ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Δημιουργία Wall Following ρομπότ με Ultrasonic αισθητήρες τόσο για εσωτερικό όσο και για εξωτερικό χώρο με την χρήση Arduino.

Efficient wall following robot with ultrasonic sensors that works in both indoor and outdoor environments using Arduino

Θεόδωρος Γκίσης

Επιβλέπων καθηγητής**: Φραγκούλης Γεώργιος**.

Κοζάνη 2021

Πίνακας Περιεχομένων

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

Περίληψη

Στόχος της παρούσας διπλωματικής εργασίας είναι η δημιουργία ενός ρομπότ το οποίο έχει σχεδιαστεί για να ακολουθεί παράλληλα τοίχους. Σαν ‘’εγκέφαλο’’ όλου αυτού του ενσωματωμένου συστήματος θα χρησιμοποιήσουμε τον μικροελεγκτή Arduino Uno.Θα δούμε σε βάθος τι ακριβώς είναι ένα Arduino, καθώς επίσης θα αναλύσουμε και τα χαρακτηριστικά του.

Ως ‘’μάτια’’ του συστήματος θα χρησιμοποιήσουμε δύο ultrasonic αισθητήρες για τον εντοπισμό και την αποφυγή εμποδίων. Θα επεκτείνουμε τις γνώσεις μας στο τι ακριβώς είναι ένας τέτοιος αισθητήρας, πως λειτουργεί και τι χαρακτηριστικά έχει.

Το τσιπ l239D θα αποτελέσει το ‘’σώμα’’ του ρομπότ μας, καθώς παίζει τον βασικό ρόλο για την κίνησή του. Και σε αυτό το κεφάλαιο θα δούμε λεπτομερώς το τι πραγματικά είναι αυτό το τσιπ και γιατί είναι τόσο διαδομένο, θα αναλύσουμε το τρόπο με τον οποίο καταφέρνει να ορίσει την κατεύθυνση και την ταχύτητα ενός dc κινητήρα καθώς επίσης και κάποια κύρια χαρακτηριστικά του.

Αφού δούμε και αναλύσουμε όλα τα σημαντικά εξαρτήματα του συστήματος, επόμενο βήμα είναι να εξετάσουμε πώς μπορούμε στην πράξη να υλοποιήσουμε την κατασκευή αυτή. Θα υπάρξει αναλυτική περιγραφή και φωτογραφίες για την κατασκευή του αμαξιδίου καθώς επίσης και αναλυτική επεξήγηση όλου του κυκλώματος για την καλύτερη κατανόηση.

Στο τελευταίο και πιο σημαντικό κομμάτι θα δούμε όλο το λογισμικό του συστήματος, θα αναλύσουμε τα πιο σημαντικά του κομμάτια και θα δούμε τί είναι ένας PID controller και γιατί αποφασίσαμε να το χρησιμοποιήσουμε στην εργασία μας.

Λέξεις κλειδιά: Arduino , Wall following robot , PID, L239D motor controller, Ultrasonic Sensors.

Abstract

In this thesis the goal is to create a robot that can follow parallel a wall. As ‘’brain’’ of the integrated system we use the microcontroller Arduino Uno. We will see in depth what is exactly an Arduino board and we will analyze all of his characteristics.

As ‘’eyes’’ of our system we will use two ultrasonic sensors. They will be responsible for detecting and avoiding obstacles. Also we will study what exactly an ultrasonic sensor is, how do they work and what characteristics they have.

The l239d chip will be the ‘’body’’ of our robot because it’s responsible for his movement. At this chapter we will see all the details about this chip, we will talk why it is so popular and how it can manage to control the speed and the direction of one dc motor.

When we done with the hardware of our system, the next step is to exam how we can make our robot works .There will be a full description and photos on how to build the main body of the robot. At the end we will design and explain with much details all the electrical circuit of our system.

The last and most important chapter will be the software of our robot, we will analyze the most important parts of our code and we will discuss what a PID controller is and why we use it.

Key Word: Λέξεις κλειδιά: Arduino , Wall following robot , PID, L239D motor controller, Ultrasonic Sensors.

Ευχαριστίες

Με την ολοκλήρωση της πτυχιακής εργασίας, θα ήθελα να ευχαριστήσω την οικογένειά μου για την στήριξη και την υπομονή που έδειξαν όλα αυτά τα χρόνια κατά την πανεπιστημιακή μου θητεία.

Ένα μεγάλο ευχαριστώ θέλω να πω στους καθηγητές του Πανεπιστημίου και ιδιαίτερα στον επιβλέπων καθηγητή Φραγκούλη Γεώργιο για την πλήρη καθοδήγησή του, τις συμβουλές και τον χρόνο που αφιέρωσε για να ολοκληρώσουμε την εργασία.

Τέλος οφείλω ένα μεγάλο ευχαριστώ στους φίλους και συναδέρφους για όλες τις όμορφες στιγμές και αναμνήσεις που περάσαμε όλα αυτά τα χρόνια.

Δήλωση Πνευματικών Δικαιωμάτων

Δηλώνουμε ρητά ότι, σύμφωνα με το άρθρο 8 του Ν. 1599/1986 και τα άρθρα 2,4,6 παρ.3 του Ν. 1256/1982, η παρούσα Διπλωματική Εργασία με τίτλο «Σχεδίαση και υλοποίηση συστήματος ρομποτικής κατασκευής με υποστήριξη μηχανικής όρασης» καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στα πλαίσια αυτής της εργασίας και αναφέρονται ρητώς μέσα στο κείμενο που συνοδεύουν, και η οποία έχει εκπονηθεί στο Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Πανεπιστημίου Δυτικής Μακεδονίας, υπό την επίβλεψη του μέλους του Τμήματος κ. Φραγκούλη Γεώργιο, αποτελεί αποκλειστικά προϊόν προσωπικής εργασίας και δεν προσβάλλει κάθε μορφής πνευματικά δικαιώματα τρίτων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον. Τα σημεία όπου έχουμε χρησιμοποιήσει ιδέες, κείμενο, αρχεία ή / και πηγές άλλων συγγραφέων, αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή. Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τους συγγραφείς. Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τους συγγραφείς και μόνο.

ΚΕΦΑΛΑΙΟ 1 – Εισαγωγή στο Arduino

1.1 Ιστορία του Arduino

Η ιστορία του Arduino [1][14] ξεκινάει το 2005 στην Ιβρέα, μια κωμόπολη δίπλα στο Τορίνο, από τους ιδρυτές Massimo Banzi και David Cuartielles, με απώτερο σκοπό να κατασκευάσουν έναν φθηνό και αποτελεσματικό μικροελεγκτή για την εποχή εκείνη έτσι ώστε να βοηθήσουν τους φοιτητές να ενταχθούν στον κόσμο των ενσωματωμένων συστημάτων και των ηλεκτρονικών. Το Arduino, χτίστηκε γύρο από την ιδέα της καλωδίωσης ‘’Wiring’’.Η ιδέα αυτή ήταν μια διπλωματική εργασία που την είχε αναλάβει ο προγραμματιστής και καλλιτέχνης Hernando Barragan υπό την καθοδήγηση του Massimo Banzi, στο Ινστιτούτο της Ιβρέας.Σαν πρώτο όνομα του έργου αυτού, οι εφευρέτες επέλεξαν να του δώσουν την ονομασία Arduino of Ivrea,ενώ αξιοσημείωτο θα ήταν να αναφέρουμε πως το Arduino είναι ένα ιταλικό όνομα που στα ελληνικά σημαίνει ΄΄γενναίος φίλος’’.

Από τον Μάιο του 2011 περισσότερα από 300.000 Arduinos είχανε πουληθεί στην αγορά, ενώ την χρονιά 2013 ο αριθμός ξεπέρασε τις 700.000.Στις μέρες μας το Arduino λόγο της χαμηλής τιμής του και της αξιοπιστίας του, είναι διαδεδομένο παγκοσμίως με εκατομμύρια χρήστες να το επιλέγουν για τις ο καθένας για τις δικές του ανάγκες.

1.2 Πλεονεκτήματα Arduino

- Κόστος : Το Arduino είναι μια οικονομική λύση καθώς το κόστος αγοράς μιας πλακέτας είναι πολύ μικρό(περίπου 10 $).Ακόμα η αρχιτεκτονική του είναι ανοιχτή για όποιον θέλει και μπορεί να την ανάπτυξη από μόνος του.
- Λειτουργεί σε όλα τα λειτουργικά συστήματα: Η εφαρμογή του Arduino είναι σχεδιασμένη να τρέχει τόσο στα Windows και στα Linux όσο και στα Mac,κάνοντάς την διαθέσιμη για όλους τους χρήστες.
- Απλό και καθαρό προγραμματιστικό περιβάλλον: Το περιβάλλον του Arduino είναι εύκολο και κατανοητό για τους αρχάριους χρήστες που θέλουν να ασχοληθούν με τα ηλεκτρονικά, αλλά και πολύ ευέλικτο για τους έμπειρους χρήστες για δοκιμάσουν πιο εξειδικευμένα projects.

- Επεκτάσιμη: Τόσο το λογισμικό όσο και το υλικό της πλατφόρμας Arduino είναι ανοιχτά και ελεύθερα για όλους. Αυτό σημαίνει ότι καθημερινά αναπτύσσονται νέες βιβλιοθήκες για την υποστήριξη της πλατφόρμας.

1.3 Λογισμικό Arduino

Το περιβάλλον ανάπτυξης του Arduino, είναι γραμμένο στην γλώσσα προγραμματισμού Java πράγμα που το καθιστά μεταφέρσιμο στα περισσότερα λειτουργικά συστήματα. Στο IDE του Arduino μπορεί κάποιος να βρει έναν μεταγλωττιστή της C και C++ ,ένα τερματικό για σειριακή επικοινωνία μεταξύ υπολογιστή και πλακέτας κ.α. Ο καθένας μπορεί να κατεβάσει δωρεάν την τελευταία έκδοση έκδοση του IDE από το επίσημο site του Arduino.

Η γλώσσα του Arduino είναι βασισμένη στην γλώσσα Wiring ,δηλαδή μια παραλλαγή της C και C++ για μικροελεγκτές ,υποστηρίζοντας όλες τις δομές τόσο της C όσο και της C++.Για την σωστή μεταγλώττιση του προγράμματος από την γλώσσα C/C++ στις κατάλληλες εντολές γλώσσας μηχανής, το IDE του Arduino χρησιμοποιεί τον AVR gcc compiler ,καθώς και το εργαλείο avradude για την αποστολή του προγράμματος στην μνήμη της πλακέτας μας.

1.4 Πλατφόρμες Arduino

Όπως είναι γνωστό η εταιρία έχει δημιουργήσει πάρα πολλές και διάφορες πλακέτες οι οποίες έχουν διαφορετικά χαρακτηριστικά έτσι ώστε να καλύπτουν όλες τις ανάγκες των καταναλωτών [2][14] .Στον παρακάτω πίνακα θα δούμε τις πιο δημοφιλής πλατφόρμες που έχει κυκλοφορήσει η εταιρία καθώς και κάποια κύρια χαρακτηριστικά για την κάθε μια.

Board Name	***Processor***	***VCC(V)***	***Clock (MHz)***	***PWM PINS***	***ANALOG INPUT PINS***	***DIGITAL I/O PINS***
Arduino Uno	ATmega328	5	16	6	6	14
Arduino Nano	ATmega168	5	16	6	8	14
Arduino Leonardo	ATmega32U4	5	16	7	12	20
Arduino Micro	ATmega32U4	5	16	7	12	20
Arduino Fio	ATmega328P	3.3	8	6	8	14
Arduino Duemilanove	ATmega168	5	16	6	6	14
Arduino Mega	ATmega1280	5	16	15	16	54
ArduinoBT	ATmega328	5	16	6	6	14

Πίνακας 1.1: Γενικά χαρακτηριστικά διάφορων πλακετών Arduino

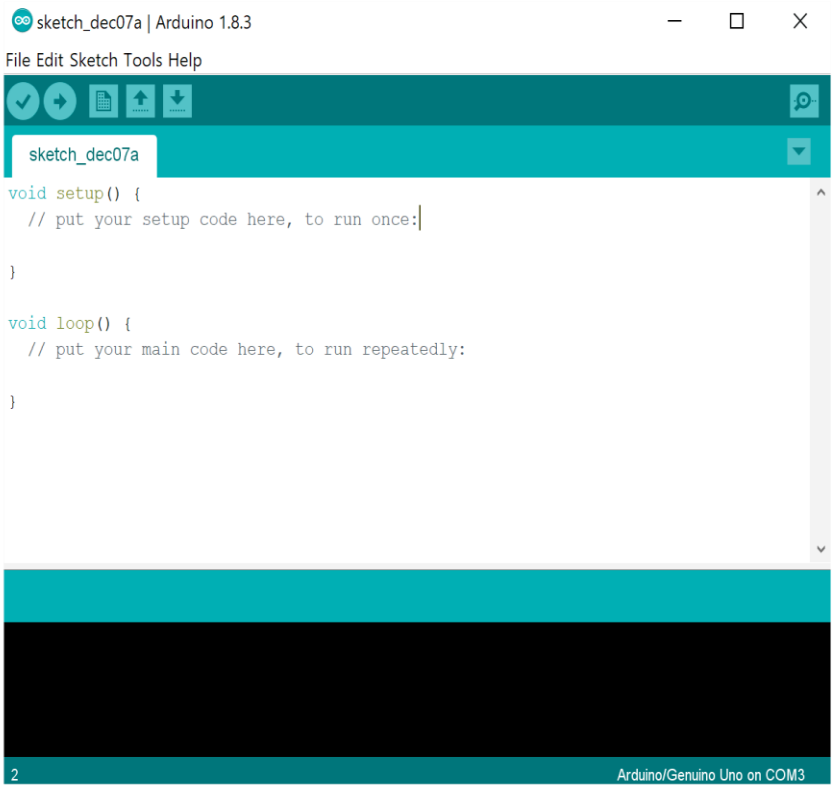
Είναι σημαντικό σε αυτό το σημείο να σημειωθεί πως για κάθε Arduino, υπάρχει η δυνατότητα πρόσθεσης ενός Arduino Shield.Το Arduino Shield ουσιαστικά είναι ένα κύκλωμα σε μια πλακέτα, που το κουμπώνουμε στο πάνω από το Arduino και με αυτόν τον τρόπο επεκτείνουμε τις δυνατότητές του.

Στην παρούσα διπλωματική εργασία, για την δημιουργία του Wall following robot, χρησιμοποιήσαμε ως καρδία του συστήματος τον μικροελεγκτή Arduino Uno , λόγο χαμηλού κόστους, καλής απόδοσης αλλά και λόγο διαθεσιμότητας στην ελληνική αγορά.

1.5 Περιβάλλον ανάπτυξης (IDE) Arduino

Για τον προγραμματισμό και την διαχείριση του Arduino ο χρήστης χρειάζεται να έχει εγκατεστημένο το Arduino IDE[3].Στο περιβάλλον αυτό ο χρήστης μπορεί να βρει, έναν χώρο για την συγγραφή του κώδικα ή αλλιώς στην ορολογία του Arduino sketch, με συντακτική χρωματική σήμανση.Η μαύρη κονσόλα δείχνει την έξοδο του προγράμματος από το Arduino IDE, συμπεριλαμβάνοντας μηνύματα λάθους και errors.

Επιπρόσθετα για τους νέους χρήστες διαθέτει πολλαπλά παραδείγματα για εξάσκηση και εκμάθηση και πολλές εύχρηστες βιβλιοθήκες. Τέλος διαθέτει διάφορα μενού και κουμπιά, το καθένα με την δικιά του λειτουργία.



Εικόνα 1.1: Περιβάλλον Arduino (IDE)

 Verify / compile κουμπί: Πατώντας το κουμπί αυτό, ο compiler τρέχει το πρόγραμμα και σε περίπτωση κάποιου λάθους το εμφανίζει στην μαύρη κονσόλα. Αν δεν υπάρχει κάποιο σφάλμα εμφανίζει κατάλληλο μήνυμα.

 Upload κουμπί: Η λειτουργία του είναι να ανεβάζει το πρόγραμμα στην πλακέτα μας. Κύρια προϋπόθεση το Arduino να είναι συνδεδεμένο με τον υπολογιστή μέσο USB

 New: Δημιουργία παραθύρου για νέο sketch

 Open: Ανοίγει κάποιο από τα παρουσιαζόμενα sketch

 Save: Αποθηκεύει το Sketch που έχουμε δημιουργήσει

 Serial monitor: Ανοίγει η σειριακή επικοινωνία ώστε να δούμε τα δεδομένα που στέλνει ο μικροελεγκτής.

