Ατομική Διπλωματική Εργασία

ΧΑΡΤΟΓΡΑΦΗΣΗ ΠΕΡΙΟΧΗΣ ΜΕ ΧΡΗΣΗ ΡΟΜΠΟΤΙΚΟΥ ΕΞΟΠΛΙΣΜΟΥ ΚΑΙ ROS

Γεωργία Κούτη

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάιος 2019

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Χαρτογράφηση περιοχής με χρήση ρομποτικού εξοπλισμού και ROS

Γεωργία Κούτη

Επιβλέπων Καθηγητής
Ανδρέας Πιτσιλλίδης

Η Ατομική Διπλωματική Εργασία υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων απόκτησης του πτυχίου Πληροφορικής του Τμήματος Πληροφορικής του Πανεπιστημίου Κύπρου

Μάιος 2019

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον υπεύθυνο καθηγητή μου Δρ. Αντρέα Πιτσιλλίδη, ο οποίος αποτέλεσε στήριγμα για εμένα σε όλη την πορεία εκπόνησης της διπλωματικής μου, στηρίζοντας τις αποφάσεις μου και δίνοντας μου με αυτόν τον τρόπο την ευκαιρία να αποκομίσω σπουδαία μαθήματα ζωής και πολύτιμες γνώσεις.

Τις πιο ιδιαίτερες μου ευχαριστίες θα ήθελα επίσης να προσδώσω στο σπουδαίο μου φίλο και συμπαραστάτη στην όλη μου πορεία, Χριστόφορο Καραμολέγκο.

Ένα μεγάλο ευχαριστώ, στην οικογένειά μου.

Και ένα τελευταίο ευχαριστώ, το οποίο εμπερικλείει όλους εκείνους οι οποίοι έβαλαν έστω και ένα μικρό λιθαράκι στην αποπεράτωση της διπλωματικής μου εργασίας.

Ποιος δεν θα ζήλευε τη μαγεία ενός μα παρότι δαιδαλώδους˙ όμορφου συνάμα ταξιδιού!

Ευχαριστώ πολύ!

Περίληψη

Με τη ραγδαία εξέλιξη της ρομποτικής, παρατηρείται η ολοένα και περεταίρω χρήση της με σκοπούς τη βελτίωση της ζωής μας, τη διευκόλυνση στη διεκπεραίωσης καθημερινών διαδικασιών ή και σε περιπτώσεις υψηλού κινδύνου για τον άνθρωπο. Για την πλοήγηση ρομπότ σε κάποιο χώρο είναι πολύ χρήσιμη η κατοχή χάρτη του χώρου αυτού. Οπόταν, η κατασκευή χάρτη κτιρίων ή εξωτερικών χώρων μπορεί να αποδειχτεί χρήσιμη σε πάρα πολλές περιπτώσεις, όπως για σκοπούς μεταφοράς αντικειμένων από ρομπότ, μετάδοσης πληροφοριών ή σε περιπτώσεις υψηλού κινδύνου. Στη παρούσα διπλωματική εργασία εξετάζεται η χαρτογράφηση μιας περιοχή με τη χρήση φτηνού ρομποτικού εξοπλισμού, ο οποίος είναι: 2 ρομποτικά οχήματα εδάφους, raspberry pi, arduino, motor shield, αισθητήρες υπερήχων. Αναλυτικότερα, σκοπός είναι το ρομπότ να εξερευνήσει μία περιοχή μέσω εντολών που λαμβάνει απομακρυσμένα από κάποιο χρήστη και καθώς, διασχίζει την περιοχή να εξετάζει το περιβάλλον γύρω του, να μαθαίνει πληροφορίες για αυτό και να δημιουργεί δυσδιάστατο χάρτη με τις περιοχές όπου μπορεί να διακινηθεί ελεύθερα και τις περιοχές όπου υπάρχουν εμπόδια. Η επίλυση του προβλήματος αυτού γίνεται μέσω της παροχής εντολών κίνησης από το raspberry pi στο arduino. Το arduino μέσω του motor shield δίνει εντολές στους κινητήρες του ρομποτικού οχήματος για εκτέλεση της κίνησης. Μέσω των αισθητήρων υπερήχων ενημερώνεται το περιβάλλον με νέα γνώση για την ύπαρξη εμποδίων – τοίχων και της τοποθεσίας τους. Τέλος, το arduino επικοινωνεί τις πληροφορίες για το περιβάλλον στο raspberry pi, το οποίο κατασκευάζει το δυσδιάστατο χάρτη της περιοχής. Η ανταλλαγή μηνυμάτων μεταξύ raspberry pi και arduino γίνεται μέσω του ROS (Robotic Operating System), ένα λογισμικό που χρησιμοποιείται ευρέως για την ανάπτυξη ρομποτικών εφαρμογών.

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

1.1 Ορισμός προβληματισμού

Στη σημερινή εποχή, η χρήση ρομπότ για τη διευκόλυνση της ζωή μας γίνεται όλο και πιο έντονη. Τα ρομπότ εξελίσσονται ραγδαία και καταλαμβάνουν μέρος σε κάθε τομέα της ζωής μας. Ένα είδος ρομπότ που έχει εξελιχθεί σε μεγάλο βαθμό είναι τα ρομποτικά οχήματα εδάφους, όπως για παράδειγμα τα οχήματα αυτόματης οδήγησης (Automated Guided Vehicle) τα οποία είναι παρόν από την δεκαετία του 1950. Τα οχήματα αυτόματης οδήγησης είναι μηχανικά οχήματα χωρίς οδηγό, τα οποία συνήθως ελέγχονται με ηλεκτροκινητήρες και μπαταρίες και συνδέονται κυρίως με κέντρα διανομής, γραμμές παραγωγής και ορυχεία.[65]

Ένα πρόσφατο γεγονός που υπογραμμίζει την αναγκαιότητα της χρήσης ρομποτικών οχημάτων εδάφους, είναι το συμβάν στις 15 Απριλίου 2019, όταν ο ναός της Παναγίας των Παρισίων τυλίχτηκε στις φλόγες. Καταλυτικό παράγοντα εξετέλεσε ο Κολοσσός, ένα τηλεχειριζόμενο αυτόματο πυροσβεστικό ρομπότ, το οποίο μοιάζει με στρατιωτικό τανκ, το οποίο και συνέβαλε στις προσπάθειες της Πυροσβεστικής Υπηρεσίας του Παρισιού για καταπολέμηση της πυρκαγιάς.[66]

Η γνώση του χάρτη κάποιου κτιρίου ή περιοχής μπορεί να αποδειχτεί χρήσιμη σε πάρα πολλές περιπτώσεις, όπως για παράδειγμα για σκοπούς μεταφοράς αντικειμένων από ρομπότ, μετάδοσης πληροφοριών ή σε περιπτώσεις υψηλού κινδύνου, ειδικά όπου μπορεί να κινδυνέψουν ανθρώπινες ζωές. Μια τέτοια περίπτωση ήταν και η κατάσβεση της πυρκαγιάς στο ναό της Παναγίας των Παρισίων.

1.2 Σκοπός εργασίας

Σκοπός της εργασίας αυτής είναι να εξετάσει πώς μπορεί να χαρτογραφήσει μια περιοχή με τη χρήση φτηνού ρομποτικού εξοπλισμού. Στη διάθεση μας έχουμε τον πιο κάτω εξοπλισμό: 2 ρομποτικά οχήματα εδάφους, raspberry pi, arduino, motor shield, αισθητήρες υπερήχων. Τα 2 ρομποτικά οχήματα που έχουμε στη διάθεση μας είναι 2 ρομπότ από την εταιρεία DFRobot. Στο παρόν έγγραφο θα αναφερόμαστε στο ένα ρομπότ ως ρομπότ Λεονάρντο [1] και στο άλλο ως ρομπότ Ούνο.[2].

Αναλυτικότερα, σκοπός είναι το ρομπότ να εξερευνήσει μία περιοχή μέσω εντολών που λαμβάνει απομακρυσμένα από κάποιο χρήστη. Επεξηγώντας˙ το ρομπότ λαμβάνει οδηγίες για το πως και προς τα που να κινηθεί. Καθώς, διασχίζει την περιοχή, εξετάζει το περιβάλλον γύρω του και μαθαίνει πληροφορίες για αυτό. Στόχος είναι η δημιουργία δυσδιάστατου χάρτη με τις περιοχές όπου μπορεί να διακινηθεί ελεύθερα και τις περιοχές όπου υπάρχουν εμπόδια.

Κεφάλαιο 2

Προσέγγιση επίλυσης προβλήματος

2.1 Προκλήσεις

Κατά τη διάρκεια μελέτης του συγκεκριμένου θέματος, εντοπίστηκαν διάφορες προκλήσεις. Πιο συγκεκριμένα εντοπίστηκε ότι τα ρομποτικά οχήματα εδάφους μπορούν να παρουσιάσουν διάφορες ανωμαλίες στην προκαθορισμένη κίνησή τους, καθιστώντας δύσκολη έως ανέφικτη την χρήση μαθηματικών υπολογισμών για τον εντοπισμό της θέσης τους ενώ βρίσκονται σε κίνηση.

Αυτό παρατηρήθηκε κυρίως με το ρομπότ Λεονάρντο, το οποίο είναι κατασκευασμένο με δύο απλούς κυκλικούς τροχούς διαμέτρου 4 εκ. οι οποίοι είναι τοποθετημένοι παράλληλα πάνω σε κυκλική βάση και συνδεδεμένοι με μικρούς κινητήρες για την περιστροφή τους. Στους κινητήρες και στους τροχούς έτυχε πολλές φορές να μπουν ακαθαρσίες από το έδαφος με αποτέλεσμα να εμποδιστεί η ομαλή τους κίνηση, δηλαδή να μην παρουσιάζουν την αναμενόμενη ταχύτητα και κατεύθυνση. Επίσης, παρατηρήθηκε πως ο ένας από τους δύο κινητήρες ήταν πιο ισχυρός σε σχέση με το άλλο με αποτέλεσμα να χρειάζεται να εφαρμοστεί διαφορετική ταχύτητα στο καθένα. Τα παραπάνω οφείλονται και στο γεγονός ότι ο εξοπλισμός που χρησιμοποιήθηκε είναι φτηνός, άρα όχι απόλυτα αξιόπιστος. Ακόμα, ένα πρόβλημα που εντοπίστηκε είναι ότι σε περίπτωση εξάντλησης μέρους της ενέργειας των μπαταριών, επηρεάζεται επίσης η κίνηση των ρομπότ.

Η ίδια μελέτη έγινε και με το ρομποτικό όχημα εδάφους Ούνο, το οποίο ομοιάζει με ένα στρατιωτικό τανκ και κινείται επίσης με κινητήρες αλλά κατά πολύ πιο αξιόπιστα και με ιμάντες που περιστρέφονται σε τροχαλίες. Αυτά τα χαρακτηριστικά, καθιστούν πολύ πιο σταθερή την κίνησή του. Επομένως, το ρομπότ Ούνο επιτρέπει την παραγωγή πιο αξιόλογων αποτελεσμάτων.

Συνακόλουθα, για τη μελέτη της συγκεκριμένης διατριβής επιλέγηκε το ρομπότ Ούνο, διότι επιτρέπει την παραγωγή πιο αξιόλογων αποτελεσμάτων συγκριτικά με το ρομπότ Λεονάρντο.

2.2 Αντιμετώπιση προκλήσεων

Γενικά, στην οδομετρία τείνουμε να αντιμετωπίζουμε αυτού του είδους τα προβλήματα (βλ. τμήμα 2.1) εξαιτίας της ανάγκης για μέτρηση της ταχύτητας στην πάροδό του χρόνου για εκτίμηση της θέσης, καθιστώντας την τεχνική αυτή ευαίσθητη σε σφάλματα. Αυτό μπορεί να αντιμετωπιστεί με τη χρήση επιπλέον τεχνολογικού εξοπλισμού όπως rotary encoder ή gps.

Στην παρούσα εργασία λόγω μη κατοχής κατάλληλου εξοπλισμού για υπολογισμό της ακριβής θέσης του ρομπότ ενώ βρίσκεται σε κίνηση απλοποιήθηκε το πρόβλημα με την αποτροπή της ελεύθερης διακίνησης του ρομποτικού οχήματος σε κάθε κατεύθυνση και την εφαρμογή προκαθορισμένης ταχύτητας και χρόνου διέλευσης.
Συγκεκριμένα, καθορίστηκαν 4 επιτρεπτές κινήσεις:

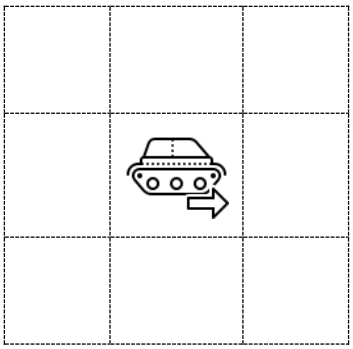
1. Περιστροφή ρομπότ 90˚ αριστερόστροφα ως προς το κέντρο βάσης του
2. Περιστροφή ρομπότ 90˚ δεξιόστροφα ως προς το κέντρο βάσης του
3. Μπροστινή κίνηση απόστασης όσο του μήκους του ρομπότ (22.5 εκ.)
4. Κίνηση όπισθεν απόστασης όσο του μήκους του ρομπότ (22.5 εκ.)

Ο συνδυασμός αυτών των κινήσεων επιτρέπει στο ρομπότ να διακινείται πάνω, κάτω, δεξιά και αριστερά με όπισθεν κίνηση ή με μπροστινή κίνηση.

<u>Επεξήγηση επιτρεπτών κινήσεων με σχήματα:</u>
Στα πιο κάτω σχήματα το ρομπότ παρουσιάζεται με το εξής εικονίδιο:

Το βέλος στο εικονίδιο αυτό δείχνει την κατεύθυνσή της μπροστινής κίνησης για το ρομπότ (στάση).
Η θέση του ρομπότ αντιπροσωπεύεται με την αναπαράσταση του σε ένα πίνακα (θέση).

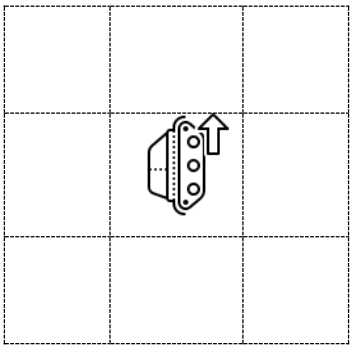
Έστω ότι το ρομπότ ξεκινά σε αυτή τη στάση/θέση:



(στάση: προς τα δεξιά / θέση: 2,2)

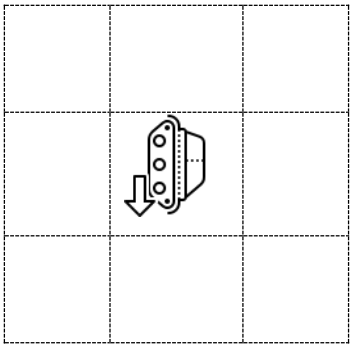
Μπορεί να επιλέξει μία από τις 4 επιτρεπόμενες κινήσεις. Ανάλογα με την κίνηση που θα επιλέξει, θα βρίσκεται στην εξής στάση/θέση:

α) περιστροφή ρομπότ 90˚ αριστερόστροφα ως προς το κέντρο βάσης του



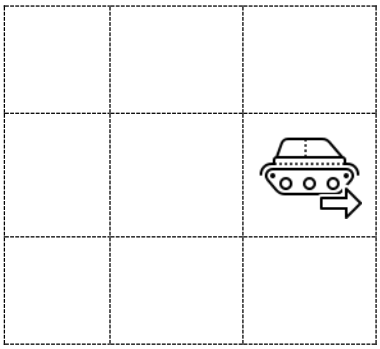
(στάση: προς τα πάνω / θέση: 2,2)

β) περιστροφή ρομπότ 90˚ δεξιόστροφα ως προς το κέντρο βάσης του



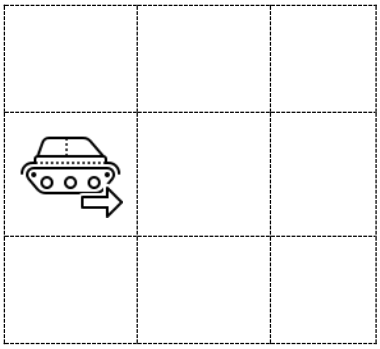
(στάση: προς τα κάτω / θέση: 2,2)

γ) μπροστινή κίνηση απόστασης όσο του μήκους του ρομπότ



(στάση: προς τα δεξιά / θέση: 2,3)

δ) κίνηση όπισθεν απόστασης όσο του μήκους του ρομπότ

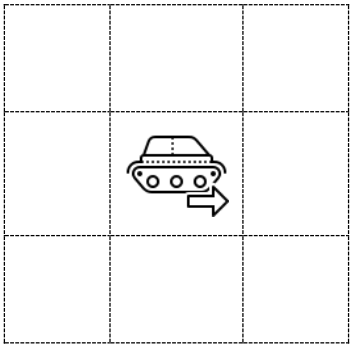


(στάση: προς τα δεξιά / θέση: 2,1)

Ουσιαστικά μετατρέπουμε την κίνησή του σε ελεγχόμενη. Οπόταν, γνωρίζοντας την αρχική θέση του ρομπότ, μπορούμε στη διάρκεια του χρόνου να υπολογίζουμε τη θέση του αναφορικά ως προς την αρχική του θέση, λαμβάνοντας υπόψη το συνδυασμό των κινήσεων που κάνει στην πορεία του.

