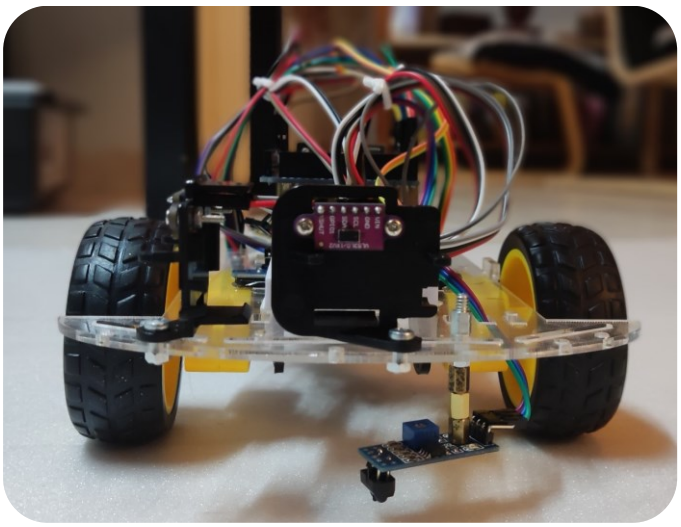
ΔΙΕΘΝΕΣ ΠΑΝΕΠΙΣΤΗΜΙΟ ΤΗΣ ΕΛΛΑΔΟΣ

MSc in ROBOTICS

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«Υλοποίηση αλγορίθμων της οικογένειας BUG-Algorithms σε πραγματικό αυτοκινούμενο ρομπότ για αποφυγή εμποδίων»



Θεοδοσιάδης Ευάγγελος

Αρ.Μητρώου:23

Επιβλέπων καθηγητής:
Δρ. Καζαρλής Σπυρίδων

ΣΕΡΡΕΣ, 2021

Δήλωση

Βεβαιώνω ότι είμαι συγγραφέας αυτής της διπλωματικής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της, είναι πλήρως αναγνωρισμένη και αναφέρεται μέσα σ' αυτήν. Επίσης έχω αναφέρει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επίσης βεβαιώνω ότι αυτή η διπλωματική εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για τις απαιτήσεις του προγράμματος σπουδών του Μεταπτυχιακού «MSc in Robotics» του Τμήματος Μηχανικών Πληροφορικής του Διεθνούς Πανεπιστημίου Ελλάδος.

Ο Δηλών

Θεοδοσιάδης Ευάγγελος

to Hope

Περιεχόμενα

Περίληψη

Στα πλαίσια της παρούσας διπλωματικής εργασίας αναπτύχθηκε ένα αυτόνομο ρομποτικό όχημα αποφυγής εμποδίων. Για την τελική κατασκευή του ρομποτικού οχήματος χρησιμοποιήθηκε η πλατφόρμα Arduino Mega 2560, DC κινητήρες και μια σειρά από διαφορετικούς αισθητήρες. Η ανάπτυξη του κώδικα έγινε με το ελεύθερο λογισμικό "Arduino IDE" και ο αλγόριθμος με τον οποίο έγιναν οι δοκιμές για την αποτελεσματικότητα του, είναι ο Bug-0. Κατά τη διάρκεια της κατασκευής του ρομποτικού οχήματος, χρειάστηκε να δοκιμαστούν και να αντικατασταθούν αρκετά υλικά, με σκοπό τη βελτίωση της κίνησης, της ανίχνευσης εμποδίων και της αντίληψης θέσης.

Abstract

In the context of this thesis, an autonomous robotic obstacle avoidance vehicle was build. For the final development of the robotic vehicle, the Arduino Mega 2560 platform, DC motors and a variety of different sensors were used. The code was developed using the free software "Arduino IDE" and the algorithm that was used to test the effectiveness is Bug-0. During the development of the robotic vehicle, several materials had to be tested and replaced in order to improve the movement, the obstacle detection and position perception.

1. Εισαγωγή

Το πρόβλημα σχεδιασμού διαδρομής είναι γνωστό στη ρομποτική και παίζει σημαντικό ρόλο στην πλοήγηση αυτόνομων ρομποτικών οχημάτων. Η πλοήγηση, η οποία είναι μια διαδικασία σχεδιασμού και καθοδήγησης ενός μονοπατιού, είναι μια εργασία που ένα αυτόνομο ρομπότ πρέπει να εκτελεί σωστά προκειμένου να κινείται με ασφάλεια από ένα σημείο σε ένα άλλο χωρίς να χάνεται ή να συγκρούεται με άλλα αντικείμενα. Τα τρία γενικά προβλήματα της πλοήγησης είναι ο εντοπισμός, ο σχεδιασμός διαδρομής και ο έλεγχος κίνησης. Μεταξύ αυτών των τριών προβλημάτων, μπορεί να θεωρηθεί ότι ο σχεδιασμός διαδρομής είναι ένα από τα πιο σημαντικά ζητήματα στη διαδικασία της πλοήγησης.

Η έρευνα για το σχεδιασμό διαδρομής του αυτόνομου ρομποτικού οχήματος έχει προσελκύσει την προσοχή από τη δεκαετία του 1970. Τα τελευταία χρόνια και πρόσφατα, η έρευνα σε αυτόν τον τομέα έχει αυξηθεί λόγω του ότι τα αυτοκινούμενα ρομπότ χρησιμοποιούνται πλέον σε πάρα πολλές εφαρμογές. Έτσι, αυτά τα ρομπότ πρέπει πλέον να λειτουργούν με περισσότερη ακρίβεια σε διάφορους τομείς. Ο ακριβής σχεδιασμός διαδρομής επιτρέπει στα αυτόνομα ρομπότ να ακολουθούν μια βέλτιστη διαδρομή από τη θέση εκκίνησης έως στη θέση στόχου χωρίς να συγκρούονται με πιθανά εμπόδια που βρίσκονται στο χώρο.

Προκειμένου να απλοποιηθεί το πρόβλημα του σχεδιασμού διαδρομής και να διασφαλιστεί ότι το ρομπότ κινείται ομαλά αποφεύγοντας τα εμπόδια σε ένα ακατάστατο περιβάλλον, ο χώρος διαμόρφωσης πρέπει να ταιριάζει με τον αλγόριθμο που χρησιμοποιείται. Οι αλγόριθμοι που έχουν αναπτυχθεί για τη δημιουργία συστήματος σχεδιασμού διαδρομής σε πραγματικό χρόνο για αυτόνομα ρομπότ είναι πολυάριθμοι. Η επιλογή ενός κατάλληλου αλγορίθμου σε κάθε στάδιο της διαδικασίας σχεδιασμού διαδρομής είναι πολύ σημαντική για να εξασφαλιστεί η ομαλή διεξαγωγή της διαδικασίας πλοήγησης. [1]

2. Αλγόριθμοι BUG

Οι αλγόριθμοι Bug είναι οι απλούστεροι αλγόριθμοι σχεδιασμού διαδρομής που χρησιμοποιούν μόνον εντοπισμό της τρέχουσας θέσης και δεν χρειάζονται χάρτη του χώρου στον οποίο θα κινηθούν. Επομένως, οι αλγόριθμοι αυτοί είναι κατάλληλοι σε περιπτώσεις όπου ο χάρτης του χώρου είναι άγνωστος, ή αλλάζει συχνά και επίσης όταν το αυτοκινούμενο ρομπότ έχει περιορισμένη υπολογιστική ισχύ. Αυτοί οι αλγόριθμοι χρησιμοποιούν πληροφορίες που λαμβάνουν τοπικά από τους αισθητήρες τους (π.χ. αισθητήρας αφής, αισθητήρες απόστασης) και πληροφορίες σχετικά με τη θέση και την κατεύθυνση του στόχου. Οι λειτουργία τους αποτελείται από δύο απλές ενέργειες: πορεία σε ευθεία γραμμή προς το στόχο και ακολούθηση του περιγράμματος του εμποδίου που θα συναντήσουν.

Τα αυτοκινούμενα ρομπότ που χρησιμοποιούν αυτούς τους αλγορίθμους μπορούν να αποφεύγουν τα εμπόδια και να κινούνται προς το στόχο τους. Αυτοί οι αλγόριθμοι απαιτούν μικρή χρήση μνήμης και η διαδρομή που ακολουθείται διαφέρει συνήθως από τη βέλτιστη. Οι αλγόριθμοι Bug πρωτοεμφανίστηκαν και υλοποιήθηκαν από τους Lumelsky και Stepanov [2] και ακολούθησαν αρκετές παραλλαγές και βελτιώσεις. [3] [4]

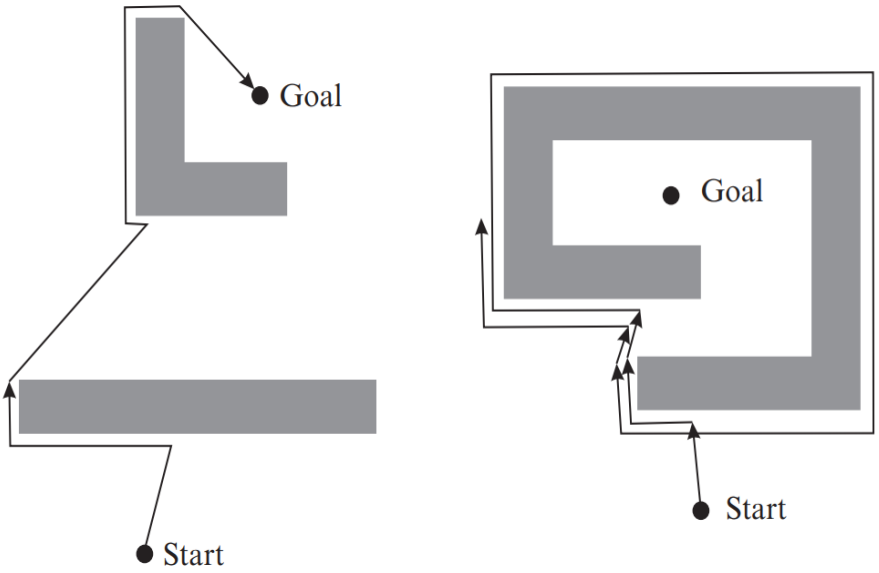
Στη συνέχεια θα περιγράψουμε τέσσερεις βασικές μεθόδους των αλγορίθμων Bug.

2.1. Αλγόριθμος Bug-0

Ο αλγόριθμος Bug-0 έχει δύο βασικές λειτουργίες:

- Μετακινήσου σε ευθεία γραμμή προς το στόχο μέχρι να ανιχνευθεί κάποιο εμπόδιο ή να φθάσεις στο στόχο. Στην περίπτωση που φθάσεις στο στόχο η αναζήτηση διαδρομής ολοκληρώνεται και ο αλγόριθμος τερματίζει.
- Εάν ανιχνευθεί εμπόδιο, τότε στρίψε αριστερά (ή δεξιά, αλλά πάντα προς την ίδια κατεύθυνση κάθε φορά) και ακολούθησε το περίγραμμα του εμποδίου μέχρι η κίνηση σε ευθεία γραμμή προς το στόχο να είναι και πάλι δυνατή.

Ένα παράδειγμα της εκτέλεσης του αλγορίθμου Bug-0 βλέπουμε στην εικόνα 1. Ο αλγόριθμος Bug-0 βρίσκει με επιτυχία μια διαδρομή προς το στόχο, στο σχήμα αριστερά, ενώ στο σχήμα δεξιά καταλήγει σε βρόχο. [5]



Εικόνα 1: Αλγόριθμος Bug-0

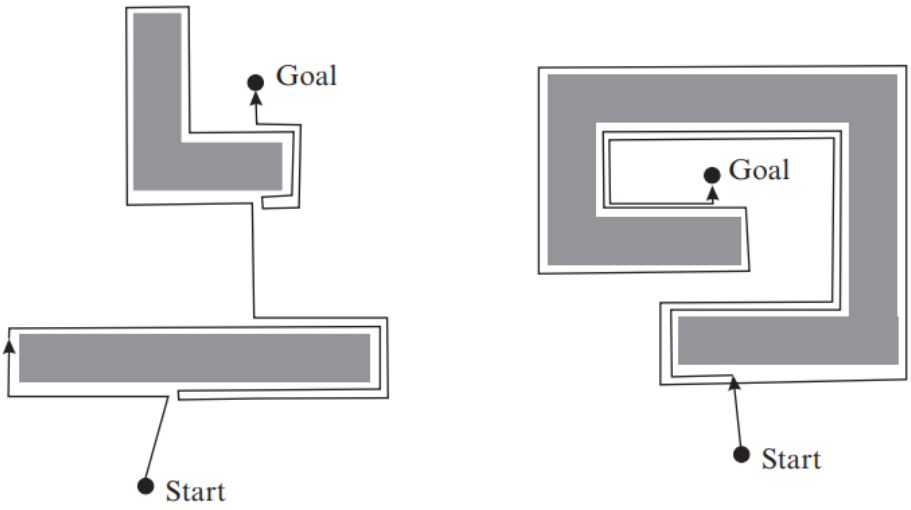
2.2. Αλγόριθμος Bug-1

Ο αλγόριθμος Bug-1 σε σύγκριση με τον Bug-0 χρησιμοποιεί περισσότερη μνήμη και χρειάζεται να εκτελέσει περισσότερους υπολογισμούς. Σε κάθε επανάληψη πρέπει να υπολογίζει την απόσταση από το στόχο και να θυμάται το πλησιέστερο σημείο στο περίγραμμα του εμποδίου προς το στόχο.