1.6 Εντολές και δομές του Arduino

Όπως προαναφέραμε η γλώσσα προγραμματισμού του Arduino είναι βασισμένη στην C++.Στους παρακάτω πίνακες διακρίνονται οι περισσότερες δομές αλλά και εντολές [4][15] που χρειάζεται κάποιος για να προγραμματίσει έναν μικροελεγκτή Arduino:

- Έλεγχο ροής

If	Έλεγχος για μια συνθήκη
If…. else	Έλεγχος για πολλαπλές συνθήκες
For	Δομή επανάληψης
While	Δομή επανάληψης
Do…. while	Δομή επανάληψης
Break	Διακοπή επανάληψης
Switch	Έλεγχος περιπτώσεων
Continue	Συνέχεια της επανάληψης
Return	Επιστροφή από μια συνάρτηση

- Λογικοί τελεστές

&&	Λογική σύζευξη
||	Λογική διάζευξη
!	Λογική άρνηση

- Αριθμητικοί τελεστές

=	Ισότητα
+	Πρόσθεση

-	Αφαίρεση
/	Διαίρεση
*	Πολλαπλασιασμός
%	Ακέραια διαίρεση

- Τύποι δεδομένων

Char	Χαρακτήρας 8 ψηφίων
Int	Ακέραιος 8 ψηφίων
Float	Αριθμός απλής ακρίβειας με υποδιαστολή
Boolean	Λογική δυαδική τιμή
Unsigned char	Χαρακτήρας 8 ψηφίων
Byte	Μη προσημασμένος χαρακτήρας 8 ψηφίων
Unsigned int	
Long	Ακέραιος αριθμός 32 ψηφίων
String	Αλφαριθμητικό
Double	Αριθμός διπλής ακρίβειας με υποδιαστολή
Unsigned long	Ακέραιος αριθμός 32 ψηφίων

- Σταθερές

HIGH	Τιμή υψηλής στάθμης για μια επαφή
LOW	Τιμή χαμηλής στάθμης για μια επαφή
INPUT	Ορισμός εισόδου για μια επαφή
OUTPUT	Ορισμός εξόδου για μια επαφή
True	Αληθής συνθήκη
False	Ψευδής συνθήκη

- Συνάρτηση εισόδου-εξόδου

PinMode()	Ορίζει μια είσοδο είτε σαν είσοδο είτε σαν έξοδο

- Συναρτήσεις για ψηφιακές επαφές

digitalWrite()	Δίνει μια τιμή σε μια ψηφιακή επαφή
digitalRead()	Διαβάζει μια τιμή απο ψηφιακη επαφή

- Συναρτήσεις για αναλογικές επαφές

AnalogWrite()	Ορίζει την τιμή PWM στις αναλογικές επαφές
AnalogRead()	Διαβάζει τιμές από αναλογικές επαφές
AnalogReference()	Ορίζει την τάση αναλογικής αναφοράς

- Τελεστές σύγκρισης

==	Ισότητα
=>	Μεγαλύτερο ή ίσο
=<	Μικρότερο ή ίσο
<	Μικρότερο
>	Μεγαλύτερο
!=	Αδιάφορο

- Συναρτήσεις χρόνου

millis	Διάρκεια εκτέλεσης προγράμματος σε ms
micros	Διάρκεια εκτέλεσης προγράμματος σε μs
delay(x)	Καθυστέρηση προγράμματος για x ms
DelayMicroseconds(y)	Καθυστέρηση προγράμματος για y μs

- Interrupts

attachInterrupt	Ενεργοποίηση ρουτίνας εξυπηρέτησης διακοπτών
detachInterrupt	Απενεργοποίηση ρουτίνας εξυπηρέτησης διακοπτών
Interrupts	Ενεργοποιεί τα σήματα διακοπής
noInterrupts	Απενεργοποιεί τα σήματα διακοπής

Κεφάλαιο 2 - Arduino Uno

2.1 Γενικές πληροφορίες για το Arduino Uno

Η καρδιά όλου του συστήματός μας θα είναι ο μικροελεγκτής Arduino Uno[5][16], οπότε θα ήταν χρήσιμο να δούμε πιο αναλυτικά τα χαρακτηριστικά του. Αρχικά όσον αφορά την αρχιτεκτονική του, το Arduino Uno είναι μια πλατφόρμα βασισμένη στον μικροεπεξεργαστή ATmega328P.Ο μικροεπεξεργαστής αυτός είναι ο ‘’εγκέφαλος’’ όλης της πλακέτας και είναι ρυθμισμένος έτσι ώστε να ελέγχει τους 14 ψηφιακούς ακροδέκτες, 6 εκ των οποίων μπορούν να χρησιμοποιηθούν και σαν έξοδοι PWM(διαμόρφωση πλάτους παλμού), και τους 6 αναλογικούς ακροδέκτες που βρίσκονται πάνω στην πλακέτα. Στις υποδοχές αυτές τοποθετούνται όλα τα εξωτερικά στοιχεία καθώς και όλοι οι αισθητήρες που θέλει να συνδέσει ο χρήστης με την πλακέτα του.

Επάνω στην πλακέτα μπορεί κάποιος να βρει ακόμη ένα κουμπί reset ,σε περίπτωση βραχυκυκλώματος μια υποδοχή για εξωτερική τροφοδοσία αλλά και μια θύρα USB η οποία μπορεί είτε να λειτουργήσει ως πηγή τροφοδοσίας για την πλακέτα, αλλά και μέσο αυτής γίνεται η μεταφορά του πηγαίου κώδικα από τον υπολογιστή στο Arduino

2.2 Βασικά χαρακτηριστικά του Arduino Uno

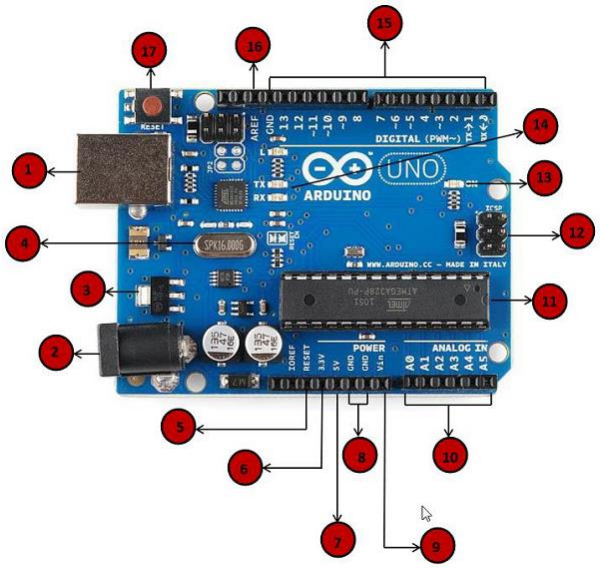
Μερικά από τα κύρια χαρακτηριστικά της πλακέτας Arduino Uno παρουσιάζονται παρακάτω:

Τάση λειτουργίας	5 Volt
Συνιστάμενη τάση εισόδου	Από 7 Volt έως 12 Volt
Όρια τάσης εισόδου	Από 6 Volt έως 20 Volt
Αναλογικές θήρες	6
Ψηφιακές θήρες	14
Flash Memory	32 KB
SRAM	2KB
Clock Speed	16 MHz
EEPROM	1KB
DC ρεύματος για κάθε I/O θήρα	40mA

Πίνακας 2.1: Κύρια χαρακτηριστικά πλακέτας Arduino Uno

2.3 Arduino Uno diagram

Αναλυτικά στο παρακάτω πίνακα θα δούμε πώς ακριβώς μοιάζει η πλακέτα μας, που βρίσκεται η κάθε είσοδος και γενικά όλα όσα βλέπουμε πάνω στο Arduino.



Εικόνα 2.1:Arduino Uno board

1	Υποδοχή USB
2	Υποδοχή τροφοδοσίας
3	Ρυθμιστής Τάσης
4	Κρυσταλλικός ταλαντωτής
5	Υποδοχή για reset button
6	Ακροδέκτης ρεύματος 3.3 Volt (έξοδος)
7	Ακροδέκτης ρεύματος 5 Volt (έξοδος)
8	Γείωση
9	Ακροδέκτης ρεύματος(είσοδος)
10	Αναλογικοί ακροδέκτες
11	Μικροεπεξεργαστής ATmega328P
12	ICSP για ATmega328
13	Led ενεργοποίησης
14	TX / NX Leds
15	Ψηφιακοί ακροδέκτες
16	Γείωση και AREF ακροδέκτες
17	Κουμπί επανεκκίνησης

Πίνακας 2.2: Απεικόνιση στοιχείων στην πλακέτα

2.4 Μνήμη

Η πλακέτα Arduino Uno διαθέτει στο σύνολο τρείς βασικές μνήμες [6][16] :

1. Flash memory: Σε αυτή την μνήμη αποθηκεύεται το πρόγραμμα το οποίο εκτελούμε. Το μέγεθος της συγκεκριμένης μνήμης είναι 32Kbytes
2. SRAM: Σε αυτή την μνήμη αποθηκεύονται προσωρινά όλα τα στατικά δεδομένα καθώς και οι μεταβλητές του προγράμματός μας. Το μέγεθος της μνήμης είναι 2Kbytes
3. EEPROM: Σε αυτή την μνήμα αποθηκεύονται όλες οι τιμές των μεταβλητών όταν η συσκευή σβήσει, δηλαδή γίνεται αποθήκευση μακροπρόθεσμων πληροφοριών. Το μέγεθος της μνήμης είναι 1Kbyte.

Σημαντικό είναι να σημειώσουμε πως οι μνήμες Flash και EEPROM είναι σταθερές, δηλαδή οι πληροφορίες παραμένουν ακόμα και μετά την απενεργοποίηση του ρεύματος, σε αντίθεση με την μνήμη SRAM όπου οι πληροφορίες χάνονται όταν απενεργοποιηθεί το ρεύμα.

2.5 Τροφοδοσία πλακέτας

Υπάρχουν δύο τρόποι για να τροφοδοτήσουμε την πλακέτα Arduino,είτε με την χρήση μιας εξωτερικής πηγής ενέργειας, που την συνδέουμε στην υποδοχή 2(βλέπε εικόνα 2.1), είτε απευθείας από τον υπολογιστή με USB ,μέσο την υποδοχής 1(βλέπε εικόνα 2.1).

Με την έννοια εξωτερικής πηγής ενέργειας, εννοούμε ότι για να δώσουμε τάση στην πλακέτα μας, χρησιμοποιούμε είτε μια μπαταρία είτε έναν μετασχηματιστή των 9V από 220V. Για την σωστή τροφοδοσία της πλακέτας , θα πρέπει να γνωρίζουμε πως ο μικροελεγκτής μας λειτουργεί με εξωτερική παροχή των 6 έως 20 Volt. Αυτό σημαίνει πως εάν η τάση είναι μικρότερη των 6 Volt τότε οι ακροδέκτες εξόδου δεν θα μπορέσουν να παράγουν τάση (μιας και όλες οι έξοδοι του Arduino παράγουν τάση 5 Volt).Από την άλλη, αν τροφοδοτήσουμε την πλακέτα με περισσότερα από 12 Volt, τότε υπάρχει

πιθανότητα να υπερθερμάνουμε την πλακέτα και να την αχρηστέψουμε. Με λίγα λόγια μια ιδανική τάση για να τροφοδοτήσουμε την πλακέτα μας είναι 9-12Volt.

Ακροδέκτες τροφοδοσίας:

1. VIN: Εδώ συνδέεται μια εξωτερική πηγή τροφοδοσίας, όπως μια μπαταρία καθώς είναι ακροδέκτης μη σταθεροποιημένης τάσης.
2. 5V:Είναι ακροδέκτης που βγάζει σταθερή τάση 5V,και είναι υπεύθυνο για την τροφοδοσία του μικροελεγκτή καθώς και άλλων στοιχείων της πλακέτας
3. 3V3: Πηγή τροφοδοσίας 3.3V παραγόμενη από το FTDI chip.
4. GND: Γείωση.

Κεφάλαιο 3 - Ultrasonic Sensor HC-SR04

3.1 Γενικά

Ένα πολύ σημαντικό κομμάτι του συστήματος μας είναι τα Ultrasonic sensors HC-SR04 ή αλλιώς στα ελληνικά οι αισθητήρες απόστασης – αισθητήρες υπερήχων [7].Ο υπερηχητικός αισθητήρας είναι ένα όργανο που μετράει την απόσταση από ένα αντικείμενο χρησιμοποιώντας υπερηχητικά κύματα και θα αποτελέσουν τα ‘’μάτια’’ του ρομπότ μας, καθώς θα είναι υπεύθυνα στο να ανιχνεύουν τοίχους. Στην δικιά μας εργασία θα χρησιμοποιήσουμε δύο τέτοιους αισθητήρες ,έναν από την αριστερή μεριά έτσι ώστε να ‘’βλέπει’’ το τοίχος και να κινείται παράλληλα με αυτόν και έναν στο μπροστινό μέρος του αμαξιδίου για τον εντοπισμό και την αποφυγή εμποδίων. Όμως τι ακριβώς είναι ένας Ultrasonic sensor;

Μιλάμε για έναν μικρό, φθηνό και εύκολο στην χρήση αισθητήρα που μπορεί να χρησιμοποιηθεί σε πολλαπλά έργα, και προσφέρει ανίχνευση αντικειμένων από τα 2cm μέχρι και 4m με την ακρίβειά του να ανέρχεται στα 3mm.Όσον αφορά την δομή του, αποτελείται από δύο μετατροπείς υπερήχων. Ο ένας είναι ο λεγόμενος πομπός υπεύθυνος για την παραγωγή των υπερηχητικών παλμών και ο άλλος είναι ο δέκτης ,υπεύθυνος για λήψη παλμών.

3.2 Χαρακτηριστικά

Μερικά από τα κύρια χαρακτηριστικά ενός HC-SR04 είναι:

Τάση Λειτουργίας	5 Volt
Συχνότητα Λειτουργίας	40 Khz
Ρεύμα Λειτουργίας	10mA(max)
Γωνία ανίχνευσης	30 μοίρες
Γωνία μέτρησης	15 μοίρες
Ψηφιακή έξοδος	5 Volt
Θερμοκρασίες Λειτουργίας	Από -15 °C έως 70 °C
Εύρος ανίχνευσης αντικειμένου	Από 2cm έως 400cm
Εύρος ακρίβειας	3mm
Διαστάσεις	45 x 20 x 15mm

Πίνακας 3.1: Χαρακτηριστικά HC-SR04

3.3 Διάγραμμα

Εικόνα 3.1:Ultrasonic sensor HC-SR04

Όπως βλέπουμε και στην εικόνα 3.1 ο αισθητήρας διαθέτει 4 ακροδέκτες. Πιο συγκεκριμένα:

1-VCC: Ο ακροδέκτης αυτός είναι υπεύθυνος για την τροφοδοσία του αισθητήρα με τάση 5V.Μπορεί να συνδεθεί απευθείας στο Arduino,καθώς είπαμε ότι η πλακέτα εξάγει τάση 5V.

2-Trig:Ο συγκεκριμένος είναι ένας ακροδέκτης εισόδου, καθώς χρησιμοποιείται για την εκκίνηση της μέτρησης μεταδίδοντας ένα κύμα υπερήχων. Πρέπει να σημειωθεί πως ο ακροδέκτης αυτός πρέπει να είναι ενεργός για τουλάχιστον 10ms.

3-Echo:Το συγκεκριμένο pin του αισθητήρα είναι ένας ακροδέκτης εξόδου το οποίο γίνεται ενεργό για ένα χρονικό διάστημα το οποίο ισούται με τον χρόνο που έκανε να λάβει πίσω το παραγόμενο παλμό.

4-GRN:Σε αυτόν τον ακροδέκτη γίνεται η γείωση του συστήματος.

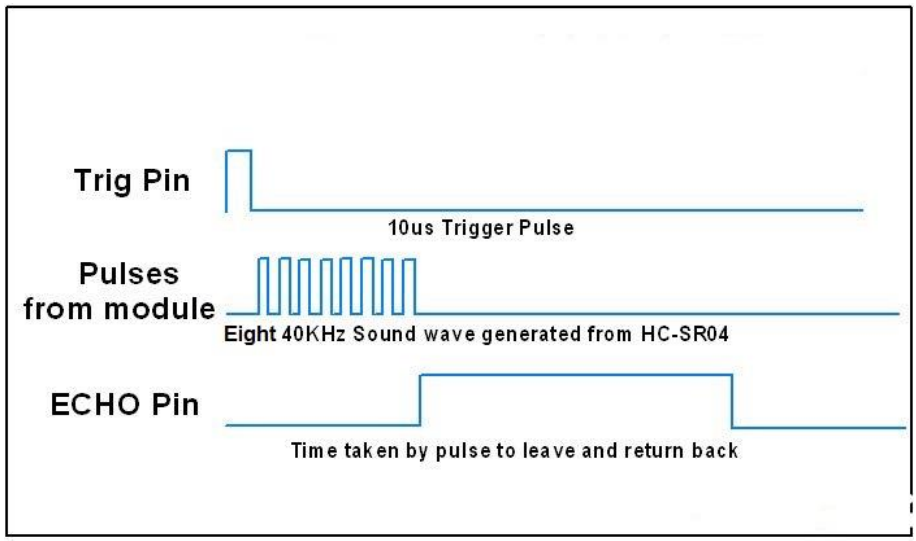
3.4 Τρόπος λειτουργίας

Ο τρόπος λειτουργίας του Ultrasonic sensor HC-SR04 είναι παρόμοιος με την λειτουργία ενός radar ή sonar. Οι αισθητήρες αυτοί λειτουργούν στέλνοντας ένα ηχητικό κύμα σε συχνότητα πάνω από το εύρος της ανθρώπινης ακοής.