Παραδείγματος χάρη:

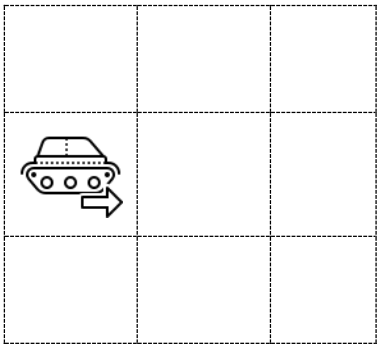
Έστω ότι το ρομπότ ξεκινά σε αυτή τη στάση/θέση:



(στάση: προς τα δεξιά / θέση: 2,2)

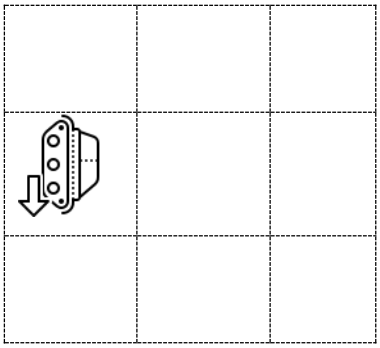
Σειρά κινήσεων και νέα στάση/θέση:

1. Κίνηση όπισθεν απόστασης όσο του μήκους του ρομπότ



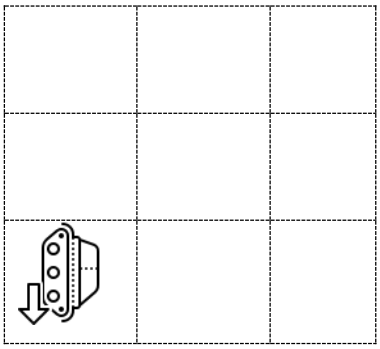
(στάση: προς τα δεξιά / θέση: 2,1)

2. Περιστροφή ρομπότ 90˚ δεξιόστροφα ως προς το κέντρο βάσης του



(στάση: προς τα κάτω / θέση: 2,1)

γ) μπροστινή κίνηση απόστασης όσο του μήκους του ρομπότ



(στάση: προς τα κάτω / θέση: 3,1)

2.3 Κατασκευή αρένας περιήγησης

Για την εξέταση του προβλήματος δημιουργήθηκε μια αρένα για περιήγησή του ρομπότ μέσα σε αυτή με σκοπό τη δημιουργία του χάρτη της αρένας σε ηλεκτρονική μορφή. Η αρένα αυτή είναι κατασκευασμένη με κάποιους περιορισμούς, ούτως ώστε να μειωθεί η περιπλοκότητα στη λειτουργία της δημιουργίας χάρτη της περιοχής περιήγησής του ρομπότ. Πιο κάτω παρατίθενται τα χαρακτηριστικά της αρένας:

1) Αποτελείται από τοίχους και διαδρόμους. Οι τοίχοι είναι φτιαγμένοι από χαρτόκουτα και οι διάδρομοι είναι κενός χώρος στο δάπεδο.

2) Οι τοίχοι έχουν ύψος μεγαλύτερο από το ύψος του ρομπότ για να μπορούν να εντοπίζονται από τους αισθητήρες υπερήχων οι οποίοι είναι τοποθετημένοι στην οροφή του ρομπότ.
3) Το πλάτος των διαδρόμων είναι κατά ελάχιστο μεγαλύτερο από το πλάτος/μήκος του ρομπότ ώστε να το χωράει να κινείται στους διαδρόμους και να περιστρέφεται.

Παράδειγμα αρένας:

2.4 Αναπαράσταση χάρτη

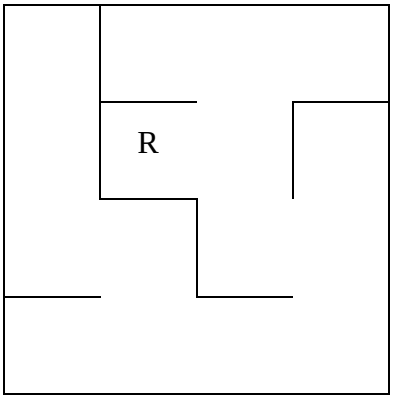
Ο χάρτης δημιουργείται με δύο διαφορετικές αναπαραστάσεις. Οι τρόποι αναπαράστασης που επιλέγηκαν για τη δημιουργία του χάρτη περιοχής, βασίζονται στη παραδοχή ότι η πραγματικότητα μπορεί να μοντελοποιηθεί με τρόπους που να επικοινωνούν αποτελεσματικά τις χωρικές πληροφορίες.

Ο ένας τρόπος αναπαράστασής του χάρτη είναι με σύμβολα (βλ. αναπαράσταση Α). Το (-) αντιπροσωπεύει τον τοίχο, το (ο) αντιπροσωπεύει το διάδρομο και τα (v, ^, >, <) αντιπροσωπεύουν το ρομπότ και συγκεκριμένα την κατεύθυνση της μπροστινής

κίνησης του (προς τα που πρόκειται να κινηθεί αν πάει μπροστά) (v : προς τα κάτω, ^ : προς τα πάνω, > : προς τα δεξιά, < : προς τα αριστερά).

Ο δεύτερος τρόπος αναπαράστασης του χάρτη είναι με γραφικό περιβάλλον επικοινωνίας (βλ. αναπαράσταση Β). Συγκεκριμένα η αρένα παρουσιάζεται με ένα καμβά, όπου όπου υπάρχει γραμμή αναπαριστά τοίχο και όπου υπάρχει κενό στο χρώμα του καμβά αναπαριστά διάδρομο. Το ρομπότ αναπαρίσταται με εικονίδιο και η κατεύθυνσή της μπροστινής κίνησης του με βέλος πάνω στο εικονίδιο με την ανάλογη κατεύθυνση.

Έστω μία αρένα:

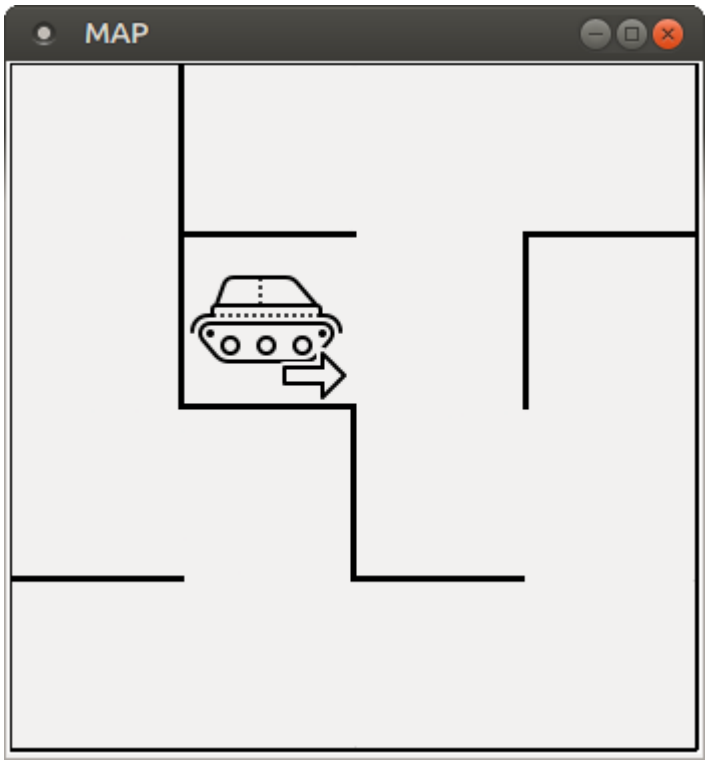


(όπου R συμβολίζει το ρομπότ)

Αναπαράσταση Α: Με σύμβολα

```
  _   _   _   _
| O | O   O   O |
      _       _
| O | >   O | O |
      _
| O   O | O   O |
  _       _
| O   O   O   O |
  _   _   _   _
```

Αναπαράσταση Β: Με γραφικό περιβάλλον επικοινωνίας



2.5 Μετακίνηση ρομπότ

Για τη μετακίνηση του ρομπότ χρησιμοποιούνται 2 διαφορετικοί τρόποι:

Α) Γραφικό περιβάλλον επικοινωνίας το οποίο περιέχει τα εξής 4 κουμπιά:

Κουμπί Turn Left:

Περιστροφή ρομπότ 90˚ αριστερόστροφα ως προς το κέντρο βάσης του

Κουμπί Turn Right:

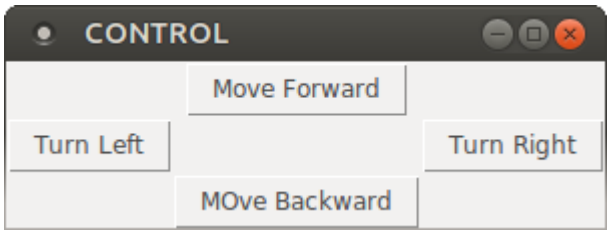
Περιστροφή ρομπότ 90˚ δεξιόστροφα ως προς το κέντρο βάσης του

Κουμπί Forward:

Μπροστινή μετακίνηση ρομπότ απόστασης όσο το μήκος του ρομπότ. Μπροστινή μετακίνηση θεωρείται η μετακίνηση προς την μπροστινή πλευρά του ρομπότ.

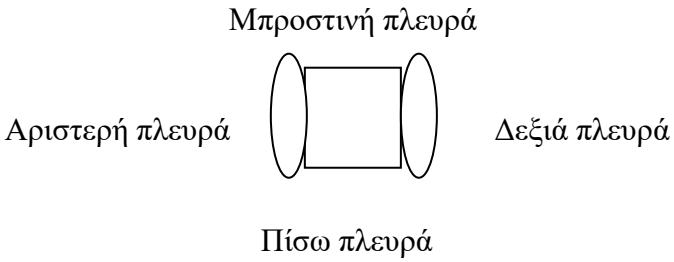
Κουμπί Backward:

Οπίσθια μετακίνηση ρομπότ απόστασης όσο το μήκος του ρομπότ. Οπίσθια μετακίνηση θεωρείται η μετακίνηση προς την πίσω πλευρά του ρομπότ.



Εικόνα γραφικού περιβάλλοντος επικοινωνίας για μετακίνηση ρομπότ.

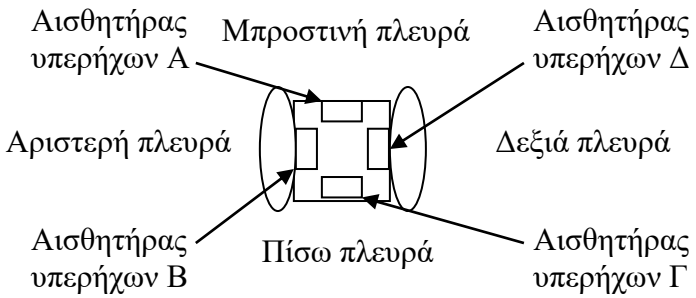
Β) Τα κουμπιά βέλη στο πληκτρολόγιο:
Κουμπί ←: Περιστροφή ρομπότ 90˚ αριστερόστροφα ως προς το κέντρο βάσης του
Κουμπί →: Περιστροφή ρομπότ 90˚ δεξιόστροφα ως προς το κέντρο βάσης του
Κουμπί ↑: Μπροστινή μετακίνηση ρομπότ απόστασης όσο το μήκος του ρομπότ. Μπροστινή μετακίνηση θεωρείται η μετακίνηση προς την μπροστινή πλευρά του ρομπότ.
Κουμπί ↓: Οπίσθια μετακίνηση ρομπότ απόστασης όσο το μήκος του ρομπότ. Οπίσθια μετακίνηση θεωρείται η μετακίνηση προς την οπίσθια πλευρά του ρομπότ.



Σχηματική αναπαράσταση ρομπότ.

2.6 Εντοπισμός τοίχων

Αφού γνωρίζουμε τη θέση του ρομπότ στην αρένα/χάρτη, για τον εντοπισμό των τοίχων αρκεί να τοποθετήσουμε 4 αισθητήρες υπερήχων στην οροφή του ρομπότ σε κάθε μια από τις 4 πλευρές του (μπροστινή, πίσω, αριστερά, δεξιά). Η επιλογή αυτή, μας εξυπηρετεί γιατί καθώς το ρομπότ κινείται στους διαδρόμους και έχει αίσθησή της θέσης του, μπορεί να ελέγχει μπροστά, πίσω, αριστερά και δεξιά του αν υπάρχουν τοίχοι, και να τους τοποθετεί στην ανάλογη θέση με βάση τους δύο τρόπους αναπαράστασης του χάρτη της αρένας.



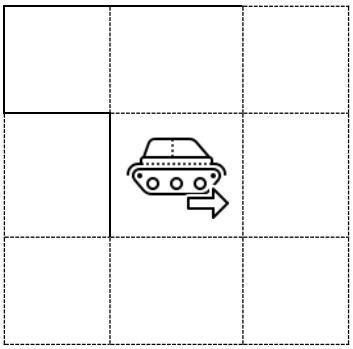
Σχηματική αναπαράσταση τοποθέτησης αισθητήρων υπερήχων στο ρομπότ.

Αναλυτικότερα, το ρομπότ με την εκτέλεση μίας εντολής κίνησης, του παρέχεται η δυνατότητα να μάθει νέες πληροφορίες για το περιβάλλον του. Σε περίπτωση που το ρομπότ εκτελεί μία από τις εντολές περιστροφή 90˚ δεξιόστροφα ή αριστερόστροφα από το κέντρο βάσης του, αλλάζει η στάση του (κατεύθυνση μπροστινής κίνησης), χωρίς να αλλάζει η θέση του. Σε αυτήν την περίπτωση όπου αλλάζει η στάση του ρομπότ, δεν υπάρχει κάποια νέα πληροφορία να μάθει για τους τοίχους που βρίσκονται γύρω από αυτό, καθώς δεν έχει μετακινηθεί σε κάποια νέα θέση. Επομένως, δεν χρειάζεται να ελέγξει για την ύπαρξη τοίχων, αφού γνωρίζει ήδη τις πληροφορίες στο περιβάλλον του. Σε περίπτωση που το ρομπότ εκτελεί μία από τις εντολές μπροστινή ή όπισθεν κίνηση απόστασης όσο το μήκος του ρομπότ, αλλάζει η θέση του, χωρίς να αλλάζει η στάση του. Σε αυτήν την περίπτωση όπου αλλάζει η θέση του ρομπότ, υπάρχουν νέες πληροφορίες να μάθει για τους τοίχους που βρίσκονται γύρω από αυτό, καθώς έχει μετακινηθεί σε κάποια νέα θέση. Επομένως, χρειάζεται να ελέγξει για την ύπαρξη τοίχων, αφού δεν γνωρίζει ήδη τις πληροφορίες στο περιβάλλον του. Οπόταν, εξετάζει για κάθε ένα από τους αισθητήρες υπερήχων του το σήμα που λαμβάνει. Αν η απόσταση που δείχνει κάποιος αισθητήρας υπερήχων, είναι μικρότερη των 10 εκ., τότε θεωρούμε πως υπάρχει τοίχος στην πλευρά του ρομπότ που είναι τοποθετημένος ο αισθητήρας αυτός. Επιλέγηκε η τιμή 10 εκ, γιατί 6 εκ. είναι η απόσταση από τον αισθητήρα υπερήχων μέχρι τον τροχό (ιμάντα) και τα υπόλοιπα 4 εκ. είναι για να χωράει να κινείται στους διαδρόμους.

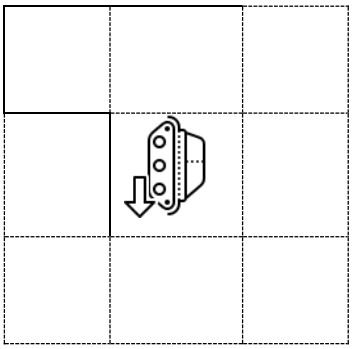
Σχηματική επεξήγηση:

Όπου συνεχής γραμμή, δηλώνει τοίχο. Όπου διακεκομμένη γραμμή, δηλώνει είτε μη ύπαρξη τοίχου, είτε άγνοια για ύπαρξη τοίχου.

Έστω το ρομπότ έχει γνώση μέρους του περιβάλλοντος του:

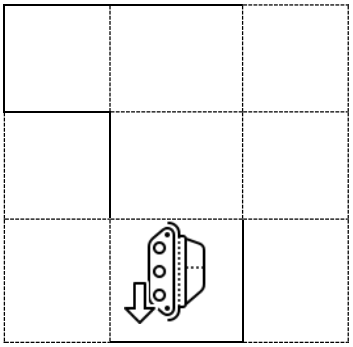


Στη συνέχεια, του δίνεται εντολή να περιστραφεί 90˚ δεξιόστροφα:



Όπως παρατηρούμε, παραμένει στην ίδια θέση και αλλάζει στάση, χωρίς να χρειάζεται να εντοπίσει νέες πληροφορίες στο περιβάλλον του.

Στη συνέχεια, του δίνεται εντολή να κινηθεί ευθεία:

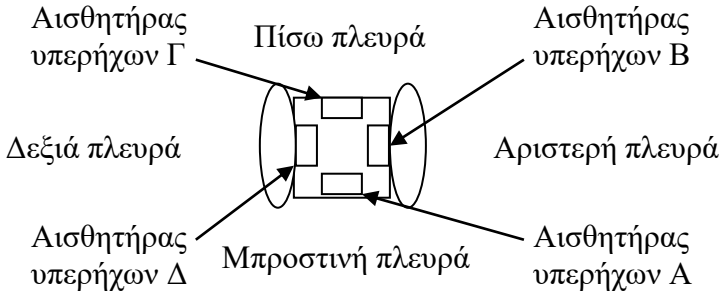


Όπως παρατηρούμε, ενώ διατηρεί την ίδια στάση, αλλάζει θέση και εντοπίζει νέες πληροφορίες στο περιβάλλον του.

Ο τρόπος με τον οποίο γίνεται, η εύρεση της θέσης των τοίχων είναι ο εξής.

Η θέση του ρομπότ είναι η (3,2).

Η στάση (κατεύθυνση μπροστινής κίνησης του) του ρομπότ είναι προς τα κάτω.