Η λειτουργία του περιγράφεται ως εξής:

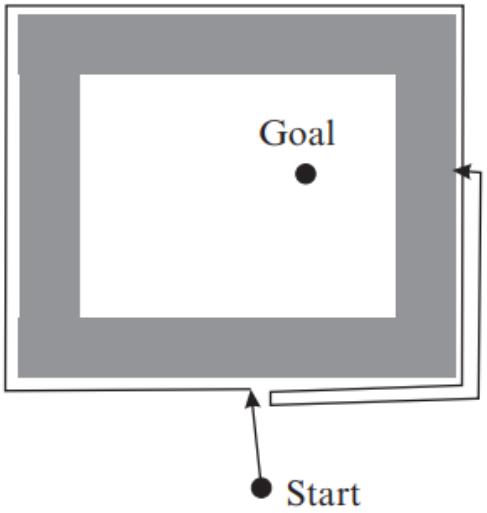
- Μετακινήσου σε ευθεία γραμμή προς το στόχο μέχρι να ανιχνευθεί κάποιο εμπόδιο ή να φτάσει στο στόχο. Στην περίπτωση που φθάσεις στο στόχο η αναζήτηση διαδρομής ολοκληρώνεται και ο αλγόριθμος τερματίζει.
- Εάν ανιχνευθεί εμπόδιο, τότε στρίψε αριστερά και ακολούθησε ολόκληρο το περίγραμμα του εμποδίου μετρώντας ταυτόχρονα και την απόσταση από το στόχο. Όταν φτάσεις ξανά στο σημείο όπου αρχικά ανιχνεύθηκε το εμπόδιο, ακολουθήσε το περίγραμμα του εμποδίου προς την κατεύθυνση που είναι η συντομότερη προς το σημείο του περιγράμματος που βρίσκεται πιο κοντά στο στόχο. Στη συνέχεια, συνέχισε να κινείσαι προς το στόχο σε ευθεία γραμμή.

Ένα παράδειγμα εκτέλεσης του αλγορίθμου Bug-1 δίνεται στην εικόνα 2. Ο αλγόριθμος Bug-1 βρίσκει το μονοπάτι προς το στόχο αν αυτό υπάρχει, όπως φαίνεται στην παρακάτω εικόνα. [5]



Εικόνα 2: Αλγόριθμος Bug-1

Ο αλγόριθμος Bug-1 θα μπορούσε να εντοπίζει και το αν η διαδρομή προς το στόχο δεν είναι εφικτή. (εικόνα 3) [5]



Εικόνα 3: Αλγόριθμος Bug-1 (χωρίς λύση)

Είναι ένας πλήρης αλγόριθμος, δηλαδή σε πεπερασμένο χρόνο, βρίσκει μια διαδρομή εάν υπάρχει ή τελειώνει με αποτυχία αν δεν υπάρχει.

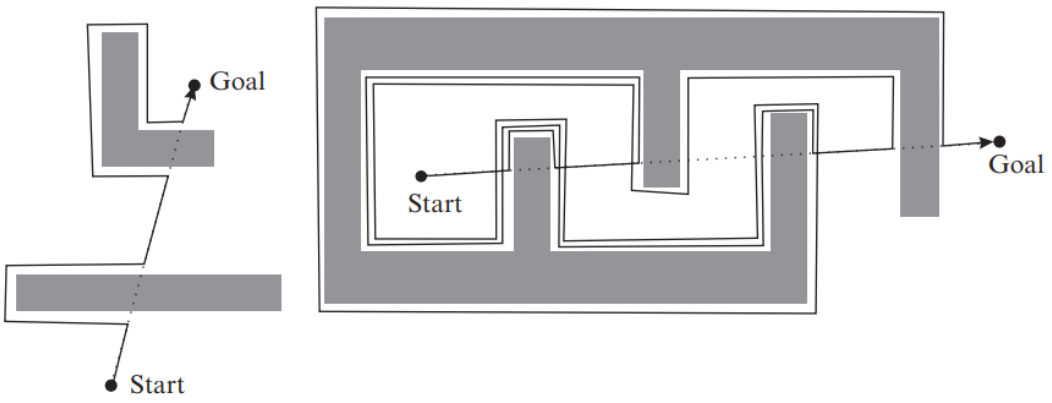
2.3. Αλγόριθμος Bug-2

Ο αλγόριθμος Bug-2 είναι τροποποιημένη μορφή του αλγορίθμου Bug-1. Αντί για την αναζήτηση του πλησιέστερου σημείου απόστασης προς το στόχο, ο αλγόριθμος Bug 2 επικεντρώνεται στη διατήρηση της κατεύθυνσης κίνησης προς το στόχο Ο αλγόριθμος Bug-2 κινείται πάντα στην κύρια γραμμή που έχει οριστεί ως η ευθεία γραμμή που συνδέει το σημείο εκκίνησης και το σημείο στόχου. Η γραμμή αυτή ονομάζεται m-line.

Η λειτουργία του περιγράφεται ως εξής:

- Μετακινήσου στη γραμμή m-line μέχρι να ανιχνευθεί κάποιο εμπόδιο ή να φτάσεις στο στόχο. Στην περίπτωση που φθάσεις στο στόχο η αναζήτηση διαδρομής ολοκληρώνεται και ο αλγόριθμος τερματίζει.
- Εάν ανιχνευθεί εμπόδιο, τότε στρίψε αριστερά (ή δεξιά) και ακολούθησε το περίγραμμα του εμποδίου μέχρι να φτάσεις στη γραμμή m-line, όπου η απόσταση από το σημείο στόχου είναι μικρότερη από την απόσταση από το σημείο όπου εντοπίστηκε για πρώτη φορά το εμπόδιο.

Αν και ο αλγόριθμος Bug-2 φαίνεται γενικά πολύ πιο αποτελεσματικός από τον Bug-1 (βλέπε αριστερό τμήμα της εικόνας 4), δεν εγγυάται ότι το ρομπότ θα ανιχνεύσει κάποια εμπόδια μόνο μία φορά. Σε ορισμένες διατάξεις εμποδίων το ρομπότ με τον αλγόριθμο Bug-2 θα μπορούσε να κάνει περιττούς κύκλους γύρω από τα εμπόδια μέχρι να φτάσει στο στόχο, όπως φαίνεται στο δεξί σχήμα της εικόνας 4. [5]

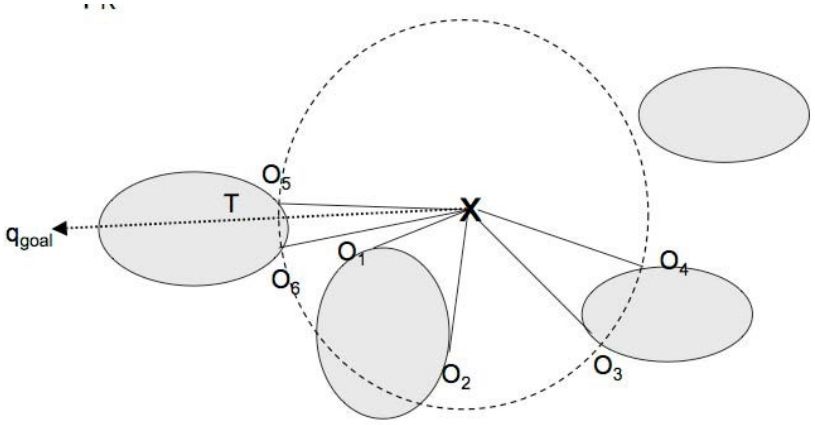


Εικόνα 4: Αλγόριθμος Bug-2

2.4. Αλγόριθμος Tangent Bug

Ο αλγόριθμος Tangent Bug θεωρεί ότι υπάρχει ελεύθερο μονοπάτι χωρίς εμπόδια προς τον τελικό στόχο. Απαιτείται η χρήση αισθητήρα απόστασης μεγαλύτερου βεληνεκούς (πχ Laser, Lidar) απ' ότι χρειαζόμασταν για τους προηγούμενους αλγόριθμους.

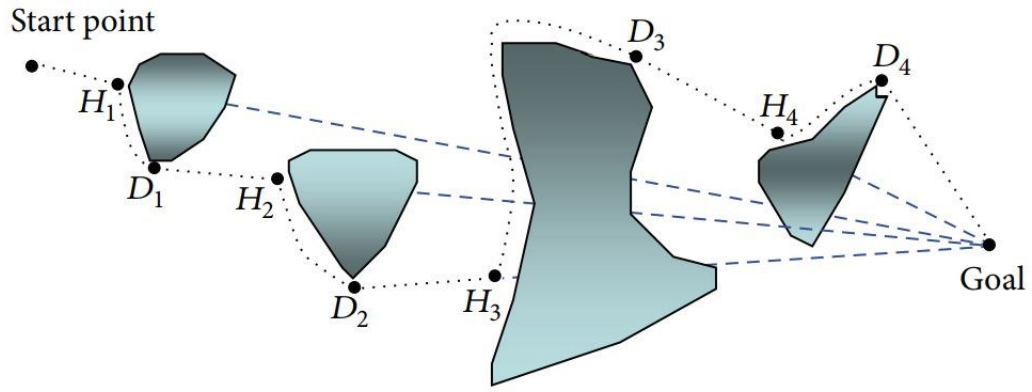
Η λειτουργία του βασίζεται στην εύρεση των τελικών σημείων O_i πεπερασμένων συνεχών τμημάτων, στα όρια των εμποδίων. Επιλέγεται το σημείο-στόχος O_i που ελαχιστοποιεί το $X \rightarrow O_i + Oi \rightarrow q_{goal}$ (εικόνα 5)



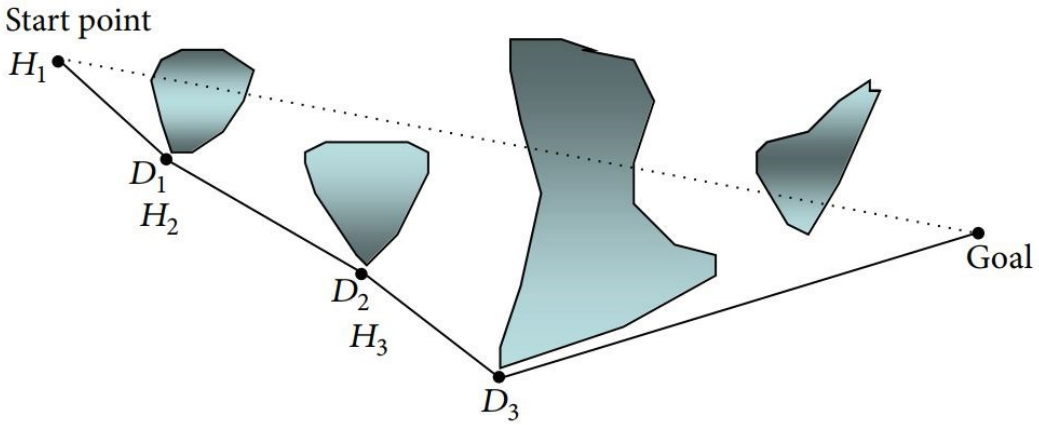
Εικόνα 5: Tangent Bug - Τελικά σημεία O_i

Στους τρεις πρώτους αλγορίθμους το ρομπότ διαθέτει μόνο αισθητήρα αφής ή αισθητήρα περιορισμένης εμβέλειας. Για την εφαρμογή αυτού του αλγορίθμου το ρομπότ θα πρέπει να διαθέτει αισθητήρα μεγάλης εμβέλειας για να ανιχνεύει τα εμπόδια γύρω του. Το ρομπότ κινείται προς το στόχο μέχρι να ανιχνεύσει κάποιο εμπόδιο. Στη συνέχεια κατευθύνεται προς το σημείο του εμποδίου που έχει το μικρότερο δυνατό ευρετικό κόστος. Η συνάρτηση ευρετικού κόστους είναι η απόσταση μεταξύ του ρομπότ και του σημείου που ανιχνεύεται στο εμπόδιο συν την απόσταση μεταξύ του σημείου που ανιχνεύεται στο εμπόδιο και του στόχου. Εάν η ευρετική συνάρτηση κόστους αρχίσει να γίνεται μεγαλύτερη, το ρομπότ αρχίζει να ακολουθεί το περίγραμμα του εμποδίου. Η λειτουργία παρακολούθησης συνεχίζεται μέχρι η απόσταση του ρομπότ από το στόχο να είναι μικρότερη από την απόσταση μεταξύ του σημείου που ξεκινά η λειτουργία παρακολούθησης και του στόχου. Ο αλγόριθμος τερματίζεται όταν φτάσουμε στο στόχο.