Όλη η διαδικασία ξεκινάει δίνοντας έναν παλμό διάρκειας τουλάχιστον 10 μs στον ακροδέκτη Trig του ultrasonic αισθητήρα. Ως αποτέλεσμα αυτής της ενέργειας, είναι η δημιουργία μιας σειράς οκτώ παλμών στα 40 Khz από τον αισθητήρα. Ο παλμός αυτός ξεκινάει να ταξιδεύει στον αέρα μακριά από τον πομπό με απώτερο σκοπό να βρει μπροστά του κάποιο εμπόδιο.

Εν τω μεταξύ μετά την δημιουργία των οκτώ παλμών, ο ακροδέκτης Echo του αισθητήρα ενεργοποιείται (δηλαδή γίνεται HIGH),και παραμένει σε αυτή την λειτουργία μέχρι ο δέκτης να δεχτεί πίσω κάποιο κύμα ή μέχρι να περάσουν 38mS.Τα 38 Ms αποτελούν κριτήριο, πως οι παλμοί ναι μεν ταξίδεψαν στον αέρα αλλά δεν έχουν βρει κάποιο εμπόδιο στο εύρος λειτουργίας του αισθητήρα.

Στην περίπτωση που οι παλμοί καταφέρουν και βρουν κάποιο εμπόδιο μπροστά τους, τότε αντανακλώνται προς τα πίσω με τελικό προορισμό τον δέκτη του αισθητήρα. Μόλις ο δέκτης δεχτεί τους παλμούς ,ο ακροδέκτης Echo απενεργοποιείται(δηλαδή γίνεται LOW) και έχει ως αποτέλεσμα την δημιουργία ενός νέου παλμού όπου το πλάτος του κυμαίνεται μεταξύ 150μS και 25 mS ανάλογα με τον χρόνο που χρειάστηκε για να λάβει την σειρά των οκτώ παλμών.

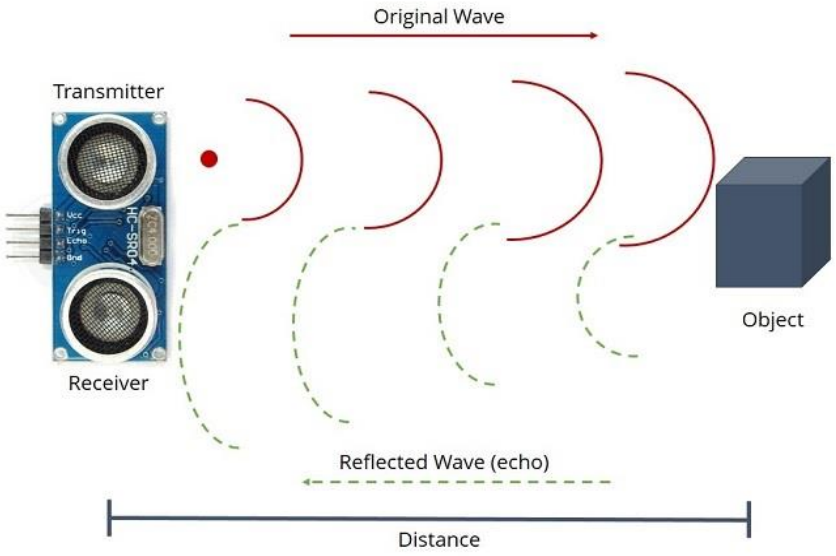


Εικόνα 3.2: Ultrasonic HC-SR04 timing diagram

Μέσο του πλάτους αυτού του νέου κύματος μπορούμε άμεσα να υπολογίσουμε την απόσταση μεταξύ του αισθητήρα και του εμποδίου, χρησιμοποιώντας κάποια βασικά μαθηματικά. Ο πυρήνας για τον υπολογισμό της απόστασης είναι η μαθηματική εξίσωση που ορίζει την απόσταση ίση με την ταχύτητα πολλαπλασιασμένη με τον χρόνο.

Σχέση 3.1: Απόσταση = Ταχύτητα * Χρόνο

Για να το κατανοήσουμε καλύτερα ας δούμε ένα παράδειγμα. Ας υποθέσουμε ότι απέναντι από τον αισθητήρας μας υπάρχει ένα εμπόδιο και θέλουμε να βρούμε ποια είναι η μεταξύ τους απόσταση, γνωρίζοντας πως ο παλμός που έχει δεχτεί ο δέκτης (Reflected Wave εικόνας 3.3) είναι 200μs.



Εικόνα 3.3:Απεικόνιση λειτουργίας HC-SR04

Όπως προαναφέραμε και προηγουμένως, θα χρησιμοποιήσουμε την σχέση3.1. Παρατηρώντας την σχέση βλέπουμε ότι γνωρίζουμε μόνο τον χρόνο και όχι την ταχύτητα. Η παράμετρος της ταχύτητας είναι εύκολο να βρεθεί μιας και γνωρίζουμε ότι το κύμα ταξιδεύει στον αέρα, συνεπώς θα έχει ταχύτητα 340m/s.Σημαντικό είναι να θυμηθούμε πως ο παλμός υποδεικνύει το χρόνο που χρειάστηκε για να σταλεί και να αντανακλαστεί το σήμα, συνεπώς θα χρειαστεί να κάνουμε και την απαραίτητη διαίρεση. Μετατρέποντας τα 340 m/s σε cm/μs και μέσο της σχέσης 3.1 έχουμε ότι:

Απόσταση = (0.034cm/μs * 200 μs) / 2 = 3,4cm.

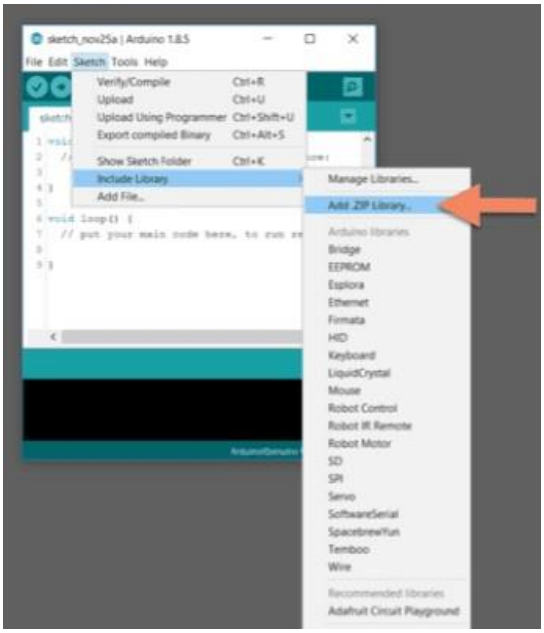
Η όλη παραπάνω διαδικασία αν και φαίνεται περίπλοκή δεν είναι καθόλου ,καθώς με τις κατάλληλες βιβλιοθήκες που θα χρησιμοποιήσουμε στο πρόγραμμά μας, ο υπολογισμός των αποστάσεων θα γίνεται αυτόματα.

3.5 NewPing Βιβλιοθήκη

Η βιβλιοθήκη NewPing[8], είναι προσχεδιασμένη για τους αισθητήρες ultrasonic και θα την χρησιμοποιήσουμε και εμείς στην δικιά μας εργασία. Είναι μια απλή, εύχρηστή και εύκολη στην χρήση βιβλιοθήκη που μέσο των μεθόδων της μετράει αυτοματοποιημένα την απόσταση μεταξύ του αισθητήρα και ενός εμποδίου.

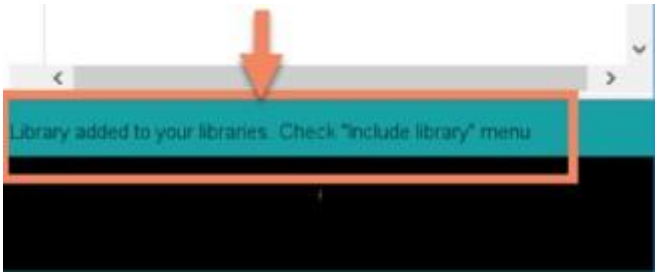
Για αρχή πρέπει να κάνουμε την εγκατάσταση την βιβλιοθήκης μιας και το Arduino IDE δεν την συμπεριλαμβάνει από μόνο του.

1. Πρώτο βήμα: Με μια γρήγορη αναζήτηση στο διαδίκτυο κατεβάζουμε το NewPing library zip αρχείο.
2. Δεύτερο βήμα: Αφού έχουμε κατεβάσει το zip αρχείο αμέσως μετά πρέπει να το κάνουμε unzip και να το τοποθετήσουμε στον φάκελο Documents>Arduino>Libraries.
3. Τρίτο βήμα: Τώρα πρέπει να συμπεριλάβουμε την βιβλιοθήκη στο Arduino IDE.Πηγαίνουμε στο IDE, και πατάμε το μενού sketch>Include Library>Add ZIP library (βλέπε εικόνα 3.4).



Εικόνα 3.4:Εισαγωγή ZIP Library

4. <u>Τέταρτο βήμα</u>: Βρείτε και επιλέξτε το αρχείο που έχουμε αποθήκευση στο βήμα 2
5. <u>Πέμπτο βήμα</u>: Αν όλα πήγαν σωστά, στην μαύρη κονσόλα θα πρέπει να εμφανιστεί ένα σχετικό μήνυμα(βλέπε εικόνα 3.5).



Εικόνα 3.5:Μήνυμα εγκατάστασης βιβλιοθήκης

Τώρα που έχουμε και επίσημα εγκαταστήσει την βιβλιοθήκη μας, θα δούμε πως να δηλώσουμε έναν ultrasonic αισθητήρα καθώς και κάποιες από τις κύριες μεθόδους της βιβλιοθήκης. Η δήλωση γίνεται με την βοήθεια του constructor της βιβλιοθήκης:

<u>NewPing sonar (trigger_pin, echo_pin, MAX_CM_DISTANCE).</u>

όπου το trigger_pin θα δηλώνει τον αριθμό του ακροδέκτη στο Arduino που έχουμε συνδέσει το trig pin του αισθητήρα, το echo_pin τον αντίστοιχο αριθμό του ακροδέκτη

echo pin και το MAX_CM_DISTANCE θα δηλώνει την μέγιστη απόσταση που θέλουμε να μετράει ο αισθητήρας μας.

Για παράδειγμα ο NewPing sonar(11,12,250),θέλει να μας δηλώσει έναν αισθητήρα όπου ο ακροδέκτης trig είναι συνδεδεμένος στην είσοδο 11 του Arduino , ο ακροδέκτης echo στην 12 και θέλουμε να μετράει απόσταση μέχρι τα 250cm.

Στον παρακάτω πίνακα απεικονίζονται μερικές κύριες μέθοδοι της βιβλιοθήκης:

sonar.ping_in()	Επιστρέφει την απόσταση σε ίντσες ή μηδέν σε περίπτωση που δεν υπάρχει εμπόδιο
Sonar.ping_cm()	Επιστρέφει την απόσταση σε εκατοστά ή μηδέν σε περίπτωση που δεν υπάρχει εμπόδιο
sonar.ping()	Επιστρέφει τον χρόνο σε ms που έκανε ο δέκτης να δεχτεί το κύμα του πομπού
sonar_convert_in(echotime)	Μετατρέπει τον χρόνο echo από milliseconds σε ίντσες
sonar_convert_cm(echotime)	Μετατρέπει τον χρόνο echo από milliseconds σε εκατοστά

Πίνακας 3.2: Μέθοδοι και συναρτήσεις της NewPing βιβλιοθήκης

Κεφάλαιο 4 – Motor controller L293D

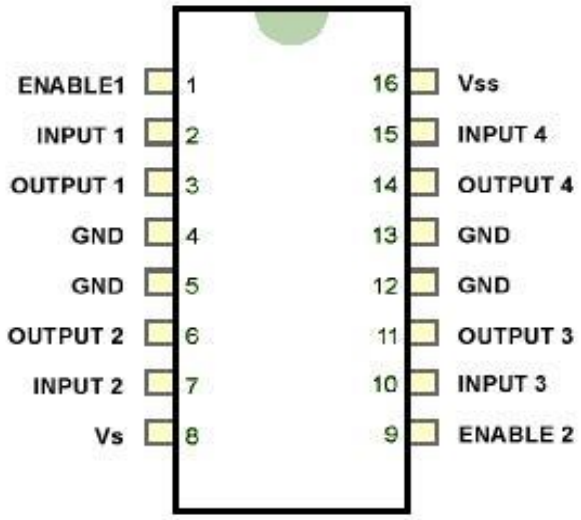
4.1 ΓΕΝΙΚΑ

Ένα επίσης σημαντικό κομμάτι που πρέπει να αναλύσουμε είναι το τσιπ L293D[9][10].Με το εξάρτημα αυτό θα μπορούμε να ρυθμίσουμε τους DC κινητήρες του αμαξιδίου μας να κινούνται προς όποια κατεύθυνση θέλουμε. Το L293D διαθέτει στο εσωτερικό του τέσσερις half H-Bridge που μπορούν όμως να λειτουργήσουν και ως δύο Full H-Bridge.Αυτό σημαίνει ότι μπορεί να οδηγήσει είτε τέσσερις DC κινητήρες μόνο με έλεγχο ταχύτητας είτε δύο κινητήρες με έλεγχο ταχύτητας αλλά και κατεύθυνσης. Στην παρούσα εργασία θα αξιοποιήσουμε την λειτουργία του ως Full H-bridge,μιας και θέλουμε να ελέγχουμε τόσο την ταχύτητα όσο και την κατεύθυνση του κινητήρα μας.

Το εξάρτημα διαθέτει 16 ΄΄ποδαράκια΄΄ καθένα υπεύθυνο για μια λειτουργία, μπορεί να παρέχει ρεύμα 600 mA ανά κανάλι και μπορεί να δώσει τάση σε κινητήρες από 4,5 έως 36V.Σημαντικό είναι να αναφέρουμε ότι το τσιπ διαθέτει ενσωματωμένες διόδους ανάκαμψης υπεύθυνες για την αποφυγή ζημιών όταν ο κινητήρας είναι απενεργοποιημένος.

Εικόνα 4.1: H-Bridge L239D

4.2 Διάγραμμα



Εικόνα 4.2: Διάγραμμα L239D

1-Enable1: Ο ακροδέκτης αυτός μπορεί είτε να είναι ενεργός είτε ανενεργός (HIGH ή LOW) και ανάλογα με την κατάστασή του ενεργοποιεί ή απενεργοποιεί το αριστερό μέρος του chip.

2-INPUT1: Είναι ένας ακροδέκτης που είναι υπεύθυνος για τον έλεγχο κατεύθυνσης του αριστερού κινητήρα. Όταν η κατάστασή του είναι HIGH τότε το ρεύμα βγαίνει από τον ακροδέκτη OUTPUT1.

3-OUTPUT1: Είναι μια έξοδος όπου συνδέεται με το ένα άκρο του κινητήρα.

4,5-GND: Γειώσεις.

6-OUTPUT2:Είναι μια έξοδος που συνδέεται στο άλλο άκρο του κινητήρα.

7-INPUT2:Είναι υπεύθυνος για τον έλεγχο κατεύθυνσης του αριστερού κινητήρα. Όταν η κατάστασή του είναι HIGH τότε το ρεύμα βγαίνει από τον ακροδέκτη OUTPUT2.

8-Vs: Είναι η τάση που θα τροφοδοτεί τους κινητήρες.

16-Vss:Είναι η τάση που θα τροφοδοτεί το τσιπ.

15-INPUT4: Είναι ένας ακροδέκτης υπεύθυνος για τον έλεγχο κατεύθυνσης του δεξιού κινητήρα. Όταν η κατάστασή του είναι HIGH τότε το ρεύμα βγαίνει από τον ακροδέκτη OUTPUT4.

14:OUTPUT4:Συνδέεται με το ένα άκρο του δεξιού κινητήρα

13,12-GND: Γειώσεις

11-OUTPUT3:Έξοδος που συνδέεται με το άλλο άκρο του δεξιού κινητήρα.

10-INPUT3: Υπεύθυνος για τον έλεγχο κατεύθυνσης του αριστερού κινητήρα. Όταν η κατάστασή του είναι HIGH το ρεύμα βγαίνει από τον ακροδέκτη OUTPUT3.

9-ENABLE2: Ο ακροδέκτης αυτός μπορεί είτε να είναι ενεργός είτε ανενεργός (HIGH ή LOW) και ανάλογα με την κατάστασή του ενεργοποιεί ή απενεργοποιεί το δεξιό μέρος του chip.

