιβλιοθήκες της κάμερας που ήταν να εγκατασταθεί στην συσκευή δεν ήταν συμβατες παρά μόνο με το λειτουργικό Raspberry Pi Os της ίδιας εταιρείας που παράγει τον RPi4 και την κάμερα όπως κι άλλα. Αυτά οδήγησαν σε πολλές εγκαταστάσεις και απεγκαταστάσεις λογισμικών και εκκίνησης απο το μηδέν μέχρι να βρεθεί ο συγκεκριμένος λειτουργικός συνδυασμός των λογισμικών και βιβλιοθηκών που αναφέρθηκαν παραπάνω.

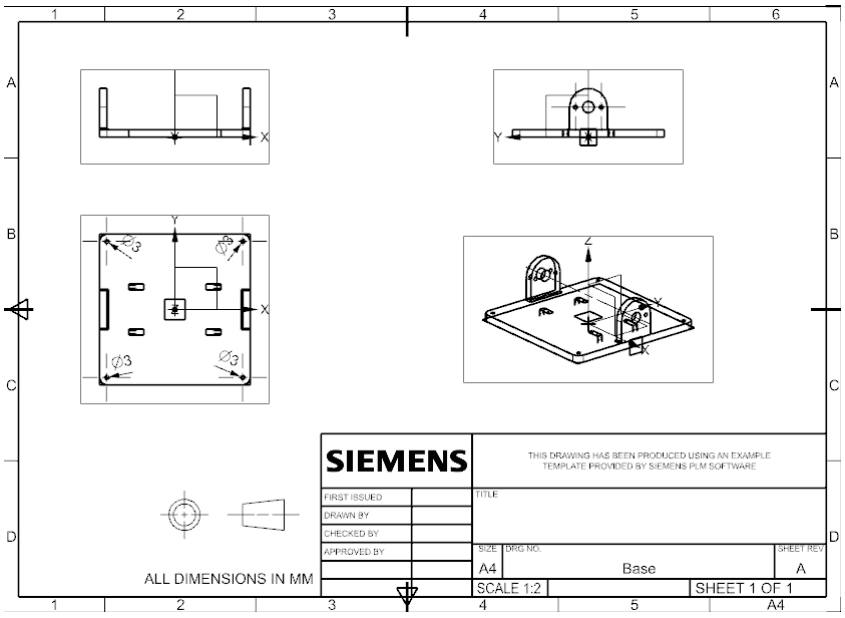
Αρχική επιλογή για την εργασία αυτή υπήρξε ο Nvidia Jetson Nano που είναι ικανότερος του RPi4 και δίνει την δυνατότητα για αλγορίθμους μηχανικής μάθησης. Ωστόσο, λόγω έλλειψης στην αγορά δεν μπόρεσε να γίνει διαθέσιμος. Σε περίπτωση επιλογής εξέλιξης του ρομποτικού οχήματος, ο Jetson Nano θα αποτελούσε μια σημαντική αναβάθμιση.

6.2 CAD

Στον σχεδιασμό της Κάτω Βάσης και των Πάνω Βάσεων θα ήταν καλό οι οπές που προορίζονται για την στήριξη τους (οι τέσσερις οπές στις γωνίες) να σχεδιαστούν και να τυπωθούν με διάμετρο 0.5mm λιγότερο έτσι ώστε οι αποστάτες να μπορούν να σφηνώνουν σε αυτές. Αυτό έτσι ώστε να αφαιρεθούν τα παξιμάδια που έχουν χρησιμοποιηθεί στο κάτω μέρος τους τα οποία εμποδίζουν την κίνηση του ρομποτικού οχήματος. Έπειτα η περίσσια των αποστατών στο κάτω μέρος της βάσης μπορεί να αφαιρεθεί με χρήση τροχού καθώς εμποδίζουν την κίνηση του οχήματος.

