ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΚΡΗΤΗΣ
ΣΧΟΛΗ ΕΦΑΡΜΟΣΜΕΝΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ Τ. Ε

Ανάπτυξη και υλοποίηση έντροχου ρομποτικού οχήματος ελεγχόμενο από μικροελεγκτή

υπό

Τραχανατζής Εμμανουήλ

Χανιά, 2015

ΠΕΡΙΛΗΨΗ

Στην παρούσα εργασία διερευνήθηκε ο τρόπος υλοποίησης μιας αυτόνομης ρομποτικής κατασκευής χρησιμοποιώντας εξαρτήματα χαμηλού κόστους που είναι ευρέως διαθέσιμα στο εμπόριο. Αναπτύχθηκε ένα έντροχο ρομποτικό σύστημα διαφορικής οδήγησης βασισμένο στον μικροελεγκτή Arduino και μελετήθηκε ο τρόπος ελέγχου είτε με κώδικα που εκτελείται πάνω στο όχημα είτε σε απομακρυσμένο Η/Υ με χρήση MATLAB. Η εφαρμοσιμότητας της συγκεκριμένης προσέγγισης δοκιμάστηκε σε ένα δομημένο χώρο με εμπόδια.

ΚΕΦΑΛΑΙΟ 1^{Ο}

1.1 Εισαγωγή

Τα ρομποτικά οχήματα μπορούν να εκτελέσουν λειτουργίες σε μη δομημένο και πολλές φορές εχθρικό περιβάλλον. Σημαντικό στοιχείο προκειμένου να μπορέσουν να γίνουν χρήσιμα για τον άνθρωπο είναι η δυνατότητα αυτόνομης λειτουργίας και κατ' επέκταση αυτόνομης πλοήγησης.

Με την εξέλιξη της τεχνολογίας και την μείωση του κόστους σήμερα είναι εφικτό να αναπτυχθούν μικρά ρομποτικά οχήματα ικανά να λειτουργήσουν αυτόνομα με χρήση απλών ηλεκτρονικών διατάξεων.

Στην παρούσα εργασία διερευνήθηκε ο τρόπος υλοποίησης μιας αυτόνομης ρομποτικής κατασκευής χρησιμοποιώντας εξαρτήματα χαμηλού κόστους που είναι ευρέως διαθέσιμα στο εμπόριο. Αναπτύχθηκε ένα έντροχο ρομποτικό σύστημα διαφορικής οδήγησης βασισμένο στον μικροελεγκτή Arduino και μελετήθηκε ο τρόπος ελέγχου είτε με κώδικα που εκτελείται πάνω στο όχημα είτε σε απομακρυσμένο Η/Υ με χρήση MATLAB. Η εφαρμοσιμότητας της συγκεκριμένης προσέγγισης δοκιμάστηκε σε ένα δομημένο χώρο με εμπόδια.

1.2 Στόχοι της εργασίας

Η εργασία έχει εκπαιδευτικό σκοπό και στοχεύει στην ευκολότερη κατανόηση των βασικών εννοιών που σχετίζονται με τη μηχατρονική, τα αυτόνομα ρομποτικά οχήματα καθώς και των προγραμματισμό ενσωματωμένων συστημάτων.

Οι στόχοι που καλύπτει η εργασία είναι:

1) Η κατανόηση του τρόπου λειτουργίας ενός αυτόνομου ρομποτικού οχήματος.

2) Η ανάπτυξη ενός ενσωματωμένου συστήματος βασισμένο στον arduino ικανού να ελέγξει πλήρως ένα όχημα αυτού του τύπου.

3) Η επαφή με τεχνολογίες και διαδικασία ανάπτυξης και κατασκευής ενός ρομποτικού συστήματος.

ΚΕΦΑΛΑΙΟ 2^{Ο}

Αυτόνομα ρομποτικά οχήματα χαμηλού κόστους.

2.1 Εισαγωγή

Τα τελευταία χρόνια, ένα σημαντικό πεδίο δράσης της επιστήμης της Ρομποτικής αφορά στον τομέα της ανάπτυξης και εξέλιξης αυτόνομων οχημάτων επίγειων, εναέριων, πλωτών, υποβρύχιων και διαστημικών. Οι σύγχρονες απαιτήσεις για ευέλικτα, αυτόνομα συστήματα, που θα υποβοηθούν ή θα αντικαθιστούν τον ανθρώπινο παράγοντα σε επικίνδυνες ή μη εφαρμογές, έχουν οδηγήσει στην εξέλιξη οχημάτων ικανών να εκτελούν δύσκολες αποστολές και να συμμετέχουν σε ποικιλία εφαρμογών. Στο συγκεκριμένο κεφάλαιο θα γίνει μια προσπάθεια να αναφερθούν κάποια αυτόνομα ρομποτικά οχήματα που έχουν συγκεκριμένα χαρακτηριστικά ανάλογα με αυτά του οχήματος που θέλουμε να σχεδιάσουμε στη συγκεκριμένη εργασία.

2.2 Αυτόνομο Ρομποτικό Όχημα: Οδυσσέας [1]

Ο Οδυσσέας (Εικόνα 1), είναι ένα ρομποτικό σύστημα, χαμηλού κόστους που μπορεί να χρησιμοποιηθεί ως πλατφόρμα πειραματισμού στην αυτόνομη πλοήγηση σε εξωτερικούς χώρους. Η ανάπτυξη του οχήματος βασίστηκε σε ένα απλό ηλεκτροκίνητο παιδικό αμαξίδιο. Στόχος ήταν η μετατροπή του αμαξιδίου από τηλεκατευθυνόμενο όχημα σε όχημα με δυνατότητα αυτόνομης πλοήγησης, που θα είχε την ικανότητα να ακολουθεί προκαθορισμένες διαδρομές και να αποφεύγει εμπόδια.

Το ρομποτικό όχημα φέρει σύστημα αυτόνομης πλοήγησης το οποίο θα συνδυάζει ένα δέκτη GPS και μια ψηφιακή πυξίδα. Το όχημα επίσης έχει μηχανισμό αποφυγής εμποδίων, προκειμένου το αμαξίδο να ολοκληρώνει την κίνηση του δίχως ανθρώπινη παρέμβαση. Έχει τη δυνατότητα να ανιχνεύει εμπόδια που είναι συμπαγή και ογκώδη, με αρκετά μεγάλες επιφάνειες, και όχι κάτι μικρό, όπως κάποιο κοντάρι ή λεπτό ξύλο. Λόγω της κατασκευής του έχει τη δυνατότητα να κινείται με σχετική άνεση σε χωματόδρομους με ψιλό χαλίκι και αρκετά φαρδείς, με την ύπαρξη, ενδεχομένως, κάποια χαμηλής βλάστησης. Η ενεργειακή αυτονομία του αμαξιδίου φτάνει τα 45 λεπτά. Όλες οι αποφάσεις για το πώς θα ενεργήσει το αμαξίδιο λαμβάνονται από τον Arduino UNO που αποτελεί και την κεντρική υπολογιστική του μονάδα. Για τον εντοπισμό της γεωγραφικής θέσης, επιλέχθηκε ο δέκτης GPS EM-406A.Η πυξίδα που επιλέχθηκε ήταν η HMC6352. Τα αισθητήρια υπερήχων που επιλέχθηκαν προκειμένου να γίνεται ο εντοπισμός των εμποδίων ήταν τα HC-SR04. Χρειάστηκαν επίσης δύο Servo. Το ένα Servo απαιτήθηκε για την περιστροφή του τιμονιού και ήταν το HS-5755 της Hitec, το οποίο φτάνει σε ροπή και τα 25Kg/cm. Το δεύτερο Servo αφορά στην περιστροφή του αισθητηρίου υπερήχων και ήταν το MG995 της TowerPro με μέγιστη ροπή 15Kg/cm. Τέλος, χρησιμοποιήθηκε μια οθόνη υγρών κρυστάλλων (LCD) με δύο γραμμές και 16 στήλες και ένα πληκτρολόγιο με 16 κουμπιά.

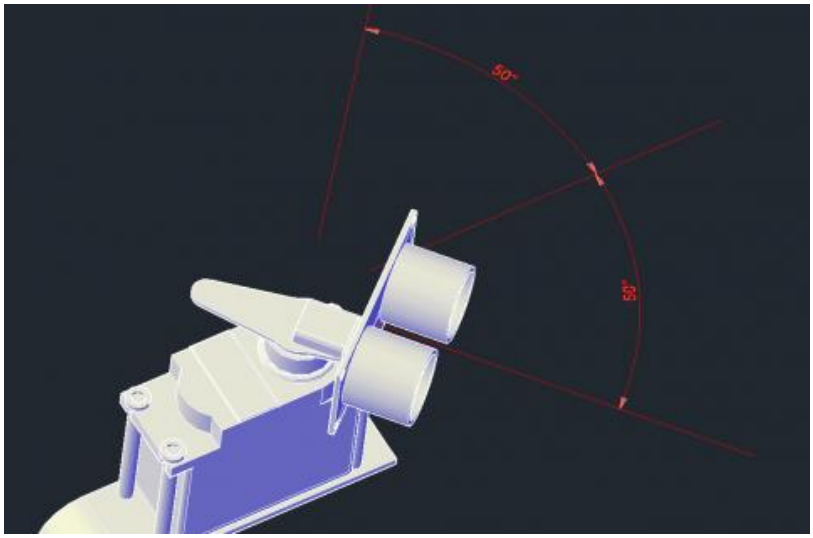
Εικόνα 1: Ρομποτικό όχημα Οδυσσέας [1]

Για την υλοποίηση του συστήματος αποφυγής εμποδίων χρησιμοποιήθηκε ένα φθηνό ψηφιακό Servo, όπου πάνω στον άξονα περιστροφής εφαρμόστηκε το αισθητήριο (Εικόνα 2). Το σέρβο περιστρέφετε στην επιθυμητή γωνία όπου είναι επιθυμητό να εξεταστεί για την ύπαρξη εμποδίων και στην συνέχεια με το αισθητήριο δίνει μια μέτρηση. Το ψηφιακό Servo που επιλέχτηκε ήταν το MG995 της TowerPro.

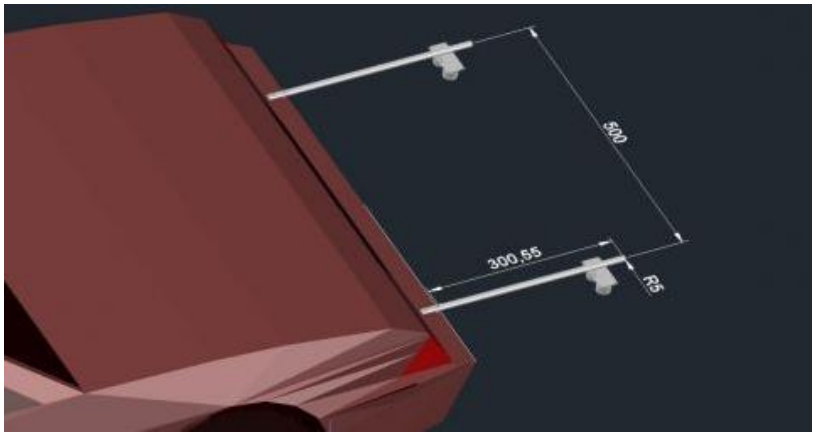
Ο τρόπος με τον οποίο λειτουργεί το σύστημα είναι να γίνεται περιστροφή κάθε φορά του σέρβο στην κατεύθυνση όπου είναι γυρισμένες οι ρόδες του αμαξιδίου και να παίρνονται μετρήσεις σε τακτά χρονικά διαστήματα. Εάν βρεθεί εμπόδιο ελέγχετε η θέση του σημείου στόχου και συγκεκριμένα εάν βρίσκεται αριστερά ή δεξιά αυτής της γραμμής.

Περιστρέφετε το σέρβο σύμφωνα με την κατεύθυνση όπου βρέθηκε το σημείο στόχος και αν ο χώρος είναι ελεύθερος τότε το όχημα κινείται. Αν και στην κατεύθυνση αυτή υπάρχει εμπόδιο τότε γίνεται έλεγχος για την ύπαρξη εμποδίου και αν δεν υπάρχει αντικείμενο το όχημα κινείται. Σε περίπτωση που και πάλι υπάρχει εμπόδιο τότε το όχημα κινείται προς τα πίσω.

Το ρομποτικό όχημα Οδυσσεάς φέρει σύστημα αποφυγής εδαφικών ανωμαλιών. Το σύστημα αυτό, αποτελείτε από δύο αισθητήρια υπερήχων HC-SR04 που βρίσκονται τοποθετημένοι στις δύο μπροστινές γωνίες του προφυλακτήρα του αμαξιδίου. Τα αισθητήρια βρίσκονται τοποθετημένα 10cm μπροστά από το αμαξίδιο. Εκεί συγκρατούνται σε ειδικές βάσεις στηρίξεως. Οι δύο αυτές βάσεις, είναι απλά δύο ατσάλινες βέργες διαμέτρου 3mm και μήκους 20cm και παρουσιάζονται στην εικόνα 3.



Εικόνα 2: Αισθητήριο στερεωμένο σε ψηφιακό servo [1]

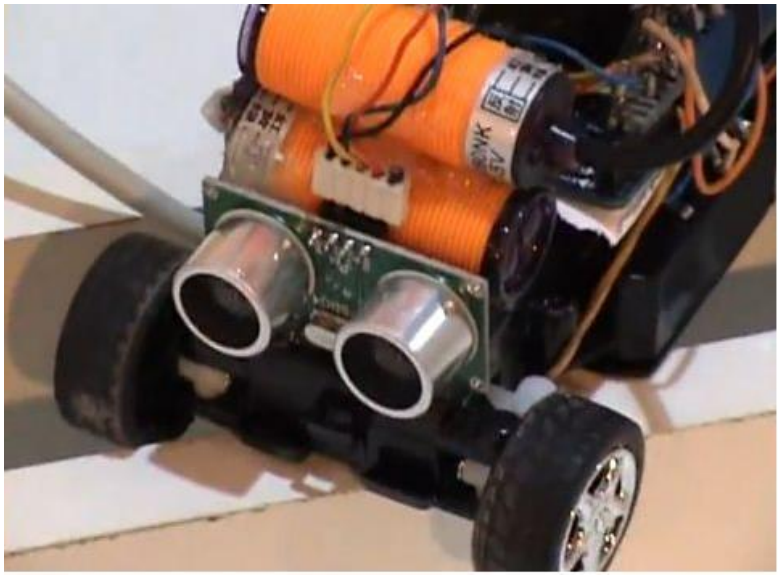


Εικόνα 3: Αισθητήρια στερεωμένα στο προφυλακτήρα του αμαξιδίου [1]

Ο τρόπος με τον οποίο λειτουργεί αυτό το σύστημα είναι αρκετά απλός. Η απόσταση μεταξύ των αισθητηρίων και επίπεδου εδάφους είναι συγκεκριμένη και ισούται με 35cm. Σε περίπτωση που από τα αισθητήρια υπερήχων μετρηθεί απόσταση μεγαλύτερη των 35cm, τότε αυτό σημαίνει ότι υπάρχει λακκούβα την οποία αδυνατεί το αμαξίδιο να περάσει. Το ίδιο ισχύει και στην αντίθετη περίπτωση, όταν δηλαδή, τα αποστασιόμετρα μετρήσουν απόσταση μικρότερη των 28cm. Τότε σημαίνει ότι υπάρχει κάποιο χαμηλό εμπόδιο, όπως ένα κράσπεδο πεζοδρομίου.