4.3 Τεχνικά χαρακτηριστικά

Τάση τροφοδοσίας,Vs	36V
Τάση εξόδου, Vcc	36V
Τάση εισόδου, Vi	7V
Μέγιστο ρεύμα ανά κανάλι	600mA
Εύρος τάσης	4.5V έως 36V
Εύρος τάσης εξόδου	-3V έως +3V
Peak ρεύματος εξόδου	1.2A
Συνεχές ρεύμα εξόδου	0/6 A

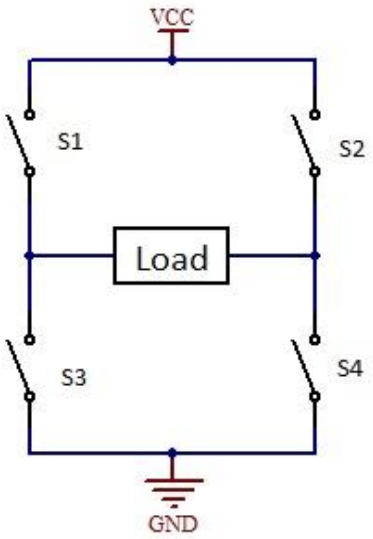
Πίνακας 4.1: Χαρακτηριστικά L239D

4.4 Τρόπος λειτουργίας

Το συγκεκριμένο τσιπ χρησιμοποιεί την ίδια λειτουργία με μια H-Bridge για τον ορισμό

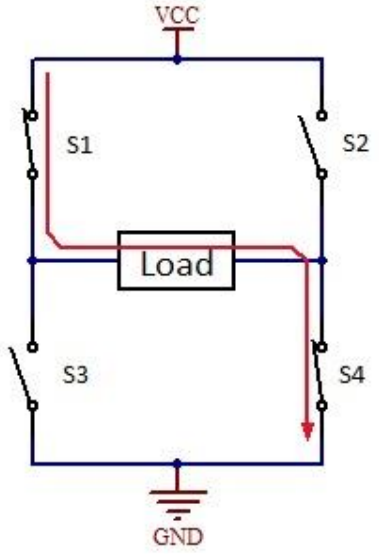
της ταχύτητα και της κατεύθυνση των κινητήρων. Εάν θέλουμε να αλλάξουμε κατεύθυνση σε έναν dc κινητήρα, δεν έχουμε παρά μόνο να αλλάξουμε την πολικότητα της τάσης στα άκρα του. Αυτός είναι κατά κύριο λόγο και τρόπος λειτουργίας του ln239d.Ας τα δούμε όμως λίγο πιο αναλυτικά.

Μια H-Bridge διαθέτει τέσσερις διακοπές ή αλλιώς τρανζίστορ με τον κινητήρα να βρίσκετε στην μέση σχηματίζοντας το αγγλικό γράμμα ‘’H’’.



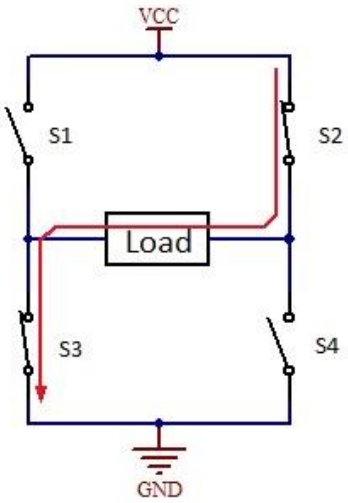
Εικόνα 4.3: Κύκλωμα H-Bridge

Ας υποθέσουμε ότι στην εικόνα 4.3 απεικονίζεται το κύκλωμα H-bridge με το LOAD να είναι ο κινητήρας μας και τα s1,s2,s3 και s4 να είναι τα transistors.Εάν θέλουμε να οδηγήσουμε τον κινητήρα προς την μια κατεύθυνση δεν έχουμε παρά να κλείσουμε τους διακόπτε s1 και s2(βλέπε εικόνα 4.4),δημιουργώντας ένα κλειστό κύκλωμα όπου το ρεύμα διαπερνάει το φορτίο μας(κινητήρας).Κλείνοντας τους δυο αυτούς διακόπτες ουσιαστικά οδηγούμε τον κινητήρα δεξιόστροφα.



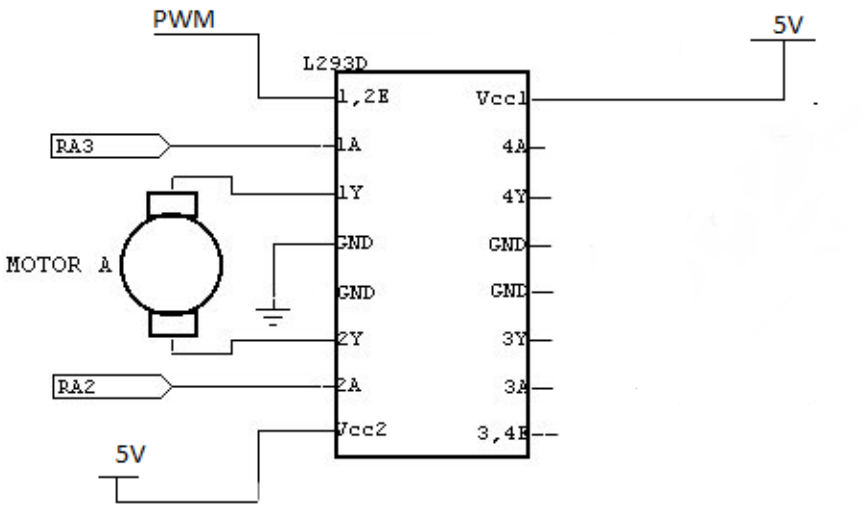
Εικόνα 4.4: Οδήγηση κινητήρα προς την μια κατεύθυνση με την χρήση H-Bridge

Σε περίπτωση όμως που θέλουμε να τον οδηγήσουμε από την άλλη κατεύθυνση αρκεί να κλείσουμε τους διακόπτες s3,s2 και να ανοίξουμε τους s1,s4(βλέπε εικόνα 4.5).Με αυτόν τον τρόπο οδηγούμε τον κινητήρα αριστερόστροφα.



Εικόνα 4.5: Οδήγηση κινητήρα προς την άλλη κατεύθυνση με την χρήση H-bridge.

Ο motor controller L239D εμπεριέχει δύο full H-Bridge κυκλώματα όπως τα παραπάνω, το καθένα υπεύθυνο για έναν κινητήρα. Πέρα όμως από την κατεύθυνση μέσο του τσιπ μπορούμε να ρυθμίσουμε και την ταχύτητα περιστροφής του. Ας δούμε ένα παράδειγμα για να το κατανοήσουμε καλύτερα.



Εικόνα 4.6: Κύκλωμα ενός κινητήρα με τον L239D.

Αρχικά βλέποντας και την εικόνα 4.6, πρέπει να τροφοδοτήσουμε τα pins 16 και 8 με τάση 5V, για την ενεργοποίηση τόσο του motor controller,όσο και του κινητήρα. Στην συνέχεια πρέπει να γειώσουμε έναν από τους δύο ακροδέκτες του L239D ή τον 4 ή τον 5.Στο δικό μας παράδειγμα έχουμε γειώσει τον 4.Επόμενο βήμα είναι να συνδέσουμε τα άκρα του κινητήρα στα pins 3 και 6 αντίστοιχα. Μέσο του ακροδέκτη 1 μπορούμε να ελέγξουμε την ταχύτητα περιστροφής του κινητήρας μας, απλά στέλνοντας PWM κύμα ενώ μέσο των ακροδεκτών 2 και 7 καθορίζουμε την περιστροφή του κινητήρα, ανάλογα με το ποιος ακροδέκτης είναι HIGH.Στον παρακάτω πίνακα αληθείας φαίνονται τα αποτελέσματα του κινητήρα για της διάφορες καταστάσεις που τον ορίζουμε:

1,2 E	**1A**	**2A**	**MOTOR DIRECTION**	**MOTOR SPEED**
0% PWM	LOW	LOW	ΚΑΜΙΑ	0%
50% PWM	HIGH	LOW	ΔΕΞΙΟΣΤΡΟΦΑ	50%
75% PWM	LOW	HIGH	ΑΡΙΣΤΕΡΟΣΤΡΟΦΑ	75%

Πίνακας4.2: Πίνακας αληθείας για έναν κινητήρα

Στο δικό μας ενσωματωμένο σύστημα θα χρησιμοποιήσουμε δύο DC κινητήρες όπου θα οδηγούνε δύο τροχούς για την κίνηση του αμαξιδίου. Με τον ίδιο ακριβώς τρόπο που αναφέραμε στο προηγούμενο παράδειγμα συνδέουμε και τον δεύτερο κινητήρα από την δεξιά μεριά του motor controller.Για τους δύο κινητήρες ο πίνακας αληθείας, αλλά και η αντίδραση που θα έχει το αμαξίδιο είναι[11]:

IN1	IN2	IN3	IN4	CAR DIRECTION
LOW	LOW	LOW	LOW	STOP
HIGH	LOW	HIGH	LOW	ΜΠΡΟΣΤΑ
LOW	HIGH	LOW	HIGH	ΟΠΙΣΘΕΝ
HIGH	HIGH	HIGH	HIGH	STOP
HIGH	LOW	LOW	HIGH	ΔΕΞΙΑ
LOW	HIGH	HIGH	LOW	ΑΙΡΣΤΕΡΑ

Πίνακας4.3: Πίνακας αληθείας για δυο κινητήρες

Άρα λοιπόν για να οδηγήσουμε το ρομπότ μας προς όποια κατεύθυνση θέλουμε, δεν έχουμε πάρα μόνο να αλλάζουμε την τάση των ακροδεκτών 2,7,15 και 10 ανάλογα με τον πίνακα αληθείας.

Κεφάλαιο 5 – Υλοποίηση κατασκευής

5.1 Υλικά κατασκευής

Για την υλοποίηση του συστήματός μας θα χρειαστούμε:

Μηχανολογικά μέρη:

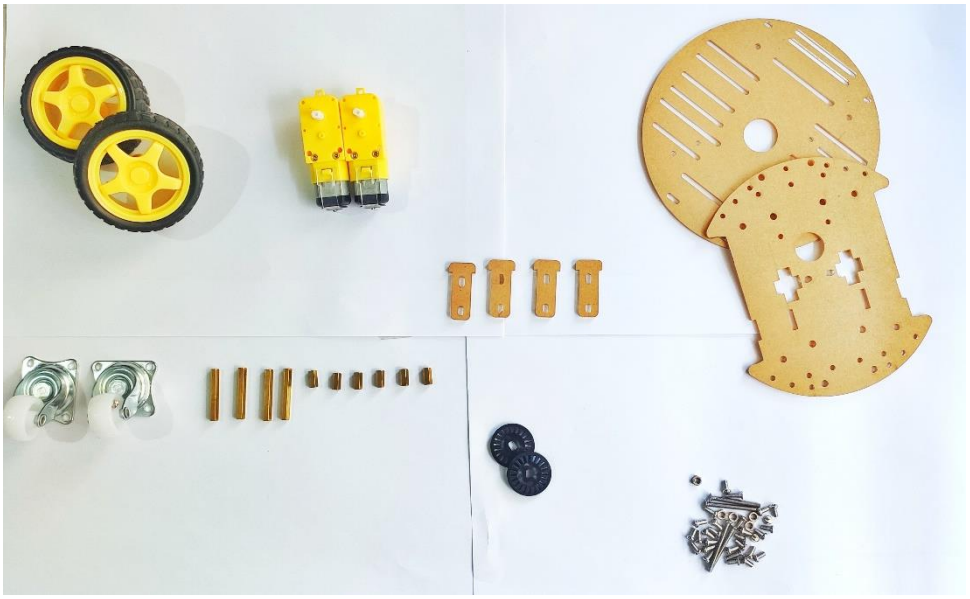
- 1 x Robot kit
- 2 x DC motors
- Βίδες και παξιμάδια
- 2 x ρόδες για σύνδεση με τους κινητήρες
- 2 x βοηθητικές ρόδες

Ηλεκτρολογικά μέρη:

- 1 x Motor controller L293D
- 1 x breadboard
- 2 x Ultrasonic sensor HC-SR04
- 2 x Ultrasonic sensor HC-SR04 holder
- 8 x Μπαταρίες 1300 mAh 1.2V NiMH επαναφορτιζόμενες
- 1 x Θήκη για τις μπαταρίες
- Καλώδια

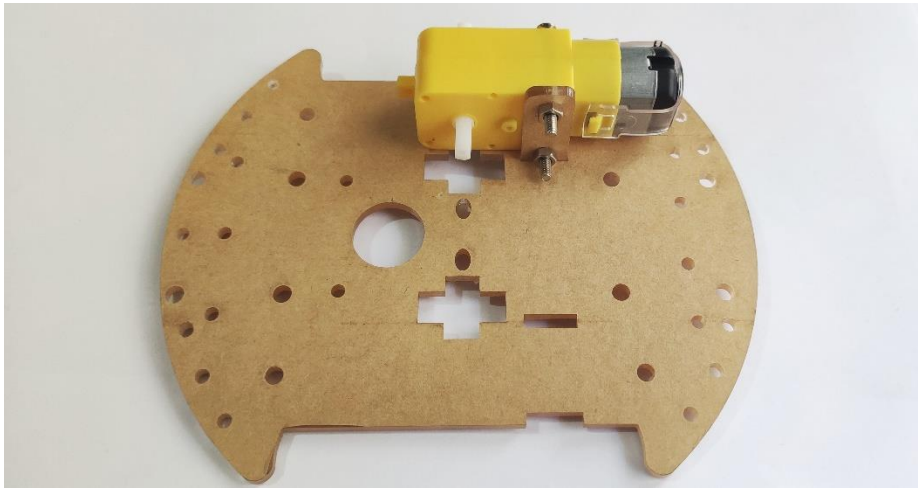
1.2 Κατασκευή αμαξιδίου

Το αμαξίδιό μας είναι ένα φθηνό robot kit το οποίο το προμηθευτήκαμε από το διαδίκτυο σε χαμηλή τιμή. Μέσα σε αυτό θα βρούμε τέσσερις μεγάλους πυλώνες, δυο βοηθητικές ρόδες, δύο dc κινητήρες ,τέσσερα στηρίγματα για να βιδώσουμε πάνω τους δύο κινητήρες, οκτώ μικρούς πυλώνες , δύο ρόδες και πολλές βίδες και παξιμάδια. Στις παρακάτω εικόνες γίνεται η επεξήγηση στο πως να γίνει η κατασκευή του.



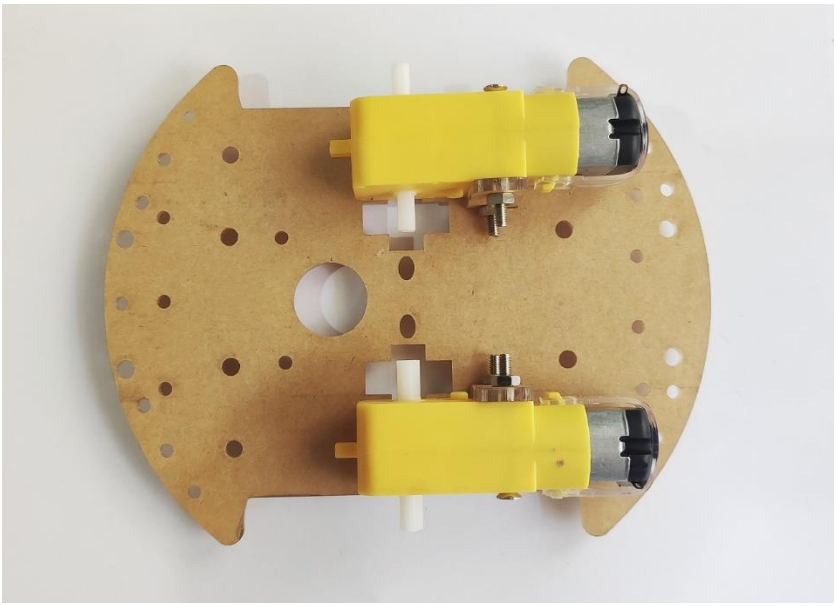
Εικόνα 5.1:Εξαρτήματα που θα βρείτε στο robot kit.

1. Βήμα πρώτο: Παίρνουμε τα δύο από τα τέσσερα στηρίγματα πάνω στα οποία θα βιδώσουμε τον πρώτο κινητήρα, τα τοποθετούμε στις κατάλληλες τρύπες του μη κυκλικού κομματιού και με δύο βίδες και δύο παξιμάδια βιδώνουμε στο κάτω μέρος του τον πρώτο κινητήρα όπως φαίνεται στην εικόνα 5.2.