Οι αισθητήρες υπερήχων Α και Β, δείχνουν αποστάσεις μικρότερες των 10 εκ., ενώ οι Γ και Δ, μεγαλύτερες των 10 εκ.. Άρα, στην μπροστινή πλευρά και αριστερή πλευρά του ρομπότ έχει τοίχο, ενώ στην πίσω πλευρά και δεξιά πλευρά του ρομπότ δεν έχει τοίχο.

Συνεπώς, γνωρίζοντας τη θέση του ρομπότ στο χάρτη και σε ποιες πλευρές του ρομπότ υπάρχουν τοίχοι, υπολογίζουμε την ακριβή θέση των τοίχων.

Κεφάλαιο 3

Γενική ιδέα υλοποίησης

3.1 Αλγόριθμος:

Ο χάρτης ξεκινά ως κενός.

Τοποθετούμε το ρομπότ σε μια θέση στην αρένα.

Ενόσω το ρομπότ δεν έχει περιηγηθεί σε όλη την αρένα

Δίνουμε μια εντολή κίνησης στο ρομπότ

Ανανεώνουμε τη θέση και την κατεύθυνση του ρομπότ.

Εάν η εντολή κίνησης είναι μπροστινή ή οπίσθια μετακίνηση:

Για όσους αισθητήρες απόστασης εντοπίζουν τοίχο:

Καταγράφουμε τη θέση του τοίχου

Παρουσιάζουμε τη μέχρι τώρα κατασκευή του χάρτη.

Παρουσιάζουμε τον τελικό χάρτη.

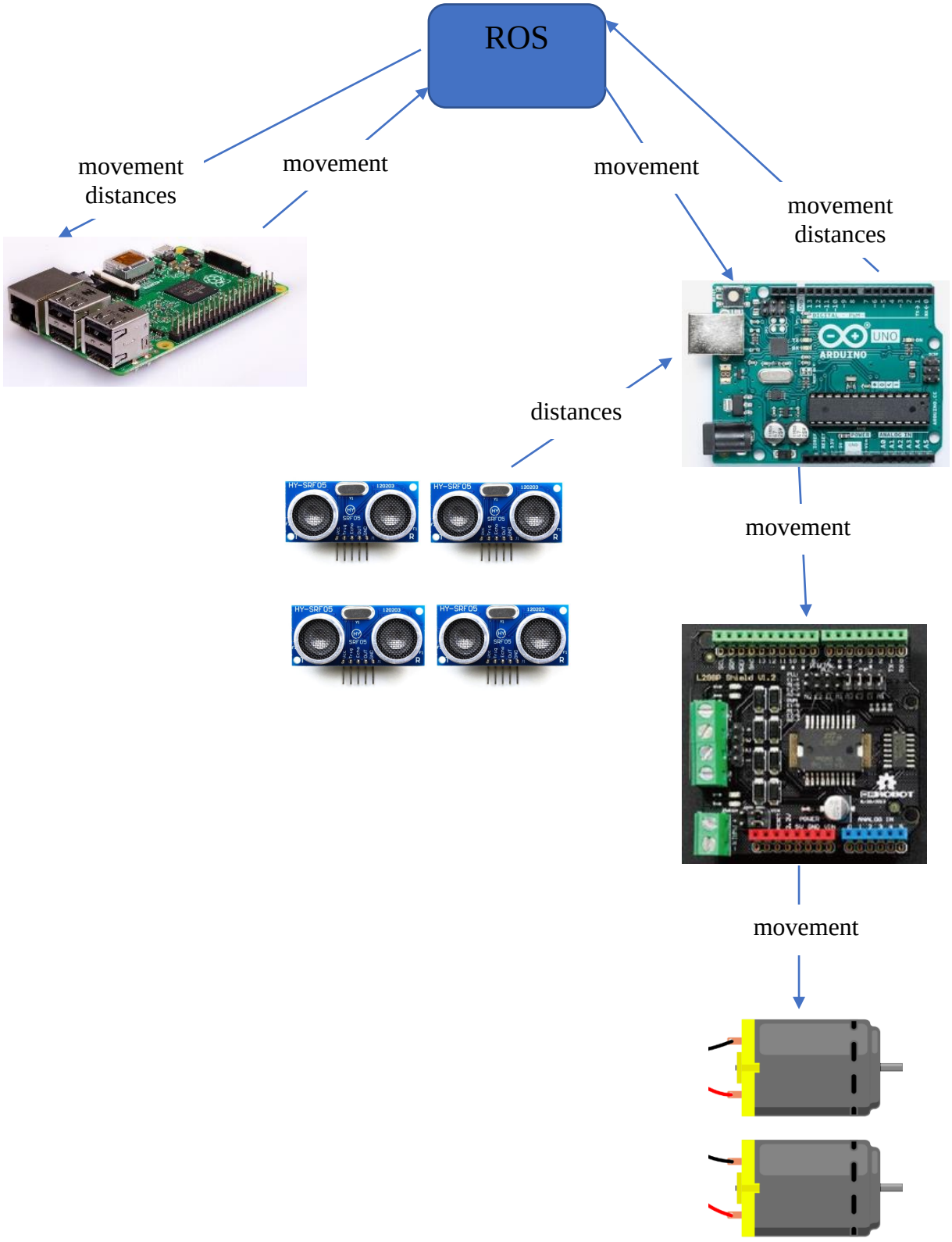
Συνδεσμολογία σύστημάτων:

3.2 Περιγραφή ιδέας υλοποίησης

Η όλη ιδέα της υλοποίησης είναι η εξής:

Ο χρήστης δίνει εντολές μέσω του raspberry pi για την επιθυμητή κίνηση του ρομπότ ώστε να εξερευνήσει την περιοχή – αρένα. Το arduino Uno λαμβάνει την εντολή και δίνει εντολές στους κινητήρες του ρομποτικού οχήματος εδάφους ώστε να πραγματοποιήσουν την εντολή - κίνηση. Ουσιαστικά, το arduino Uno είναι υπεύθυνο για την κίνησή του ρομπότ, το οποίο επιτυγχάνει με τη βοήθεια του 2x2A DC Motor Shield. Με την εκτέλεση κάποιας κίνησης το arduino Uno επικοινωνεί με το raspberry pi ώστε να το ενημερώσει για την κίνηση αυτή, αλλά και για να το ενημερώσει για τις νέες πληροφορίες που λαμβάνει από το περιβάλλον γύρω του μέσω των αισθητήρων υπερήχων. Το raspberry pi τότε ενημερώνει τη γνώση του για τη θέση και στάση του ρομπότ, καθώς επίσης εισάγει και τις νέες πληροφορίες για την ύπαρξη εμποδίων – τοιχών στο χάρτη.

Η όλη επικοινωνία και η ανταλλαγή μηνυμάτων μεταξύ του raspberry pi και του arduino Uno πραγματοποιείται με τη χρήση του λογισμικού ROS.


ROS
movement
distances
movement
movement
movement
distances
distances
movement
movement

Πιο κάτω θα αναλυθεί εξονυχιστικά ο τρόπος επίλυσης του προβλήματος. Θα δοθούν λεπτομέρειες για το ρομποτικό εξοπλισμό, την επικοινωνία των συστημάτων, την τροφοδοσία ρεύματος, τις ανάγκες μνήμης αποθήκευσης, τη χρήση λογισμικού και για το ROS.

Κεφάλαιο 4

Ρομποτικός Εξοπλισμός

4.1 Ρομπότ Λεονάρντο - Ούνο

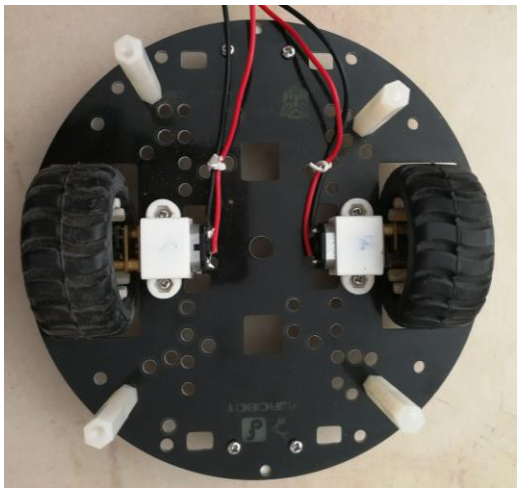
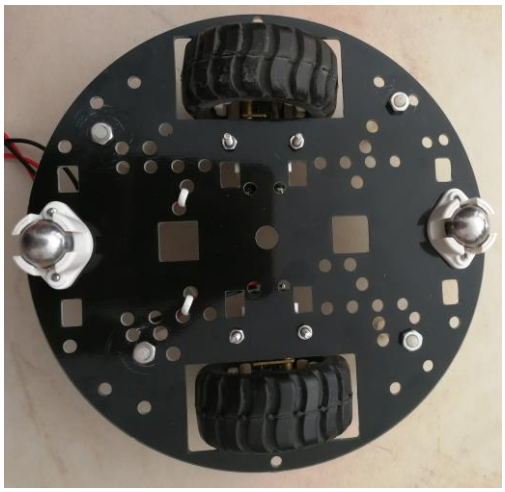
Η DFRobot είναι ένας παροχέας ρομποτικού υλικού και υλικού ανοιχτού λογισμικού. [59]
Από τη συγκεκριμένη εταιρεία προμηθευτήκαμε 2 ρομποτικά οχήματα εδάφους, τα οποία συναρμολογήσαμε.

Συναρμολόγηση ρομπότ Λεονάρντο:
Εξαρτήματα: [60]

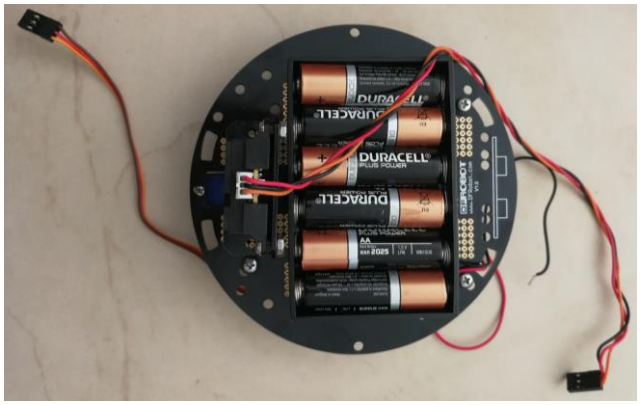
- Romeo V2
- 9g micro servo
- Sharp GP2Y0A21 Distance Sensor
- Sharp IR Sensor Mounting Bracket
- 6Xaa battery holder
- 2wd miniQ Robot chassis
- Upper Deck for MiniQ
- Nylon standoffs
- Nuts
- Screws
- Black nylon gaskets

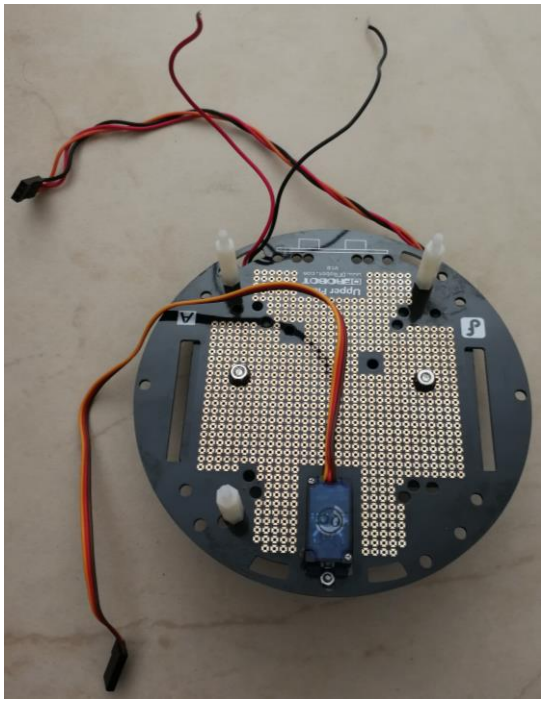
Διαδικασία Συναρμολόγησης:

1. Σύνδεση εξαρτημάτων βάσης.

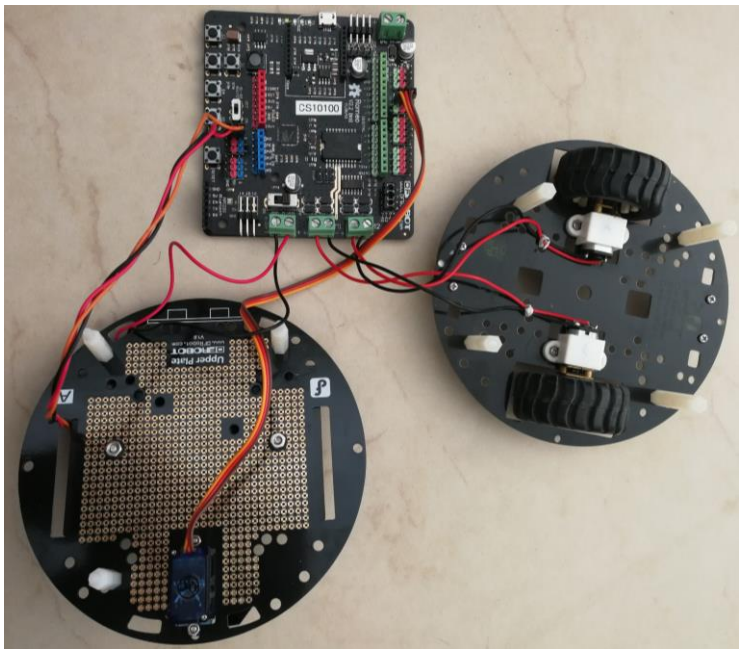



2. Σύνδεση εξαρτημάτων οροφής

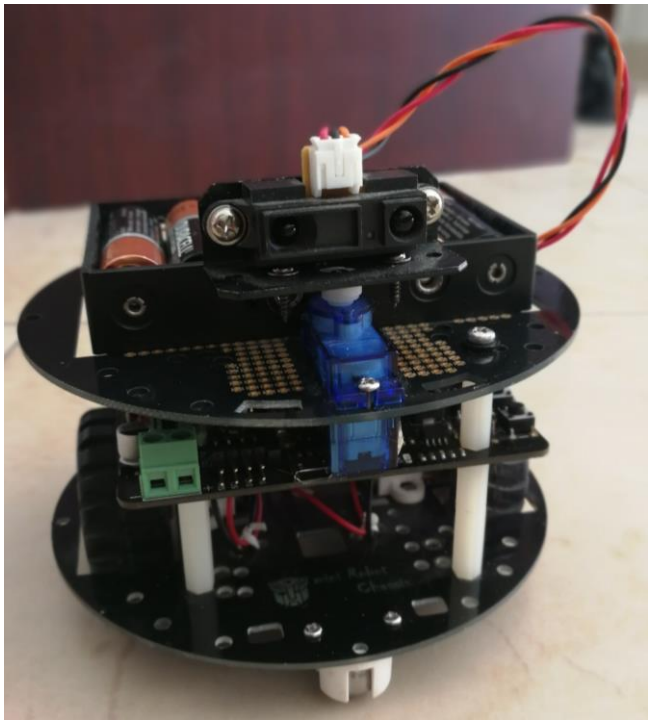




3. Σύνδεση βάσης, πλακέτας Romeo V2, οροφής



4. Τελική συναρμολόγηση



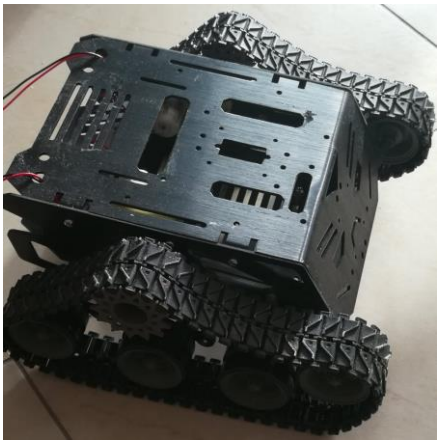
Συναρμολόγηση ρομπότ Ούνο:

Εξαρτήματα:

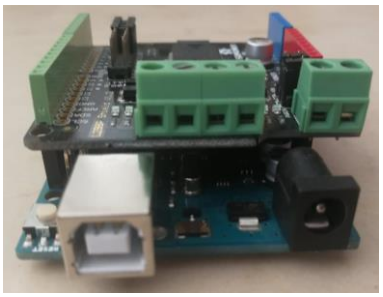
- Υλικά για το Devastator Tank Mobile Robot Platform [2]
- Arduino Uno [8]
- 2x2A DC Motor Shield for Arduino [3]

Διαδικασία Συναρμολόγησης:

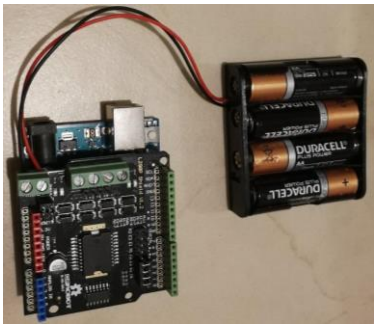
1. Εφαρμογή οδηγιών από το εγχειρίδιο συναρμολόγησης. [4]



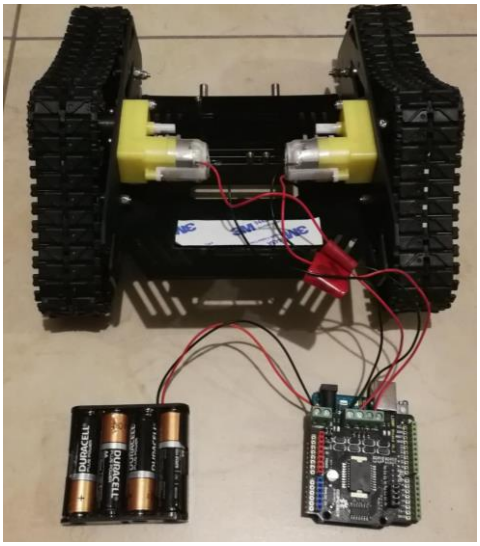
2. Σύνδεση Arduino Uno με 2x2A DC Motor Shield