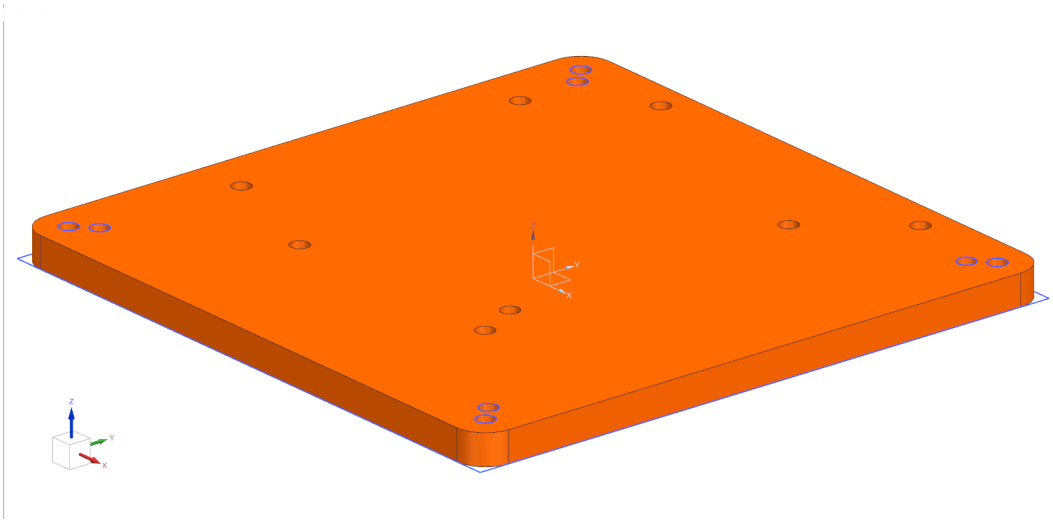


Σχήμα 6.1: Κάτω Βάση - Μικρότερες οπές στήριξης

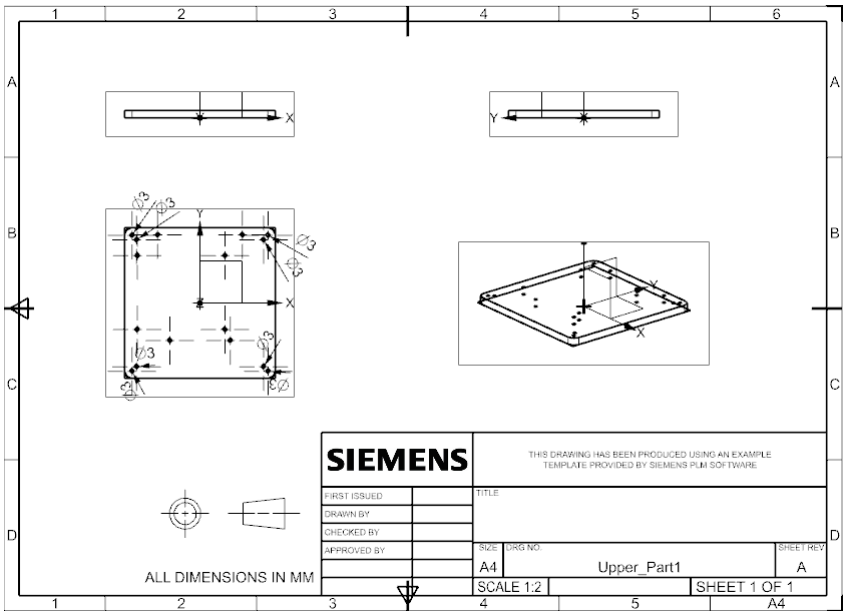


Σχήμα 6.2: Μηχανολογικά Σχέδια Κάτω Βάσης - Μικρότερες οπές στήριξης

Μια επίσης βελτιωτική παρατήρηση θα ήταν να δημιουργηθούν 4 επιπλέον οπές στις Πάνω Βάσεις οι οποίες να βρίσκονται σε γωνία 45°και σε μικρή απόσταση από τις ήδη υπάρχουσες οπές που προορίζονται για την στήριξη. Αυτό θα δώσει την δυνατότητα στον χρήστη αν θέλει να αφαιρέσει ένα από τα επίπεδα της κατασκευής, χωρίς να είναι απαραίτητο να αφαιρέσει όλα τα από πάνω επίπεδα για να φτάσει σε αυτό.



Σχήμα 6.3: Πάνω Βάσεις - Μικρότερες και νέες οπές στήριξης



Σχήμα 6.4: Μηχανολογικά Σχέδια Πάνω Βάσεων - Μικρότερες και νέες οπές στήριξης

Οι υπόλοιπες διαστάσεις έχουν παραμείνει ίδιες γιάυτο παραλήφθηκαν στα σχέδια.

