ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ
ΕΛΕΓΧΟΣ ΠΟΡΕΙΑΣ (CRUISE CONTROL) ΜΙΚΡΟΥ ΚΙΝΗΤΟΥ ΡΟΜΠΟΤΙΚΟΥ ΣΥΣΤΗΜΑΤΟΣ

Φαρσάκογλου Θωμάς

Τμήμα Μηχανικών Αυτοματισμού Σχολή Τεχνολογικών Εφαρμογών

Εισηγητής καθηγητής: Χρήστος Υφούλης

Αλεξάνδρειο Τεχνολογικό Εκπαιδευτικό Ίδρυμα

9 Δεκεμβρίου 2016

Περιεχόμενα

1 Περίληψη 4

2 Summary 5

3 Πρόλογος 6

4 Εισαγωγή 7

4.1 Arduino ..... Error! Bookmark not defined.

4.2 Αποκωδικοποιητές ..... Error! Bookmark not defined.

5 Κατασκευή Ρομποτικού Οχήματος 10

5.1 Συναρμολόγηση 10

5.1.1 Βασική δομή 10

5.1.2 Προσθήκη ηλεκτρονικών εξαρτημάτων 14

5.2 Αρχές Λειτουργίας Εξαρτημάτων 17

5.2.1 Parallax PING))) 17

5.2.2 DC geared motors 17

5.2.3 Μπαταρίες τροφοδοσίας 18

5.2.4 Σασί για κινητήρες και ρόδες 18

5.2.5 Adafruit motor shield v2 18

5.2.6 Arduino Mega2560 r3 20

5.2.7 Αποκωδικοποιητές 20

5.2.8 Διάφορα εξαρτήματα Arduino 21

6 Πειράματα 22

6.1 Θεωρητική Προσέγγιση 22

6.1.1 Κατασκευή της συνάρτησης μεταφοράς των κινητήρων 22

6.1.2 Ρύθμιση του ελεγκτή 25

6.2 Πειραματικά Δεδομένα 26

6.2.1 Απόκριση PID σε βηματική μεταβολή 26

6.2.2 Ευθυγράμμιση οχήματος 29

6.2.3 Μετάβαση σε κεκλιμένο επίπεδο 30

6.3 Κοστολόγηση υλικών 31

7 Προγραμματιστικό μέρος 32

7.1 Διαγράμματα ροής 32

7.1.1 Αισθητήρας απόστασης PING))) 32

7.1.2 Έλεγχος ταχύτητας 33

7.1.3 Συνάρτηση εξομάλυνσης 34

7.2 Κώδικας Arduino 35

8 Επίλογος 40

8.1 Προβλήματα 40

8.1.1 Τοποθέτηση αποκωδικοποιητών 40

8.1.2 DC κινητήρες 41

8.1.3 Ελλατωματικό εξάρτημα 41

8.1.4 Αισθητήριο απόστασης 41
8.1.5 Κατανάλωση ενέργειας 41
8.2 Πιθανές επεκτάσεις 42
8.2.1 Έλεγχος θέσης 42
8.2.2 Line/wall following 42
8.2.3 Οδομετρία 42
9 Βιβλιογραφία 43

1 Περίληψη

Η παρούσα εργασία έχει ως αντικείμενο τον έλεγχο πορείας (cruise control) μικρού κινητού ρομποτικού οχήματος. Για τις ανάγκες της πτυχιακής σχεδιάστηκε και κατασκευάστηκε ένα μικρό κινητό ρομποτικό όχημα με δυνατότητες αποτελεσματικού ελέγχου της πορείας του, το οποίο είναι σε θέση να διατηρεί την ταχύτητα του σε επιθυμητά επίπεδα σε ευθεία πορεία αλλά και σε κεκλιμένο επίπεδο χωρίς να παρεκκλίνει από την διαδρομή του. Παράλληλα, έχει την δυνατότητα ανίχνευσης εμποδίων σε κοντινή απόσταση και ανάλογης προσαρμογής της πορείας του, δηλαδή είναι προγραμματισμένο να ακολουθεί κινητά εμπόδια και να ακινητοποιείται σε σταθερά εμπόδια.

Ο έλεγχος πορείας πραγματοποιείται με την βοήθεια ελεγκτών PID που υλοποιούνται προγραμματιστικά σε πλακέτα Arduino και σχεδιάστηκαν συστηματικά με την εξαγωγή του μαθηματικού μοντέλου του συστήματος. Η δυνατότητα ανίχνευσης εμποδίων παρέχεται από έναν υπερηχητικό αισθητήρα απόστασης ο οποίος τοποθετήθηκε στο μπροστινό μέρος του οχήματος και σύμφωνα με τις ενδείξεις του γίνεται επέμβαση στα setpoint των ελεγκτών ώστε να δράσουν ανάλογα. Τέλος ο τρόπος κατασκευής της πτυχιακής αφήνει ελεύθερο το περιθώριο μελλοντικών επεκτάσεων όπως έλεγχο θέσης, οδομετρία καθώς και ακολούθηση τοίχου η γραμμής (wall or line following).

2 Summary

The subject of this project is the cruise control of a small robotic vehicle. In order to meet the needs of the project, a small robotic vehicle has been constructed and programmed. It is able to keep its speed steady either at a straight or at an incline surface without straying from its route. Moreover, it is able to detect obstacles in the proximity ahead and change its course accordingly, namely it has been programmed to follow moving obstacles and immobilizing upon meeting a fixed barrier.

Cruise control has been achieved with PID controllers which had been implemented using an Arduino board and designed by extracting the system's mathematical model. The ability to detect objects is provided by a supersonic sensor which is mounted at the front part of the vehicle and, proportionally to its readings, the controllers setpoints change accordingly. Finally, the project is constructed in a way to offer freedom of future expansions such as position control, odometry and wall or line following.

3 Πρόλογος

Η εργασία διήρκησε από το Μάρτιο μέχρι τον Δεκέμβριο του έτος 2016 και έγινε παράλληλα με την πρακτική άσκηση στο ερευνητικό κέντρο ΕΚΕΤΑ. Στο σημείο αυτό θα ήθελα να ευχαριστήσω τους υπαλλήλους του ΕΚΕΤΑ για την βοήθεια και τις προτάσεις τους καθώς και για την ευκαιρία που μου δόθηκε να χρησιμοποιήσω τον επαγγελματικό εξοπλισμό που διέθεταν διευκολύνοντας έτσι την διαδικασία κατασκευής. Τέλος, η υλοποίηση της πτυχιακής εργασίας θα ήταν αδύνατη χωρίς την επίβλεψη και καθοδήγηση του επίκουρου καθηγητή Δρ. Χρήστου Υφούλη.

4 Εισαγωγή

Τα τελευταία χρόνια, καθώς ο τομέας του αυτοματισμού αναπτύσσεται ραγδαία, αναπτύσσονται και πολλά συστήματα για τα σύγχρονα οχήματα. Από τα πιο διαδεδομένα είναι ο έλεγχος πορείας (cruise control) και ο προσαρμοστικός έλεγχος πορείας (adaptive cruise control) με τα οποία και ασχολείται η συγκεκριμένη πτυχιακή. Σκοπός του ελέγχου πορείας είναι να διατηρεί το όχημα σταθερή την ταχύτητα του σε κάποια επιθυμητή τιμή ανεξαρτήτως των διάφορων μεταβολών του εδάφους η των εξωτερικών συνθηκών ενώ ταυτοχρόνως το δεν θα πρέπει να εκτρέπεται από την πορεία του. Ο προσαρμοστικός έλεγχος πορείας προσθέτει επιπλέον την δυνατότητα ανίχνευσης προπορευόμενων οχημάτων και εμποδίων και επεμβαίνει στην ταχύτητα του οχήματος προκειμένου να αποφευχθούν πιθανές προσκρούσεις.

Στην συγκεκριμένη πτυχιακή επιτυγχάνουμε τον έλεγχο πορείας με την κατασκευή ενός μικρού ρομποτικού κινητού οχήματος. Η κατασκευή του ελεγκτή πραγματοποιήθηκε με χρήση της πλακέτας Arduino mega2560 r3 που είναι μια ιδιαίτερα δημοφιλής πλακέτα λόγο του χαμηλού της κόστους και των ποικίλων δυνατοτήτων που μας παρέχει. Η γλώσσα προγραμματισμού που χρησιμοποιήθηκε είναι η Arduino, η οποία μοιάζει πολύ με την γλώσσα C++, σε περιβάλλον Arduino IDE. Για την καταμέτρηση της τρέχουσας ταχύτητας του οχήματος χρησιμοποιήθηκαν οπτικοί αποκωδικοποιητές οι οποίοι αποτελούν έναν από τους πιο προσιτούς και αξιόπιστους τρόπους για να πραγματοποιηθεί έλεγχος ταχύτητας.

Ο κώδικας που δημιουργήθηκε είναι σε θέση να γνωρίζει την ταχύτητας του οχήματος ανά ένα δέκατο του δευτερολέπτου και να παράγει ανάλογα δύο παλμοσειρές PWM στους δύο κινητήρες του οχήματος ώστε να το όχημα να διατηρεί σταθερή την ταχύτητα του ανεξαρτήτως της κλίσης του εδάφους η άλλων εξωτερικών μεταβολών. Επιπλέον, με την δυνατότητα ανίχνευσης εμποδίων προς την κατεύθυνση της κίνησης μπορεί να μειώνει την ταχύτητά του έτσι ώστε να διατηρεί μια συγκεκριμένη απόσταση από προπορευόμενα οχήματα ή να ακινητοποιείται σταδιακά εάν το εμπόδιο είναι ακίνητο.

Στα πλαίσια της εργασίας πραγματοποιήθηκαν διάφορα πειράματα ώστε να δοκιμαστεί η αποδοτικότητα του ελεγκτή. Τα πειράματα έγιναν σε κλειστό εσωτερικό χώρο ώστε να αποφευχθούν εξωτερικές μεταβολές και τα δεδομένα συγκεντρώθηκαν με χρήση επιμήκους καλωδίου USB A/B 4,5 μέτρων και στην συνέχεια αναπαραστάθηκαν σε περιβάλλον Matlab με την μορφή διαγραμμάτων.

Η δομή της εργασίας έχει ως εξής:

Κεφάλαιο 5: Με φωτογραφίες και σύντομες περιγραφές φαίνεται η διαδικασία κατασκευής του ρομποτικού οχήματος επεξηγώντας παράλληλα τις επιλογές που έγιναν κατά την διάρκεια υλοποίησης. Επιπλέον δίνεται μια περιγραφή των επιμέρους εξαρτημάτων του τονίζοντας τις δυνατότητες και τους περιορισμούς που παρέχονται από το κάθε ένα.

Κεφάλαιο 6: Αρχικά επεξηγείται η θεωρητική προετοιμασία που έγινε προτού προχωρήσουμε στην υλοποίηση της πτυχιακής, η οποία περιλαμβάνει την εξαγωγή της συνάρτησης μεταφοράς των κινητήρων και στη συνέχεια τον υπολογισμό των παραμέτρων του ελεγκτή. Έπειτα, με την βοήθεια τριών πειραμάτων γίνεται εμφανής η λειτουργικότητα του ελεγκτή καθώς και οι τρόποι με τους οποίους αυτός

αντιμετωπίζει τα διάφορα προβλήματα που καλείται να αντιμετωπίσει. Τέλος γίνεται η κοστολόγηση των υλικών που χρησιμοποιήθηκαν.

Κεφάλαιο 7: Σε πρώτο στάδιο παρουσιάζεται ο κώδικας με χρήση διαγραμμάτων ροής ώστε να είναι εύκολα κατανοητός χωρίς να απαιτείται γνώση της αντίστοιχης γλώσσας προγραμματισμού. Κλείνοντας, υπάρχει διαθέσιμος ο πηγαίος κώδικας με αναλυτικά σχόλια προκειμένου να είναι σε θέση κανείς να τον μελετήσει με περισσότερη λεπτομέρεια.

5 Κατασκευή Ρομποτικού Οχήματος

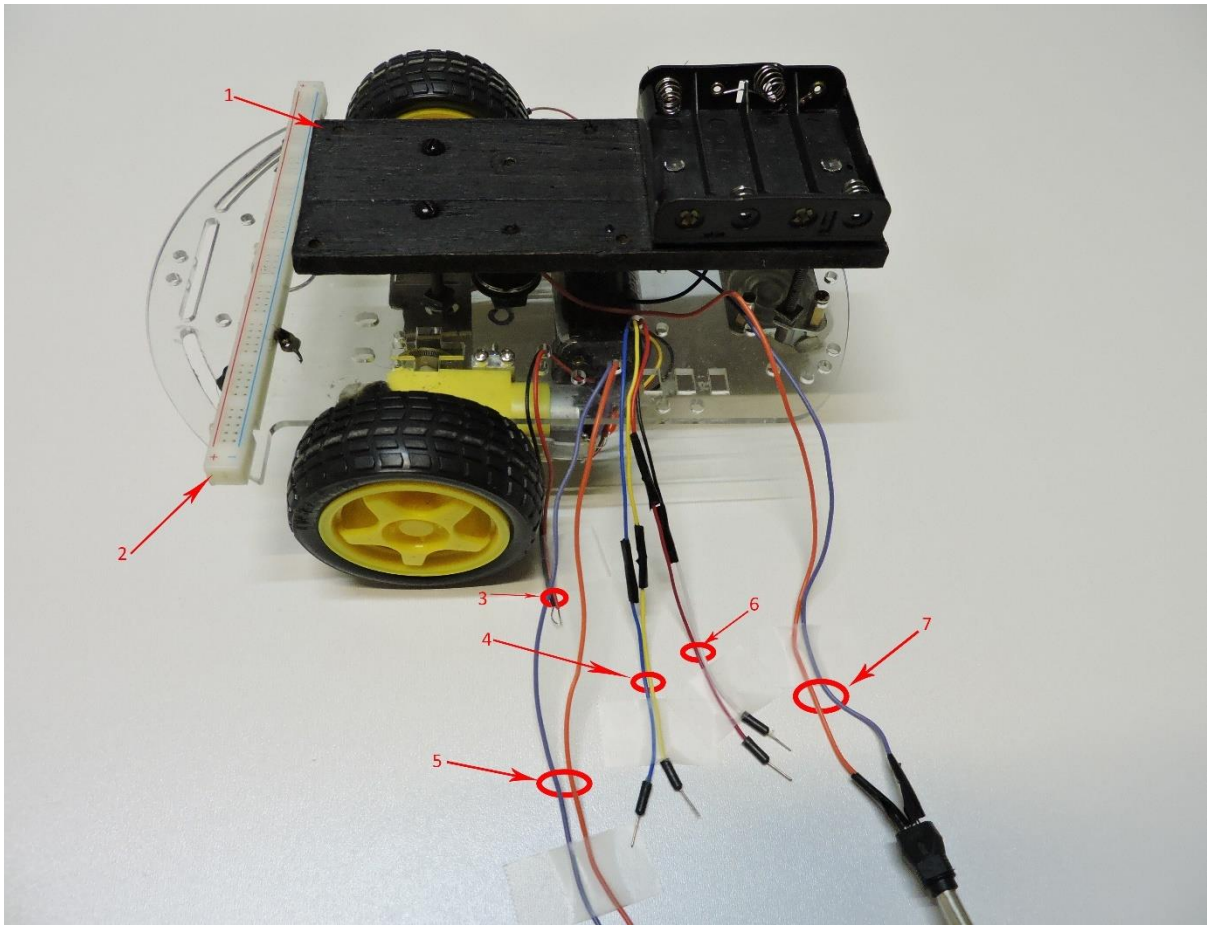
Στο κεφάλαιο αυτό αναλύεται η διαδικασία που ακολουθήθηκε για την κατασκευή του οχήματος. Παράλληλα δίνονται διάφορες πληροφορίες που αφορούν κυρίως την περιγραφή των κομματιών καθώς και επεξηγήσεις για τις διάφορες επιλογές που έγιναν κατά την διάρκεια σχεδιασμού.