2.3 Αυτόνομο Ρομποτικό Όχημα με ικανότητα αναγνώρισης περιοχής [2]

Μια εναλλακτική κατασκευή χαμηλού κόστους παρουσιάζεται στην εικόνα 4. Πρόκειται εάν ρομποτικό όχημα βασισμένο στο σασί ενός παιδικού παιχνιδιού. Το ρομποτικό όχημα έχει ικανότητα ασύρματης επικοινωνίας με ένα σταθμό βάσης μέσω του οποίου μπορεί να αλληλεπιδράσει ο χρήστης με αυτό. Στο όχημα έχουν προσαρμοστεί αισθητήρες που μετρούν την απόσταση καθώς και μια κεντρική μονάδα βασισμένη σε ένα μικροελεγκτή arduino.



Εικόνα 4: Όχημα αναγνώρισης περιοχής [2]

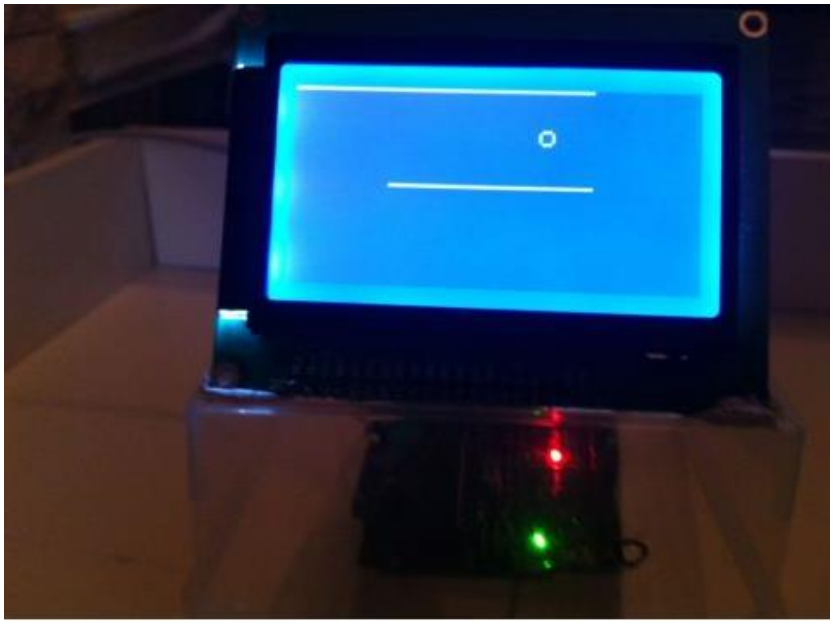
Το όχημα φέρει ένα ειδικό ασύρματο module το οποίο μεταδίδει στη συχνότητα των 2.4GHz τις μετρήσεις απόστασης του οχήματος στο σταθμό βάσης. Ο σταθμός αυτός διαθέτει γραφική οθόνη LCD καθώς και ένα module επικοινωνίας επίσης στα 2.4GHz το οποίο λαμβάνει τις μετρήσεις από το ρομποτικό όχημα.

Στην LCD (εικόνα 5) οθόνη γίνεται η απεικόνιση πληροφοριών για τον χώρο που κινείται το όχημα δίνοντας χρήσιμες πληροφορίες στο χρήστη. Για τον έλεγχο του συστήματος λήψης και απεικόνισης γίνεται χρήση επίσης μιας ψηφιακής πλατφόρμας (arduino). Το arduino είναι μια ψηφιακή πλατφόρμα ανοιχτής αρχιτεκτονικής χαμηλού κόστους η οποία μπορεί να χρησιμοποιηθεί με σχετικά απλό τρόπο από κάθε ενδιαφερόμενο που επιθυμεί να ξεκινήσει την ανάπτυξη ρομποτικών εφαρμογών.

Εικόνα 5: LCD οθόνη απεικόνισης πληροφοριών[2]

Το όχημα αυτό αποτέλεσε τη βάση και για περαιτέρω δοκιμές με την προσθήκη επιπλέον αισθητήρων. Συγκεκριμένα προστέθηκα δύο βοηθητικά IR Sensors που βοηθούσαν ουσιαστικά στην αποφυγή σύγκρουσης του οχήματος και όχι στην ακριβή μέτρηση της απόστασης (από παράπλευρα εμπόδια). Στη συνέχεια προστέθηκε ένας βηματικός κινητήρας και πάνω του ενσωματώθηκε ένας αισθητήριας υπερήχων. Το όχημα είχε τη δυνατότητα να αντιλαμβάνεται τη θέση στην οποία βρισκόταν σύμφωνα με τα δεδομένα που παίρνει από τον υπέρηχο αισθητήρα σε γωνία 180 μοιρών. Η πληροφορία αποτυπωνόταν γραφικά στην οθόνη σε πραγματικό χρόνο.

Με βάση τη παραπάνω πληροφορία το όχημα είχε τη δυνατότητα να χαρτογραφεί το χώρο και να τον χωρίζει σε μικρότερα τετράγωνα μέρη. Σε κάθε βήμα κίνησης που έκανε πραγματοποιούσε έλεγχο των τετραγώνων σε γωνία 180 μοιρών (μπροστά, δεξιά και αριστερά). Συνυπολογίζοντας τη μέτρηση δεξιά και αριστερά με τη θέση του αισθητήρα σε σχέση με το όχημα μπορούσε να προσδιορίσει η θέση του οχήματος. Οι μετρήσεις των αισθητήρων αξιοποιούνται τόσο για την απεικόνιση της περιοχής όσο και για την ασφαλή κίνηση του οχήματος. Με αυτό το τρόπο, το ρομποτικό όχημα γνώριζε ανά πάσα στιγμή σε ποιο βήμα και σε ποια θέση της περιοχής βρίσκεται ενώ ταυτόχρονα έστελνε ασύρματα τα ανάλογα αποτελέσματα στην οθόνη για την απεικόνιση του χάρτη.



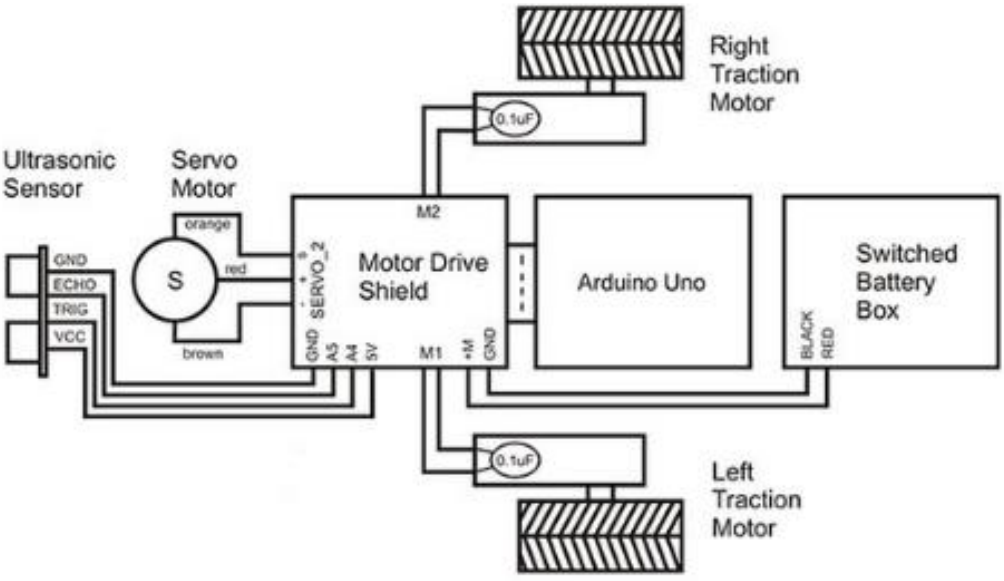
Εικόνα 6: LCD οθόνη για τη γραφική αποτύπωση του χώρου στον οποίο κινείται το ρομπότ [2]

ΚΕΦΑΛΑΙΟ 3^{Ο}

Σχεδιασμός της Ρομποτικής Κατασκευής και έλεγχος ορθής λειτουργίας ηλεκτρονικών εξαρτημάτων

3.1 Εισαγωγή

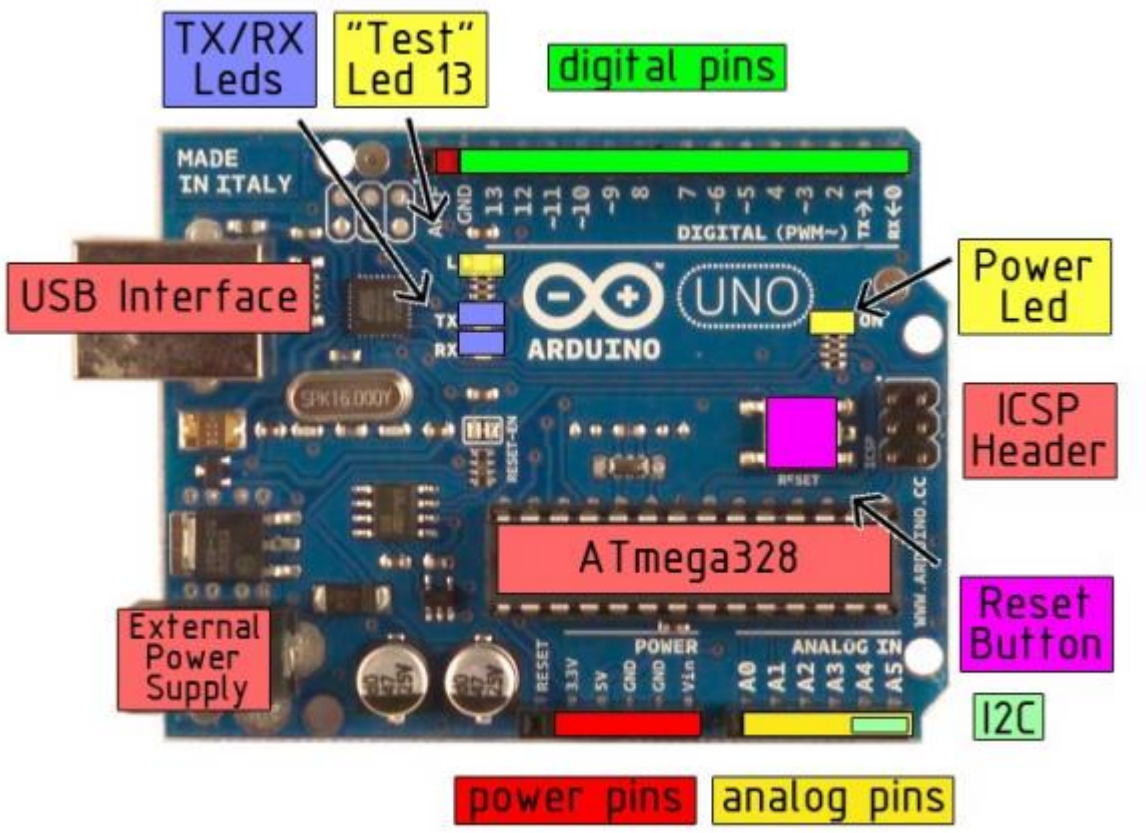
Η βασική ιδέα πάνω στην οποία στηρίχθηκε η ανάπτυξη της ρομποτικής κατασκευής ήταν η ανάπτυξη ενός εντρόχου ρομποτικού οχήματος ανάλογης φιλοσοφίας με αυτά που περιγράφηκαν στο 2^{ο} κεφάλαιο. Επιλέχθηκε ένας σχεδιασμός βασιζόμενος σε ένα όχημα διαφορικής οδήγησης, δηλαδή κίνησης που βασίζεται στην εναλλαγή της ταχύτητας στους κατευθυντήριους τροχούς. Το όχημα έφερε επίσης μια μπίλια που χρησίμευε στη στήριξη του και στην ομαλή εναλλαγή της κίνησης του. Έγινε η επιλογή κατάλληλου αισθητήρα που του επέτρεπε την αναγνώριση εμποδίων και μέσω της τοποθέτησης του σε ένα σερβοκινήτηρα είχε την επιπλέον δυνατότητα να ανιχνεύει σε ένα ευρύ φάσμα (180^{ο}). Η κεντρική μονάδα έλεγχου του οχήματος ήταν ένας μικροελεγκτής που είχε τη δυνατότητα να στέλνει παλμούς και να ελέγχει τους κινητήρες του οχήματος. Οι κινητήρες που επιλέχθηκαν για την κίνηση του ρομπότ, λειτουργούν στα 5V και απαιτούν 150mA για να πραγματοποιήσουν 48 στροφές/λεπτό. Η τροφοδοσία του συστήματος γίνεται μέσω τεσσάρων μπαταριών τύπου 2000mAh AA-size NiMH. Στην εικόνα 7 παρουσιάζεται ένα διάγραμμα των δομικών μερών του ρομποτικού οχήματος.



Εικόνα 7: Διάγραμμα δομικών μερών του ρομποτικού οχήματος[3]

3.2 Μονάδα ελέγχου [4]

Η κεντρική μονάδα ελέγχου του οχήματος είναι ο μικροελεγκτής ATMega328, οποίος είναι ενσωματομένος σε ένα αναπτυξιακό γνωστό Arduino UNO. Το συγκεκριμένο αναπτυξιακό θα αποτελεί και τον "εγκέφαλο" του ρομπότ ή αλλιώς τη μονάδα ελέγχου. "Uno" στα Ιταλικά σημαίνει "ένα" και ονομάστηκε έτσι για να σηματοδοτήσει την έξοδο του Arduino 1.0 στην αγορά. Στη εικόνα 8 παρουσιάζεται το αναπτυξιακό με αναφορά του κάθε εξαρτήματος.



Εικόνα 8: Arduino Uno [5]

Έχει 14 ψηφιακά Pins εισόδου/εξόδου εκ των οποίων τα 6 μπορούν να χρησιμοποιηθούν ως διαμόρφωση εύρους παλμών (PWM). Περιέχει επίσης 6 αναλογικές εισόδους, ένα κρύσταλλο συντονισμού στα 16Mhz, επικοινωνία με υπολογιστή με θύρα USB, καλώδια και ένα κουμπί reset.