6.3 Τροχοί

Το αρχικό πλάνο για τους τροχούς, όπως αναφέρθηκε, ήταν να αγοραστούν από το εμπόριο. Ωστόσο, δεν βρέθηκαν κατάλληλοι και αποφασίστηκε να σχεδιαστούν και να εκτυπωθούν. Επομένως, σε μελλοντικό χρόνο θα ήταν καλό να αναζητηθούν καθώς οι συγκεκριμένοι που σχεδιάστηκαν είναι απλοί στο σχεδιασμό τους και απαιτούν κάποιο τρίτο υλικό να εγκατασταθεί στο εξωτερικό τους, για να μην ολισθαίνουν στην επιφάνεια με την οποία έρχονται σε επαφή.

6.4 Σχηματικό Ηλεκτρονικών

Το σχηματικό για την σύνδεση των ηλεκτρονικών που παρουσιάστηκε παραπάνω στο κεφάλαιο 4 δεν είναι απολύτως σωστό με βάση τις αρχές της ρομποτικής.

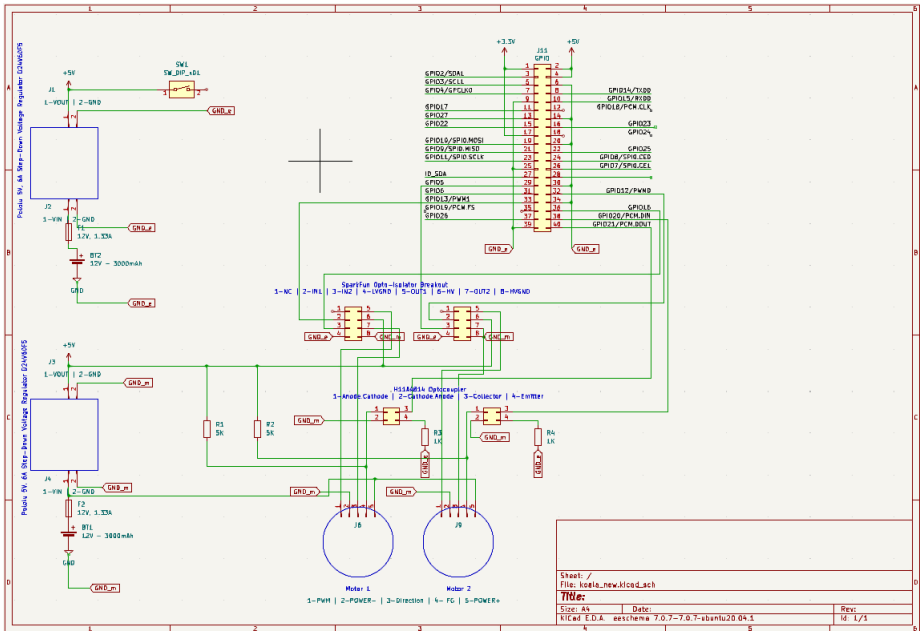
Αρχικά, συνδέονται οι γειώσεις των δύο συστοιχιών συσσωρευτών που χρησιμοποιήθηκαν, γεγονός που μπορεί να προκαλέσει προβλήματα στην λειτουργία του ρομποτικού ελεγκτή RPi4. Αυτο συμβαίνει διότι οι κινητήρες μπορεί να παράγουν θόρυβο κατά την λειτουργία τους. Έτσι, με την σύνδεση των γειώσεων μεταφέρεται αυτός ο θόρυβος στον ρομποτικό ελεγκτή και μπορεί να επηρεάσει την λειτουργία του αρνητικά.

Δεύτερον, δεν έχουν τοποθετηθεί ασφάλειες αμέσως μετά τους συσσωρευτές, πράγμα που θεωρητικά δεν επηρεάζει την λειτουργία του παραπάνω σχεδίου. Ωστόσο, σε περίπτωση κάποιας λάθος σύνδεσης ή υπέρτασης μπορεί χωρίς την χρήση ασφάλειας να προκληθεί ζημιά στα μέρη του ρομποτικού οχήματος. Έτσι, υπολογίστηκε ότι πρέπει να τοποθετηθεί μία ασφάλεια 12V, 1.33A στον θετικό πόλο των συσσωρευτών.