Στα παρακάτω σχήματα βλέπουμε την εκτέλεση του αλγορίθμου Tangent Bug με μηδενική εμβέλεια ανίχνευσης (χωρίς αισθητήρα απόστασης) και με άπειρη. (εικόνες 6,7) [6]



Εικόνα 6: Tangent bug με μηδενική εμβέλεια ανίχνευσης



Εικόνα 7: Εικόνα 7: Tangent bug με άπειρη εμβέλεια ανίχνευσης

2.5. Συμπεράσματα

Συγκρίνοντας τους αλγορίθμους μπορούμε να συμπεράνουμε τα εξής:

- Ο Bug-1 είναι ο πιο εξαντλητικός αλγόριθμος αναζήτησης, επειδή υπολογίζει όλες τις πιθανότητες πριν καταλήξει σε μια επιλογή.
- Ο Bug-2 είναι άπληστος αλγόριθμος, επειδή επιλέγει την πρώτη επιλογή που φαίνεται καλύτερη.
- Ο Bug-2 είναι απλός, και εύκολος να κατανοηθεί.
- Στις περισσότερες περιπτώσεις ο αλγόριθμος Bug-2 είναι πιο αποτελεσματικός από τον Bug-1, ωστόσο η λειτουργία του αλγορίθμου Bug-1 είναι ευκολότερο να προβλεφθεί.
- Ο αλγόριθμος Bug-1 δεν είναι πολύ αποδοτικός ως προς το μέγεθος του μονοπατιού που ακολουθεί το ρομπότ, αφού χρειάζεται πρώτα να κάνει μια

πλήρη περιστροφή από κάθε εμπόδιο, γεγονός που επιφέρει στη σχεδίαση πολύ μεγάλων αποστάσεων μέχρι το ρομπότ να καταφέρει να φτάσει στο τελικό του προορισμό. Αν και μπορεί να μην είναι τόσο αποδοτικός ο αλγόριθμος Bug-1, εγγυάται όμως σίγουρο τερματισμό.

- Στη συνέχεια ο αλγόριθμος Bug-2, βελτιώνει αυτό το μειονέκτημα του Bug-1 αφού με την προσθήκη της m-line το αυτοκινούμενο ρομπότ δεν χρειάζεται να κάνει κάθε φορά μια πλήρη περιστροφή γύρω από κάθε εμπόδιο.
- Ο Bug-2 όμως παρουσιάζει σαν μειονέκτημα τη δημιουργία ίδιων μονοπατιών σε μερικές περιπτώσεις.
- Ο Tangent Bug είναι πιο πλήρες όταν υποστηρίζεται από αισθητήρες απόστασης μεγάλου βεληνεκούς. Προσπαθεί να ελαχιστοποιήσει την ευρετική απόσταση έτσι ώστε το ρομπότ να μετακινηθεί ελάχιστα για να φτάσει στην επιθυμητή θέση.

3. Εισαγωγή στο Arduino

Το Arduino είναι μία πλατφόρμα ενός προγραμματιζόμενου μικροελεγκτή του ATmega328P της εταιρείας Atmel, προσαρμοσμένο σε μια πλακέτα και είναι ανοιχτού κώδικα. Είναι στην ουσία ένας μίνι υπολογιστής του οποίου η χρήση είναι να ελέγχει άλλες συσκευές εισόδου – εξόδου με τον κατάλληλο προγραμματισμό. Ο προγραμματισμός του γίνεται μέσω του λογισμικού Arduino IDE το οποίο χρησιμοποιείται για την εγγραφή, τη μετάφραση αλλά και τη μεταφόρτωση των προγραμμάτων. Το κόστος του είναι χαμηλό σε σχέση με άλλους μικροελεγκτές (περίπου 20€).

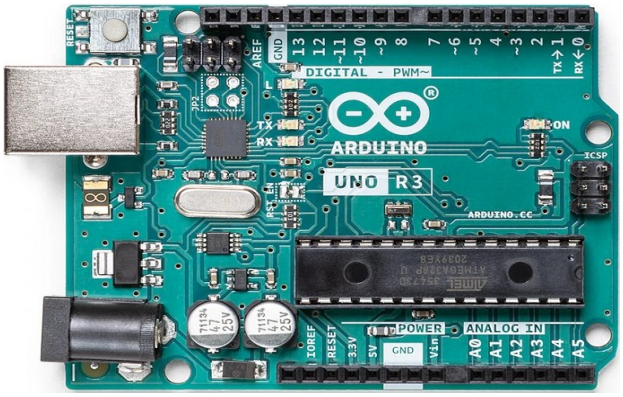
Οι διαφορές της πλατφόρμας με έναν κοινό υπολογιστή είναι:

- Δεν διαθέτει λειτουργικό σύστημα
- Τρέχει συνεχώς ένα πρόγραμμα τη φορά.
- Όταν ξεκινήσει η λειτουργία του εκτελεί το πρόγραμμα, το οποίο αποθηκεύεται μόνιμα μέσα στη μνήμη flash.
- Οι πλακέτες Arduino καταναλώνουν ελάχιστη ενέργεια.
- Είναι πολύ φθηνότερο από έναν υπολογιστή.
- Διαθέτει ακροδέκτες που μπορούν να χρησιμοποιηθούν σαν είσοδοι και έξοδοι για να μπορούν να συνδεθούν διάφορες συσκευές όπως διάφοροι αισθητήρες, διακόπτες, οθόνες LED, κινητήρες κ.α..
- Για τον προγραμματισμό του Arduino θα πρέπει να αναπτυχθεί κώδικας στον υπολογιστή και να μεταφορτωθεί μέσω της θύρας USB.

Υπάρχουν δεκάδες διαφορετικές πλακέτες Arduino με κυριότερη διαφορά στη μνήμη και το πλήθος εισόδων/εξόδων. Στη συνέχεια θα δούμε μερικές από τις πιο βασικές και γνωστές πλακέτες Arduino. [7]

3.1. Arduino UNO Rev3[1]

Ο πιο δημοφιλής τύπος Arduino είναι ο Uno Rev3 που φαίνεται στην εικόνα 8.



Εικόνα 8: Arduino Uno R3

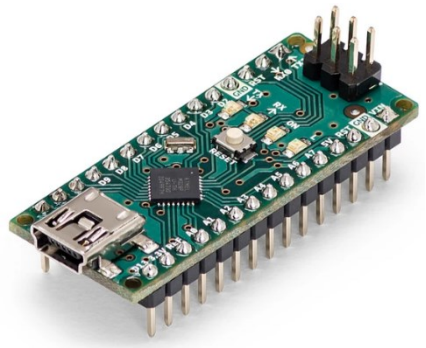
Διαθέτει 14 ψηφιακά pins εισόδου/εξόδου, 6 από τα οποία έχουν δυνατότητα εξόδου σήματος PWM (Pulse Width Modulation). Έχει επίσης 6 αναλογικά pins εισόδου. Η χωρητικότητα της flash μνήμης είναι 32 KB. Υπάρχει επίσης μια μικρή στατική RAM 2 KB.

Χαρακτηριστικά :

- Τάση λειτουργίας 5V (7-12V)
- Επεξεργαστής Atmega328P 16MHz
- 2kBμνήμη RAM
- 32 kB μνήμη Flash
- 14 Ψηφιακά και 6 Αναλογικά I/O Pins
- USB θύρα επικοινωνίας
- Διαστάσεις: 68.6mm x 53.4mm
- Βάρος 25gr

[1] https://store.arduino.cc/collections/boards/products/arduino-uno-rev3

3.2. Arduino NANO[2]



Εικόνα 9: Arduino Nano

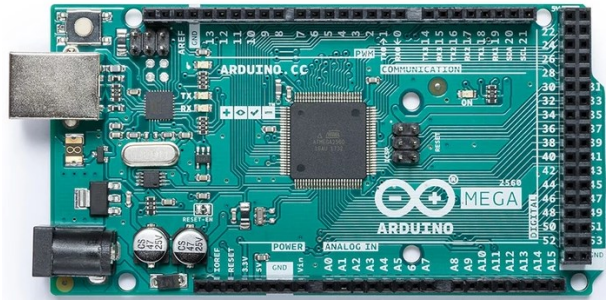
Το Arduino Nano, είναι ένα μικρότερο Arduino που μπορεί να χρησιμοποιηθεί όταν ο χώρος είναι περιορισμένος. Παρά το μικρό του μέγεθος, διαθέτει 14 ψηφιακά pins εισόδου/εξόδου, 6 από τα οποία έχουν δυνατότητα εξόδου σήματος PWM (Pulse Width Modulation), όπως ακριβώς και με το Uno Rev3. Το Nano διαθέτει επίσης 8 αναλογικά pins εισόδου, 2 περισσότερα από το Uno! Η χωρητικότητα της flash μνήμης είναι 32 KB. Υπάρχει επίσης μια μικρή στατική RAM 2 KB.

Χαρακτηριστικά :

- Τάση λειτουργίας 5V
- Επεξεργαστής Atmega328P 16MHz
- 2kBμνήμη RAM
- 32 kB μνήμη Flash
- 14 Ψηφιακά και 8 Αναλογικά I/O Pins
- USB θύρα επικοινωνίας
- Διαστάσεις: 45 mm ×18 mm
- Βάρος 7gr

[2] https://store.arduino.cc/collections/boards/products/arduino-nano

3.3. Arduino Mega2560[3]



Εικόνα 10: Arduino Mega

Το Arduino Mega2560 είναι ο μεγαλύτερος αδελφός του Uno. Διαθέτει περισσότερα Pins και από τα δυο προηγούμενα. 54 ψηφιακά pins εισόδου/εξόδου και 15 από αυτά έχουν δυνατότητα εξόδου σήματος PWM. Έχει επίσης 16 αναλογικά pins εισόδου. Το Mega2660 βασίζεται στο μικροελεγκτή ATmega2560 και διαθέτει 256KB μνήμης flash για την υποστήριξη πολύ μεγαλύτερων προγραμμάτων από ό, τι μπορεί το Uno. Έχει επίσης 8KB SRAM και όπως το Uno και το Nano λειτουργεί στα 16 Megahertz.

Χαρακτηριστικά :

- Τάση λειτουργίας 5V (7-12V)
- Επεξεργαστής Atmega2560 16MHz
- 8kBμνήμη RAM
- 256 KB μνήμη Flash
- 54 Ψηφιακά και 6 Αναλογικά I/O Pins
- USB θύρα επικοινωνίας
- Διαστάσεις: 101.6 mm ×53.3 mm
- Βάρος 37gr

Όπως μπορούμε να δούμε, υπάρχει μια ποικιλία από Arduino για να διαλέξουμε. Καθώς μοιράζονται μια κοινή βάση κώδικα, είναι δυνατό να αναπτυχθεί ένα πρόγραμμα σε μια πλακέτα και στη συνέχεια να μεταφερθεί σε μια άλλη χωρίς πρόβλημα. Αυτό συνέβη και στην παρούσα διπλωματική, καθώς χρειάστηκε να μεταβούμε από την πλακέτα Uno στη Mega, λόγω έλλειψης χώρου μνήμης. Αυτό καθιστά τη σειρά μικροελεγκτών Arduino μία από τις πιο ευέλικτες διαθέσιμες συσκευές.

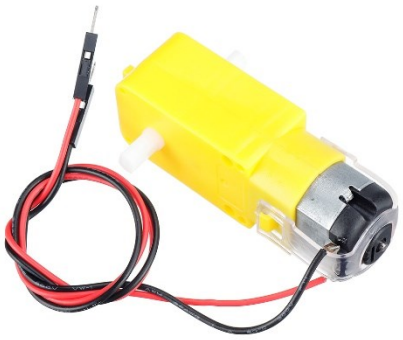
[3] https://store.arduino.cc/products/arduino-mega-2560-rev3

4. Κινητήρες, Ελεγκτής, Τροφοδοσία

4.1. Dc motors

Χαρακτηριστικά:

- Τάση λειτουργίας: 3V - 12VDC
- Μέγιστη ροπή: 0.8 kg.cm
- Σχέση γραναζιών: 1:48
- Ταχύτητα περιστροφής: 200 rpm +/- 10%
- Ρεύμα: τυπικό 150mA +/- 10%
- Μέγεθος: 70x22x18 mm
- Βάρος 29gr



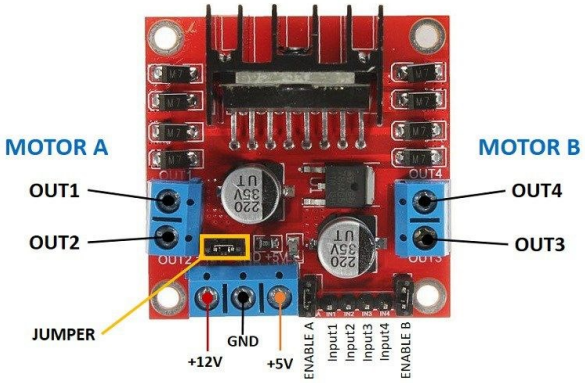
Εικόνα 11: DC κινητήρας

Οι κινητήρες θα οδηγούν τροχούς διαμέτρου 66 mm και πάχους 27mm με πέλμα καουτσούκ για καλύτερη πρόσφυση στο έδαφος.