Το συγκεκριμένο αναπτυξιακό έχει πλήθος πλεονεκτημάτων και συγκεκριμένα:

- Ο σχεδιασμός του είναι ανοιχτού τύπου, δηλαδή μπορεί να βρει ο καθένας τα σχηματικά και PCB στο διαδίκτυο και να φτιάξει από μόνος του μια ίδια ή παρόμοια πλακέτα εφόσον επιθυμεί να κάνει αλλαγές. Το ότι ο σχεδιασμός του είναι ανοιχτού τύπου, σημαίνει ότι υπάρχουν πολλές ομάδες ανθρώπων ανά το κόσμο, σε διάφορα blog ή site που χρησιμοποιούν τον Arduino. Αυτό έχει ως αποτέλεσμα να μπορούμε να βρούμε εύκολα κάποιον να του απευθύνουμε την απορία μας ή να ανταλλάξουμε απόψεις πάνω σε ένα project.
- Επικοινωνία με υπολογιστή μέσω της θύρας USB. Μέσω αυτής της διασύνδεσης μπορούμε να περάσουμε εύκολα και γρήγορα το πρόγραμμα στο μικροελεγκτή. Επίσης μπορούμε να εκμεταλλευτούμε και τη σειριακή ώστε να στέλνουμε ή να λαμβάνουμε σειριακά τα δεδομένα που θέλουμε.
- Έχει εξαιρετική διαχείριση της ενέργειας και ενσωματωμένο σταθεροποιητή τάσης. Μπορούμε να συνδέσουμε μια εξωτερική πηγή τάσης μέχρι 12V και να το ρυθμίσει στα 5V και 3,3V. Επίσης μπορεί να τροφοδοτείται απ'ευθείας από μια θύρα USB χωρίς καμία εξωτερική πηγή αλλά με αυτό το τρόπο δε μπορούμε να αξιοποιήσουμε ολόκληρη την ισχύ σε φορτία που απαιτούν πολύ ρεύμα και αυτός είναι ο λόγος που γίνεται χρήση μπαταριών ή άλλων εξωτερικών πηγών τάσης.
- Έχει ένα κρύσταλλο συντονισμού στα 16Mhz. Αυτό δε τον κάνει το γρηγορότερο μικροελεγκτή της αγοράς αλλά και πάλι είναι αρκετά γρήγορος για τις περισσότερες εφαρμογές.
- 32KB Flash Memory για να αποθηκεύουμε το πρόγραμμα μας.

Ο Arduino Uno μπορεί να τροφοδοτηθεί μέσω της θύρας USB ή με μια εξωτερική πηγή τάσης. Η εξωτερική πηγή τάσης μπορεί να είναι είτε μέσω AC-to-DC Adapter ή από μια μπαταρία. Η πλακέτα μπορεί να λειτουργήσει από εξωτερική πηγή τάσης από 6 έως 20 volts. Εάν τροφοδοτηθεί με λιγότερα από 7V, τότε και πάλι το pin των 5V θα τροφοδοτεί τη πλακέτα με λιγότερο από 5V με αποτέλεσμα η λειτουργία του κυκλώματος να είναι ασταθής.

Εάν τώρα τροφοδοτήσουμε τη πλακέτα με πάνω από 12V, ο σταθεροποιητής τάσης θα ανεβάσει υψηλή θερμοκρασία και υπάρχει πιθανότητα να προκληθούν βλάβες στο κύκλωμα. Ο κατασκευαστής προτείνει η τροφοδοσία της πλακέτας βρίσκεται σε ένα εύρος από τα 7 έως και τα 12 volts. Συγκεκριμένα τα pin της τροφοδοσίας είναι ως εξής:

- **VIN**. Είναι η τάση εισόδου του Arduino όταν θέλουμε να χρησιμοποιήσουμε μια εξωτερική πηγή τάσης (όπως τα 5V της USB). Μπορούμε να δώσουμε τροφοδοσία στο κύκλωμα μέσω αυτού του pin.
- **5V.** Είναι το σταθεροποιημένο τροφοδοτικό του Arduino το οποίο χρησιμοποιείται για να τροφοδοτεί τον μικροελεγκτή και τα άλλα στοιχεία του κυκλώματος. Η τάση αυτή

έρχεται είτε από το VIN διαμέσω on-board σταθεροποιητή ή από καλώδιο USB ή κάποια άλλη πηγή τάσης.

- **3V3.** Μια τάση 3,3 volt η οποία παράγεται από το on-board σταθεροποιητή. Το μέγιστο ρεύμα κυμένεται στα 50mA.
- **GND.** Αυτό είναι το pin της γείωσης.

Καθένα από τα 14 ψηφιακά pin του Uno μπορούν να χρησιμοποιηθούν σαν εισόδοι ή εξόδοι, χρησιμοποιώντας τις συναρτήσεις **pinMode(), digitalWrite(), digitalRead().** Όλες λειτουργούν στα 5volt. Κάθε pin μπορεί να μας παρέχει ή να τραβάει μέγιστο ρεύμα 40mA και έχει εσωτερική pull-up αντίσταση τιμής 20-50kOhms. Επιπρόσθετα, κάποια pin έχουν ειδικές λειτουργίες όπως:

- **Σειριακή: 0(Rx) και 1(Tx).** Χρησιμοποιούνται για να λαμβάνουν (Rx) και για να στέλνουν (Tx) TTL σειριακά δεδομένα.
- **Εξωτερικά Διακοπτάκια: 2 και 3.** Αυτά τα pin χρησιμοποιούνται για να ενεργοποιήσουν κάποιο interrupt σε κατάσταση low. Χρησιμοποιούμε τη συνάρτηση **attachInterrupt().**
- **PWM: 3,5,6,9,10 και 11.** Παρέχουν ένα 8-bit PWM με τη χρήση της συνάρτησης **analogWrite().**
- **SPI: 10(SS), 11(MOSI), 12(MISO), 13(SCK).** Αυτά τα pin παρέχουν SPI (Serial Peripheral Interface) επικοινωνία.
- **LED: 13** Υπάρχει ένα on-board LED συνδεδεμένο στο ψηφιακό ποδαράκι 13. Όταν το συγκεκριμένο ποδαράκι έχει υψηλό δυναμικό τότε το LED ανάβει, ενώ όταν έχει χαμηλό δυναμικό τότε το LED είναι σβηστό.

Ο Uno έχει πολλούς τρόπους να επικοινωνήσει με ένα υπολογιστή, ένα άλλο Arduino ή άλλους μικροελεγκτές. Ο ATmega328 παρέχει UART TTL (5V) σειριακή επικοινωνία, η οποία είναι διαθέσιμη στα ψηφιακά ποδαράκια 0 (RX) και 1(TX).

Το λογισμικό του Arduino περιέχει ένα σειριακό μόνιτορ το οποίο επιτρέπει να στέλνουμε και να λαμβάνουμε σειριακά δεδομένα απ'τον Arduino. Τα on-board RX και TX LED θα ανάψουν όταν θα στέλνονται δεδομένα από τη USB στον υπολογιστή. Μια SoftwareSerial βιβλιοθήκη επιτρέπει τη σειριακή επικοινωνία σε καθένα από του Uno τα ψηφιακά pin.

3.3 Λογισμικό Προγραμματισμού

Ο Arduino Uno μπορεί να προγραμματιστεί με το software της **Arduino (IDE).** Αυτό το software μπορεί ο οποιοσδήποτε να το κατεβάσει από το site της Arduino. Ο χειρισμός του προγράμματος επίσης είναι εξίσου απλός καθώς έχει από μόνο του πολλά πρωτότυπα παραδείγματα κώδικα, ώστε να εξοικειωθούμε αρχικά και μετέπειτα να είμαστε σε θέση να δουλέψουμε μόνοι μας. Μπορούμε να χρησιμοποιήσουμε κάποιο από αυτά για να δούμε πως δουλεύει αλλά και για να μάθουμε το σετ εντολών του.

Ενδεικτικά παρουσιάζονται τα ακόλουθα προγράμματα προκειμένου να παρουσιαστεί ο τρόπος προγραμματισμού και να γίνει έλεγχος ορθής λειτουργίας του αναπτυξιακού.

```
void setup()

{

    Serial.begin(9600);

}

void loop()

{

    Serial.println("Hello, world!");

    delay(1000);

}
```

Αρχικά γίνεται μεταγλώττιση (compile) για τον έλεγχο τυχών συντακτικών λαθών στο πρόγραμμα και στη συνέχεια γίνεται μεταφορά ώστε να κατέβει το πρόγραμμα μέσω του καλωδίου USB στον μικροελεγκτή. Το παραπάνω πρόγραμμα είναι πολύ απλό, απλά εμφανίζει τη πρόταση “Hello, world!” στο Serial Monitor του IDE. Ο μικροελεγκτής στέλνει τη πρόταση μέσω της σειριακής θύρας και εμείς έχουμε τη δυνατότητα να δούμε αυτά τα αποτελέσματα στο Serial Monitor του IDE.

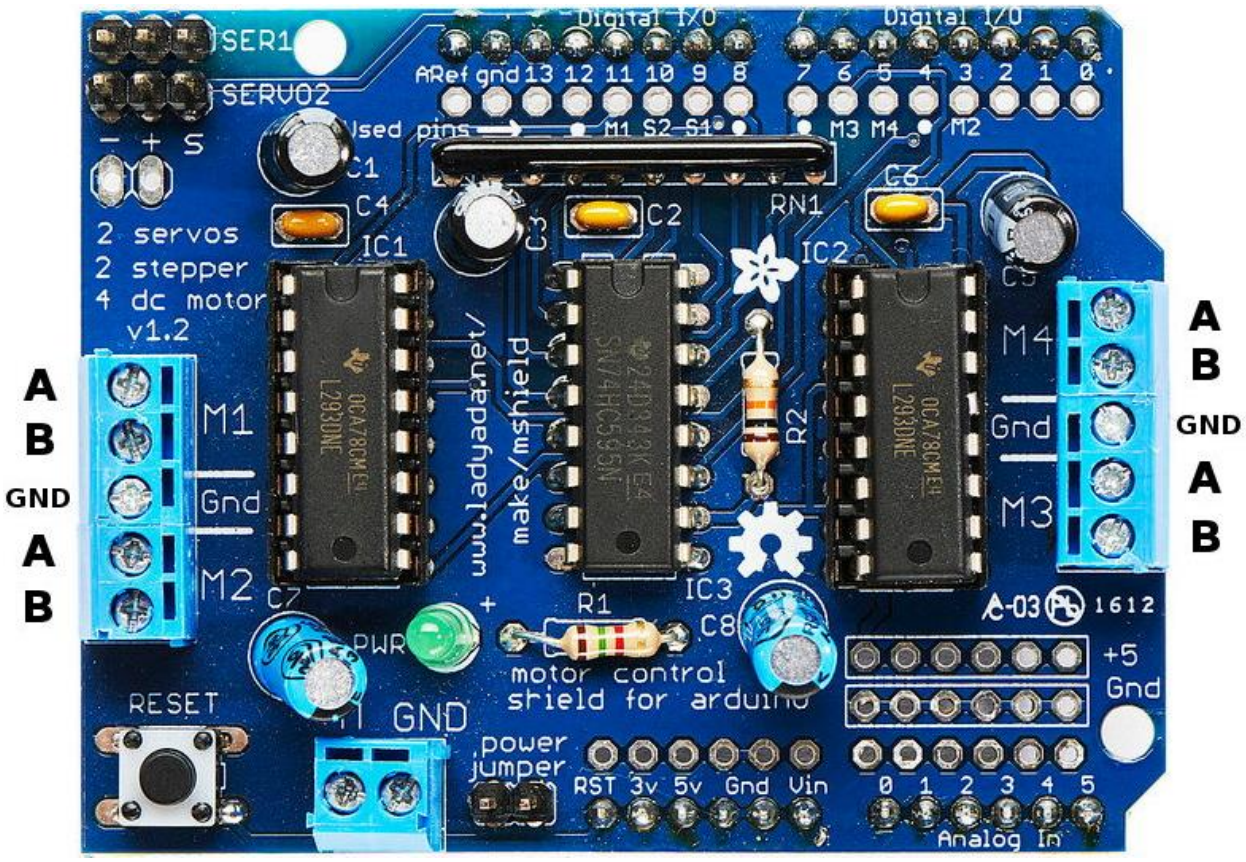
Στη συνέχεια, πραγματοποιήθηκε μια πολύ μικρή μετατροπή στο παραπάνω πρόγραμμα. Αυτή τη φορά τέθηκε ένα pin (ποδαράκι) σαν έξοδος (το pin 13) ώστε να το εκμεταλλευτούμε για κάποιο σκοπό. Σύμφωνα με το παρακάτω πρόγραμμα, θέλουμε να συνδέσουμε ένα LED στο συγκεκριμένο pin και να το κάνουμε να αναβοσβήνει, σύμφωνα με το χρόνο που του έχει τεθεί στο delay (χρονοκαθυστέρηση). Το πρόγραμμα είναι το παρακάτω:

```
void setup() {

    Serial.begin(9600);

    pinMode(13, OUTPUT);

}

void loop() {

    Serial.println("Hello, world!");

    digitalWrite(13, HIGH);

    delay(500);

    digitalWrite(13, LOW);

    delay(500);

}
```

Μετά από αυτές τις απλές δοκιμές που έγιναν, είδαμε ότι ο Arduino ανταποκρινόταν σε αυτά που του λέγαμε να κάνει και επομένως συνεχίστηκε η ίδια διαδικασία και για τα υπόλοιπα εξαρτήματα με σκοπό την εξακρίβωση της ορθής λειτουργίας τους αλλά και τη κατανόηση της λειτουργίας τους.

3.4 Έλεγχος των κινητήρων - Arduino Motor Shield

Ο έλεγχος των κινητήρων του ρομποτικού οχήματος γίνεται μέσω των Arduino Motor Drive Shield (Εικόνα 9). Η συγκεκριμένη πλακέτα είναι υπεύθυνη για την κίνηση των δυο μοτέρ, τα που είναι συνδεμένα στις ρόδες του ρομπότ, ώστε να μπορεί να κινείται. Αυτή η πλακέτα αποτελείται από δύο γέφυρες-Η (H-Bridge), δηλαδή από δυο τσιπ L293D όπου περιέχουν στο εσωτερικό τους 4 mosfet για να δημιουργήσουν τη γέφυρα-Η, και ένα 74HC595 shift register. Τα συγκεκριμένα τσιπ στέλνουν παλμούς PWM στους κινητήρες, και με αυτό το τρόπο αρχίζει το ρομπότ να κινείται, ανάλογα με την ταχύτητα θα ορίσουμε στο PWM σήμα που θα στείλουμε. Η έξοδος των L293D είναι απ'ευθείας συνδεδεμένη στις εξόδους PWM του Arduino. Η πλακέτα Motor Shield είναι δυνατόν να οδηγήσει 2 κινητήρες σέρβο και έχει 8 εξόδους half-bridge για 2 step motor ή 4 full-bridge εξόδους μοτέρ.