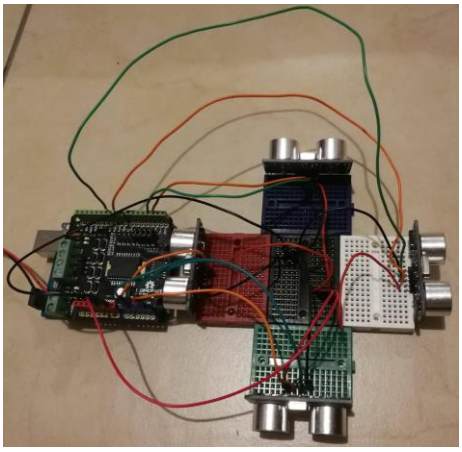
3. Σύνδεση μπαταριών με 2x2A DC Motor Shield



4. Σύνδεση κινητήρων με 2x2A DC Motor Shield



5. Σύνδεση αισθητήρων υπερήχων με breadboards και με 2x2A DC Motor Shield



4.2 Raspberry pi

Raspberry Pi είναι ένας υπολογιστής μικρός σε μέγεθος και οικονομικός σε τιμή ο οποίος μπορεί να χρησιμοποιηθεί για σκοπούς προγραμματισμού. [5]
Στην παρούσα εργασία χρησιμοποιήθηκε το Raspberry Pi 2 Model B, που είναι το δεύτερης γενεάς Raspberry Pi. [6]

Raspberry pi 2 Model B

4.3 Arduino

Το Arduino είναι οικοσύστημα υλικού και λογισμικού ανοιχτού κώδικα. Προσφέρει μια σειρά από εργαλεία λογισμικού, πλατφόρμες υλικού και τεκμηρίωση που επιτρέπει τη δημιουργία τεχνολογικών προϊόντων όπως η ανάπτυξη προϊόντων IoT (Internet of Things) ή bεργαλεία για την εκπαίδευση STEM (Science, Technology, Engineering, Mathematics) / STEAM (Science, Technology, Engineering, Arts, Mathematics). [7]

Στην παρούσα εργασία χρησιμοποιήθηκε το Arduino Uno. Το Arduino Uno είναι μία πλακέτα μικροελεγκτών που βασίζεται στον μικροεπεξεργαστή ATmega328P. Διαθέτει 14 ψηφιακές ακίδες εισόδου / εξόδου (από τις οποίες 6 μπορούν να χρησιμοποιηθούν ως έξοδο PWM), 6 αναλογικές εισόδους, 16 MHz κρυστάλλου χαλαζία, σύνδεση ενιαίου σειριακού διαύλου (USB), υποδοχή τροφοδοσίας ρεύματος, κεφαλίδα ICSP και κουμπί επαναφοράς. Για τη ρευματοδότησή του, συνδέεται είτε με έναν υπολογιστή με καλώδιο USB ή τροφοδοτείται με έναν προσαρμογέα εναλλασσόμενου ρεύματος (AC - alternating current) σε συνεχές ρεύμα (DC - direct current) ή μπαταρία. [8]

PWM (Pulse Width Modulation):

Είναι μια τεχνική για την επίτευξη αναλογικών αποτελεσμάτων μέσω ψηφιακών μέσων κατά την οποία δημιουργείται ένα τετράγωνο κύμα (ένα σήμα το οποίο εναλλάσσεται μεταξύ ενεργοποιημένου και απενεργοποιημένου) προσομοιώνοντας με αυτόν τον τρόπο τις τάσεις ανάμεσα στα 5 Volts και στα 0 Volts. Μεταβάλλοντας, δηλαδή, το τμήμα του χρόνου που το σήμα είναι ενεργοποιημένο (πλάτος παλμού) σε σχέση με το χρόνο που είναι απενεργοποιημένο, επιτυγχάνονται οι διάφορες αναλογικές τιμές. Με την γρήγορη επανάληψη του μοτίβου ενεργοποίησης – απενεργοποίησης, το σήμα είναι σαν να παίρνει μια σταθερή τάση μεταξύ 0 και 5v. [62]

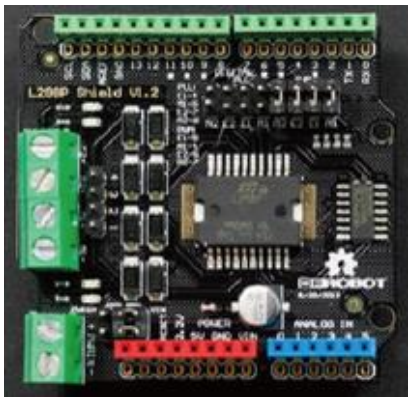
USB (Universal Serial Bus):

Ενιαίος Σειριακός Δίαυλος

ICSP (In-circuit serial programming)

4.4 2x2A DC Motor Shied

Το 2x2A DC Motor Shield για Arduino επιτρέπει τον έλεγχο κινητήρα με το Arduino. Το 2x2A DC Motor Shield τοποθετείται πάνω στο Arduino και χρησιμοποιεί το τσιπ L298P που επιτρέπει την οδήγηση 2 κινητήρων συνεχούς ρεύματος (DC motors) με υψηλότερες απαιτήσεις ισχύος (7-12V DC με μέγιστο ρεύμα 2A).
Ο έλεγχος ταχύτητας επιτυγχάνεται μέσω συμβατικού PWM που μπορεί να ληφθεί από τα pins εξόδου 5 και 6 του PWM του Arduino. Η λειτουργία ενεργοποίησης / απενεργοποίησης του ελέγχου κινητήρα σηματοδοτείται από τα ψηφιακά pin του Arduino 4 και 7. [3]



4.5 Αισθητήρες υπερήχων

Η μέτρηση της απόστασης από τους τοίχους πραγματοποιείται με τη χρήση αισθητήρων υπερήχων. Η αρχή της μέτρησης απόστασης από υπέρηχο γίνεται με τον εξής τρόπο: ο πομπός υπερήχων εκπέμπει παλμό σε μία κατεύθυνση και ξεκινά η χρονομέτρηση, το υπερηχητικό κύμα μόλις συναντήσει εμπόδιο στον αέρα επιστρέφει στο δέκτη υπερήχων και όταν ο δέκτης λάβει το ανακλώμενο κύμα, η χρονομέτρηση σταματάει. Αν η ταχύτητα διάδοσης του υπερηχητικού κύματος στον αέρα είναι v και η απόσταση S μεταξύ του σημείου μετάδοσης και του εμποδίου μπορεί να υπολογιστεί σύμφωνα με

τη διαφορά χρόνου Δt που καταγράφεται από το χρονόμετρο που μεταδίδει και δέχεται την ηχώ (echo), το S μπορεί να υπολογιστεί ως εξής: S = v · Δt / 2 [63]
Διαιρούμε δια 2 γιατί το ζητούμενο μας είναι η απόσταση από τον αισθητήρα υπερήχων μέχρι το εμπόδιο, χωρίς να συνυπολογίζουμε το χρόνο που χρειάζεται το κύμα να επιστρέψει αφού αντανακλαστεί.

Οι αισθητήρες υπερήχων που χρησιμοποιήθηκαν είναι οι HY-SRF05. [9]

Ο αισθητήρας υπερήχων HY-SRF05 αποτελείται από 5 ακίδες (pins) οι οποίες αντιπροσωπεύουν τα Vcc, Trig, Echo, OUT, GND.
Το Vcc συνδέεται με ρεύμα τάσης 5 V.
Το Trig χρησιμοποιείται για να στείλει το υπερηχητικό κύμα και συνδέεται με κάποιο pin το οποίο δίνει την εντολή για να σταλεί το κύμα.
Το Echo χρησιμοποιείται για να λάβει το υπερηχητικό κύμα που αποστέλλει το Trig και συνδέεται με κάποιο pin που θα λάβει την τιμή του κύματος.
Το OUT δεν το χρησιμοποιούμε.
Το GND συνδέεται με τη γείωση (GND).

Συνδεσμολογία κυκλώματος στο σύστημα μας:

Τα pins των αισθητήρων υπερήχων συνδέονται με καλώδια πάνω σε breadboards και έπειτα από τα breadboards με καλώδια πάω στο 2x2A DC Motor Shield.
Ο λόγος που συνδέονται πάνω σε breadboards και μετά στο 2x2A DC Motor Shield είναι για να διευκολύνει την τροφοδοσία τους με ρεύμα και για να μπορούμε να τοποθετήσουμε τους αισθητήρες υπερήχων με ευκολία και σταθερότητα πάνω στην οροφή του ρομπότ.

Τα pin 9, 2, A0, A3 του 2x2A DC Motor Shield χρησιμοποιούνται ως trigger pins.

Τα pin 10, 3, A1, A4 του 2x2A DC Motor Shield χρησιμοποιούνται ως echo pins.

Τα trigger pins και τα echo pins λειτουργούν ως ομάδες 9 - 10, 2 - 3, A0 - A1, A3 - A4

Για κάθε ένα από τους 4 αισθητήρες υπερήχων αντιστοιχεί μια από τις 4 ομάδες trigger pin - echo pin.

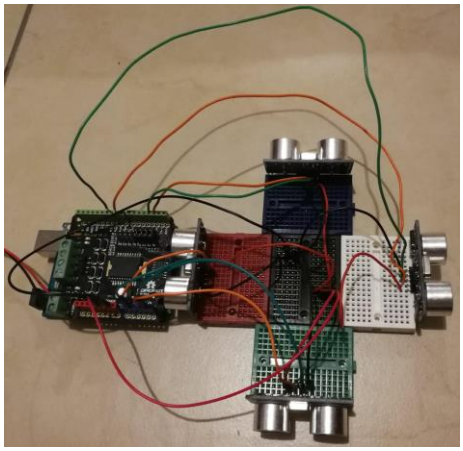
Ανά ομάδα:

Το trigger pin του 2x2A DC Motor Shield ενώνεται με το trigger pin του αισθητήρα υπερήχων.

Το echo pin του 2x2A DC Motor Shield ενώνεται με το echo pin του αισθητήρα υπερήχων.

Το ground του 2x2A DC Motor Shield ενώνεται με το ground του αισθητήρα υπερήχων.

Το 5V του 2x2A DC Motor Shield ενώνεται με το 5V του αισθητήρα υπερήχων.



Κεφάλαιο 5

Επικοινωνία συστημάτων

Επικοινωνία Raspberry pi – Arduino Uno:

Το Raspberry pi επικοινωνεί με το Arduino Uno μέσω του Arduino UART (Universal Asynchronous Receiver/Transmitter)

Γενικά το Arduino UART χρησιμοποιείται για επικοινωνία μεταξύ της πλακέτας Arduino και ενός υπολογιστή ή άλλων συσκευών. Όλες οι πλακέτες Arduino έχουν τουλάχιστον μία σειριακή θύρα. Στο Uno, τα pins 0 και 1 χρησιμοποιούνται για επικοινωνία με τον υπολογιστή (0 – RX δέκτης , 1 – TX πομπός). Συνδέοντας οτιδήποτε με αυτές τις ακίδες μπορεί να επηρεάσει την επικοινωνία αυτή, όπως επίσης, και να προκαλέσει αποτυχημένες μεταφορτώσεις του προγράμματος στην πλακέτα.

Η βιβλιοθήκη SoftwareSerial επιτρέπει την σειριακή επικοινωνία με οποιαδήποτε ψηφιακό pin του Uno. [10]

Η σύνδεση του Raspberry pi με το Arduino Uno γίνεται με προσαρμογέα USB σε σειριακό (USB-to-serial adaptor). Συγκεκριμένα χρησιμοποιούμε το USB 2.0 Cable Type A/B. [11]

USB 2.0 Cable Type A/B

Κεφάλαιο 6

Τροφοδοσία ρεύματος

6.1 Arduino Uno

Τροφοδοτούμε την πλακέτα Arduino Uno με εξωτερική παροχή ρεύματος μέσω προσαρμογέα εναλλασσόμενου ρεύματος σε συνεχές ρεύμα (AC-to-DC adapter). Ο προσαρμογέας συνδέεται στο Arduino Uno συνδέοντας ένα κεντρικό βύσμα 2,1 χιλ. στην υποδοχή τροφοδοσίας της πλακέτας.

Η πλακέτα μπορεί να λειτουργεί με εξωτερική τροφοδοσία από 6 έως 20 βολτ. Ωστόσο, εάν τροφοδοτείται με λιγότερα από 7V, το 5V pin ενδέχεται να παρέχει λιγότερα από 5 βολτ και η πλακέτα μπορεί να γίνει ασταθής. Αν τροφοδοτείται με περισσότερα από 12V, ο ρυθμιστής τάσης μπορεί να υπερθερμανθεί και να προκαλέσει βλάβη στην πλακέτα. Το συνιστώμενο εύρος τιμών είναι 7 έως 12 βολτ. [8]

Επομένως, επιλέγουμε να τροφοδοτήσουμε το arduino Uno μέσω του πιο πάνω τρόπου με 9V.

Εναλλακτικά, τροφοδοτούμε το arduino Uno μέσω της σύνδεσης USB με 5V.

6.2 2x2A DC Motor Shield

Τροφοδοτείται με ρεύμα τάσης 5V από το Arduino. [3]

6.3 Raspberry pi

Το Raspberry pi Model B χρησιμοποιεί 700-1000mA αναλόγως των περιφερειακών συσκευών που είναι συνδεδεμένο. Η μέγιστη ισχύς που μπορεί να χρησιμοποιήσει είναι 1 Amp. Αν χρειάζεται να συνδεθεί με μία συσκευή USB η οποία θα έχει απαιτήσεις ισχύος πάνω από 1 Amp, τότε πρέπει να συνδεθεί σε ένα εξωτερικό τροφοδοτικό USB hub. [12]

Εδώ τροφοδοτούμε το raspberry pi με ρεύμα τάσης 5V.

6.4 Αισθητήρες υπερήχων HY-SRF05

Οι αισθητήρες υπερήχων HY-SRF05 τροφοδοτούνται με ρεύμα τάσης 5V από το 2x2A DC Motor Shield.

Κεφάλαιο 7

Λογισμικό

7.1 Λογισμικό για Raspberry pi

Debian:

Λειτουργικό σύστημα το οποίο χρησιμοποιεί κυρίως είτε τον πυρήνα Linux είτε τον πυρήνα FreeBSD και συνοδεύεται με πάνω από 51000 πακέτα, ένα διαχειριστή πακέτων (APT) και άλλα βοηθητικά προγράμματα για τη διαχείριση των πακέτων στους υπολογιστές. [13]

Ubuntu Mate:

Ubuntu είναι ένα λειτουργικό σύστημα επιτραπέζιων υπολογιστών που βασίζεται στο Linux. Το Ubuntu MATE είναι ένα σταθερό, εύκολο στη χρήση λειτουργικό σύστημα (έχει το βασικό λειτουργικό σύστημα Ubuntu) με διαμορφωμένο περιβάλλον επιφάνειας εργασίας (MATE Desktop). Το MATE Desktop είναι μια παραδοσιακή μεταφορική επιφάνεια εργασίας. Το Ubuntu MATE έχει μικρές απαιτήσεις υλικού καθιστώντας το κατάλληλο για σύγχρονους σταθμούς εργασίας, υπολογιστές μονής πλακέτας και παλαιότερο υλικό. [14,15]

Το Raspbian είναι το επίσημο λειτουργικό σύστημα για όλα τα μοντέλα του Raspberry Pi. [16] Συνεπώς, η εγκατάσταση του ROS Kinetic στο Raspberry Pi 2 θα μπορούσε να γίνει με το Raspbian Jessie. Ωστόσο, σήμερα είναι ταχύτερη και ευκολότερη η χρήση του Ubuntu Mate 16.04 μαζί με την αρχιτεκτονική ARM . Κατά συνέπεια επιλέγουμε το λειτουργικό σύστημα Ubuntu Mate 16.04. [61]

7.2 Λογισμικό για Arduino

Το Arduino και το Arduino IDE είναι εργαλεία για γρήγορο και εύκολο προγραμματισμό υλικού.

Arduino IDE:

Λογισμικό ανοιχτού κώδικα Arduino που επιτρέπει την συγγραφή κώδικα για προγράμματα και τη μεταφόρτωση του σε πλατφόρμα υλικού. Είναι συμβατό με Windows, Mac OS X και Linux και μπορεί να χρησιμοποιηθεί με οποιοδήποτε πλακέτα Arduino.

Το Arduino IDE καθιστά εύκολο το γράψιμο του κώδικα και τη μεταφόρτωσή του στην πλακέτα Arduino. [17]

7.3 ROS

[Εκτενής επεξήγηση στο κεφάλαιο 9]
Η βασική λειτουργία του προγράμματός μας στηρίζεται στο λογισμικό ROS.
Το ROS είναι ένα πλαίσιο λογισμικού ανοιχτού κώδικα που αναπτύχθηκε και συντηρείται από το Ίδρυμα Ρομποτικής Ανοιχτού Κώδικα (Open Source Robotics Foundation). Ως ένα από τα πιο δημοφιλή έργα για τη ρομποτική, το ROS μπορεί να τρέξει σε διάφορα λειτουργικά συστήματα που βασίζονται σε Unix, συμπεριλαμβανομένων των διανομών Linux και του Mac OS X. Περιέχει μια σειρά από βιβλιοθήκες, πακέτα και αλγόριθμους. Το ROS δεν έχει μόνο ένα σταθερό σύστημα διαχείρισης πακέτων, αλλά και ένα προηγμένο πλαίσιο υπηρεσίας μεταφοράς μηνυμάτων, το οποίο ενισχύει τη διαλειτουργικότητα των πακέτων του.[64]

Το ROS ως τεχνολογία αναπτύχθηκε το 2007 και από τότε έχει συναντήσει ευρεία εφαρμογή σε πάρα πολλά έργα. Κύριος λόγος για αυτήν την τεράστια αποδοχή και χρήση του από την επιστημονική κοινότητα, είναι και ο ίδιος ο σκοπός της δημιουργίας του, δηλαδή να διευκολύνει τη δημιουργία εφαρμογών ρομπότ μέσα από την παροχή εργαλείων που στοχεύουν κυρίως στην εύκολη επικοινωνία των διαφόρων ρομποτικών μερών. Απότοκα, αυτού, είναι η εξοικονόμηση χρόνου ανάπτυξης λογισμικού και η επικέντρωση στον πυρήνα της ανάπτυξης ρομπότ ή εφαρμογών ρομπότ. Επομένως, το ROS αποτελεί πλέον μία κοινή πλατφόρμα χρήσης μεταξύ των επιστημόνων, προγραμματιστών κ.λπ. γεφυρώνοντας με αυτόν το τρόπο το κενό που υπήρχε στη συμβατότητα των διαφόρων έργων ανάπτυξης λογισμικού με χρήση ρομπότ.

