ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Robot using Mindstorms NXT

Μάιος 2011

Ατομική Διπλωματική Εργασία

Robot using Mindstorms NXT

Κωνσταντίνος Χριστοφόρου

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάιος 2011

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Robot using Mindstorms NXT

Κωνσταντίνος Χριστοφόρου

Επιβλέπων Καθηγητής

Παρασκευάς Ευριπίδου

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

Μάιος 2011

Ευχαριστίες

Αρχικά, θα ήθελα να ευχαριστήσω θερμά τον επιβλέπων καθηγητή μου Κύριο Παρασκευά Ευριπίδου για την πολύτιμη βοήθεια, την συμβουλευτική καθοδήγηση, τις εισηγήσεις και τα μέσα που μου παρείχε για την εκπόνηση της Ατομικής Διπλωματικής μου Εργασίας.

Τέλος, θα ήθελα να ευχαριστήσω την οικογένεια και τους φίλους μου για την ηθική στήριξη που μου παρείχαν για την εκπόνηση αυτής της διπλωματικής εργασίας καθώς και καθ' όλη την διάρκεια των σπουδών μου.

Περίληψη

Ο σκοπός της Ατομικής Διπλωματικής μου Εργασίας είναι ο υλοποίηση μιας ρομποτικής κατασκευής χρησιμοποιώντας το Lego Mindstorms NXT σετ (ένα προγραμματιζόμενο ρομποτικό σετ) που θα έχει την δυνατότητα να εκτελεί κάποιες συμπεριφορές όπως και να ελέγχετε από ένα τηλεχειριστήριο.
Υλικό που χρησιμοποιήθηκε για ρομποτική κατασκευή:

- Ένα Lego Mindstorms NXT εκπαιδευτικό σετ.
- Ένα επιπλέων Nxt brick (ενσωματωμένο συστήματα με δυνατότητα να ελέγχει τρείς αισθητήρες και τρία μοτέρ).
- Αισθητήρες (Compass ,Gyro, Infrared, Ultrasonic, Camera sensor).
- Ένα κινητό τηλέφωνο με λειτουργικό Android που χρησιμοποιείται σαν τηλεχειριστήριο του ρομπότ.

Η γλώσσα προγραμματισμού χρησιμοποιήθηκε είναι η leJOS. Στα δύο NXT bricks εγκαταστάθηκε το leJOS NXJ που είναι ένα Java Virtual Machine και επιτρέπει στα NXT bricks να προγραμματιστούν με την γλώσσα προγραμματισμού Java. Η πλατφόρμα leJOS είναι η κατάλληλη επιλογή για αυτό το ερευνητικό έργο αφού μας δίνει την δυνατότητα μέσο βιβλιοθηκών που παρέχουν διάφορες ψηλές επιπέδου διεργασίες όπως η πλοήγηση (robot navigation) και behavior-based robotics να απλουστεύσουν την πλοήγηση και έλεγχο του ρομπότ.

Συμπεριφορές της ρομποτικής κατασκευής που υλοποιήθηκαν :

- Ακολουθία γραμμής στο πάτωμα.
- Ακολουθία αντικειμένου με συγκεκριμένο χρώμα.
- Αποφυγή αντικειμένων όταν το ρομπότ κατευθύνετε σε συγκεκριμένο προορισμό.

Αποφυγή αντικειμένων με το ρομπότ να ακλουθεί τυχαία πορεία

Περιεχόμενα

Ευχαριστίες **1**

Περίληψη **2**

Κεφάλαιο 1 Εισαγωγή **4**

1.1 Ιστορία Ρομποτικής 4
1.2 Εφαρμογές Ρομποτικής 8
1.2.1 Βιομηχανικά Ρομπότ 8
1.2.2 Εγχώρια ή Οικιακά Ρομπότ 8
1.2.3 Ιατρικά ρομπότ 9
1.2.4 Στρατιωτικά Ρομπότ 9
1.2.5 Ψυχαγωγικά Ρομπότ 10
1.2.6 Εξερευνητικά ρομπότ 10

Κεφάλαιο 2 Τεχνικές Εντοπισμού (Robot Localization techniques) **11**

2.1 Έννοια Εντοπισμού Ρομπότ 11
2.2 Σχετικός ή Τοπικός Εντοπισμός (local) 11
2.3 Απόλυτος ή παγκόσμιος εντοπισμός 12
2.4 Πιθανολογικός Εντοπισμός (Probabilistic localization) 12
2.5 Τεχνικές Χαρτογράφησης(Mapping) 12

Κεφάλαιο 3 Lego Mindstorms NXT **14**

3.1 Κύρια χαρακτηριστικά Nxt brick [9] 14
3.2 Αισθητήρες Nxt από Mindstorm 15
3.3 Αισθητήρες Nxt από άλλους Κατασκευαστές 17
3.4 Προγραμματίστηκες γλώσσες Nxt 20
3.5 LeJOS NXJ 21

Κεφάλαιο 4 Αρχιτεκτονική Ρομπότ **23**

4.1 Υλικό που Χρησιμοποιήθηκε 23
4.2 Επικοινωνία Μεταξύ Συστατικών Υλικού 24
4.3 Σχεδίαση και Συναρμολόγηση του Ρομπότ 26
4.4 Πλοήγηση Ρομπότ 28
4.5 Φωτογραφίες ρομπότ 32

Κεφάλαιο 5 Λειτουργίες και Συμπεριφορές Ρομπότ που Υλοποιήθηκαν **33**

5.1 Έννοια Pid Contoller 33
5.2 Ακολουθία Γραμμής στο πάτωμα 34
5.3 Ακολουθία αντικειμένου με συγκεκριμένο χρώμα 37
5.4 Αποφυγή αντικειμένων όταν το ρομπότ να ακολουθεί τυχαία πορεία 38
5.5 Αποφυγή αντικειμένων όταν το ρομπότ κατευθύνεται σε συγκεκριμένο προορισμό 40
5.6 Απομακρυσμένος έλεγχος ρομπότ από κινητό τηλέφωνο 43

Κεφάλαιο 6 Γενικά Συμπεράσματα **45**

Βιβλιογραφία **46**

Κεφάλαιο 1

Εισαγωγή

1.1 Ιστορία Ρομποτικής 4
1.2 Εφαρμογές Ρομποτικής 8
1.2.1 Βιομηχανικά Ρομπότ 8
1.2.2 Εγχώρια ή Οικιακά Ρομπότ 8
1.2.3 Ιατρικά ρομπότ 9
1.2.4 Στρατιωτικά Ρομπότ 9
1.2.5 Ψυχαγωγικά Ρομπότ 10
1.2.6 Εξερευνητικά ρομπότ 10

1.1 Ιστορία Ρομποτικής

Η ιστορία της ρομποτικής είναι συνδεδεμένη με την ιστορία της τεχνολογίας, της επιστήμης και της βασικής αρχής της προόδου. Η τεχνολογία που χρησιμοποιείται στην πληροφορική, η ηλεκτρική ενέργεια, ακόμα και η μηχανική των αερίων και τα υδραυλικά συστήματα μπορούν να θεωρηθούν ως μέρος της ιστορίας της ρομποτικής. Η ρομποτική αποτελεί σήμερα ένα από τα μεγαλύτερα επιτεύγματα της ανθρωπότητας και αποτελεί την ενιαία μεγαλύτερη προσπάθεια της ανθρωπότητας να παράγουν ένα τεχνητό, αισθανόμενο ον. Ο στόχος αυτού του χρονοδιαγράμματος είναι να προσφέρει στον αναγνώστη μια γενική επισκόπηση στον τομέα της ρομποτικής (με έμφαση περισσότερο στα κινητά ρομπότ) και να δοθεί εκτίμηση στους εφευρέτες και καινοτόμους στον τομέα αυτό που βοήθησαν τη ρομποτική για να γίνει αυτό που είναι σήμερα[1].

- 77-100BC

  Το 1901, ανάμεσα στα νησιά της Κρήτης και Κυθήρων, ένας δύτης βρήκε τα απομεινάρια του τι θα μπορούσε να θεωρηθεί μηχανικός υπολογιστής. Η συσκευή είναι ένας περίπλοκος μηχανισμός από γρανάζια τα οποία κατά πάσα πιθανότητα υπολογίζουν την θέση του ήλιου, του φεγγαριού ή άλλων ουράνιων σώματα.

- 10-70AD

  Ο ήρωας της Αλεξάνδρειας, ένας μαθηματικός, φυσικός και μηχανικός μεταξύ πολλών

εφευρέσεων, σχεδίασε ένα οδόμετρο που τοποθετιόταν σε κάρο και μπορούσε να μετρήσει την απόσταση που το κάρο ταξίδευσε.

- 1645

  Ο Blaise Pascal επινόησε μια υπολογιστική μηχανή για να βοηθήσει τον πατέρα του με την μέτρηση των φόρων. Η συσκευή ονομάστηκε Pascaline και περίπου50 Pascalines κατασκευάστηκαν.

- 1709

  Η πιο διάσημη δημιουργία του Jacques de Vaucanson ήταν "Η πάπια". Αυτό το μηχανικό σύστημα μπορούσε να κουνήσει φτερά του, να τρώει και να χωνέψει σιτάρι. Κάθε πτερύγιο του περιέχει πάνω από τετρακόσια κινούμενα μέρη και ακόμη και σήμερα παραμένει μυστήριο. Η αρχική πάπια έχει εξαφανιστεί.

- 1941

  Ο συγγραφέας βιβλίων επιστημονικής φαντασίας Isaac Asimov χρησιμοποίησε για πρώτη φορά τη λέξη "ρομποτική" για να περιγράψει την τεχνολογία των ρομπότ και προέβλεψε την άνοδος μιας ισχυρής βιομηχανίας ρομπότ. Ο όρος ρομποτική αναφέρεται στη μελέτη και τη χρήση των ρομπότ. Ήρθε περίπου το1941και εγκρίθηκε για πρώτη φορά από τον Isaac Asimov. Επίσης ο Asimov ήταν αυτός που πρότεινε τους νόμους της : «Νόμοι της Ρομποτικής».

- 1947

  Στις 14 Νοεμβρίου του 1947, Walter Brattain είχε ένα ατύχημα κατά την προσπάθεια μελετήσει πώς τα ηλεκτρόνια δρουν στην επιφάνεια ενός ημιαγωγού. Αυτό το ατύχημα επέφερε τη δημιουργία του πρώτου τρανζίστορ.

- 1948

  W. Grey Walter δημιούργησε το πρώτο ρομπότ του Elmer και Elsie, επίσης γνωστό ως ρομπότ χελώνα, ήταν σε θέση να βρουν σταθμό φόρτισης τους όταν η ισχύος της μπαταρίας ήταν σε χαμηλά επίπεδα.

- 1951

  Raymond Goertz σχεδίασε τον πρώτο αρθρωτό βραχίονα για την Επιτροπή Ατομικής Ενέργειας.

- 1954

  George Devol σχεδίασε το πρώτο πραγματικά προγραμματιζόμενα ρομπότ και το ονόμασε Unimate για "Universal Automation ". Αργότερα, το 1956, ο George Devol και Joseph Engelberger διαμόρφωσε την πρώτη εταιρία ρομπότ στον κόσμο και τιην ονόμασαν "Unimation.

  Έτσι, ο Engelberger έχει χαρακτηριστεί ο «πατέρας της ρομποτικής». Η Unimation εξακολουθεί να είναι σε παραγωγή σήμερα, με ρομπότ για πώληση.

- 1964

  Τεχνητά ερευνητικά εργαστήρια νοημοσύνης άνοιξαν στο MIT, το StanfordResearch Institute (SRI), το Πανεπιστήμιο του Στάνφορντ και το Πανεπιστημίου τουEdinburg.

- 1968

  SRI κατασκεύασε το "Shakey" ένα κινητό ρομπότ εξοπλισμένο με σύστημα όρασης και ελεγχόμενο από έναν υπολογιστή που είχε το μέγεθος ενός δωματίου.

- 1973

  V.S. Gurfinkel, A.SHNEIDER, V. Gurfinkel και οι συνεργάτες του στο Τμήμα του ελέγχου της κίνησης της Ρωσικής Ακαδημίας Επιστημών δημιούργησαν τα πρώτα με έξι πόδια οχήματα.

- 1973

  Ichiro Kato δημιουργήθηκε το WABOT που ήταν το πρώτο ανθρωπόμορφο ρομπότ μεγάλης κλίμακας στον κόσμο. Είχε σύστημα για τον έλεγχο των άκρων, το ορατότητας, και συνομιλία. Εκτιμάται ότι είχε την πνευματική ικανότητα ενός παιδιού 18 μηνός.

- 1975

  Η MITS ALTAIR δημιούργησε το πρώτο 8080 τσιπ υπολογιστή που ήταναναμφισβήτητα η έναρξη του προσωπικού υπολογιστή.

- 1985

  Από τη General Robotics Corp δημιουργήθηκε το RB5X ένα προγραμματιζόμενο ρομπότ εξοπλισμένο με υπέρυθρους αισθητήρες, απομακρυσμένη μετάδοση βίντεο/ήχου και ένα συνθεσάιζερ φωνής. Είχε λογισμικό που θα του επέτρεπαν να μάθει για το περιβάλλον του.

- 1990

  Η iRobot εταιρία, η οποία ιδρύθηκε από τους Rodney Brooks, Colin Angle και HelenGreiner, έκανε παραγωγή στρατιωτικών ρομπότ.

- 1996-1997

  Η Honda δημιούργησε το P2, που ήταν το πρώτο σημαντικό βήμα για τη δημιουργία του ASIMO. Η P2 ήταν το πρώτο αυτορυθμιζόμενο, δίποδο ανθρωποειδές ρομπότ. Το 1997 δημιούργησε το P3, το δεύτερο μεγάλο βήμα για τη δημιουργία ASIMO. Το P3 ήταν το πρώτο εντελώς αυτόνομο ανθρωποειδές ρομπότ της Honda.

- 1998-1999

  Κατασκευάστηκαν παιχνίδια ρομπότ όπως LEGO MINDSTORMS Nxt, καθώς και το ρομπότ σκύλος της Sony .

- 2000

  Η Sony παρουσίασε το Sony Dream Robots (SDR). Το SDR ήταν σε θέση νααναγνωρίζει 10 διαφορετικά πρόσωπα, να εκφράζει συναισθήματα μέσα από την ομιλία και το σώμα, και να μπορεί να περπατήσει σε επίπεδες και ανώμαλες επιφάνειες.

- 2002

  Η Honda δημιούργησε την επόμενη εκδοση ASIMO, η οποία μπορούσε να αναγνωρίσει το πρόσωπο του ιδιοκτήτη του, τη φωνή, και όνομα του. Μπορούσε να διαβάσει e-mail και να είναι σε θέση να κάνει streaming βίντεο από την κάμερα του σε ένα Η / Υ.

- 2003

  Στο πλαίσιο της αποστολής τους να διερευνήσουν τον Άρη, η ΝΑΣΑ εκτόξευσε δύο ρομποτικά ρόβερ.