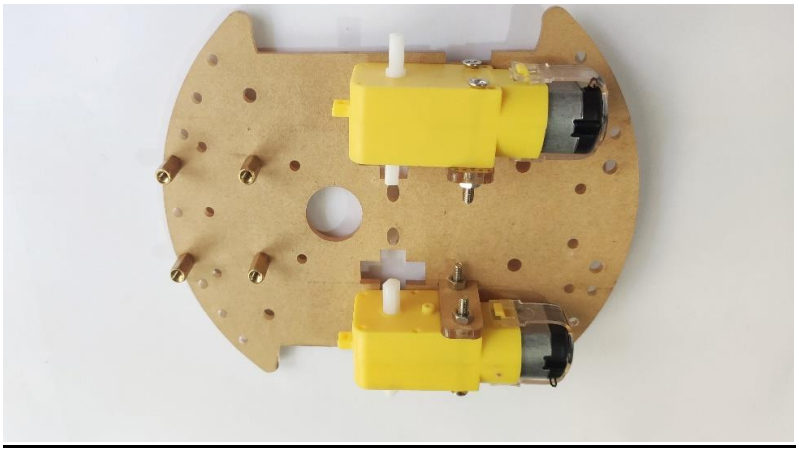
Εικόνα 5.2: Βίδωμα πρώτου dc κινητήρα

2. Βήμα δεύτερο: Χρησιμοποιούμε τα άλλα δύο στηρίγματα, και ακριβώς απέναντι και παράλληλα από τον πρώτο κινητήρα, επαναλαμβάνουμε την διαδικασία του πρώτου βήματος και για τον δεύτερο κινητήρα.



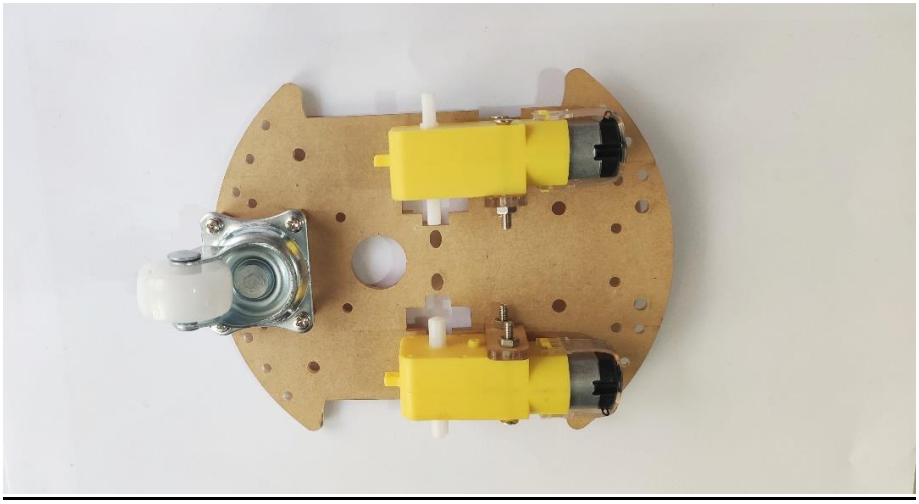
Εικόνα 5.3: Βίδωμα δεύτερου dc κινητήρα

3. Βήμα τρίτο: Παίρνουμε τους τέσσερις από τους οκτώ μικρούς πυλώνες, που θα αποτελέσουν την βάση για να τοποθετήσουμε την βοηθητική ρόδα, και τους βιδώνουμε στο κάτω και πίσω μέρος της μη κυκλικής επιφάνειας, στις κατάλληλες υποδοχές με την βοήθεια τεσσάρων βιδών.



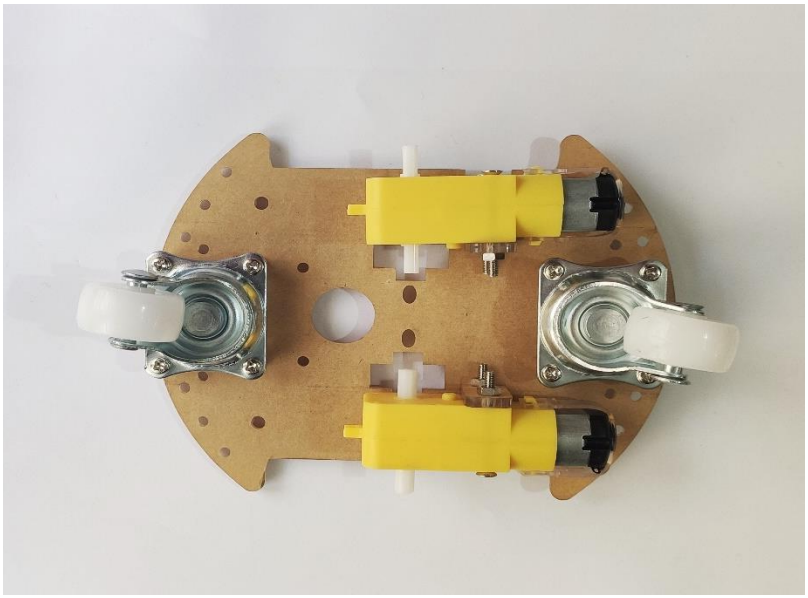
Εικόνα 5.4: Βίδωμα μικρών πυλώνων στο πίσω μέρος του αμαξιδίου.

4. Βήμα τέταρτο: Τώρα που έχουμε βιδώσει τα στηρίγματα ,παίρνουμε μία βοηθητική ρόδα και με την χρήση τεσσάρων μικρών βιδών, την βιδώνουμε πάνω σε αυτές όπως υποδείχνει η εικόνα 5.5.



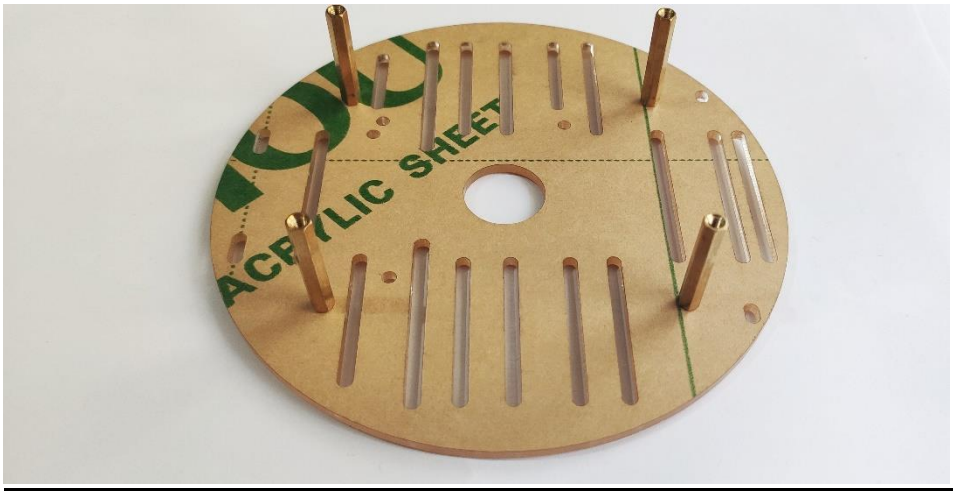
Εικόνα 5.5:Προσθήκη βοηθητικής ρόδας στο πίσω μέρος του αμαξιδίου.

5. Βήμα πέμπτο: Τα βήματα τρία και τέσσερα τα επαναλαμβάνουμε και για την δεύτερη μπροστινή βοηθητική ρόδα .



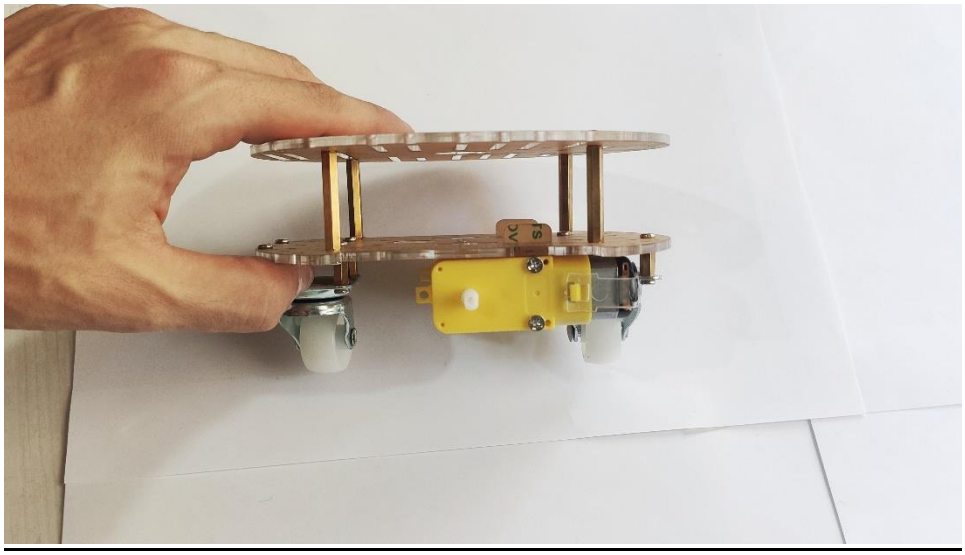
Εικόνα 5.6: Προσθήκη δεύτερης βοηθητικής ρόδας στο μπροστινό μέρος του αμαξιδίου.

6. Βήμα έκτο: Στο κυκλικό κομμάτι που θα βρούμε μέσα στο kit,και στο κάτω μέρος αυτού παίρνουμε και βιδώνουμε τους τέσσερις μεγάλους πυλώνες στις κατάλληλες υποδοχές.



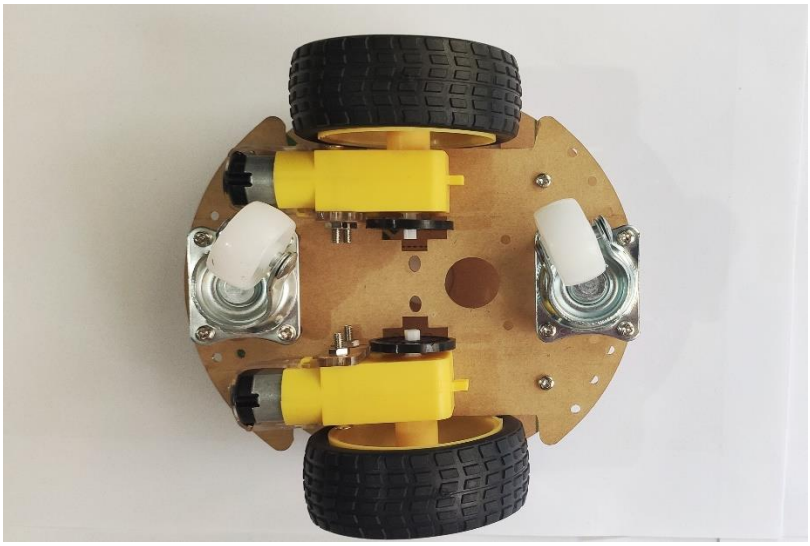
Εικόνα 5.7:Βίδωμα μεγάλων πυλώνων στο κάτω μέρους της κυκλικής πλακέτας.

7. Έβδομο βήμα: Παίρνουμε το προηγούμενο μη κυκλικό κομμάτι του αμαξιού και το κυκλικό και τα βιδώνουμε έτσι ώστε να δημιουργηθεί ένα ενιαίο κομμάτι.



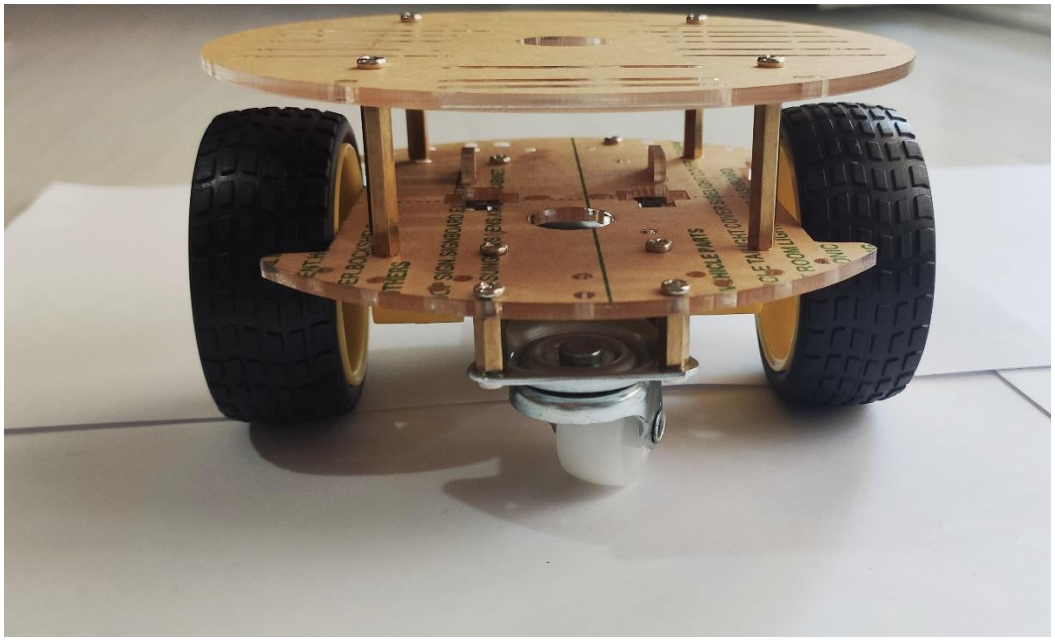
Εικόνα 5.8: Σύνδεση πάνω και κάτω μέρος του αμαξιδίου.

8. Βήμα οκτώ: Γυρίζουμε το αμάξι ανάποδα και πλέον είμαστε έτοιμοι να συνδέσουμε τις δύο ρόδες στο σύστημά μας. Οι ρόδες θα κουμπώσουν με εύκολο τρόπο στους λευκούς ‘’πασσάλους’’ που έχουν οι κινητήρες. Τοποθετούμε μια ρόδα στον έναν κινητήρα και την άλλη στον άλλον.



Εικόνα 5.9:Κούμπωμα και των δύο τροχών στα άκρα των κινητήρων.

Πλέον και με την ολοκλήρωση όλων των παραπάνω βημάτων έχουμε καταφέρει να δημιουργήσουμε τον κορμό όλου του συστήματος, πάνω στον οποίο θα τοποθετηθούν όλα τα επιμέρους εξαρτήματα, όπως το Arduino board οι ultrasonic αισθητήρες το breadboard κ.α.



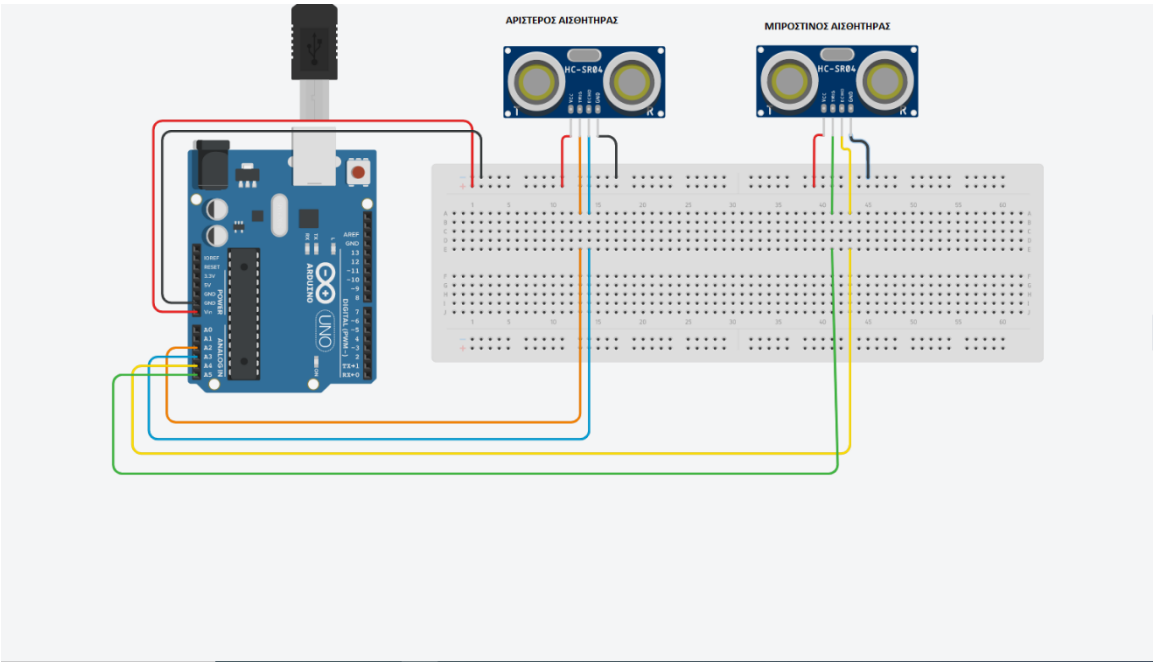
Εικόνα 5.10 : Τελική εικόνα του αμαξιδίου

5.2 Κύκλωμα συστήματος

Στο επάνω μέρος του αμαξιδίου, θα τοποθετήσουμε όλα τα επιμέρους εξαρτήματα για να ολοκληρώσουμε την κατασκευή του, δηλαδή το Arduino ,τους ultrasonic αισθητήρες τις μπαταρίες και τον motor controller.Η συνδεσμολογία όλων των παραπάνω ,αποτελεί ένα ηλεκτρολογικό κύκλωμα το οποίο θα το διασπάσουμε σε δύο επιμέρους κυκλώματα για την καλύτερη κατανόηση.