Εικόνα 9: Arduino Motor Drive Shield [6]

[7] Οι κινητήρες σέρβο τροφοδοτούνται από τα +5V του Arduino για αυτό έχει ως συνέπεια πιθανή υπερθέρμανση του Arduino. Για να αποφευχθεί αυτό, οι νέες εκδόσεις των Motor Drive Shield δίνουν ξεχωριστή τροφοδοσία +5V.

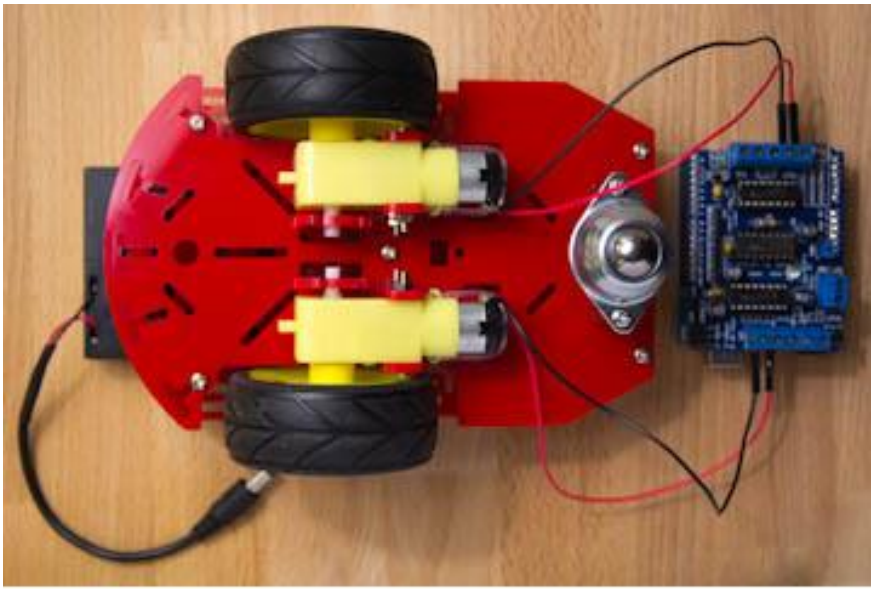
Πολλές φορές όμως, λόγω του χαμηλού συνδυασμού τάσης-ρεύματος του Uno, οι κιντήρες είναι δυνατό να μην αποδίδουν μέγιστη ισχύ, με σκοπό να μη λειτουργεί σωστά το όχημα. Για να αποφευχθεί το γεγονός αυτό, καλό θα ήταν να συνδεθεί εξωτερική πηγή τάσης. Το Arduino Motor Shield περιέχει:

- Δύο διαφορετικές συνδέσεις για 5V
- Δυνατότητα σύνδεσης έως και 4 κινητήρων DC, δύο για κάθε γέφυρα-Η. Εφόσον το Motor Shield περιέχει δύο L293D, τότε μπορούμε να συνδέσουμε δυο μοτέρ σε κάθε ολοκληρωμένο.
- 4 H-Bridge: Το L293D παρέχει 0,6A σε κάθε γέφυρα (1,2A peak) με προστασία από υψηλές θερμοκρασίες.

- Κλέμες PCB: Το Motor Drive Shield περιέχει και ορισμένες κλέμες για διευκόλυνση στην εξωτερική σύνδεση των εξαρτημάτων. Στις κλέμες αυτές μπορούμε να συνδέσουμε τα μοτερ μας (κλέμες M1,M2,M3,M4) και άλλη μια για εξωτερική τροφοδοσία εαν και εφόσον το επιθυμούμε (EXT_PWR).
- **Κουμπί Reset:** Λόγω του ότι το Motor Shield συνδέεται στους κονέκτορες του Arduino, το κουμπί reset του Arduino είναι καλυμμένο. Για αυτό το λόγο λοιπόν, το Motor Shield, έχει κουμπί reset και έτσι όποτε το θέλουμε θα μπορούμε να κάνουμε reset χωρίς να χρειαστεί να βγάλουμε τη πλακέτα απ'τον Arduino.

3.5 Σασί και προσαρμογή κινητήρων

Προκειμένου να δοκιμαστεί η συμβατότητα των παραπάνω δομικών μερών, επιλέχθηκε ένα σασί χαμηλού κόστους ικανού να διαμορφωθεί σε όχημα διαφορικής οδήγησης καθώς και ικανού να φέρει τα προαναφερόμενα δομικά μερή. Πάνω σε αυτό προσαρμόστηκαν οι κινητήρες και στη συνέχεια έγινε η σύνδεση των Motor Drive Shields. Η διάταξη των δοκιμών παρουσιάζεται στην εικόνα 10.



Εικόνα 10: Συνδεσμολογία για τις δοκιμές των κινητήρων

Μετά την επιτυχή σύνδεση των εξαρτημάτων αναπτύχθηκε το παρακάτω πρόγραμμα ελέγχου των κινητήρων. Προκειμένου το πρόγραμμα να είναι λειτουργικό πρέπει να συμπεριληφθεί η βιβλιοθήκη <AFMotor.h> η οποία είναι απαραίτητη για να δουλέψει το Motor Drive Shield.

```
#include <AFMotor.h>

AF_DCMotor Motor1(1);

AF_DCMotor Motor2(3);

void setup()

{

}

void loop()

{

    Motor1.setSpeed(255);

    Motor2.setSpeed(255);

    Motor1.run(FORWARD);

    Motor2.run(FORWARD);

    delay(1000);

    Motor1.run(BACKWARD);

    Motor2.run(BACKWARD);

    delay(1000);

    Motor1.run(FORWARD);

    Motor2.run(BACKWARD);

    delay(1000);

    Motor1.run(BACKWARD);

    Motor2.run(FORWARD);

    delay(1000);

    Motor1.setSpeed(0);

    Motor2.setSpeed(0);
```

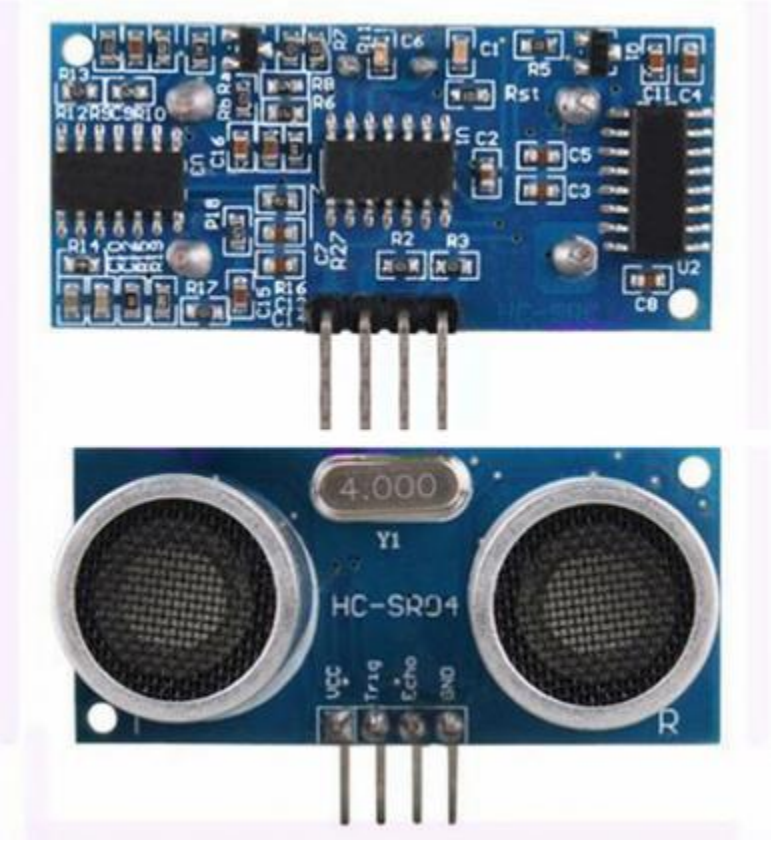
```
    Motor1.run(BRAKE)

    Motor2.run(BRAKE);

    delay(1000);

}
```

Αρχικά, μετά την εντολή **#include <AFMotor.h>** [3], συμπεριλαμβάνεται η βιβλιοθήκη των κινητήρων. Στη συνέχεια, δηλώνουμε ότι οι κινητήρες είναι συνδεδεμένοι στις ηλεκτρονικές κλέμες M1 και M3 του Motor Drive Shield. Οι κινητήρες ελέγχονται από δυο συναρτήσεις. Τη **setSpeed** και τη **run**. Η συνάρτηση **setSpeed** λαμβάνει μια αριθμητική σταθερά εύρους 0 έως 255 και με αυτό το τρόπο δηλώνουμε τη ταχύτητα τους. Όσο μεγαλύτερη τιμή δώσουμε, τόσο μεγαλύτερη ταχύτητα θα έχει ο κινητήρας και αντίστροφα. Η συνάρτηση **run** καθορίζει τη κίνηση του μοτερ, η οποία μπορεί να είναι μπροστά (FORWARD), πίσω (BACKWARD) και σταμάτημα (BRAKE).

Έτσι, η λειτουργία του παραπάνω προγράμματος είναι η εξής: Αρχικά για 1 δευτερόλεπτο και οι δυο ρόδες κινούνται μπροστά, μετά για 1 δευτερόλεπτο και οι δυο ρόδες κινούνται πίσω. Έπειτα για 1 δευτερόλεπτο η μια ρόδα πάει μπροστά και η άλλη πίσω και τέλος για άλλο 1 δευτερόλεπτο και οι δυο ρόδες σταματάνε να κινούνται προς οποιαδήποτε κατεύθυνση.

3.5 Αισθητήρας Μέτρησης Απόστασης HC-SR04

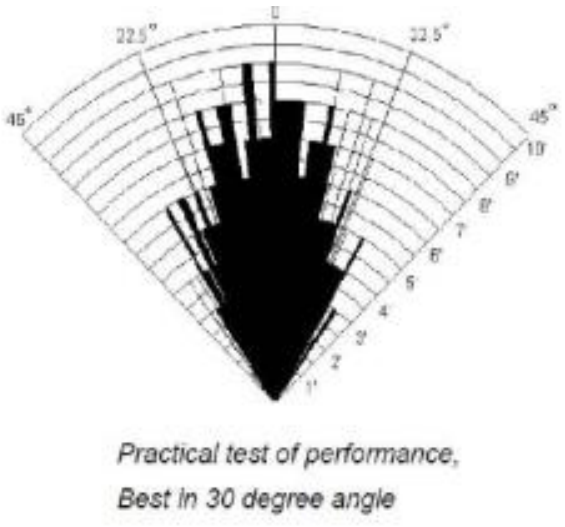
Ο αισθητήρας που χρησιμοποιήθηκε για τον υπολογισμό της απόστασης από τα αντικείμενα που υπάρχουν στο χώρο, ώστε να τα αποφεύγει το ρομπότ, είναι ο HC-SR04. Ο τρόπος με τον οποίο μετράει την απόσταση από ένα αντικείμενο ο συγκεκριμένος αισθητήρας, είναι με τη χρήση ενός sonar (ηχοεντοπιστής). Στέλνει δηλαδή ένα ηχητικό κύμα, το οποίο χτυπάει πάνω στο αντικείμενο και λόγω της ανάκλασης που γίνεται, εντοπίζει τον ήχο ο αισθητήρας και αναλόγως του χρόνου που χρειάζεται να εντοπίσει τον ήχο, ο αισθητήρας υπολογίζει την απόσταση. Στην εικόνα 11 παρουσιάζεται ο αισθητήρας HC-SR04.



Εικόνα 11: Αισθητήρας μέτρησης απόστασης (HC-SR04)[8]

Ο συγκεκριμένος αισθητήρας παρέχει αρκετή ακρίβεια. Το εύρος των τιμών σε εκατοστά το οποίο είναι ικανός να λειτουργήσει αξιόπιστα κυμαίνεται από 2 εκατοστά έως και 400. Η λειτουργία του αισθητήρα δεν επηρεάζεται από το φως του ήλιου ή από μαύρα υλικά. Όπως φαίνεται από τη παραπάνω εικόνα, ο αισθητήρας έχει 4 ποδαράκια. Το Vcc είναι η τάση λειτουργίας του, το Trig είναι η είσοδος του, το Echo η έξοδος και το Gnd η γείωση.

Στη εικόνα 12 παρουσιάζεται δυνατότητα του αισθητήρα να μετράει καλύτερα την απόσταση σε γωνία 30 μοιρών, σύμφωνα με το datasheet του κατασκευαστή.



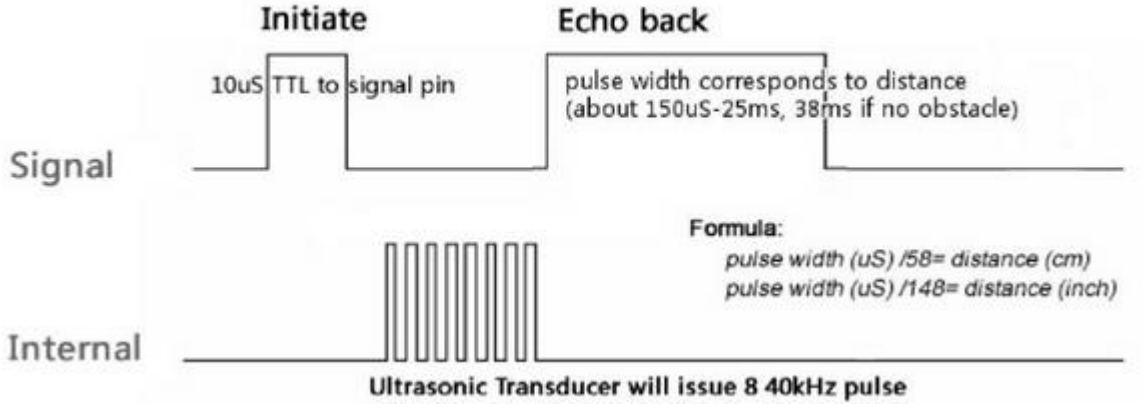
Εικόνα 12: Εύρος ανίχνευσης αντικειμένω από τον αισθητήρα [8]

[8]Τα χαρακτηριστικά του HC-SR04 είναι τα παρακάτω:

- **Τάση τροφοδοσίας:** +5V DC
- **Ρεύμα Λειτουργίας:** 15mA
- **Περιοχή Ανίχνευσης:** 2cm - 400cm / 1" – 13ft
- **Γωνία Μέτρησης:** 30 μοίρες
- **Εύρος παλμού Trigger:** 10uS
- **Διαστάσεις:** 45mm x 20mm x 15mm

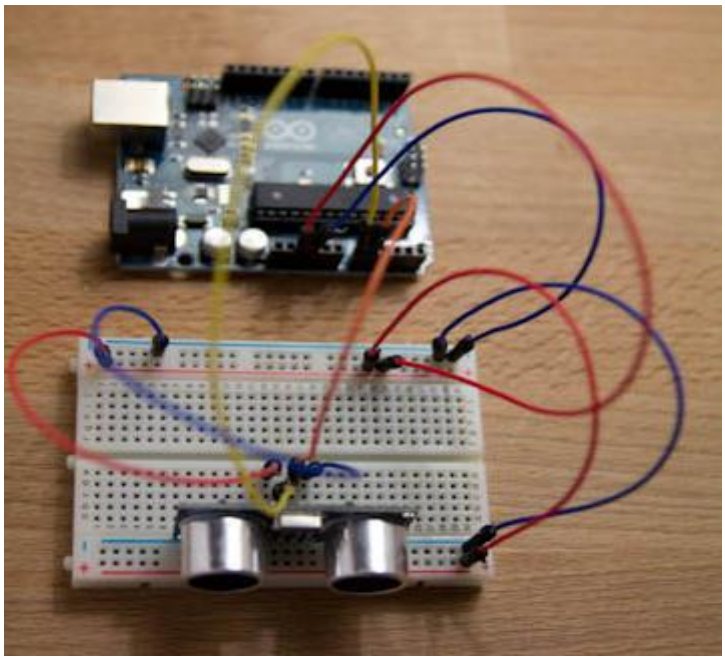
Η λειτουργία του αισθητήρα είναι η εξής**:** Για να ξεκινήσει να υπολογίζει την απόσταση, το ποδαράκι Trig του αισθητήρα θα πρέπει να λάβει ένα παλμό TTL (5V) για τουλάχιστον 10uS. Αυτό θα κάνει τον αισθητήρα να αρχίσει να εκπέμπει ριπές για 8 κύκλους με συχνότητα 40kHz, περιμένοντας αυτές τις ριπές να ανακλαστούν. Όταν ο αισθητήρας λάβει αυτές τις ριπές θα αναγκάσει το ποδαράκι Echo να μπει σε υψηλό δυναμικό (5V) με κάποια καθυστέρηση (σε μορφή πλάτους παλμού) ώστε να υπολογίσει την απόσταση. Ο αισθητήρας υπολογίζει το πλάτος αυτού του παλμού και έτσι υπολογίζει την απόσταση.