Για τους παραπάνω λόγους επιλέγηκε να γίνει χρήση της τεχνολογίας ROS και στην παρούσα διπλωματική εργασία.

Ως γλώσσα προγραμματισμού, για την ανάπτυξη λογισμικού, χρησιμοποιούμε την Python, η οποία είναι μια απλή αλλά ισχυρή προγραμματιστική γλώσσα η οποία χρησιμοποιείται στο ROS. Το ROS έχει εγγενή υποστήριξη χαμηλού επιπέδου για την

Python. Οι κόμβοι (nodes) και τα θέματα (topics) μπορούν να χρησιμοποιηθούν σε ένα script Python χρησιμοποιώντας raspy, μια βιβλιοθήκη πελάτη Python και API (Application Programming Interface). Έτσι, τα δεδομένα μπορούν να μεταφερθούν μεταξύ διαφορετικών ενοτήτων και πακέτων. [69]

Κεφάλαιο 8

Μνήμη αποθήκευσης

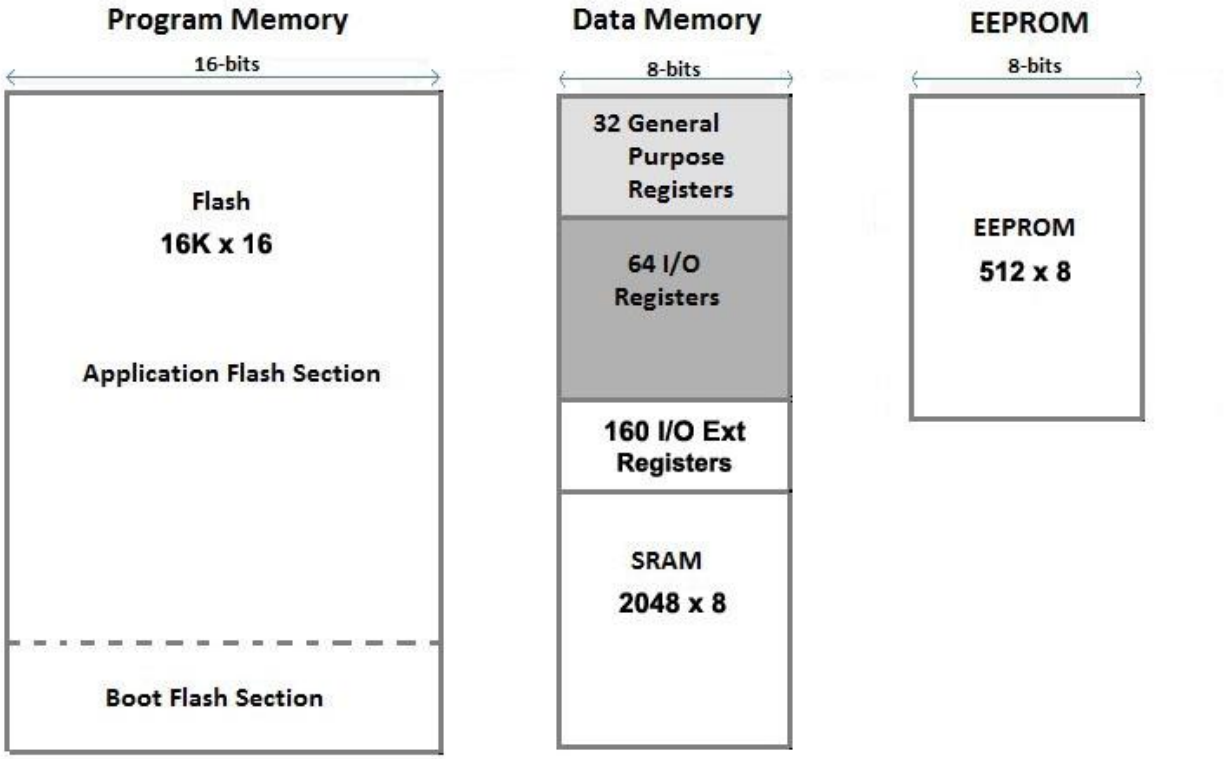
8.1 Micro SDXC Card

Επιλέγουμε κάρτα μνήμης micro SDXC χωρητικότητας 64 GB.

Το λειτουργικό σύστημα Ubuntu Mate για το Raspberry Pi Μοντέλο Β προϋποθέτει τη χρήση κάρτας microSD χωρητικότητας 8GB ή μεγαλύτερη ώστε να χωράει η εικόνα του λειτουργικού συστήματος. Το σύστημα αρχείων αλλάζει αυτόματα το μέγεθος ώστε να καταλαμβάνει το μη εκχωρημένο χώρο της κάρτας microSD. [18]

8.2 Μνήμη ATmega328

Το Arduino Uno έχει επεξεργαστή τον ATmega328 ο οποίος έχει διαθέσιμη μνήμη 32 KB (τα 0.5 KB καταλαμβάνονται από το bootloader). Διαθέτει επίσης 2 KB SRAM και 1 KB EEPROM (τα οποία μπορούν να διαβαστούν και να εγγραφούν με τη βιβλιοθήκη EEPROM) [8,19]



Ο χάρτης μνήμης ενός ATmega328

Από τη μνήμη του Arduino Uno χρησιμοποιούμε 11302 bytes (35%) από το χώρο αποθήκευσης προγραμμάτων. Το μέγιστο είναι 32256 bytes. Χρησιμοποιούμε επίσης 1478 bytes για global μεταβλητές (72%) της δυναμικής μνήμης, αφήνοντας 570 bytes για τοπικές μεταβλητές. Το μέγιστο είναι 2048 bytes.

Κεφάλαιο 9

ROS

9.1 Τι είναι ROS

Το ROS (Robot Operating System) είναι ένα ευέλικτο πλαίσιο για την απλούστευση της δημιουργίας περίπλοκων και ισχυρών εφαρμογών ρομπότ σε μια μεγάλη ποικιλία ρομποτικών πλατφόρμων. Αυτό επιτυγχάνεται μέσα από την παροχή μιας συλλογής εργαλείων όπως αφαιρετικότητα υλικού, προγράμματα οδήγησης συσκευών, βιβλιοθήκες, οπτικοακουστικά μέσα, πέρασμα μηνυμάτων, διαχείριση πακέτων και άλλων συμβάσεων. Το ROS έχει άδεια ανοικτού κώδικα λογισμικού BSD. [20,21]

9.2 Έκδοση ROS Kinetic Kame

Είναι η δέκατη διανομή ROS. Διανεμήθηκε στις 23 Μαΐου 2016. Αποτελείται από ένα σετ εκδόσεων πακέτων ROS. Επικεντρώνεται στη διανομή του Ubuntu 16.04 (Xenial). [22]

9.3 Πακέτα Rosserial Arduino Kinetic και rosseria_arduino

Rosserial Arduino Kinetic:
Πρωτόκολλο για την περιτύλιξη τυποποιημένων σειριακών μηνυμάτων ROS και την πολυπλεξία πολλαπλών θεμάτων και υπηρεσιών μέσω μιας συσκευής χαρακτήρων όπως μια σειριακή θύρα ή πρίζα δικτύου. Σε αυτό απαντώνται τρεις τύποι πακέτων: Βιβλιοθήκες Πελατών, Διεπαφές ROS, Παραδείγματα και χρήση σεναρίων. Οι βιβλιοθήκες πελατών επιτρέπουν στους χρήστες να χρησιμοποιούν εύκολα κόμβους ROS σε διάφορα συστήματα. Αυτοί οι πελάτες είναι θύρες της γενικής βιβλιοθήκης ANSI C ++ rosserial_client. Οι διεπαφές ROS επιτρέπουν στις συσκευές που χρησιμοποιούν rosserial κώδικα να έχουν έναν κόμβο στο μηχάνημα κεντρικού υπολογιστή (host machine) για να γεφυρώνει τη σύνδεση από το σειριακό πρωτόκολλο στο γενικότερο δίκτυο ROS. [67]

rosserial_arduino:
Το πακέτο αυτό περιέχει εξειδικευμένες επεκτάσεις Arduino που απαιτούνται για την εκτέλεση του rosserial_client σε ένα Arduino. Σκοπό έχει την εύκολη ενσωμάτωση

αυτοσχέδιου υλικού και φτηνών αισθητήρων σε ένα έργο ROS με τη χρήση ενός Arduino. [68]

Επιπλέον, το πακέτο rosserial_arduino, επιτρέπει την απευθείας χρήση του ROS με το Arduino IDE. Πιο συγκεκριμένα, το rosserial παρέχει ένα πρωτόκολλο επικοινωνίας για το ROS, το οποίο λειτουργεί με το UART του Arduino. Κατά συνέπεια, με αυτόν τον τρόπο, παρέχεται η δυνατότητα στο Arduino να λειτουργεί ως ένας πλήρης κόμβος ROS, ο οποίος μπορεί να δημοσιεύσει και να εγγραφεί απευθείας σε μηνύματα ROS, να δημοσιεύσει μετασχηματισμούς TF και να ενημερωθεί για το χρόνο του συστήματος ROS. [54]

Μετασχηματισμός TF:

Είναι ένα πακέτο που επιτρέπει στον χρήστη να παρακολουθεί πολλαπλά πλαίσια συντεταγμένων με την πάροδο του χρόνου. Διατηρεί τη σχέση μεταξύ πλαισίων συντεταγμένων σε μια δομή δέντρου ρυθμισμένη με το χρόνο και επιτρέπει στον χρήστη να μετασχηματίζει σημεία, διανύσματα κλπ μεταξύ οποιωνδήποτε δύο πλαισίων συντεταγμένων σε οποιοδήποτε επιθυμητό χρονικό σημείο. [23]

9.4 Βασικές έννοιες - Υπόβαθρο

Catkin workspace:

Ένας χώρος εργασίας catkin είναι ένας φάκελος όπου μπορούν να τροποποιηθούν, να δημιουργηθούν και να εγκαταστήσουν πακέτα catkin. Μπορεί να περιέχει έως και τέσσερις διαφορετικούς χώρους οι οποίοι εξυπηρετούν διαφορετικό ρόλο στη διαδικασία ανάπτυξης λογισμικού.

- Source space: περιέχει τον πηγαίο κώδικα των πακέτων catkin. Κάθε φάκελος εντός του source space περιέχει ένα ή περισσότερα πακέτα catkin.
- Build space: εδώ γίνεται η επίκληση του CMake για την κατασκευή (build) των πακέτων catkin στο source space.
- Development (Devel) space: εδώ τοποθετούνται οι κατασκευασμένοι στόχοι (built targets) προτού εγκατασταθούν.
- Install space: μόλις κατασκευαστούν οι στόχοι, μπορούν να εγκατασταθούν εδώ.

- Result space: γενικός όρος για την αναφορά σε ένα φάκελο ο οποίος μπορεί να είναι development space ή install space. [24]

ROS Package:

Το λογισμικό στο ROS είναι οργανωμένο σε πακέτα. Ένα πακέτο μπορεί να περιέχει κόμβους ROS, μια βιβλιοθήκη ανεξάρτητη από ROS, ένα σύνολο δεδομένων, αρχεία ρυθμίσεων, ένα λογισμικό τρίτου μέρους ή οτιδήποτε άλλο αποτελεί χρήσιμη ενότητα. Ο στόχος αυτών των πακέτων είναι η παροχή αυτής της χρήσιμης λειτουργικότητας με εύκολο τρόπο κατανάλωσης, ώστε το λογισμικό να μπορεί να επαναχρησιμοποιηθεί εύκολα. Σε γενικές γραμμές, τα πακέτα ROS ακολουθούν μια αρχή "Goldilocks": αρκετή λειτουργικότητα για να είναι χρήσιμη, αλλά όχι πάρα πολύ που το πακέτο να είναι βαρύ και δύσκολο να χρησιμοποιηθεί από άλλο λογισμικό. [25]

Ένα πακέτο πρέπει να πληροί τις εξής προϋποθέσεις:

- περιέχει ένα αρχείο package.xml συμβατό με το catkin.
- περιέχει ένα αρχείο CMakeLists.txt που χρησιμοποιεί το catkin.
- κάθε πακέτο πρέπει να έχει δικό του φάκελο. Αυτό σημαίνει ότι δεν υπάρχουν ενσωματωμένα πακέτα, ούτε πολλά πακέτα που μοιράζονται τον ίδιο κατάλογο. [44]

Τα πακέτα ROS τείνουν να ακολουθούν μια κοινή δομή. Ορισμένοι από τους καταλόγους και τα αρχεία τους είναι:

- include/package_name: C++ περιλαμβάνει επικεφαλίδες (headers)
- msg/: Φάκελος που περιλαμβάνει τύπους μηνυμάτων (Message (msg) types)
- src/package_name/: Αρχεία πηγαίου κώδικα, ειδικά σε Python τα οποία εξάγονται σε άλλα πακέτα.
- srv/: Φάκελος που περιέχει τύπους υπηρεσιών (service (srv) types)
- scripts/: εκτελέσιμα scripts
- CMakeLists.txt: CMake build file (catkin/CMakeLists.txt)
- package.xml: Package catkin/package.xml
- CHANGELOG.rst: Πολλά πακέτα ορίζουν ένα changelog το οποίο μπορεί να εγχυθεί αυτόματα σε δυαδικό πακέτο και στη σελίδα wiki για το πακέτο [25]

package.xml:

Το package manifest είναι ένα αρχείο XML που ονομάζεται package.xml που πρέπει να συμπεριληφθεί στον φάκελο-ρίζα του πακέτου-catkin. Αυτό το αρχείο ορίζει ιδιότητες σχετικά με το πακέτο, όπως το όνομα του πακέτου, οι αριθμοί έκδοσης, οι συγγραφείς, οι συντηρητές και οι εξαρτήσεις σε άλλα πακέτα catkin.

Υπάρχει ένα ελάχιστο σύνολο ετικετών που πρέπει να ενσωματωθούν μέσα στην ετικέτα <package> για να γίνει πλήρης το πακέτο.

Οι ετικέτες αυτές είναι:

<name> - Το όνομα του πακέτου

<version> - Ο αριθμός έκδοσης του πακέτου (απαιτείται να είναι 3 ακέραιοι αριθμοί που χωρίζονται από τελεία)

<description> - Περιγραφή του περιεχομένου του πακέτου

<maintainer> - Το όνομα του/των ατόμου/ών που συντηρεί το πακέτο

<license> - Οι άδειες χρήσης λογισμικού (π.χ. GPL, BSD, ASL) υπό τις οποίες απελευθερώνεται ο κώδικας.

Τα πακέτα μπορούν να έχουν έξι τύπους εξαρτήσεων:

- Build Dependencies καθορίζουν τα πακέτα που χρειάζονται για την κατασκευή αυτού του πακέτου.
- Build Export Dependencies καθορίζουν τα πακέτα που χρειάζονται για την κατασκευή βιβλιοθηκών σε αυτό το πακέτο.
- Execution Dependencies καθορίζουν τα πακέτα που απαιτούνται για την εκτέλεση κώδικα σε αυτό το πακέτο
- Test Dependencies ορίζουν μόνο πρόσθετες εξαρτήσεις για δοκιμές μονάδων (unit tests).
- Build Tool Dependencies καθορίζουν τα εργαλεία κατασκευής του συστήματος (build system tools) τα οποία το πακέτο αυτό χρειάζεται για να κτιστεί (build).
- Documentation Tool Dependencies καθορίζουν τα εργαλεία τεκμηρίωσης που χρειάζεται το πακέτο για τη δημιουργία τεκμηρίωσης.

Αυτοί οι έξι τύποι εξαρτήσεων καθορίζονται χρησιμοποιώντας τις ακόλουθες αντίστοιχες ετικέτες:

<depend> καθορίζει ότι μια εξάρτηση είναι μια εξάρτηση κατασκευής, εξαγωγής και εκτέλεσης, <buildtool_depend>, <build_depend>, <build_export_depend>, <exec_depend>, <test_depend>, <doc_depend> [26]

CMakeLists.txt:

Το αρχείο CMakeLists.txt είναι η είσοδος στο σύστημα CMake build για τη δημιουργία πακέτων λογισμικού. Κάθε πακέτο-CMake περιέχει ένα ή περισσότερα αρχεία CMakeLists.txt που περιγράφουν τον τρόπο κατασκευής του κώδικα και τον τόπο εγκατάστασης του.

Το αρχείο CMakeLists.txt πρέπει να ακολουθήσει αυτή τη μορφή με τη συγκεκριμένη σειρά.