2005
Στο Cornell University δημιουργήθηκε αυτό-αναπαραγόμενο ρομπότ

1.2 Εφαρμογές Ρομποτικής

Σήμερα, τα ρομπότ εκτελούν πολλά διαφορετικά καθήκοντα σε διάφορους τομείς, όπου μέρα με την μέρα ο αριθμός εργασιών που ανατίθεται σε κάθε ρομπότ αυξάνεται σταθερά. Γι 'αυτό ένας από τους καλύτερους τρόπους για κατανομή των ρομπότ σε κατηγορίες είναι ανάλογα με την εφαρμογή τους

1.2.1 Βιομηχανικά Ρομπότ

Ένα βιομηχανικό ρομπότ ορίζεται από το ISO, ως ένα αυτόματα ελεγχόμενο σύστημα πολλαπλών βραχιόνων που είναι προγραμματιζόμενοι σε τρεις ή περισσότερους άξονες. Το πεδίο της ρομποτικής μπορεί να είναι πιο πρακτικό όταν ορίζεται ως η μελέτη, ο σχεδιασμός και η χρήση των συστημάτων ρομπότ για την κατασκευή Τυπικές εφαρμογές των ρομπότ περιλαμβάνουν την συγκόλληση, τον σχεδιασμό, τη συναρμολόγηση, την επιλογή και τοποθέτηση, τη συσκευασία και παλετοποίησης, την επιθεώρηση και τον έλεγχο, και όλα αυτά με μεγάλη αντοχή, ταχύτητα και ακρίβεια.

Τα πιο συχνά χρησιμοποιημένα διαμορφωμένα ρομπότ είναι τα αρθρωτά ρομπότ, τα SCARA ρομπότ και το Cartesian coordinate ρομπότ, (γνωστό και ως ρομπότ xyz). Τα ρομπότ εμφανίζουν διαφορετικό βαθμό αυτονομίας, όπου μερικά ρομπότ είναι προγραμματισμένα να εκτελούν πιστά συγκεκριμένες ενέργειες, ξανά και ξανά (επαναλαμβανόμενες πράξεις) χωρίς μεταβολή και με υψηλό βαθμό ακρίβειας. Οι δράσεις αυτές καθορίζονται από προγραμματισμένες ρουτίνες που καθορίζουν την κατεύθυνση, επιτάχυνση, ταχύτητα, επιβράδυνση και την απόσταση από μια σειρά συντονισμένων κινήσεων.

Άλλα ρομπότ είναι πολύ πιο ευέλικτα ως προς τον προσανατολισμό τους με βάση το αντικείμενο στο οποίο θα εκτελέσουν κάποια λειτουργούν. Για παράδειγμα τα ρομπότ για να έχουν πιο ακριβείς καθοδήγηση συχνά περιέχουν συστήματα μηχανικής όρασης που λειτουργούν ως τα «μάτια» τους, που συνδέονται με ισχυρούς υπολογιστές ή ελεγκτές[2].

1.2.2 Εγχώρια ή Οικιακά Ρομπότ

Το οικιακό ρομπότ είναι ένα ρομπότ που χρησιμοποιούνται για δουλειές του σπιτιού. Μέχρι στιγμής, υπάρχουν μόνο μερικά μοντέλα, αν και οι συγγραφείς επιστημονικής φαντασίας και άλλοι κερδοσκόποι έχουν προτείνει ότι θα μπορούσε να καταστεί συνηθέστερη μελλοντικά η χρήση των ρομπότ. Το 2006, ο Bill Gates έγραψε ένα άρθρο για το περιοδικό Scientific American με τίτλο "A Robot in Every Home".Πολλά εγχώρια ρομπότ που χρησιμοποιούνται για τις βασικές δουλειές του σπιτιού, όπως η Electrolux trilobite και Roomba βασίζονται στη

ρομποτική ηλεκτρική σκούπα, Neato. Τα οικιακά ρομπότ αποτελούν βασικό στοιχείο στο εγχώριο ρομποτικό σύστημα, διότι εκτελούν εξειδικευμένες διαδικασίες, όπως να μεταφέρουν αντικείμενα στο σπίτι(π.χ. ρούχα από το μπάνιο με το πλυντήριο ρούχων ή τα γυαλιά από το τραπέζι για το πλυντήριο πιάτων):Το 2006, η Sharp ανακοίνωσε ότι έχει αναπτύξει ένα ανθρωποειδές ρομπότ που καθαρίζει τα πιάτα από το τραπέζι και τα τοποθετεί σε ένα πλυντήριο πιάτων. Ανοίγει την πόρτα του πλυντηρίου πιάτων, παίρνει τη λαβή του τσαγιού και πιάτα, όπου και τα τοποθετεί στο πλυντήριο και κλείνει την πόρτα[3].

1.2.3 Ιατρικά ρομπότ

Ίσως η πιο σπουδαία εφαρμογή των ρομπότ είναι στην ιατρική. Στις μέρες μας χρησιμοποιείται η τεχνικής των ζευγαριών ,όπου ένας χειρούργος και κάποιο είδος μηχανισμού επιτρέπουν να πραγματοποιούνται χειρουργικές επεμβάσεις με πολύ μικρές τομές, μειώνοντας σημαντικά τον κίνδυνο για τους ασθενείς. Η ικανότητα του χειρουργού να ελέγχει το μηχανισμό ενισχύεται με την παροχή ανάδρασης στους ελέγχους, επιτρέποντας στον χειριστή να έχει την αίσθηση της αφής για να μπορεί να βοηθήσει στον έλεγχο του ρομπότ. Αυτό το είδος ρομπότ δεν είναι εντελώς ανεξάρτητο, και είναι πιο σωστά να ονομάζεται teleoperated συσκευή, όπου χρησιμοποιεί την τεχνολογία ενός ανεξάρτητου ρομπότ που θα μπορούσε να έχει τον έλεγχο της κίνησης του.Στόχος της ιατρικής κοινότητας είναι να αντικατασταίνονται τα ελλείποντα άκρα και τα όργανα με μηχανικές αντικαταστάσεις όπου θα μπορούν να αλληλεπιδρούν με το ανθρώπινο βιολογικό σώμα. Έρευνες για αντικαταστάσεις στην καρδιά, μάτια, αυτιά και άλλα όργανα έχουν προσφέρει ελπίδες για την ανάπτυξη αποτελεσματικών εμφυτευμένων συσκευών και οι αντικαταστάσεις στα άκρα λειτουργούν για μεγάλο χρονικό διάστημα. Επίσης οι ρομποτικές συσκευές μπορούν να παρέχουν βοήθεια σε άτομα με σοβαρούς περιορισμούς στη διακίνηση, όπου σε πολλές περιπτώσεις τους επιτρέπει τουλάχιστον κάποια ικανότητα να μετακινούνται[4] .

1.2.4 Στρατιωτικά Ρομπότ

Η κατηγορία των στρατιωτικών ρομπότ περιλαμβάνει αυτόνομα ρομπότ ή τηλεχειριζόμενες συσκευές σχεδιασμένες για στρατιωτικές εφαρμογές. Αυτά τα συστήματα αποτελούν σημαντικό αντικείμενο έρευνας από διάφορες ένοπλες δυνάμεις. Ένα από τα συστήματα που δημιουργήθηκαν είναι ένα αυτοματοποιημένο όπλο το οποίο αναπτύχτηκε περισσότερο για εμπορική και στρατιωτική χρήση. Επίσης κατασκευάστηκε ένα τετράτροχο ρομπότ εξοπλισμένο με πολλές κάμερες, ραντάρ, και, ενδεχομένως, ένα πυροβόλο όπλο, που εκτελεί αυτόματα τυχαία ή προγραμματισμένες εκ των προτέρων περιπολίες γύρω από μια

στρατιωτική βάση ή άλλη εγκατάσταση της κυβέρνηση. Ο χειριστής μπορεί να ειδοποιηθεί από το ρομπότ όταν ανιχνεύει κίνηση σε παράνομες περιοχές, ή σε άλλες προγραμματισμένες συνθήκες. Έτσι ο χειριστής μπορεί αναθέσει στο ρομπότ να αγνοήσει το γεγονός, ή να αναλάβει το τηλεχειριστήριο για να ασχοληθεί με έναν εισβολέα, ή για να έχει μια καλύτερη οπτική γωνία σε μια κατάσταση έκτακτης ανάγκης[5].

1.2.5 Ψυχαγωγικά Ρομπότ

Το ρομπότ ψυχαγωγίας είναι, όπως υποδηλώνει το όνομα, ένα ρομπότ που δεν κατασκευάζετε για χρηστική χρήση, όπως στην παραγωγή ή τις εγχώριες υπηρεσίες, αλλά αποκλειστικά για αναψυχή, το συναίσθημα αυτό οι μηχανές ακόμη και οι υπολογιστές, δεν είναι ικανά να το έχουν, όμως μπορούν να εξυπηρετούν τον άνθρωπο. Οι ρομποτικές τεχνολογίες εφαρμόζονται σε πολλούς τομείς του πολιτισμού και της ψυχαγωγίας, όπως για τη δημιουργία περιβάλλοντος αφήγησης στα εμπορικά κέντρα όπου χρησιμοποιούνται μοτέρ, αερομηχανική και υδραυλικά για τη δημιουργία κίνησης με προγραμματισμένες εκ των προτέρων συμπεριφορές, όπως η βόλτα στο στοιχειωμένο σπίτι στο Disneyland[6].

1.2.6 Εξερευνητικά ρομπότ

Οι άνθρωποι ενδιαφέρονται συνήθως για μέρη όπου πολλές φορές είναι γεμάτα κινδύνους, όπως το διάστημα ή ο ωκεανός. Όμως η πρόσβαση σε τέτοια μέρη είναι δύσκολη για τους ίδιους του ανθρώπους, γι αυτό δημιούργησαν ρομπότ τα οποία μπορούν να φτάσουν στις τοποθεσίες που θέλουν. Τα ρομπότ μπορούν να εκτελούν φωτογραφίσεις και να έχουν άλλα μέσα για την συλλογή πληροφοριών όπως επίσης και να στέλλουν τις πληροφορίες πίσω στον ανθρώπινο χειριστεί τους. Το υποβρύχιο ρομπότ «Οδύσσεια ΙΙβ» αναπτύχθηκε από τους επιστήμονες του M.I.T για την εξερεύνηση των ωκεανών. Το "Sojourner", είναι το ρομπότ όπου κατασκευάστηκε με σκοπό να σταλεί στον πλανήτη Άρη. Το Sojourner προσγειώθηκε στην επιφάνεια του Άρη στις 4 Ιουλίου, 1998[7].

Κεφάλαιο 2

Τεχνικές Εντοπισμού (Robot Localization techniques)

2.1 Έννοια Εντοπισμού Ρομπότ 11
2.2 Σχετικός ή Τοπικός Εντοπισμός (local) 11
2.3 Απόλυτος ή παγκόσμιος εντοπισμός 12
2.4 Πιθανολογικός Εντοπισμός (Probabilistic localization) 12
2.5 Τεχνικές Χαρτογράφησης(Mapping) 12

2.1 Έννοια Εντοπισμού Ρομπότ

Το πρόβλημα εντοπισμού περιλαμβάνει την ερώτηση «Που είμαι »από την πλευρά του ρομπότ. Αυτό σημαίνει ότι το ρομπότ πρέπει να ξέρει την θέση του σε σχέση με το περιβάλλον. Όταν μιλάμε για την θέση (location) ή τοποθεσία (pose), εννοούμε την συντεταγμένη x και y και την γωνία κατεύθυνσης ενός ρομπότ σε ένα παγκόσμιο σύστημα συντεταγμένων.

Το πρόβλημα εντοπισμού είναι ένα σημαντικό πρόβλημα. Είναι ένα βασικό συστατικό σε πολλά επιτυχημένα αυτόνομα ρομποτικά συστήματα. Εάν ένα ρομπότ δεν γνωρίζει πού είναι σε σχέση με το περιβάλλον του τότε είναι δύσκολο να αποφασίσει τι θα κάνει. Θα πρέπει να έχει τουλάχιστον κάποια ιδέα για το που βρίσκεται για να μπορεί να λειτουργεί και να ενεργεί με επιτυχία. Από μερικούς συγγραφείς, το πρόβλημα εντοπισμού του ρομπότ έχει δηλωθεί ως το πιο θεμελιώδες πρόβλημα για την παροχή ρομπότ με πραγματικά αυτόνομες ικανότητες.Αυτό το τμήμα εξετάζει τις βασικές ιδέες για να εντοπισμό (robot localization) των κινητών ρομπότ.Οι κύριες τεχνικές για τον εντοπισμό (localization) είναι ο σχετικός ή τοπικός εντοπισμός και ο απόλυτος ή παγκόσμιος εντοπισμό[8].

2.2 Σχετικός ή Τοπικός Εντοπισμός (local)

Οι σχετικές τεχνικές εντοπισμού βασίζονται στον προσδιορισμό της θέσης και τον προσανατολισμό του ρομπότ γνωρίζοντας ένα αρχικό σημείο. Για την παροχή των πληροφοριών αυτών χρησιμοποιούνται διάφοροι αισθητήρες όπως κωδικοποιητές (encoders), γυροσκόπια (gyroscopes), επιταχυνσιόμετρα (accelerometers) κλπ. Οι πιο συνηθισμένες

τεχνικές σχετικού εντοπισμού είναι η οδομετρία (odometry) και το « dead - reckoning» (οδομετρία και αδρανειακά συστήματα πλοήγησης). Η οδομετρία απασχολεί απλές γεωμετρικές εξισώσεις (κινηματική του κινητού ρομπότ) με κωδικοποιητές τροχών που παρέχουν τις γωνιακές ταχύτητες του .Με την ενσωμάτωση των προηγούμενων εξισώσεων υπολογίζεται η θέση και ο προσανατολισμός του οχήματος.

2.3 Απόλυτος ή παγκόσμιος εντοπισμός

Οι τεχνικές του απόλυτου εντοπισμού καθορίζουν τη θέση του ρομπότ σε σχέση με ένα παγκόσμιο πλαίσιο αναφοράς, για παράδειγμα, χρησιμοποιώντας φάρους ή ορόσημα. Η πιο δημοφιλής τεχνική είναι το GPS (Global Positioning System) η οποία χρησιμοποιώντας δορυφορικά σήματα μπορεί να προσδιορίσει την απόλυτη θέση (γεωγραφικό μήκος, πλάτος και ύψος) ενός αντικειμένου στη Γη. Στην περίπτωση του απόλυτου εντοπισμού η αύξηση του σφάλματος μειώνεται όταν οι μετρήσεις είναι διαθέσιμες. Η θέση του ρομπότ δεν εξαρτάται από το χρόνο και την αρχική θέση. Τα κύρια προβλήματα των τεχνικών που βασίζονται στα ορόσημα(landmarks) είναι:

- Απαιτεί μια δαπανηρή εγκατάσταση σχετικά με τον τομέα στον οποίο λειτουργεί το ρομπότ.
- Το κινητό ρομπότ μπορεί μόνο να περιηγηθεί πάνω από την περιοχή στην οποία βρίσκονται τα ορόσημα.
- Μεταξύ ορόσημων το ρομπότ δεν μπορεί να καθορίσει τον εντοπισμό του.