Εικόνα 12: Τροχοί

4.2. Ελεγκτής κινητήρων L298N (Dual H-Bridge)

Το L298N Dual H-Bridge είναι ένας ελεγκτής που μας επιτρέπει να ελέγχουμε δύο ανεξάρτητους DC κινητήρες χρησιμοποιώντας ψηφιακά λογικά σήματα 5V από τα ψηφιακά pins εξόδου ενός Arduino.



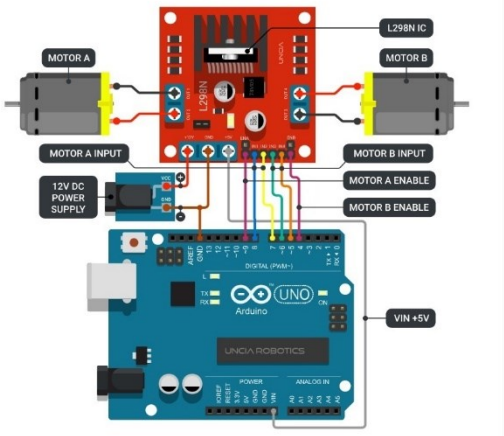
Εικόνα 13: L298N - Dual H-Bridge

Για τη σύνδεση των κινητήρων χρειαζόμαστε τρία pins για τον καθένα, συνολικά έξι δηλαδή. Έχουμε τα pin ENA και ENB, από όπου πρέπει να αφαιρέσουμε τα jumper για μπορέσουμε να ρυθμίσουμε την ταχύτητα των κινητήρων A και B αντίστοιχα, IN1, IN2 για το χειρισμό του A κινητήρα και IN3, IN4 για το χειρισμό του B κινητήρα (εικόνα 13). Για τη ρύθμιση της ταχύτητας εφαρμόζουμε τιμές από 0 (0%) έως 255(100%). Επίσης, υπάρχουν pins για την τροφοδοσία +12V και τη γείωση GND, ενώ υποστηρίζετε και παροχή σταθεροποιημένης τάσης +5V την οποία συνδέουμε στο +5V στο pin του Arduino. Τέλος, στο πλάι, αριστερά και δεξιά έχουμε δύο υποδοχές για κάθε κινητήρα, τις MOTOR A και MOTOR B αντίστοιχα.

Στον παρακάτω πίνακα μπορούμε να δούμε τη συμπεριφορά του κινητήρα ανάλογα με τα σήματα που εφαρμόζονται στις εισόδου του ελεγκτή.

ENA	IN1	IN2	DC Motor
0	X	X	off
1	0	0	brake
1	0	1	forward
1	1	0	reverse
1	1	1	brake

Το 0 αντιστοιχεί LOW κατάσταση και το 1 σε HIGH



Εικόνα 14:Συνδεσμολογία με L298N & Arduino

4.3. Μπαταρία

Rhino 1350mAh 3S 11.1v 25C Lipoly Pack

Χαρακτηριστικά:

- Capacity : **1350mAh**
- Constant discharge: **25C**
- Burst rate: **37C (15sec)**
- Configuration : **3S 11.1v**
- Pack size: **77x35x21mm**
- Weight : **126g**
- Discharge plug: **XT60**

Εικόνα 15: LiPo Battery

5. Αισθητήρες

5.1. Αισθητήρας απόστασης (υπερήχων)HC-SR04

Εικόνα 16: Αισθητήρας υπερήχων HC-SR04

Ο HC-SR04 είναι ένας αισθητήρας απόστασης χαμηλής ακρίβειας που βασίζεται σε υπερήχους. Μπορεί να μετράει αποστάσεις αλλά συνήθως χρησιμοποιείται για την ανίχνευση εμποδίων.

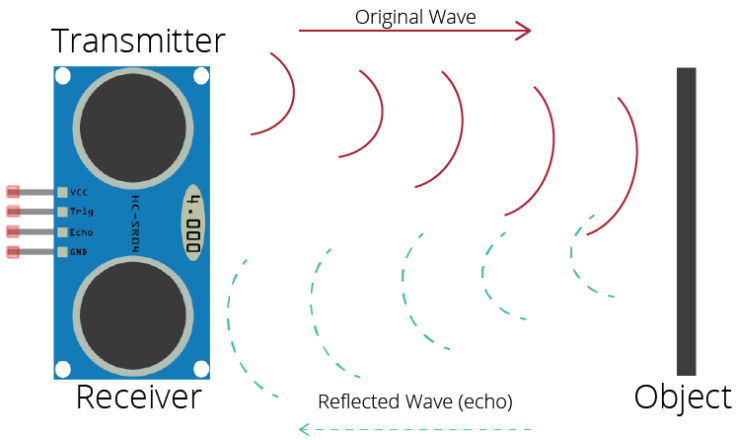
Η λειτουργία του είναι η εξής: (εικόνα 17)

1. Ο αισθητήρας εκπέμπει έναν υπέρηχο.
2. Ο υπέρηχος ταξιδεύει μέσα στον αέρα.
3. Μόλις συναντήσει ένα αντικείμενο, ανακλάται πίσω στον αισθητήρα.
4. Ο δέκτης υπερήχων λαμβάνει τον ανακλώμενο ήχο (ηχώ).

Ο χρόνος μεταξύ της εκπομπής και της λήψης του υπέρηχου μας επιτρέπει να υπολογίσουμε την απόσταση από το αντικείμενο. Καθώς γνωρίζουμε την ταχύτητα του ήχου στον αέρα (=343m/s), η απόσταση από το αντικείμενο θα είναι:

$$απόσταση = (ταχύτητα\ ήχου * χρόνος)/2$$

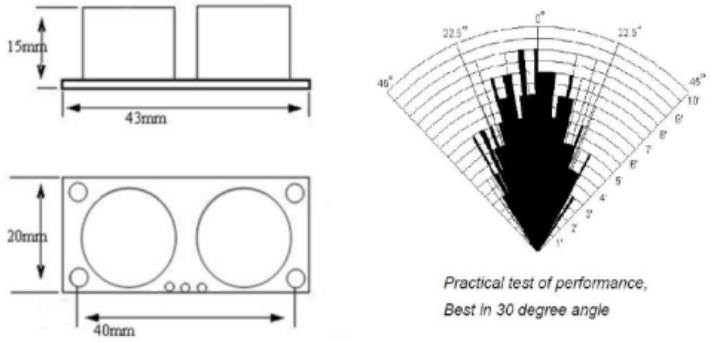
Διαιρούμε την απόσταση με το 2, γιατί στο συνολικό χρόνο συμπεριλαμβάνεται και ο χρόνος επιστροφής του υπέρηχου στο δέκτη.



Εικόνα 17: Ultrasonic wave

Χαρακτηριστικά:

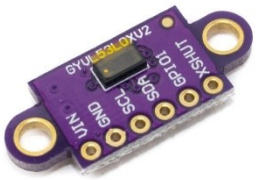
- Τάση λειτουργίας :+5V DC
- Ρεύμα ηρεμίας: <2mA
- Ρεύμα λειτουργίας: 15mA
- Αποτελεσματική γωνία: <15°
- Εμβέλεια: 2cm – 400 cm
- Γωνία μέτρησης: 30º
- Διαστάσεις: 45x20x15 mm



Εικόνα 18: Εύρος αισθητήρα

Η παραπάνω εικόνα 18 δείχνει τις διαστάσεις του αισθητήρα υπερήχων HC-SR04 καθώς και τη γωνία ανίχνευσης. Όπως μπορείτε να δείτε, ο αισθητήρας είναι πιο ακριβής όταν το προς ανίχνευση αντικείμενο βρίσκεται ακριβώς μπροστά του. Υπάρχει όμως και περίπτωση να ανιχνευθεί κάποιο αντικείμενο εκτός του εύρους των 30 μοιρών. Ο κατασκευαστής συνιστά τον περιορισμό αυτού του ανοίγματος στις 30 μοίρες (15 μοίρες σε κάθε πλευρά) για ακριβείς μετρήσεις.

5.2. Αισθητήρας απόστασης (laser) VL53L0X[4]



Εικόνα 19: Αισθητήρας απόστασης VL53L0X

Για τη μέτρηση της απόστασης ο αισθητήρας διαθέτει μια δέσμη λέιζερ και τεχνολογία Time-of-Flight (ToF), που μοιάζει σαν ένα μικρό LiDAR (Light Detection And Ranging.

Η λειτουργία του αισθητήρα απόστασης VL53L0X είναι παρόμοια με του αισθητήρα υπερήχων HC-SR04, με τη διαφορά ότι αντί να εκπέμπει υπερήχους, εκπέμπει μια δέσμη λέιζερ. Έχει μεγαλύτερη ακρίβεια από τους αισθητήρες που βασίζονται σε υπερήχους ή υπέρυθρες (IR). Αυτό συμβαίνει γιατί η δέσμη λέιζερ δεν επηρεάζεται από την ηχώ ή την ανάκλαση σε αντικείμενα όπως στις άλλες περιπτώσεις. Η επικοινωνία με το Arduino γίνεται μέσω του πρωτοκόλλου I^2C.

Περιγραφή των Pins:

- VIN – 5V
- GND - Γείωση
- SDA - I^2C data line
- SCL - I^2C clock line
- GPIO - Έξοδος interrupt
- SHDN – Σε κατάσταση LOW θέτει τον αισθητήρα σε κατάσταση αναμονής

Χαρακτηριστικά:

- Τάση λειτουργίας: 2.6 to 3.5 V
- Θερμοκρασία λειτ.: -20 to 70°C
- Διαστάσεις: 4.40 × 2.40 × 1.00 mm
- Επικοινωνία: I2C, εως 400 kHz
- Εμβέλεια: 30 mm to 2000 mm
- Ακρίβεια: (Εξαρτάται από το φως του περιβάλλοντος) ±3%

[4] https://www.hwlibre.com/el/vl53l0x/

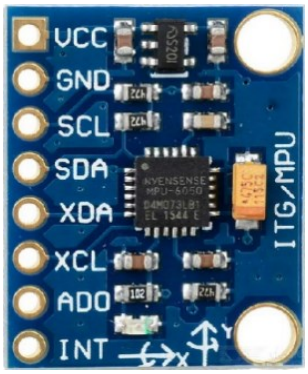
- Χρόνος μέτρησης:
  - 30 ms (Default mode)
  - 200 ms (High accuracy mode)
  - 33 ms (Long range mode)
  - 20 ms (High Speed mode)
- Οπτικό πεδίο: 25 μοίρες

Πίνακας 1: Σύγκριση αισθητήρων απόστασης

	HC-SR04	VL53L0X
Τύπος	Υπερήχων	Laser
Απόσταση	2-400cm	2-125cm (long-range mode 2-210cm)
Διαστάσεις	45x20x15mm	10x25x2mm
Γωνία μέτρησης	30º	25º

5.3. MPU6050

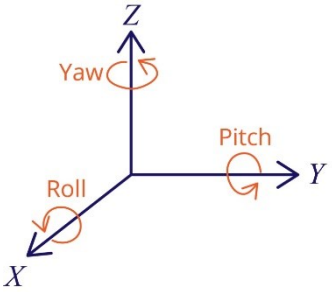
Η MPU-6050, ανήκει σε μια κατηγορία συσκευών που ονομάζεται Inertial Measurements Units (IMU). Μονάδα αδρανειακών μετρήσεων.



Εικόνα 20: MPU6050

Αυτός ο αισθητήρας έχει πολλές διαφορετικές εφαρμογές σε πολλούς τομείς, όπως η ρομποτική, τα drones, τα smartphones κ.α. Είναι ένας αισθητήρας χαμηλού κόστους και διαθέτει εσωτερικό γυροσκόπιο και επιταχυνσιόμετρο. Το επιταχυνσιόμετρο είναι τριών αξόνων, δηλαδή μετράει την επιτάχυνση στους άξονες x, y και y. Το γυροσκόπιο που χρησιμοποιεί ονομάζεται MEMS gyroscope (Micro Electro Mechanical System). Αποτελείται από τρεις αισθητήρες (ένα σε κάθε άξονα) που μας δίνουν μια τάση όταν περιστρέφονται.