1. Required CMake Version (cmake_minimum_required)
2. Package Name (project())
3. Find other CMake/Catkin packages needed for build (find_package())
4. Enable Python module support (catkin_python_setup())
5. Message/Service/Action Generators (add_message_files(), add_service_files(), add_action_files())
6. Invoke message/service/action generation (generate_messages())
7. Specify package build info export (catkin_package())
8. Libraries/Executables to build (add_library()/add_executable()/target_link_libraries())
9. Tests to build (catkin_add_gtest())
10. Install rules (install()) [27]

catkin_make:

Είναι ένα εργαλείο για την κατασκευή κώδικα (build code) σε ένα χώρο εργασίας catkin (catkin workspace). Το catkin_make ακολουθεί την τυποποιημένη διάταξη τους χώρου εργασίας catkin. Πάντα το catkin_make καλείται στη ρίζα (root) του χώρου εργασίας catkin. Με την εκτέλσεση του catkin_make, δημιουργούνται δύο νέοι φακέλοι στη ρίζα του χώρου εργασίας catkin: οι build και devel φάκελοι. Ο φάκελος build είναι ο χώρος

όπου εκτελείται το cmake και το make και ο φάκελος devel περιέχει όλα τα παραγόμενα αρχεία και τους στόχους, καθώς και setup.* sh αρχεία ώστε να μπορεί να χρησιμοποιηθεί σαν να είναι εγκατεστημένο. [28]

9.5 Τρόπος λειτουργίας

ROS node:

Ένας κόμβος είναι μια διαδικασία που εκτελεί υπολογισμό. Οι κόμβοι συνδυάζονται μαζί σε ένα γράφημα και επικοινωνούν μεταξύ τους χρησιμοποιώντας θέματα συνεχούς ροής (streaming topics), υπηρεσίες RPC (Remote Procedure Call) και διακομιστή παραμέτρων (parameter server). Ένα σύστημα ελέγχου ρομπότ περιλαμβάνει συνήθως πολλούς κόμβους.

Η χρήση κόμβων στο ROS παρέχει πολλά οφέλη στο συνολικό σύστημα αφού το εφοπλίζει με τα εξής χαρακτηριστικά:

1. Πρόσθετη ανοχή σφάλματος καθώς οι συγκρούσεις απομονώνονται σε μεμονωμένους κόμβους.
2. Η πολυπλοκότητα του κώδικα μειώνεται σε σύγκριση με τα μονολιθικά συστήματα.
3. Οι λεπτομέρειες εφαρμογής είναι καλά κρυμμένες καθώς οι κόμβοι εκθέτουν ένα ελάχιστο API (Application Programming Interface) στο υπόλοιπο γράφημα.
4. Οι εναλλακτικές υλοποιήσεις, ακόμη και σε άλλες γλώσσες προγραμματισμού, μπορούν εύκολα να αντικατασταθούν.

Όλοι οι κόμβοι που βρίσκονται σε εκτέλεση (running nodes) έχουν ένα όνομα πόρων γραφήματος (graph resource name) που τους ταυτοποιεί με μοναδικό τρόπο στο υπόλοιπο σύστημα. Οι κόμβοι έχουν επίσης έναν τύπο κόμβου, ο οποίος απλοποιεί τη διαδικασία παραπομπής σε ένα εκτελέσιμο κόμβο στο σύστημα αρχείων. Αυτοί οι τύποι κόμβων είναι ονόματα πόρων πακέτων (package resource names) με το όνομα του κόμβου του πακέτου και το όνομα του κόμβου του εκτελέσιμου αρχείου.

Ένας κόμβος ROS γράφεται με τη χρήση μιας βιβλιοθήκης πελατών ROS, όπως roscpp ή rospy. [29]

ROS Messages:

Οι κόμβοι επικοινωνούν μεταξύ τους δημοσιεύοντας μηνύματα (messages) σε θέματα (topics). Ένα μήνυμα είναι μια απλή δομή δεδομένων, που περιλαμβάνει πεδία. Οι τυπικοί πρωταρχικοί τύποι (standard primitive types) (integer, floating point, boolean, etc.) υποστηρίζονται, όπως και οι πίνακες πρωτόγονων τύπων. Τα μηνύματα μπορούν να περιλαμβάνουν αυθαίρετες ένθετες δομές και πίνακες (arbitrarily nested structures and arrays) (όπως οι δομές C).

Οι κόμβοι μπορούν επίσης να ανταλλάξουν ένα μήνυμα αίτησης και απάντησης ως μέρος μιας κλήσης υπηρεσίας ROS. Αυτά τα μηνύματα αίτησης και απάντησης ορίζονται στα αρχεία srv.
Τα αρχεία msg είναι απλά αρχεία κειμένου για τον προσδιορισμό της δομής δεδομένων ενός μηνύματος. Αυτά τα αρχεία αποθηκεύονται στον υποφάκελο msg του πακέτου. [30]

ROS Topics:

Τα θέματα (topics) είναι ονομασμένοι δίοδοι μέσω των οποίων οι κόμβοι ανταλλάσσουν μηνύματα. Τα topics έχουν ανώνυμη σημασιολογία δημοσίευσης / εγγραφής (publish/subscribe), η οποία αποσυνδέει την παραγωγή πληροφοριών από την κατανάλωσή της. Σε γενικές γραμμές, οι κόμβοι δεν γνωρίζουν με ποιον επικοινωνούν. Αντ 'αυτού, οι κόμβοι που ενδιαφέρονται για δεδομένα, εγγράφονται στο σχετικό θέμα και οι κόμβοι που παράγουν δεδομένα, δημοσιεύονται στο σχετικό θέμα. Μπορούν να υπάρχουν πολλοί εκδότες (publishers) και συνδρομητές (subscribers) σε ένα topic.
Τα topics προορίζονται για επικοινωνία μονής κατεύθυνσης. Οι κόμβοι που χρειάζονται να εκτελέσουν απομακρυσμένες κλήσεις διαδικασίας, δηλαδή να λαμβάνουν απάντηση σε ένα αίτημα, πρέπει να χρησιμοποιούν υπηρεσίες (services) αντί topics. Υπάρχει επίσης ο διακομιστής παραμέτρων για τη διατήρηση μικρών ποσοτήτων κατάστασης.
Το ROS υποστηρίζει τη μεταφορά μηνυμάτων που βασίζεται σε TCP / IP και UDP. Η μεταφορά βασισμένη στο TCP / IP είναι γνωστή ως TCPROS και μεταδίδει δεδομένα μηνυμάτων μέσω συνεχών συνδέσεων TCP / IP. Το TCPROS είναι η προεπιλεγμένη μεταφορά που χρησιμοποιείται στο ROS και είναι η μόνη μεταφορά που χρειάζονται να υποστηρίζουν οι βιβλιοθήκες πελάτη. Η μεταφορά UDP, η οποία είναι γνωστή ως UDPROS και επί του παρόντος υποστηρίζεται μόνο από το roscpp, χωρίζει τα

μηνύματα σε πακέτα UDP. Το UDPROS είναι μια μεταφορά χαμηλών λανθάνοντων χρόνων, η οποία είναι καταλληλότερη για εργασίες όπως η τηλεμεταφορά. [31]

Ros Master:

Το ROS Master παρέχει υπηρεσίες ονοματοδοσίας και εγγραφής στους υπόλοιπους κόμβους του συστήματος ROS. Παρακολουθεί εκδότες (publishers) και συνδρομητές (subscribers) σε θέματα (topics) καθώς και σε υπηρεσίες (services). Ο ρόλος του Master είναι να επιτρέψει σε μεμονωμένους κόμβους ROS να εντοπίσουν ο ένας τον άλλον. Μόλις οι κόμβοι αυτοί εντοπίσουν ο ένας τον άλλον, επικοινωνούν μεταξύ τους peer-to-peer. Το Master παρέχει επίσης τον διακομιστή παραμέτρων (Parameter Server). Το πρόγραμμα Master συνήθως εκτελείται χρησιμοποιώντας την εντολή roscore, η οποία φορτώνει το Master ROS μαζί με άλλα βασικά στοιχεία. [32]

ROS core:

Το roscore είναι μια συλλογή από κόμβους και προγράμματα που είναι προαπαιτούμενα ενός συστήματος ROS. Για να επικοινωνούν οι κόμβοι ROS, πρέπει να υπάρχει ένα roscore να εκτελείται, το οποίο εκκινείται χρησιμοποιώντας την εντολή roscore. Με τη χρήση του roslaunch, σε περίπτωση που ανιχνεύσει ότι δεν εκτελείται ήδη το roscore, τότε το ξεκινάει αυτόματα.
Το roscore ξεκινάει:

- ένα ROS Master
- ένα ROS Parameter Server
- ένα rosout logging node [33]

ROS Parameter Server:

Ο διακομιστής παραμέτρων είναι ένα κοινόχρηστο λεξικό πολλαπλών επιλογών που είναι προσβάσιμο μέσω API δικτύου. Οι κόμβοι χρησιμοποιούν αυτόν τον διακομιστή για την αποθήκευση και ανάκτηση παραμέτρων κατά το χρόνο εκτέλεσης. Καθώς δεν έχει σχεδιαστεί για υψηλής απόδοσης, χρησιμοποιείται καλύτερα για στατικά, μη δυαδικά δεδομένα όπως είναι οι παράμετροι διαμόρφωσης. Έχει σκοπό να είναι ορατός παντού, ώστε τα εργαλεία να μπορούν εύκολα να επιθεωρήσουν την κατάσταση διαμόρφωσης του συστήματος και να τροποποιήσουν εάν είναι απαραίτητο. [34]

rosout:

Το rosout είναι το όνομα του μηχανισμού αναφοράς αρχείων καταγραφής σε κονσόλα (console log reporting mechanism) στο ROS. [35]

ROS run:

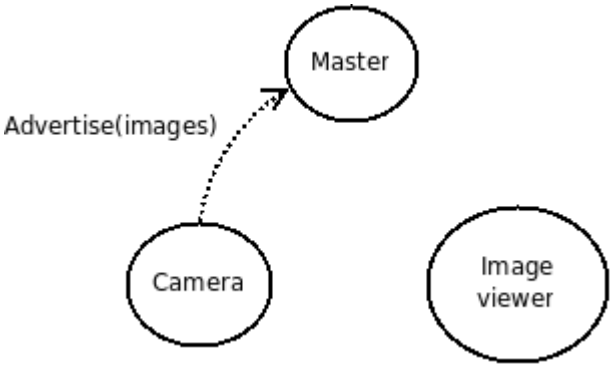
Το rosrun επιτρέπει την εκτέλεση ενός εκτελέσιμου αρχείου σε ένα αυθαίρετο πακέτο από οπουδήποτε χωρίς να χρειάζεται να δοθεί πρώτα η πλήρη διαδρομή του ή το cd / roscd.

Χρήση: rosrun <package> <executable> [36]

9.6 Σχηματική επεξήγηση λειτουργίας ROS

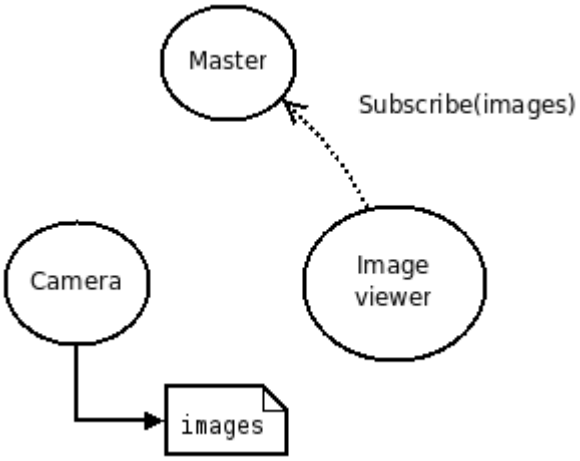
Θα εξετάσουμε ένα παράδειγμα [15] όπου έχουμε δύο κόμβους (Nodes). Έναν κόμβο κάμερα και έναν κόμβο Image_viewer.

Σε πρώτο στάδιο η κάμερα ενημερώνει το master ότι θέλει να εκδόσει (publish) εικόνες στο θέμα (topic) "images":

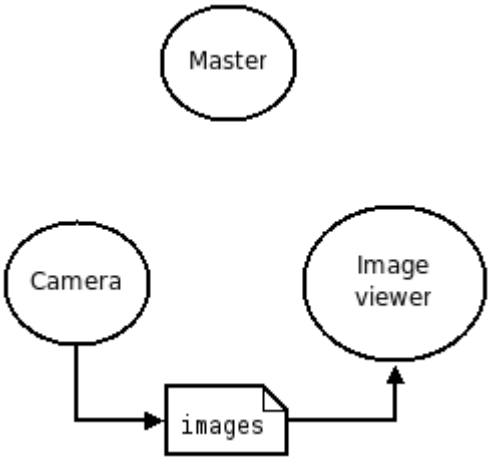


Επομένως, η κάμερα τώρα εκδίδει (publish) εικόνες στο "images" topic. Ωστόσο, δεν έχει εγγραφεί κανένας σε αυτό το topic ακόμη, επομένως δεν έχουν σταλεί δεδομένα.

Σε δεύτερο στάδιο, ο Image_viewer ζητάει να εγγραφεί στο topic "images" για να δει αν υπάρχουν εικόνες εκεί:



Εφόσον τώρα το topic "images" έχει και publisher και ένα subscriber, ο master node ειδοποιεί την Camera και το Image_viewer για την ύπαρξη του κάθε ενός, ώστε να μπορούν να μπορούν να μεταφέρουν εικόνες ο ένας στον άλλο:



Κεφάλαιο 10

Διαδικασία εγκατάστασης λογισμικού

10.1 Μορφοποίηση κάρτας μνήμης

Η κάρτα μνήμης, όπως έχει προαναφερθεί, είναι micro SDXC card 64 GB. Μορφοποιούμε την κάρτα μνήμης σε μορφή συστήματος αρχείων fat32 για να επιτρέπεται η εγκατάσταση του λειτουργικού συστήματος.
Σύμφωνα με τις προδιαγραφές SD, οποιαδήποτε κάρτα SD χωρητικότητας μεγαλύτερης από 32GB θεωρείται ως κάρτα SDXC. Οι κάρτες SDXC πρέπει να μορφοποιηθούν με το σύστημα αρχείων exFAT, πράγμα το οποίο συνεπάγεται ότι πάντοτε το επίσημο εργαλείο μορφοποίησης SD διαμορφώνει κάρτες χωρητικότητας 64GB ή μεγαλύτερες ως exFAT. [37]
Εντούτοις, στο Raspberry Pi βρίσκεται ενσωματωμένος στη μονάδα επεξεργασίας γραφικών (Graphics Processing Unit, GPU) ο φορτωτής εκκίνησης/λειτουργικού (bootloader) ο οποίος είναι μη ανανεώσιμος και υποστηρίζει μόνο την ανάγνωση από συστήματα αρχείων FAT (FAT16 και FAT32), ενώ δεν μπορεί να εκκινήσει από ένα σύστημα αρχείων exFAT. Επομένως, προτού προχωρήσουμε σε οποιαδήποτε αντιγραφή αρχείων εκκίνησης συστήματος στην κάρτα μνήμης των 64GB, η μετατροπή της κάρτας μνήμης σε μορφή FAT32 είναι αναγκαία.
Συνεπώς, για το σκοπό αυτό χρησιμοποιήθηκε ένα πλήρως εξοπλισμένο εργαλείο τρίτων για επεξεργασία διαμερισμάτων (partitions) μνήμης, μιας και τα προκαθορισμένα εργαλεία μορφοποίησης που είναι ενσωματωμένα στα Windows έχουν περιορισμένες δυνατότητες, καθώς επιτρέπουν την μορφοποίηση μόνο διαμερισμάτων μέχρι 32GB ως FAT32. [38]
Το πρόγραμμα το οποίο χρησιμοποιήθηκε είναι το EaseUS Partition Master 13.0 [39]

10.2 Εγκατάσταση Ubuntu Mate

Λαμβάνουμε την εικόνα της SD κάρτας μνήμης για το λειτουργικό συστήμα Ubuntu Mate 16.04.2 για το Raspberry Pi Model B 2 κατάλληλη για επεξεργαστή ARMv7 32-bit. [40]
Η προτιμώμενη μέθοδος εγκατάστασης για το Ubuntu είναι τα πακέτα Debian για διάφορες πλατφόρμες Ubuntu όπως WILY (Ubuntu 15.10), Xenial (Ubuntu 16.04) και Jessie (Debian 8), μιας και είναι πιο αποδοτικά από τις κατασκευές που βασίζονται σε

πηγές (source-based builds). Τα πακέτα του Ubuntu είναι κατασκευασμένα για τις παρακάτω διανομές και αρχιτεκτονικές.

Distro	amd64	i386	Armhf
Wily	X	X	
Xenial	X	X	X

Όπως, έχει επισημανθεί η επιλογή του λειτουργικού συστήματος Ubuntu Mate 16.04.2 θεωρείται η κατάλληλη σε συνδυασμό με την έκδοση ROS Kinetic του λογισμικού ROS. Άρα, χρησιμοποιούμε το πακέτο Debian για την πλατφόρμα Xenial (Ubuntu 16.04) για αρχιτεκτονική armhf (ARMv7 32-bit) [42]

10.3 Εγκατάσταση Arduino IDE

Λαμβάνουμε το λογισμικό Arduino IDE για Linux ARM 32 bits (έκδοση ARDUINO 1.8.9) [41]
Εγκαθιστούμε το Arduino IDE στο raspberry pi.