5.1 Συναρμολόγηση

Περιγραφή της διαδικασίας σύνδεσης όλων των εξαρτημάτων βήμα προς βήμα με φωτογραφικό υλικό

5.1.1 Βασική δομή

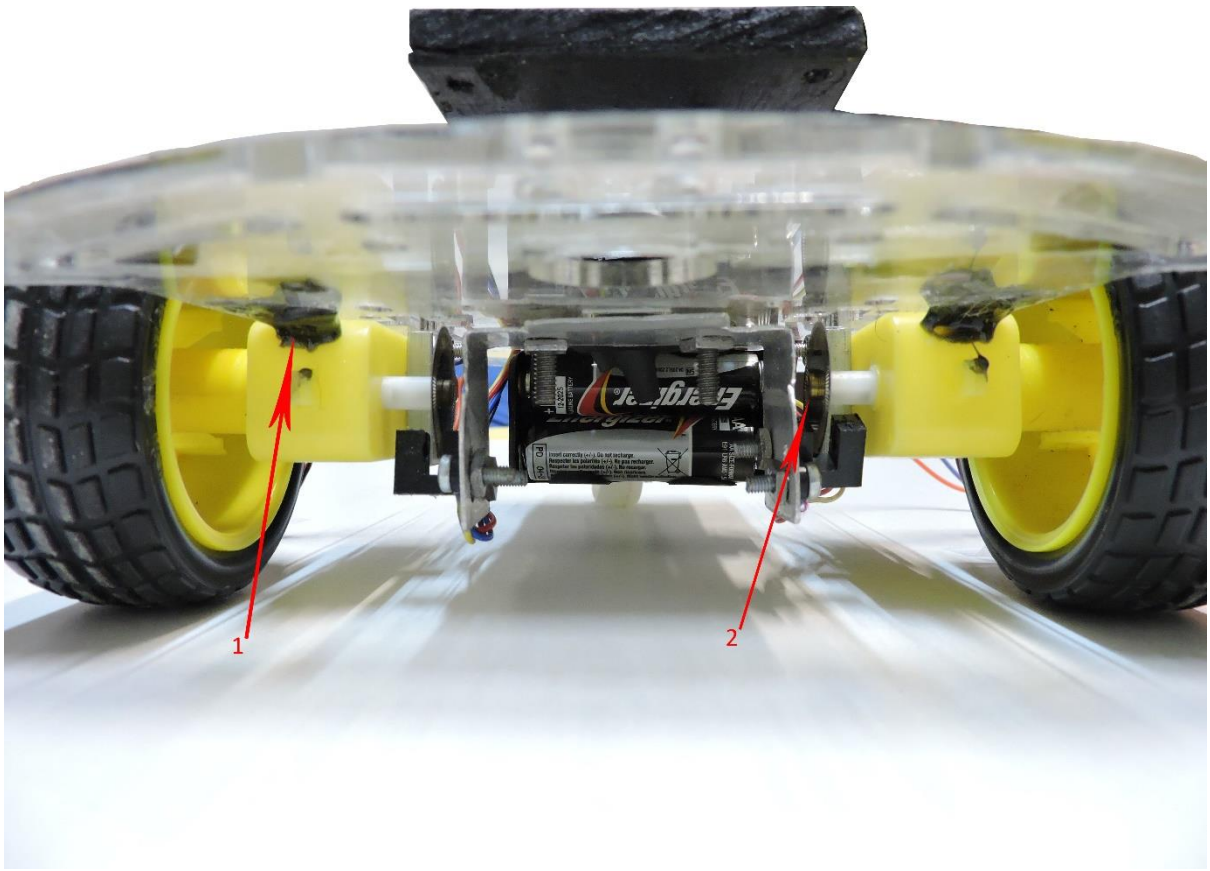
Στις παρακάτω εικόνες φαίνεται το αμαξίδιο χωρίς την προσθήκη των ηλεκτρονικών εξαρτημάτων του.



Εικόνα 5-1

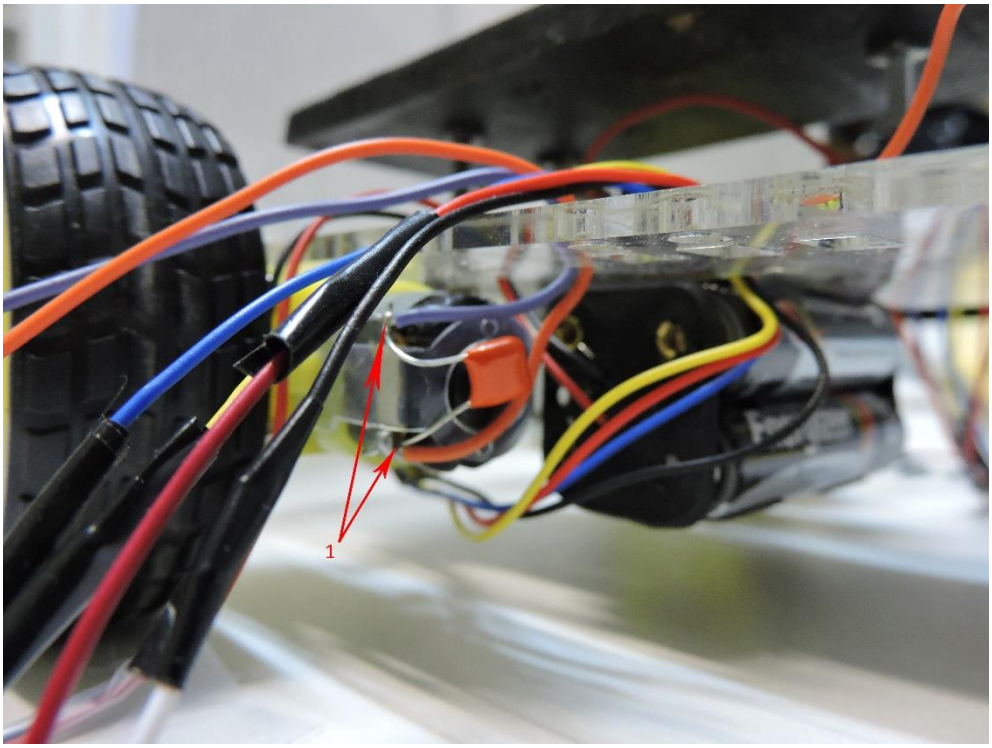
1. Η ξύλινη βάση κατασκευάστηκε διότι το σασί δεν διέθετε τις κατάλληλες οπές προκειμένου να τοποθετηθεί αφήνοντας αρκετό ελεύθερο χώρο για τα υπόλοιπα εξαρτήματα. Έχει αφεθεί απόσταση 3cm από το σασί ώστε να υπάρχει η δυνατότητα διέλευσης καλωδίων στον ενδιάμεσο χώρο.
2. Ένα τμήμα από ράστερ τοποθετήθηκε στο μπροστά μέρος καθώς το Arduino διαθέτει μόνο ένα pin για τροφοδοσία 5V και δύο για GND.

3. Τροφοδοσία 5.8V για πλακέτα motor shield, ώστε οι κινητήρες να είναι γαλβανικά απομονωμένοι από τον ελεγκτή.
4. Κανάλια αποκωδικοποιητών, για τις ανάγκες της πτυχιακής απαιτείται μόνο το ένα από κάθε κωδικοποιητή.
5. Τα καλώδια του DC κινητήρα
6. Καλώδια τροφοδοσίας των αποκωδικοποιητών
7. Τροφοδοσία **DC Barrel Jack** για την πλακέτα Arduino 5.8V.



Εικόνα 5-2

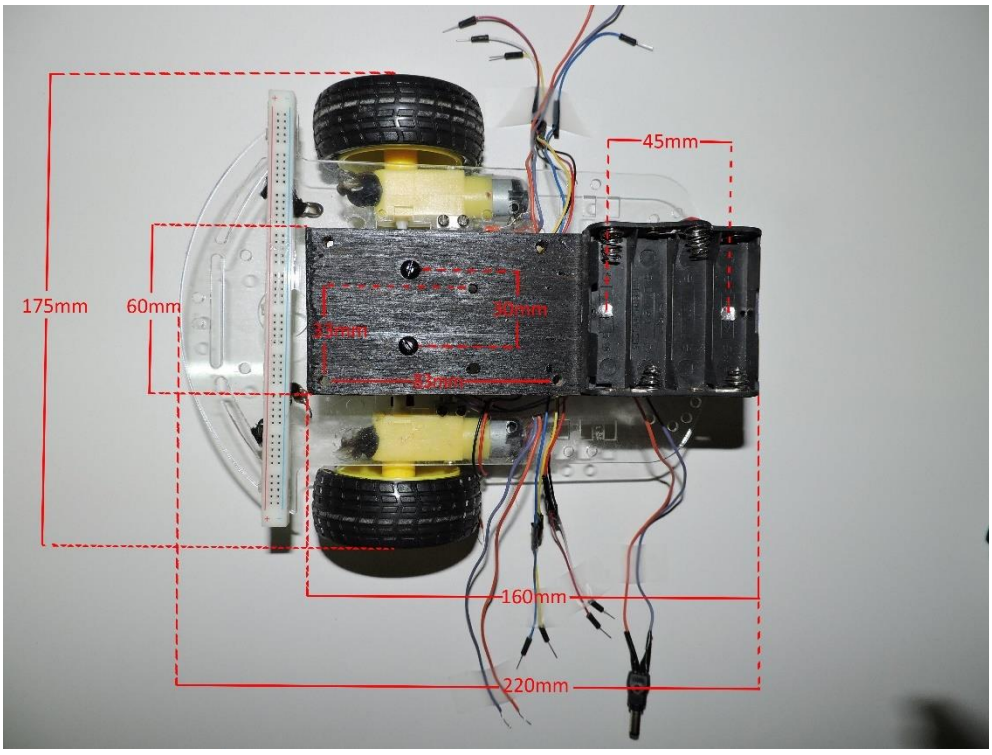
1. Προκειμένου οι κινητήρες να είναι σταθεροί και να αποφύγουμε έτσι απρόσμενες στροφές αλλά και για να μην δημιουργείται θόρυβος στους αποκωδικοποιητές προστέθηκε λίγη σιλικόνη ώστε να πακτωθούν επάνω στο σασί.
2. Οι τροχοί των αποκωδικοποιητών τοποθετήθηκαν επάνω στους άξονες των τροχών ανοίγοντας πάσο με μια βίδα. Προτιμήθηκε αυτή η λύση λόγω αδυναμίας εύρεσης εναλλακτικής, καθώς δεν υπήρχε τρόπος ώστε να τοποθετηθούν με μεγάλη ακρίβεια. Ως αποτέλεσμα ο τροχός είναι ελαφρώς έκκεντρος ως προς τους άξονες της ρόδας.



Εικόνα 5-3

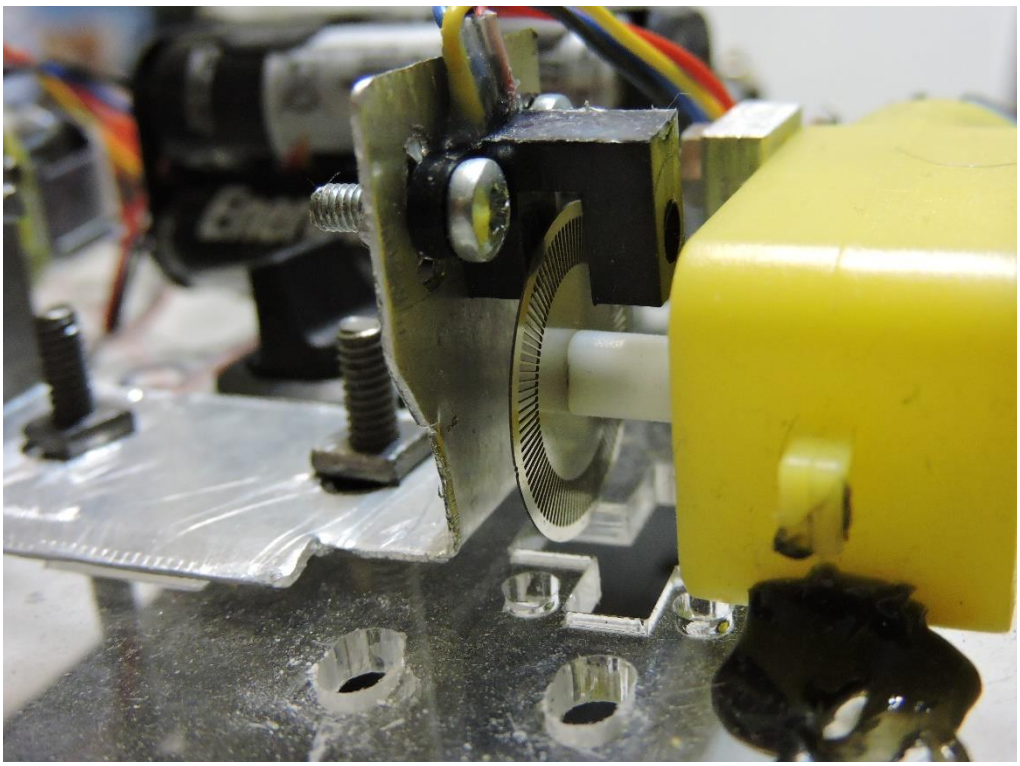
1. Θέλοντας να μειώσουμε όσο το δυνατόν περισσότερο τον θόρυβο στους κινητήρες και σύμφωνα με τις οδηγίες του *εγχειρίδιου λειτουργίας* του motor shield τοποθετήθηκαν παράλληλα στην τροφοδοσία των κινητήρων πυκνωτές χωρητικότητας 700nF.

Στην παρακάτω κάτοψη φαίνονται οι διαστάσεις του οχήματος και των περιφερειακών εξαρτημάτων.