Για την καλύτερη απεικόνιση των κυκλωμάτων, ο προσχεδιασμός τους θα γίνει μέσω ενός διαδικτυακού προγράμματος τρισδιάστατης μοντελοποίησης, το Tinkercad. Ο ισότοπος αυτός είναι δωρεάν για όλους και χαρακτηρίζεται για την ευκολία και την απλότητά του.

Σαν πρώτο μέρος θα δούμε την συνδεσμολογία των δύο ultrasonic αισθητήρων. Θα χρησιμοποιήσουμε ένα στο μπροστινό μέρος του ρομπότ και ένα στο αριστερό μέρος. Το ηλεκτρολογικό κύκλωμα φαίνεται στην εικόνα 5.10.



Εικόνα 5.11: Ηλεκτρολογικό κύκλωμα δύο ultrasonic αισθητήρων και Arduino.

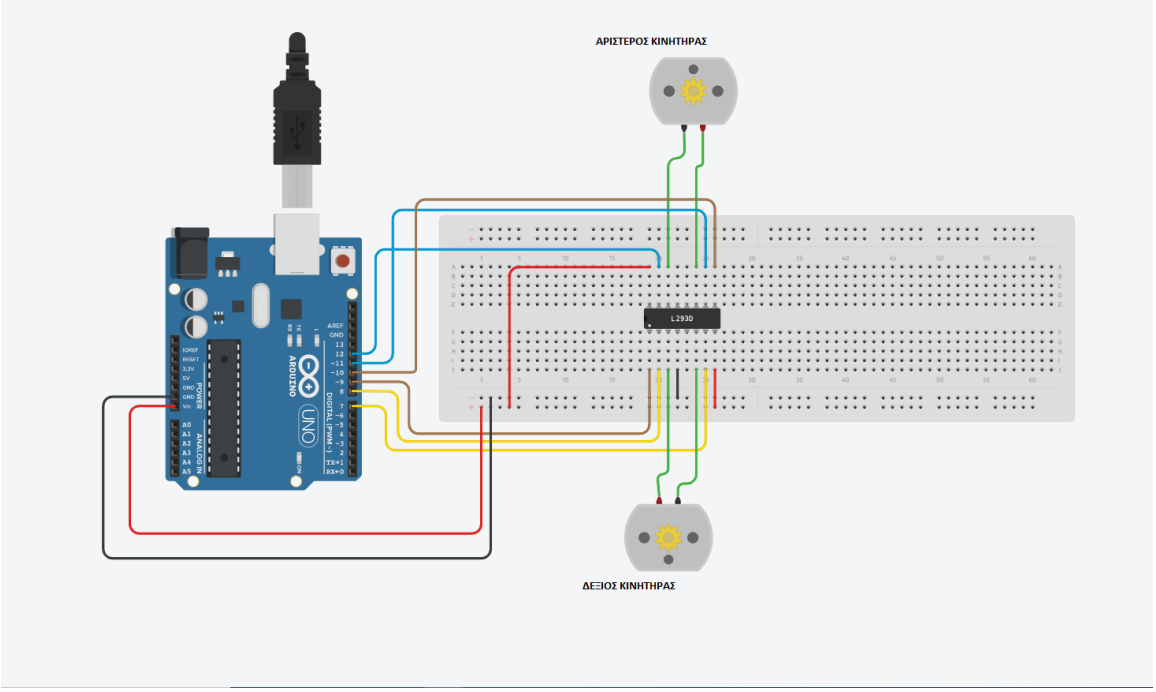
Με το κόκκινο χρώμα απεικονίζονται τα καλώδια τροφοδοσίας του συστήματος. Ένα βλέπουμε για την τροφοδοσία του breadboard από το Arduino,ενώ υπάρχουν και άλλα δύο τέτοια καλώδια τα οποία τροφοδοτούν με τάση 5V τους δύο αισθητήρες.

Με μαύρο καλώδιο απεικονίζονται οι γειώσεις του συστήματος. Με το πορτοκαλί και το πράσινο χρώμα είναι τα καλώδια τροφοδοσίας για τους ακροδέκτες Trig και τα έχουμε συνδέσει στις αναλογικές εισόδους του Arduino A2 και A5 αντίστοιχα.

Τέλος το μπλε και το κίτρινο καλώδιο είναι υπεύθυνα για την τροφοδοσία των ακροδεκτών ECHO και είναι συνδεδεμένα στους στους αναλογικούς ακροδέκτες A3 και A4 του Arduino.

Πρέπει να αναφέρουμε πως η τροφοδοσία του Arduino θα γίνει από μια εξωτερική πηγή ενέργειας και συγκεκριμένα από οκτώ μπαταρίες 1.2V συνδεδεμένες σε σειρά παράγωντας τάση 9.6V.

Στο δεύτερο ηλεκτρολογικό κύκλωμα θα δούμε την συνδεσμολογία του L293D motor controller με το Arduino, αλλά και με τους δύο κινητήρες.



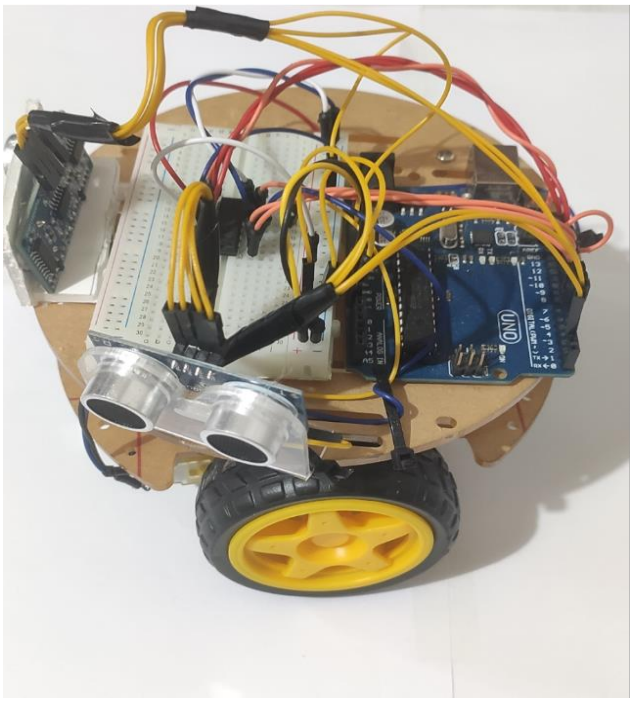
Εικόνα 5.12:Ηλεκτρολογικό κύκλωμα κινητήρων,Arduino και L239D.

Με το κόκκινο χρώμα απεικονίζονται πάλι τα καλώδια τροφοδοσίας, ένα για την τροφοδοσία του breadboard από το Arduino, ένα για την ενεργοποίηση του ακροδέκτη Vs και ένα για την ενεργοποίηση του Vss του motor controller.

Τα κίτρινα καλώδια είναι συνδεδεμένα στους ακροδέκτες 8 και 9 του Arduino και καταλήγουν στα ποδαράκια 2 και 7 του motor controller.Είναι δηλαδή υπεύθυνα για το ρεύμα εισόδου που θα δέχεται ο δεξιός κινητήρας και με βάση ποιο καλώδιο στέλνει ρεύμα, θα οδηγείτε και ανάλογα ο κινητήρας. Αντίστοιχα και για τον αριστερό κινητήρα υπάρχουν τα μπλε καλώδια που είναι συνδεδεμένα στους ακροδέκτες 11 και 12 του Arduino.

Τα πράσινα καλώδια είναι οι έξοδοι τάσης από τον motor controller στα άκρα των δύο κινητήρων. Τα καφέ καλώδια, που είναι συνδεδεμένα στους ακροδέκτη 9 και 10 του Arduino και καταλήγουν στους ακροδέκτες 1 και 9 του motor controller,καθορίζουν την ταχύτητα περιστροφής των κινητήρων στέλνοντας έναν PWM.

Η σύνθεση και τον δύο κυκλωμάτων σε ένα αποτελεί και το κύκλωμα που θα τοποθετήσουμε στο πάνω μέρος του αμαξιδίου.



Εικόνα 5.13: Κύκλωμα συστήματος Arduino ,Ultrasonic αισθητήρων και L239D στο πάνω μέρος του αμαξιδίου.

5.3 Επεξήγηση κώδικα

Αφού έχουμε ολοκληρώσει το κατασκευαστικό κομμάτι, επόμενος στόχος είναι να γράψουμε και να φορτώσουμε στην μνήμη του Arduino τον κώδικα. Σε αυτή την ενότητα θα δούμε και θα σχολιάσουμε ορισμένα κομμάτια του προγράμματός μας.

```
#include <NewPing.h>
#define MAX_DISTANCE 300
#define MAX_DISTANCE_FRONT 200
#define TRIGGER_PIN 2
#define ECHO_PIN 3
#define ECHO_PIN_FRONT 4
#define TRIGGER_PIN_FRONT  5
#define setpoint 19
#define setpoint_front 50

NewPing Left(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);
NewPing Front(TRIGGER_PIN_FRONT, ECHO_PIN_FRONT, MAX_DISTANCE_FRONT);
```

Εικόνα 5.14: Defines,includes και δήλωση αισθητήρων.

Στις πρώτες γραμμές του κώδικα μας δηλώνουμε τις βιβλιοθήκες , τα defines καθώς και τους δύο αισθητήρες που θα χρησιμοποιήσουμε. Μια βιβλιοθήκη δηλώνετε "#include + όνομα βιβλιοθήκης.h»,όπως φαίνεται στην πρώτη γραμμή του κώδικα. Στην δικιά μας εργασία θα χρησιμοποιήσουμε μόνο την βιβλιοθήκη NewPing που αναλύσαμε στο κεφάλαιο 3.5 για τον υπολογισμό των αποστάσεων μεταξύ αισθητήρων και τειχών.

Με τα defines δηλώνουμε κάποιες σταθερές τιμές που μένουν αναλλοίωτες καθ' όλη την διάρκεια του προγράμματος και μπορούμε να τις χρησιμοποιήσουμε σε όλο τον κώδικα. Με το MAX_DISTANCE έχουμε δηλώσει την μέγιστη απόσταση που θέλουμε να μετράει ο αριστερός αισθητήρας. Το TRIGGER_PIN είναι η δήλωση του ακροδέκτη Trig του αισθητήρα, στην αναλογική είσοδο 2 του Arduino, ενώ το ECHO_PIN είναι το αντίστοιχο Echo pin όπου έχει δηλωθεί στην αναλογική είσοδο 3 του Arduino.

Αντίστοιχα το MAX_DISTANCE_FRONT αποτελεί το μέγιστο εύρος ανίχνευσης του μπροστινού αισθητήρα ,καθώς το ECHO_PIN_FRONT και TRIGGER_PIN_FRONT είναι οι ακροδέκτες Echo και Trig του μπροστινού αισθητήρα οι οποίοι έχουν συνδεθεί στις αναλογικές εισόδους 4 και 5 αντίστοιχα.

Με τα setpoint και setpoint_front δηλώνουμε τις αποστάσεις στις οποίες θέλουμε το ρομπότ μας να ανταποκρίνεται. Με λίγα λόγια, όταν ο μπροστινός αισθητήρας μέτρηση μια απόσταση η οποία είναι μικρότερη από 50cm τότε αυτομάτως σημαίνει πως στην απόσταση αυτή υπάρχει κάποιο εμπόδιο το οποίο πρέπει και να αποφύγει. Το setpoint που είναι δηλωμένο 19 υποδηλώνει πως το αμαξίδιο θα πρέπει να βρίσκεται στα 19cm μακριά από τον αριστερό τοίχο. Σε περίπτωση που είναι μεγαλύτερη ή μικρότερη η

απόσταση, τότε ενεργοποιείται ο PID controller που για την λειτουργία του θα μιλήσουμε παρακάτω.

Τέλος τα NewPing Left() και NewPing Front() είναι οι constructors της βιβλιοθήκης που αναλύσαμε στο κεφάλαιο 3.5 που αποτελεί ουσιαστικά την δήλωση των δύο αισθητήρων.

```
float read_distance_left() {
  int iterations = 5;
  float duration;
  duration = Left.ping_median(iterations);
  distance = (duration / 2) * 0.0343;
  return distance;
}
```

```
int read_distance_front() {
  int iterations1 = 1;
  float duration1;
  duration1 = Front.ping_median(iterations1);
  distance1 = (duration1 / 2) * 0.0343;
  return distance1;
}
```

Εικόνα 5.15:Συναρτήσεις υπολογισμού απόστασης

Στο τέλος και έξω από την main θα δηλώσουμε όλες τις συναρτήσεις που θα χρησιμοποιήσουμε στο πρόγραμμά μας. Δύο από αυτές είναι η read_distance_left() και read_distance_front().Οι συναρτήσεις αυτές έχουν ως αποτέλεσμα να μετράνε και να επιστρέφουν την τιμή της απόστασης για το μπροστινό και τον αριστερό αισθητήρα.

Πιο συγκεκριμένα μέσα στην κάθε συνάρτηση είναι δηλωμένες δύο μεταβλητές, η iterations που είναι ακέραια μεταβλητή και η duration που είναι αριθμός απλής ακρίβειας με υποδιαστολή. Στην συνάρτηση αυτή έχουμε χρησιμοποίηση μια μέθοδο της βιβλιοθήκης την ping_median(iterations).Με την μέθοδο αυτή , ο αισθητήρας παράγει περισσότερους από έναν παλμό, με αποτέλεσμα να δέχεται και περισσότερους, αγνοεί

τους παλμούς οι οποίοι δεν είναι έγκυροι και επιστρέφει τον μέσο όρο όλο τον υπόλοιπων. Με αυτόν τον τρόπο έχουμε μεγαλύτερη ακρίβεια στις μετρήσεις μας και αποφεύγουμε τυχόν άκυρες τιμές. Στην δικιά μας περίπτωση για τον αριστερό αισθητήρα έχουμε δηλώσει 5 επαναλήψεις καθώς θέλουμε μεγάλη ακρίβεια ,ενώ για τον μπροστινό αρκούμαστε με μία.

Τέλος υπολογίζουμε την απόσταση για κάθε αισθητήρα με τον τρόπο που εξηγήσαμε στην ενότητα 3.4 και επιστέφουμε την τιμή στην main με την χρήση του return.

```
void right_turn() {
  digitalWrite(a_dir1_pin, HIGH);
  digitalWrite(a_dir2_pin, LOW);
  analogWrite(a_pwm_pin, 85);

  digitalWrite(b_dir1_pin, HIGH);
  digitalWrite(b_dir2_pin, LOW);
  analogWrite(b_pwm_pin, 0);

}
```

Εικόνα 5.16:Συνάρτηση για δεξιά στροφή.

Μία ακόμη συνάρτηση που έχουμε δημιουργήσει είναι η συνάρτηση για την δεξιά στροφή(right_turn())του οχήματος. Όμως πως καταλαβαίνει το όχημα το πότε να στρίψει δεξιά, πότε αριστερά και πότε να συνεχίσει ευθεία;

Αυτό είναι ένα ερώτημα το οποίο μπορεί εύκολα να απαντηθεί μέσο ενός πίνακα αληθείας για τους δύο αισθητήρες. Ας υποθέσουμε ότι με την τιμή 1 ο αισθητήρας έχει εντοπίσει ένα εμπόδιο και με την τιμή 0 δεν έχει εντοπίσει.

Αριστερός αισθητήρας	**Μπροστινός αισθητήρας**	**Απόφαση**
1	0	Το ρομπότ ακολουθεί τον τοίχο

1	1	Το ρομπότ έχει φτάσει σε μια γωνία και πρέπει να στύψει δεξιά
0	1	Το ρομπότ φεύγει από τον τοίχο και πρέπει να στύψει αριστερά.
0	0	Το ρομπότ είναι εκτός τροχιάς

Πίνακας 5.1: Πίνακας αληθείας για τους δύο αισθητήρες.

Η συνάρτηση που απεικονίζεται στην εικόνα 5.15, αναφέρεται στην περίπτωση που ο μπροστινός αισθητήρας αλλά και ο αριστερός έχουν βρει κάποιο εμπόδιο δηλαδή το ρομπότ βρίσκετε σε γωνία.

Πιο αναλυτικά μέσα στην συνάρτηση έχουμε χρησιμοποιήσει δύο ειδικές συναρτήσεις την digitalWrite() , και την analogWrite().

```
//left motor
const int a_pwm_pin = 9;
const int a_dir1_pin = 8;
const int a_dir2_pin = 7;

//right motor
const int b_pwm_pin = 10;
const int b_dir1_pin = 11;
const int b_dir2_pin = 12;
```

Εικόνα 5.17:Δήλωση ακροδεκτών κινητήρων.

Τα a_pwm_pin, a_dir1_pin , a_dir2_pin, είναι οι ακροδέκτες όπου έχουν δηλωθεί στην αρχή του κώδικα και αποτελούν τους ακροδέκτες του αριστερού κινητήρα, ενώ αντίθετα τα b_pwm_pin, b_dir1_pin , b_dir2 είναι οι ακροδέκτες του δεξιού κινητήρα.