2.4 Πιθανολογικός Εντοπισμός (Probabilistic localization)

Οι Πιθανολογικής τεχνικές στηρίζονται στην εκτίμηση του εντοπισμού των κινητών ρομπότ συνδυάζοντας δεδομένα μετρήσεων και γνώσεων με τέτοιο τρόπο ούτως ώστε το σφάλμα να είναι στατιστικά πιο ελάχιστο. Η πιο εκτεταμένη τεχνική βασίζεται στο Kalman Filter. Μία από αυτές τις τεχνικές είναι το SLAM (Ταυτόχρονος Εντοπισμός και Χαρτογράφηση). Το SLAM, localization λειτουργά όπως το Kalman Filter, αλλά κάνει ενημέρωση του χάρτη. Ένα από τα μειονεκτήματα του SLAM είναι οι μεγάλες δομές δεδομένων που χρειάζονται να υπολογιστούν για να κατασκευαστεί και να γίνεται ενημέρωση του χάρτη.

2.5 Τεχνικές Χαρτογράφησης(Mapping)

Ένα από τα σημαντικότερα προβλήματα στον τομέα της ρομποτικής είναι γνωστή ως SLAM (Ταυτόχρονος Εντοπισμός και χαρτογράφηση), που αποτελείται από μια τεχνική που

χρησιμοποιείται από ρομπότ και αυτόνομα οχήματα για τη δημιουργία ενός χάρτη μέσα σε ένα άγνωστο περιβάλλον, παρακολουθώντας παράλληλα την τρέχουσα θέση τους. Αυτό δεν είναι τόσο απλό όσο ακούγεται λόγω της έμφυτης αβεβαιότητας στις μετρήσεις των διάφορων αισθητήρων του ρομπότ.

Οι τεχνικές για εντοπισμό με την χρησιμοποίηση χαρτών χρησιμοποιούν γεωμετρικά χαρακτηριστικά του περιβάλλοντος για τον υπολογισμό της θέση του ρομπότ. Παραδείγματα γεωμετρικών σχημάτων είναι γραμμές που περιγράφουν τοίχους ή τετράγωνα ή κύκλους για την απεικόνιση αντικειμένων.

Η αναπαράσταση των χαρτών μπορεί είναι διαφορετική. Μπορεί είτε να είναι μετρική ή τοπολογική. Μετρική χάρτες περιέχουν το περιβάλλον σε παγκόσμιο σύστημα συντεταγμένων ενώ οι τοπολογικοί χάρτες περιέχουν το περιβάλλον με τη μορφή ενός δικτύου όπου οι κόμβοι αντιπροσωπεύουν θέσεις στο περιβάλλον. Χρησιμοποιώντας τεχνικές μοντέλων ταίριαξης για να προσδιορισμό της απόλυτη θέση του ρομπότ έχει το μειονέκτημα ότι πρέπει να υπάρχουν επαρκής πληροφορίες από αισθητήρες για να συνδυάζονται με το χάρτη και να εντοπίζεται η θέση .Επιπλέον οι τεχνικές για την ταίριασμα δεδομένων των αισθητήρων με χάρτες συχνά απαιτούν μεγάλες ποσότητες υπολογιστικής ισχύς και αίσθησης[8].

Κεφάλαιο 3

Lego Mindstorms NXT

3.1	Κύρια χαρακτηριστικά Nxt brick [9]	14
3.2	Αισθητήρες Nxt από Mindstorm	15
3.3	Αισθητήρες Nxt από άλλους Κατασκευαστές	17
3.4	Προγραμματίστηκες γλώσσες Nxt	20
3.5	LeJOS NXJ	21

3.1 Κύρια χαρακτηριστικά Nxt brick [9]

- Κύριος επεξεργαστής : Atmel® 32bit.
  - ARM® processor, AT91SAM7S256.
  - 256 KB FLASH.
  - 64 KB RAM.
  - 48 MHz.

- Δεύτερος επεξεργαστής :Atmel® 8bit AVR processor, ATmega48.
  - 4 KB FLASH.
  - 512 Byte RAM.
  - 8 MHz.
- Ασύρματη επικοινωνία Bluetooth CSR BlueCoreTM 4 v2.0 +EDR System.
  - Υποστηρίζει το Serial Port Profile (SPP).
  - Εσωτερική 47 KByte RAM.
  - Εξωτερική 8 MBit FLASH.
  - 26 MHz.

- USB 2.0 επικοινωνία με ταχύτητα 12 Mbit/s.
- 4 θύρες εισόδων που υποστηρίζουν ψηφιακές και αναλογικές διασυνδέσεις.
  - Μία θύρα υψηλής ταχύτητας IEC 61158 Type 4/EN 50170 compliant.

- 3 θύρες εξόδων που υποστηρίζουν είσοδο από κωδικοποιητή.
- Μαυρόασπρη οθόνη 100 x 64 pixel LCD.
- Μεγάφωνο με κανάλι εξόδου ήχου 8-bit ανάλυση.
    - Υποστηρίζει ρυθμό δειγματοληψίας δείγματος 216 KHz.
- 4 κουμπιά για διεπαφή με το χρήστη.

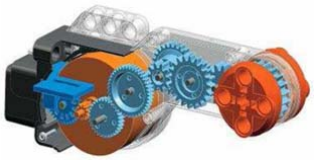
Πηγή ισχύος 6 μπαταρίες AA

3.2 Αισθητήρες Nxt από Mindstorm

Η εκπαιδευτική έκδοση του Lego Mindstorms NXT περιλαμβάνει ένα προγραμματιζόμενο τούβλο, τρεις κινητήρες, έναν αισθητήρα φωτός, ένα αισθητήρα υπερήχων (σόναρ), έναν αισθητήρα ήχου, και δύο αισθητήρες αφής. Επίσης υπάρχουν διαθέσιμοι αισθητήρες από δευτερογενής αγορές όπως η Mindsensors ή HiTechnic, οι οποίες παρέχουν συστήματα όρασης, υπέρυθρους ανιχνευτές, αισθητήρες χρώματος και πυξίδες

Nxt μοτέρ

Περιλαμβάνει κωδικοποιητή περιστροφή και έτσι μπορεί να δώσει στο Nxt πληροφορίες για τη θέση του άξονα με 1 ανάλυση[10]



Αισθητήρας φωτός

Ο αισθητήρας φωτός βοηθά το ρομπότ να "βλέπει". Επιτρέπει στο ρομπότ να διακρίνει ανάμεσα στο φως και σκοτάδι, και να καθορίζει την ένταση του φωτός σε ένα δωμάτιο ή την ένταση του φωτός σε διάφορα χρώματα[10].

Αισθητήρας υπέρηχων

Ο αισθητήρας υπέρηχων βοηθά το Nxt να μετρά αποστάσεις και να ανακαλύπτει που βρίσκονται αντικείμενα. Ο αισθητήρας αυτός χρησιμοποίει την ίδια τεχνική που χρησιμοποιούν οι νυχτερίδες .Μετρά αποστάσεις υπολογίζοντας τον χρόνο που χρειάζεται ένα ηχητικό κύμα να χτυπήσει σε αντικείμενο και να επιστρέψει πίσω(όπως η ηχώ).Μεγάλα αντικείμενα με σκληρές επιφάνειες δίνουν καλές μετρήσεις ενώ αντικείμενα με απαλές επιφάνειες και καμπυλωτά σχήματα δύσκολα ανιχνεύονται από τον αισθητήρα αυτό[10].

Αισθητήρας αφής

Ο αισθητήρας αυτός ανιχνεύει πότε ο αισθητήρας πατιέται από και πότε αφήνεται.

Αισθητήρας ήχου

Αισθητήρας ήχου μπορεί να ανιχνεύσει dB (decibels) και dBA (adjusted decibel). Το ντεσιμπέλ είναι μονάδα μέτρησης του ήχου. dBA είναι οι ήχοι που το ανθρώπινο αυτί μπορεί να ακούσει ενώ στο dBA μπορεί να υπάρχουν ήχοι πολύ χαμηλή ή πολύ ψηλή που δεν μπορεί να ακούσει το ανθρώπινο αυτί[10].

3.3 Αισθητήρες Nxt από άλλους Κατασκευαστές

Αισθητήρας πυξίδα

Αυτός ο αισθητήρας περιέχει μια μαγνητική πυξίδα που μετρά τα μαγνητικά πεδία της Γής και υπολογίζει την γωνία κατεύθυνσης. Παρέχει 8 κατεύθυνσης (N,NE,E,SE,S,SW,W και NW) και 4 σήματα N,S,E,W διαβάζονται από τον κοντρόλερ για να καθορίσει την κατεύθυνση του ρομπότ[11].

Αισθητήρας Γυροσκόπιο (Gyro sensor)

Αυτός ο αισθητήρας επιτρέπει στο NXT να εντοπίζει περιστροφή με ακρίβεια. Το NXT Gyro Sensor επιστρέφει τον αριθμό των μοιρών ανά δευτερόλεπτο περιστροφή όπως επίσης και την κατεύθυνση της περιστροφής[12].

Αισθητήρας απόστασης (Optical distance sensor)

Όπως και ο αισθητήρας υπερήχων έτσι και αυτός μετρά απόσταση από κάποιο αντικείμενο αλλά χρησιμοποιώντας υπέρυθρες. Στέλλει υπέρυθρες και όταν αντανακλούνται από το αντικείμενο ,επιστρέφουν πίσω στον αισθητήρα όπου μετριέται η γωνία τον αντανακλώμενων υπέρυθρων και υπολογίζεται η απόσταση του αντικειμένου. Δίνει καλύτερες μετρήσεις από τον αισθητήρας υπερήχων[13].

NxtSumoEyes

Είναι ένας αισθητήρας για εντοπισμό αντικειμένων. Μπορεί να ανίχνευση αντικείμενα στα αριστερά του, μπροστά και στα δεξιά του. Υπάρχουν δύο επιλογές η μικρή ακτίνα ανίχνευσης (Short range zone) που ανιχνεύει αντικείμενα στα 15 cm και η μεγάλη (Long Range zone) που ανιχνεύει αντικείμενα στα 30 cm[15]

NxtCam

Αυτός ο αισθητήρας σχεδιάστηκε για να δώσει τεχνική όραση στο Nxt και παρέχει τις πιο κάτω ικανότητες[14]:

- Να εντοπίζει 8 διαφορετικά χρωματιστά αντικείμενα στα 30 frames/second.
- Ρυθμίζεται μέσω Usb διεπαφής σε Windows Xp και Windows Vista.
- Υποστηρίζει δύο λειτουργίες εντοπισμού: Εντοπισμός γραμμής και αντικειμένου.
- Παρέχει σε πραγματικό χρόνο στατιστικές για το αντικείμενο που εντοπίζεται (αριθμός αντικειμένων, χρώματα αντικειμένων και συντεταγμένες).
- Λαμβάνει εικόνες με ανάλυση 88 x 144 se 30 frames/second.
- Δίνει την δυνατότητα μεταφοράς εικόνας στο Pc μέσο Usb θύρας.
- Χαμηλή κατανάλωση ρεύματος(60 mA).
- Χρησιμοποιεί το I2C για επικοινωνία με το Nxt

Διαδικασία αποστολής μοτίβων χρωμάτων στην NxtCam

Πρώτα βγάζουμε φωτογραφία του αντικειμένου που θέλουμε να εντοπίσουμε(Fig.1).

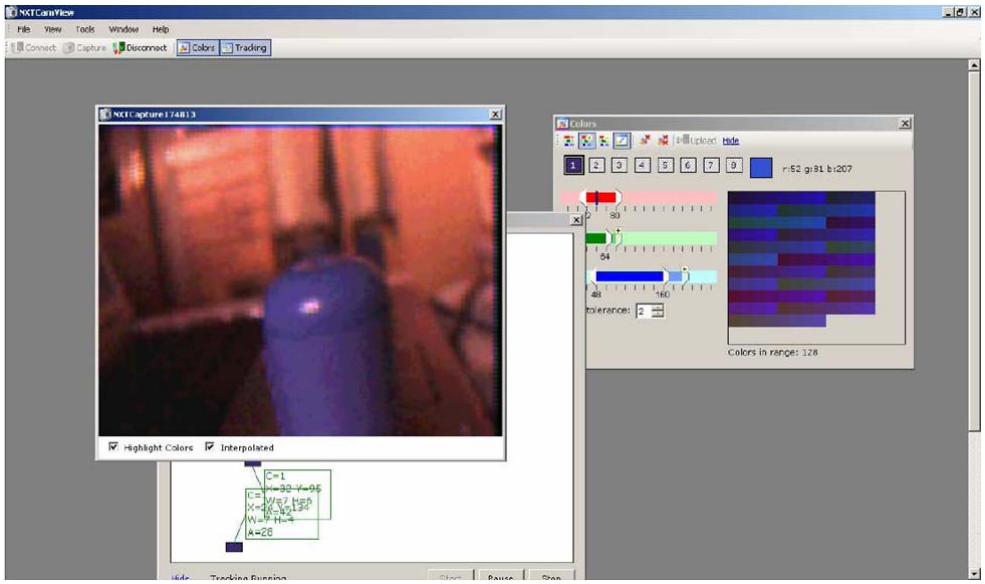


Fig.1. NxtCamView – Capture Image

Στην συνέχεια επιλέγουμε το μοτίβο χρώματος πάνω στην φωτογραφία του αντικειμένου που πήραμε και το στέλλουμε στον αισθητήρα όπως φαίνεται πιο κάτω(Fig.2).

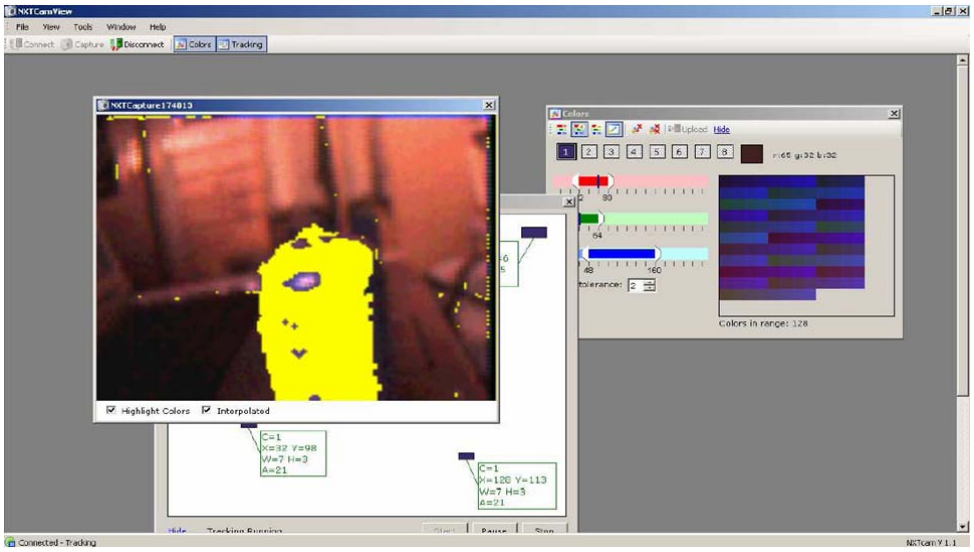


Fig.2. NxtCamView – Select Colour Patterns

Μπορούμε να δοκιμάσουμε τον εντοπισμό του μοτίβου χρώματος που στείλαμε στην NxtCam χρησιμοποιώντας την επιλογή εντοπισμού στο NxtCamView πρόγραμμα όπως φαίνεται πιο κάτω(Fig.3).