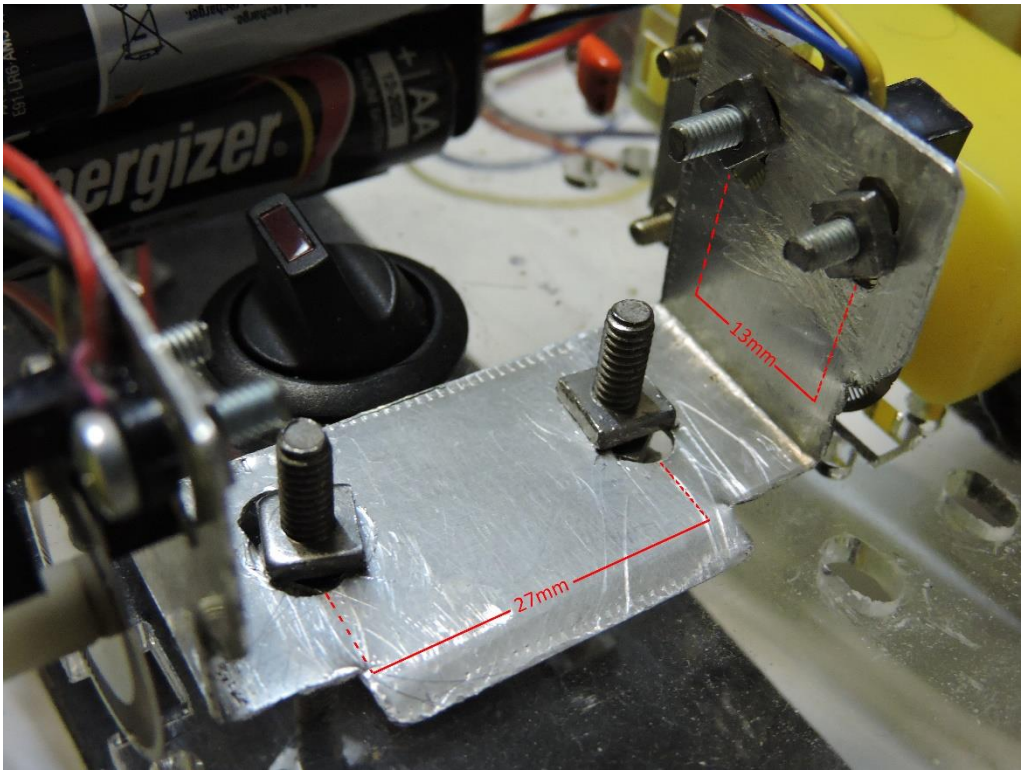


Εικόνα 5-4

Παρακάτω βλέπουμε τον τρόπο τοποθέτησης των αποκωδικοποιητών



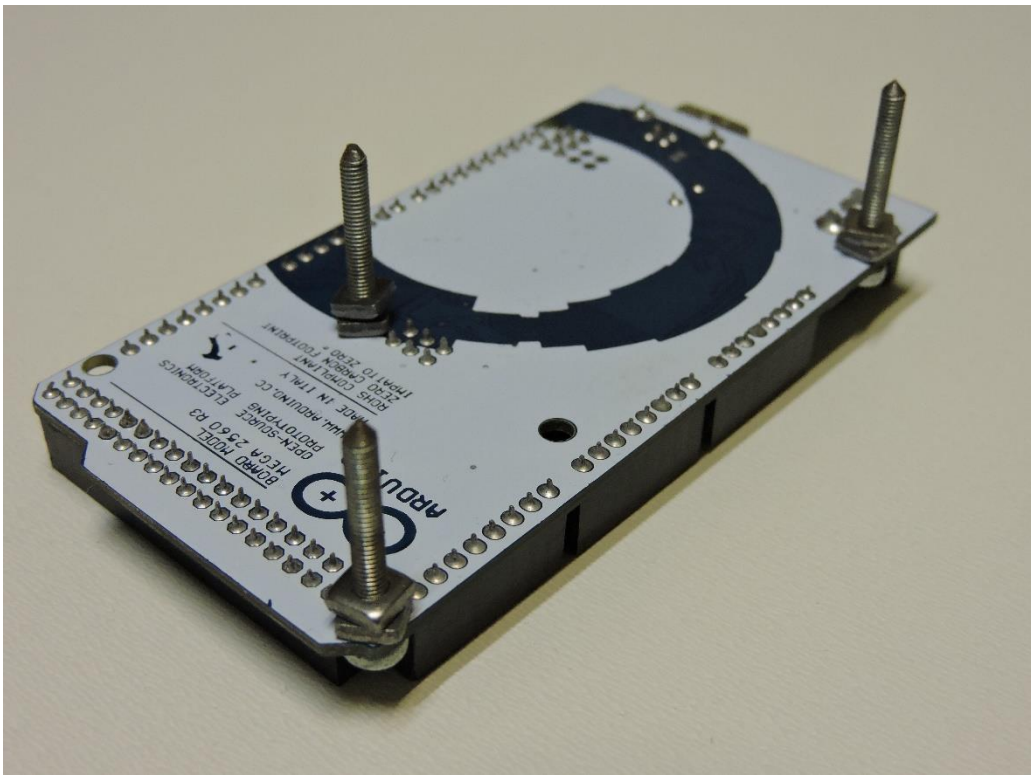
Εικόνα 5-5



Εικόνα 5-6

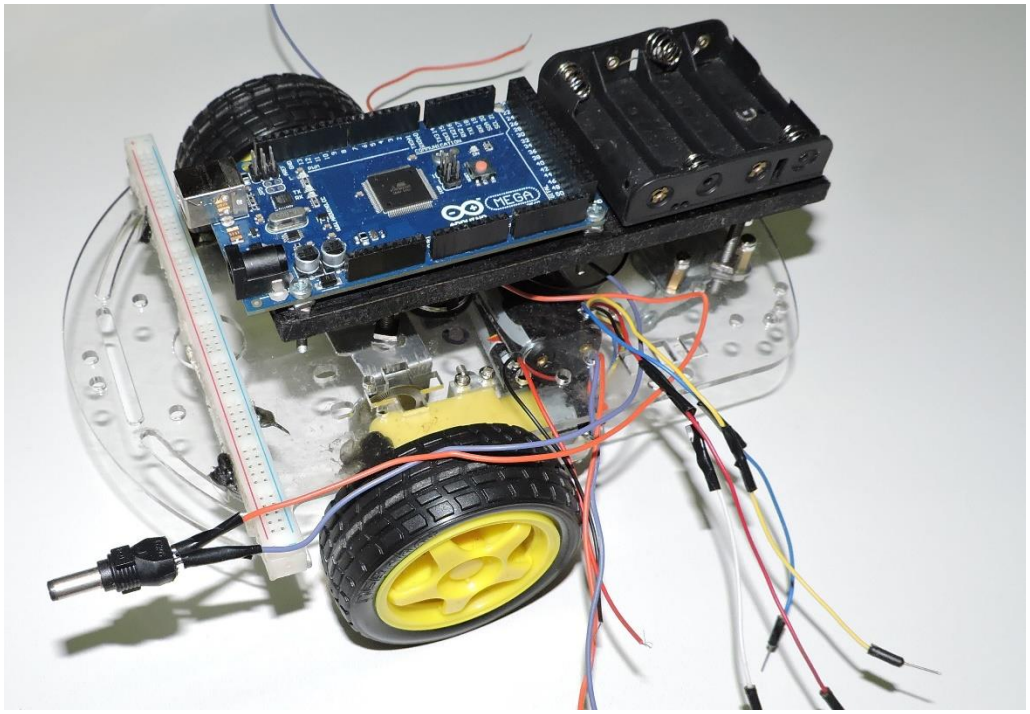
Η βάση έχει κατασκευαστεί από αλουμίνιο και έχουν ανοιχτεί επίτηδες οπές πλατύτερες και επιμήκεις ώστε να υπάρχει δυνατότητα ρύθμισης της θέσης του αποκωδικοποιητή ως προς τον τροχό.

5.1.2 Προσθήκη ηλεκτρονικών εξαρτημάτων

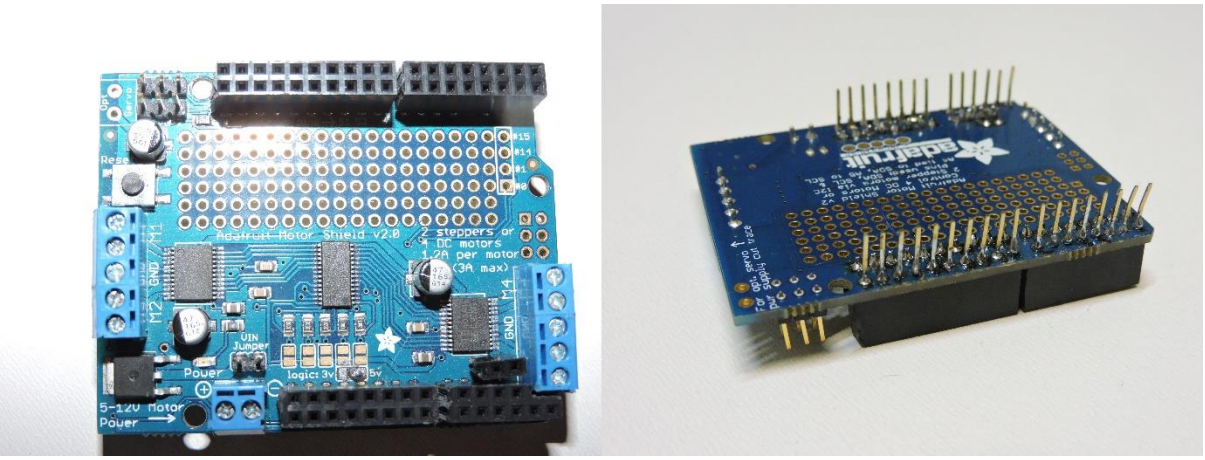


Εικόνα 5-7

Αρχικά προσθέτουμε την πλακέτα Arduino mega2560 r3 από την οποία και θα γίνεται ο έλεγχος του οχήματος. Έχουν τοποθετηθεί διπλά παξιμάδια από το κάτω μέρος της πλακέτας ώστε να μην υπάρχει άμεση επαφή των pins με την βάση και να αποφευχθεί έτσι πιθανός θόρυβος.

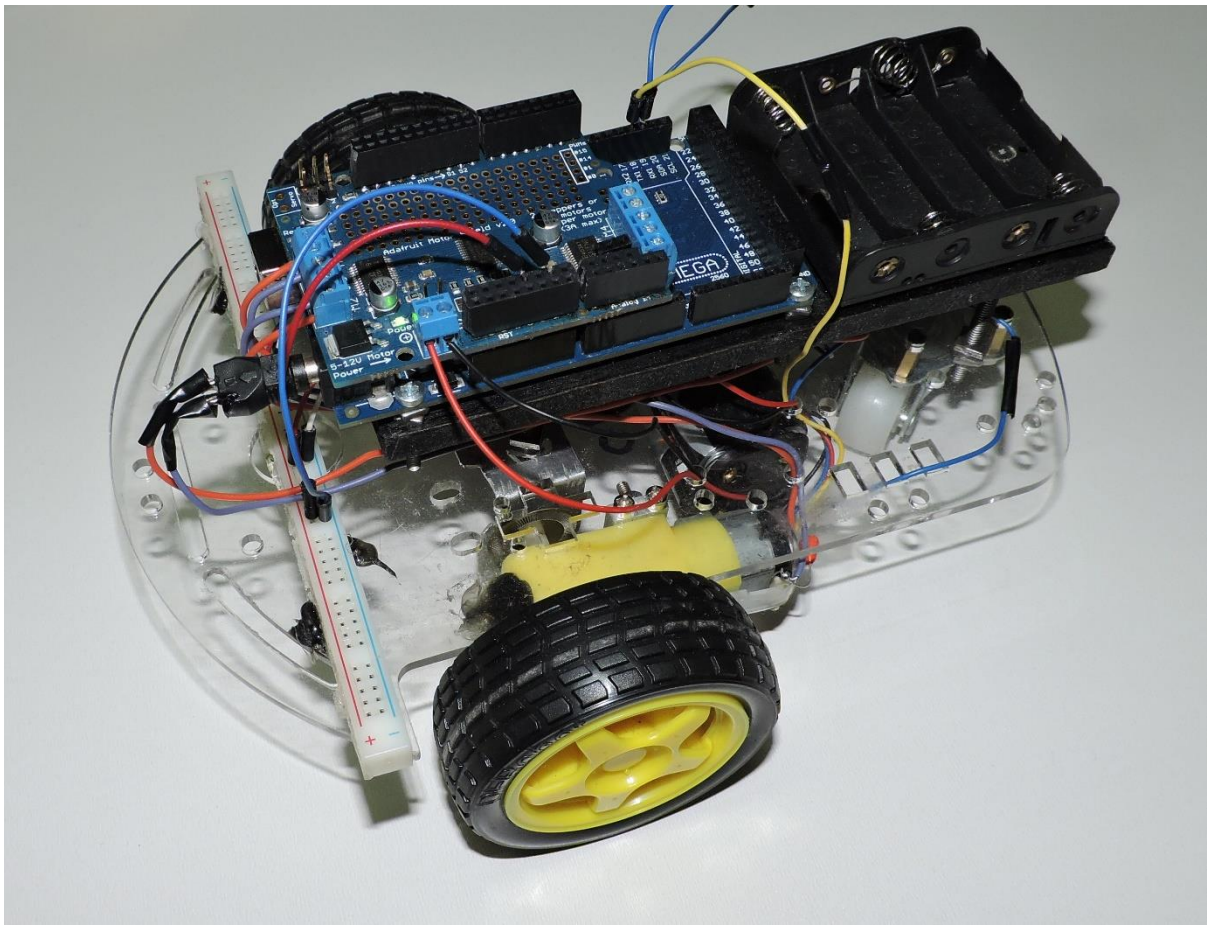


Εικόνα 5-8



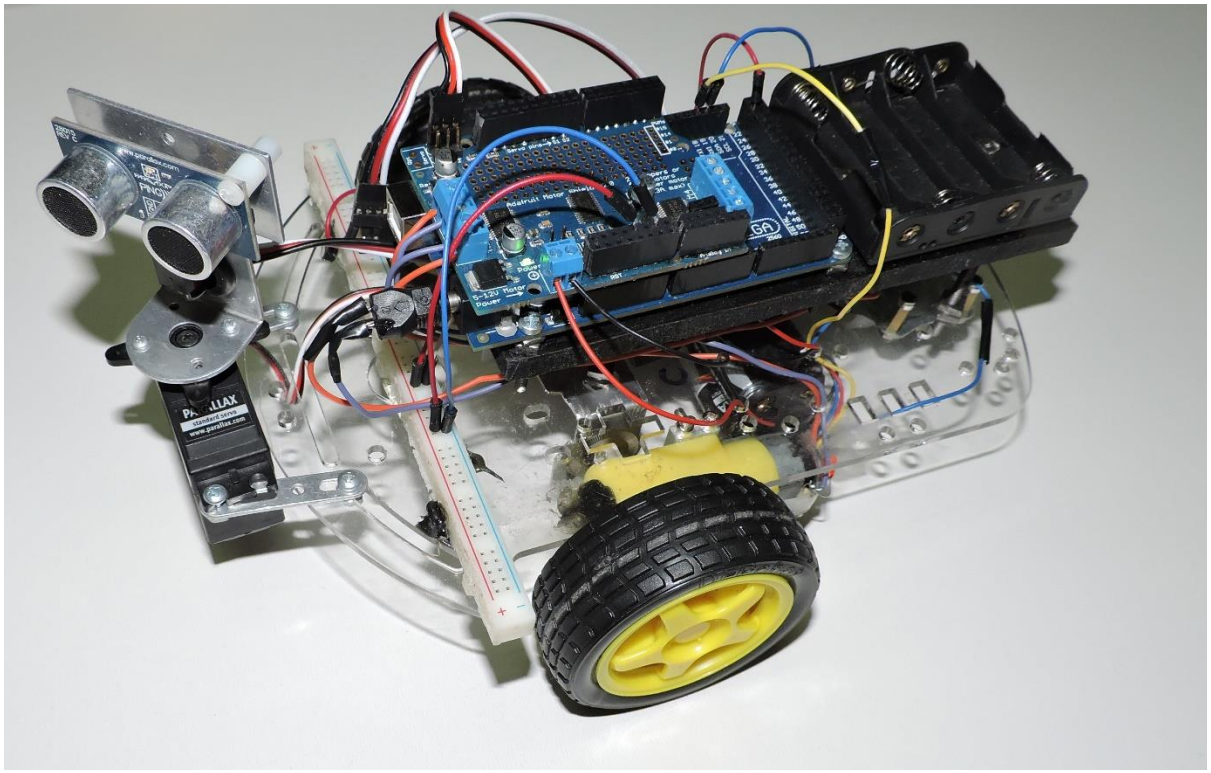
Εικόνα 5-9

Θέλοντας να έχουμε την ελευθερία να προσθέσουμε πολλαπλά shield προτιμήσαμε θηλυκές ακιδοσειρές με εκτενέστερες ακίδες.



Εικόνα 5-10

Τέλος προσθέτουμε και το αισθητήριο απόστασης PING))) στο μπροστινό μέρος του οχήματος μας ώστε να είναι σε θέση να ανιχνεύει και να αντιμετωπίζει εμπόδια κατά την διάρκεια της πορείας του.

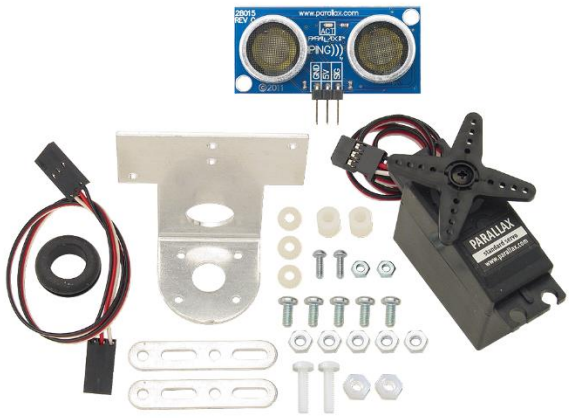


Εικόνα 5-11

5.2 Αρχές Λειτουργίας Εξαρτημάτων

Σε αυτό το κεφάλαιο περιγράφεται αναλυτικά η λειτουργία των επιμέρους εξαρτημάτων του ρομποτικού οχήματος.

5.2.1 Parallax PING)))



Είναι ένας αισθητήρας απόστασης ο οποίος τοποθετείται επάνω σε ένα σερβοκινητήρα ο οποίος μας επιτρέπει να περιστρέψουμε με μεγάλη ακρίβεια τον αισθητήρας μας από 0-180^{o}.

Η αρχή λειτουργίας του αισθητήρα είναι η εξής : στέλνουμε ένα ηχητικό κύμα υψηλής συχνότητας 40kHz από το ηχείο και μετράμε σε πόσο χρόνο το μικρόφωνο μας θα ανιχνεύσει της επιστροφή του ηχητικού κύματος. Γνωρίζοντας την ταχύτητα του ήχου στον αέρα (περίπου 1235km/h) μπορούμε εύκολα να υπολογίσουμε την απόσταση ενός αντικειμένου.

Οι περιορισμοί αυτού του αισθητήρα είναι ότι μπορεί να κάνει ανίχνευση αντικειμένου από 2cm έως και 5m με ακρίβεια εκατοστού. Επιπλέον δεν είναι σε θέση να ανιχνεύσει με ακρίβεια αντικείμενα με ανομοιογενή επιφάνεια (πχ την παλάμη του χεριού μας) καθώς τα ηχητικά κύματα διαχέονται πολύ και το μικρόφωνο δεν τα αντιλαμβάνεται.

5.2.2 DC geared motors

Οι κινητήρες είναι συνδεδεμένοι απευθείας με ρόδες διαμέτρου 7cm και πάχους 2,5cm. Τροφοδοτούνται με 3-6V και καθώς περιέχουν γρανάζια μπορούν να περιστραφούν έως 210 φορές το λεπτό. Στην τροφοδοσία των κινητήρων τοποθετήθηκαν πυκνωτές 700nF ώστε να μην υπάρχει πολύς θόρυβος κατά την λειτουργία τους και να μπορούμε αν τους ελέγξουμε με μεγαλύτερη ακρίβεια.