Με την συνάρτηση digitalWrite() και με βάση τον πίνακα αληθείας που αναλύσαμε στο κεφάλαιο 4.4 -Πίνακας ,δίνουμε τις κατάλληλες τιμές στους ακροδέκτες των δύο

κινητήρων, ενώ με την συνάρτηση analogWrite() καθορίζουμε την ταχύτητα περιστροφής των δύο κινητήρων.

Στην δικιά μας περίπτωση βλέπουμε πως η ταχύτητα του δεξιού κινητήρα είναι μηδέν ενώ του αριστερού κινητήρα 85.Αυτό άμεσα δηλώνει πως το αμαξίδιό μας θα πάρει μια στροφή 90^{o} προς τα δεξιά.

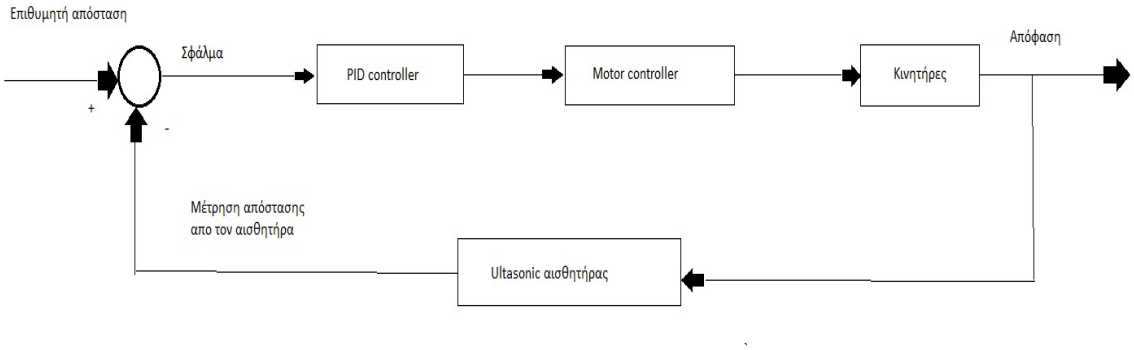
5.3.1 PID CONTROLLER

Στην εργασία μας για τον απόλυτο έλεγχο της απόστασης μεταξύ του αμαξιδίου και του τοίχου θα χρησιμοποιήσουμε δύο PID controllers, ένας υπεύθυνος για την παράλληλη οδήγηση του ρομπότ στην επιθυμητή απόσταση και ένας σε περίπτωση αριστερής στροφής για την ομαλή οδήγηση.

Η βασική ιδέα πίσω από έναν PID controller[12] είναι να διαβάζει συνέχεια έναν αισθητήρα και μετά να υπολογίζει την επιθυμητή έξοδο με την βοήθεια τριών ελεγκτών, του P(Proportional)-αναλογικού ελεγκτή, I(Integral)-ολοκληρωτικού ελεγκτή και του D(Derivative)-διαφορικού ελεγκτή.

Σε ένα τυπικό σύστημα ελέγχου υπάρχει πάντα μια μεταβλητή όπου πρέπει να ελεγχθεί και στην δικιά μας περίπτωση είναι η απόσταση που πρέπει να έχει το αμαξίδιο από τον τοίχο. Όπως προαναφέραμε, ένας αισθητήρας είναι υπεύθυνος πάντα για να μετράει την μεταβλητή αυτή και να παρέχει ανατροφοδότηση στο σύστημα ελέγχου. Ο αισθητήρας που θα κάνει αυτή την διεργασία στο σύστημά μας είναι ο αριστερός ultrasonic αισθητήρας. Στο πρόγραμμά μας έχουμε δηλώσει μια προκαθορισμένη απόσταση στην οποία θέλουμε να κινείται το ρομπότ. Κάθε χρονική στιγμή η διαφορά μεταξύ της προκαθορισμένης απόστασης και τις παρούσας απόστασης που μετράει ο αισθητήρας χρησιμοποιείται από τον αλγόριθμο για να καθορίσει την επιθυμητή έξοδο. Για παράδειγμα, εάν ο αισθητήρας μετράει εκείνη την στιγμή 15cm και η προκαθορισμένη απόσταση είναι 10cm από τον αριστερό τοίχο, τότε ο PID controller

είναι υπεύθυνος να οδηγήσει το ρομπότ στα 10cm.Με λίγα λόγια το σύστημά μας είναι ένα σύστημα κλειστού βρόγχου όπως απεικονίζεται στην εικόνα 5.17.



Εικόνα 5.18:Διάγραμμα κλειστού βρόγχου συστήματος.

Ο όρος P(Potential) παρουσιάζει την έξοδο του ελεγκτή ανάλογη του σφάλματος ελέγχου και έχει συνάρτηση μεταφοράς

$$C(s) = Kp.$$

Ο ελεγκτής αυτός είναι ένα αναλογικό κέρδος ο οποίος δρα διορθωτικά με σκοπό την μείωση του σφάλματος. Εάν αυξήσουμε το κέρδος υπάρχει καλύτερη αντιμετώπιση του σφάλματος και συνεπώς καλύτερη ανταπόκριση του συστήματος. Σε περίπτωση όμως που αυξήσουμε σε υπερβολικό βαθμό το κέρδος τότε το σύστημά μας θα αρχίσει να παρουσιάζει ταλαντώσεις και αστάθειες. Πρέπει να σημειωθεί, πως με την χρήση μόνο αυτού του ελεγκτή δεν μπορούμε να εξαλείψουμε πλήρως το σφάλμα.

Ο όρος I(Integral) αθροίζει τον όρο του σφάλματος με την πάροδου του χρόνου και έχει ως αποτέλεσμα, ότι ακόμα και ένας μικρός όρος σφάλματος θα προκαλέσει αργή αύξηση του όρου. Η συνάρτηση μεταφοράς του είναι

$$C(s) = Ki / s.$$

Η αύξηση του όρου θα συνεχίσει να αυξάνεται κατά την πάροδο του χρόνου μέχρι το σφάλμα της μόνιμης κατάστασης να γίνει μηδέν όμως πρέπει να σημειωθεί πως η αντίδρασή του είναι αργή επειδή η δράση του είναι μικρή καθώς χρειάζεται χρόνος. Η δράση του αυξάνεται όσο το σφάλμα είναι θετικό, ακόμα και αν το σφάλμα αρχίζει να πλησιάζει το μηδέν.

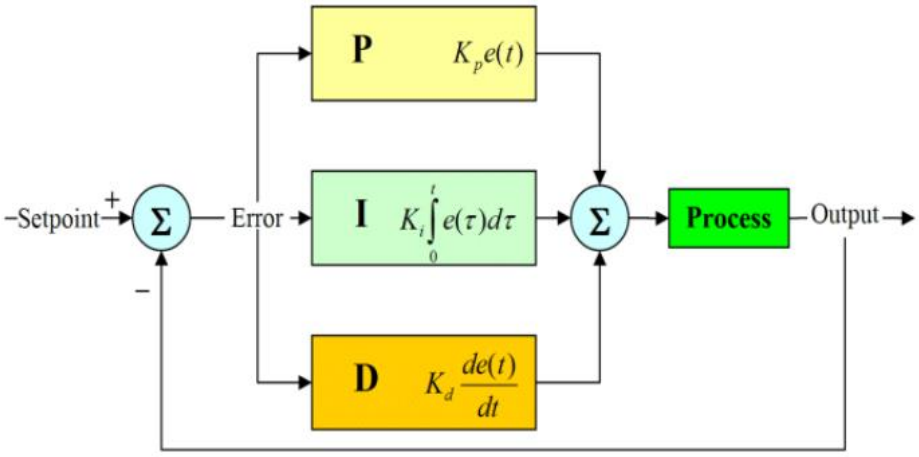
Ο όρος D(Derivative) είναι υπεύθυνος έτσι ώστε η τιμή της εξόδου του ελεγκτή να μειώνεται εάν η έξοδος του συστήματος αυξάνεται με ραγδαίο ρυθμό. Η συνάρτηση μεταφοράς του όρου είναι

$$C(s) = Kd * s.$$

Ουσιαστικά αυτό που θέλει να πετύχει ο ελεγκτής αυτός , είναι να προβλέψει και να εξαλείψει το σφάλμα πριν την εμφάνισή του, χωρίς όμως να σημαίνει ότι το σφάλμα μόνιμης κατάστασης θα μηδενιστεί.

Ο PID controller[13] αποτελεί έναν συνδυασμό όλων των παραπάνω ελεγκτών και έχει ως αποτέλεσμα να δίνει στο σύστημά μας υψηλότερες ταχύτητες λειτουργίας μηδενικό σφάλμα, καθώς και μείωση υπερύψωσης. Η συνάρτηση μεταφοράς ενός PID controller είναι

$$C(s) = Kp + Ki / s + Kd * s.$$



Εικόνα 5.19: PID block diagram

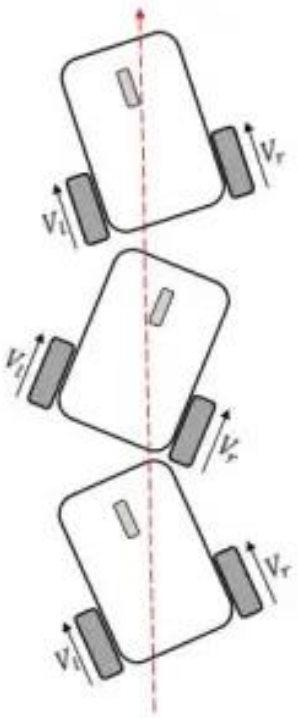
Για να την σωστή λειτουργία του συστήματός μας, είναι απαραίτητο να ρυθμίσουμε τα κέρδη Kp, Ki και Kd στις σωστές τιμές έτσι ώστε να ανταποκρίνονται στις απαιτήσεις μας. Μία σύνοψη για την λειτουργία των κερδών φαίνεται στον παρακάτω πίνακα.

Κέρδος	**Χρόνος Ανύψωσης**	**Υπερύψωση**	**Χρόνος αποκατάστασης**	**Σφάλμα**
Kp	Μείωση	Αύξηση	Μικρή αλλαγή	Μείωση
Ki	Μείωση	Αύξηση	Αύξηση	Εξάλειψη
Kd	Μικρή αλλαγή	Μείωση	Μείωση	Μικρή αλλαγή

5.3.2 ΣΥΝΤΟΝΙΣΜΟΣ PID

Υπάρχουν πολλοί και διάφοροι τρόποι για να καταφέρει κάποιος να συντονίσει έναν PID controller.Ένας από τους πιο αποτελεσματικούς αλλά ταυτόχρονα και πιο χρονοβόρος τρόπος είναι ο χειρωνακτικός συντονισμός των Kp ,Ki και Kd.Αυτός είναι και ο τρόπος που διαλέξαμε για τον συντονισμό των δικών μας PID.

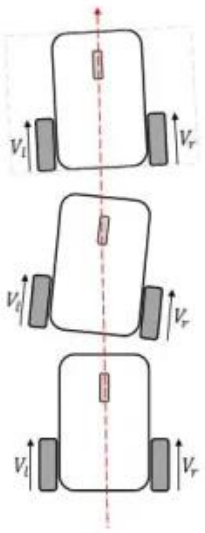
Ως πρώτο βήμα ορίζουμε όλες τις τιμές των Kp ,Ki και Kd ίσες με μηδέν. Στην συνέχεια αυξάνουμε το κέρδος Kp μέχρι να δούμε το σύστημά μας να παρουσιάζει ταλάντωση και μειώνουμε την τιμή του Kp περίπου στο μισό της τιμής της.



Εικόνα 5.20:Κίνηση ρομπότ μόνο με Kp κέρδος.

Στην συνέχεια αυξάνουμε σταδιακά το κέρδος Ki μέχρι να διορθωθεί οποιαδήποτε μετατόπιση σε επαρκή χρόνο για την διαδικασία. Πρέπει να δώσουμε μεγάλη προσοχή

να μην υπερβάλουμε με την τιμή που θα δώσουμε στο Ki διότι θα προκαλέσει αστάθεια στο σύστημα. Στο τελευταίο βήμα αυξάνουμε σταδιακά το Kd μέχρι να δούμε το σύστημα ότι έχει σταθεροποιηθεί.



Εικόνα 5.21: Κίνηση ρομπότ με PID controller.

5.3.3 ΕΠΕΞΗΓΗΣΗ ΚΩΔΙΚΑ PID

```
  if (distanceFront > 45 || distanceFront == 0)
{
    error = distanceLeft - setpoint;
    derivative = (error - lasterror);

    if (abs(error) < 10 && error != 0) {

      integral = error + integral;

    }
    else {

      integral = 0;

    }

    if (error == 0) {

      derivative = 0;

    }
    if (error > 30) {

      error = 30;
    }

    pid_value = (kp * error + kd * derivative + ki * integral);

    if (error < 12)
    {
      if (pid_value > 80 && pid_value > 0) {
        pid_value = 80;
      }
      if (pid_value < -80 && pid_value < 0) {
        pid_value = -80;
      }

      leftl = leftspeed - (pid_value);
      right = rightspeed + pid_value;
      forward(leftl, right);
    }
```

Εικόνα 5.22 : Κώδικας για PID

Στον κώδικα που φαίνεται στη εικόνα 5.21 βλέπουμε την δημιουργία του PID για την παράλληλη οδήγηση του αμαξιδίου με τον αριστερό τοίχο. Αρχικά όλα ξεκινάνε με τον έλεγχο του μπροστινού αισθητήρα. Εάν ο αισθητήρας δεν έχει βρει κάποιο εμπόδιο μπροστά του τότε ξεκινάμε τις αρχικοποιήσεις που είναι απαραίτητες για τον PID.Παρατηρώντας βλέπουμε πως μέσα στην if έχουμε βάλει και την περίπτωση η απόσταση του αισθητήρα να είναι μηδέν. Αυτό έγινε, διότι ο αισθητήρας επιστρέφει μηδέν σε περίπτωση μη έγκυρης τιμής. Σε αυτή την περίπτωση δεν θέλουμε να σταματήσει το όχημα και για αυτό το συμπεριλάβαμε στον έλεγχο.

Η μεταβλητή error είναι το λεγόμενο λάθος του συστήματος και ισούται με την στιγμιαία μέτρηση του αριστερού κινητήρα μείον το setpoint, που αποτελεί την προκαθορισμένη απόσταση που έχουμε ορίσει το σύστημα να οδηγείτε.

Η μεταβλητή derivative όπως είπαμε προσπαθεί να προβλέψει και να εξάλειψη το σφάλμα πριν την εμφάνισή του. Αυτό προγραμματιστικά σημαίνει ότι σε κάθε επανάληψη πρέπει από το νέο error που έχουμε υπολογίσει να αφαιρέσουμε το προηγούμενο(lasterror).Αυτό που δεν φαίνεται στην εικόνα είναι η αλλαγή τιμής της μεταβλητής lasterror.Στο τέλος της main αυτό που έχουμε να κάνουμε είναι στην τιμή του lasterror να δώσουμε την τιμή του error (δηλαδή lasterror = error).

Στην συνέχεια βλέπουμε ένα έλεγχο στον οποίο μέσα είτε θα αυξάνουμε την μεταβλητή integral είτε θα την μηδενίζουμε. Ουσιαστικά με αυτόν τον έλεγχο διασφαλίζουμε πως οι τιμές του integral δεν θα είναι πάρα πολύ μεγάλες μιας και μπορεί να προκαλέσουν μεγάλη αστάθεια στο σύστημα είτε ακόμα και να προκαλέσουν ζημιά στους κινητήρες.

Επίσης βλέπουμε κάποιος άλλους ελέγχους ,που ελέγχουν εάν το error είναι μηδέν. Σε αυτή την περίπτωση το ρομπότ μας έχει φτάσει στην προκαθορισμένη απόσταση και έτσι οι ελεγκτές δεν χρειάζεται να επηρεάζουν πλέον το σύστημα, για αυτό και βλέπουμε ότι μηδενίζουμε τόσο το derivative όσο και το integral.

Στην συνέχεια και πριν τον υπολογισμό της τιμής του PID κάνουμε ακόμα έναν έλεγχο για την τιμή του error που την οριοθετούμε max στην τιμή 30.Αμέσως μετά γίνεται ο υπολογισμός του PID.

Ακολουθεί ο έλεγχος για το εάν το error είναι μικρότερο του 12.Αυτό έμπρακτα σημαίνει πως το ρομπότ τείνει να οδηγείτε προς τον τοίχο και πως ο PID πρέπει να αναλάβει δράση. Μέσα στην if γίνεται ένας έλεγχος για την οριοθέτηση της τιμής του PID και μετά υπολογίζονται οι ταχύτητες των δύο τροχών. Ουσιαστικά την στιγμή που το όχημα τείνει να ακουμπήσει τον τοίχο, η τιμή του PID έχει αρνητικό πρόσημο για αυτό και βλέπουμε στον αριστερό κινητήρα να αφαιρείται η τιμή του PID ενώ στον δεξιό να προστίθεται.