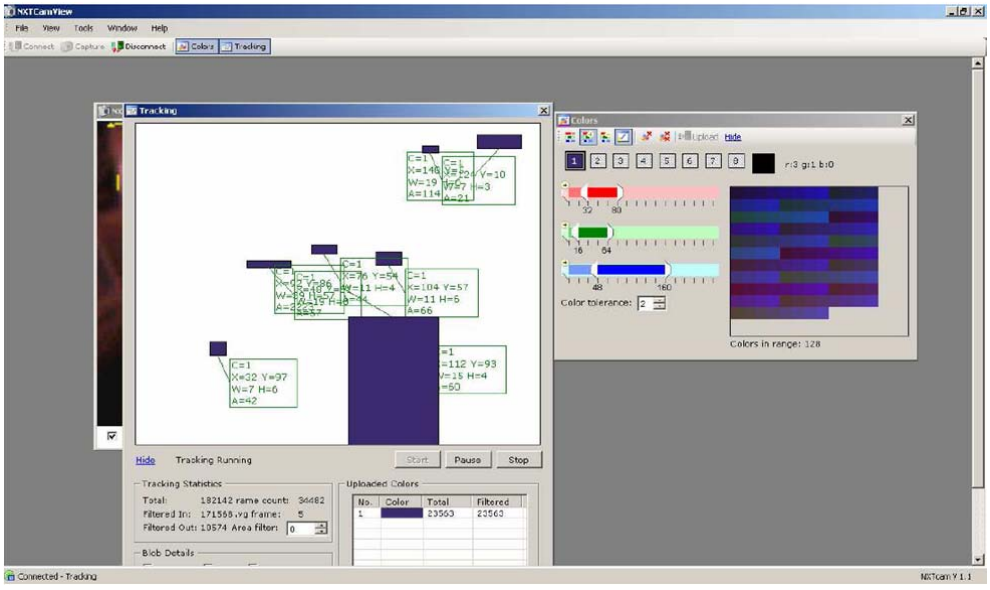


Fig.3. NxtCamView – Tracking Mode

3.4 Προγραμματίστηκες γλώσσες Nxt

NXT-G

NXT-G v1.0 είναι το λογισμικό προγραμματισμού που έρχεται πακέτο με τον το NXT. Αυτό το λογισμικό είναι κατάλληλο για βασικό προγραμματισμό ,όπως είναι η οδήγηση μοτέρ, είσοδος τιμών από αισθητήρων, να κάνει υπολογισμούς, και για τη μάθηση της βασικής δομή προγραμματισμού και του ελέγχου ροής. Ο προγραμματισμός γίνεται με την χρήση εικονιδίων που εκτελούν κάποιες λειτουργίες σε ένα γραφικό περιβάλλον[16].

C# Με Microsoft Robotics Developer Studio

Δωρεάν εργαλεία (Visual Studio Express και το Robotics Developer Studio) επιτρέπουν τον προγραμματισμό με την χρήση της γλώσσας C#[16].

BricxCC, Next Byte Codes, Not eXactly C

Bricx Command Center (BricxCC) είναι το ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) που χρησιμοποιείται για γράψιμο, την μεταγλώττιση, και επεξεργασία NBC και NXC προγραμμάτων για το NXT. Επίσης το BricxCC έγινε αρχικά γιατο RCX και έτσι τα προγράμματα για να μπορεί να γραφτούν χρησιμοποιώντας NQC μέσω BricxCC. BricxCC έχει πολλά εργαλεία όπως το NeXTExplorer (upload /download αρχείων), NeXTScreen (μπορείς να βλέπεις στην οθόνη LCD του NXT, και να συλλάβει τις εικόνες και βίντεο).Next Byte (NBC) είναι μια απλή ανοιχτή γλώσσα προγραμματισμού με σύνταξη μιας συμβολικής γλώσσας προγραμματισμού και μπορεί να χρησιμοποιηθεί για να προγραμματίσετε το NXT. Όχι ακριβώς C (NXC) είναι μια γλώσσα ανοικτού κώδικα ,υψηλού επιπέδου, παρόμοια με C, χτισμένη στον μεταγλωττιστή NBC. Πρόκειται για μια από τις πιο ευρέως χρησιμοποιούμενες γλώσσες προγραμματισμού για το NXT[16].

Robot C

Αναπτύχθηκε από την Ακαδημία του Carnegie Mellon Ρομποτικής , η ROBOTC είναι μια γλώσσα προγραμματισμού που βασίζεται σε C για VEX Cortex ρομπότ και τα Lego Mindstorms. Η ROBOTC τρέχει ένα πολύ βελτιστοποιημένο firmware που επιτρέπει το NXT να τρέχει προγράμματα πολύ γρήγορα, καθώς επίσης και .συμπιέζει .τα .αρχεία ώστε να μπορείτε να χωράει μεγάλο ποσό προγραμμάτων στο NXT.Όπως και άλλες γλώσσες NXT, ROBOTC απαιτεί δικό της firmware για να τρέξει[16]

Lejos NXJ

leJOS NXJ είναι μια υψηλού επιπέδου γλώσσα ανοικτού κώδικα βασισμένη στην JAVA που χρησιμοποιεί δικό της firmware και αναπτύχτηκε από την ομαδα leJOS [16].

3.5 LeJOS NXJ

Το όνομα leJOS επινοήθηκε από τον Jose Solórzano, με βάση το ακρωνύμιο για Java Operating System (JOS) και την ισπανική λέξη "lejos".
Το leJOS NXJ είναι ένα έργο για την ανάπτυξη ενός Java Virtual Machine (JVM) για το Lego Mindstorms NXT, με αντικατάσταση του αρχικού firmware του NXT,επιτρέπει στα ρομπότ NXT να προγραμματιστούν στην γλώσσα προγραμματισμού Java, παρέχοντας βιβλιοθήκες που υποστηρίζουν διάφορες ενδιαφέρουσες υψηλότερες λειτουργίες όπως η πλοήγηση και η ρομποτική με βάση συμπεριφορών[17].
Ξεκίνησε ως χόμπι, ένα έργο ανοικτού πηγαίου κώδικα, του José Solórzano στα τέλη του 1999, η leJOS NXJ έχει εξελιχθεί από τότε με νέα χαρακτηριστικά. Η τελευταία έκδοση από την ερευνητική ομάδα leJOS είναι η 0.85 διαθέσιμη για τα Microsoft Windows και Linux / Mac OS X λειτουργικά συστήματα. Περιλαμβάνει plug-in για Netbeans και για Eclipse IDE που διευκολύνουν πολύ τον χρήστη [18].
Από προσωπική μου εμπειρία η πλατφόρμα leJOS NXJ, αποδείχθηκε να είναι μια εξαιρετική επιλογή, όχι μόνο για αυτό το έργο, αλλά για κάθε είδους ρομποτικών έργων που χρησιμοποιούν το Lego Mindstorms NXT λόγω του ότι το leJOS NXJ προσφέρει διάφορες δυνατότητες όπως πολλαπλά νήματα, συγχρονισμό, πολυδιάστατους πίνακες και καλή τεκμηρίωσης των ρομποτικών βιβλιοθηκών

Νήματα

Στην LeJOS, το multithreading υποστηρίζεται με κληρονομιά του αντικειμένου Thread. Τα νήματα έπαιξαν σημαντικό ρόλο στο σχεδιασμό του λογισμικού για το έργο αυτό για τον λόγο ότι επιτρέπουν πολλαπλές διεργασίες να εκτελούνται παράλληλα σε ένα πρόγραμμα. Όταν εμπλέκονται επικοινωνίες ,ένα νήμα μπορεί να ακούει για εντολές σε ένα κανάλι επικοινωνίας, ενώ το ρομπότ στέλνει δεδομένα σε μια απομακρυσμένη συσκευή ή να εκτελεί κάποια άλλη εργασία. Επιπλέον τα νήματα επιτρέπουν πολλαπλά κανάλια επικοινωνίας να αξιοποιηθούν, επιτρέποντας σε έναν NXTτούβλο για να στέλλει δεδομένα σε έναν σταθμό βάσης ενώ παράλληλα να συλλέγει.Ένας περιορισμός σε threading σε ένα τούβλο NXT είναι το μέγεθος της διαθέσιμης μνήμης. Επειδή κάθε νήμα καταλαμβάνει ένα μέρος της μνήμης,

έχει προταθεί ότι ο μέγιστος αριθμός των νημάτων που μπορούν να δημιουργηθούν σε ένα πρόγραμμα περιορίζεται στα οκτώ

Επικοινωνίες και Πρωτόκολλα Επικοινωνίας στην LeJOS

Η LeJOS υποστηρίζει Bluetooth ασύρματη επικοινωνία μέσω του πρωτοκόλλου σειριακής θύρας (serial port protocol) καθώς επίσης και επικοινωνία μέσω σύνδεσης USB. Ανεξάρτητα αν χρησιμοποιηθεί Bluetooth ή USB η σύνδεση με το NXT γίνεται μέσο ενός DataInputStream και ενός DataOutputStream που δημιουργούνται όταν η σύνδεση είναι ανοιχτή μεταξύ του NXT και μια απομακρυσμένη συσκευή[18]. Για απομακρυσμένο έλεγχο του ρομπότ μπορεί να χρησιμοποιηθεί το πρωτόκολλο της Lego LCP(Ego Communication Protocol) που δεν απαιτεί αντικατάσταση του firmware στο Nxt ούτε να τρέχει κάποιο πρόγραμμα σε αυτό. Αυτή η τεχνική μπορεί να χρησιμοποιηθεί από πρόγραμμα στο PC όπως επίσης και από κινητά τηλέφωνα η άλλες συσκευές που τρέχουν Java ME και Android.

.

Κεφάλαιο 4

Αρχιτεκτονική Ρομπότ

4.1	Υλικό που Χρησιμοποιήθηκε	23
4.2	Επικοινωνία Μεταξύ Συστατικών Υλικού	24
4.3	Σχεδίαση και Συναρμολόγηση του Ρομπότ	26
4.4	Πλοήγηση Ρομπότ	28
4.5	Φωτογραφίες ρομπότ	32

Όπως και στα περισσότερα συστήματα η σχεδίαση της αρχιτεκτονική καθορίζει τα δομικά στοιχεία τα οποία θέτουν τα θεμέλια για το συστήματος . Ένα σύστημα συνήθως αποτελείται από διάφορα συστατικά, όπου κάθε συστατικό έχει τον δικό του σκοπό. Τα συστατικά αυτά πρέπει να επικοινωνούν μεταξύ τους, ώστε το σύστημα να λειτουργεί ομαλά. Η επέκταση του συστήματος και η προσθήκη νέων στοιχείων σε αυτό, κατά τη φάση εξέλιξης του συστήματος, είναι επίσης ένα άλλο θέμα προς εξέταση κατά το σχεδιασμό της αρχιτεκτονικής. Η ποιότητα ενός συστήματος έγκειται στην ποιότητα της αρχιτεκτονικής του.

Στο έργο μας υπάρχουν μερικά ζητήματα που πρέπει να εξεταστούν κατά το σχεδιασμό της αρχιτεκτονικής του ρομπότ .

Αυτά περιλαμβάνουν:

1. Το υλικό που θα χρησιμοποιηθεί - που θα είναι αισθητήρες, Nxt μονάδες, καθώς και κάποιο κινητό τηλέφωνο.
2. Η επικοινωνία μεταξύ συστατικών υλικού.

Η σχεδίαση και συναρμολόγηση του ρομπότ ώστε να είναι πρακτικό στην κίνηση του για να μπορεί να εκτελεί τις βασικές κινήσεις(περιστροφές και να κινείται σε ευθεία) και η τοποθέτηση των αισθητήρων σωστά πάνω στο ρομπότ.

4.1 Υλικό που Χρησιμοποιήθηκε

Χρησιμοποιήθηκαν δύο μονάδες Nxt όπου θα επικοινωνούν μεταξύ τους για ανταλλαγή δεδομένων όπως τιμές από αισθητήρες και αποτελέσματα υπολογισμών. Η χρησιμοποίηση δύο μονάδων Nxt δίνει την δυνατότητα στο ρομπότ να έχει συνδεδεμένους περισσότερους αισθητήρες όπως και να μπορεί να κινήσει περισσότερα μοτέρ αφού κάθε Nxt έχει μόνο 3

θύρες για μοτέρ και 4 για αισθητήρες. Για την πλοήγηση του ρομπότ χρησιμοποιήθηκαν δύο αισθητήρες ,ένα γυροσκόπιο και μια πυξίδα ,όπου δίνουν την δυνατότητα στο ρομπότ να γνωρίζει την κατεύθυνση του. Επίσης χρησιμοποιήθηκε ένας αισθητήρας απόστασης με συνδυασμό με ένα μοτέρ για την υλοποίηση ενός ραντάρ όπου θα ανιχνεύει τα αντικείμενα μπροστά του ώστε να μην προσκρούει σε αυτά. Άλλος βασικός αισθητήρας που δίνει μηχανική όραση στο ρομπότ είναι η NxtCam η οποία έχει την δυνατότητα να εντοπίζει μοτίβα-χρώματα και να επιστρέφει τις συντεταγμένες τους. Αυτός ο αισθητήρας χρησιμοποιήθηκε για να δώσει την δυνατότατα στο ρομπότ να μετακινεί ορισμένα αντικείμενα προκαθορισμένου χρώματος όπως και να ακλουθεί κάποιο αντικείμενο. Για να μπορεί το ρομπότ ακολουθεί γραμμές στο πάτωμα χρησιμοποιήθηκε ένας αισθητήρας φωτός και τέλος χρησιμοποιήθηκε ένα NxtSumoEyes που ανιχνεύει αντικείμενα (δεξιά, αριστερά και μπροστά του)που δίνει στο ρομπότ την δυνατότητα να γνωρίζει πότε θα κλίσει την δαγκάνα του για να πιάσει κάποιο αντικείμενο.

4.2 Επικοινωνία Μεταξύ Συστατικών Υλικού

Για την επικοινωνία μεταξύ των δύο μονάδων Nxt χρησιμοποιήθηκε το πρωτοκόλλου RS485/BitBus που υποστηρίζεται από την Θύρα 4 στο Nxt . Το RS485/BitBus πρωτόκολλο υλοποιήθηκε από την ομάδα LeJOS , μια master/slave υλοποίηση του δικτύου χρησιμοποιώντας SDLC πακέτα και CRC-16-CITT έλεγχο λαθών[18]. Αυτό το πρωτόκολλο ενσωματώθηκε με το πρότυπο των συνδέσεων της LeJOS και προσφέρει έως και 7 συνδέσεις.
Υλοποιήθηκαν δύο πρωτόκολλα υψηλού επιπέδου για να μπορεί το master Nxt να κινεί τα μοτέρ του slave Nxt όπως και επίσης να λαμβάνει τις τιμές που παίρνονται από τούς αισθητήρες του.