10.4 Εγκατάσταση ROS

Εγκαθιστούμε το ROS Kinetic.[42]
Εγκαθιστούμε το Rosserial Arduino Kinetic.[54]

Κεφάλαιο 11

Επίλυση προβλήματος

11.1 ROS στο σύστημα μου

Για τη λειτουργία του συστήματος μας χρειαζόμαστε 3 κόμβους ROS:

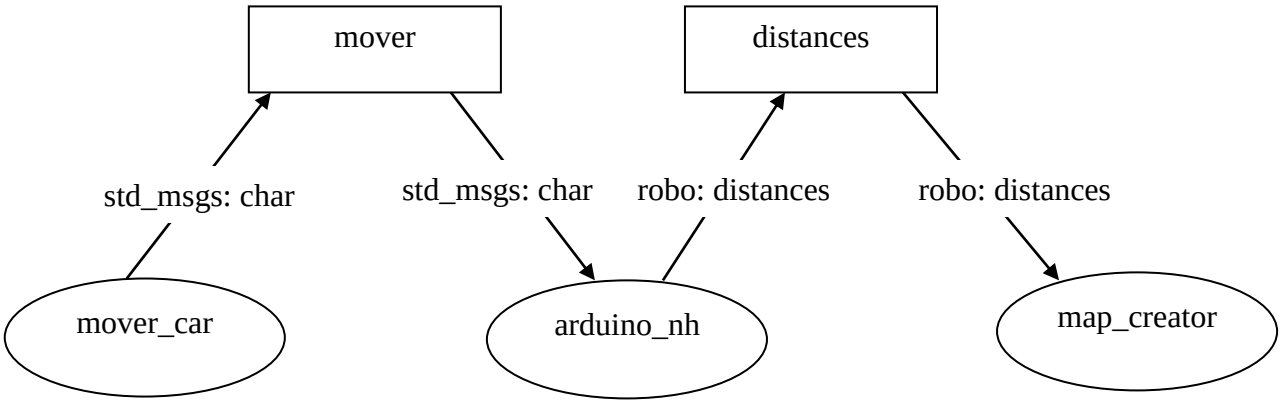
- Ένας κόμβος δίνει εντολές για την κίνησή του ρομπότ (mover_car).
- Ένας κόμβος ελέγχει τους κινητήρες τροχών του ρομπότ (adruino_nh).
- Ένας κόμβος δημιουργεί το χάρτη (map_creator).

Ο mover_car και ο map_creator τρέχουν πάνω στο raspberry pi, ενώ ο arduino_nh τρέχει πάνω στο arduino Uno.

Υπάρχουν 2 topics για να πραγματοποιηθεί η επικοινωνία μεταξύ των κόμβων.

- Mover: Ο τύπος των μηνυμάτων του είναι μια μεταβλητή char από το πακέτο std_msgs.
- Distances: Ο τύπος των μηνυμάτων του είναι προσαρμοσμένο (custom) μήνυμα distances το οποίο περιλαμβάνει μια μεταβλητή char η οποία αντιπροσωπεύει την κίνηση που πραγματοποίησε το ρομπότ και ένα πίνακα 4 θέσεων τύπου uint16 ο οποίος αντιπροσωπεύει τις 4 αποστάσεις που καταγράφονται από τους αισθητήρες υπερήχων.

Ο κόμβος mover_car εκδίδει (publish) μηνύματα στο topic mover.
Ο κόμβος arduino_nh εγγράφεται (subscribe) για παραλαβή μηνυμάτων από το topic mover και εκδίδει (publish) μηνύματα στο topic pub_distances.
Ο map_creator εγγράφεται (subscribe) για παραλαβή μηνυμάτων από το topic pub_distances.



Σχηματική επεξήγηση χρήσης ROS για επίλυση του προβλήματος

Διαδικασία χαρτογραφίας:

Ο χάρτης ξεκινά ως κενός.

Τοποθετούμε το ρομπότ σε μια θέση στην αρένα.

Ενόσω το ρομπότ δεν έχει περιηγηθεί σε όλη την αρένα

Δίνουμε μια εντολή κίνησης στο ρομπότ

Ο κόμβος mover_car εκδίδει μήνυμα στο topic mover με την εντολή κίνησής του ρομπότ.

Ο κόμβος arduino_nh παραλαμβάνει μήνυμα από το topic mover για το ποια κίνηση πρέπει να εκτελέσει το ρομπότ.

Το ρομπότ εκτελεί την κίνηση.

Ο arduino_nh εκδίδει μήνυμα στο topic pub_distances με την κίνηση που έχει εκτελέσει και τις αποστάσεις που καταγράφει από τους αισθητήρες υπερήχων.

Ο map_creator παραλαμβάνει μήνυμα από το topic pub_distances ώστε να ανανεώσει τη θέση/στάση του ρομπότ και τους τοίχους στο χάρτη.

Ανανεώνουμε τη θέση και την κατεύθυνση του ρομπότ στο χάρτη.

Εάν η εντολή κίνησης είναι μπροστινή ή οπίσθια μετακίνηση:

Για όσους αισθητήρες απόστασης εντοπίζουν τοίχο:

Καταγράφουμε τη θέση του τοίχου

Παρουσιάζουμε τη μέχρι τώρα κατασκευή του χάρτη.

11.2 Προετοιμασία περιβάλλοντος εργασίας ROS

1) Δημιουργούμε το χώρο εργασίας για την κατασκευή του πακέτου catkin [43]

2) Δημιουργούμε πακέτο ROS

Το πακέτο catkin ορίζεται ως τέτοιο εάν και εφόσον περιέχει ένα αρχείο package.xml συμβατό με το catkin και ένα αρχείο CMakeLists.txt το όποιο χρησιμοποιεί το catkin. Το αρχείο package.xml χρησιμοποιείται με σκοπό την παροχή μετα-πληροφοριών σχετικά με το πακέτο. Ταυτόχρονα, κάθε πακέτο πρέπει να έχει δικό του φάκελο το οποίο σημαίνει πως δεν υποστηρίζονται ενσωματωμένα πακέτα ούτε πολλά πακέτα τα οποία μοιράζονται τον ίδιο κατάλογο. [44]

3) Κατασκευή (Build) πακέτου χρησιμοποιώντας το catkin_make

Το catkin_make είναι ένα εργαλείο γραμμής εντολών το οποίο διευκολύνει την τυπική ροή εργασίας του catkin. Ουσιαστικά, συνδυάζει τις κλήσεις του cmake και make ώστε να αναπαραγάγει την τυποποιημένη ροή εργασίας του CMake. [45]

4) Δημιουργία κόμβων - εκτελέσιμων αρχείων

Ένας κόμβος είναι ένα εκτελέσιμο που χρησιμοποιεί το ROS για να επικοινωνεί με άλλους κόμβους. Το εκτελέσιμο αρχείο βρίσκεται μέσα σε ένα πακέτο ROS. Οι κόμβοι ROS χρησιμοποιούν τις βιβλιοθήκες πελατών του ROS, rospy, rosserial_arduino για να επικοινωνούν με άλλους κόμβους και να μπορούν να δημοσιεύσουν ή να εγγραφούν σε ένα θέμα. [46]
Για την υλοποίησή του συστήματος μας δημιουργούμε, συνεπώς 3 εκτελέσιμα αρχεία στη γλώσσα προγραμματισμού python:

- mover_gui.py: Δίνει εντολές για την κίνησή του ρομπότ (κόμβος mover_car).
- arduino.ino: Ελέγχει τους κινητήρες τροχών του ρομπότ (κόμβος adruino_nh).
- map_creator_gui.py: Δημιουργεί το χάρτη (κόμβος map_creator).

Τα αρχεία αυτά περιλαμβάνουν:

- Ros Topics [47]
- Publishers και subscribers με βάση το rosserial Arduino [48,49,50]
- Messages (msg) [51]
  Standard και custom messages με βάση το rosserial Arduino [52,53,54,55]
- Γραφικό περιβάλλον επικοινωνίας

5) Εκκίνηση αρχείων

Φορτώνουμε το εκτελέσιμο αρχείο arduino.ino στο arduino Uno.
Για την εκκίνηση του εκτελέσιμου αρχείου για το arduino δημιουργούμε launch file, το οποίο χρησιμοποιεί το πακέτο rosserial_python, είναι τύπου serial_node.py και συνδέεται στο port του arduino /dev/ttyACM0 με baud rate 57600. [56, 68]

Για την εκκίνηση του εκτελέσιμου αρχείου mover_gui.py χρησιμοποιούμε την εντολή rosrun <package> <executable> στο terminal.
Για την εκκίνηση του εκτελέσιμου αρχείου map_creator_gui.py χρησιμοποιούμε την εντολή rosrun <package> <executable> στο terminal.

11.3 Λεπτομέρειες υλοποίησης

CMakeList.txt:
Σε αυτό το αρχείο χρειάζεται να προσθέσουμε την εύρεση πακέτων βιβλιοθηκών όπως rospy, std_msgs, message_generation, να προσθέσουμε τα μηνύματα Distances.msg για να παραχθούν στο φάκελο 'msg', να προσθέσουμε τις εξαρτήσεις στο std_msgs και στο message_runtime.

package.xml:
Σε αυτό το αρχείο χρειάζεται να προσθέσουμε τα πιο κάτω tags:
<build_depend>message_generation</build_depend>
<exec_depend>message_runtime</exec_depend>
<buildtool_depend>catkin</buildtool_depend>
<build_depend>rospy</build_depend>
<build_depend>std_msgs</build_depend>
<build_export_depend>rospy</build_export_depend>
<build_export_depend>std_msgs</build_export_depend>
<exec_depend>rospy</exec_depend>
<exec_depend>std_msgs</exec_depend>

mover_gui.py:
Είναι υπεύθυνο για την παροχή εντολών για τη μετακίνηση του ρομπότ.
Δημιουργεί γραφικό περιβάλλον επικοινωνίας με 4 κουμπιά:
Move Forward, Move Backward, Turn Left, Turn Right.
Επίσης, αναθέτει τα αντίστοιχα 4 κουμπιά στο πληκτρολόγιο (τα βελάκια ↑ ↓ ← →).
Επομένως, η αντιστοιχία κουμπιών – κίνησης είναι:
Turn Left ή ← : Περιστροφή ρομπότ 90˚ αριστερόστροφα ως προς το κέντρο βάσης του
Turn Right ή → : Περιστροφή ρομπότ 90˚ δεξιόστροφα ως προς το κέντρο βάσης του

Move Forward ή ↑ : Μπροστινή μετακίνηση ρομπότ απόστασης μήκους ρομπότ
Move Backward ή ↓ : Οπίσθια μετακίνηση ρομπότ απόστασης μήκους ρομπότ
Παράλληλα, χρησιμοποιεί το ROS.
Δημιουργεί ένα κόμβο ROS με το όνομα mover_car.
Ο κόμβος αυτός βρίσκεται σε βρόγχο με ρυθμό 1 Hz, δηλαδή επαναλαμβάνει το βρόγχο 1 φορά ανά δευτερόλεπτο (όσο ο χρόνος επεξεργασίας μας δεν υπερβαίνει το 1 δευτερόλεπτο)
Δημιουργεί ένα publisher με το όνομα pub και εκδίδει μηνύματα στο topic mover για την επιθυμητή κίνησή του ρομπότ.
Το μέγεθος της ουράς εξερχομένων μηνυμάτων του pub, που χρησιμοποιείται για ασύγχρονη δημοσίευση είναι 10.
Ο τύπος του μηνύματος που αποστέλλει το pub είναι μια μεταβλητή char από το πακέτο std_msgs
Ανάλογα με την επιλεγόμενη κίνηση στέλνει τον αντίστοιχο χαρακτήρα.
‘d’ : Περιστροφή ρομπότ 90˚ αριστερόστροφα ως προς το κέντρο βάσης του
‘a’ : Περιστροφή ρομπότ 90˚ δεξιόστροφα ως προς το κέντρο βάσης του
‘w’ : Μπροστινή μετακίνηση ρομπότ απόστασης μήκους ρομπότ
‘x’ : Οπίσθια μετακίνηση ρομπότ απόστασης μήκους ρομπότ

arduino.ino:

Τον κώδικα αυτό το φορτώνουμε στην πλακέτα Arduino Uno.
Κύριες λειτουργίες αυτού είναι:

1. να ενημερώνεται για την εντολή κίνησης που πρέπει να εκτελέσει
2. να μετακινεί το ρομπότ, δηλαδή να δίνει εντολές για την μετακίνηση των τροχών
3. να ελέγχει για την ύπαρξη τοίχων
4. να ενημερώνει για την κίνηση που πραγματοποίησε
5. να ενημερώνει για την ύπαρξη τοίχων

Ο τρόπος με τον οποίο πραγματοποιεί τους πιο πάνω στόχους είναι ο εξής:
Για να μπορούν να επιτευχθούν οι στόχοι 1, 4, 5 χρησιμοποιεί το ROS.
Πιο συγκεκριμένα δημιουργεί ένα κόμβο ROS με το όνομα arduino_nh.

Επίτευξη στόχου 1: Να ενημερώνεται για την εντολή κίνησης που πρέπει να εκτελέσει

Δημιουργεί ένα subscriber με το όνομα sub_move ο οποίος κάνει subscribe στο topic mover. Από το topic mover λαμβάνει μηνύματα για το ποια κίνηση να εκτελέσει

Ο τύπος του μηνύματος που λαμβάνει το sub_move είναι μια μεταβλητή char από το πακέτο std_msgs

Ανάλογα με το χαρακτήρα που λαμβάνει, εκτελεί την αντίστοιχη κίνηση.

‘d’ : Περιστροφή ρομπότ 90˚ αριστερόστροφα ως προς το κέντρο βάσης του

‘a’ : Περιστροφή ρομπότ 90˚ δεξιόστροφα ως προς το κέντρο βάσης του

‘w’ : Μπροστινή μετακίνηση ρομπότ απόστασης μήκους ρομπότ

‘x’ : Οπίσθια μετακίνηση ρομπότ απόστασης μήκους ρομποτ

Επίτευξη στόχου 2: Να μετακινεί το ρομπότ

Για την εκτέλεση της κίνησης εφαρμόζει τα πιο κάτω:

Στην πλακέτα 2x2A DC Motor Shield ισχύει το standard PWM DC control:

Τα pin 4, 5, 6, 7 χρησιμοποιούνται για τον έλεγχο της ταχύτητας και της διεύθυνσης για τους δύο κινητήρες.

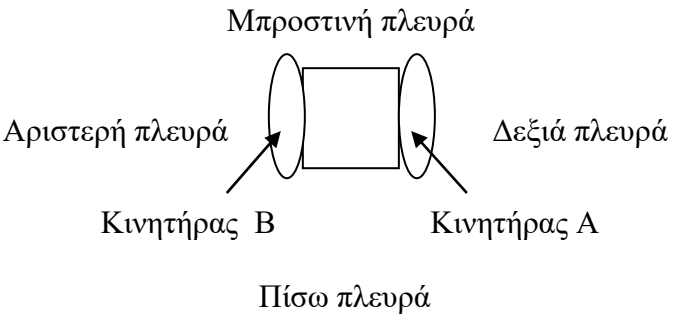
4: Έλεγχος διεύθυνσης για κινητήρα A (συμβολισμός M1)

5: Έλεγχος ταχύτητας για κινητήρα A (συμβολισμός E1)

6: Έλεγχος ταχύτητας για κινητήρα B (συμβολισμός E2)

7: Έλεγχος διεύθυνσης για κινητήρα B (συμβολισμός M2)

Αναπαράσταση τοποθεσίας για τους κινητήρες πάνω στο ρομπότ. Ο κινητήρας A ελέγχει τον τροχό στη δεξιά πλευρά του ρομπότ και ο κινητήρας B ελέγχει τον τροχό στην αριστερή πλευρά του ρομπότ.



Ανάλογα με τις τιμές που εφαρμόζονται στα pins αυτά, εκτελούνται τα παρακάτω.

M1 = LOW – ο κινητήρας γυρίζει αριστερόστροφα με αποτέλεσμα ο τροχός στη δεξιά πλευρά του ρομπότ να κινείται προς τα πίσω
M1 = HIGH - ο κινητήρας γυρίζει δεξιόστροφα με αποτέλεσμα ο τροχός στη δεξιά πλευρά του ρομπότ να κινείται προς τα μπροστά
M2 = LOW - ο κινητήρας γυρίζει αριστερόστροφα με αποτέλεσμα ο τροχός στην αριστερή πλευρά του ρομπότ να κινείται προς τα μπροστά
M2 = HIGH - ο κινητήρας γυρίζει δεξιόστροφα με αποτέλεσμα ο τροχός στην αριστερή πλευρά του ρομπότ να κινείται προς τα πίσω
E1 = E2 = 0-255 - Ανάλογα με την τιμή που θα επιλέξουμε, εξαρτάται το πόσο γρήγορα θα κινείται το ρομπότ (στην παρούσα εργασία έγινε επιλογή του 255 δίνοντας τη μέγιστη ταχύτητα)

Χρησιμοποιείται επίσης η εντολή delay(ms) η οποία βάζει σε παύση το πρόγραμμα για το χρονικό διάστημα (χιλιοστά του δευτερολέπτου) που καθορίζεται ως παράμετρος. Σκοπός της, λοιπόν, για τη λειτουργία του προγράμματός αυτού είναι να επιτρέπει την κάθε κίνηση για συγκεκριμένο χρονικό διάστημα ώστε το ρομπότ να μπορεί να περιστραφεί ως προς το κέντρο της βάσης του 90˚ ή να προχωρήσει για 22.5 εκ. [58]

Με βάση τα πιο πάνω οι επιτρεπτές κινήσεις του ρομπότ υλοποιούνται ως εξής:
Περιστροφή ρομπότ 90˚ αριστερόστροφα ως προς το κέντρο βάσης του:
E1 = E2 = 255 / M1 = HIGH / M2 = HIGH
delay(850)
E1 = E2 = 0

Περιστροφή ρομπότ 90˚ δεξιόστροφα ως προς το κέντρο βάσης του:
E1 = E2 = 255 / M1 = LOW / M2 = LOW
delay(850)
E1 = E2 = 0

Μπροστινή μετακίνηση ρομπότ για 22.5 εκ.:
E1 = E2 = 255 / M1 = HIGH / M2 = LOW
delay(1100)
E1 = E2 = 0

Οπίσθια μετακίνηση ρομπότ για 22.5 εκ.:
E1 = E2 = 255 / M1 = LOW / M2 = HIGH
delay(1100)
E1 = E2 = 0

Επίτευξη στόχου 3: Να ελέγχει για την ύπαρξη τοίχων

Για την επίτευξη αυτού του στόχου, όπως έχει προαναφερθεί το ρομπότ διαθέτει 4 αισθητήρες υπερήχων τύπου HY-SRF05.
Τα pin 9, 2, A0, A3 χρησιμοποιούνται ως trigger pins, οπόταν τα ορίζουμε ως output pins και θέτουμε την τιμή τους σε LOW.
Τα pin 10, 3, A1, A4 χρησιμοποιούνται ως echo pins, οπόταν τα ορίζουμε ως input pins.
Τα trigger pins και τα echo pins μπαίνουν σε ομάδες 9 - 10, 2 - 3, A0 - A1, A3 - A4
Για κάθε ένα από τους 4 αισθητήρες υπερήχων αντιστοιχεί μια από τις 4 ομάδες trigger pin - echo pin.
Για να λάβουμε την απόσταση από κάποιο αισθητήρα υπερήχων:
Θέτουμε την τιμή του trigger pin σε HIGH.
Περιμένουμε 10 μικροδευτερόλεπτα (microseconds) ώστε να σταλεί το σήμα.
Θέτουμε την τιμή του trigger pin σε LOW.
Διαβάζουμε την τιμή που λαμβάνει το echo pin σε περίπτωση που είναι HIGH.
Αυτή η τιμή μας δίνει τη διάρκεια σε μικροδευτερόλεπτα που έκανε το σήμα να πάει από το trigger pin, να χτυπήσει πάνω στον τοίχο (αν και εφόσον υπάρχει), να αντανακλαστεί και να φτάσει πίσω στο echo pin. Οπόταν για να υπολογίσουμε την απόσταση διαιρούμε την τιμή αυτή δια δύο γιατί το ζητούμενο μας είναι η διάρκεια για να ταξιδέψει το σήμα από τον αισθητήρα υπερήχων μέχρι τον τοίχο, χωρίς να περιλαμβάνεται η επιστροφή. Τέλος, για να μετατρέψουμε τη διάρκεια σε εκατοστόμετρα διαιρούμε δια 29. Αυτό προκύπτει από το ότι η ταχύτητα του ήχου είναι 340 m/s, το οποίο μας δίνει ~29 μικροδευτερόλεπτα ανά εκατοστόμετρο, οπόταν προσεγγιστικά διαιρούμε με 29.
Αν η απόσταση που λαμβάνουμε είναι μικρότερη από 10 cm τότε σημαίνει πώς υπάρχει τοίχος.
Επομένως, ανάλογα με την πλευρά του ρομπότ που βρίσκεται ο κάθε αισθητήρας και τη θέση του ρομπότ, συνεπάγεται ότι μπορούμε να γνωρίζουμε που ακριβώς βρίσκεται ο τοίχος.