Τέλος καλείτε η συνάρτηση forward() με δύο ορίσματα με σκοπό να στείλει τις κατάλληλες εντολές για να οδηγηθεί το σύστημά μας.

5.4 Συνολικό κόστος συστήματος

Arduino Uno board	20 Ευρώ
Robot kit	15 Ευρώ
2 x Ultrasonic sensors	2 x 1,80 = 3,6 Ευρώ
Mini breadboard	2,5 Ευρώ
L293D Motor controller	0,70 Λεπτά
8 x Μπαταρίες 1300 mAh 1.2V NiMHεπαναφορτιζόμενες	10 Ευρώ
8xAA Battery Holder Box with Switch and DC connector	1,60 Ευρώ
Πακέτο 70 καλωδίων όλων των ειδών	3 Ευρώ
ΣΥΝΟΛΟ	56,40 Ευρώ

Κεφάλαιο 6 – Δυσκολίες και τρόποι αντιμετώπισης

Κατά την διάρκεια της υλοποίησης του wall following robot αντιμετωπίσαμε διάφορα προβλήματα τόσο στο μηχανολογικό κομμάτι όσο και στο λογισμικό. Με υπομονή, έρευνα και μελέτη στο διαδίκτυο, αλλά και φυσικά υπό την καθοδήγηση του διδάσκοντα καταφέραμε να λύσουμε τα προβλήματα και να πετύχουμε τον στόχο μας.

Ένα από τα πρώτα προβλήματα που αντιμετώπισα, ήταν στα πρώτα βήματα της εργασίας και αφορούσε την τροφοδοσία του συστήματος. Σαν πηγή χρησιμοποιούσα μια αλκαλική μπαταρία 9 Volt και ήταν υπεύθυνη για να δίνει τάση τόσο στην πλακέτα

Arduino όσο και στους δύο αισθητήρες. Αυτό όμως που δεν είχα υπολογίσει ήταν η χωρητικότητα της μπαταρίας, η οποία δεν ξεπερνούσε τα 600mAH με αποτέλεσμα λόγο των πολλών δοκιμών σε μία μέρα να αδειάζει και να μην μπορεί να τροφοδοτήσει τους κινητήρες. Έπειτα από ψάξιμο στο διαδίκτυο ως καλύτερη λύση ήταν να χρησιμοποιήσω 8 μπαταρίες 1300mAH των 1.2V συνδεδεμένες σε σειρά, παράγοντας τάση 9.6V.Η διαφορά είτε ορατή και το κόστος των επαναφορτιζόμενων μπαταριών συμφέρει πιο πολύ σε projects που χρειάζονται δοκιμές, όπως το δικό μας.

Λόγο των φθηνών dc κινητήρων που διαλέξαμε δημιουργήθηκε ακόμα ένα πρόβλημα, όπου στην αρχή ήταν δύσκολο να ανιχνεύσω την ρίζα του. Όταν προσπαθούσα να οδηγήσω το ρομπότ σε μια ευθεία γραμμή, δίνοντας ίσες ταχύτητες και στους δύο κινητήρες, αυτό είχε ως αποτέλεσμα το ρομπότ μας να μην οδηγείτε ευθεία αλλά να έχει μια κλήση προς τα αριστερά. Έπειτα από συζήτηση με τον καθηγητή και αναζήτηση στο διαδίκτυο, φτάσαμε στο συμπέρασμα πως οι δύο κινητήρες αν και φαινομενικά έδειχνα ίδιοι, δεν ήταν. Η λύση του προβλήματος έγινε μέσα από το πρόγραμμα του συστήματος. Αυτό που πραγματοποιήσαμε ουσιαστικά ήταν να μειώσουμε την ταχύτητα του δεξιού κινητήρα στα 80 από τα 95, αφήνοντας την ταχύτητα του αριστερού στα 90 και έτσι πετύχαμε εξισορρόπηση του οχήματος.

Ο PID controller ίσως και να ήταν το μεγαλύτερο εμπόδιο που αντιμετώπισα στην εργασία. Ως πρώτη σκέψη, όλο το σύστημα θα δούλευε χωρίς την χρήση του PID,και θα χρησιμοποιούσε τις κατάλληλες συναρτήσεις για να οδηγείτε. Μετά από πολλές δοκιμές το τελικό αποτέλεσμα δεν ήταν το επιθυμητό μιας και ήταν εμφανές ότι το σύστημά μας ήταν ασταθές. Με τον όρο ασταθές εννοώ ότι κ’ αθόλη την διαρκεί της οδήγησής του σε ευθεία ακολουθούσε μια κίνηση ταλάντωσης, σε περίπτωση αριστερής στροφής η ταχύτητά του δεν μειωνόταν με αποτέλεσμα να βγαίνει εκτός τροχιάς, χωρίς να μπορεί να επιστρέψει στο επιθυμητό σημείο. Γενικά αυτός ο τρόπος δεν αποδείχθηκε αντάξιος των προσδοκιών μου. Επόμενο βήμα η επίλυση του

προβλήματος από τον δρόμο του PID controller.Έπειτα από βαθιά αναζήτηση και μελέτη για το πως λειτουργεί ένας PID τον ενσωματώσαμε στο πρόγραμμά μας κάνοντας ακόμα ένα μεγάλο πρόβλημα να βγει στην επιφάνεια. Ο συντονισμός του PID ήταν η πιο χρονοβόρα διαδικασία όλης της εργασίας. Αρχικά ακολούθησα διάφορες μεθόδους συντονισμού όπως αυτή του Ziegler – Nichols και του Cohen – Coon, χωρίς τα επιθυμητά αποτελέσματα. Έτσι ο χειρωνακτικός συντονισμός έμοιαζε αδιέξοδο. Μετά από εκατοντάδες δοκιμές διάφορων τιμών και παρατηρήσεων καταφέραμε να βρούμε τις κατάλληλες τιμές των κερδών και το αποτέλεσμα δικαίωσε και την τελευταία δοκιμή που είχαμε κάνει.

Συμπεράσματα και μελλοντικές επεκτάσεις.

Ο στόχος της διπλωματικής εργασίας για το wall following ρομπότ, έπειτα από πολλές δοκιμές, ώρες αναζήτησης στο διαδίκτυο, σφάλματα και συζητήσεις τελικά επετεύχθη.

Τα συμπεράσματα που προκύπτουν είναι πολλά! Κατανοήσαμε και στην πράξη την σημαντικότητα και την αξιοπιστία της πλακέτας Arduino Uno και είδαμε ότι με κάποιες βασικές γνώσεις στις γλώσσες προγραμματισμού C/C++ μπορείς εύκολα να τον προγραμματίσεις . Άλλωστε λόγο της αξιοπιστίας και του κόστους της ο μικροελεγκτής αυτός έχει αποτελέσει μια μεγάλη επανάσταση στον τομέα της ρομποτικής.

Είδαμε και έμπρακτα την λειτουργία των ultrasonic αισθητήρων και συμπεραίνουμε ότι η λειτουργία τους είναι παραπάνω από το ικανοποιητικό εάν διαλογιστή κανείς το κόστος αγοράς των συγκεκριμένων εξαρτημάτων. Όσον αφορά τον τρόπο συνδεσιμότητας και προγραμματισμού, εάν και στην αρχή η επεξήγηση φαινόταν λίγο περίπλοκη και μη κατανοητή είδαμε η χρήση των κατάλληλων βιβλιοθηκών μας κάνουν την ζωή πολύ πιο εύκολη από ότι νομίζαμε.

Ο συνδυασμός των εξαρτημάτων αυτόν και του τσιπ ln29D αποτέλεσε το ρομπότ. Χωρίς την χρήση του κατάλληλου λογισμικού είδαμε ότι τα αποτελέσματα δεν ήταν τα επιθυμητά. Με την εισαγωγή του PID καταφέραμε να εντοπίσουμε και να εξαλείψουμε τα τυχόν λάθη κάνοντας το ρομπότ μας να κινείτε όπως το είχαμε σχεδιάσει. Καταλαβαίνουμε λοιπόν μόνο με την χρήση αξιόπιστων εξαρτημάτων, δεν μπορεί να λειτουργήσει ένα σύστημα, χρειάζεται επίσης και ένα ορθό και αξιόπιστο software για να συντονίζει όλες τις επιμέρους διεργασίες.

Η δημιουργία του wall following ρομπότ αποτελεί ένα μικρό παράδειγμα για το τι πραγματικά μπορεί να πετύχει κάποιος στον τομέα της ρομποτικής. Φυσικά οι δυνατότητες του συγκεκριμένου συστήματος μπορούν να εξελιχθούν είτε με την προσθήκη νέων εξαρτημάτων και αισθητήρων , είτε με την προσθήκη η βελτίωση του λογισμικού. Στο ρομπότ για παράδειγμα σε κάποιες μελλοντικές επεκτάσεις, μπορούμε να προσθέσουμε ακόμα έναν αισθητήρα στο δεξί μέρος και να το αναβαθμίσουμε σε maze wall following robot, δηλαδή ένα σύστημα που θα μπορεί να βγαίνει από έναν λαβύρινθο βλέποντας και ακολουθώντας τοίχους. Επιπρόσθετα μπορούμε να εντάξουμε και ένα λογισμικό machine learning κάνοντας το ρομπότ μας να καταλαβαίνει περισσότερες λεπτομέρειες για το περιβάλλον με αποτέλεσμα να γίνεται πιο ‘’έξυπνο’’.

Κώδικας

```
#include <NewPing.h>

#define MAX_DISTANCE 300

#define MAX_DISTANCE_FRONT 200

#define TRIGGER_PIN 2

#define ECHO_PIN 3

#define ECHO_PIN_FRONT 4

#define TRIGGER_PIN_FRONT    5

#define setpoint 19

#define setpoint_front    50


NewPing Left(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

NewPing Front(TRIGGER_PIN_FRONT, ECHO_PIN_FRONT, MAX_DISTANCE_FRONT);



double distanceLeft, distance, error, integral, lasterror, derivative, distanceFront,
distance1;

double error_front, derivative_front, last_error_front, integral_front;


float kp = 0.6;

float ki = 0.003;

float kd = 2.5;
```

```
float Kp_left = 0.2;
float Ki_left = 0;
float Kd_left = 0.3;
int totalLeft, totalRight;

int pid_value;

int rightspeed = 80;
int leftspeed = 95;

//left motor
const int a_pwm_pin = 9;
const int a_dir1_pin = 8;
const int a_dir2_pin = 7;

//right motor
const int b_pwm_pin = 10;
const int b_dir1_pin = 11;
const int b_dir2_pin = 12;

void setup() {
   pinMode(a_pwm_pin, OUTPUT);
   pinMode(a_dir1_pin, OUTPUT);
   pinMode(a_dir2_pin, OUTPUT);
   pinMode(b_pwm_pin, OUTPUT);
```

```
    pinMode(b_dir1_pin, OUTPUT);
    pinMode(b_dir2_pin, OUTPUT);
    Serial.begin(9600);
}

void loop() {

    distanceLeft = read_distance_left();
    distanceFront = read_distance_front();

    if (distanceFront > 45 || distanceFront == 0)
{
        error = distanceLeft - setpoint;
        derivative = (error - lasterror);

        if (abs(error) < 10 && error != 0) {
            integral = error + integral;
        }
        else {
            integral = 0;
        }

        if (error == 0) {

            derivative = 0;

        }
```

```
if (error > 30) {
    error = 30;
}
pid_value = (kp * error + kd * derivative + ki * integral);
if (error < 12)
{
    if (pid_value > 80 && pid_value > 0) {
        pid_value = 80;
    }
    if (pid_value < -80 && pid_value < 0) {
        pid_value = -80;
    }
    totalLeft = leftspeed - (pid_value);
    totalRight = rightspeed + pid_value;
    forward(totalLeft, totalRight);
}
else if(error>12){
    int lefturn = Kp_left * error +     Kd_left *derivative + Ki_left * integral;
        if(lefturn >80 && lefturn >0){
            lefturn=80;
        }
        if(lefturn <-80 && lefturn <0){
            lefturn =-80;
        }
    totalLeft= 80-( lefturn);

    totalRight = 80    +(lefturn);
```

```
            forward(totalLeft,totalRight);
        }

}
    else if (distanceFront < 45 && distanceFront != 0) {
        right_turn();
        delay(380);
      }

lasterror = error;
}


//-------FUNCTIONS----------------
void right_turn() {
    digitalWrite(a_dir1_pin, HIGH);
    digitalWrite(a_dir2_pin, LOW);
    analogWrite(a_pwm_pin, 85);

    digitalWrite(b_dir1_pin, HIGH);
    digitalWrite(b_dir2_pin, LOW);
    analogWrite(b_pwm_pin, 0);
}


void forward(int speed1, int speed2) {
```

```
    // Check to make sure speed is 0-255
    if ( speed1 < 0 ) {
        speed1 = 0;
    }
    if ( speed1 > 255 ) {
        speed1 = 255;
    }
    if (speed2 < 0) {
        speed2 = 0;

    }
    if (speed2 > 255) {
        speed2 = 255;
    }
    digitalWrite(a_dir1_pin, HIGH);
    digitalWrite(a_dir2_pin, LOW);
    analogWrite(a_pwm_pin, speed1);

    digitalWrite(b_dir1_pin, HIGH);
    digitalWrite(b_dir2_pin, LOW);
    analogWrite(b_pwm_pin, speed2);
}


void Stop() {

    digitalWrite(a_dir1_pin, HIGH);
```

```
    digitalWrite(a_dir2_pin, LOW);
    analogWrite(a_pwm_pin, 0);

    digitalWrite(b_dir1_pin, HIGH);
    digitalWrite(b_dir2_pin, LOW);
    analogWrite(b_pwm_pin, 0);

}

float read_distance_left() {
    int iterations = 5;
    float duration;
    duration = Left.ping_median(iterations);
    distance = (duration / 2) * 0.0343;
    return distance;
}
int read_distance_front() {
    int iterations1 = 1;
    float duration1;
    duration1 = Front.ping_median(iterations1);
    distance1 = (duration1 / 2) * 0.0343;

    return distance1;
}
```

Βιβλιογραφικές αναφορές

[1] https://en.wikipedia.org/wiki/Arduino Οκτώβριος 2021

[2] https://www.arduino.cc/en/main/boards Οκτώβριος 2021

[3] https://en.wikipedia.org/wiki/Arduino_IDE Οκτώβριος 2021

[4] https://arduino-seminars.gr/%CE%B3%CE%BB%CF%8E%CF%83%CF%83%CE%B1-%CF%80%CF%81%CE%BF%CE%B3%CF%81%CE%B1%CE%BC%CE%BC%CE%B1%CF%84%CE%B9%CF%83%CE%BC%CE%BF%CF%8D-arduino/ Οκτώβριος 2021

[5] https://www.elprocus.com/atmega328-arduino-uno-board-working-and-its-applications/ Οκτώβριος 2021

[6] https://www.arduino.cc/en/Tutorial/Foundations/Memory Οκτώβριος 2021

[7] https://randomnerdtutorials.com/complete-guide-for-ultrasonic-sensor-hc-sr04/ Οκτώβριος 2021

[8] https://www.arduino.cc/reference/en/libraries/newping/ Οκτώβριος 2021

[9] https://www.elprocus.com/h-bridge-motor-control-circuit-using-l293d-ic/ Οκτώβριος 2021

[10] https://maker.pro/custom/projects/all-you-need-to-know-about-l293d Οκτώβριος 2021

[11] https://technobyte.org/arduino-dc-motor-single-multiple-motors-interface-code/ Οκτώβριος 2021

[12] https://www.power-and-beyond.com/pid-controller--definition-and-explanations-a-915227/ Οκτώβριος 2021

[13] http://gun.teipir.gr/DSAELAB/Ergastiriakes/pidtutorial.pdf Οκτώβριος 2021

[14] John-David, Josh Adams and Harald Molle, Arduino Robotics (Technology in Action), 2011

[15] Michael Margolis Cookbook, 2012.

[16] Massimo Banzi and Michael Shiloh, Getting started with Arduino: The Opensource Electronics Prototyping Platform ,2014

Πηγές εικόνων

Εικόνα 3.1: https://lastminuteengineers.com/arduino-sr04-ultrasonic-sensor-tutorial/

Εικόνα 3.2: https://www.electronicwings.com/sensors-modules/ultrasonic-odulehcsr04

Εικόνα 3.3: https://lastminuteengineers.com/arduino-sr04-ultrasonic-sensor-tutorial/

Εικόνα 4.1: https://www.robotics.org.za/L293D-DIP

Εικόνα 4.2: http://bidyut-roboguru.blogspot.com/2014/07/l293d-motor-driver.html

Εικόνα 4.3,4.4,4.5: https://electronics.stackexchange.com/questions/207319/multiple-motor-h-bridge

Εικόνα 5.19: https://en.wikipedia.org/wiki/Control_system

Εικόνα 5.20, 5.21: https://iamzxlee.wordpress.com/2014/06/21/wall-following-robot/