Ένα σημαντικό μειονέκτημα είναι ότι δεν σταθεροποιούνται ικανοποιητικά και επομένως κρίνεται απαραίτητο να πακτωθούν με χρήση σιλικόνης.

5.2.3 Μπαταρίες τροφοδοσίας

Προκειμένου να τροφοδοτήσουμε το αμαξίδιο χωρίς να είναι απαραίτητο να το συνδέσουμε με τροφοδοτικό χρησιμοποιήθηκαν 2 θήκες μπαταριών 4xAA. Προτιμήθηκε αυτή η λύση καθώς οι μπαταρίες μας παρέχουν πολύ σταθερή τάση και κοντά στην τιμή που κρίθηκε απαραίτητο, περίπου 6V (5,8V).

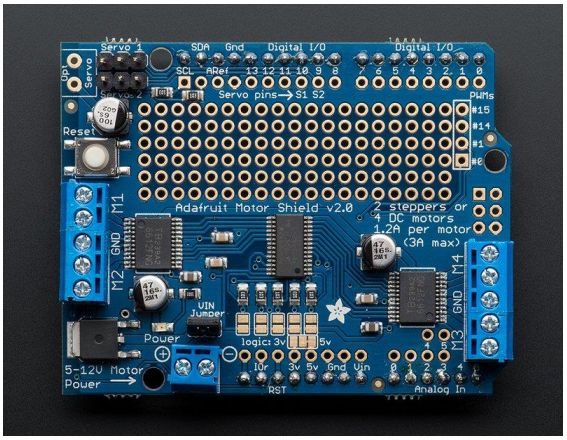
Χρησιμοποιήθηκαν δύο ώστε έχουμε περισσότερα διαθέσιμα αμπερώρια αλλά κυρίως ώστε να έχουμε γαλβανική απομόνωση πλακέτας-κινητήρων αφού οι αλλαγές στην ταχύτητα και ροπή των κινητήρων προκαλεί απότομες αλλαγές στο ρεύμα και την τάση που αντλούσαν από τις μπαταρίες με αποτέλεσμα την δυσλειτουργία της πλακέτας Arduino.

5.2.4 Σασί για κινητήρες και ρόδες

Η πλαστική αυτή βάση προτιμήθηκε καθώς ήταν κατασκευασμένη για ρομποτικά οχήματα. Διαθέτει εσοχές στα κατάλληλα σημεία ώστε να τοποθετηθούν οι κινητήρες με τις ρόδες καθώς και διάφορες άλλες για αντίστοιχες εφαρμογές.

Καθώς το συγκεκριμένο σασί δεν είχε αρκετό χώρο κατασκευάστηκε ένα επιπλέον από ξύλο στο οποίο και ανοίχτηκαν οι κατάλληλες υποδοχές ώστε να τοποθετηθούν επάνω σε αυτό η πλακέτα Arduino Mega2560 καθώς και μια πλαστική θήκη μπαταριών.

5.2.5 Adafruit motor shield v2

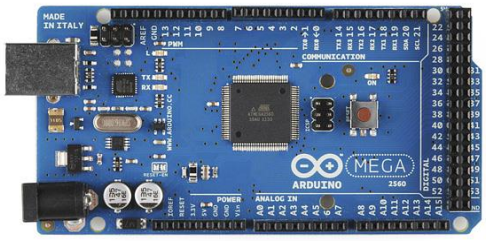


Η πλακέτα αυτή είναι ιδιαιτέρως χρήσιμη καθώς διευκολύνει τον έλεγχο των κινητήρων. Καθώς είναι shield ταιριάζει ακριβώς επάνω στην πλακέτα του Arduino μας χωρίς να στερεί την χρήση των pin , πλην αυτών των οποίων απαιτεί για την λειτουργία της (SDA, SCL, Analog pins 4, 5 και Digital pin 10). Με την βοήθεια αυτής της πλακέτας ρυθμίζουμε με απλές εντολές της

ταχύτητα και τη φορά περιστροφής των DC κινητήρων καθώς και την θέση του σερβοκινητήρα.

5.2.6 Arduino Mega2560 r3

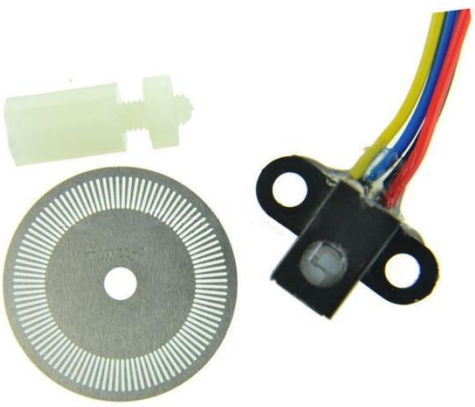
Αποτελεί το βασικότερο μέρος της κατασκευής αφού αναλαμβάνει πλήρως την αυτοματοποίηση της κατασκευής.



Οι πλακέτες Arduino είναι open-source. Αυτό σημαίνει ότι οποιοσδήποτε μπορεί να έχει πρόσβαση στα ηλεκτρολογικά σχέδια καθώς και στο προγραμματισμό τους. Έτσι κατόρθωσαν να αναπτυχθούν ιδιαίτερα γρήγορα με αποτέλεσμα να είναι ιδιαίτερα φτηνοί και κατάλληλοι για απλά project και εφαρμογές.

Είναι εξαιρετικά απλοί στον προγραμματισμό τους, παρέχουν δυνατότητα σύνδεσης άμεσα με στον υπολογιστή με χρήση καλωδίου USB A/B και το λογισμικό για τον προγραμματισμό τους (Arduino IDE) παρέχεται δωρεάν από την εταιρία.

Για την συγκεκριμένη πτυχιακή κρίθηκε καταλληλότερος ο Mega2560 r3 ο οποίος λειτουργεί με 5V που είναι πολύ βολικό σε σύγκριση με το εναλλακτικό των 3.3V καθώς τα περισσότερα αισθητήρια λειτουργούν με 5V, επιπλέον είναι αρκετά παλαιά πλακέτα ώστε να έχουν γίνει αρκετές διορθώσεις στην λειτουργία του. Επίσης υπάρχουν διαθέσιμα στο διαδίκτυο αρκετά προγράμματα και βιβλιοθήκες λόγο της διαδεδομένης χρήσης του. Τέλος διαθέτει περισσότερα pins από τις περισσότερες πλακέτες Arduino, αυτό την καθιστά μια ιδιαίτερα ευέλικτη πλακέτα με πολλές ελευθερίες.



5.2.7 Αποκωδικοποιητές

Οι αποκωδικοποιητές αποτελούνται από δύο μέρη, τον διάτρητο μεταλλικό τροχό και τον αισθητήρα. Ο τροχός τοποθετείτε με το κέντρο του στον άξονα της ρόδας του οχήματος ώστε να έχει τις ίδιες στροφές ανά λεπτό με την ρόδα. Ο αισθητήρας διαθέτει δυο δέσμες φωτός (η μια κάτω από την άλλη), καθώς περιστρέφεται ο τροχός διακόπτει αυτές της ακτίνες και σε κάθε διακοπή ο αισθητήρας παράγει ένα παλμό.

Αναλόγως με το πόσους παλμούς παίρνουμε σε ένα συγκεκριμένο χρονικό διάστημα είμαστε σε θέση να υπολογίσουμε και την ταχύτητα της ρόδας. Επιπλέον καθώς υπάρχουν δύο δέσμες φωτός ο τροχός διακόπτει πρώτα την μια και μετά την άλλη αναλόγως με την φορά περιστροφής πράγμα που μας δίνει την δυνατότητα να γνωρίζουμε ανά πάσα στιγμή και την κατεύθυνση του τροχού. Ο συγκεκριμένος τροχός διαθέτει 100 διατρήσεις, ο αισθητήρας λειτουργεί με 5V.

5.2.8 Διάφορα εξαρτήματα Arduino

- Για την κατασκευή απαιτήθηκαν πολύκλωνα καλώδια 1mm τύπου Θηλυκό-Θηλυκό, Αρσενικό-Αρσενικό και Θηλυκό-Αρσενικό καθώς και μονόκλωνα καλώδια.
- 2xΒίδες M2
- 17xΒίδες M3
- 4xΒίδες M4
- Αντίστοιχα παξιμάδια M3 και M4
- 4χαποστάτες M3 μήκους 1cm
- 1xSwivel Caster
- 2xΠυκνωτές 700nF
- 8xΜπαταρίες AA αλκαλικές
- 1xDc Barrel Jack
- 1xΔιακόπτης για τάσεις 5V

6 Πειράματα

Κατόπιν μιας θεωρητικής εισαγωγής μελετάμε την απόδοση του οχήματος στην πράξη με διάφορες δοκιμές.

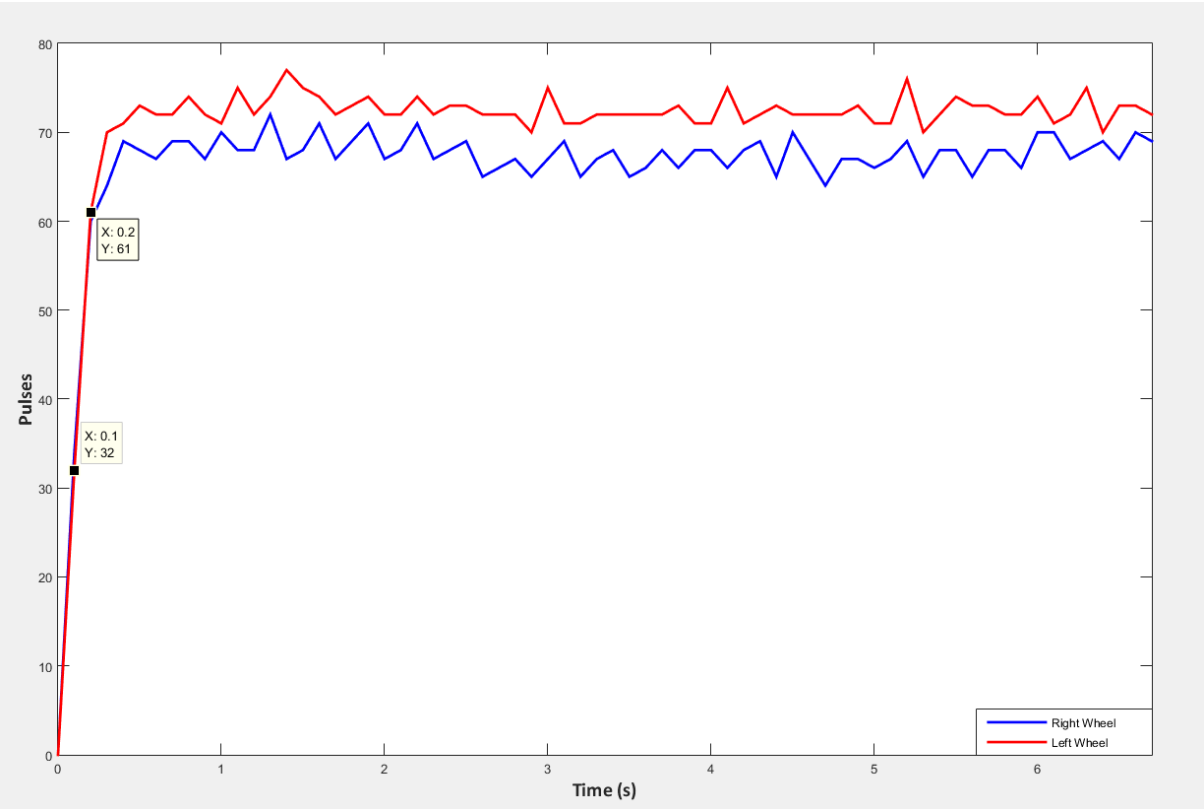
6.1 Θεωρητική Προσέγγιση

Στο κεφάλαιο αυτό αναλύεται η διαδικασία σχεδίασης του συστήματος καθώς και η θεωρητική προσέγγιση που έγινε αρχικά προκειμένου να επιτευχθεί το βέλτιστο αποτέλεσμα.

6.1.1 Κατασκευή της συνάρτησης μεταφοράς των κινητήρων

Προκειμένου να είμαστε σε θέση να υπολογίσουμε εύκολα την συνάρτηση μεταφοράς θα θεωρήσουμε ότι είναι πρώτου βαθμού. Ο έλεγχος των κινητήρων γίνεται με σταθερή τάση 6V αλλά μεταβάλλοντας το *duty cycle* χρησιμοποιώντας δύο κανάλια PWM του μικροελεγκτή (με εύρος τιμών από 0-255.)

Αρχικά θέλουμε να υπολογίσουμε πόσους παλμούς μετράει ο αποκωδικοποιητής σε διάστημα 0.1sec ενώ οι ρόδες λειτουργούν στην μέγιστη τους ταχύτητα.



Εικόνα 6-1

Όπως φαίνεται στο διάγραμμα η δεξιά ρόδα, η οποία είναι και ελαφρώς γρηγορότερη, κινείται με περίπου 72 παλμούς ανά 0.1sec ενώ η αριστερή με περίπου 68 παλμούς ανά 0.1sec,επομένως θα θέσουμε ως τον αριθμητή της συνάρτησης μας το 70. Εφόσον προσεγγίζουμε το σύστημα σαν πρωτοβάθμιο και προκειμένου να υπολογίσουμε την χρονική μεταβλητή της συνάρτησης μας, χρειάζεται να γνωρίζουμε σε πόσο χρόνο το σύστημα φτάνει το 63.2% της μέγιστης τιμής του, δηλαδή σε πόσο χρόνο φτάνει τους 44 παλμούς ανά 0.1sec.

Εδώ η ανάλυση των αποκωδικοποιητών εισάγει έναν πολύ βασικό περιορισμό καθώς δεν παίρνουμε αρκετά δείγματα για το διάστημα στο οποίο το σύστημα φτάνει από το 0 στο 44 Η απόκριση του συστήματος όμως φαίνεται να είναι αρκετά κοντά σε γραμμική, είμαστε λοιπόν σε θέση να υπολογίσουμε μια χρονική τιμή αρκετά κοντά στην πραγματική:

$$\tau = \frac{44 \times 0.1}{30} = 0.145$$

Συνδυάζοντας τα παραπάνω συμπεράσματα είμαστε πλέον σε θέση να εξάγουμε την συνάρτηση μεταφοράς του συστήματος:

$$G = \frac{70}{0.145s + 1}$$

Αφού η συνάρτηση μεταφοράς είναι πρώτου βαθμού γνωρίσουμε ότι ένας ελεγκτής PI ο οποίος περιέχει δύο μεταβλητές τιμές είναι ικανός να ελέγξει το σύστημα μας. Ο ελεγκτής PI είναι της μορφής:

$$G_c = \mathrm{K}\frac{(s+z)}{s}$$

Η συνάρτηση ανοιχτού βρόγχου που προκύπτει λοιπόν έχοντας τον ελεγκτή σε σειρά είναι:

$$GG_c = \frac{70K(s+z)}{0.145s^2 + s}$$

Επομένως κλείνοντας τον βρόγχο έχουμε την τελική μορφή της συνάρτησης μας:

$$\mathrm{T} = \frac{70K(s+z)}{0.145s^2 + (70\mathrm{K}+1)s + 70Kz} = \frac{483K(s+z)}{s^2 + (483\mathrm{K}+7)s + 483Kz}$$

Στοχεύοντας στο να έχουμε την βέλτιστη απόκριση στο σύστημα μας θα ρυθμίσουμε τις μεταβλητές Κ,τ σύμφωνα με την συνάρτηση βέλτιστης απόκρισης:

$$\mathrm{T}_2 = \frac{\omega_n^2}{s^2 + 1.4\omega_n s + \omega_n^2}$$

Για να υπολογίσουμε το επιθυμητό ω_n θα χρησιμοποιήσουμε την εξίσωση του χρόνου αποκατάστασης σύμφωνα με το κριτήριο του 2%.

$$\mathrm{T}_s = \frac{4}{\zeta\omega_n} = 0.2\,\mathrm{sec} \xrightarrow{\zeta=0.707} \omega_n = 28r/s$$

Επομένως η συνάρτηση βέλτιστης απόκρισης γίνεται.