Υπολογίζει τη σχετική θέση (x, y, z) και τον προσανατολισμό (roll, pich, yaw) εικόνα 21, την ταχύτητα και την επιτάχυνση του κινούμενου ρομπότ. Τα δεδομένα από το



Εικόνα 21

επιταχυνσιόμετρο και το γυροσκόπιο μεταφέρονται σε έναν εσωτερικό Ψηφιακό Επεξεργαστή Κίνησης (Digital Motion Processor ή DMP). Αυτός ο επεξεργαστής συνδυάζει τις πληροφορίες από τους δυο αισθητήρες και τις διαμορφώνει κατάλληλα ώστε να χρησιμοποιηθούν μέσω του I^2C.

Έχει 6 DOF (βαθμούς ελευθέριας), τρεις για το επιταχυνσιόμετρο και τρεις για το γυροσκόπιο. Διαθέτει επίσης εσωτερικό αισθητήρα θερμοκρασίας

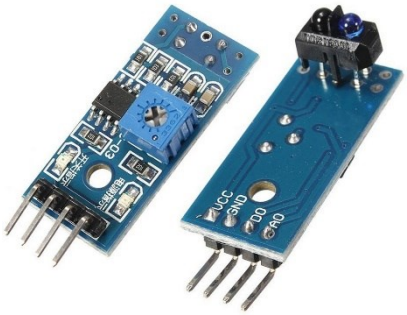
Μεγάλο πλεονέκτημα, είναι η ακρίβειά του σε σχέση με μια απλή πυξίδα και ο πολύ γρήγορος ρυθμός ανανέωσης των δεδομένων. Η επικοινωνία με το Arduino γίνεται μέσω του πρωτοκόλλου I2C.

Περιγραφή των pins:

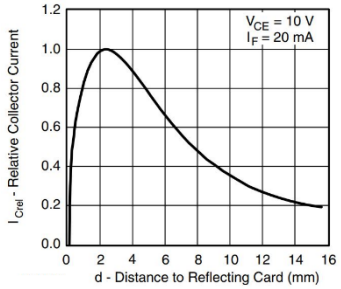
- VCC – 5V
- GND – Γείωση
- SCL – I^2C clock line
- SDA – I^2C data line
- XDA – Εξωτερική I^2C data line. (για σύνδεση εξωτερικών αισθητήρων)
- XCL – Εξωτερική I^2C clock line.
- AD0 – Με αυτό το pin μπορούμε να αλλάξουμε την εσωτερική I^2C διεύθυνση του MPU-6050
- INT – Έξοδος interrupt.

5.4. Αισθητήρας Υπέρυθρων TCRT5000

Και αυτός ο αισθητήρας ακολουθεί παρόμοια λειτουργία με τους προηγούμενους. Συνήθως χρησιμοποιείται στα ρομποτικά οχήματα για την ανίχνευση μαύρης γραμμής στο έδαφος (Line follower) και όχι τόσο για την ανίχνευση εμποδίων, λόγω τα περιορισμένης εμβέλειας του. Η εμβέλεια του κυμαίνεται από 1mm έως 25mm με μέγιστο ρεύμα περίπου στα 2,5mm. Υπάρχει επίσης ένα ποτενσιόμετρο όπου μπορούμε να ρυθμίσουμε την ευαισθησία.



Εικόνα 22: Αισθητήρας Υπερύθρων TCRT5000



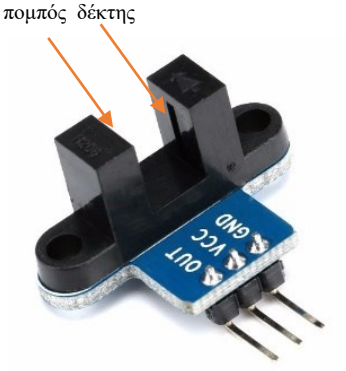
Εικόνα 23

Χαρακτηριστικά:

- Τάση λειτουργίας: 3V~5V
- Απόσταση ανίχνευσης: 1mm-25mm
- Ψηφιακή ή αναλογική έξοδος
- Ενσωματωμένο LED για ένδειξη ανίχνευσης.
- Ενσωματωμένο ποτενσιόμετρο για τη ρύθμιση της ευαισθησίας

5.5. Photo Interrupter Sensor & Wheel encoder

Αυτός ο αισθητήρας αποτελείται από έναν υπέρυθρο πομπό στη μία άκρη και έναν δέκτη στην άλλη σε σταθερά σημεία (εικόνα 24). Εκπέμπει διαρκώς μια δέσμη υπέρυθρου φωτός και έτσι μπορεί να ανιχνεύσει πότε παρεμβάλλεται ένα αντικείμενο διακόπτοντας τη δέσμη (εικόνα 26). Χρησιμοποιείται συνήθως σε συνδυασμό με δίσκους κωδικοποίησης (wheel encoder με σχισμές(οπές) (εικόνα 25) σε εφαρμογές όπως ανεμόμετρα η μετρητές ταχύτητας.

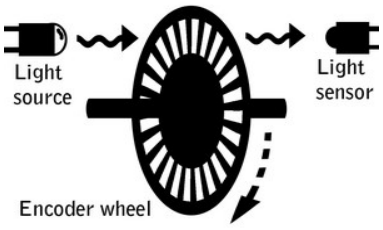


Εικόνα 24: Photo Interrupter

Εικόνα 25: Wheel encoder

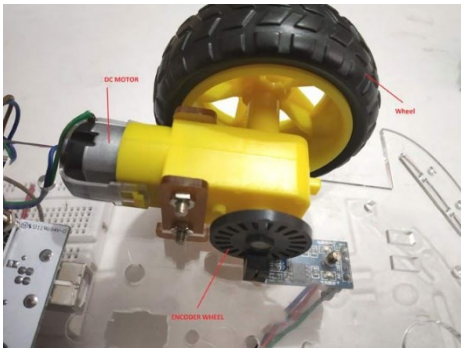
Χαρακτηριστικά:

- Τάση λειτουργίας: 3,3 V έως 5 V
- Διαστάσεις: 26.8×15×18.7mm
- Μέγεθος οπών τοποθέτησης: 3mm
- Πλάτος διάκενου: 6mm
- Λειτουργία εξόδου ψηφιακού σήματος (παλμικό σήμα)



Εικόνα 26: Wheel encoder

Ο δίσκος κωδικοποίησης συνδέεται στον άξονα του κινητήρα μαζί με τη ρόδα (εικόνα 27). Καθώς η ρόδα περιστρέφεται, ο photo Interrupter μετρά τον αριθμό των οπών του δίσκου κωδικοποίησης, και έτσι μπορούμε να μετατρέψουμε τον αριθμό αυτό σε διανυθείσα απόσταση. Αν υποθέσουμε ότι N είναι οι οπές που έχει δίσκος κωδικοποίησης και R είναι η ακτίνα της ρόδας, τότε θα έχουμε:



Εικόνα 27

Η διαδρομή που διανύθηκε σε μία περιστροφή θα είναι = $2 \cdot \pi \cdot R$

Η διαδρομή που διανύθηκε σε μια οπή είναι = $\frac{2 \cdot \pi \cdot R}{N}$

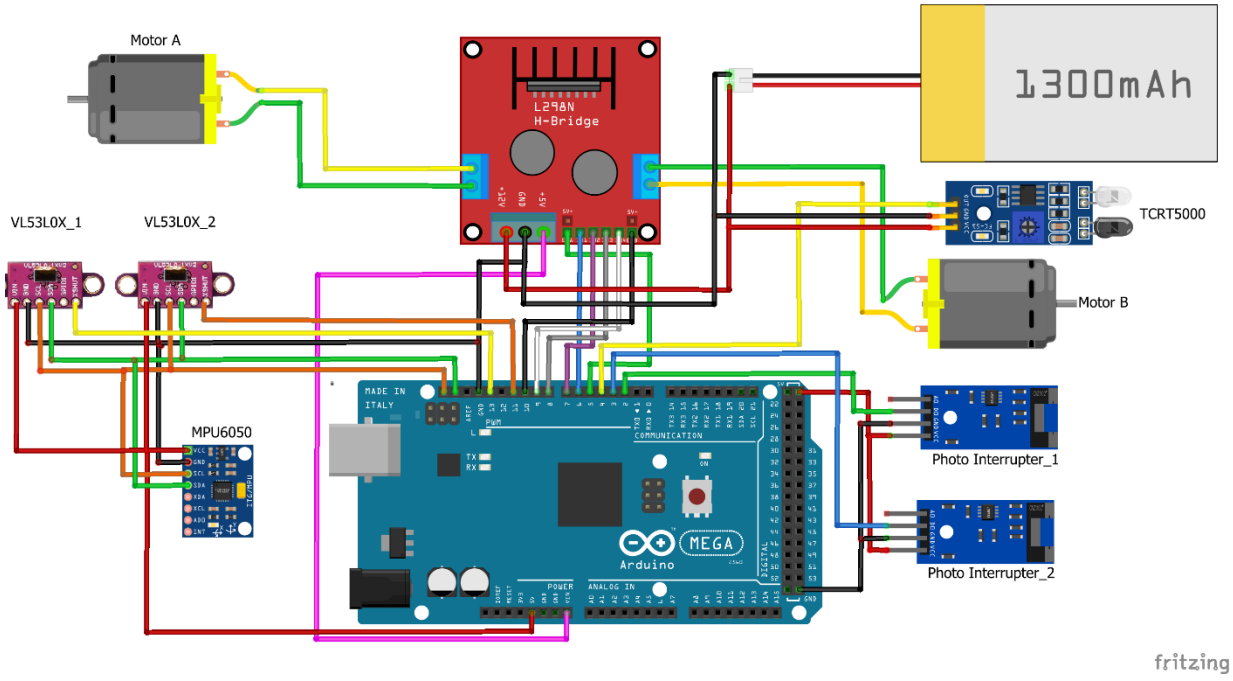
6. Κατασκευή ρομπότ

6.1. Υλικά που χρησιμοποιήθηκαν

Στην τελική κατασκευή χρησιμοποιήθηκαν:

1) Arduino Mega2560
2) Ελεγκτής κινητήρων L298N H-Bridge
3) 2 x DC Κινητήρες
4) 2 x τροχοί με πέλμα καουτσούκ
5) Μπαταρία Li-Po 1300mAh
6) 2 x Photo interrupter
7) 2 x Wheel encoders
8) 2 x Αισθητήρες laser VL53L0X
9) Μονάδα MPU6050
10) Αισθητήρας IR TCRT5000
11) Σασί Plexiglas
12) Διακόπτης on off
13) Breadboard mini
14) Καλώδια

