ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΕΛΛΑΔΟΣ

ΔΙΕΘΝΕΣ ΠΑΝΕΠΙΣΤΗΜΙΟ ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ, ΥΠΟΛΟΓΙΣΤΩΝ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

«Σχεδίαση και υλοποίηση μικρού αυτόνομου οχήματος με βάση τον ελεγκτή Arduino»

Μπάμπη Ελένη

Αριθμός Μητρώου: 4605

Επιβλέπων: Ιωάννης Καλόμοιρος, Καθηγητής

Υπεύθυνη Δήλωση

Βεβαιώνω ότι είμαι συγγραφέας αυτής της διπλωματικής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της, είναι πλήρως αναγνωρισμένη και αναφέρεται στην πτυχιακή εργασία. Επίσης έχω αναφέρει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επίσης βεβαιώνω ότι αυτή η πτυχιακή εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για τις απαιτήσεις του προγράμματος σπουδών του Τμήματος Μηχανικών Πληροφορικής Τ.Ε. της Σχολής Τεχνολογικών Εφαρμογών του Τ.Ε.Ι. Κεντρικής Μακεδονίας.

Η Δηλούσα

Μπάμπη Ελένη

Περίληψη

Στην παρακάτω πτυχιακή εργασία καλούμαστε να λύσουμε το πρόβλημα της σχεδίασης και υλοποίησης δυο ρομποτικών οχημάτων με τη χρήση απλών ελεγκτών χαμηλού κόστους. Οι δύο κατασκευές θα χρησιμοποιούν παρόμοια αλλά και διαφορετικά υλικά.

Στην πρώτη σχεδίαση, το όχημα θα κινείται αυτόνομα χωρίς ανθρώπινη παρέμβαση με τη βοήθεια αισθητήρων και θα λαμβάνει δεδομένα οδικού δικτύου. Θα έχει ως βάση του την πλακέτα Arduino στην οποία θα συνδέονται οι αισθητήρες. Τα εργαλεία που θα χρησιμοποιηθούν περιλαμβάνουν κινητήρες DC, αισθητήρας αποστάσεων, οπτικοί αισθητήρες στροφών και το Arduino IDE για τον προγραμματισμό των εξαρτημάτων.

Στη δεύτερη σχεδίαση, το όχημα θα κινείται και θα ελέγχεται ασύρματα μέσω οικιακού δικτύου. Θα χρησιμοποιήσουμε την πλατφόρμα NodeMcu Lua ESP8266 ως τη βάση για το ρομποτικό όχημα. Τα εργαλεία που θα χρησιμοποιηθούν περιλαμβάνουν κινητήρες DC, ασύρματες μονάδες επικοινωνίας, το Arduino IDE για τον προγραμματισμό της πλακέτας και την πλατφόρμα Blynk για τον σχεδιασμό και την εκτέλεση της εφαρμογής ελέγχου του οχήματος.

Η μεθοδολογία που ακολουθήσαμε περιλαμβάνει τη συγκέντρωση και τη σύνδεση των απαραίτητων εξαρτημάτων και αισθητήρων, καθώς και τον προγραμματισμό τους ώστε να λειτουργούν αξιόπιστα.

Η πτυχιακή εργασία αναλύει τη διαδικασία δημιουργίας του κάθε ρομποτικού οχήματος ξεχωριστά, εξετάζει την εφαρμογή των αλγορίθμων και των τεχνικών ελέγχου, και παρουσιάζει τα αποτελέσματα των πειραμάτων. Επιπλέον, εξετάζει τη σημασία του έργου σε σχέση με τη ρομποτική και την αυτοματοποίηση, και προτείνει πιθανές μελλοντικές επεκτάσεις και βελτιώσεις στον τομέα της ασύρματης ρομποτικής. Τέλος, η εργασία προσφέρει μια περιεκτική λύση για τη δημιουργία ενός αξιόπιστου τηλεκατευθυνόμενου ρομποτικού οχήματος με τη χρήση του ESP8266 καθώς και ενός αυτόνομου ρομποτικού οχήματος με τη χρήση Arduino IDE.