Κίνηση μοτέρ slave Nxt

Για την κίνηση των μοτέρ του slave Nxt υλοποιήθηκαν οι εξής κλάσεις :

- NXTRemoteMotor
- NXTMotorCommand
- RemoteMotorCommands
- NXTRemoteMotorControl

Το πρωτόκολλο αυτό λειτούργει ως ακόλουθος :

1. Στο slave Nxt τρέχει ένα αντικείμενο που είναι υλοποιημένο ως νήμα τύπου NXTRemoteMotorControl όπου αναμένει σύνδεση από το master Nxt για να παίρνει εντολές και τις εκτελεί.
2. Η σύνδεση ανοίγεται από το master Nxt με το αντικείμενο NXTMotorCommand
3. Στην συνέχεια μπορούν να δημιουργηθούν πάνω στην σύνδεση αυτή έως και τρία αντικείμενα NXTRemoteMotor.

Παράδειγμα Χρησιμοποίησης

- Στο slave Nxt :

```
NXTRemoteMotorControl control= new
NXTRemoteMotorControl(MotorPort.A,MotorPort.B, MotorPort.C);
control.setPriority(Thread.MAX_PRIORITY);
control.setDaemon(true);
control.start();
```

- Στο master Nxt:

```
NXTMotorCommand command=new NXTMotorCommand();
command.connect("NXT");
NXTRemoteMotor motorA=new NXTRemoteMotor(command,1);
NXTRemoteMotor motorB=new NXTRemoteMotor(command,2);
```

Διάβασμα τιμών αισθητήρων slave Nxt

Για να λαμβάνει το master Nxt τις τιμές που διαβάζονται από τις αισθητήρες του slave Nxt υλοποιήθηκαν οι εξής κλάσεις :

- `NXTRemoteSensorControl`
- `NXTSensorCommand`
- `RemoteSensorCommands`
- `NXTRemoteSumoEyesSensor`
- `NXTRemoteCompassSensor`
- `NXTRemoteColorSensor`

Το πρωτόκολλο αυτό λειτούργει με παρόμοιο τρόπο όπως για τον έλεγχο των μοτέρ :

1. Στο slave Nxt τρέχει ένα αντικείμενο που είναι υλοποιημένο ως νήμα τύπου NXTRemoteSensorControl όπου αναμένει σύνδεση από το master Nxt για να παίρνει εντολές και τις εκτελεί.
2. Η σύνδεση ανοίγεται από το master Nxt με το αντικείμενο NXTSensorCommand
3. Στην συνέχεια μπορούν να δημιουργηθούν πάνω στην σύνδεση αυτή αντικείμενα για έλεγχο των τριών αισθητήρων που αυτά είναι NXTRemoteSumoEyesSensor, NXTRemoteCompassSensor και NXTRemoteColorSensor

Παράδειγμα Χρησιμοποίησης

– Στο slave Nxt

```
NXTRemoteSensorControl sensorControl= new NXTRemoteSensorControl();
sensorControl.setPriority(Thread.MAX_PRIORITY);
sensorControl.setDaemon(true);
sensorControl.start();
```

– Στο master Nxt.

```
NXTSensorCommand command=new NXTSensorCommand();
command.connect(“NXT”);
NXTRemoteCompassSensor compass=newNXTRemoteCompassSensor(command,1);
NXTRemoteSumoEyesSensor sumoEyes=new NXTRemoteSumoEyesSensor
(command,2);
NXTRemoteColorSensor sumoEyes=new NXTRemoteColorSensor (command,3);
```

4.3 Σχεδίαση και Συναρμολόγηση του Ρομπότ

Η πιο σημαντική απόφαση που έπρεπε να παρθεί είναι η κινηματική του ρομπότ που με βάση θα καθοριστούν οι τριγωνομετρικές εξισώσεις για ώστε το ρομπότ ανά πάσα στιγμή να γνωρίζει την θέση του(καρτεσιανές συντεταγμένες) και την κατεύθυνση του.Με βάση του υλικού επιλεκτικέ η μέθοδος της Διαφορικής οδήγησης (Differential drive) που είναι μια απλή μέθοδος και πολύ διαδιδόμενη. Η μέθοδος αυτή απαιτεί το ρομπότ να έχει δυο διαφορετικά ανεξάρτητα μοτέρ με κωδικοποιητές πάνω σε αυτά. Το όνομα σχετίζεται με το

γεγονός ότι το διάνυσμα κίνησης του ρομπότ είναι το άθροισμα των ανεξάρτητων κινήσεων των τροχών, κάτι που ισχύει και για το μηχανικό διαφορικό (ωστόσο, το σύστημα αυτό το δεν χρησιμοποιεί ένα μηχανικό διαφορικό όπως τα αυτοκίνητα). Οι κινητήριοι τροχοί συνήθως τοποθετούνται σε κάθε πλευρά του ρομπότ και στο μπροστινό του μέρος. Η γραμμική κίνηση ρομπότ επιτυγχάνεται με στροφή των μοτέρ με τον ίδιο ρυθμό στην ίδια κατεύθυνση ενώ η επιτόπου στροφή επιταχύνεται με την περιστροφή των μοτέρ με το ίδιο ρυθμό στην αντίθετη κατεύθυνση(Fig.1)[19,21].

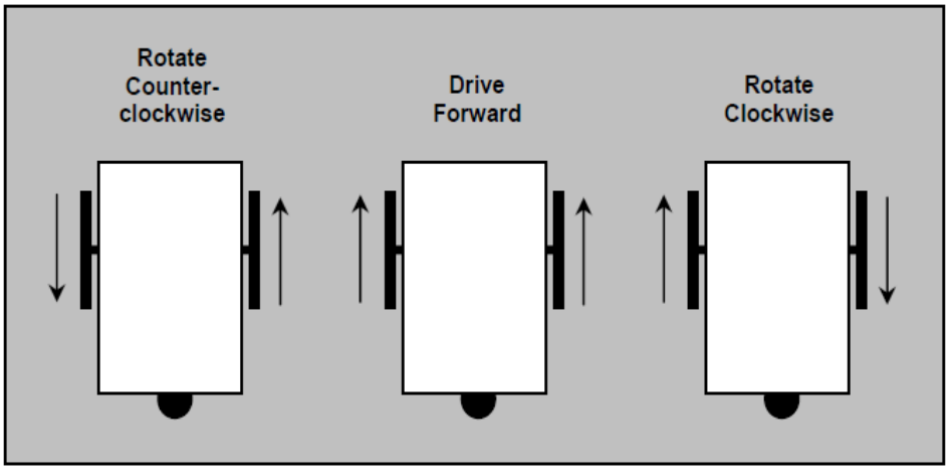


Fig.1. Differential drive

Εξισώσεις Dead Reckoning για Διαφορικής Οδήγηση

Έχουμε τρις εξισώσεις για τον υπολογισμό της χ ,ψ συντεταγμένης και την κατεύθυνσης θ:

$$\Delta x = R_W \cos(\theta)(T_1 + T_2)\frac{\pi}{T_R}$$

$$\Delta y = R_W \sin(\theta)(T_1 + T_2)\frac{\pi}{T_R}$$

$$\Delta\theta = 2\pi\frac{R_W}{D}\frac{T_1 - T_2}{T_R}$$

Όπου T1 είναι τα αριθμός των κλικ του κωδικοποιητή στο μοτέρ ένα,T2 ο αριθμός των κλικ του κωδικοποιητή στο μοτέρ δύο , Rw είναι η ακτίνα κάθε τροχού που είναι συνδεδεμένος στο μοτέρ,D είναι η απόσταση μεταξύ των δύο μοτέρ και το TR είναι αριθμός των κλικ του κωδικοποιητή που απαιτητέ για μια πλήρη περιστροφή του ρομπότ.

Στο δικό μας ρομπότ για τον λόγο ότι τα μοτέρ της Lego δεν έχουν ικανοποιητική ροπή για να μπορούν μόνο δύο μοτέρ να κινήσουν το ρομπότ αποφασίστηκε να χρησιμοποιηθούν τέσσερα μοτέρ για διαφορική οδήγηση αντί για δύο. Έτσι το ρομπότ θα έχει μεγαλύτερη ροπή για να κινείται περισσότερο ευέλικτα. Για τον αριθμό των κλικ του κωδικοποιητή για τον υπολογισμό των εξισώσεων της διαφορικής οδήγησης χρησιμοποιήθηκε ο μέσος όρος των κλικ για τα δύο μοτέρ. Αυτό μας δίνει και το πλεονέκτημα να μειώσουμε τον θόρυβο που δημιουργείται στους κωδικοποιητές και να έχουμε πιο ακριβή υπολογισμό της θέσης του ρομπότ στο περιβάλλον[20].

4.4 Πλοήγηση Ρομπότ

Η LeJOS προσφέρει τρία αφαιρετικά επίπεδα κλάσεων για έλεγχο οχημάτων με δύο τροχούς . Αυτή η αρχιτεκτονική επιτρέπει στον χρήστη να επιλέξει το επίπεδο πού είναι περισσότερο χρήσιμο και βολικό για την εφαρμογή του ώστε να μην ασχολείται με τα χαμηλότερα επίπεδα [18]

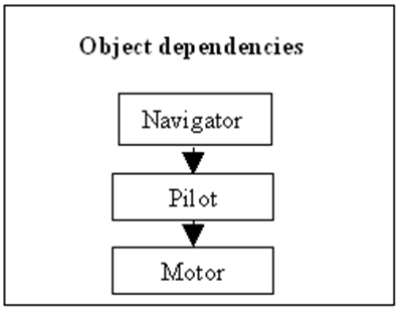


Fig.1. Navigation class hierarchy

Όπως φαίνεται από την πιο πάνω ιεραρχία κλάσεων (Fig.1)στον χαμηλότερο επίπεδο έχουμε την κλάση Motor όπου έχει τις βασικές μεθόδους για έλεγχο των μοτέρ του Nxt όπως [18]:

- forward()
- backward()
- stop()
- isMoving()
- isStalled()
- rotate(int angle, boolean immediateReturn)
- rotateTo(int limitAngle, boolean immediateReturn)
- setSpeed(int speed)

- setAcceleration(int acceleration)
- resetTachoCount()
- getTachoCount()

Στο μεσαίο επίπεδο βρίσκεται η κλάση Pilot
όπου οδηγεί το όχημα ελέγχοντας την ταχύτητα και την περιστροφή των μοτέρ. Η κλάση Pilot χρειάζεται να ξέρει την διάμετρο των τροχών και το πλάτος του οχήματος για να μπορεί να υπολογίσει την απόσταση που ταξίδεψε και την γωνία που περιστράφηκε το όχημα[18].
Περιλαμβάνει μεθόδους όπως[18] :

- setSpeed(final int leftSpeed, final int rightSpeed)
- setTravelSpeed(final double travelSpeed)
- setAcceleration(int accel)
- forward()
- backward()
- rotateLeft()
- rotateRight()
- rotate(final double angle)
- rotate(final double angle, final boolean immediateReturn)
- stop()
- travel(final double distance, final boolean immediateReturn)
- arc(final double radius, final double angle)
- travelArc(double radius, double distance)
- steer(final double turnRate, double angle)
- isMoving()

Στο ψηλότερο επίπεδο βρίσκεται η κλάση Navigator η οποία χειρίζεται την πλοήγηση του ρομπότ σε καρτεσιανές συντεταγμένες. Η κλάση αυτή έχει ένα αντικείμενο τύπου DeadReckonerPoseProvider που παρακολουθεί τις κινήσεις του Pilot με την χρήση Listeners, και με την χρήσης της Dead Reckoning τεχνικής υπολογίζει την θέση του ρομπότ σε συντεταγμένες (x, και y) και την κατεύθυνση του[18].
Η βασικές μέθοδοι που παρέχει η κλάση Navigator είναι[18]:

- goTo(double x, double y) .Όπου το ρομπότ θα πάει στην συντεταγμένη(χ,ψ) .

- goTo(WayPoint destination). Όπου το WayPoint είναι ένα αντικείμενο που περιέχει μια συντεταγμένη(χ,ψ) και την κατεύθυνση θ που θα βλέπει το ρομπότ όταν φτάσει σε αυτήν την συντεταγμένη.
- followRoute(Collection<WayPoint>aRoute, boolean immediateReturn).Το ρομπότ θα ακόλουθη ένα μονοπάτι που αποτελείται από μια σειρά από WayPoint.
- setPathFinder(PathFinder pathFinder). Υποστηρίζονται αλγόριθμοι (ShortestPathFinder και DijkstraPathFinder) για την εύρεση διαδρομής από ένα σημείο αφετηρίας σε ένα σημείο τερματισμού.

Υλοποίηση κλάσης Pilot για το δικό μας ρομπότ

Η LeJOS προσφέρει την κλάση CompassPilot που είναι επέκταση της κλάσης Pilot. Υλοποιεί τις ίδιες μεθόδους που αναφέραμε πιο πάνω αλλά χρησιμοποιώντας ένα αισθητήρα πυξίδα εξασφαλίζει ότι το ρομπότ δεν αποκλίνει από την σωστή γωνία κατεύθυνσης του.Στην αρχή έγινε μετατροπή της κλάσης CompassPilot ώστε να πληροί της απαιτήσεις του δικού μας ρομπότ που οδηγείται από τέσσερα μοτέρ αντί για δύο. Στην συνέχεια εφαρμοστήκαν διάφορες δοκιμές στην κλάση αυτή για να εξακριβωθεί πόσο αποτελεσματικά δουλεύει. Δυστυχώς τα αποτελέσματα που πήραμε δεν ήταν ικανοποιητικά και αποφασίστηκε να χρησιμοποιηθούν άλλες τεχνικές που θα βελτίωναν την πλοήγηση του ρομπότ. Ο κύριος λόγος που τα αποτελέσματα που πήραμε δεν ήταν τόσο καλά είναι γιατί ο αισθητήρας πυξίδα επηρεάζεται από ηλεκτρομαγνητικά πεδία και έτσι σε περιοχές που υπάρχουν μέταλλά ο αισθητήρας μας έδινε λανθασμένες μετρήσεις. Άλλο χαρακτηριστικό που παρατηρήθηκε για τον συγκεκριμένο αισθητήρας πυξίδα είναι ότι για μεγάλες περιστροφές οι μετρήσεις που δίνει είναι πιο ακριβείς ενώ σε μικρές περιστροφές δεν είναι τόσο ακριβής.Σαν αρχική λύση αποφασίστηκε να χρησιμοποιηθεί ένας αισθητήρας γυροσκόπιο για να αντικαταστήσει την πυξίδα. Υλοποιήθηκε ένα μικρό δοκιμαστικό πρόγραμμα που εμφανίζει στην οθόνη του Nxt την τιμή που διαβάζεται από το γυροσκόπιο για να μπορέσουμε να ελέγξουμε την ακρίβεια του αισθητήρα αυτού. Τα συμπεράσματα που πάρθηκαν ήταν ότι το λάθος στις μετρήσεις που παίρνει το γυροσκόπιο μεγαλώνει ανάλογα με τον χρόνο. Με άλλα λόγια ο μόνος τρόπος για να περνούμε πιο ακριβείς μετρήσεις από τον αισθητήρα αυτό ήταν να μηδενίζουμε την τιμή του πριν από κάθε περιστροφή ώστε να είναι ενεργοποιημένος σε μικρό χρονικό διάστημα και να μετρούμε την γωνία περιστροφής όταν το ρομπότ τελειώσει την κίνηση του (Fig.2).Για να μπορέσει το ρομπότ μας να κινείται με μεγαλύτερη ακρίβεια αποφασίστηκε να χρησιμοποιηθούν δύο αισθητήρες. Ένας αισθητήρας πυξίδα και ένα γυροσκόπιο που όπως είδαμε και οι δυο αισθητήρες έχουν τα δικά τους μειονεκτήματα αλλά συνδυάζοντας τους θα έχουμε καλύτερα αποτελέσματα.

```
Public void Rotate(Angle){

      1. Reset Gyroscope Degrees

      2. Perform Rotation with Feedback from encoder (Angle)

      3. Get Rotation Error(Angle)
         !Rotation Error= Get Gyroscope Degrees – Angle

      4. WHILE(Rotation Error >Offset)

          4.1 Reset Gyroscope Degrees
          4.2 Perform Rotation with Feedback from encoder
          4.3 Get Rotation Error(Rotation Error)



      5. Update Robot Heading
}
```