$$\mathrm{T}_2 = \frac{\omega_n^2}{s^2 + 40s + 784}$$

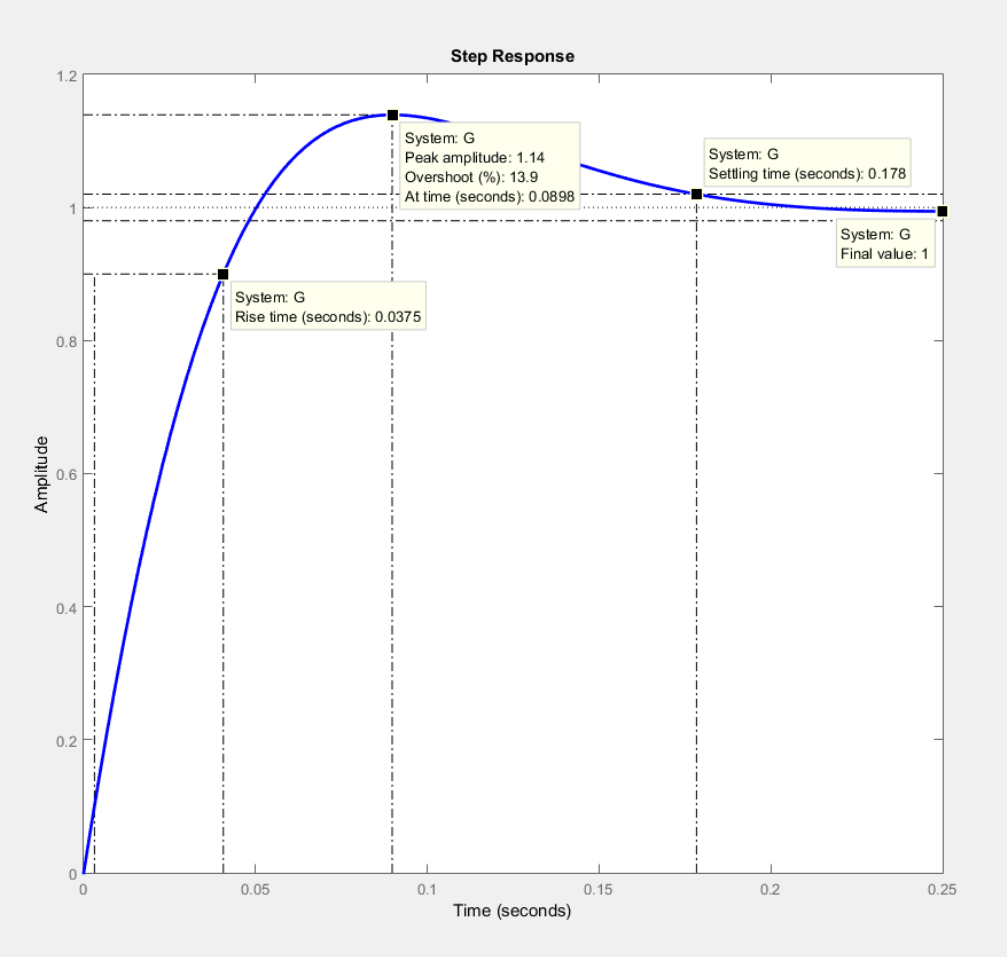
Εξισώνοντας τις δύο συναρτήσεις υπολογίζουμε τις τιμές των Κ και z.

$$483\mathrm{K} + 7 = 40 \Rightarrow \mathrm{K} = 0.07 \qquad 483Kz = 784 \Rightarrow z = 23.2$$

Αντικαθιστώντας λοιπόν στην συνάρτηση μεταφοράς έχουμε.

$$T = \frac{33(s+23.2)}{s^2+40s+784} = \frac{33s+784}{s^2+40s+784}$$

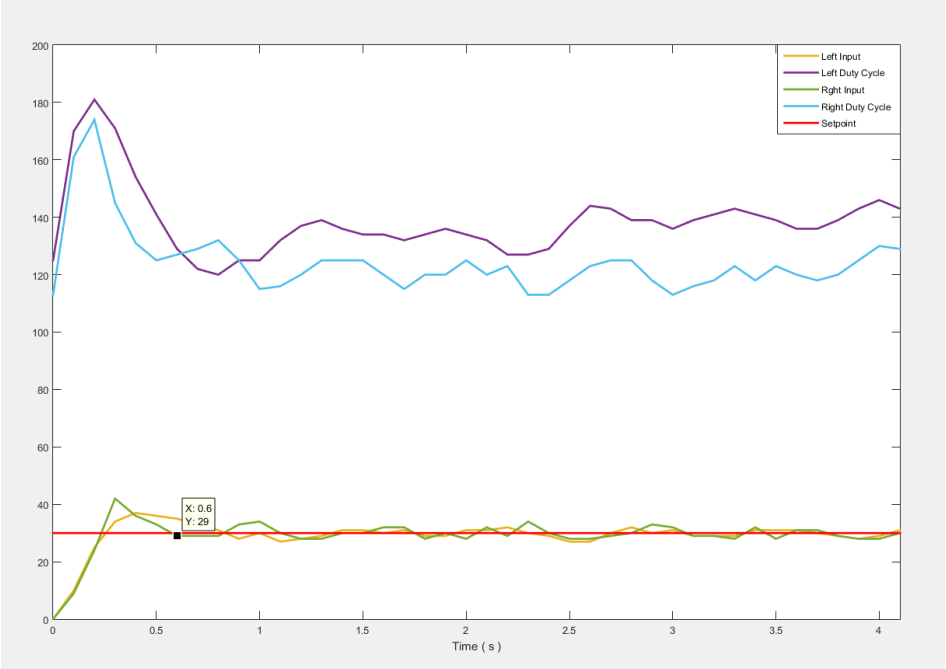
Κάνοντας προσομοίωση στο MATLAB μπορούμε να δούμε την απόκριση που θα είχε το σύστημα εάν ήταν πραγματικά πρώτου βαθμού.



Εικόνα 6-2

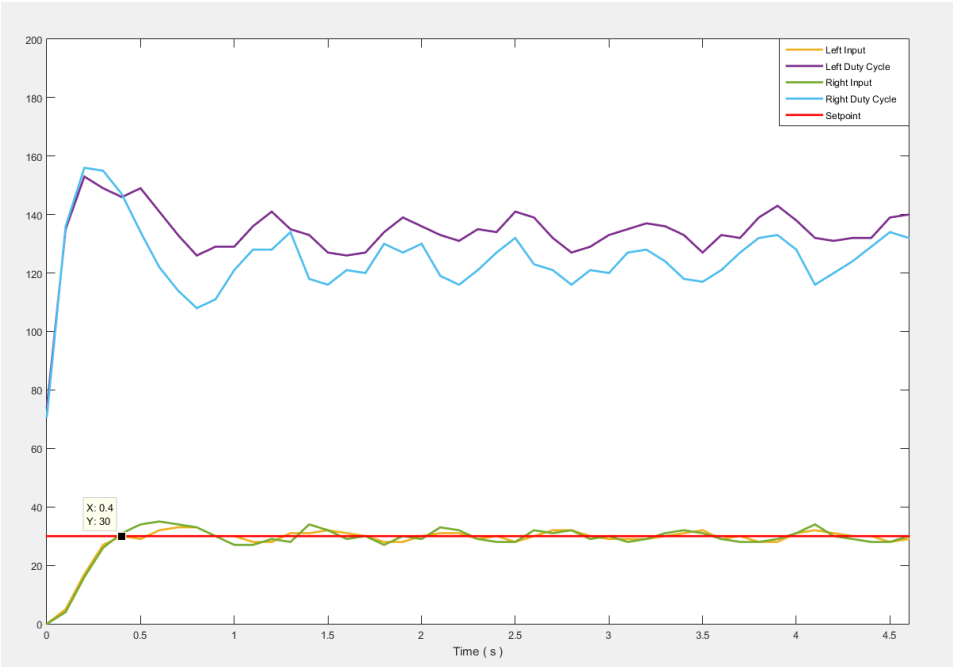
6.1.2 Ρύθμιση του ελεγκτή

Χρησιμοποιούμε τις υπολογισμένες τιμές ως ενδεικτικές για να ξεκινήσουμε να ρυθμίζουμε τον ελεγκτή μας. Θέτοντας $K_P = 0.07$ και $K_I = 23.2$ παίρνουμε την παρακάτω απόκριση.



Εικόνα 6-3

Παρατηρείτε ότι ο χρόνος αποκατάστασης καθώς και η υπερύψωση διαφέρει σημαντικά από τις υπολογισμένες τιμές. Προκειμένου να έχουμε μια καλύτερη απόκριση θέτουμε πειραματικά τις μεταβλητές μας στις εξής τιμές: $K_P = 1.053$, $K_I = 23.2$ και έχουμε την παρακάτω απόκριση.



Εικόνα 6-4

Αυξάνοντας το K_P μειώσαμε σημαντικά την υπερύψωση και τον χρόνο αποκατάστασης

6.2 Πειραματικά Δεδομένα

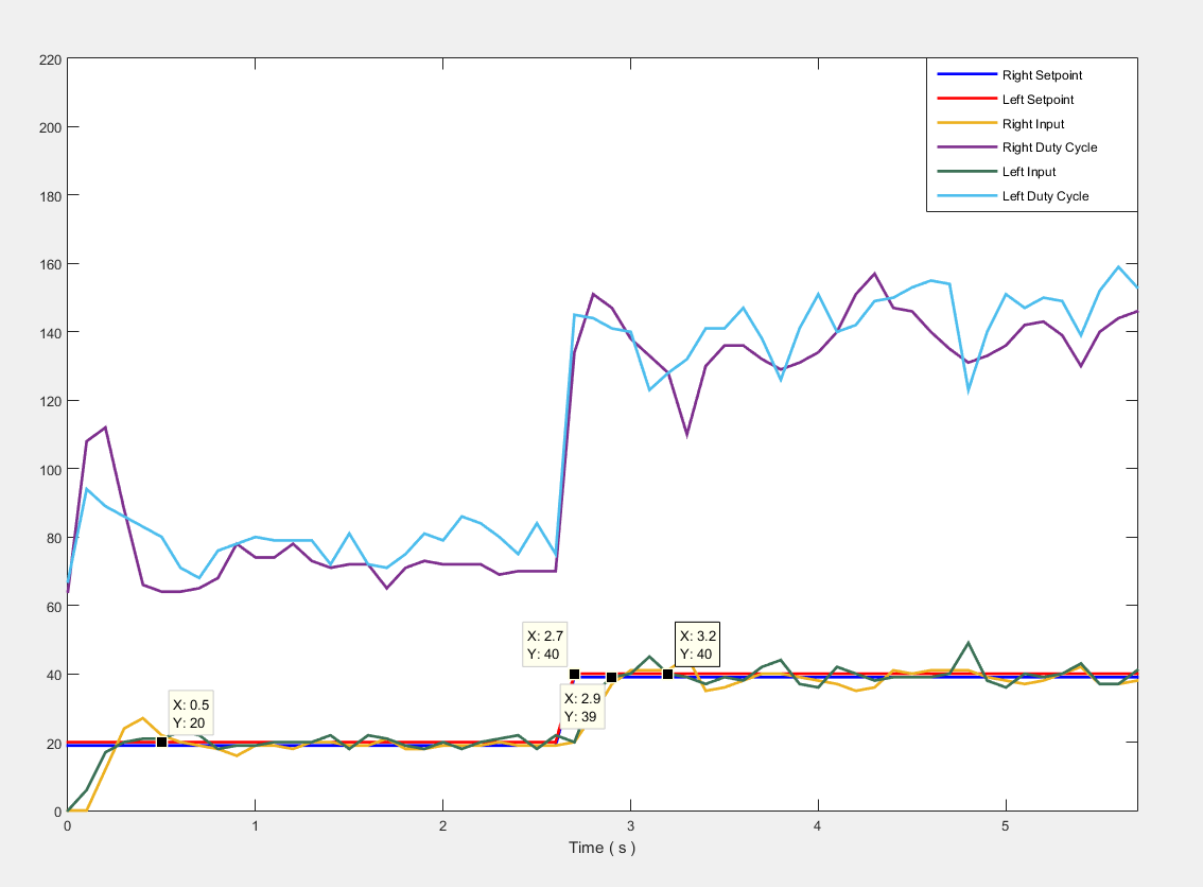
Σε αυτό το κεφάλαιο φαίνεται η λειτουργία του οχήματος στην πράξη καθώς και η αποτελεσματικότητα του. Τα πειράματα έχουν γίνει σε εσωτερικό χώρο και ως εκ τούτου δεν υπάρχουν εξωτερικές επιρροές (πχ αέρας). Προκειμένου να παίρνουμε δεδομένα κατά την διάρκεια της εκτέλεσης του πειράματος συνδέσαμε την πλακέτα Arduino με Η/Υ μέσω ενός καλωδίου USB A/B μήκους 4,5m.

6.2.1 Απόκριση PID σε βηματική μεταβολή

Έχουμε ρυθμίσει το όχημα ώστε να κινείται με ταχύτητα 20 παλμών/δέκατο του δευτερολέπτου. Σε χρόνο 3 δευτερολέπτων (από στην στιγμή εκκίνησης του προγράμματος) έχουμε ρυθμίσει μια αυτόματη επιτάχυνση 100%.

6.2.1.1 Χωρίς χρήση φίλτρου εξομάλυνσης

Εδώ ο ελεγκτής μας είναι ρυθμισμένος στις τιμές $K_P = 1.053, K_I = 23.4$ και $K_D = 0$.

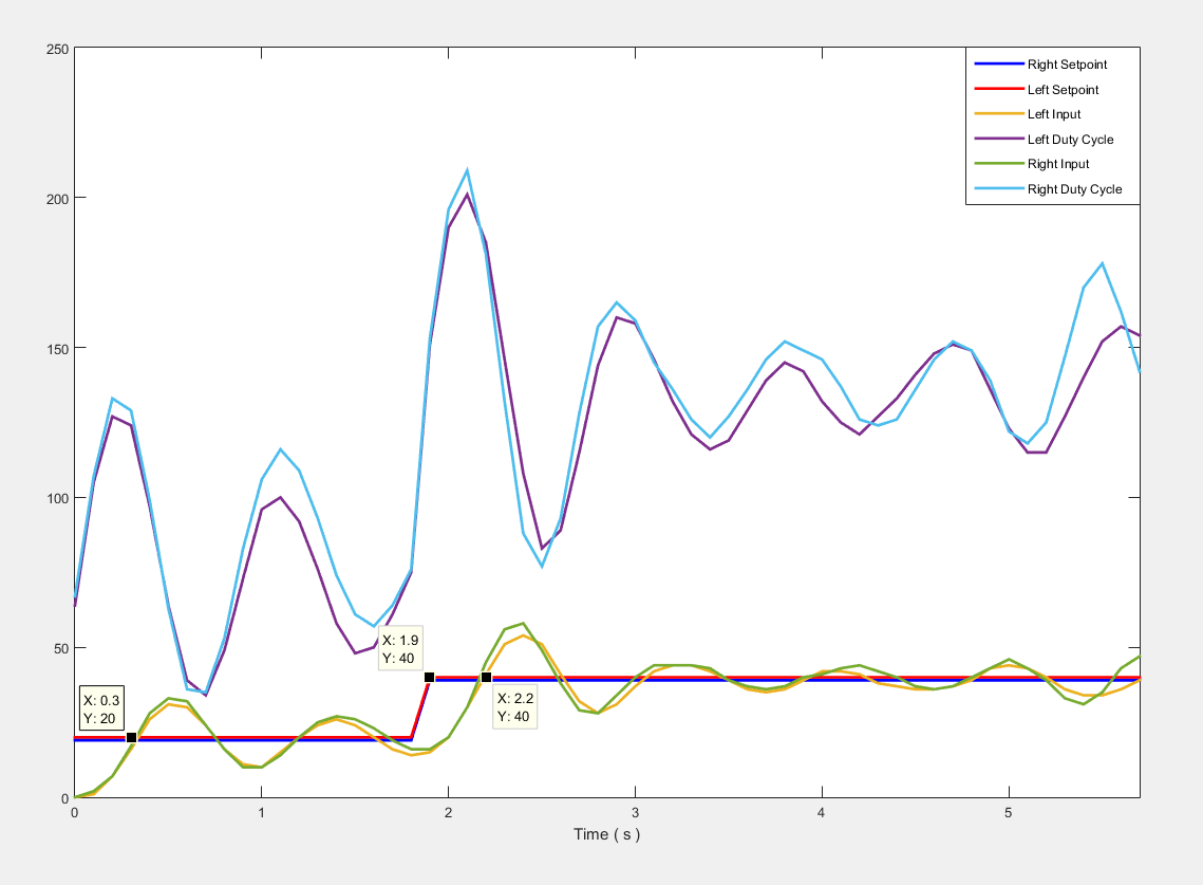


Εικόνα 6-5

Όπως φαίνεται στο διάγραμμα ο ελεγκτής μας είναι αρκετά γρήγορος αφού φτάνει την επιθυμητή τιμή σε 0.2 δευτερόλεπτα ενώ σταθεροποιείται σε 0.5 δευτερόλεπτα. Γίνεται φανερό όμως ότι οι τιμές που διαβάζονται από τους αποκωδικοποιητές δεν είναι ιδιαίτερα σταθερές με αποτέλεσμα οι ταχύτητα στις ρόδες να μεταβάλλεται χωρίς λόγο. Αυτές οι διαταραχές εμποδίζουν τον ελεγκτή να ευθυγραμμίσει το όχημα, έτσι προγραμματίζουμε εμπειρικά ελαφρώς διαφορετικά *setpoint* ώστε το όχημα να κινείται ευθεία. Προκειμένου να διορθώσουμε το πρόβλημα των διαταραχών συμπεριλάβαμε στο πρόγραμμα μας μια συνάρτηση εξομάλυνσης ώστε να παίρνουμε τον μέσω όρο των τιμών ανά 3 δείγματα. Είναι αναμενόμενο ότι αυτή η συνάρτηση θα εισάγει μια καθυστέρηση στο σύστημα η οποία θα αλλάξει και την απόκριση του ελεγκτή.