Επιπλέον, τα σήματα PWM, Direction, FG των κινητήρων ζητούν σύνδεση 5V για την σωστή λειτουργία τους, όπως έχει γίνει δηλαδή για τα σήματα Direction στο σχήμα του 4ου κεφαλαίου. Ωστόσο, ο RPi4 παρέχει τάση 3.3V στα pin του. Για την λύση του συγκεκριμένου προβλήματος πρέπει να χρησιμοποιηθούν δύο SparkFun Opto-Isolator Breakout. Η πλακέτα αυτή χρησιμοποιείται ακριβώς στην περίπτωση που είναι επιθυμητή σύνδεση συσκευών υψηλής τάσης με άλλα χαμηλότερης και κυρίως κινητήρων με ρομποτικούς ελεγκτές για την αποφυγή του προβλήματος που αναφέρθηκε παραπάνω. Αυτή η υλοποίηση όμως μπορεί να χρησιμοποιηθεί για τα σήματα PWM, Direction που στέλνονται από τον ρομποτικό ελεγκτή. Για το σήμα FG που η λογική είναι ανάστροφη (το σήμα δηλαδή πηγαίνει από τους κινητήρες προς τον ρομποτικό ελεγκτή) πρέπει να χρησιμοποιηθούν δύο H11AA814 (ή γενικά οποιοσδήποτε συμβατός optocoupler).

Οι παραδοχές κι η απλοποίηση του κυκλώματος που παρουσιάστηκε στο 4ο κεφάλαιο οφείλεται σε ελλείψης των συσκευών αυτών στο εργαστήριο. Κατασκευάστηκε, δοκιμάστηκε με επιτυχία και παρουσιάζεται ως προσωρινή λύση για την συγκεκριμένη διπλωματική εργασία.

Παρακάτω φαίνεται το θεωρητικά και πρακτικά σωστό και ολοκληρωμένο σχεδιάγραμμα του κυκλώματος.



Σχήμα 6.5: Προτεινόμενο Σχηματικό Σύνδεσης Συσκευών

6.5 Επικοινωνία Κόμβων ROS 2 σε Python και C++

Κατά την ανάπτυξη της εργασίας δοκιμάστηκε και βρέθηκε ότι είναι εφικτή και λειτουργική η επικοινωνία κόμβων σε Python με κόμβους σε C++. Μάλιστα, η βιβλιοθήκη που έχει εγκατασταθεί για την επικοινωνία του Lidar με τον ρομποτικό ελεγκτή είναι σε κώδικα C++ ενώ ο κόμβος που αναπτύχθηκε για την εγγραφή των δεδομένων από τις σαρώσεις είναι σε κώδικα Python.

6.6 Κινητήρες

Κατά την εκτέλεση των δοκιμών του ρομποτικού οχήματος έγινε αντιληπτό ότι ο δεξιός κινητήρας περιστρέφεται πιο αργά από τον αριστερό για την ίδια τιμή ταχύτητας. Αυτό, προφανώς είναι κατασκευαστικό σφάλμα του δεξιού κινητήρα καθώς είναι ίδιοι. Μέσω μικρού πειραματισμού βρέθηκε ότι ο δεξιός είναι περίπου 1/4 της στροφής αργότερος του αριστερού. Ωστόσο, επιπλέον πειραματισμός είναι απαραίτητος για την επιβεβαίωση αυτού.

6.7 Εξέλιξη του Ρομποτικού Οχήματος

Σε συνέχεια αυτής της διπλωματικής εργασίας, για την εξέλιξη του ρομποτικού οχήματος που αναπτύχθηκε, μπορούν να γίνουν οι παραπάνω βελτιώσεις που παρουσιάστηκαν. Προτεραιότητα θα πρέπει να αποτελέσει η δημιουργία κόμβου ROS 2 για τους κινητήρες της συσκευής. Επιπλέον, μπορούν να εγκατασταθούν περισσότεροι αισθητήρες, να δημιουργηθούν αλγόριθμοι χαρτογράφησης και αποφυγής εμποδίων. Ακόμα, μπορεί να μεταφερθεί η συσκευή για εικονοποίηση της λειτουργίας της ή προσομοίωση