Fig.2. Pseudo code for rotation of robot

Ταξιδεύοντας σε ευθεία και διατήρηση κατεύθυνση του ρομπότ

Το κλειδί για να διατηρηθεί την κατεύθυνσης του ρομπότ και να κινείται προς τα εμπρός είναι να αναγνωρίσουμε ότι θα κινηθεί περίπου ευθεία όταν θα κινήσουμε όλα τα μοτέρ με την ίδια ισχύ. Ο πλοηγός (Pilot) μπορεί να κατευθύνει το ρομπότ αριστερά ή δεξιά καθώς κινείται προς τα εμπρός με την εφαρμογή ελαφρώς περισσότερης ισχύ σε ένα μοτέρ από ότι στο άλλο **(**Fig.3**)**. Όσο μεγαλύτερο είναι το λάθος στην κατεύθυνσης του ρομπότ τότε τόσο πιο μεγάλη θα είναι η διάφορα της ισχύς που πρέπει να εφαρμοστεί στα μοτέρ για να οδηγηθεί το ρομπότ στη αρχική του κατεύθυνσης. Η τεχνική αυτή αποτελεί εφαρμογή του «Proportional control», μια εξαιρετικά ευρέως χρησιμοποιουμένη μέθοδος για έλεγχο δυναμικών συστημάτων. Παίρνει το όνομά της επειδή η έξοδος του ελεγκτή είναι ανάλογη με το λάθος. Σε αυτή την περίπτωση το λάθος είναι η διαφορά της πραγματικής κατεύθυνσης του ρομπότ με την κατεύθυνσης την οποία επιθυμούμε[22].

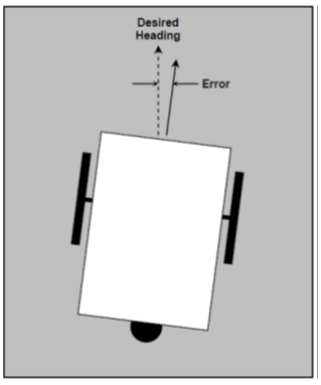
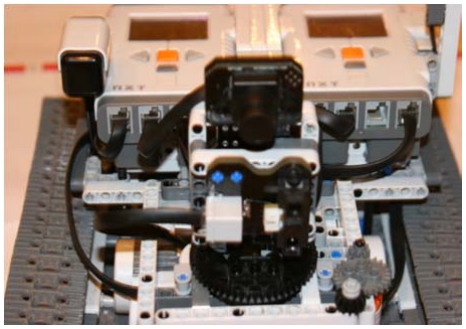


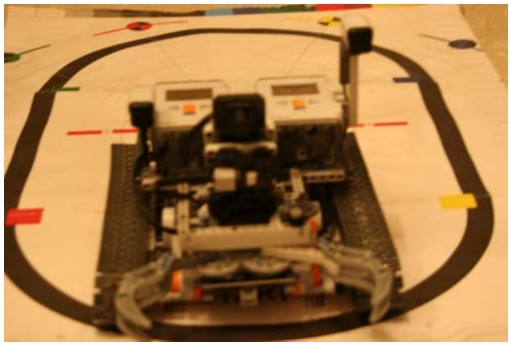
Fig.3. Travelling in Straight Line –Heading

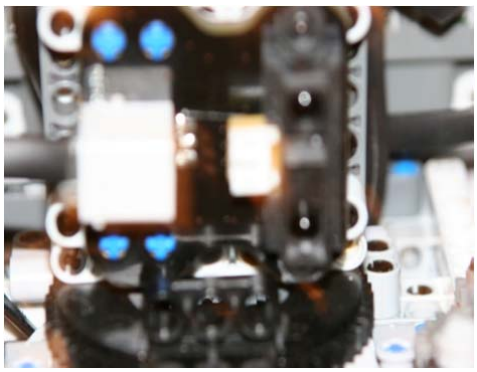
```
1. Compass Set Degrees = Robot Heading

2. Pilot start Travel

3. WHILE( Pilot is Traveling )

    3.1 Get Heading Error
      !Heading Error= Compass Get Degrees- Robot Heading

    3.2 Differential= Gain* Heading Error
     !Where Gain is a constant value that controls how
       aggressively the navigator will respond to error

    3.3 Set Left Motor Power= DrivePower – Differential

    3.4 Set Right Motor Power= DrivePower + Differential
```

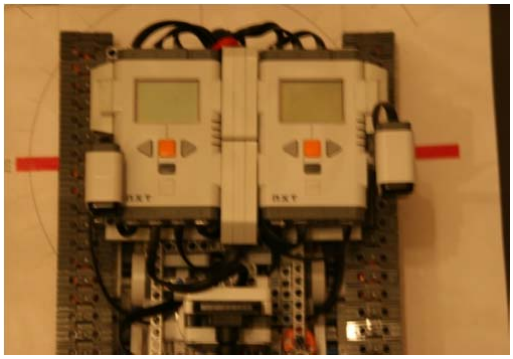
Fig.4. Pseudo code for driving in straight line

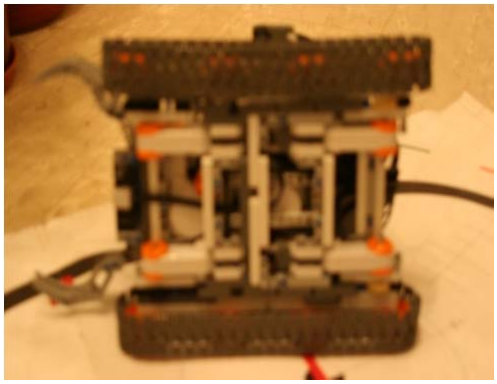
4.5 Φωτογραφίες ρομπότ

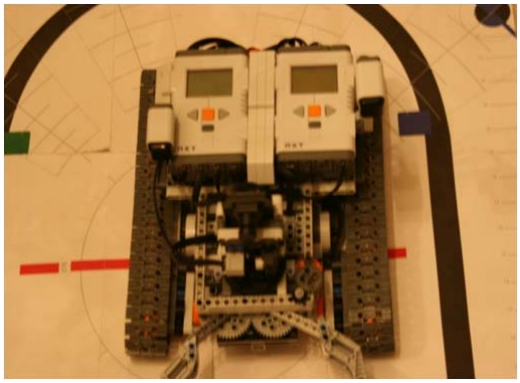












Κεφάλαιο 5

Λειτουργίες και Συμπεριφορές Ρομπότ που Υλοποιήθηκαν

5.1 Έννοια Pid Contoller 33
5.2 Ακολουθία Γραμμής στο πάτωμα 34
5.3 Ακολουθία αντικειμένου με συγκεκριμένο χρώμα 37
5.4 Αποφυγή αντικειμένων όταν το ρομπότ να ακολουθεί τυχαία πορεία 38
5.5 Αποφυγή αντικειμένων όταν το ρομπότ κατευθύνεται σε συγκεκριμένο προορισμό 40
5.6 Αποτελέσματα 42
5.7 Απομακρυσμένος έλεγχος ρομπότ από κινητό τηλέφωνο 43

5.1 Έννοια Pid Contoller

Είναι μια τεχνική για που χρησιμοποιείται ευρέως στα βιομηχανικά συστήματα ελέγχου. Ένας Pid ελεγκτής υπολογίζει μια τιμή "σφάλματος " σαν διαφορά μεταξύ της τιμής που διαβάστηκε από κάποιο αισθητήρα και της επιθυμητής τιμής που πρέπει να έχει το σύστημα. Ο ελεγκτής προσπαθεί να μειώσει αυτό το "σφάλμα " προσαρμόζοντας τις τιμές που διαβάζονται από το σύστημα. Ονομάστηκε έτσι από τις τρις διαφορετικές μαθηματικές εξισώσεις που εφαρμόζοντας στο "σφάλμα ", P - Proportional, I - Integral, D – Derivative(PID). Αυτές οι τρις τιμές μπορούν να ερμηνευθούν σε σχέση με τον χρόνο. Η τιμή της παραμέτρου P εξαρτάται από το παρόν σφάλμα, η παράμετρος I από την συσσώρευση των σφαλμάτων του παρελθόντος και η παράμετρος D είναι μία πρόβλεψη των μελλοντικών σφαλμάτων με βάση τον ρυθμό μεταβολής του σφάλματος. Το σταθμισμένο άθροισμα αυτών των τριών παραμέτρων είναι η έξοδος του ελεγκτή u(t) πού δίνεται από τον αλγόριθμο[23]:

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

Όπου:

Pout: Proportional όρος της εξόδου.

Kp: Proportional gain, παράμετρος για ρύθμιση.

Ki: Integral gain, παράμετρος για ρύθμιση.

Kd: Derivative gain, παράμετρος για ρύθμιση.

e: Error = SP – PV (SP:επιθυμητή τιμή και PV:παροντική τιμή που διαβάστηκε)

t: Χρόνος.

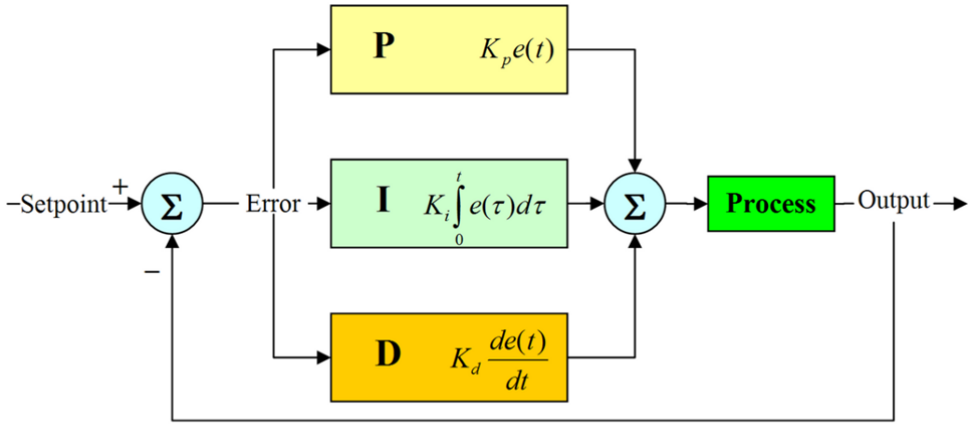


Fig.1. Block diagram of a PID controller

Μερικές εφαρμογές μπορούν να απαιτήσουν τη χρήση μόνο ενός ή δύο ενεργειών για την παροχή του κατάλληλου συστήματος ελέγχου. Αυτό επιτυγχάνεται αναθέτοντας τις υπόλοιπες παραμέτρους σε μηδέν. Ένας ελεγκτής PID θα λέγεται ελεγκτής PI, PD, P ή I σε περίπτωση απουσίας των αντίστοιχων ενεργειών ελέγχου. Οι PI ελεγκτές είναι αρκετά κοινοί, δεδομένου ότι ο όρος Derivative είναι ευαίσθητος στο θόρυβο μέτρησης, ενώ η απουσία του όρου Integral μπορεί να εμποδίσει το σύστημα να φθάσει την τιμή στόχο της

5.2 Ακολουθία Γραμμής στο πάτωμα

Στόχος μας είναι να κάνουμε το ρομπότ να ακολουθήσει μία μαύρη γραμμή. Η ακολουθία γραμμής είναι μια βασική ρομποτική συμπεριφορά και συχνά είναι ένα από τα πρώτα πράγματα που μαθαίνουν οι άνθρωποι που ασχολούνται με τον τομέα αυτό . Μια φορητή συσκευή που μπορεί να ακολουθήσει μια γραμμή έχει όλα τα χαρακτηριστικά ενός πραγματικού ρομπότ. Χρησιμοποιεί αισθητήρες για να συγκεντρώσει πληροφορίες για το περιβάλλον γύρω του και να αλλάζει τη συμπεριφορά του ανάλογα με τις πληροφορίες αυτές. Στο σχήμα φαίνεται η βασική ιδέα ενός ρομπότ που οδηγείται με την μέθοδο της Διαφορικής οδήγησης και θα μπορούσε να ακολουθήσει μια γραμμή που βρίσκεται σε δάπεδο. Αυτό το ρομπότ διαθέτει ένα αισθητήρα φωτός τοποθετημένο στο μπροστινό μέρος ο οποίος δείχνει ευθεία προς τα κάτω ώστε να βλέπει το δάπεδο.

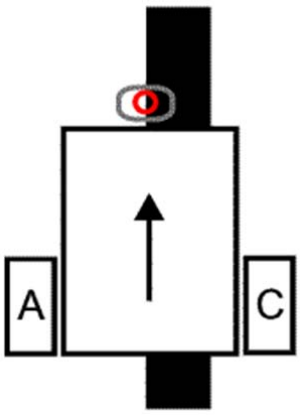


Fig.1. Follow Line Spot

Ο κόκκινος κύκλος που φαίνετε στο σχήμα αντιπροσωπεύει το μικρό σημείο όπου ο αισθητήρας φωτός μπορεί να "δει"[24].

Ρομπότ που ακολουθούν γραμμή μπορούν να υλοποιηθούν χρησιμοποιώντας ένα , δύο ή όσους αισθητήρες φωτός θέλουμε. Γενικά όσο περισσότερους αισθητήρες χρησιμοποιήσουμε τόσο το ρομπότ θα ακολουθεί καλυτέρα την γραμμή. Εδώ θα περιοριστούμε σε ένα αισθητήρας φωτός, που ακόμη και με ένα αισθητήρας το ρομπότ μας θα μπορεί να ακολουθήσει την γραμμή αρκετά καλά ακόμα και όταν έχει καμπύλες.

Για την υλοποίηση αυτής της συμπεριφοράς χρησιμοποιήθηκε ένας ελεγκτής PID**.** Το σφάλμα αντιπροσωπεύει την διαφορά της τιμής του επιπέδου φωτεινότητας που διαβάζεται από τον αισθητήρα φωτός και της επιθυμητής τιμής επιπέδου φωτεινότητας που είναι η τιμή που δίνει ο αισθητήρας όταν βρίσκεται ακριβώς στην μέση του άσπρου δαπέδου και της μαύρης γραμμής (όπως φαίνεται στο σχήμα από τον κόκκινο κύκλο). Ο ελεγκτής προσπαθεί να μειώσει αυτό το "σφάλμα " προσαρμόζοντας τις τιμές που διαβάζονται από τον αισθητήρα κινώντας το ρομπότ δεξιά και αριστερά και το επιτυγχάνει αυτό εφαρμόζοντας διαφορετική ισχύ σε κάθε μοτέρ. Η επιθυμητή τιμή επιπέδου φωτεινότητας βρέθηκε προσθέτοντας την τιμή του αισθητήρα όταν βρίσκεται πάνω από το άσπρο με την τιμή του αισθητήρα όταν βρίσκεται πάνω από την μαύρη γραμμή και διαιρώντας το αποτέλεσμα με το δύο.