6.2.1.2 Με φίλτρο εξομάλυνσης

Χρησιμοποιούμε τον ελεγκτή χωρίς να κάνουμε καμία αλλαγή ώστε να ελέγξουμε την απόκριση του.

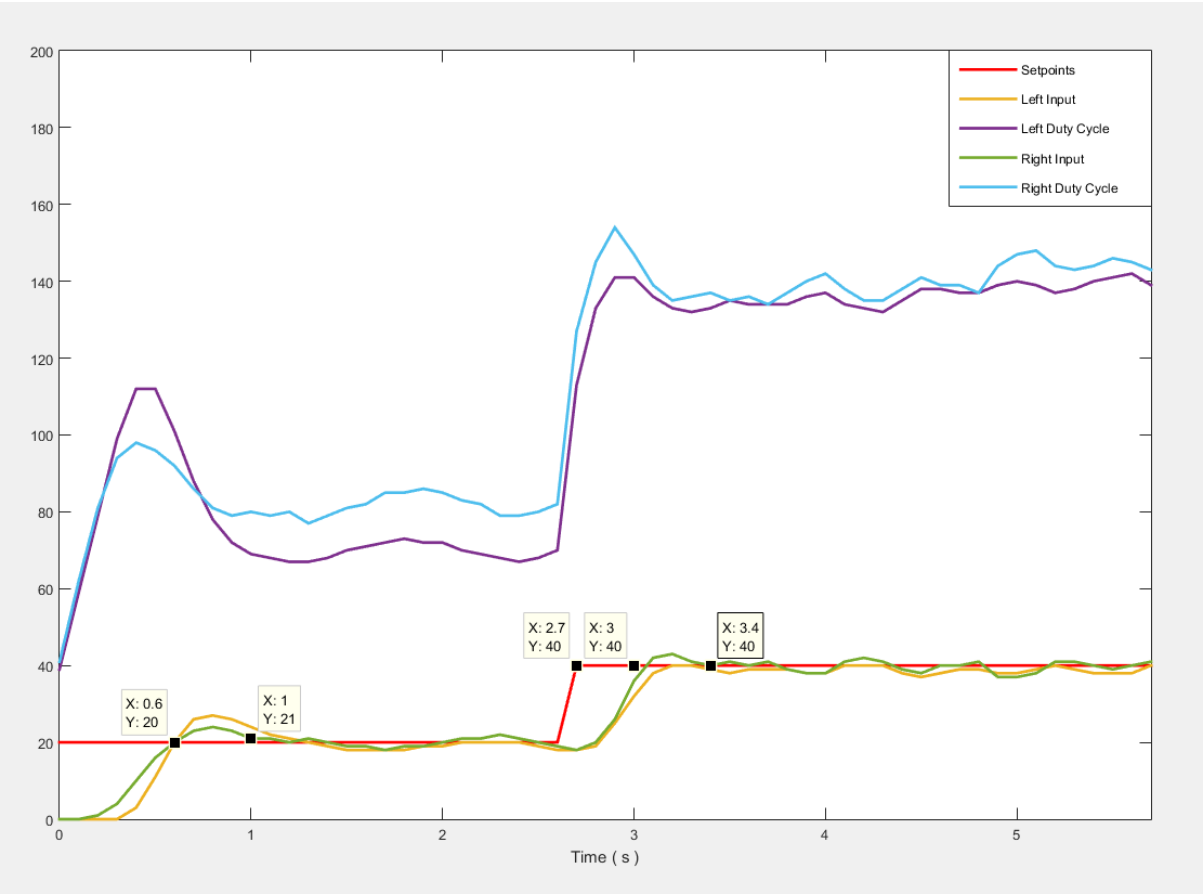


Εικόνα 6-6

Η εισαγωγή της καθυστέρησης που προέκυψε από την συνάρτηση εξομάλυνσης είχε μεγάλη επιρροή στο σύστημα μας. Πλέον αργεί να σταθεροποιηθεί στην επιθυμητή τιμή με αποτέλεσμα να έχει απότομες επιβραδύνσεις και επιταχύνσεις. Κρίνεται λοιπόν, απαραίτητο να ρυθμίσουμε τον ελεγκτή μας ώστε να έχει μια ομαλότερη απόκριση.

6.2.1.3 Με φίλτρο εξομάλυνσης και ρύθμιση

Μετά από κάποιες πειραματικές δόκιμες καταλήξαμε ότι πρέπει να μειωθεί η **ολοκληρωτική** δράση του ελεγκτή. Επομένως ρυθμίζουμε τον PID στις τιμές $K_P = 1.053$, $K_I = 10.4$ και $K_D = 0$.

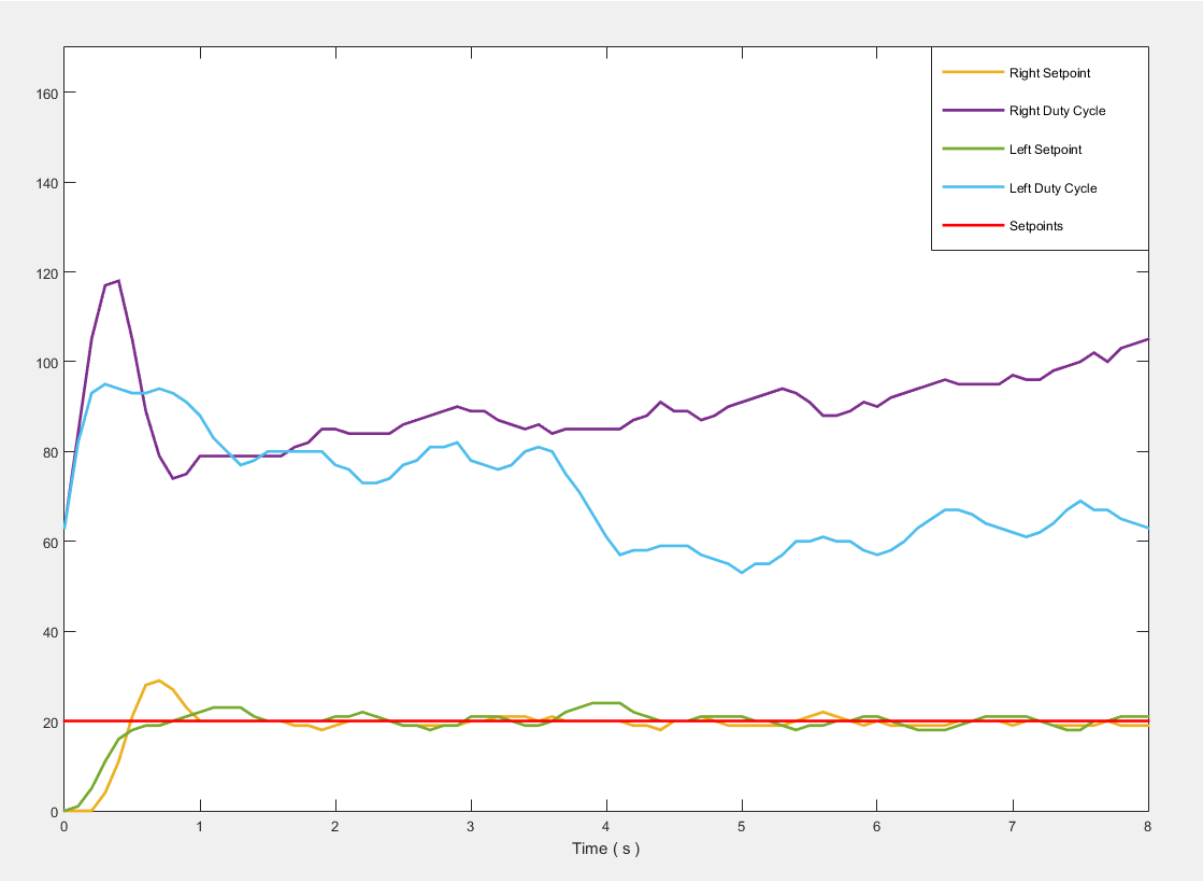


Εικόνα 6-7

Με κατάλληλη ρύθμιση καταφέραμε να έχουμε μια ικανοποιητική απόκριση. Ο ελεγκτής κατορθώνει να φτάσει την επιθυμητή τιμή σε 0.3 δευτερόλεπτα ενώ σταθεροποιείται σε 0.7 δευτερόλεπτα. Η χρήση φίλτρου λοιπόν επιβαρύνει το σύστημα με μόλις 0.2 δευτερόλεπτα αλλά σταθεροποιεί κατά πολύ όπως φαίνεται την ταχύτητα στις ρόδες πράγμα που μας προσφέρει ομαλότερη πορεία και αλλά και χαμηλότερη κατανάλωση ισχύος. Τέλος, καθώς οι τιμές που παίρνουμε από τους αποκωδικοποιητές είναι πλέον σταθερότερες ο ελεγκτής είναι σε θέση να ευθυγραμμίσει το όχημα χωρίς να απαιτείται παρέμβαση από χειριστή.

6.2.2 Ευθυγράμμιση οχήματος

Με αυτό το πείραμα κάνουμε εμφανής την δράση του PID. Αρχικά κάνουμε το όχημα να κινηθεί σε ευθεία πορεία, μετά από διάστημα 3.7 δευτερολέπτων εισάγουμε στην δεξιά ρόδα μια διαταραχή 20 παλμών αναπαριστώντας έτσι μια εξωτερική επίδραση που θα μπορούσε να ασκηθεί στο όχημα σε εξωτερικό περιβάλλον η οποία θα επιτάχυνε την δεξιά ρόδα και παρατηρούμε πώς ο ελεγκτής θα επαναφέρει το όχημα σε ευθύγραμμη πορεία.

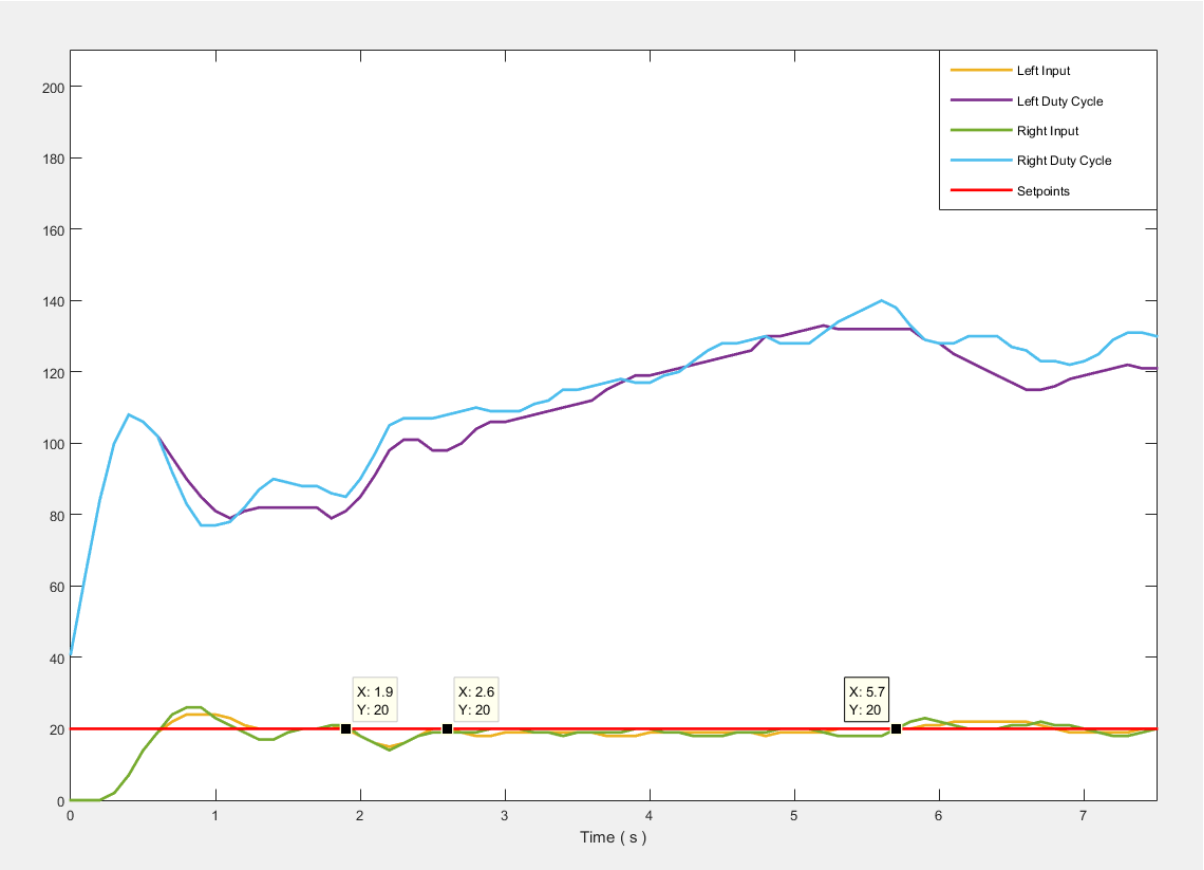


Εικόνα 6-8

Ο ελεγκτής είναι σε θέση να αντιμετωπίσει την διαταραχή, όπως γίνεται εμφανές και στο παραπάνω διάγραμμα, μειώνοντας αντίστοιχα τους παλμούς εξόδου κατά 20. Πρακτικά αυτό σημαίνει ότι το όχημα θα συνεχίσει ευθύγραμμη πορεία αλλάζοντας όμως ελαφρά κατεύθυνση αφού στο διάστημα που διαμεσολάβησε μέχρι ο ελεγκτής να αντιμετωπίσει την διαταραχή το όχημα έχει προλάβει να στρίψει ελαφρώς προς τα αριστερά.

6.2.3 Μετάβαση σε κεκλιμένο επίπεδο

Σκοπός του πειράματος είναι να δειχθεί το πώς ο ελεγκτής φροντίζει να διατηρεί σταθερή την ταχύτητα του οχήματος όταν αυτή αλλάζει λόγο εξωτερικών παραγόντων όπως στην συγκεκριμένη περίπτωση η αλλαγή κλίσης του εδάφους. Το όχημα θα διανύσει μια απόσταση περίπου 30cm και έπειτα θα ανέβει μια ανηφόρα μήκους 142cm με κλίση 7°.



Εικόνα 6-9

Από τη στιγμή που ξεκινάει να ανεβαίνει την ράμπα διορθώνει το μεγαλύτερο μέρος του σφάλματος σε 0.7 δευτερόλεπτα όμως είναι σε θέση να επαναφέρει πλήρως την ταχύτητα σε 3.8 δευτερόλεπτα.

6.3 Κοστολόγηση υλικών

Στον παρακάτω πίνακα φαίνονται τα επιμέρους εξαρτήματα της πτυχιακής καθώς και οι τιμές στις οποίες αγοράστηκαν. Ένα σημαντικό τμήμα των υλικών διατέθηκε από τον εισηγητή καθηγητή κ.Χρήστο Υφούλη, πάραυτα τοποθετήθηκαν τιμές ώστε να είναι εμφανής το οικονομικό κόστος της κατασκευής.