6.2. Ηλεκτρονικό κύκλωμα

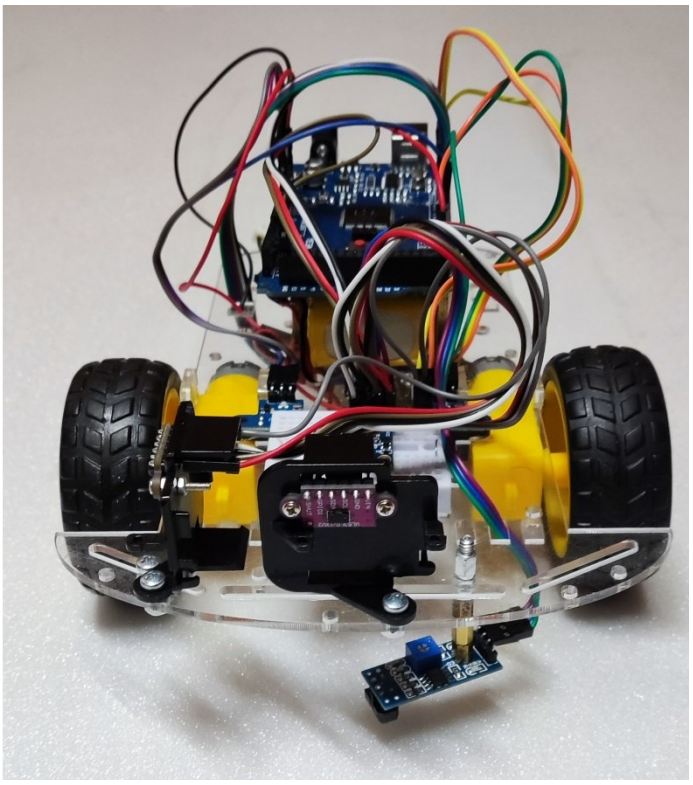


Εικόνα 28: Ηλεκτρονικό κύκλωμα του ρομποτικού οχήματος

7. Το αυτόνομο ρομποτικό όχημα

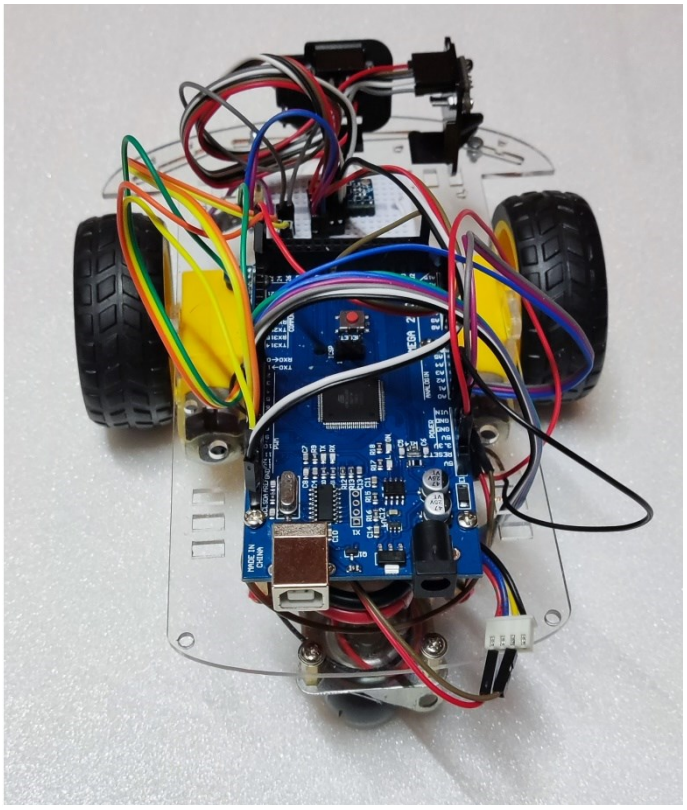
7.1. Μπροστινή όψη

Στη μπροστινή όψη έχουμε τον 1ο αισθητήρα laser, ενώ ο 2ος βρίσκεται τοποθετημένος στη δεξιά μεριά του οχήματος (αριστερά στην εικόνα 29) σε γωνία 90ο σε σχέση με τον 1ο. Ακριβώς από κάτω βλέπουμε και τον αισθητήρα υπέρυθρων TCRT5000, ο οποίος είναι βοηθητικός



Εικόνα 29: Μπροστινή όψη

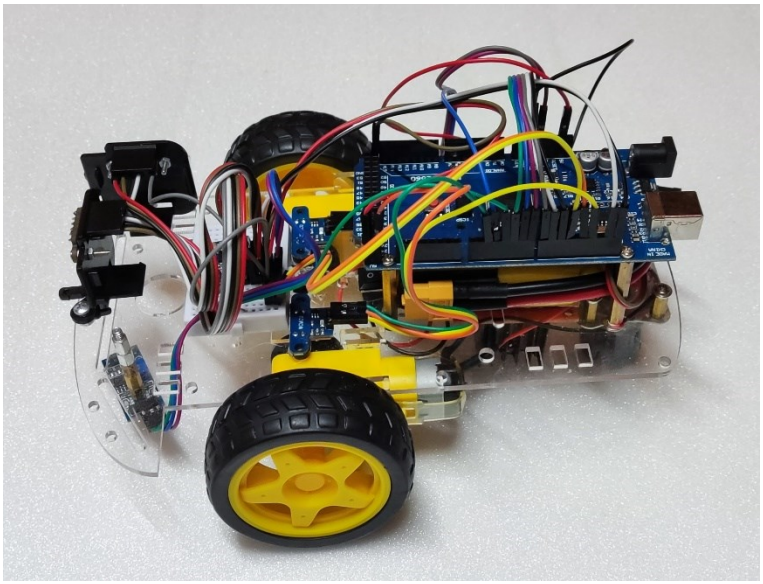
7.2. Πίσω όψη



Εικόνα 30: Πίσω όψη

Στην πίσω όψη έχει τοποθετηθεί το Arduino Mega2560

7.3. Αριστερή πλευρά

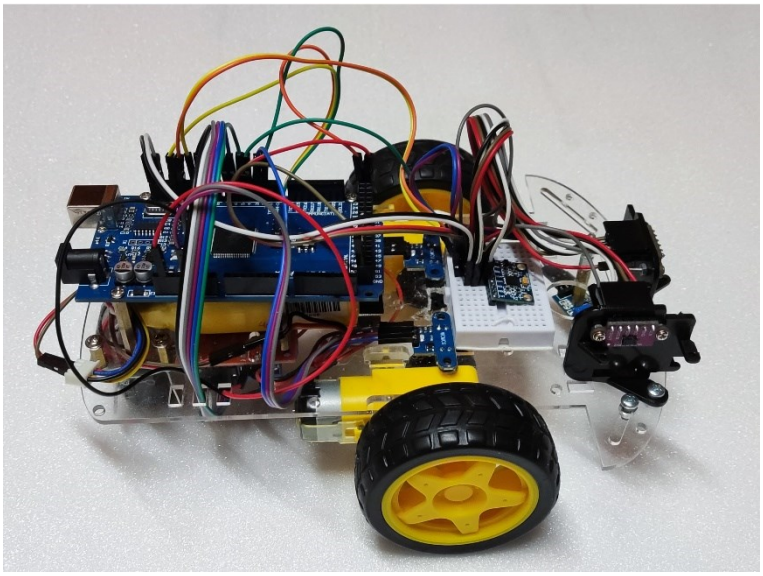


Κάτω από το Arduino, το οποίο είναι υπερυψωμένο, έχουμε τοποθετήσει τη μπαταρία για εξοικονόμηση χώρου.

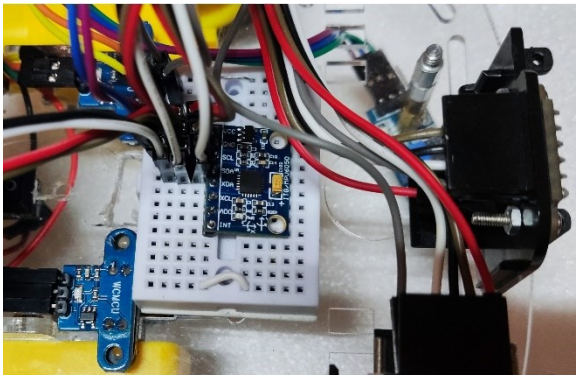
Εικόνα 31: Αριστερή πλευρά

7.4. Δεξιά πλευρά

Από τη δεξιά πλευρά βλέπουμε το 2ο αισθητήρα laser και ακριβώς από πίσω, το mini breadboard όπου είναι τοποθετημένη η μονάδα MPU6050. Λίγο πιο πίσω είναι και οι 2 photo interrupter.



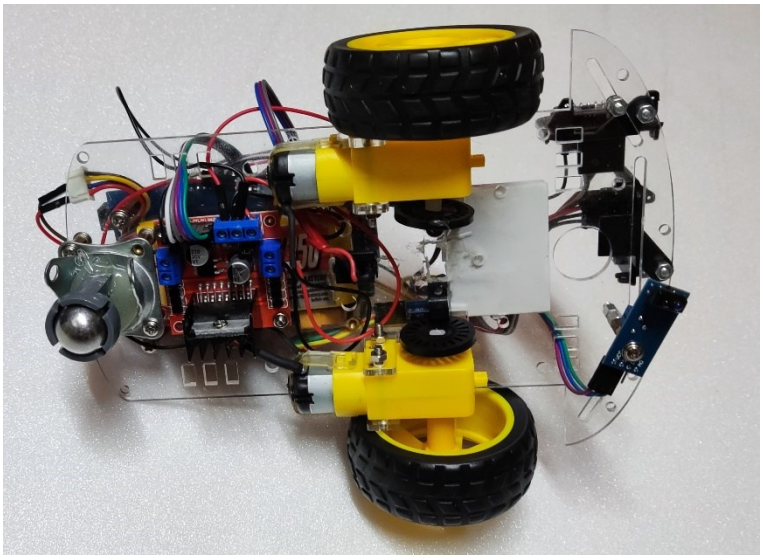
Εικόνα 32: Δεξιά πλευρά



Εικόνα 33: Μονάδα MPU6050

Η μονάδα MPU6050 έχει τοποθετηθεί στη μέση του ρομποτικού οχήματος και όσο πιο κοντά στους άξονες των κινητήρων. Έτσι θα έχουμε περισσότερη ακρίβεια στις μετρήσεις μας σε επιτόπιες στροφές.

7.5. Κάτω όψη

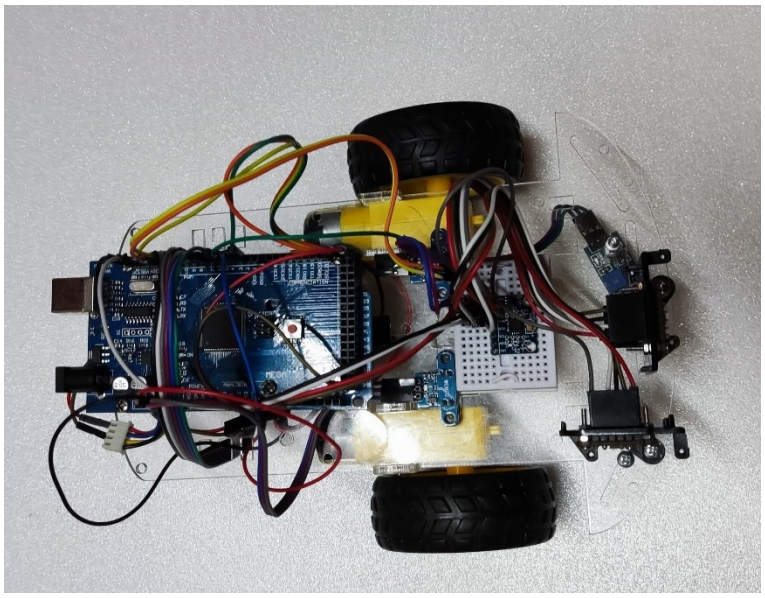


Στην κάτω πλευρά μαζί με τους κινητήρες DC, έχει τοποθετηθεί και ο ελεγκτής κινητήρων. Βλέπουμε επίσης και τους δύο wheel encoders πάνω στους άξονες των κινητήρων.

Εικόνα 34: Κάτω όψη

7.6. Πάνω όψη

Τέλος, στην εικόνα 35 βλέπουμε το ρομποτικό όχημα από πάνω.



Εικόνα 35: Πάνω όψη

8. Περιγραφή λειτουργίας

Πριν βάλουμε σε λειτουργία το ρομποτικό μας όχημα, του δίνουμε τη συνολική απόσταση από το στόχο σε ευθεία γραμμή. Επίσης τοποθετούμε το όχημα με φορά προς το στόχο, έτσι ώστε να «κοιτάει» προς αυτόν. Ο στόχος έχει οριστεί στα 200cm από την εκκίνηση και έχει σημειωθεί στο έδαφος με μαύρο χρώμα.

Καθώς το όχημα ξεκινάει να κινείται, η μονάδα MPU6050 είναι αυτή που διορθώνει την πορεία του, όταν αυτό παρεκκλίνει από τη νοητή ευθεία που του έχουμε ορίσει. Αυτό έγινε διότι, σχεδόν ποτέ το ρομπότ δεν κινείται ευθεία μόνο με τη ρύθμιση των στροφών του κινητήρα. Είτε λόγω μειωμένης πρόσφυσης των τροχών είτε κάποιας ανωμαλίας στο έδαφος, το ρομπότ έχανε την ευθυγράμμιση του. Με τη χρήση του γυροσκοπίου, όταν ξεπεράσει τις ±3 μοίρες, αυξομειώνει τις στροφές στους κινητήρες και το επαναφέρει στην ευθεία. Ταυτόχρονα οι αισθητήρες photo interrupters έχουν ξεκινήσει τη μέτρηση της απόστασης που διανύουν.

A. Στην περίπτωση που δεν ανιχνευθεί κανένα εμπόδιο από τον αισθητήρα laser μπροστά από το όχημα, το ρομπότ θα διανύσει σε ευθεία γραμμή την απόσταση και ο αλγόριθμος θα τερματιστεί όταν οι αισθητήρες photo interrupters έχουν μετρήσει τα 200cm. Ο αισθητήρας υπερύθρων TCRT5000 είναι βοηθητικός και μπορεί σε περίπτωση σφάλματος της μέτρησης από τους photo interrupters, αν ανιχνεύσει μαύρο χρώμα στο έδαφος να τερματίσει και αυτός πρώτος τον αλγόριθμο. Τουλάχιστον έτσι δεν θα ξεφύγουμε από το στόχο.