Έχουμε Offset= (Black Light Value + White Light Value)/2=(44+12)/2=28.5

Με δοκιμές Trial and error το Offset αποφασίστηκε να είναι 25

```
1. Kp = 15.2  !Tuning parameter for Proportional term
2. Ki = 0.2   !Tuning parameter for Integral term
3. Kd=10 !Tuning parameter for Derivative term
4. offset=25  !Offset for calculating error
5. Tp=300    !Target power for motors
6. integral=0
7. lastError=0
8. derivative=0
9. WHILE( forever)
  9.1    LightValue =read light sensor
  9.2    error= LightValue- offset
  9.3    integral=integral+error
  9.4    derivative=error-lastError
  9.5    Turn=Kp*error +Ki*integral+Kd*derivative
         !The P term, the I term and the D term
  9.6    SET powerLeftMotors=Tp+Turn  !Drive leftMotors
  9.7    SET powerRightMotors=Tp-Turn   !Drive
    rightMotors
```

Fig.2. Pseudo Code of Line Following

Υπάρχουν διαφορές τεχνικές για ρύθμιση του ελεγκτή PID. Η πιο γνώστες είναι Manual Tuning,Ziegler-Nichol,Cohen-Coon όπως επίσης υπάρχουν εργαλεία λογισμικού. Η μέθοδος που χρησιμοποιήθηκε εδώ είναι Manual Tuning και περιγράφεται πιο κάτω:

- Θέτουμε στις παραμέτρους Ki ,Kd τιμή μηδέν. Αυξάνουμε το Kp ώστε όταν το ρομπότ ξεφεύγει από την γραμμή να είναι φανερή η αντίδραση που κάνει για να πάει πίσω. Όταν βρούμε αυτή την τιμή τότε θέτουμε το Kp το μισό της τιμής αυτής.
- Αυξάνουμε το Ki ώστε κάθε offset να είναι σωστό. Ωστόσο μεγάλο Ki θα δώσει αστάθεια.
- Τέλος αυξάνουμε το Kd αν χρειάζεται ώστε το ρομπότ να προσπαθεί να διορθώσει πιο γρήγορα το σφάλμα. Ωστόσο μεγάλο Kd θα προκαλέσει το ρομπότ να αντιδρά υπερβολικά στο σφάλμα.

5.3 Ακολουθία αντικειμένου με συγκεκριμένο χρώμα

Η υλοποίηση της συμπεριφοράς αυτής έγινε με τον αισθητήρα NxtCam η οποία έχει την δυνατότητα να εντοπίζει μοτίβα-χρώματα και να επιστρέφει τις συντεταγμένες τους. Πρώτα ενώθηκε με το πρόγραμμα NxtCamView για να σταλεί το χρώμα-μοτίβο που θέλουμε να εντοπίσουμε με την διαδικασία που περιγράψαμε στην παράγραφο 3.3.Για την υλοποίηση αυτής της συμπεριφοράς χρησιμοποιήθηκε και πάλη ελεγκτής PID**.** Το σφάλμα αντιπροσωπεύει την διαφορά της συντεταγμένης χ του κέντρου του αντικειμένου που διαβάζεται από την NxtCam και της επιθυμητής συντεταγμένης χ που διαβάζεται από τον αισθητήρα όταν το αντικείμενο βρίσκεται ακριβώς μπροστά του. Ο ελεγκτής προσπαθεί να μειώσει αυτό το ‘‘σφάλμα ’’ προσαρμόζοντας τις τιμές που διαβάζονται από τον αισθητήρα κινώντας το ρομπότ μπροστά, δεξιά , αριστερά ακόμα και πίσω όταν το αντικείμενο είναι πολύ κοντά στο ρομπότ. Η επιθυμητή τιμή της συντεταγμένης χ του αντικειμένου βρέθηκε προσθέτοντας την μικρότερη και την μεγαλύτερη τιμή της συντεταγμένης χ που διαβάζεται από την NxtCam και διαιρώντας την με το δύο. Για να βρούμε αυτές τις τιμές κατασκεπάστηκε ένα μικρό πρόγραμμα που τις διαβάζει τις συντεταγμένες όταν κινούμε το αντικείμενο δεξιά και αριστερά της NxtCam. Πήραμε ότι η μικρότερη και η μεγαλύτερη τιμή είναι 12 και 166 αντίστοιχος. Για την κίνηση του ρομπότ χρησιμοποιήθηκε η μέθοδος steer(turnRate) όπου κινεί το ρομπότ σε κυκλική τροχαία. Η παράμετρος turnRate καθορίζει την ακτίνα της διαδρομής. Θετική τιμή σημαίνει ότι το κέντρο του κύκλου είναι στα αριστερά του ρομπότ (έτσι το αριστερό μοτέρ κινεί τον εσωτερικό τροχό). Αρνητική τιμή σημαίνει ότι το αριστερό μοτέρ είναι ο εξωτερικό τροχός. Η απόλυτη τιμή κυμαίνεται μεταξύ 0 και 200, και αυτό καθορίζει την αναλογία της μέσα προς τα έξω ταχύτητας του μοτέρ. Το εξωτερικό μοτέρ λειτουργεί με την καθορισμένη ταχύτητα του ρομπότ ενώ το εσωτερικό μοτέρ επιβραδύνει ώστε το ρομπότ να κάνει στροφή. Για turnRate 0, η αναλογία ταχύτητας είναι 1,0 και το ρομπότ ταξιδεύει σε ευθεία γραμμή. Για turnRate 200, η αναλογία ταχύτητας είναι -1 και το ρομπότ περιστρέφεται στη θέση του.

Ο τύπος είναι: speed Ratio= 100 - abs (turnRate).

Για υπολογισμό του έχουμε Offset= (Minimum Value + Maximum Value)/2=(12+166)/2=89

Το σφάλμα κυμαίνεται από -77 έως 77 .Στην περίπτωση μας για να μπορούμε να χρησιμοποιήσουμε την μέθοδο steer(turnRate) πρέπει να μετατρέπουμε το σφάλμα να κυμαίνεται από -200 έως 200.Αυτό το επιτυγχάνουμε πολλαπλασιάζοντας το με 200/77=2.6 .

Το Kp για τον ελεγκτής PID είναι 2.6

```
10.     Kp = 2.6  !Tuning parameter for Proportional term
11.     Ki = 0.2   !Tuning parameter for Integral term
12.     Kd=2   !Tuning parameter for Derivative term
13.     offset=89  !Offset for calculating error
14.     integral=0
15.     lastError=0
16.     derivative=0
17.  WHILE( forever)
  17.1   Object Rectagle=read maximum rectangle from NxtCam
  17.2  Find coordinate x of Objects Center
        !xcenter= (Object Rectagle.x + Object Rectagle.width)/2
  17.3   error= xcenter - offset
  17.4   integral=integral+error
  17.5   derivative=error-lastError
  17.6   Turn=Kp*error +Ki*integral+Kd*derivative
         !The P term, the I term and the D term
  17.7   If(Turn>200)  !if turn rate >200 or <200 rotate robot a
    bit and continue
              17.7.1      Pilot.rotate(-7,true)
              17.7.2      Else if(Turn<200)
              17.7.3      Pilot.rotate(7,true)
```

Fig.1. Pseudo Code of Object Following

5.4 Αποφυγή αντικειμένων όταν το ρομπότ να ακολουθεί τυχαία πορεία

Η αποφυγή αντικειμένων είναι μια από τις πιο σημαντικές πτυχές της ρομποτικής. Εδώ θα μελετηθεί η πιο απλή μέθοδος που είναι το ρομπότ να μπορεί να αποφύγει τα αντικείμενα ακλουθώντας τυχαία πορεία. Για να μπορεί το ρομπότ να αποφύγει τα αντικείμενα πρώτα πρέπει να μπορεί να τα εντοπίσει και να τα συσχετίσει με την θέση του. Για την ανίχνευση των αντικειμένων χρησιμοποιήθηκε ένας αισθητήρας απόστασης με συνδυασμό με ένα μοτέρ για την υλοποίηση ενός ραντάρ όπου θα μπορεί να καθορίσει αν τυχόν βρίσκεται αντικείμενο αριστερά, μπροστά και δεξιά του. Όταν το ρομπότ γνωρίζει που βρίσκεται ένα αντικείμενο τότε θα μπορεί να κάνει και τις κατάλληλες κινήσεις για να το αποφύγει.

Υλοποιήθηκε η κλάση RotatingEyeScanner (Fig.1) UML Class Datagram όπου παρέχει τις βασικές μεθόδους scanLeft(), scanRight() και scanForward(). Το ραντάρ όταν σαρώνει δεξιά

και αριστερά τότε παίρνει μετρήσεις μόνο εάν βρίσκεται σε ορισμένες μοίρες και αυτές είναι 30˚, 60˚ και 80 ˚ και -30˚, -60˚ και -80 ˚ αντιστοίχως . Αυτό γίνεται για μείωση των δεδομένων που πρέπει να επεξεργαστούν αφού περισσότερες πληροφορίες θα είναι περιττές για την συγκεκριμένη εφαρμογή. Η μέθοδος scanForward() απλά επιστρέφει την απόσταση που διαβάζεται από τον αισθητήρα απόστασης όταν ο αισθητήρας είναι σε γωνία 0˚.

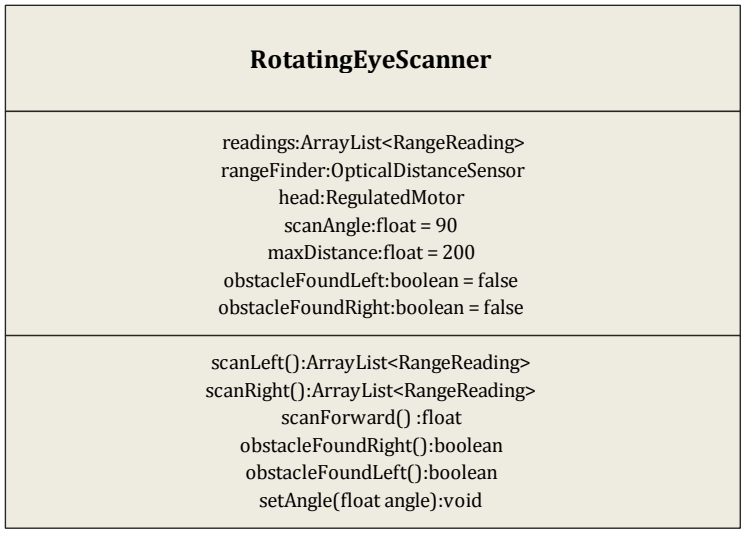


Fig.1. RotatingEyeScanner UML Class Datagram

Ο αλγόριθμος αποφυγής αντικειμένων που υλοποιήθηκε είναι σχετικά απλός. Το ρομπότ ενώσω δεν έχει κάτι μπροστά του κινείται ευθεία ενώ όταν εντοπίσει αντικείμενο σταματά. Στην συνέχεια σαρώνει αριστερά και αν το πεδίο είναι ελεύθερο τότε περιστρέφεται αριστερά και συνεχίζει. Στην περίπτωση που εντοπιστεί και αντικείμενο στα αριστερά τότε σαρώνει και δεξιά για να αποφασίσει πια είναι η κατάλληλη κατεύθυνση να κινηθεί. Για να παρθεί αυτή η απόφαση ελέγχονται οι τρείς μετρήσεις που παίρνονται όταν σαρώνει αριστερά με τις τρεις μετρήσεις που παίρνονται όταν σαρώνει δεξιά και το ρομπότ εκτελεί της ανάλογες κινήσεις για αποφυγή των αντικειμένων. Αυτός ο αλγόριθμός υλοποιήθηκε στην κλάση RandomAvoidance (Fig.2) όπου προσφέρει τις βασικές μεθόδους startAvoiding() και stop().

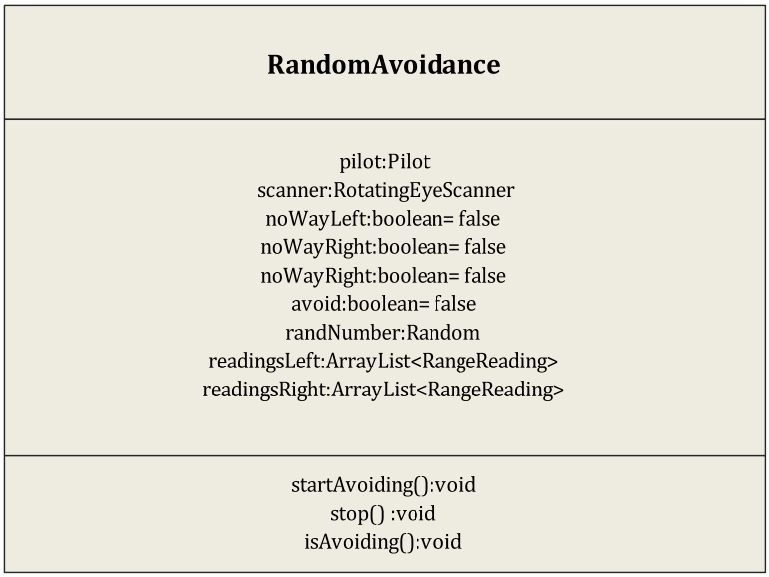


Fig.2. RandomAvoidance UML Class

```
1. WHILE( No Object at front)
     2.1 Drive Forward
2. Scan Left
   3.1  IF (Clear) Rotate Left   !sent command to motors
   3.2  ELSE scan Right          ! to move but continue
       3.1.1 IF(Clear) Rotate Right
             3.1.2.1 ELSE Decide what way is better
```

Fig.3. Obstacle Avoidance Pseudo-Code

5.5 Αποφυγή αντικειμένων όταν το ρομπότ κατευθύνεται σε συγκεκριμένο προορισμό

Σκοπός της λειτουργίας αυτής είναι να δίνεται εντολή στο ρομπότ να κινηθεί σε συγκεκριμένη καρτεσιανή συντεταγμένη χ,ψ και το ρομπότ να το επιτυχαίνει αυτό αποφεύγοντας τα αντικείμενα που βρίσκει στην πορεία του όπως και επίσης να χαρτογραφεί τα αντικείμενα. Για την ανίχνευση των αντικειμένων χρησιμοποιήθηκε ένας αισθητήρας απόστασης με συνδυασμό με ένα μοτέρ για την υλοποίηση ενός ραντάρ όπου θα μπορεί να καθορίσει την καρτεσιανή συντεταγμένη της αρχής του αντικειμένου και την συντεταγμένη του τέλους του αντικειμένου(Fig.1). Πρώτα το ραντάρ σαρώνει από τα αριστερά στα δεξιά 180˚ μοίρες και καταγράφει την γωνία και την απόσταση (Πολικές συντεταγμένες)της αρχής και του τέλους του αντικειμένου που εντοπίζει. Στην συνέχεια οι Πολικές συντεταγμένες