Πίνακας 1

Όνομα εξαρτήματος	**Κατάστημα**	**Ποσότητα**	**Τιμή (€)**
Arduino Mega2560 r3	http://www.ebay.com	1	4.2
adafruit motorshield v2	Διατέθηκε από εισηγητή καθηγητή κ.Χρήστο Υφούλη	1	20
οπτικός αποκωδικοποιητής	http://www.ebay.com	2	6
DC κιντήρας με ρόδα	Διατέθηκε από εισηγητή καθηγητή κ.Χρήστο Υφούλη	2	3.8
αισθητήρας απόστασης PING))) με σερβοκινητήρα	Διατέθηκε από εισηγητή καθηγητή κ.Χρήστο Υφούλη	1	49.99
θήκες μπαταριών AA	Μουτσιούλης	2	0.8
ακιδοσειρές	http://www.robotstore.gr	2	8.54
Μπαταρίες AA	Μουτσιούλης	8	7
Λοιπά υλικά	Διαθέσημα από προηγούμενες εργασίες	-	10
		Σύνολο	110.3

7 Προγραμματιστικό μέρος

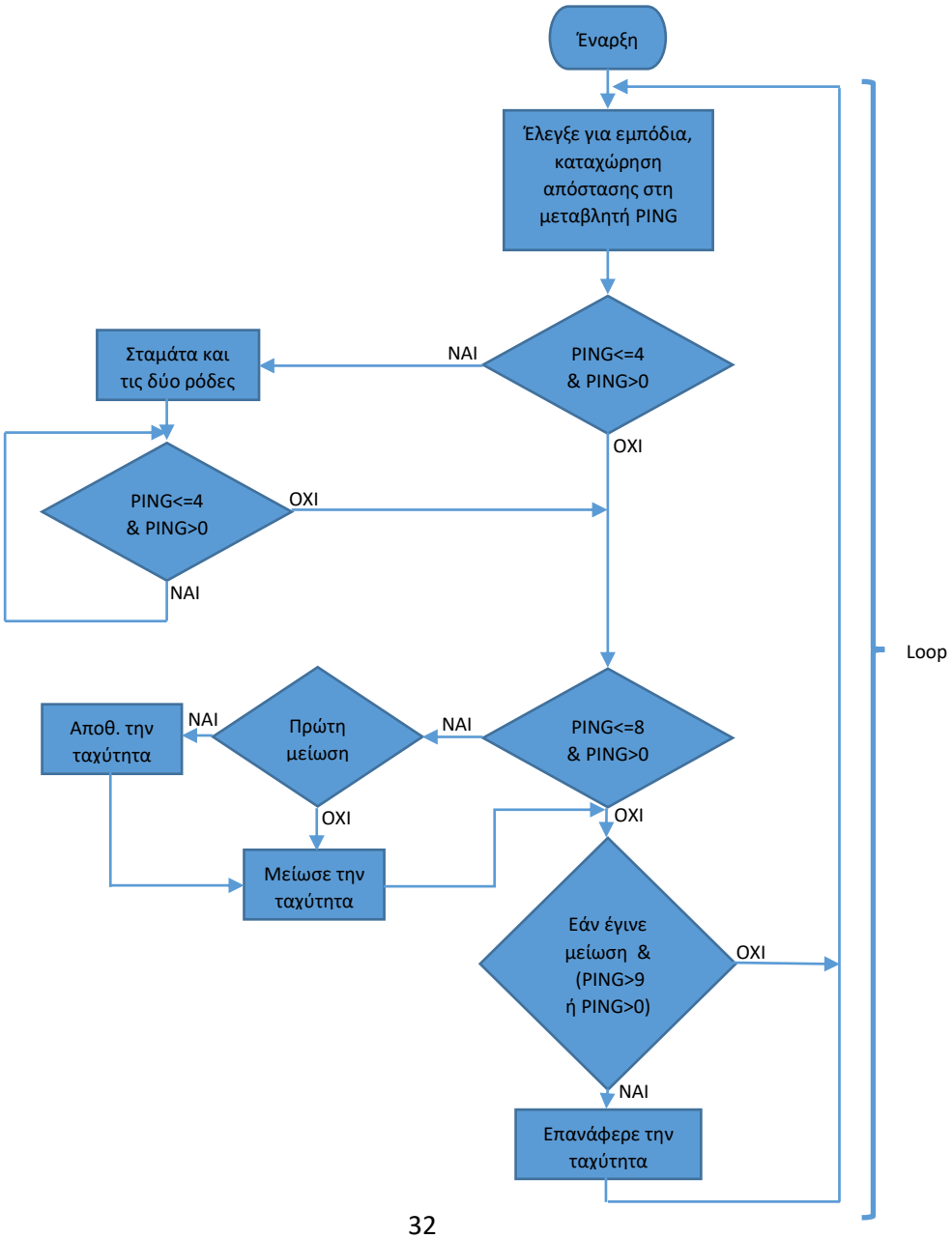
Παρακάτω αναλύεται ο κώδικας που χρησιμοποιήθηκε για την υλοποίηση του αυτοματισμού αρχικά με διαγράμματα ροής και τέλος ο πηγαίος κώδικας με σχόλια

7.1 Διαγράμματα ροής

7.1.1 Αισθητήρας απόστασης PING)))

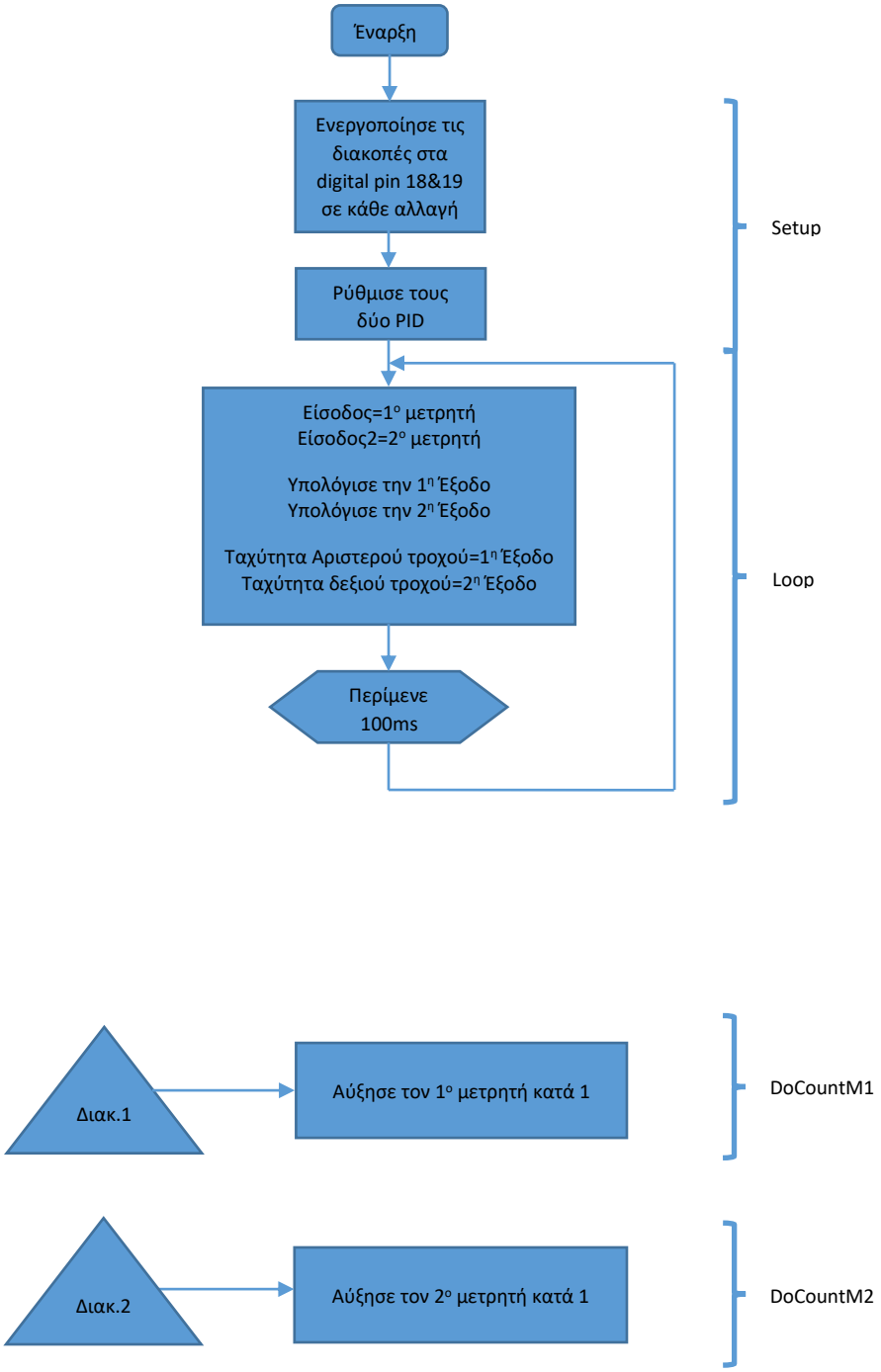
Ο ρόλος του αισθητήρα απόστασης είναι να ανιχνεύει πιθανά εμπόδια κατά την διάρκεια της πορείας του οχήματος και να επιδρά αντίστοιχα στην ταχύτητα αυτού. Οι κανόνες που έχουμε θέσει είναι οι εξής:

- Εάν εμφανιστεί εμπόδιο σε απόσταση **8cm** ξεκίνα να μειώνεις σταδιακά την ταχύτητα του οχήματος (έτσι αν το εμπόδιο είναι κινητό η ταχύτητα θα μειωθεί έως ότου να το ακολουθεί εάν είναι ακίνητο τότε θα ακινητοποιηθεί σταδιακά.)
- Εάν εμφανιστεί εμπόδιο σε απόσταση μικρότερη ή ίση από **4cm** τότε ακινητοποίησε το όχημα (έτσι εάν εμφανιστεί απότομα εμπόδιο μπροστά στο όχημα τότε αυτό ακινητοποιείται ακαριαία.



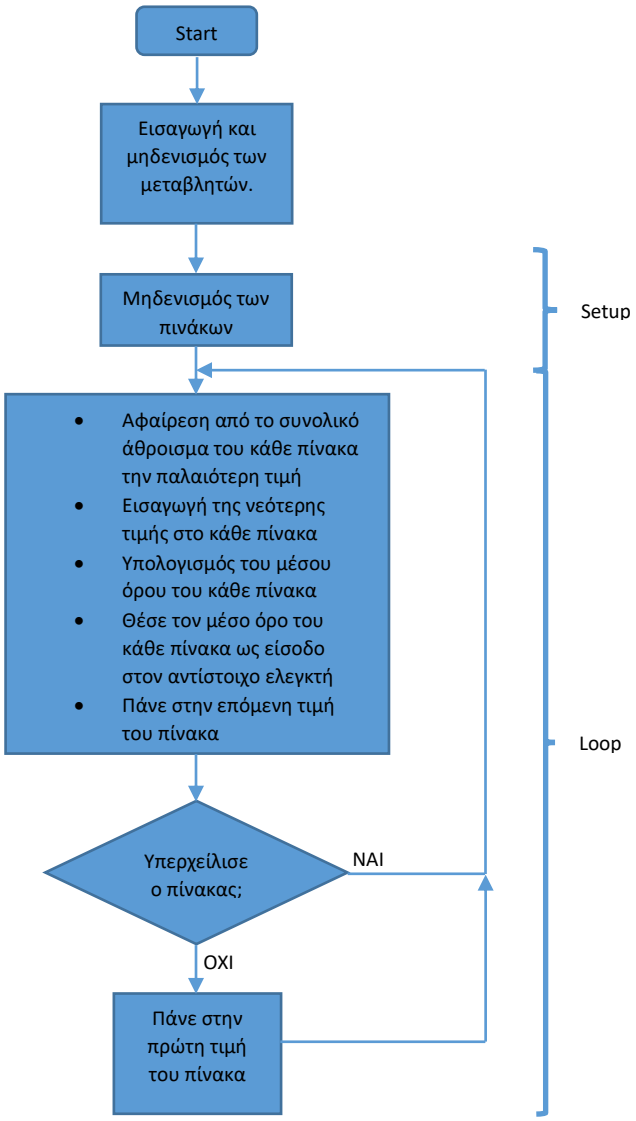
7.1.2 Έλεγχος ταχύτητας

Χρησιμοποιώντας τους αποκωδικοποιητές που έχουμε προσαρμόσει στους άξονες των τροχών υπολογίζουμε την ταχύτητα της κάθε ρόδας και βάζοντας την ως είσοδο σε έναν ελεγκτή PID ρυθμίζουμε την ταχύτητα ώστε να είναι πάντα η επιθυμητή (μετατρέπουμε δηλαδή το σύστημα από ανοιχτού σε κλειστού βρόγχου.)



7.1.3 Συνάρτηση εξομάλυνσης

Χρησιμοποιώντας αυτή τη συνάρτηση σκοπός μας είναι να εξομαλύνουμε τις απότομες μεταβολές που παίρνουμε από τους αποκωδικοποιητές έτσι ο ελεγκτής θα είναι σε θέση να παράγει ένα σταθερότερο *duty cycle* στις ρόδες με αποτέλεσμα το όχημα να κινείται ομαλότερα.