B. Σε περίπτωση που ανιχνευθεί εμπόδιο μπροστά, τότε το ρομπότ στρίβει 90 μοίρες αριστερά. Η μονάδα MPU6050 τώρα θα ελέγχει την πορεία σε ευθεία από 90 και ±3 μοίρες. Οι photo interrupters ξεκινάνε νέα μέτρηση (πορεία αριστερά) και το ρομπότ συνεχίζει την πορεία ελέγχοντας τώρα ταυτόχρονα και μπροστά και δεξιά με τους αισθητήρες laser για εμπόδια. Όσο υπάρχει εμπόδιο στα δεξιά και όχι μπροστά το ρομπότ συνεχίζει ευθεία.

    a. Αν σταματήσει να ανιχνεύει εμπόδιο στα δεξιά τότε στρίβει δεξιά 90 μοίρες και συνεχίζει ευθεία. Συνεχίζεται η μέτρηση της απόστασης από τους αισθητήρες, προσθέτοντας τη στην προηγούμενη(πορεία ευθεία).
        i. Αν δεν ανιχνεύσει άλλο εμπόδιο στα δεξιά, σταματάει, υπολογίζεται η ευκλείδειος γωνία από τις μετρήσεις (πορεία ευθεία, πορεία αριστερά και υπόλοιπο απόστασης) και το ρομπότ στρίβει τόσες μοίρες και κατευθύνεται προς το στόχο.
        ii. Αλλιώς συνεχίζει την πορεία ευθεία.

C. Αν ανιχνευθεί εμπόδιο μπροστά ενώ υπάρχει και εμπόδιο στα δεξιά, τότε το ρομπότ ξανά στρίβει αριστερά (έχουμε φτάσει 180μοίρες) και συνεχίζεται η ίδια διαδικασία. Η μόνη διαφορά τώρα είναι ότι οι photo interrupters αφαιρούν από την προηγούμενη απόσταση (πορεία ευθεία) αυτό που μετράνε.

9. Συμπεράσματα

Από τη μελέτη της κίνησης του ρομποτικού οχήματος στο χώρο έγινε κατανοητό το πόσο δύσκολο είναι το πρόβλημα της πλοήγησης και οδομετρίας. Αποδείχθηκε ότι για τη σωστή πλοήγηση σε οποιοδήποτε έδαφος με ένα ή δυο μόνο όργανα μετρήσεων δεν είναι αρκετά ώστε να μας δώσουν ακριβή δεδομένα. Για το λόγο αυτό, απαιτείται ο συνδυασμός των μετρήσεων από περισσότερους αισθητήρες. Μόνο σε αυτή την περίπτωση μπορούν να δημιουργηθούν αξιόπιστα δεδομένα οδομετρίας. Η μονάδα μέτρησης αδρανειακών μεγεθών, MPU, με τη σωστή βαθμονόμηση και κατανόηση του τρόπου λειτουργίας της, μας παρείχε μεγάλη αξιοπιστία.

Παρά το γεγονός ότι στα πειράματα δεν υπήρχε ακρίβεια στις κινήσεις ούτε ομαλότητα στην ταχύτητα, η οδομετρία που επιτεύχθηκε, έδωσε ενθαρρυντικά αποτελέσματα όσον αφορά το συνδυασμό μονάδας μέτρησης αδρανειακών μεγεθών και των κωδικοποιητών ταχύτητας

10.Δυσκολίες

- Οι 3 αισθητήρες HC-SR04 που χρησιμοποιήθηκαν στον αρχικό σχεδιασμό, παρουσίαζαν μεγάλη αστάθεια όταν το ρομπότ βρισκόταν σε κίνηση και επίσης όταν υπήρχε μεγάλη γωνία αισθητήρα με εμπόδιο υπήρχε μεγάλη ανακρίβεια στις μετρήσεις. Για αυτό και αντικαταστάθηκαν με τους laser. Αν και είχαν μικρότερη εμβέλεια από τους HC-SR04, αυτό δεν αποτέλεσε πρόβλημα επειδή η κίνηση του ρομπότ προοριζόταν σε κλειστό χώρο.

- Αλλαγή του Arduino Uno σε Mega λόγω αύξησης του μεγέθους του προγράμματος εξαιτίας των βιβλιοθηκών (laser & MPU) που χρησιμοποιήθηκαν. Η χωρητικότητα σε μνήμη του Arduino Uno δεν αρκούσε.

- Δυσκολία ρύθμισης της μονάδας MPU6050. Η μονάδα στέλνει τα δεδομένα στο buffer μια φορά κάθε 10μsec και ενημερώνει το Arduino ότι είναι έτοιμα. Για να καταφέρουμε να κάνουμε μετρήσεις πάνω στον άξονα y (yaw) έπρεπε να διαβάζουμε αυτά τα δεδομένα, χωρίς να ξεπεράσουμε το όριο των 10μsec, διαφορετικά ο buffer υπερχείλιζε και από εκεί και πέρα λαμβάναμε λανθασμένες μετρήσεις.

- Οι κινητήρες DC με ίδιο παλμό μας έδιναν διαφορετική ταχύτητα. Με διαφορετική μπαταρία άλλαζε η ταχύτητα.

- Οι τροχοί έχαναν την πρόσφυση σε κάποιες επιφάνειες ή σε απότομες στροφές

11.Βελτιώσεις – Επεκτάσεις

- Για ακόμη μεγαλύτερη ακρίβεια θα μπορούσε να χρησιμοποιηθεί η μονάδα MPU9250 η οποία περιλαμβάνει τα ίδια με την MPU6050 συν και πυξίδα.
- Θα μπορούσε να τοποθετηθεί και τρίτος αισθητήρας laser στην αριστερή μεριά του ρομπότ.
- Η προσθήκη ενός LCD Display για την εμφάνιση μετρήσεων π.χ της απόστασης από την αρχή έως το στόχο.
- Η χρήση βηματικού κινητήρα, αν δεν μας απασχολεί η ταχύτητα, θα μας έδινε περισσότερη ακρίβεια σε ευθεία πορεία αλλά και σε στροφές.
- Μια διαφορετική προσέγγιση του προβλήματος οδομετρίας θα ήταν η χρήση μιας στερεοσκοπικής κάμερας, για τη μέτρηση της απόστασης που διανύθηκε. Σε συνδυασμό με τα δεδομένα από τον αισθητήρα μέτρησης αδρανειακών μεγεθών μπορούν να προκύψουν αξιόπιστα αποτελέσματα.