Στην εικόνα 13, παρουσιάζεται σε χρονοδιάγραμμα, η λειτουργία του αισθητήρα σύμφωνα με το datasheet του κατασκευαστή.



Εικόνα 13: Διάγραμμα λειτουργίας αισθητήρα [8]

Προκειμένου να δοκιμάσουμε την ορθή λειτουργία του αισθητήρα χρησιμοποιήσαμε το κύκλωμα που παρουσιάζεται στην εικόνα 14. Συνδέσαμε το Vcc στα +5V του Arduino, το GND στο GND, το Trigger στο ποδαράκι 7 του UNO και το Echo στο ποδαράκι 8 του Arduino.



Εικόνα 14: Συνδεσμολογία για τον έλεγχο λειτουργίας αισθητήρα

Για να δοκιμάσουμε την ορθή λειτουργία του αναπτύξαμε το ακόλουθο πρόγραμμα. Για τη σωστή εκτέλεση του προγράμματος είναι αναγκαία η βιβλιοθήκη <Newping.h>.

```
#include <NewPing.h>

#define TRIGGER_PIN  7

#define ECHO_PIN     8

#define MAX_DISTANCE 200

NewPing DistanceSensor(TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);

void setup()

{

    Serial.begin(9600);

}

void loop()

{

    unsigned int cm = DistanceSensor.ping_cm();

    Serial.print("Distance: ");

    Serial.print(cm);

    Serial.println("cm");

    delay(1000);

}
```

Οι μετρήσεις που παίρνει ο αισθητήρας, με το παραπάνω πρόγραμμα παρουσιάζονται στο Serial Monitor του IDE και αυτό γίνεται μέσω της σειριακής όπου λαμβάνει τα δεδομένα από τον.

Μετά από αρκετά πειράματα που έγιναν, κρίθηκε ότι ο συγκεκριμένος αισθητήρας μας παρέχει τη καλύτερη ακρίβεια όσο να αφορά τη μέτρηση της απόστασης και επομένως επιλέχθηκε αυτός. Ενδεικτικές μετρήσεις παρουσιάζονται στη συνέχεια.

```
Distance: 28cm

Distance: 28cm

Distance: 21cm

Distance: 18cm

Distance: 15cm

Distance: 15cm

Distance: 12cm

Distance: 12cm

Distance: 24cm
```

3.6 Σώμα και μηχανολογικά μέρη του ρομποτικού οχήματος

Το σασί, τα μηχανολογικά εξαρτήματα και γενικότερα ο σκελετός του ρομπότ, επιλέχθηκαν με πολλά κριτήρια. Αρχικά θα ήταν ωφέλιμο να μη ζυγίζουν πολλά γραμμάρια καθώς αυτό θα ήταν ένα πρόβλημα για τους δυο κινητήρες εξαιτίας του γεγονότος ότι θα κουβαλούσαν όλο το βάρος του οχήματος και σε συνδυασμό με το βάρος των εξαρτημάτων που θα είναι βιδωμένα στο σασί του, να δυσκολεύονται να δουλέψουν άνετα με αποτέλεσμα το κίνδυνο καταστροφής τους. Επομένως το πρώτο κριτήριο ήταν ο σκελετός να ζυγίζει όσο δυνατόν λιγότερο.

Εξίσου βασικό κριτήριο για την επιλογή ήταν το κόστος καθώς και η ευκολία συναρμολόγησης και προσαρμογής των εξαρτημάτων.

Αρχικά αγοράσαμε τις ρόδες του ρομπότ οι οποίες είναι φτιαγμένες από πλαστικό, και πάνε μαζί με τους κινητήρες που είναι ειδικά σχεδιασμένοι για να χωρούν στη διάμετρο της ρόδας. Οι κινητήρες είναι πλαστικοί και αυτά και λειτουργούν στα 5V, πράγμα που σημαίνει ότι μπορούν να τροφοδοτηθούν και από τον Arduino αλλά δίχως να λειτουργούν σε μέγιστη ισχύ. Οι κινητήρες με τους τροχούς προσαρμοσμένοι σε αυτούς παρουσιάζονται στην εικόνα 15.

Εικόνα 15: Το μοτερ μαζί με τη ρόδα που χρησιμοποιήθηκε

ΚΕΦΑΛΑΙΟ 4Ο

Προγραμματίζοντας το ρομποτικό όχημα μέσω MATLAB

4.1 Εισαγωγή – Τι είναι το MATLAB

Το MATLAB είναι ένα περιβάλλον αριθμητικής υπολογιστικής και μια προγραμματιστική γλώσσα τέταρτης γενιάς. Αποθηκεύει και κάνει τις πράξεις με βάση την άλγεβρα μητρών.

Ειδικότερα, το MATLAB είναι ένα πρόγραμμα υπολογιστών για ανθρώπους που χρησιμοποιούν αριθμητικούς υπολογισμούς, ειδικά στη γραμμική άλγεβρα (πίνακες). Ξεκίνησε ως ένα πρόγραμμα "Εργαστήριου Πινάκων" («MATrixLABoratory") που είχε σκοπό να παρέχει ικανότητα αλληλεπίδρασης με τις βιβλιοθήκες Linpack και Eispack. Από τότε έχει αναπτυχθεί και έχει γίνει ένα ισχυρότατο εργαλείο στην με ευρύτερη χρήση στον προγραμματισμό, στην έρευνα, στην επιστήμη των μηχανικών και στις επικοινωνίες.

Χρησιμοποιείται κατά κύριο λόγο για την επίλυση μαθηματικών προβλημάτων, ωστόσο είναι πολύ ισχυρό και μπορεί να χρησιμοποιηθεί και για προγραμματισμό καθώς περιέχει εντολές από την C++ όπως τη while, την switch και την if. Στο τομέα των γραφικών όσον αφορά τον μαθηματικό κλάδο μπορεί να υλοποιήσει συναρτήσεις πραγματικές, μιγαδικές, πεπελεγμένες συναρτήσεις δυο μεταβλητών και άλλες. Όσον αφορά το στατιστικό κλάδο μπορεί να υλοποιήσει ιστογράμματα, τομεογράμματα, ραβδοδιαγράμματα, εμβαδογράμματα και άλλα.

Υπάρχουν βιβλιοθήκες που περιλαμβάνουν πλήθος αλγορίθμων για διαφορετικές εφαρμογές, δυνατότητες χειρισμού δεδομένων σε μεγάλη κλίμακα και ισχυρά προγραμματιστικά εργαλεία. Το MATLAB δεν είναι σχεδιασμένο για συμβολικούς υπολογισμούς, αλλά αντισταθμίζει αυτή την αδυναμία του επιτρέποντας στο χρήστη να συνδέεται άμεσα με το Maple.

4.2 Σειριακή επικοινωνία Arduino - MATLAB

Με το MATLAB έχουμε τη δυνατότητα να επικοινωνήσουμε σειριακά (μέσω της θύρας USB) με τον Uno. Αυτό μπορεί να γίνει ακολουθώντας τα παρακάτω βήματα:

- Συνδέουμε τον Arduino Uno στον υπολογιστή μας με ένα καλώδιο USB
- Προγραμματίζουμε το μικροελεγκτή χρησιμοποιώντας το Arduino IDE.
- Βρίσκουμε σε ποια σειριακή θύρα είναι συνδεμένος ο Uno και τη δηλώνουμε στο MATLAB
- Δημιουργούμε ένα script στο MATLAB, γράφοντας τις κατάλληλες εντολές και συναρτήσεις. Το αποθηκεύουμε και το καλούμε στη γραμμή εργασίας του MATLAB.

Στη συνέχεια παρατίθεται ένα απλό παράδειγμα επικοινωνίας. Γίνεται αρχικοποίηση της σειριακής θύρας ώστε να είναι έτοιμη να δεχτεί και να λάβει δεδομένα από τον Uno προς το

MATLAB και το αντίστροφα. Αρχικά θα γράψουμε το παρακάτω απλό πρόγραμμα στο Arduino IDE και θα το κατεβάσουμε στο μικροελεγκτή.

```
void setup()

{

    // Θέτουμε BaudRate 9600bps;

       Serial.begin(9600);

    // Τσεκάρουμε τη σειριακή επικοινωνία - Ελέγχουμε αν υπάρχει.

       Serial.println('a'); // Στέλνουμε ένα χαρακτήρα

       char a = 'b';

       while (a !='a')

        {

         //Περιμένουμε ένα χαρακτήρα που θα στείλουμε από το πληκτρολόγιο

         a=Serial.read();

        }

}



void loop()

   {

   }
```

Στη συνέχεια γράφεται το ακόλουθο πρόγραμμα σε περιβάλλον MATLAB, που μέσω σειριακής θύρας, είμαστε σε θέση να στείλουμε και να λάβουμε δεδομένα. Το script που γράφτηκε φαίνεται παρακάτω.

```
function [s,flag] = setupSerial(comPort)
flag=1;
```

```
s=serial(comPort);
set(s,'Databits', 8);
set(s,'Stopbits', 1);
set(s,'BaudRate', 9600);
set(s,'Parity', 'none');
fopen(s);
a='b';
while (a~='a')
    a=fread(s,1,'uchar');
end

if(a=='a')
    disp('serial read');
end

fprintf(s,'%c','a');
mbox=msgbox('Serial Communication setup.');
uiwait(mbox);
fscanf(s,'%u');

end
```

Προκειμένου να ολοκληρωθεί η διαδικασία γράφονται οι ακόλουθες εντολές στη γραμμή εντολής του MATLAB ώστε να ενεργοποιήσουμε το script.

```
>> comport = 'COM3';   //Δηλώνουμε τη σειριακή θύρα όπου είμαστε συνδεμένοι
>> [s,flag] = setupSerial(comport);  // Εδώ καλούμε το script.
Serial read  //Όταν εμφανιστεί αυτό το μήνυμα στη γραμμή εντολών του MATLAB τότε είμαστε έτοιμοι να επικοινωνήσουμε σειριακά.
```

Στη συνέχεια παρουσιάζεται ένα παράδειγμα σειριακής επικοινωνίας. Όπως αναφέρθηκε στην αρχή του κεφαλαίου, το MATLAB έχει τη δυνατότητα να κάνει ιστογράμματα, ραβδοδιαγράμματα και γενικότερα γραφικές παραστάσεις. Αυτό μπορούμε να το εκμεταλλευτούμε με πολλούς τρόπους όπως για παράδειγμα να διαβάζουμε ορισμένες τιμές (π.χ. από αισθητήρα μέτρησης απόστασης) και να τις βλέπουμε σε μια γραφική παράσταση βγάζοντας πολύτιμα συμπεράσματα. Παρακάτω φαίνεται χαρακτηριστικό παράδειγμα όπου

στέλνονται διάφορες τιμές σειριακά από τον Uno τις οποίες διαβάζουμε από το MATLAB και τις αναπαριστούμε γραφικά. Το πρώτο πρόγραμμα γράφτηκε στο Arduino IDE και αυτό που κάνει είναι να αυξάνει αριθμητικά μια μεταβλητή i.