στα περιβάλλοντα του RViZ και του Gazebo. Αυτό ωστόσο, θα πρέπει να συμβέι σε διαφορετική συσκευή από το RPi4 καθώς το λογισμικό Ubuntu που εγκαταστάθηκε δεν περιέχει γραφικό περιβάλλον. Επιπρόσθετα, μία σημαντική αλλαγή που πρέπει να γίνει, είναι στην επιλογή των ταχυτήτων στον κώδικα Koala.py θα πρέπει η ταχύτητα να ξεκινάει πάντα από χαμηλότερη και να ανέρχεται στην ζητούμενη τιμή. Δηλαδή, αν η επιθυμητή ταχύτητα είναι 0.5, το όχημα θα πρέπει να ξεκινάει παραδείγματος χάρη από το 0.1 και να ανεβαίνει με βήμα 0.1 μέχρι το 0.5. Αυτό, θα εμποδίσει το όχημα να ξεκινάει απότομα και επίσης προσφέρει ασφάλεια στους κινητήρες σε υψηλές τιμές ταχύτητας.

Παράρτημα Α΄

Κώδικες

A.1 FIT0441.py

```
from gpiozero import SmoothedInputDevice, GPIOZeroWarning,
    PhaseEnableMotor
from gpiozero.pins.pigpio import PiGPIOFactory
from threading import Event, Lock
import warnings

class FIT0441NoFG(GPIOZeroWarning):
    "Warning raised when the FIT0441 sees no FG at all"

class FIT0441(SmoothedInputDevice):
    """
    Class implementing single motor functionality
    """

    FG_LOCK = Lock() # Lock a part of the code for a single thread to
        avoid collisions

    def __init__(self, dir:(int | str)=None, pwm:(int | str)=None,
        fg:(int | str)=None,
                 speed:float=0.8, queue_len:int=9, partial:bool=True) ->
                    None:
        """Constructor of FIT0441

        Args:
            dir (int or str): The direction GPIO pin (output)
            pwm (int or str): The enable/speed GPIO pin (output)
            fg (int or str): Speed feedback GPIO pin Function Generator
                (input)
            speed (float): The initial speed of the motor (0.0 - 1.0)
                default value: 0.8
            queue_len (int): Number of values to calculate mean speed (rps)
                default value: 9
            partial (bool): Return partial approximation of speed
                default value: True
```

```
        Returns:
            None
        """

        # Call super to implement SmoothedInputDevice
        super(FIT0441, self).__init__(pin=fg, pull_up=False,
            queue_len=queue_len,
                                      partial=partial,
                                          ignore=frozenset({None}))

        # Define member variables for FIT0441
        try:
            self._motor = PhaseEnableMotor(phase=dir, enable=pwm,
                pwm=True) # instanciate PhaseEnableMotor
            self._motor.forward(1) # Stop motor from moving on setup
            self._speed = speed
            self._fg = Event() # Create a new Event for fg
            self._fg_rise = None # Initialize start of measurement to None
            self._fg_fall = None # Initialize end of measurement to None
            self.pin.edges = 'both' # Trigger event on both edges (rise
                and fall)
            self.pin.bounce = None
            self.pin.when_changed = self._fg_changed # Function to call
                when FG pin triggered
            self._queue.start()
        except:
            self.close()
            raise

    def close(self):
        super(FIT0441, self).close()

    @property
    def speed(self):
        return self._speed

    @speed.setter
    def speed(self, value):
        if value < 0 or value > 1:
            raise ValueError('must be between 0 and 1')
        self._speed = value

    @property
    def rps(self):
        """Returns the mean value of speed in rps

        Returns:
            float: rps
        """
        return self.value
```

```
    @property
    def value(self):
        return super(FIT0441, self).value

    @property
    def fg(self):
        return self.pin

    def _fg_changed(self, ticks, level):
        """ Trigger function for FG's change of state.
        """
        if level: # Start of high pulse (rise)
            self._fg_rise = ticks
        else:     # End of high pulse (fall)
            self._fg_fall = ticks
            self._fg.set() # Notify that an Event has occurred

    def _read(self):
        self._fg.clear()
        self._fg_fall = None
        self._fg_rise = None

        with FIT0441.FG_LOCK: # Lock this part of code for current thread
            if self._fg.wait(0.1): # Wait for Event to happen
                if self._fg_fall is not None and self._fg_rise is not
                    None: # Check that rise and fall timestamps exist
                    diff = self.pin_factory.ticks_diff(self._fg_fall,
                        self._fg_rise) # Calculate duration
                    return 0.0 if diff == 0 else 0.001851852/diff # Formula
                        to convert pulse width to rps
                else:
                    return None
            else:
                warnings.warn(FIT0441NoFG('no FG received'))
                return None

    # Clockwise motor rotation
    def cw(self, speed=None):
        if speed is not None:
            self._speed = speed
        self._motor.forward(1-self.speed) # FIT0441 is inverse logic
            compared to PhaseEnableMotor

    # Counter clockwise motor rotation
    def ccw(self, speed=None):
        if speed is not None:
            self._speed = speed
        self._motor.backward(1-self.speed) # FIT0441 is inverse logic
            compared to PhaseEnableMotor

    def stop(self):
```

```
        self._motor.forward(1)

    def set_speed(self, value):
        self.speed = value
```