12.Κώδικας Arduino

```
/*
 *Διεθνές Πανεπιστήμιο Ελλάδος
 *MSC in Robotics
 *Θεοδοσιάδης Ευάγγελος - AM:23
 *Διπλωματική Εργασία
 */


#include <Wire.h>
#include <MPU6050_light.h>  //Η βιβλιοθήκη που χρησιμοποιήθηκε για
την IMU
#include "Adafruit_VL53L0X.h" //Η βιβλιοθήκη για τον αισθητήρα laser

// Καθορισμός διευθύνσεων των VL53L0X επειδή χρησιμοποιήσαμε 2
#define LOX1_ADDRESS 0x30
#define LOX2_ADDRESS 0x31

// Οι αισθητήρες VL53L0X συνδέονται στα Pin 11& 13
#define SHT_LOX1 11
#define SHT_LOX2 13

// Δημιουργία objects για τους  VL53L0X
Adafruit_VL53L0X lox1 = Adafruit_VL53L0X();
Adafruit_VL53L0X lox2 = Adafruit_VL53L0X();

// Εδώ αποθηκεύονται προσωρινά η μετρήσεις από τους VL53L0X
VL53L0X_RangingMeasurementData_t measure1;
VL53L0X_RangingMeasurementData_t measure2;

MPU6050 mpu(Wire);

// Timer
unsigned long currentTime = 0; // Βοηθητικός timer για τον υπολογισμό
περιστροφής στον άξονα z (yaw)

//Μεταβλητές
float yaw = 0;  //Αρχική θέση ρομπότ σε σχέση με
float yaw_turn=0; //μετράει τις μοίρες σε σχέση με την αρχική θέση
float target_distance=200; //Εδώ δηλώνουμε την απόσταση του στόχου
από την εκκίνηση σε cm
float distance_R=0; //Συνολική απόσταση που έχει διανύσει ο δεξιός
τροχός
float distance_L=0; //Συνολική απόσταση που έχει διανύσει ο αριστερός
τροχός
float total_distance=0;
float total_distance_straight= 0;  //Συνολική απόσταση σε πορεία
ευθεία
float total_distance_left= 0;  //Συνολική απόσταση σε πορεία αριστερά
int angle=0;
int final_angle=0;  // Η τελική γωνία επιστροφής προς το στόχο

// Στα pin 2 & 3 συνδέονται οι photo interrupter
// Είναι τα pins που χρησιμοποιούνται για Interrupts
const byte MOTOR_R = 3;  // INT 1 - Right Motor
const byte MOTOR_L = 2;  // INT 0 - Left Motor

// Μετρητές για τους παλμούς από τα interrupt pin
volatile int counter_R = 0; //Για το δεξί τροχό
```

```
volatile int counter_L = 0; //Για τον αριστερό τροχό

// Σταθερά για τις σχισμέ (οπές) που έχει ο wheel encoder
const float stepcount = 20.00;  // Δηλώνουμε τον αριθμό οπών

// Σταθερά για τη διάμετρο του τροχού
const float wheeldiameter = 66; // Διάμετρος τροχού σε mm (περίμετρος
=2πR=207.3mm)
// Υπολογισμός απόστασης που διανύει ο τροχός σε κάθε σήμα από το
wheel encoder
float cm_step=1.3; // περίμετρος/οπές = 207.3mm/20=1,3 cm

// Δεξιός κινητήρας
int ena = 5;   //10->5
int IN1 = 6; // RIGHT --  8->6
int IN2 = 7; // RIGHT ++  9->7

// Αριστερός κινητήρας
int IN3 = 8; // LEFT ++   7->8
int IN4 = 9; // LEFT --   6->9
int enb = 10;    //5->10

int c,r; // Μεταβλητές για αποθήκευση απόστασης από εμπόδια
int turn=0; // Αριθμός στροφών 90μοιρών
int terminateIR = 4; // Pin για τον αισθητήρα TCRT5000
int deg=69;  // Αριθμός για στροφή 90 μοιρών

// Μετρητής παλμών δεξιού κινητήρα
void ISR_countR() {
  counter_R++;  // αυξάνει την τιμή του μετρητή του δεξιού κινητήρα
}

// Μετρητής παλμών αριστερού κινητήρα
void ISR_countL() {
  counter_L++;  // αυξάνει την τιμή του μετρητή του αριστερού
κινητήρα
}

void setID() {  // Διαδικασία ανάθεσης διευθύνσεων στους 2 αισθητήρες
VL53L0X

  // all reset
  digitalWrite(SHT_LOX1, LOW);
  digitalWrite(SHT_LOX2, LOW);
  delay(10);
  // all unreset
  digitalWrite(SHT_LOX1, HIGH);
  digitalWrite(SHT_LOX2, HIGH);
  delay(10);

  // activating LOX1 and reseting LOX2
  digitalWrite(SHT_LOX1, HIGH);
  digitalWrite(SHT_LOX2, LOW);

  // initing LOX1
  if(!lox1.begin(LOX1_ADDRESS)) {
    Serial.println(F("Failed to boot first VL53L0X"));
    while(1);
  }
  delay(10);
```

```
  // activating LOX2
  digitalWrite(SHT_LOX2, HIGH);
  delay(10);

  //initing LOX2
  if(!lox2.begin(LOX2_ADDRESS)) {
    Serial.println(F("Failed to boot second VL53L0X"));
    while(1);
  }
}

void setup() {

  Serial.begin(115200);

   // wait until serial port opens for native USB devices
  while (! Serial) { delay(1); }

  pinMode(SHT_LOX1, OUTPUT);
  pinMode(SHT_LOX2, OUTPUT);

  Serial.println(F("Shutdown pins inited..."));

  digitalWrite(SHT_LOX1, LOW);
  digitalWrite(SHT_LOX2, LOW);

  Serial.println(F("Both in reset mode...(pins are low)"));


  Serial.println(F("Starting..."));

  setID();
   //Διαδικασία έναρξης μονάδας MPU6050
   Wire.begin();
   byte status = mpu.begin(); //Start communication with the MPU6050
device
   Serial.print(F("MPU6050 status: "));
   Serial.println(status);
   while (status != 0) { } // stop everything if could not connect to
MPU6050
   Serial.println(F("Calculating offsets, do not move MPU6050"));
   delay(1000);
   mpu.calcOffsets(); // Compute gyroscope and accelerometer offsets
to remove measurement bias.
                      //MPU6050 device must be on a flat surface
during the calibration
   Serial.println("Done!\n");

   // Attach the Interrupts to their ISR's (Interrupt Service
Routines)
  attachInterrupt(digitalPinToInterrupt (MOTOR_R), ISR_countR,
RISING);  //Αύξηση του μετρητή R όταν το pin του photo interrupt
γίνεται High
  attachInterrupt(digitalPinToInterrupt (MOTOR_L), ISR_countL,
RISING);  //Αύξηση του μετρητή L όταν το pin του photo interrupt
γίνεται High

// Δήλωση των pin του ελεγκτή κινητήρων ως έξοδοι
pinMode(ena, OUTPUT);
pinMode(enb, OUTPUT);
pinMode(IN1, OUTPUT);
```

```
pinMode(IN2, OUTPUT);
pinMode(IN3, OUTPUT);
pinMode(IN4, OUTPUT);

pinMode(terminateIR, INPUT); // Pin του TCRT5000 ως είσοδος

}

void loop() {

  read_yaw(); //Διαβάζουμε την τιμή περιστροφή κατά τον άξονα z (yaw)
  read_dual_sensors(); // Τιμές από τους αισθητήρες VL530LX

 // Serial.print(" Yaw = "); //βοηθητικές εντολές
 // Serial.print(yaw);

  if (yaw_turn==0){  //έλεγχος αν η πορεία είναι στην ευθεία με το
στόχο
    if(c>20 && r>10){  // έλεγχος για εμπόδιο μπροστά
      if ( yaw + final_angle >= -3+yaw_turn && yaw + final_angle <=
3+yaw_turn) {  //Έλεγχος ευθυγράμμισης
      forward();   // κίνηση ευθεία
      }

      else if ( yaw + final_angle< -2+yaw_turn){
        move_left();  // κίνηση προς αριστερά
      }

      else if (yaw + final_angle> 2+yaw_turn) {
        move_right(); // κίνηση προς τα δεξιά
      }
    }
    else if (c<=20){  // έλεγχος αν ανιχνευθεί εμπόδιο μπροστά
      pause();        // παύση κινητήρων

      // αποθήκευση απόστασης σε ευθεία πορεία
      save_distance_straight();

      while(yaw < deg + yaw_turn){   // κάνουμε κάθετη στροφή
αριστερά
        turn_left();
        read_yaw();  // διαβάζουμε την τιμή yaw συνεχώς
        pause();
      }

      counter_R=0; //μηδενίζουμε τους μετρητές απόστασης μετά από
κάθε στροφή 90μοιρών
      counter_L=0;

      turn=turn+1; //1η στροφή 90μοιρων αριστερά
      yaw_turn = yaw_turn + yaw; // θέτουμε τη νέα τιμή για πορεία
ευθεία

    }
    check_target();

  }


  read_yaw(); // διαβάζουμε την τιμή yaw συνεχώς
```

```
  if (yaw_turn!=0){    //;έλεγχος όταν η πορεία είναι σε διαφορά 90+ μοιρών από το στόχο
     read_dual_sensors();// διαβάζουμε ξανά τις τιμές από τους αισθητήρες VL530LX

    if (c<=25 && r<=25){ // Όταν υπάρχει εμπόδιο και μπροστά και δεξιά
      pause();
      save_distance_left();

      while(yaw < (deg + yaw_turn)){   // με την ταχύτητα μέχρι να σταματήσει θα είναι 90 μοιρες
        turn_left();
        read_yaw();
        pause();
      }

      counter_R=0; //μηδενίζουμε τους μετρητές απόστασης μετά από κάθε στροφή 90μοιρών
      counter_L=0;

      turn=turn+1; // αριθμός στροφών 90μοιρων αριστερά
      yaw_turn =yaw_turn+yaw;           //////////180

    }

    if (c>25 && r<=25){  // αν υπάρχει εμπόδιο μόνο στα δεξιά, το ακολουθούμε

      if ( yaw >=  7+yaw_turn && yaw <= 13+yaw_turn) {    //διόρθωση μετά τη στροφή αριστερά
        forward();
      }

      else if ( yaw < 12+yaw_turn){
        move_left();
      }

      else if (yaw > 8+yaw_turn) {
        move_right();
      }
    }

    if (c>25 && r>25){  //όταν δεν υπάρξει εμπόδιο μπροστά και δεξιά
      pause();
      save_distance_left();

      for(int i=0; i<40; i++){
        //Serial.print(i);
        forward2();
        delay(10);
        read_yaw();
        }

      while(yaw>yaw_turn-deg){   // με την ταχύτητα μέχρι να σταματήσει θα είναι 90μοιρες
        turn_right();
        read_yaw();
        pause();
      }
```

```
      counter_R=0; //μηδενίζουμε τους μετρητές απόστασης μετά από
κάθε στροφή 90μοιρών
      counter_L=0;

      turn=turn-1; //στροφή 90 μοιρών αριστερά
      yaw_turn = 0;
      calculate_angle();
    }

  }

  if (digitalRead(terminateIR)==1) { // έλεγχος αισθητήρα TCRT5000
εάν έχουμε φτάσει στο στόχο
    pause(); // παύση κινητήρων
    exit(0);  // τερματισμός αλγορίθμου
  }

}
void check_target() {

total_distance=(total_distance_straight*cm_step)+(((counter_R+counter
_L)/2)*cm_step)-(total_distance_left*cm_step);
  if (total_distance>=target_distance){
    pause(); // παύση κινητήρων
    exit(0);  // τερματισμός αλγορίθμου
    }
}

void calculate_angle() {  //υπολογισμός γωνίας επιστροφής προς το
στόχο
  float a,b,c,bc;

  a=target_distance-(total_distance_straight*cm_step); //μετατροπή
των βημάτων σε cm
  b=total_distance_left*cm_step;

  c=sqrt(pow(a, 2)+pow(b,2));  //εύρεση υποτείνουσας
  bc=b/c;

  angle=acos(bc)*180/3.14; //εύρεση γωνίας
  final_angle=angle-deg;  // οι μοίρες που πρέπει να στρίψει το
ρομπότ προς τα δεξιά

}
void save_distance_straight() {
  if (turn==2){
    total_distance_straight=total_distance_straight-
((counter_R+counter_L)/2);
  }

total_distance_straight=total_distance_straight+((counter_R+counter_L
)/2);
}

void save_distance_left() {
  if (turn==3){
    total_distance_left=total_distance_left-
((counter_R+counter_L)/2);
  }
  total_distance_left=total_distance_left+((counter_R+counter_L)/2);
}
```

```
void read_yaw(){
  mpu.update();
  if ((millis() - currentTime) > 10) {// print data every 10ms
    //Serial.print(currentTime);
    //Serial.print("  ");
   // Serial.print(millis() - currentTime);
     //Serial.print("Z : ");
     yaw = mpu.getAngleZ();
     //Serial.println(mpu.getAngleZ());
     currentTime = millis();
   }
}

void read_dual_sensors() {

  lox1.rangingTest(&measure1, false); // pass in 'true' to get debug
data printout!
  read_yaw();
  lox2.rangingTest(&measure2, false); // pass in 'true' to get debug
data printout!

  // print sensor one reading
  Serial.print(F("   1: "));
  if(measure1.RangeStatus != 4) {     // if not out of range
    Serial.print(measure1.RangeMilliMeter);
    c=measure1.RangeMilliMeter/10;
  } else {
    Serial.print(F("Out of range"));
    c=120;
  }

  Serial.print(F(" "));

  // print sensor two reading
  Serial.print(F("         2: "));
  if(measure2.RangeStatus != 4) {
    Serial.print(measure2.RangeMilliMeter);
    r=measure2.RangeMilliMeter/10;
  } else {
    Serial.print(F("Out of range"));
    r=120;
  }

  Serial.println();
}

void forward2() {
      digitalWrite(IN1, HIGH); //δεξιά εμπρός
      digitalWrite(IN2, LOW);
      analogWrite(ena, 80);
      digitalWrite(IN3,HIGH); //αριστερά εμπρός
      digitalWrite(IN4,LOW);
      analogWrite(enb, 90);
}
void forward() {
      digitalWrite(IN1, HIGH); //δεξιά εμπρός
      digitalWrite(IN2, LOW);
      analogWrite(ena, 80);
      digitalWrite(IN3,HIGH); //αριστερά εμπρός
      digitalWrite(IN4,LOW);
```

```
      analogWrite(enb, 90);
}

void pause() {
  digitalWrite(IN1, LOW);
  digitalWrite(IN2,LOW);
  analogWrite(ena, 0);
  digitalWrite(IN1, LOW);
  digitalWrite(IN2,LOW);
  analogWrite(enb, 0);
}

void turn_left() {//////////////
        digitalWrite(IN1, HIGH); //δεξιά εμπρός
        digitalWrite(IN2,LOW);
        analogWrite(ena, 80);
        digitalWrite(IN3,LOW);
        digitalWrite(IN4,HIGH); //αριστερά πίσω
        analogWrite(enb, 90);
        //delay(250); //στροφή 90 μοιρών
}

void turn_right() {
        digitalWrite(IN1, LOW);
        digitalWrite(IN2,HIGH); //δεξιά πίσω
        analogWrite(ena, 80);
        digitalWrite(IN3,HIGH); // αριστερά εμπρός
        digitalWrite(IN4,LOW);
        analogWrite(enb, 90);
}

void move_left() {
        digitalWrite(IN1, HIGH); //δεξιά εμπρός
        digitalWrite(IN2,LOW);
        analogWrite(ena, 80);
        digitalWrite(IN3,HIGH); // αριστερά εμπρός
        digitalWrite(IN4,LOW);
        analogWrite(enb, 67);
}

void move_right() {
        digitalWrite(IN1, HIGH); //δεξιά εμπρός
        digitalWrite(IN2,LOW);
        analogWrite(ena, 50);
        digitalWrite(IN3,HIGH); // αριστερά εμπρός
        digitalWrite(IN4,LOW);
        analogWrite(enb, 90);

}
```

13.Βιβλιογραφία

[1] N. Sariff και N. Buniyamin, «An Overview of Autonomous Mobile Robot Path Planning Algorithms,» *4th Student Conference on Research and Development,* pp. 183-188, 2006.

[2] V.Lumelsky και P.Stepanov, «Dynamic path planning for a mobile automaton with limited in formation on the environment,» *IEEE Trans,* τόμ. 31, αρ. 11, pp. 1058-1063, 1986.

[3] A. Sankaranarayanan και M. Vidyasagar, «A new path planning algorithm for moving a point object amidst unknown obstacles in a plane,» *in: IEEE Conference on Robotics,* p. pp. 1930–1936, 1990.

[4] I. Kamon και E. Rivlin, «Sensory-based motion planning with global proofs,» *IEEE Trans,* τόμ. 13, αρ. 6, pp. 814-821, 1997.

[5] G. Klancar, A. Zdešar, S. Blažic και I. Škrjanc, Wheeled Mobile Robotics, Oxford: Butterworth-Heinemann, 2017.

[6] L.-Y. Chung, «Remote Teleoperated and Autonomous Mobile Security Robot Development in Ship Environment,» *Mathematical Problems in Engineering,* p. 14, 2013.

[7] «Tutorials Point,» [Ηλεκτρονικό]. Available: https://www.tutorialspoint.com/arduino/index.htm. [Πρόσβαση 02 2021].

[8] S.Laubach και J.Burdick, «Anautonomous sensor-based path-planner for planetary microrovers,» *in:IEEE Conference on Robotics and Automation,* p. 347–354, 1999.

[9] R. Siegwart, I. Nourbakhsh και D. Scaramuzza, Introduction to autonomous mobile robots 2nd, Cambridge: MIT Press, 2011.

[10] A. G. Smith, Introduction to Arduino: A piece of cake., CreateSpace Independent Publishing Platform, 2011.

[11] H. Jahanshahi, N. Sari Naeimeh, V.-T. Pham, R. Khajepour και C. Volos, Recent Advances In Robot Path Planning Algorithms, New York: Nova Science Publishers, 2020.

[12] M. Margolis, Make an Arduino Controlled Robot, Maker Media, Inc,, 2012.

[13] «Comparison of various obstacle avoidance algorithms.,» [Ηλεκτρονικό]. Available: https://www.ijert.org/research/comparison-of-various-obstacle-avoidance-algorithms-IJERTV4IS120636.pdf.. [Πρόσβαση 15 07 2021].

[14] K. Berns και E. v. Puttkamer , Autonomous Land Vehicles, Wiesbaden: Vieweg+Teubner, 2009.

[15] «Η αξιοποίηση των αισθητήρων του Arduino στις εργαστηριακές και ερευνητικές δραστηριότητες,” 4ο Πανελλήνιο Εκπαιδευτικό Συνέδριο Κεντρικής Μακεδονίας,» 2016. [Ηλεκτρονικό]. Available: http://4syn-thess2016.ekped.gr/axiopiisi-ton-esthitiron-tou-tou-arduino-stis-ergastiriakes-erevnitikes-drastiriotites/. [Πρόσβαση 15 07 2021].

[16] R. Santos, «18+ Random Nerd Tutorials Projects,» [Ηλεκτρονικό]. Available: https://randomnerdtutorials.com.

[17] Ε. Πουλάκης, «Προγραμματίζοντας με το μικροελεγκτή Arduino,» 2015. [Ηλεκτρονικό]. Available: http://users.sch.gr/manpoul/docs/arduino/ProgrammingArduino.pdf.

14.Εικόνες