7.2 Κώδικας Arduino

```
#include <PID_v1.h>
#include <MsTimer2.h>
#include <Wire.h>
#include <Adafruit_MotorShield.h>
#include "utility/Adafruit_MS_PWMServoDriver.h"
#include <NewPing.h>
#include <Servo.h>

#define TRIGGER_PIN  22  // Δεσμευμένο pin του Arduino για ενεργοποίηση του πομπού του αισθητήρα απόστασης
#define ECHO_PIN     22  // Δεσμευμένο pin του Arduino για ενεργοποίηση του δέκτη του αισθητήρα απόστασης
#define MAX_DISTANCE 200 // Μέγιστη απόσταση ανίχνευσης (σε εκατοστά)

Servo myservo;           //δήλωση του σερβοκινητήρα

int countLM=0,countRM=0; //Ορισμός των αριθμητών για την κάθε ρόδα
double Input=0, Output=0, Setpoint=20;   //Εισαγωγή μεταβλητών του PID για την Αριστερή Ρόδα
double Input2=0, Output2=0,Setpoint2=20; //Εισαγωγή μεταβλητών του PID για την Δεξιά Ρόδα
int temp=0, temp2=0, CPING=0; //Μεταβλητές για τις συναρτήσεις του αισθητήρα απόστασης
int pos = 70;     //Ρύθμιση του αισθητήρα αποστασης ώστε να είναι πάντα γυρισμένος ευθεία
boolean FirstDec=true,Dec=false;            //Μεταβλητές για τις συναρτήσεις του αισθητήρα απόστασης
double Kp = 1.053;   //1.053                //Ρύθμιση των μεταβλητών των PID
double Ki = 10.4;   //10.4,,5.4
double Kd = 0.0;
int TheLastTime=0;                          //Εισαγωγή μετρητή χρόνου
int Noise=0;                          //Εισαγωγή μεταβλητής προγραμματιστικού θορύβου

const int numReadings = 3;                  //Πόσα δείγματα χρησιμοποιούνται από την συνάρτηση εξομάλυνσης

int readings[numReadings];                  // πίνακας δειγμάτων του αποκωδικοποιητή της δεξιάς ρόδας
int readings2[numReadings];                 // πίνακας δειγμάτων του αποκωδικοποιητή της αριστερής ρόδας

int readIndex = 0;                          // Το περιεχόμενο της τρέχων ανάγνωσης
int total = 0;                              // Το άθροισμα των δειγμάτων της δεξιάς ρόδας
int average = 0;                            // Ο μέσος όρος των δειγμάτων της δεξιάς ρόδας
int total2 = 0;                             // Το άθροισμα των δειγμάτων της αριστερής ρόδας
int average2 = 0;                           // Ο μέσος όρος των δειγμάτων της αριστερής ρόδας

PID myPID(&Input, &Output, &Setpoint, Kp, Ki, Kd, DIRECT); //δήλωση των PID
PID my2PID(&Input2, &Output2, &Setpoint2, Kp, Ki, Kd, DIRECT);
```

```
Adafruit_MotorShield AFMS = Adafruit_MotorShield(); //Δημιουργία του
αντικειμένου motor shield με την default διεύθυνση
Adafruit_DCMotor *myMotor = AFMS.getMotor(1);       //Δήλωση ποιών θέσεων Μ
θα χρησιμοποιήσουμε
Adafruit_DCMotor *myOtherMotor = AFMS.getMotor(2);

NewPing sonar(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE); //Δήλωση του αισθητήρα
απόστασης και των μεταβλητών του

void setup()
{
  //Καθαρισμός των πινάκων
  for (int thisReading = 0; thisReading < numReadings; thisReading++) {
    readings[thisReading] = 0;
  }

  for (int thisReading = 0; thisReading < numReadings; thisReading++) {
    readings2[thisReading] = 0;
  }

//============================================================================

  //Ρύθμιση του motorshield
  AFMS.begin();             //Χρησιμοποίησε την προεπιλεγμένη τιμή
συχνότητας (1,6KHz)

  myMotor->setSpeed(0);     //Σταμάτα τους κινητήρες
  myOtherMotor->setSpeed(0);
  myMotor->run(FORWARD);    //Κίνηση των κινητήρων προς τα εμπρός
  myOtherMotor->run(FORWARD);

//============================================================================

  //Ρύθμιση διακοπών
  attachInterrupt(digitalPinToInterrupt(18), docountLM, CHANGE);
//Ενεργοποίηση των συναρτήσεων docountLm και docountRm
  attachInterrupt(digitalPinToInterrupt(19), docountRM, CHANGE);  //σε κάθε
αλλαγή κατάστασης

//============================================================================

  //Ρύθμιση των PID
  myPID.SetMode(AUTOMATIC);
  myPID.SetSampleTime(100);
  myPID.SetOutputLimits(0, 255);

  my2PID.SetMode(AUTOMATIC);
  my2PID.SetSampleTime(100);
  my2PID.SetOutputLimits(0, 255);

//============================================================================

  //Ρύθμιση του σερβοκινητήρα
  myservo.attach(10);//Χρήση του 10ου pin για έλεγχο του σερβοκινητήρα
  myservo.write(pos);

//============================================================================
```

```
//delay(2000);
  TheLastTime=millis(); //Ενεργοποίηση του χρονομετρητή
  Serial.begin(115200); //Άνοιξε σειριακό κανάλι

}

void loop() {


  if(millis()-TheLastTime>=4000)
  {
    Noise=20;
  }

  //Συνάρτησεις αισθητήρα απόστασης
  CPING=sonar.ping_cm();//Μέτρα την απόσταση

  if((CPING<=4)&&(CPING>0))// Ακαριαίο σταμάτημα
  {
  myMotor->setSpeed(0);
  myOtherMotor->setSpeed(0);

  while((CPING<=4)&&(CPING!=0))//Αναμονή για απομάκρυνση εμποδίου
  {
    delay(50);
    CPING=sonar.ping_cm();
  }
  }

  if((CPING<=8)&&(CPING>0))//Ανίχνευση εμποδίου
  {
    if(FirstDec==true)     //Έαν αυτή είναι η πρώτη μείωση σώσε την τρέχουσα
ταχύτητα
    {
      temp=Setpoint;
      temp2=Setpoint2;
    }
    if(Setpoint>=0)    //Αποφυγή υπερχείλισης των Setpoint
    {
    Setpoint--;
    Dec=true;

    }
    if(Setpoint2>=0)
    {
     Setpoint2--;
     Dec=true;
    }

    FirstDec=false;
  }
  if((Dec==true)&&((CPING>9)||(CPING==0)))//Όταν το εμπόδιο βρεθεί σε
ασφαλής απόσταση
  {
    Setpoint=temp;  //επανέφερε τις αρχικές ταχύτητες
    Setpoint2=temp2;
    Dec=false;
    FirstDec=true;
  }

//=============================================================================
```

```
  //Input=countLM;

  //Εξομάληνση Input1
  total = total - readings[readIndex];  //Αφαίρεση των παλαιότηρων τιμών
απο συνολικό άθροισμα
  total2 = total2 - readings2[readIndex];

  readings[readIndex] = countLM;        //  ανανέωσε την παλαιότερη θέση
του πίνακα με την τρέχων τιμή
  total = total + readings[readIndex];  //  Υπολόγισε το νέο υπόλοιπο

  average = total / numReadings;  //   Υπολογισμός μέσου όρου

  Input=average;    //Θέσε ως είσοδο των μέσο όρο των 3 τελευταίων τιμών

//=============================================================================

  countLM=0;  //  μηδένισε τον μετρήτη

  //Input2=countRM;

  //Εξομάληνση Input2
  readings2[readIndex] = countRM;   //ανανέωσε την παλαιότερη θέση του
πίνακα με την τρέχων τιμή
  total2 = total2 + readings2[readIndex]; //  Υπολόγισε το νέο υπόλοιπο
  readIndex = readIndex + 1;        //Πάνε στην επόμενη θέση του πινακα

  if (readIndex >= numReadings) {   //Εάν υπερχειλίσει ο πίνκας ξεκίνα από
την αρχή
    readIndex = 0;
  }
  average2 = total2 / numReadings;  //   Υπολογισμός μέσου όρου

  Input2=average2;    //Θέσε ως είσοδο των μέσο όρο των 3 τελευταίων τιμών

//=============================================================================

  countRM=0;//  μηδένισε τον μετρήτη


   noInterrupts();    //απενεργοποίηση των διακοπών
   myPID.Compute();   //Υπολογισμός Αριστερής εξόδου
   my2PID.Compute();  //Υπολογισμός Δεξιάς εξόδου
   interrupts();      //ενεργοποίηση των διακοπών

  myMotor->setSpeed(int(Output));               //Άλλαξε την ταχύτητα στις
αριστερή ρόδα
  myOtherMotor->setSpeed(int(Output2+Noise));   //Άλλαξε την ταχύτητα στις
δεξιά ρόδα και πρόσθεσε πιθανό θόρυβο
```

```
  Serial.print(int(Setpoint));  //Στείλε όλες τις πληροφορίες των PID
  Serial.print("\t");
  Serial.print(int(Setpoint2));
  Serial.print("\t");
  Serial.print(int(Input));
  Serial.print("\t");
  Serial.print(int(Output));
  Serial.print("\t");
  Serial.print(int(Input2));
  Serial.print("\t");
  Serial.print(int(Output2));
  //Serial.print("\t");
  //Serial.print(CPING);
  Serial.print("\n");

  delay(100); //αναμονή 100ms
  //}
}

void docountLM()  // μέτρημα παλμών στην αριστερή ρόδα
{
  countLM++;  // αύξησε τον μετρητή κατά 1
}
void docountRM()  // μέτρημα παλμών στην δεξιά ρόδα
{
  countRM++;  // αύξησε τον μετρητή κατά 1
}
```

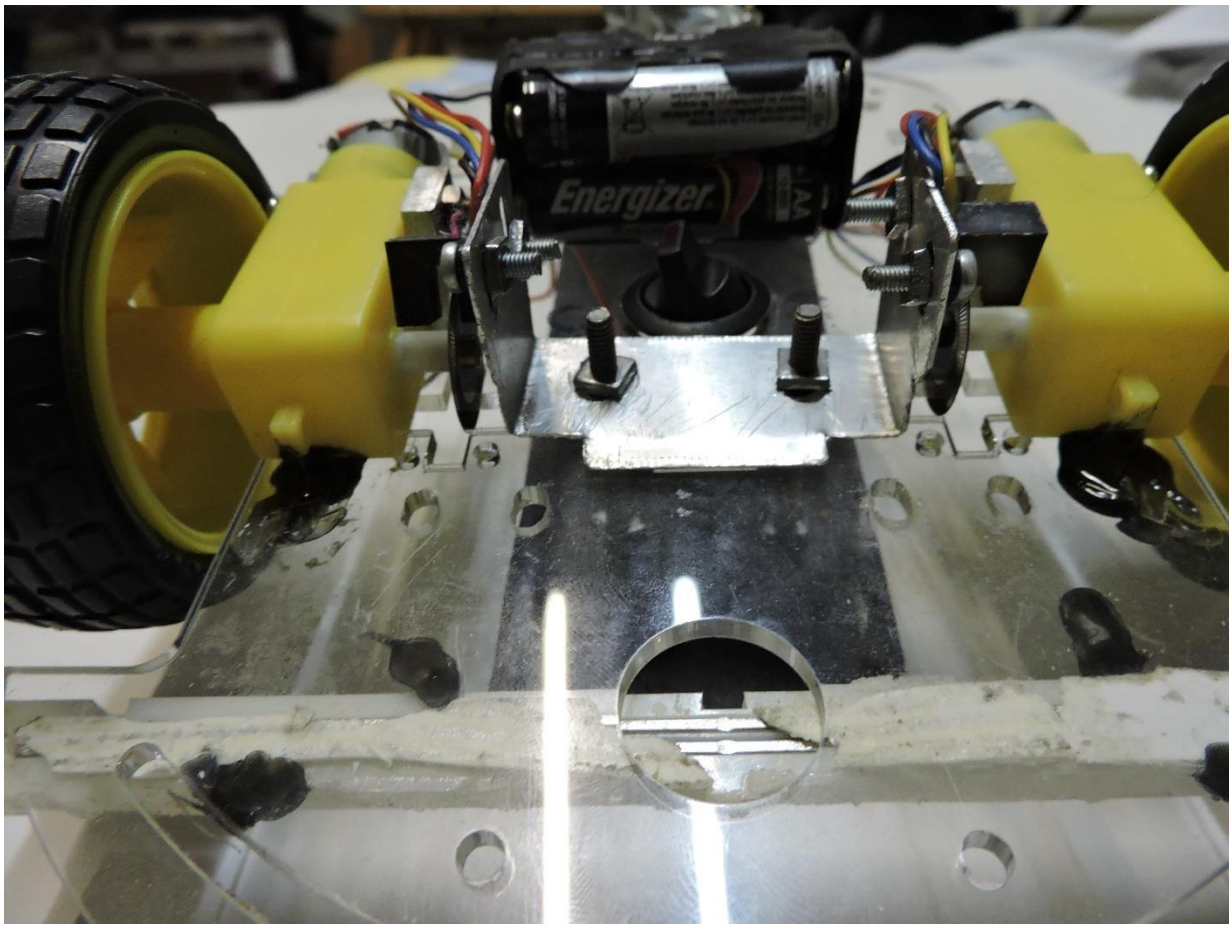
8 Επίλογος

Αφού έχει αποδειχθεί και έχει γίνει πλέον σαφής η λειτουργία και η αποδοτικότητα της εργασίας κλείνοντας, επεξηγούνται όλα τα προβλήματα τα οποία προέκυψαν κατά την υλοποίηση της και αναλύονται διάφορες πιθανές μελλοντικές επεκτάσεις.

8.1 Προβλήματα

8.1.1 Τοποθέτηση αποκωδικοποιητών

Καθώς οι κινητήρες δεν διέθεταν ενσωματωμένους αποκωδικοποιητές προκειμένου να γίνει ο έλεγχος ταχύτητας κρίθηκε απολύτως απαραίτητο να τοποθετηθούν οπτικοί αποκωδικοποιητές στους άξονες των τροχών. Όπως φαίνεται και στην παρακάτω εικόνα προκειμένου να τοποθετηθούν οι αποκωδικοποιητές χρειάστηκε να κατασκευαστεί μια αλουμινένια βάση ώστε αυτοί να είναι παράλληλα τοποθετημένοι στους μεταλλικούς δίσκους.



Εικόνα 8-1

8.1.2 DC κινητήρες

Οι κινητήρες τροφοδοτούνται από 5-6V πράγμα που καθιστά εύκολη την τροφοδοσία τους με απλές μπαταρίες AA. Ωστόσο, δεν ήταν σε θέση να αναπτύξουν ιδιαίτερα μεγάλη ροπή ή στροφές με αποτέλεσμα να εισάγουν ένα βασικό περιορισμό στο εύρος μεταβολής της ταχύτητας στερώντας από την κατασκευή την δυνατότητα να αναπτύξει μεγαλύτερη ταχύτητα ή να αντιμετωπίσει αποτελεσματικότερα μεγαλύτερες κλίσεις στο έδαφος.

8.1.3 Ελαττωματικό εξάρτημα

Ο οπτικός αποκωδικοποιητής ο οποίος τοποθετήθηκε στο δεξιό τροχό του οχήματος αποδείχτηκε αρκετά ελαττωματικός σε σύγκριση με τον αριστερό. Το ένα από τα δύο κανάλια δεν λειτουργούσε ενώ το δεύτερο είχε σφάλμα σχεδόν τριπλάσιο σε σχέση με αυτό του αριστερού αποκωδικοποιητή. Αυτό είχε ως αποτέλεσμα να μην μπορεί να γίνει αποτελεσματική ευθυγράμμιση του οχήματος καθώς ο αντίστοιχος ελεγκτής δεν έπαιρνε σωστές πληροφορίες, το πρόβλημα ωστόσο ξεπεράστηκε με την χρήση του φίλτρου εξομάλυνσης το οποίο παρείχε στον ελεγκτή μια πιο σταθερή τιμή πιο κοντά στην πραγματική.

8.1.4 Αισθητήριο απόστασης

Ένα βασικό μειονέκτημα προέκυψε λόγω του υπερηχητικού αισθητήρα αποστάσεως. Καθώς βασίζεται σε ηχητικά κύματα για να υπολογίσει την απόσταση υπάρχει μεγάλη πιθανότητα μεγάλο μέρος του κύματος να ανακλαστεί προς διαφορετική κατεύθυνση και έτσι ο δέκτης να μην μπορέσει να το εντοπίσει. Πρακτικά αυτό είχε σαν συνέπεια το όχημα να μην ανταποκρίνεται εάν πλησιάσει εμπόδιο με ιδιαίτερα ανώμαλη επιφάνεια.

8.1.5 Κατανάλωση ενέργειας

Αρχικά η πλακέτα Arduino αλλά και οι κινητήρες είχαν από κοινού παροχή ρεύματος από τέσσερεις μπαταρίες τύπου AA. Έτσι σε πολλές περιπτώσεις όπου οι κινητήρες κατανάλωναν μεγάλη ποσότητα ρεύματος αυτό είχε άμεση επίπτωση στην λειτουργία της πλακέτας η οποία δυσλειτουργούσε λόγω της πτώσης τάσης. Το πρόβλημα αυτό παρατηρήθηκε οπτικά καθώς προκλήθηκε θόρυβος στο pin 10 το οποίο ήταν υπεύθυνο για την λειτουργία του σερβοκινητήρα στον οποίο ήταν τοποθετημένο το αισθητήριο απόστασης, ως αποτέλεσμα το αισθητήριο έκανε απότομα στροφές προς τυχαίες κατευθύνσεις.

8.2 Πιθανές επεκτάσεις

8.2.1 Έλεγχος θέσης

Καθώς οι κωδικοποιητές που χρησιμοποιήθηκαν παρέχουν δύο κανάλια μας δίνεται η δυνατότητα να ελέγξουμε την φορά περιστροφής των τροχών. Χάρη σε αυτή τη δυνατότητα μπορούμε να κινήσουμε με ακρίβεια το όχημα ώστε να εκτελέσει ποικίλες λειτουργίες όπως την ισορροπία ενός εκκρεμούς. Επιπλέον η αξιοποίηση και των δύο καναλιών μπορεί να παρέχει διπλάσια ανάλυση σε αντίστοιχες στροφές καθώς τα δύο κανάλια έχουν μια διαφορά φάσης 90 μοιρών.

8.2.2 Line/wall following

Χρησιμοποιώντας έναν αισθητήρα υπέρυθρων υπάρχει η δυνατότητα εντοπισμού της θέσης του οχήματος σε σχέση με μια μαύρη γραμμή σε λευκό επίπεδο. Έτσι είναι δυνατή η πλοήγηση του οχήματος ακολουθώντας μια προκαθορισμένη πορεία καθιστώντας έτσι οποιαδήποτε πιθανά πειράματα ευκολότερα αφού πλέον το όχημα κινείται σε συγκεκριμένο χώρο χωρίς να προσκρούει σε ανεπιθύμητα εμπόδια.

Παρόμοιες δυνατότητες προσφέρονται με την τοποθέτηση αισθητηρίων απόστασης στις πλευρές του οχήματος. Έχοντας πλέον πληροφορίες για τα αντικείμενα αμφότερων των πλευρών του οχήματος είμαστε σε θέση να το καθοδηγήσουμε με ασφάλεια μέσα από στενά περάσματα χωρίς προσκρούσεις.

8.2.3 Οδομετρία

Υπολογίζοντας την περιφέρεια της ρόδας και γνωρίζοντας τον αριθμό των παλμών ανά περιστροφή καθίσταται δυνατή η μέτρηση της διανυόμενης απόστασης. Ο υπολογισμός της περιφέρειας της ρόδας γίνεται εύκολα μετρώντας την διάμετρο της δ=7cm επομένως περιφ=δ*π=7*3,14=9,42cm. Ανά περιστροφή της ρόδας επομένως το όχημα μετακινείται 9,42 εκατοστά δηλαδή σχεδόν 1mm ανά παλμό. Αξιοποιώντας την ίδη υπάρχουσα συνάρτηση για καταμέτρηση παλμών λοιπόν μπορεί να γίνει και καταμέτρηση της διανυόμενης απόστασης.

9 Βιβλιογραφία

- **Wikipedia.com**
- **Mathworks.com**
- **Arduino.cc**
- **parall.ax**
- **adafruit.com**
- **Σύγχρονα συστήματα αυτόματου ελέγχου** – (Richard C.Dorf, Roberth H.Bishop)
- **Mobile robots** – (Jones Flieg Seiger, A.K. Peters)
- **Make an Arduino-controlled robot** – (Margolis, O'Reilly)