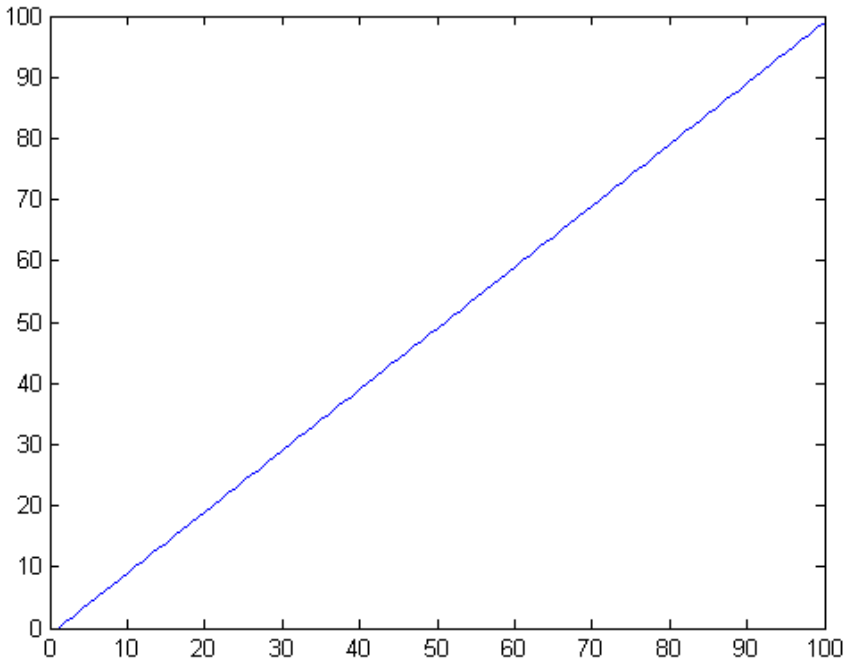
```
int i=0;
void setup()
   {
        Serial.begin(9600); //Θέτουμε BaudRate 9600.
   }

void loop()
  {
       Serial.println(i);
       i++; //Αυξάνεται συνεχώς η τιμή του i.
  }
```

Το πρόγραμμα που γράφτηκε στο MATLAB ώστε να διαβάζονται οι τιμές του i και να αναπαριστώνται σε γραφική παράσταση φαίνεται παρακάτω.

```
arduino = serial('COM4','BaudRate',9600); //Θέτουμε σε μια μεταβλητή τη σειριακή θύρα COM4.
fopen(arduino)          //Ανοίγουμε τη μεταβλητή arduino και συνεπώς και τη σειριακή επικοινωνία.
x=linspace(1,100);
for i=1:length(x)              //Φτιάχνουμε μια επανάληψη όπου θα διαβάζουμε τις τιμές από τον Uno.
       y(i) = fscanf(arduino,'%d');
end
fclose(arduino)                // Κλείνουμε τη σειριακή επικοινωνία.
disp('making plot…')  //Εμφανίζουμε ένα μήνυμα "making plot…" για να περιμένουμε μέχρι να εμφανιστεί το γράφημα.
plot (x,y);                    //Εμφανίζουμε το γράφημα.
```

Αυτό που θα εμφανιστεί στην οθόνη μας όταν τελειώσουμε τη παραπάνω διαδικασία είναι το γράφημα των τιμών του i. Στον οριζόντιο άξονα θα είναι οι τιμές ως το 100 όπου ορίσαμε με την

εντολή x=linspace(1,100) στο Matlab. Ο κάθετος άξονας θα έχει τις τιμές του i όπου αυξάνεται συνεχώς. Στην εικόνα 16 βλέπουμε τη γραφική παράσταση όπως παρουσιάζεται στο MATLAB. Είναι μια ευθεία όπου ξεκινάει από την αρχή των αξόνων.



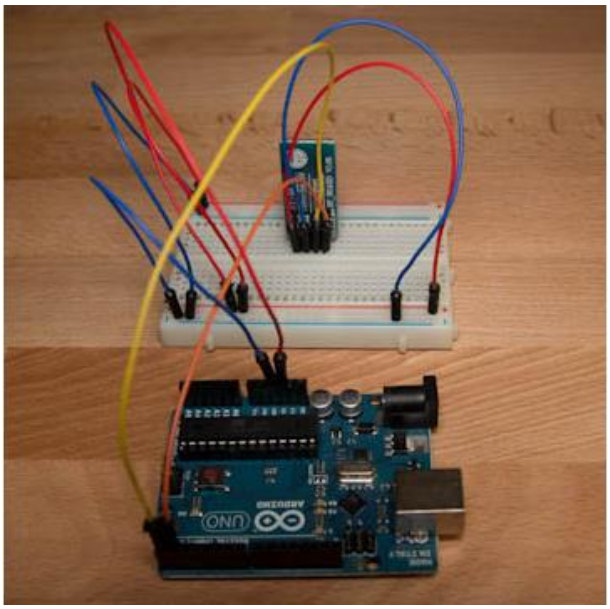
Εικόνα 16: Γραφική παράσταση στο MATLAB.

4.3 Επικοινωνία Arduino Uno-MATLAB με Bluetooth

Με το Bluetooth Slave έχουμε τη δυνατότητα να ελέγξουμε τη κίνηση του ρομποτικού οχήματος είτε από υπολογιστή είτε από Smart Phone μέσω κάποιου ειδικού application και παράλληλα έχουμε τη δυνατότητα να διαβάζουμε τις διάφορες τιμές των μοτερ ή των αισθητηρίων από το MATLAB. Έτσι λοιπόν για τον έλεγχο ορθής λειτουργίας του Bluetooth σαν εξάρτημα, πραγματοποιήθηκε η συνδεσμολογία που παρουσιάζεται στην εικόνα 17.

Το Bluetooth σαν εξάρτημα έχει 4 ποδαράκια. Το **Vcc** το συνδέθηκε στα 5V του Arduino, το **GND** στο GND, το **RXD** στο TX και το **TXD** στο RX του Arduino.

Στη συνέχεια γράψαμε το κώδικα στο Software του Arduino. Ο κώδικας που γράφτηκε φαίνεται παρακάτω.



Εικόνα 17: Συνδεσμολογία για τον έλεγχο με Bluetooth

```
void setup()
{
    Serial.begin(9600); // Θέτουμε BaudRate 9600
    pinMode(13, OUTPUT); // Θέτουμε το ποδαράκι 13 σαν έξοδο.
}
void loop()
{
  while (Serial.available())
    {
      char inChar = (char)Serial.read();
      switch(inChar)
        {
          case '1'                  //Όταν στέλνουμε την αριθμητική τιμή ‘1’.
              digitalWrite(13, HIGH);          // Το LED που είναι συνδεμένο στο ποδαράκι 13, ανάβει.
               break;
        case '0'                          // Όταν στέλνουμε την αριθμητική τιμή ‘0’.
        digitalWrite(13, LOW);         //Το LED που είναι συνδεμένο στο ποδαράκι 13, σβήνει.
      break;
```

```
        }
Serial.println(inChar);
        }
}
```

Ο παραπάνω κώδικας αφού μεταγλωττίστηκε, φορτώθηκε στον μικροελεγκτή μέσω της επιλογής Tools->Serial Port. Το παραπάνω πρόγραμμα ενεργοποιεί την σειριακή θύρα και θέτει το BaudRate σε 9600. Όποτε πατάμε τον αριθμό ‘1’ στο πληκτρολόγιο του υπολογιστή μας, τότε ανάβει ένα LED που έχουμε συνδέσει στο ποδαράκι 13 και ως προς γείωση. Αντίστοιχα, όποτε πατάμε τον αριθμό ‘0’ τότε το LED σβήνει.

Στη συνέχεια, το κομμάτι το οποίο εξετάστηκε είναι, εάν είναι εφικτό το LED να ανταποκριθεί και να κάνει την ίδια δουλειά μέσω του MATLAB. Οι εντολές που γράφτηκαν στο MATLAB φαίνονται παρακάτω.

```
s=serial('COM3');     // Δηλώνουμε στη μεταβλητή 's' τη σειριακή θύρα COM3.
fopen(s);             // Ενεργοποιούμε τη σειριακή θύρα.
fwrite(s,'1');        // Στέλνουμε τη τιμή '1'.
fwrite(s,'0');        // Στέλνουμε τη τιμή '0'.
fclose(s);            //Απενεργοποιούμε τη σειριακή επικοινωνία.
```

Με τη πρώτη εντολή ορίζουμε ότι το γράμμα s είναι μια σταθερά που θα ανοίγει τη σειριακή θύρα ‘COM3’. Με την επόμενη εντολή ενεργοποιούμε τη σειριακή θύρα. Με τις επόμενες δύο απλά στέλνουμε στη σειριακή τους αριθμούς ‘1’ και ‘0’. Αυτό που παρατηρούμε είναι ότι πράγματι το LED αρχίζει να ανταποκρίνεται όπως πρέπει. Ανάβει θέτοντας ‘1’ και σβήνει στο ‘0’.

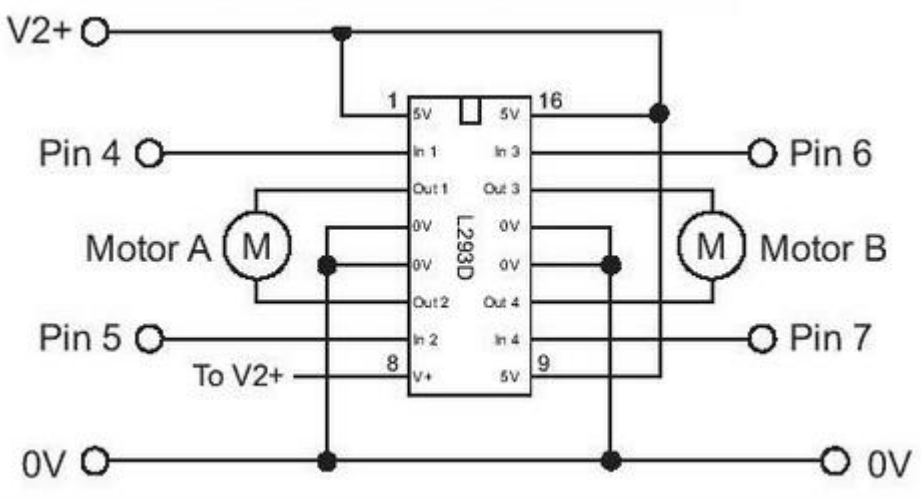
4.4 Έλεγχος του ρομποτικού οχήματος μέσω MATLAB

Ο έλεγχος του ρομπότ από το Matlab μέσω του Bluetooth Slave δεν είναι μια απλή διαδικασία. Απαιτεί πολύ χρόνο και γνώση καθώς πρέπει ανά πάσα στιγμή να είμαστε σε θέση να ελέγξουμε οποιοδήποτε λειτουργικό κομμάτι πάνω στο ρομπότ (μοτερ,αισθητήρια,μονάδα ελέγχου κλπ.) και να επεξεργαστούμε τα δεδομένα του ώστε να βγάλουμε συμπεράσματα.

Προκειμένου να επιτευχθεί ο έλεγχος έγιναν αρκετές διαφοροποιήσεις στο στήσιμο του ρομπότ καθώς με τον μέχρι τότε υπάρχων εξοπλισμό όπου χρησιμοποιήθηκε για την υλοποίηση του ρομποτ δεν ήταν εφικτό μπορούμε να πραγματοποιηθεί έλεγχος μέσω Bluetooth. Αυτό οφειλόταν στο γεγονός ότι το Motor Drive Shield δέσμεύε τα ποδαράκια TX και RX από τον Uno με αποτέλεσμα να μη μπορούμε να συνδέσουμε το Bluetooth στη μονάδα ελέγχου.

Έτσι θα έπρεπε να αφαιρέσουμε το Motor Drive Shield όπου είναι υπεύθυνο για τη κίνηση του κιντητήρα και να βρεθεί άλλο εξάρτημα το οποίο να κάνει την ίδια δουλειά. Το εξάρτημα το οποίο επιλέχθηκε είναι το τσιπάκι L293D το οποίο έχει εσωτερικά του μια γέφυρα-Η. Αυτό αυτομάτως μας εξυπηρετούσε καθώς μπορούσαμε να ελέγξουμε τους κινητήρες. Στις εικόνες 18 και 19 παρουσιάζεται το τσιπ αλλά και πως πρέπει να συνδεθεί το κάθε ποδαράκι του ώστε να μας παρέχει κίνηση στους κινητήρες.

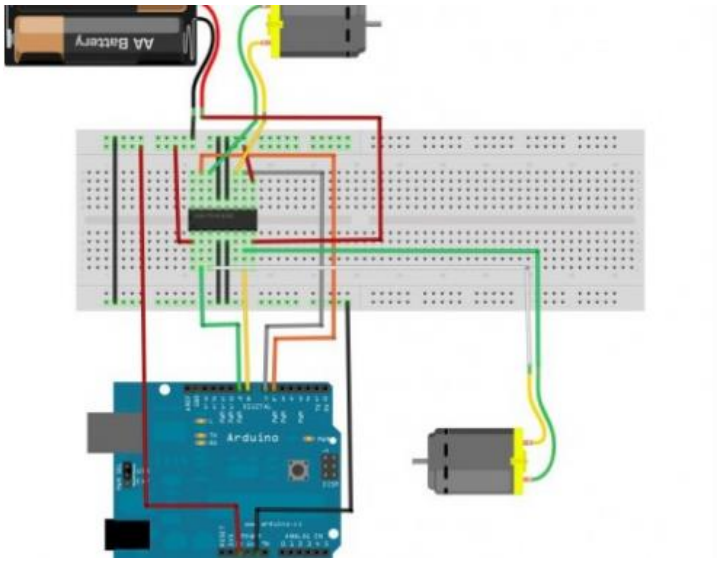
Εικόνα 18: Το L293D [9]



Εικόνα 19: Πως πρέπει να συνδεθεί το L293D

Στο V2+ πάνε τα +5V από τη μονάδα ελέγχου (Arduino Uno). Τα Pin 4, Pin 5, Pin 6, Pin 7 είναι τα ποδαράκια του Arduino. Στο Motor A και Motor B συνδέονται οι κινητήρες και στα 0V η γείωση. Στο ποδαράκι 8, μπαίνει ξεχωριστή τροφοδοσία. Είναι το ίσως πιο σημαντικό κομμάτι το οποίο πρέπει να προσέξουμε γιατί από αυτό το ποδαράκι τροφοδοτούνται τα μοτέρ μας και από αυτό ουσιαστικά εξαρτάται η κίνηση τους. Θα μπορούσαμε να συνδέσουμε εκεί μια πηγή τάσης ή μια μπαταρία. Η ιδανικότερη τάση που πρέπει να βάλουμε σε αυτό το ποδαράκι, σύμφωνα με το κατασκευαστή είναι +12V.

Η συνδεσμολογία που πραγματοποιήθηκε για τον έλεγχο των κινητήρων παρουσιάζεται στην εικόνα 20.



Εικόνα 20: Συνδεσμολογία για τον έλεγχο των μοτέρ

4.5 Έλεγχος κίνησης κινητήρα μέσω MATLAB

Για να ελέγξουμε τη κίνηση των κινητήρων από το MATLAB θα πρέπει αρχικά να κατεβάσουμε τα απαραίτητα στοιχεία και να τα εγκαταστήσουμε στους φακέλους του MATLAB ώστε να γίνει συμβατό με το περιβάλλον του IDE. Αφότου το κάνουμε αυτό τότε θα είμαστε σε θέση να προγραμματίσουμε και να στείλουμε εντολές σειριακά από το MATLAB.Οι εντολές που πρέπει να γράψουμε στο Matlab είναι οι εξής:

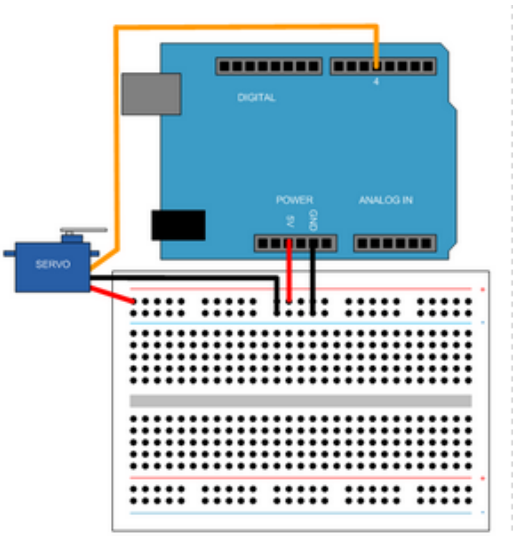
```
a=arduino('COM5');          //Δίνουμε όνομα στη μεταβλητή a.
av=a.analogRead(5);         //Από το ποδαράκι 5 θα διαβάζουμε δεδομένα.
a.pinMode(13,'OUTPUT');     //Θέτουμε σαν έξοδο το ποδαράκι 13.
a.digitalWrite(13,'1');             //Θέτουμε λογικό 1 στη ποδαράκι 13.
```

Οι παραπάνω εντολές είναι παραδείγματα σύνταξης εντολών στο Matlab. Για τον έλεγχο των μοτέρ πρέπει να γραφτούν οι εξής εντολές.

```
a.motorSpeed(4,200); //Δίνουμε ταχύτητα 200/256 στο ποδαράκι 4 όπου θα έχουμε συνδεμένο το μοτέρ μας.
a.motorRun(4,'forward');    //Θέτουμε κίνηση προς τα εμπρός στο μοτέρ.
a.motorRun(4,'backward');   //Θέτουμε κίνηση προς τα πίσω στο μοτέρ.
```

4.6 Έλεγχος κίνησης σέρβο μέσω MATLAB

Το servo που χρησιμοποιήθηκε είναι το μοντέλο SG-90 της TowerPro. Έχει τρία καλώδια εκ των οποίων το κόκκινο συνδέεται στα +5V του Uno, το μαύρο στη γείωση και το πορτοκαλί στο ψηφιακό ποδαράκι 4 του Uno (μπορούμε να το συνδέσουμε όπου μας βολεύει). Στην εικόνα 21 παρουσιάζεται η συνδεσμολογία.