Επίτευξη στόχου 4 και 5: Να να ενημερώνει για την κίνηση που πραγματοποίησε και για την ύπαρξη τοίχων

Για τους 2 αυτούς στόχους χρησιμοποιεί λειτουργίες από το ROS.

Δημιουργεί ένα publisher με το όνομα pub_distances ο οποίος κάνει publish στο topic pub_distances.

Πιο συγκεκριμένα αποστέλλει μήνυμα με την κίνηση που έχει εκτελέσει και τις αποστάσεις που καταγράφει από τους αισθητήρες υπερήχων.

Ο τύπος του μηνύματος που αποστέλλει είναι προσαρμοσμένο (custom) μήνυμα distances το οποίο περιλαμβάνει μια μεταβλητή char η οποία αντιπροσωπεύει την κίνηση που έγινε και ένα πίνακα 4 θέσεων τύπου uint16 ο οποίος αντιπροσωπεύει τις 4 αποστάσεις από τους αισθητήρες υπερήχων.

map_creator_gui.py:

Είναι υπεύθυνο για τη δημιουργία του δυσδιάστατου χάρτη.

Ζητάει από το χρήστη να δώσει το πλάτος και το μήκος της αρένας. Οι αποστάσεις αυτές μετριούνται σε αριθμό φορών που το πλάτος του ρομπότ χωράει στο πλάτος της αρένας και σε αριθμό φορών που το μήκος του ρομπότ χωράει στο μήκος της αρένας.

Ζητάει επίσης την αρχική θέση του ρομπότ στην αρένα και σε ποια κατεύθυνση βρίσκεται η πρόσοψή του ρομπότ.

Χρησιμοποιούνται οι εξής δομές δεδομένων:

- Για την καταγραφή της θέσης του ρομπότ:

  startX: αρχική θέση ρομπότ x
  srtartY: αρχική θέση ρομπότ y
  currentX: θέση ρομπότ x
  currentY: θέση ρομπότ y
- Για τη καταγραφή της διεύθυνσης του ρομπότ:

  direction: διεύθυνση ρομπότ (0: Πάνω, 1: Δεξιά, 2: Κάτω, 3: Αριστερά)
- Για την καταγραφή του χάρτη:

  w: πλάτος αρένας
  h: μήκος αρένας

horizontalWall: Σταθερός πίνακας (h+1) γραμμών και (w) στηλών που αντιπροσωπεύει τους οριζόντιους τοίχους
verticalWall: Σταθερός πίνακας (h) γραμμών και (w+1) στηλών που αντιπροσωπεύει τους κάθετους τοίχους

- Για την αναπαράστασή του χάρτη
  Canvas μεγέθους ανάλογου του μεγέθους της αρένας

Αρχικά παρουσιάζουμε ένα κενό καμβά και ένα κενό διάγραμμα, αφού δεν γνωρίζουμε καμία πληροφορία για την αρένα μας. Τοποθετούμε σε αυτά μόνο που βρίσκεται το ρομπότ (αρχική θέση) και ποια είναι η κατεύθυνση του.

Γίνεται χρήση του ROS.
Δημιουργεί ένα κόμβο ROS με το όνομα map_creator.
Δημιουργεί ένα publisher ο οποίος κάνει subscribe στο topic pub_distances για να ενημερώνεται για τις κινήσεις του ρομπότ ώστε να μπορεί να ανανεώνει τη θέση/στάση του ρομπότ, όπως επίσης να ενημερώνεται και για τις αποστάσεις από τους αισθητήρες υπερήχων ώστε να μπορεί να ανανεώνει το χάρτη με την εύρεση τοίχων.
Ο τύπος του μηνύματος που λαμβάνει είναι προσαρμοσμένο (custom) μήνυμα distances το οποίο περιλαμβάνει μια μεταβλητή char η οποία αντιπροσωπεύει την κίνηση που έγινε και ένα πίνακα 4 θέσεων τύπου uint16 ο οποίος αντιπροσωπεύει τις 4 αποστάσεις από τους αισθητήρες υπερήχων.
Με την παραλαβή κάποιου μηνύματος,

1. Αν αφορά περιστροφή ενημερώνει μόνο την κατεύθυνσή του ρομπότ
2. Αν αφορά κίνηση μπροστά ή πίσω, ενημερώνει τη θέση του ρομπότ και ελέγχει για την ύπαρξη τοίχων, δηλαδή ελέγχει αν κάποια από τις αποστάσεις που λαμβάνει είναι μικρότερη του 10. Ανάλογα τότε με τη νέα θέση του ρομπότ, την κατεύθυνση και σε ποια πλευρά του ρομπότ βρίσκεται ο αισθητήρας υπερήχων που δίνει τιμή μικρότερη του 10, υπολογίζει που βρίσκεται ακριβώς ο τοίχος στο χάρτη.

Έπειτα, παρουσιάζει το χάρτη με τις δύο διαφορετικές αναπαραστάσεις.
Για την γραφική αναπαράσταση: Σε περίπτωση που εντοπίσει τοίχο, ζωγραφίζει γραμμή στην αντίστοιχη θέση στον καμβά

Για την διαγραμματική αναπαράσταση: Σε περίπτωση που εντοπίσει τοίχο, ενημερώνει τους πίνακες horizontalWall και verticalWall ανάλογα αν ήταν οριζόντιος ή κάθετος τοίχος. Και μετά τους διαβάζει για να τυπώσει τη νέα διαγραμματική αναπαράσταση του χάρτη.
Κάθε φορά που παρουσιάζει το χάρτη, παρουσιάζει και την ανανεωμένη θέση και κατεύθυνση του ρομπότ.

Κεφάλαιο 12

Σύνοψη, Συμπεράσματα, Μελλοντική εξέλιξη

Στη διπλωματική εργασία αυτή υλοποιήθηκε η τηλεκατεύθυνση ενός ρομποτικού οχήματος εδάφους με σκοπό την εξερεύνηση μιας περιοχής και τη δημιουργία χάρτη της περιοχής αυτής.

Περιορισμοί που τέθηκαν και προβλήματα που εντοπίστηκαν μπορούν να αποτελέσουν το έναυσμα για μελλοντική εξέλιξη της δουλειάς αυτής.
Περιορισμός/Πρόβλημα/Ευκαιρία – Μελλοντική εξέλιξη:

	Περιορισμός/Πρόβλημα/Ευκαιρία	**Μελλοντική Εξέλιξη**
1.	Χρήση φτηνού ρομποτικού εξοπλισμού, ανωμαλίες στην προκαθορισμένη κίνησή των ρομπότ.	Χρήση πιο αξιόπιστου εξοπλισμού
2.	Πρόβλημα ακριβή εντοπισμού θέσης ρομπότ και μέτρησης της ταχύτητας στην πάροδό του χρόνου.	Χρήση επιπλέον τεχνολογικού εξοπλισμού όπως rotary encoder ή gps.
3.	Περιορισμοί στα χαρακτηριστικά της περιοχής εξερεύνησης (χρήση αρένας με διαδρόμους και τοίχους)	Εξερεύνηση οποιασδήποτε περιοχής.
4.	Εντοπισμός τοίχων	Εντοπισμός οποιονδήποτε αντικειμένων
5.	Χάρτης περιοχής σε συμβολική μορφή.	Χρήση του χάρτη για ασφαλή πλοήγηση ρομπότ μέσα στην αρένα. Με τη γνώση της τοποθεσίας των τοίχων στον χάρτη, μπορεί να αποφευχθεί η πρόσκρουση του ρομπότ.
6.	Χάρτης περιοχής σε συμβολική μορφή.	Χρήση του χάρτη για υπολογισμό βέλτιστων μονοπατιών για την μετακίνηση του ρομπότ από κάποιο σημείο σε κάποιο άλλο σημείο, αποφεύγοντας την πρόσκρουση σε τοίχους.

Βιβλιογραφία

[1] (). . Available: https://wiki.dfrobot.com/MiniQ_Discovery_Kit_SKU_KIT0071#target_4.

[2] (). . Available: https://wiki.dfrobot.com/Devastator_Tank_Mobile_Platform_SKU_ROB0112.

[3] (). . Available: https://www.dfrobot.com/product-69.html.

[4] Anonymous *DEVASTATOR Tank Mobile Platform Instruction Manual.*

[5] (). . Available: https://www.raspberrypi.org/.

[6] (). . Available: https://www.raspberrypi.org/products/raspberry-pi-2-model-b/.

[7] (). . Available: https://www.arduino.cc/en/Main/AboutUs.

[8] (). . Available: https://store.arduino.cc/usa/arduino-uno-rev3.

[9] (). . Available: https://create.arduino.cc/projecthub/Nicholas_N/distance-measurement-with-an-ultrasonic-sensor-hy-srf05-64554e.

[10] (). . Available: https://www.arduino.cc/en/Reference/SoftwareSerial.

[11] (). . Available: https://store.arduino.cc/usa/usb-2-0-cable-type-a-b?fbclid=IwAR0R9I3Z79AySwGzC8bqvaSv7rZPXMBY_T78PifQVJbREWzqnXG5ihQSXSU.

[12] (). . Available: https://www.raspberrypi.org/documentation/hardware/raspberrypi/power/README.md
.

[13] (). . Available: https://www.debian.org/intro/about#what.

[14] (). . Available: https://ubuntu-mate.org/about/.

[15] (). . Available: https://ubuntu-mate.org/what-is-ubuntu-mate/.

[16] (). . Available: https://www.raspberrypi.org/downloads/.

[17] (). . Available: https://www.arduino.cc/en/Main/Software.

[18] (). . Available: https://ubuntu-mate.org/raspberry-pi/.

[19] (2018/01/20). . Available: https://www.arduino.cc/en/tutorial/arduinoISP.

[20] (2018-11-29). . Available: http://wiki.ros.org/Documentation.

[21] (). . Available: http://www.ros.org/about-ros/.

[22] (2018-01-08). . Available: http://wiki.ros.org/kinetic.

[23] (2017-10-02). . Available: http://wiki.ros.org/tf.

[24] (2017-07-07). . Available: http://wiki.ros.org/catkin/workspaces#Catkin_Workspaces.

[25] (2019-04-14). . Available: http://wiki.ros.org/Packages.

[26] (2018-11-09). . Available: http://wiki.ros.org/catkin/package.xml.

[27] (2018-06-13). . Available: http://wiki.ros.org/catkin/CMakeLists.txt.

[28] (2017-05-22). . Available: http://wiki.ros.org/catkin/commands/catkin_make.

[29] (2018-12-04). . Available: http://wiki.ros.org/Nodes.

[30] (2016-08-26). . Available: http://wiki.ros.org/Messages.

[31] (2019-02-20). . Available: http://wiki.ros.org/Topics.

[32] (2018-01-15). . Available: http://wiki.ros.org/Master.

[33] (2016-11-25). . Available: http://wiki.ros.org/roscore.

[34] (2018-11-08). . Available: http://wiki.ros.org/Parameter%20Server.

[35] (2018-06-11). . Available: http://wiki.ros.org/rosout.

[36] (2019-05-08). . Available: http://wiki.ros.org/rosbash#rosrun.

[37] (). . Available: https://www.sdcard.org/developers/overview/capacity/.

[38] (). . Available: https://www.raspberrypi.org/documentation/installation/sdxc_formatting.md?fbclid=IwAR0YP4yrHzEhyRNYKQVKbTngmYa5pgAOFye7QWShGrhJthXT0bkbPn45-FU.

[39] (). . Available: https://www.easeus.com/ppc/partition-manager/partition-manager.html?gclid=EAIaIQobChMInvecn9nb4QIVU-d3Ch1qEgvXEAAYASAAEgLD6PD_BwE.

[40] (). . Available: https://ubuntu-mate.org/blog/ubuntu-mate-xenial-point-2-raspberry-pi/.

[41] (). . Available: https://www.arduino.cc/en/guide/linux.

[42] (2017-07-25). . Available: http://wiki.ros.org/kinetic/Installation/Ubuntu.

[43] (2017-05-11). . Available: http://wiki.ros.org/catkin/Tutorials/create_a_workspace.

[44] (2013-05-30). . Available: http://wiki.ros.org/ROS/Tutorials/CreatingPackage.

[45] (2012-12-24). . Available: http://wiki.ros.org/ROS/Tutorials/BuildingPackages.

[46] (2018-06-02). . Available: http://wiki.ros.org/ROS/Tutorials/UnderstandingNodes.

[47] (2018-12-06). . Available: http://wiki.ros.org/ROS/Tutorials/UnderstandingTopics.

[48] (2017-08-01). . Available: http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber%28python%29.

[49] (2017-05-22). . Available: http://wiki.ros.org/ROS/Tutorials/ExaminingPublisherSubscriber.

[50] (2019-04-28). . Available: http://wiki.ros.org/rosserial_arduino/Tutorials/Hello%20World.

[51] (2018-11-09). . Available: http://wiki.ros.org/ROS/Tutorials/CreatingMsgAndSrv.

[52] (2014-06-25). . Available: http://wiki.ros.org/ROS/Tutorials/CustomMessagePublisherSubscriber%28python%29.

[53] (2018-04-30). . Available: http://wiki.ros.org/rosserial/Tutorials/Adding%20Other%20Messages.

[54] (2019-03-19). . Available: http://wiki.ros.org/rosserial_arduino/Tutorials/Arduino%20IDE%20Setup.

[55] (2014-03-27). . Available: http://wiki.ros.org/rosserial_arduino/Tutorials/Adding%20Custom%20Messages.

[56] (2015-10-07). . Available: http://wiki.ros.org/rosserial_python.

[57] (2018-01-06). . Available: http://wiki.ros.org/ROS/Tutorials/UsingRqtconsoleRoslaunch.

[58] (). . Available: https://www.arduino.cc/reference/en/language/functions/time/delay/.

[59] (). . Available: https://www.dfrobot.com/index.php?route=information/information&information_id=4.

[60] (). . Available: https://www.dfrobot.com/product-1144.html.

[61] (2018-09-13). . Available: http://wiki.ros.org/ROSberryPi/Installing%20ROS%20Kinetic%20on%20the%20Raspberry%20Pi.

[62] (). . Available: https://www.arduino.cc/en/Tutorial/PWM.

[63] X. Nong *et al*, "Research on indoor navigation of mobile robot based on INS and ultrosound," in *2017 12th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, 2017, .

[64] J. M. O'Kane, *A Gentle Introduction to ROS.Independently Published, October 2013*, .

[65] Omara, Hesham Ibrahim Mohamed Ahmed and K. S. M. Sahari, "Indoor mapping using kinect and ROS," in *2015 International Symposium on Agents, Multi-Agent Systems and Robotics (ISAMSR)*, 2015, .

[66] L. Peskoe-Yang, "**Paris Firefighters Used This Remote-Controlled Robot to Extinguish the Notre Dame Blaze**," 22 Apr 2019, .

[67] (2018-10-01). . Available: http://wiki.ros.org/rosserial.

[68] (2018-10-01). . Available: http://wiki.ros.org/rosserial_arduino.

[69] X. Wu, M. Abrahantes and M. Edgington, "MUSSE: A designed multi-ultrasonic-sensor system for echolocation on multiple robots," in *2016 Asia-Pacific Conference on Intelligent Robot Systems (ACIRS)*, 2016, .