A.2 Koala.py

```
from FIT0441 import FIT0441

# class that controls the motors
class Koala():

    # pins of RPi4 that the left motor is connected to
    DIR_L= 5
    PWM_L = 12
    FG_L = 20

    # pins of RPi4 that the right motor is connected to
    DIR_R = 16
    PWM_R = 13
    FG_R = 21

    # left and right velocity (values given by the user)
    V_Left_Wheel =
    V_Right_Wheel =

    def __init__(self, lspeed=V_Left_Wheel, rspeed=V_Right_Wheel):
   # both velocity values must be between 0 and 1 cause motors work on
      percentage(0.5 means 50% of the max velocity)
        if (not 0 <= lspeed <= 1) or (not 0 <= rspeed <= 1):
            raise ValueError('speed must be between 0 and 1')
        self._lspeed = lspeed
        self._rspeed = rspeed
        self._lmotor = FIT0441(dir=Koala.DIR_L, pwm=Koala.PWM_L,
            fg=Koala.FG_L, speed=self._lspeed)
        self._rmotor = FIT0441(dir=Koala.DIR_R, pwm=Koala.PWM_R,
            fg=Koala.FG_R, speed=self._rspeed)

    # function for forward movement
    def forward(self):
        self._lmotor.ccw()
        self._rmotor.cw()

    # function for backward movement
    def backward(self):
        self._lmotor.cw()
        self._rmotor.ccw()

    # function for right turn
    def right_turn(self):
        # right turn around robot's axis
        if (self._lspeed == self._rspeed):
            self._lmotor.ccw()
            self._rmotor.ccw()
   # every other possible right turn
```

```
        elif (self._lspeed > self._rspeed):
            self._lmotor.ccw()
            self._rmotor.cw()
        else:
            print("INPUT ERROR: Right wheel speed greater than Left wheel
                speed")

    # function for left turn
    def left_turn(self):
        # left turn around robot's axis
        if (self._lspeed == self._rspeed):
            self._lmotor.cw()
            self._rmotor.cw()
        # every other possible left turn
        elif (self._rspeed > self._lspeed):
            self._lmotor.cw()
            self._rmotor.ccw()
        else:
            print("INPUT ERROR: Left wheel speed greater than Right wheel
                speed")

    # function for stoping the motors
    def stop(self):
        self._lmotor.stop()
        self._rmotor.stop()

    # function that returns the speed of the left motor in rps
    def left_speed(self):
        return self._lmotor.rps

    # function that returns the speed of the right motor in rps
    def right_speed(self):
        return self._rmotor.rps
```

A.3 koala_test.py - simple example

```
from Koala import Koala
from time import sleep
from signal import pause


koala = Koala()

# forward movement for 5 seconds while printing the speed of each motor
koala.forward()
print('Forward Movement')
sleep(5)
print('Right: ', koala.right_speed(), ' Left: ', koala.left_speed())

# backward movement for 5 seconds while printing the speed of each motor
koala.backward()
print('Backward Movement')
sleep(5)
print('Right: ', koala.right_speed(), ' Left: ', koala.left_speed())

# right turn for 5 seconds while printing the speed of each motor
koala.right_turn()
print('Right Turn')
sleep(5)
print('Right: ', koala.right_speed(), ' Left: ', koala.left_speed())

# left turn for 5 seconds while printing the speed of each motor
koala.left_turn()
print('Left Turn')
sleep(5)
print('Right: ', koala.right_speed(), ' Left: ', koala.left_speed())
```