Εικόνα 21: Συνδεσμολογία για τον έλεγχο Servo

Οι εντολές που χρησιμοποιήθηκαν για την κίνηση του Σέρβο είναι οι ακόλουθες.

```
a = arduino('COM3', 'UNO', 'Libraries', 'Servo');
s = servo(a , 4);
s = servo (a , 4 , 'MinPulseDuration', 700*10^-6, 'MaxPulseDuration', 2300*10^-6);
```

Στη συνέχεια παρουσιάζονται οι εντολές που γράφτηκαν για τη κίνηση του σέρβο έχοντας τη δυνατότητα να διαβάζουμε οποιαδήποτε χρονική στιγμή σε ποια θέση βρίσκεται.

```
for angle = 0:0.2:1      //Φτιάχνουμε μια επανάληψη απ'το 0 ως το 1 με βήμα 0.2.
        writePosition(s , angle);          //Γράφουμε τη τρέχουσα θέση του servo στη μεταβλητή s.
current_pos = readPosition(s);          //Διαβάζουμε τη μεταβλητή s και την αποθηκεύουμε στη μεταβλητή current_pos.
        current_pos = current_pos*180;       //Μετατρέπουμε τη γωνία σε μοίρες.
        fprintf('Current motor position is %d degrees \n', current_pos);
        pause(2);
        end
```

Το αποτέλεσμα που εμφανίζεται στην οθόνη είναι η τρέχουσα θέση του servo.

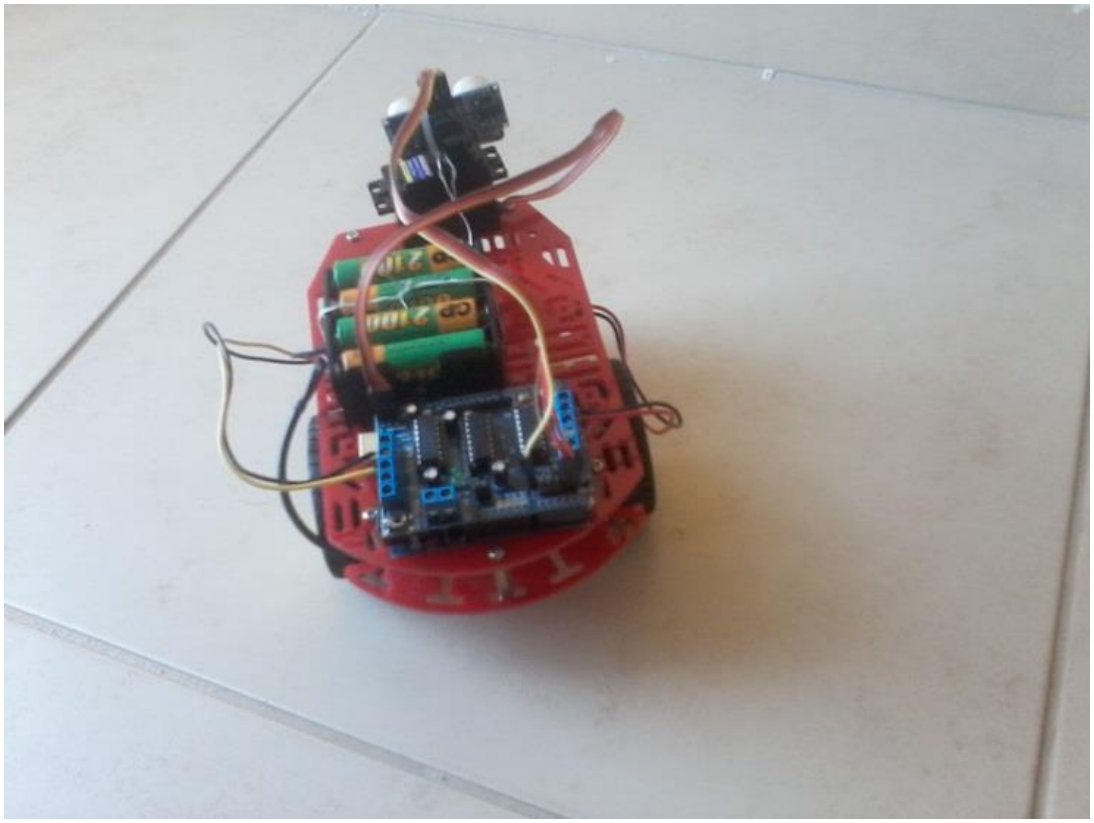
```
Current motor position is 0 degrees
Current motor position is 36 degrees
Current motor position is 72 degrees
Current motor position is 108 degrees
Current motor position is 144 degrees
Current motor position is 180 degrees
```

ΚΕΦΑΛΑΙΟ 5^{Ο}

Υλοποίηση του ρομποτικού οχήματος

5.1 Εισαγωγή

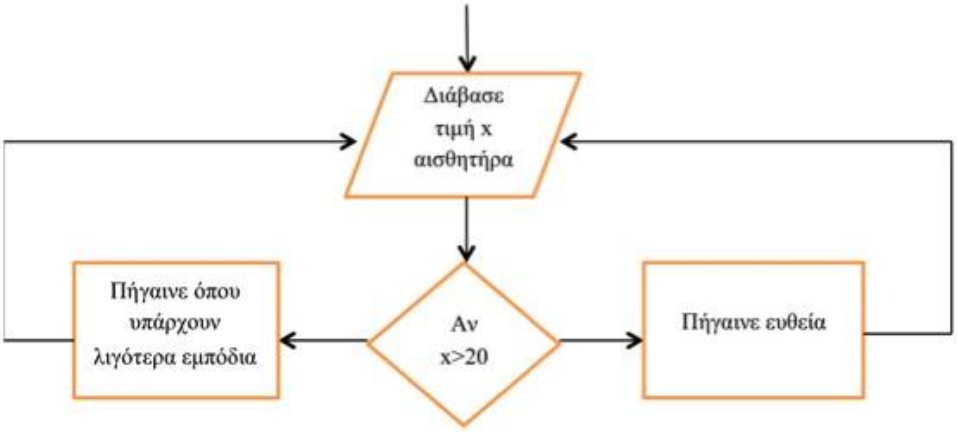
Αφού μελετήσαμε τα επιμέρους εξαρτήματα που θα απαρτίσουν το ρομποτικό όχημα προχωρήσαμε στην ολοκλήρωση της κατασκευής. Η τελική της μορφή παρουσιάζεται στην εικόνα 22. Στόχος ήταν το ρομποτικό όχημα με τη προσθήκη του κατάλληλου κώδικα να μπορεί να κινείται αυτόνομα σε άγνωστο χώρο .



Εικόνα 22: Η τελική μορφή του ρομποτικού οχήματος

5.2 Πρόγραμμα αποφυγής εμποδίων

Σχεδιάστηκε και υλοποιήθηκε ένα απλό πρόγραμμα που εξασφαλίζει την αυτόνομη λειτουργία του οχήματος. Το όχημα παίρνει ενδείξεις από το περιβάλλον και είτε κινείται σε ευθεία, είτε εάν δει εμπόδιο το αποφεύγει και στη συνέχεια κινείται σε ευθεία. Η λειτουργία του προγράμματος παρουσιάζεται συνοπτικά στο ακόλουθο δομικό διάγραμμα.



Εικόνα 23: Λογικό διάγραμμα ελέγχου

Το πρόγραμμα που αναπτύχθηκε είναι το ακόλουθο:

```
#include <AFMotor.h>
#include <Servo.h>
#include <NewPing.h>

#define TRIG_PIN A4          // Το Trigger pin του αισθητήρα το δηλώνουμε στο pin A4 του Motor Drive Shield.
#define ECHO_PIN A5          // Το Echo pin του αισθητήρα το δηλώνουμε στο pin A5 του Motor Drive Shield.
#define MAX_DISTANCE 200     // Θέτουμε τη μέγιστη μετρούμενη απόσταση του αισθητήρα στα 200cm.
#define MAX_SPEED 180        // Θέτουμε μέγιστη ταχύτητα των μοτέρω 180/256. Δηλαδή περίπου το 70% της μέγιστης ισχύος τους.
#define MAX_SPEED_OFFSET 20  //Εδώ θέτουμε τη μέγιστη διαφορά ταχύτητας μεταξύ των δυο μοτέρ.
#define COLL_DIST 10         // Θέτουμε την απόσταση κατά την οποία το ρομπότ αλλάζει κατεύθυνση στα 10cm.
#define TURN_DIST COLL_DIST+10  // Θέτουμε την απόσταση κατά την οποία το ρομπότ στρίβει από το αντικείμενο στα 20cm.
NewPing sonar(TRIG_PIN, ECHO_PIN, MAX_DISTANCE);
//Συμπεριλαμβάνουμε τη βιβλιοθήκη του αισθητήρα για να χρησιμοποιήσει τα σωστά pin ώστε να υπολογίζει την απόσταση.
```

AF_DCMotor motor1(1, MOTOR12_1KHZ); //Δημιουργούμε το motor1 χρησιμοποιώντας την έξοδο M1 στο Motor Drive Shield και στέλνοντας παράλληλα PWM σήμα συχνότητας 1kHZ.

AF_DCMotor motor2(4, MOTOR12_1KHZ); //Δημιουργούμε το motor2 χρησιμοποιώντας την έξοδο M2 στο Motor Drive Shield και στέλνοντας παράλληλα PWM σήμα συχνότητας 1kHZ.

Servo myservo; //Δημιουργούμε μια σταθερά servo ώστε να ελέγχουμε το servo μοτερ.

int pos = 0; //Σε αυτό το κομμάτι ορίζουμε τις σταθερές όπου θα χρησιμοποιήσουμε στο πρόγραμμα.

```
  int maxDist = 0;
  int maxAngle = 0;
  int maxRight = 0;
  int maxLeft = 0;
  int maxFront = 0;
int course = 0;
int curDist = 0;
String motorSet = "";
int speedSet = 0;
```