μετατρέπονται σε καρτεσιανές και το αντικείμενο απεικονίζεσαι στον χάρτη του ρομπότ σαν ορθογώνιο. Χρησιμοποιείται ένας αλγόριθμός ShortestPathFinder (που είναι υλοποιημένος από την ομάδα Lejos) όπου υπολογίζει την συντομότερη διαδρομή από ένα σημείο αφετηρίας σε ένα σημείο τερματισμού αποφεύγοντας εμπόδια που παρουσιάζονται έως ένα σύνολο από γραμμές .Το ρομπότ κινείται σε αυτή την διαδρομή ενώσω δεν εντοπίσει άλλο εμπόδιο μπροστά του. Όταν εντοπίσει τότε σταματά, σαρώνει την περιοχή, κάνει ενημέρωση του χάρτη του και συνεχίζει(Fig.2). Για να έχουμε πλήρη εικόνα της χαρτογράφησης των αντικειμένων δημιουργήθηκε ένα πρόγραμμα όπου επικοινωνεί με το ρομπότ και παίρνει τις συντεταγμένες των γραμμών κάθε αντικειμένου όπως και επίσης την συντεταγμένη της θέσης του ρομπότ και τα ζωγραφίζει στην οθόνη

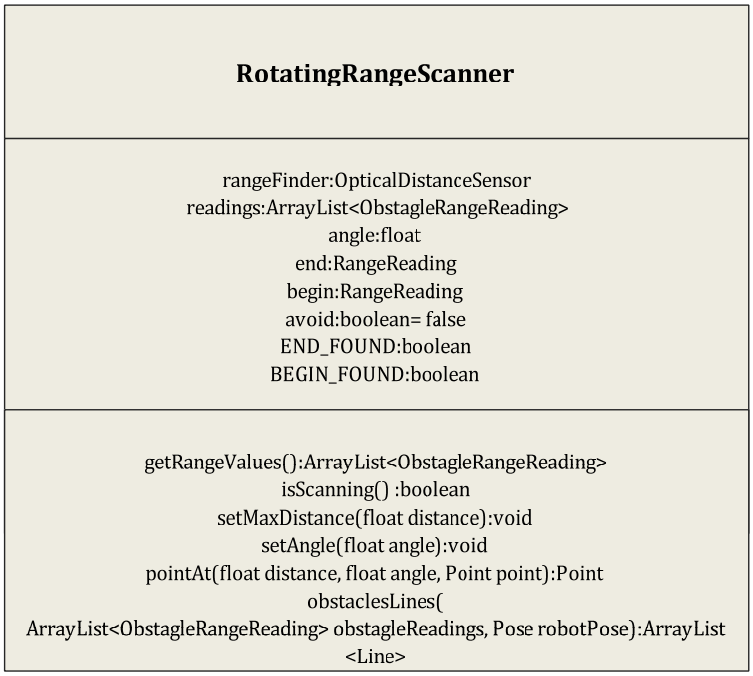


Fig.1. RotatingRangeScanner UML Class Datagram

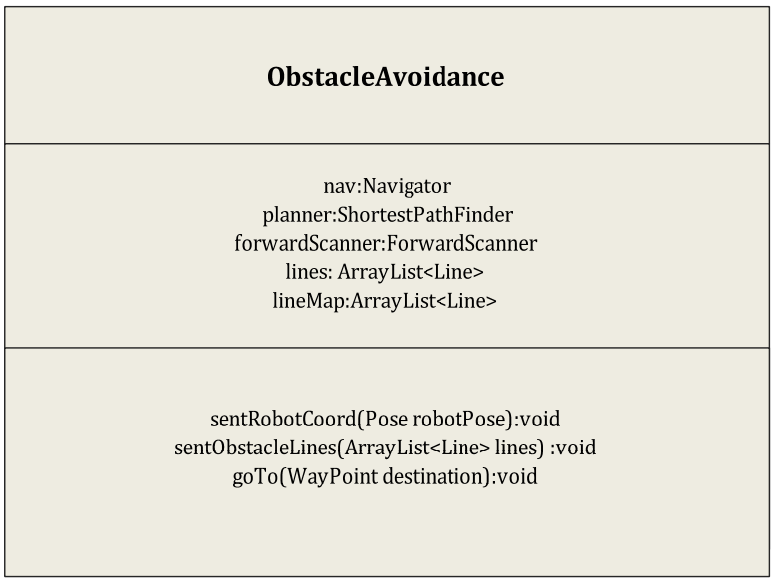


Fig.2. ObstacleAvoidance UML Class Datagram

Μετατροπή Πολικών συντεταγμένων σε Καρτεσιανές συντεταγμένες

Οι δύο Πολικές συντεταγμένες r (απόσταση) και θ (γωνία) μπορούν να μετατραπούν σε καρτεσιανές συντεταγμένες χ και ψ χρησιμοποιώντας τις τριγωνομετρικές εξισώσεις του ημιτόνου και του συνημίτονου(Fig.1)[25]

Για την εύρεση της συντεταγμένης χ:

$$x = r\cos\theta$$

Για την εύρεση της συντεταγμένης ψ:

$$y = r\sin\theta$$

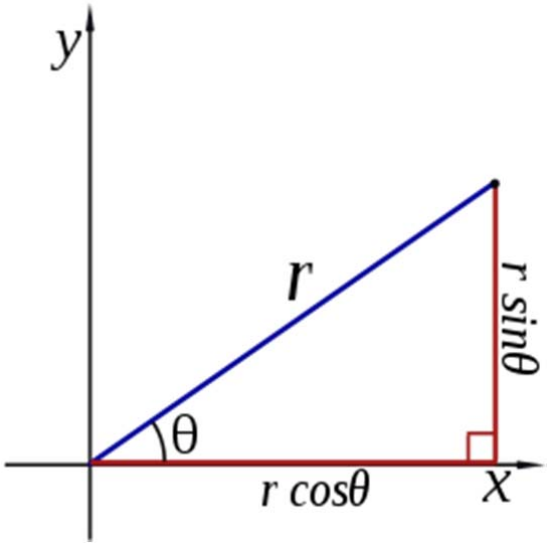


Fig.3. Relationship between polar and Cartesian coordinates

Αποτελέσματα

Τα αποτελέσματα της χαρτογράφησης που πήραμε ήταν αρκετά καλά εκτός μερικές φορές που το ραντάρ έδινε λάθος τιμές της απόστασης του αντικειμένου . Στο Fig.4 φαίνεται το πραγματικό περιβάλλον του ρομπότ και στο Fig.5 η χαρτογράφηση του περιβάλλοντος. Στο Fig.6 φαίνεται το πραγματικό περιβάλλον και στο Fig.7 είναι η χαρτογράφηση των εμποδίων όταν το ρομπότ έχει προορισμό ένα μέτρο ευθεία από την αρχική του θέση. Στην αρχή σαρώνει την περιοχή υπολογίζει μια διαδρομή και κατευθύνεται σε αυτή ενόσω δεν εντοπίσει άλλα αντικείμενα. Στο Fig.7 εντοπίζει και έτσι σταματά ,σαρώνει και συνεχίζει για τον προορισμό του.

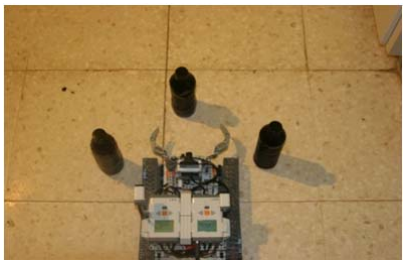


Fig.4. Actual environment

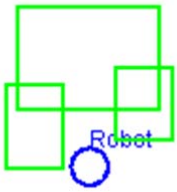


Fig.5. Obstacle map

Fig.6. Actual environment

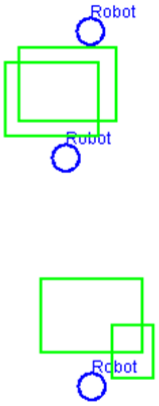


Fig.7. Obstacle map when robot is moving

5.6 Απομακρυσμένος έλεγχος ρομπότ από κινητό τηλέφωνο

Για να δοθεί η δυνατότητα απομακρυσμένου ελέγχου του ρομπότ αποφασίστηκε να χρησιμοποιηθεί σαν τηλεχειριστήριο ένα κινητό τηλέφωνο με λειτουργικό Android OS.
Η εφαρμογή που υλοποιήθηκε παρέχει της εξής λειτουργίες:

1. Απομακρυσμένος έλεγχος του ρομπότ με την χρήση της οθόνης αφής του κινητού τηλεφώνου. Δημιουργήθηκαν δύο φόρμες (Εικόνα 2 και 3) που προσφέρουν διαφορετικό έλεγχο η κάθε μια. Ο χρήστης μπορεί να κινήσει το ρομπότ πατώντας πάνω στην οθόνη του κινητού τηλεφώνου. Στην πρώτη φόρμα (εικόνα 1) ο χρήστης μπορεί να κατευθύνει το ρομπότ πατώντας στην ανάλογη κατεύθυνση και όσο μεγαλύτερη είναι η ακτίνα από το κέντρο του οβάλ τόσο μεγαλύτερη θα είναι η ταχύτητα με την οποία θα κινηθεί το ρομπότ. Στην δεύτερη φόρμα (εικόνα 2) ο χρήστης μπορεί να κατευθύνει το διαφορικό ρομπότ(Differential drive)

εφαρμόζοντας διαφορετικές ταχύτητες στα αριστερά μοτέρ πατώντας στο αριστερό ορθογώνιο και διαφορετικές ταχύτητες στα δεξιά μοτέρ πατώντας στο δεξί ορθογώνιο .

2. Εμφάνιση των προγραμμάτων που υπάρχουν πάνω στις δύο Nxt μονάδες όπως και επίσης η δυνατότητα εκτέλεσης ,τερματισμού και διαγραφής ενός προγράμματος(Εικόνα 4,5και 6).

Fig.1. Connect to Robot

Fig.2. TouchPad control

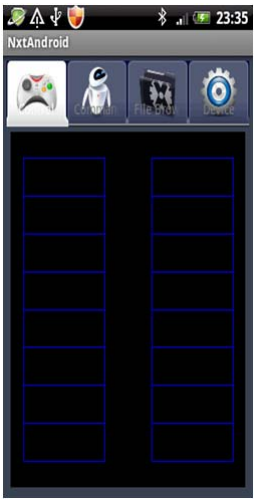


Fig.3. Motor Control

Fig.4. First Nxt File List

Fig.5. Second Nxt File List

Fig.6. Program Options

Κεφάλαιο 6

Γενικά Συμπεράσματα

Στα προηγούμενα κεφάλαια είδαμε τους διάφορους τύπους των ρομπότ που υπάρχουν όπως και τις εφαρμογές που έχει ρομποτική. Όπως αναφέραμε το πρόβλημα Εντοπισμού(Robot Localization) είναι ένα από τα σημαντικότερα προβλήματα για την κατασκευή αληθινών αυτόνομων ρομπότ και για αυτό βρίσκεται στα πιο καυτά θέματα για έρευνα.
Στο δικό μας ρομπότ οι λειτουργίες που υλοποιήθηκαν είχαν ικανοποιητικά αποτελέσματα. Στην ακολουθία γραμμής το ρομπότ όταν ρυθμιζόταν σε χαμηλές ταχύτητες το ρομπότ δεν ξέφευγε από την γραμμή ακόμα και αν είχε καμπύλες. Για την ακολουθία αντικειμένων το ρομπότ μπορούσε να ακολουθεί το αντικείμενο όταν βρισκόταν σε περιβάλλον με καλό φωτισμό. Αυτό συμβαίνει γιατί η NxtCam είναι ευαίσθητη στα επίπεδα φωτεινότητας όπου και αυτό είναι και ένα από τα μειονεκτήματα του αισθητήρα αυτού. Στην χαρτογράφηση των εμποδίων τα προβλήματα που παρουσιάστηκαν είναι σφάλμα στις μετρήσεις που παίρνονταν από τον αισθητήρα απόστασης όπως και επίσης σφάλματα στην θέση και γωνία κατεύθυνσης του ρομπότ που προέρχονταν από σφάλματα στην οδομετρία του ρομπότ.
Η προγραμματιστική γλώσσα Lejos ήταν καλή επιλογή για την υλοποίηση του ρομπότ λόγο του υψηλού επιπέδου βιβλιοθηκών που παρέχει για πλοήγηση όπως και για localization του ρομπότ. Άλλο μεγάλο πλεονέκτημα της Lejos είναι ότι είναι γλώσσα ανοιχτού πηγαίου κώδικα και αυτό ήταν αρκετά χρήσιμο αφού μπορούσαμε να κάνουμε μετατροπές ακόμα και να μάθουμε διαβάζοντας τον κώδικα. Τα κυριότερα μειονεκτήσατε της Lejos είναι ότι δεν υποστηρίζει αποσφαλμάτωση με breakpoints ούτε κάποιο εξομοιωτή για την διευκόλυνση στο γράψιμο ρομποτικών εφαρμογών. Το κόστος αυτό τον δύο είναι να κάνει την ανάπτυξη προγραμμάτων χρονοβόρα

Βιβλιογραφία

[1] Timeline of Robotics. In http://www.thocp.net/reference/robotics/robotics.html

[2] Industrial robot .In http://en.wikipedia.org/wiki/Industrial_robot

[3] Domestic or household robots. In http://www.allonrobots.com/household-robots.html

[4] Medical robots. In http://www.allonrobots.com/medical-robots.html

[5] Military robots. In http://www.allonrobots.com/military-robots.html

[6] Entertainment robots. In http://www.allonrobots.com/types-of-robots.html

[7] Exploration robots. In http://i-heart-robots.blogspot.com/2005/10/robots-for-exploration.html

[8] Rudy Negenbor, Robot Localization and Kalman Filters

[9] The Mindsorms NXT Review. In http://mindstormsnxt.blogspot.com/

[10] Jual Antonio Brena Moral,Develop Lejos Programs Step by Step

[11] Compass Sensor. In http://www.hitechnic.com/

[12] Gyro Sensor. In http://www.hitechnic.com/

[13] Dist-Nx-V3 User Guide. In http://www.mindsensors.com

[14] NxtCam V3Colormap Reference In http://www.mindsensors.com

[15] SumoEyes-User-Guide. In http://www.mindsensors.com

[16] Lego Mindstorms NXT.In http://en.wikipedia.org/wiki/Lego_Mindstorms_NXT

[17] Michael W.Lew ,Using Mindsorms Nxt and lejos in an advance software engineering course

[18] Lejos .In http://lejos.sourceforge.net/

[19] Society of robots. In http://www.societyofrobots.com/programming_differentialdrive.shtml

[20] Dead reckoning. In http://en.wikipedia.org/wiki/Dead_reckoning

[21] Programming Your Robot to Perform Basic Manoeuvres. In http://www.ridgesoft.com/

[22] Programming Your Robot to Navigate. In http://www.ridgesoft.com/