A.4 rplidar_subscriber

```
# Node that writes to a text file the data collected from the RPlidar
    node from the installed library

# The code is based on the "Write a simple publisher and subscriber
    (Python)" from the Ros 2 Documentation: Galactic tutorials
# and the RPlidar python library

import rclpy
from rclpy.node import Node
from sensor_msgs.msg import LaserScan
import math

class ScanSubscriberNode(Node):
    def __init__(self):
        super().__init__('scan_subscriber')
        self.subscription = self.create_subscription(
            LaserScan,
            '/scan',
            self.scan_callback,
            10
        )
        self.subscription

    def scan_callback(self, msg):
        with open('scan_data.txt', 'w') as file:
            for i, r in enumerate(msg.ranges):
                angle = math.degrees(msg.angle_min + (i *
                    msg.angle_increment))
                file.write(f'Angle: {angle:.2f} degrees, Distance: {r:.2f}
                    m\n')



def main(args=None):
    rclpy.init(args=args)
    node = ScanSubscriberNode()
    rclpy.spin(node)
    node.destroy_node()
    rclpy.shutdown()


if __name__ == '__main__':
    main()

#to run type in terminal: . install/setup.bash
#and then: ros2 run rplidar_subscriber subscriber
#after you have already ran the rplidar node from the installed library
    so that the scan has started
```

Bibliography

[1] iRobot Corporation. *Create® 3 DataSheet*. Sept. 2023. URL: https://experience.irobot.com/hubfs/Create%203/Create-3_DataSheet.pdf?utm_campaign=Product%20Launch&utm_source=Create%203&utm_medium=Landing%20page.

[2] Clearpath Robotics. *Turtlebot 4*. Sept. 2023. URL: https://clearpathrobotics.com/turtlebot-4/.

[3] Επιστημονικές Επιχειρήσεις ΕΠΕ. *Thymio*. Sept. 2023. URL: http://www.thymio.gr/.

[4] Πιπερίδης Σάββας και Δοϊτσίδης Ελευθέριος. *Σημειώσεις εργαστηρίου ρομποτικής*. Sept. 2023.

[5] Raspberry Pi Foundation. *Raspberry Pi 4 and Rasberry Pi HQ Camera*. Sept. 2023. URL: https://www.raspberrypi.org/.

[6] Nvidia. *Nvidia Jetson Nano*. Sept. 2023. URL: https://developer.nvidia.com/embedded/jetson-nano-developer-kit.

[7] Beagleboard.org Foundation. *Beaglebone AI*. Sept. 2023. URL: https://www.beagleboard.org/boards/beaglebone-ai.

[8] Velodyne Lidar. *What is lidar?* Sept. 2023. URL: https://velodynelidar.com/what-is-lidar/.

[9] Slamtec Global Network. *Rplidar A1*. Sept. 2023. URL: https://www.slamtec.com/en/Lidar/A1/.

[10] Velodyne Lidar. *Image Sensor*. Sept. 2014. URL: https://www.techopedia.com/definition/30413/image-sensor.

[11] Vidya Prabhu. *DIFFERENT TYPES OF MOTORS USED BY MAKERS AND HOBBYISTS*. Mar. 2020. URL: https://www.youngwonks.com/blog/Different-types-of-motors-used-by-makers-and-hobbyists.

[12] Google Images. *Google Images*. Sept. 2023. URL: https://www.google.com/imghp?hl=en&ogbl.

[13] Gregory Dudek and Michael Jenkin. *Computational Principles of Mobile Robotics*. 2nd ed. Cambridge University Press, 2010. DOI: 10.1017/CBO9780511780929.

[14] ROS 2. *Ros 2 Documentation: Galactic*. Sept. 2023. URL: https://docs.ros.org/en/galactic/index.html.

[15] DFRobot. *FIT0441 Brushless DC Motor*. Sept. 2023. URL: https://wiki.dfrobot.com/FIT0441_Brushless_DC_Motor_with_Encoder_12V_159RPM#target_0.