//---------- SETUP LOOP ----------

```
void setup() {
  myservo.attach(9);          //Δηλώνουμε το servo μοτερ ότι θα είναι στο Pin 9 (Serv0_2 στο Motor Drive Shield).
  myservo.write(90);          //Ορίζουμε τη θέση '0' για το servo στις 90 μοίρες.
  delay(2000);                // Δημιουργούμε χρονοκαθυστέρηση 2 δευτερολέπτων.
  checkPath();                // Τρέχουμε τη ρουτίνα CheckPath ώστε το ρομπότ να βρει το καλύτερο δρόμο για να ξεκινήσει να κινείται.
  motorSet = "FORWARD";       //Θέτουμε τη σταθερά motorSet σε "FORWARD".
  myservo.write(90);          //Σιγουρεύουμε ότι το servo είναι στη θέση των 90 μοιρών.
```

```
  moveForward();                    //Τρέχουμε τη συνάρτηση moveForward ώστε να ξεκινήσει
                                    το ρομπότ να κινείται.
}
//-------------------------------------------------------------------------------------------------------------------
----------------

//-----------------------------------------MAIN LOOP ---------------------------------------------------
------------------------
void loop() {
  checkForward();       // Τσεκάρουμε αν το ρομπότ είναι να κινηθεί μπροστά ότι όντως τα μοτέρ
                        κινούνται στη σωστή κατεύθυνση.
  checkPath();          //Θέτουμε τον αισθητήρα να ξεκινήσει να σκανάρει για πιθανά εμπόδια.
}
//-------------------------------------------------------------------------------------------------------------------
-----------------
void checkPath() {
  int curLeft = 0;
  int curFront = 0;
  int curRight = 0;
  int curDist = 0;
  myservo.write(144);           //Θέτουμε το servo να κινηθεί 54 μοίρες αριστερά από την αρχική
                                του θέση.
  delay(120);                   //Δημιουργούμε χρονοκαθυστέρηση 120ms ώστε το servo να φτάσει στη
                                θέση του.
  for(pos = 144; pos >= 36; pos-=18)
  //Φτιάχνουμε μια επανάληψη όπου θα κινεί το servo (άρα και τον αισθητήρα) 54 μοίρες
  αριστερά και δεξιά από την αρχική του θέση.
  {
    myservo.write(pos);  // Λέμε στο servo να πάει στη θέση όπου δείχνει η μεταβλητή "pos".
    delay(90);          //Δημιουργούμε χρονοκαθυστέρηση 90ms ώστε το servo να φτάσει στη
                        θέση του.
    checkForward();             //Τσεκάρουμε ότι το ρομπότ συνεχίζει να κινείται μπροστά.
```

```
   curDist = readPing();          //Θέτουμε στη μεταβλητή curDist τη μετρούμενη απόσταση του
                                  αισθητήρα.
   if (curDist < COLL_DIST && curDist!=0) { //Αν η curDist είναι μικρότερη από την
COLL_DIST.
     checkCourse(); // Τρέξε τη συνάρτηση checkCourse.
     break; // Πηδάμε έξω από το loop.
   }
   if (curDist < TURN_DIST) { // Αν η curDist είναι μικρότερη από τη TURN_DIST.
     changePath(); // Τρέξε τη συνάρτηση changePath.
   }
   if (curDist > curDist) {maxAngle = pos;}
   if (pos > 90 && curDist > curLeft) { curLeft = curDist;}
   if (pos == 90 && curDist > curFront) {curFront = curDist;}
   if (pos < 90 && curDist > curRight) {curRight = curDist;}
  }
  maxLeft = curLeft;
  maxRight = curRight;
  maxFront = curFront;
}
//-------------------------------------------------------------------------------------------------------------
------------------
void setCourse() {
   if (maxAngle < 90) {turnRight();}
   if (maxAngle > 90) {turnLeft();}
   maxLeft = 0;
   maxRight = 0;
   maxFront = 0;
}
//-------------------------------------------------------------------------------------------------------------
------------------
void checkCourse() {
```

```
//Εδώ το ρομπότ βρίσκεται πολύ κοντά στο να πέσει σε αντικείμεβο, οπότε κάνει πίσω σταματάει και βρίσκει που είναι το άδειο μονοπάτι ώστε να πάει.
  moveBackward();
  delay(500);
  moveStop();
  setCourse();
}
//-------------------------------------------------------------------------------------------------------------------
void changePath() {
  if (pos < 90) {veerLeft();}
  if (pos > 90) {veerRight();}
}
//-------------------------------------------------------------------------------------------------------------------

int readPing() { // Διάβασε από τον αισθητήρα την απόσταση.
  delay(70);
  unsigned int uS = sonar.ping();
  int cm = uS/US_ROUNDTRIP_CM;
  return cm;
}
//-------------------------------------------------------------------------------------------------------------------
void checkForward() { if (motorSet=="FORWARD") {motor1.run(FORWARD); motor2.run(FORWARD); } }     // Σιγουρεύουμε ότι τα μοτέρ κινούνται προς τα μπρος.
//-------------------------------------------------------------------------------------------------------------------
void checkBackward() { if (motorSet=="BACKWARD") {motor1.run(BACKWARD); motor2.run(BACKWARD); } } // Σιγουρεύουμε ότι τα μοτέρ κινούνται προς τα πίσω.
//-------------------------------------------------------------------------------------------------------------------
```

```
// Σε κάποιες περιπτώσεις το Motor Drive Shield μπορεί να σταματήσει να λειτουργεί όταν η τάση είναι χαμηλή (όταν αρχίζουν και αδειάζουν οι μπαταρίες).
// Οι παρακάτω συναρτήσεις σιγουρεύουν την ορθή λειτουργία του Motor Drive Shield.

//-------------------------------------------------------------------------------------------------------------------
void moveStop() {motor1.run(RELEASE); motor2.run(RELEASE);}  // Εδώ σταματάμε τα μοτέρ.
//-------------------------------------------------------------------------------------------------------------------
void moveForward() {
   motorSet = "FORWARD";
   motor1.run(FORWARD);      // Το μοτέρ1 κινείται μπροστά.
   motor2.run(FORWARD);      // Το μοτέρ2 κινείται μπροστά.
 for (speedSet = 0; speedSet < MAX_SPEED; speedSet +=2)
// Με αυτή την επανάληψη πετυχένουμε να μειώσουμε τη ταχύτητα των μοτέρ ώστε να μην αδειάζουν γρήγορα οι μπαταρίες.
 {
   motor1.setSpeed(speedSet);
   motor2.setSpeed(speedSet+MAX_SPEED_OFFSET);
   delay(5);
 }
}
//-------------------------------------------------------------------------------------------------------------------
void moveBackward() { // Συνάρτηση όπου το ρομπότ πρέπει να κινηθεί πίσω.
   motorSet = "BACKWARD";
   motor1.run(BACKWARD);      // Το μοτέρ1 κινείται πίσω.
   motor2.run(BACKWARD);     //  Το μοτέρ1 κινείται πίσω.
 for (speedSet = 0; speedSet < MAX_SPEED; speedSet +=2)
// Με αυτή την επανάληψη πετυχένουμε να μειώσουμε τη ταχύτητα των μοτέρ ώστε να μην αδειάζουν γρήγορα οι μπαταρίες.
 {
```

```
    motor1.setSpeed(speedSet);
    motor2.setSpeed(speedSet+MAX_SPEED_OFFSET);
    delay(5);
  }
}
//-----------------------------------------------------------------------------------------------------------------
-----------------
void turnRight() { // Συνάρτηση όπου το ρομπότ πρέπει να κινηθεί δεξιά.
  motorSet = "RIGHT";
  motor1.run(FORWARD);      // Το μοτέρ1 κινείται μπροστά.
  motor2.run(BACKWARD);     // Το μοτέρ1 κινείται πίσω.
  delay(400);               // Με αυτή τη χρονοκαθυστέρηση άνουμε να μοτέρ να συνεχίζουν τη
                            κίνηση τους για 400ms.
  motorSet = "FORWARD";
  motor1.run(FORWARD);      // Εδώ κάνουμε και τα δυο μοτέρ να κινούνται μπροστά.
  motor2.run(FORWARD);
}
//-----------------------------------------------------------------------------------------------------------------
-----------------
void turnLeft() { //Συνάρτηση όπου το ρομπότ πρέπει να κινηθεί αριστερά.
  motorSet = "LEFT";
  motor1.run(BACKWARD);     // Το μοτέρ1 κινείται πίσω.
  motor2.run(FORWARD);      // Το μοτέρ2 κινείται μπροστά.
  delay(400);               // Με αυτή τη χρονοκαθυστέρηση άνουμε να μοτέρ να συνεχίζουν τη
                            κίνηση τους για 400ms.
  motorSet = "FORWARD";
  motor1.run(FORWARD);      // Εδώ κάνουμε και τα δυο μοτέρ να κινούνται μπροστά.
  motor2.run(FORWARD);
}
//-----------------------------------------------------------------------------------------------------------------
-----------------
void veerRight() {motor2.run(BACKWARD); delay(400); motor2.run(FORWARD);}
```

// Όταν πρέπει να στρίψει δεξιά το ρομπότ τότε για 400ms θέτουμε το δεξί μοτέρ σε κίνηση προς τα πίσω.

//----------

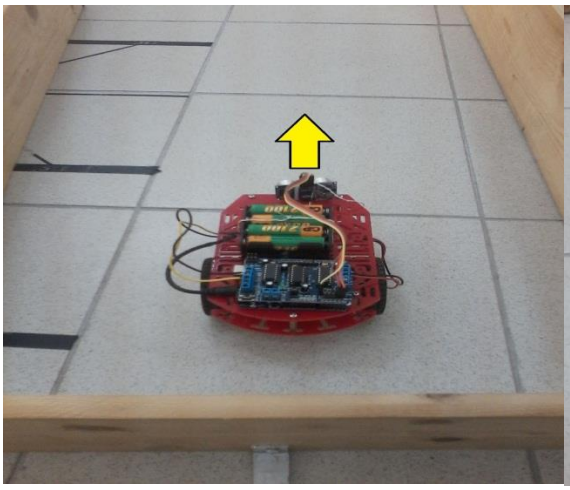
void veerLeft() {motor1.run(BACKWARD); delay(400); motor1.run(FORWARD);}

// Όταν πρέπει να στρίψει αριστερά το ρομπότ τότε για 400ms θέτουμε το αριστερά μοτέρ σε κίνηση προς τα πίσω.

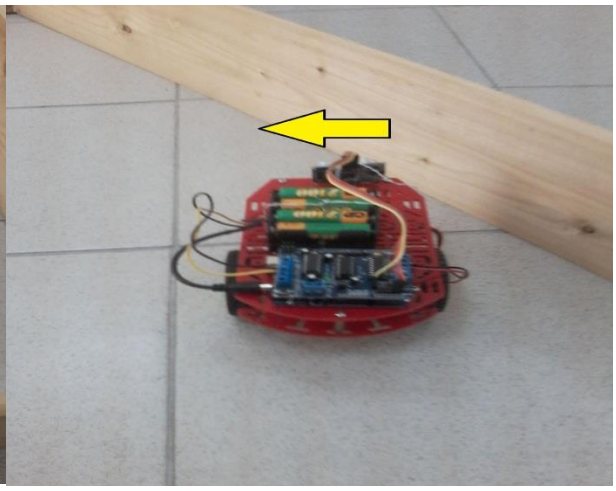
//----------

5.3 Δοκιμή ορθής λειτουργίας

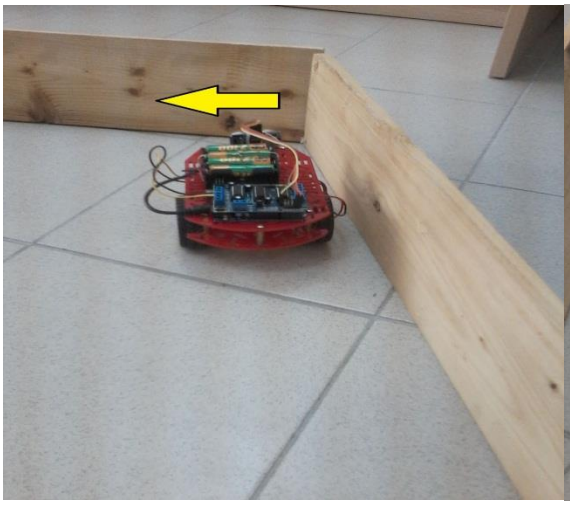
Για να δούμε αν το όχημα κινείται στο χώρο τηρώντας τις προσδοκίες αλλά και το προγραμματισμό που έγινε, φτιάχτηκε ένας διάδρομος από ξύλα. Βάλαμε το όχημα στο διάδρομο και το τροφοδοτήσαμε από τις μπαταρίες. Σκοπός είναι να δούμε αν αποφεύγει να αντικείμενα που βρίσκει μπροστά του. Στις παρακάτω εικόνες παρουσιάζονται στιγμιότυπα από την κίνηση του οχήματος.



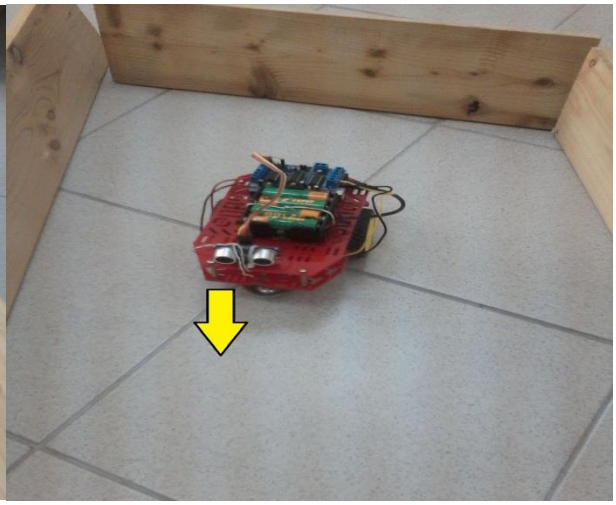
Εικόνα 23: Το όχημα ξεκινάει τη πορεία του



Εικόνα 24: Αποφυγή εμποδίου από το όχημα και συνέχεια πορείας του προς τα αριστερά.



Εικόνα 24: Αποφυγή εμποδίου απ'το όχημα



Εικόνα 25: Συνέχιση πορείας οχήματος μετά την αποφυγή του εμποδίου.

Αρχικά, στην εικόνα 23 το όχημα έχει τροφοδοτηθεί από τις μπαταρίες και ξεκινάει την πορεία του στη κατεύθυνση που δείχνει το βελάκι (ευθεία) λόγω του ότι δεν έχει εμπόδιο μπροστά του για να αποφύγει. Στην εικόνα 24 είναι πολύ κοντά στο να χτυπήσει το εμπόδιο, επομένως στρίβει ελαφρώς αριστερά, στη πορεία όπου δείχνει το βελάκι ώστε να το αποφύγει.

Στην εικόνα 25, επίσης βρίσκεται κοντά στο να χτυπήσει το εμπόδιο μπροστά του επομένως το αποφεύγει για να συνεχίσει τη πορεία του στην εικόνα 26. Αυτός ο κύκλος επαναλαμβάνεται συνεχώς. Συνεχίζει την ευθεία πορεία του και πηγαίνει προς τη κατεύθυνση όπου υπάρχουν λιγότερα εμπόδια.

ΚΕΦΑΛΑΙΟ 6^{Ο}

Συμπεράσματα

Στην παρούσα εργασία διερευνήθηκε ο τρόπος υλοποίησης μιας αυτόνομης ρομποτικής κατασκευής χρησιμοποιώντας εξαρτήματα χαμηλού κόστους που είναι ευρέως διαθέσιμα στο εμπόριο. Αναπτύχθηκε ένα έντροχο ρομποτικό σύστημα διαφορικής οδήγησης βασισμένο στον μικροελεγκτή Arduino και μελετήθηκε ο τρόπος ελέγχου είτε με κώδικα που εκτελείται πάνω στο όχημα είτε σε απομακρυσμένο Η/Υ με χρήση MATLAB. Η εφαρμοσιμότητας της συγκεκριμένης προσέγγισης δοκιμάστηκε σε ένα δομημένο χώρο με εμπόδια.

Ο συγκεκριμένος τρόπος ανάπτυξης αποδείχθηκε εύχρηστος και ιδανικός για την εκμάθηση βασικών αρχών που σχετίζονται με την επιστήμη του Ηλεκτρονικού καθώς και βασικών αρχών προγραμματισμού. Στο μέλλον θα διερευνηθεί κατά πόσο η συγκεκριμένη πλατφόρμα μπορεί να χρησιμοποιήθει με τις κατάλληλες προσθήκες σαν πλατφόρμα δοκιμών σύνθετων αλγορίθμων ελέγχου ρομποτικών συσκευών.

ΚΕΦΑΛΑΙΟ 7^{Ο}

Βιβλιογραφία

[1] ***Ρομποτικό Όχημα Όδυσσέας***

http://www.grobot.gr/index.php/2008-04-19-14-02-52/227-2011-09-15-07-05-20

[2] ***Ρομποτικό Όχημα με ικανότητα αναγνώρισης περιοχής***

http://www.grobot.gr/index.php/2008-04-19-14-02-52/249-2012-07-09-07-27-26

[3] ***Εικόνα 7: Διάγραμμα δομικών μερών του ρομποτικού οχήματος***

http://apcmag.com/arduino-masterclass-part-4-build-a-mini-robot.htm/

[4] ***Μονάδα ελέγχου***

Μετάφραση του παρακάτω ιστότοπου:

https://www.arduino.cc/en/Main/ArduinoBoardUno

[5] ***Εικόνα 8: Arduino Uno***

http://www.slideshare.net/muhammadowais94043/arduino-uno-28044428

[6] ***Εικόνα 9: Arduino Motor Drive Shield***

http://playground.arduino.cc/Main/AdafruitMotorShield

[7] ***Arduino Motor Shield***

https://www.arduino.cc/en/Main/ArduinoMotorShieldR3

[8] ***Εικόνα 11: Αισθητήρας μέτρησης απόστασης (HC-SR04)***

http://www.famosastudio.com/ultrasonic-range-sensor-hc-sr04

[9] ***Εικόνα 18: Το L293D***

https://cdn.solarbotics.com/products/photos/8269faf0ba7491d463dc7ed1010d3c3b/lrg/l293d.jpg