Abstract

In the present thesis, we are tasked with addressing the issue of designing and implementing two robotic vehicles using low-cost microcontrollers. These two constructions will utilize similar and different hardware components.

In the first design, the vehicle will operate autonomously without human intervention, relying on sensors to collect data and make informed decisions. It will be based on the Arduino platform, and various sensors will be connected to it. The tools to be used include DC motors, distance sensors, optical encoders for measuring wheel rotation, and the Arduino IDE for programming these components.

In the second design, the vehicle will move under the control of a network application, using the local home network. We will utilize the NodeMcu Lua ESP8266 platform as the foundation for this robotic vehicle. The tools involved in this design comprise DC motors, wireless communication modules, the Arduino IDE for programming the board and the Blynk platform for designing and executing the vehicle's control application.

The methodology we followed includes gathering and connecting the necessary components and sensors, as well as programming them to ensure reliable operation.

The thesis analyses the process of creating each robotic vehicle separately, examines the application of control algorithms and techniques, and presents the results of experiments conducted with these vehicles. Furthermore, it explores the significance of this work in the realm of robotics and automation and suggests possible future expansions and improvements in the field of wireless robotics. Finally, the thesis provides a comprehensive solution for creating a reliable and adaptable robotic vehicle which is controlled remotely, using the ESP8266 platform.

Περιεχόμενα

Πίνακας εικόνων

Πίνακας πινάκων

1ο ΚΕΦΑΛΑΙΟ: Εισαγωγή στα εκπαιδευτικά οχήματα της ρομποτικής

1.1 Ρομποτικά οχήματα: σχεδιασμός και αυτοματοποίηση

Η ρομποτική και η αυτοματοποίηση έχουν εξελιχθεί σημαντικά τα τελευταία χρόνια, προσφέροντας νέες δυνατότητες για την εφαρμογή τεχνολογικών λύσεων σε ποικίλους τομείς. Σε αυτό το πλαίσιο, η δημιουργία αυτόνομων ή τηλεκατευθυνόμενων οχημάτων αποτελεί έναν τομέα εξαιρετικού ενδιαφέροντος, καθώς ανοίγει νέες προοπτικές για την αυτοματοποίηση, την εκπαίδευση και την ψυχαγωγία.

Το παρόν πρόβλημα προς λύση αποσκοπεί στη δημιουργία δύο ρομποτικών οχημάτων, ενός αυτόνομου και ενός τηλεκατευθυνόμενου, με τη χρήση της πλατφόρμας Arduino, της πλατφόρμας NodeMcu Lua ESP8266 και υλικά τα οποία είναι πολύ εύκολο να βρεθούν στο εμπόριο με σχετικά χαμηλό κόστος. Η πρόκληση εδώ είναι η ανάπτυξη ρομποτικών οχημάτων που να είναι ολοκληρωμένα, ευέλικτα και να μπορούν να χρησιμοποιηθούν για εκπαιδευτικούς σκοπούς.

Η κατασκευή αυτών των οχημάτων απαιτεί τη σύνδεση και τον συντονισμό ποικίλων εξαρτημάτων, όπως κινητήρες DC, ασύρματες μονάδες επικοινωνίας, και την ανάπτυξη του κατάλληλου λογισμικού ελέγχου.

Οι προκλήσεις περιλαμβάνουν την αποτελεσματική αναγνώριση των περιβαλλοντικών συνθηκών, τον περιορισμό της αυτονομίας που προσφέρουν οι μπαταρίες και την διαφορά στο συντονισμό της ταχύτητας των δύο τροχών.

Επίσης, θα πρέπει να αναπτυχθεί κατάλληλο λογισμικό είτε για τον προγραμματισμό των αισθητήρων της πρώτης κατασκευής είτε για τον απομακρυσμένο έλεγχο της δεύτερης.

1.2 Μεθοδολογία Σχεδίασης και Υλοποίησης

Η μεθοδολογία που ακολουθείται για τη δημιουργία των ρομποτικών οχημάτων περιλαμβάνει ακριβή στάδια και διαδικασίες που θα εξεταστούν παρακάτω.

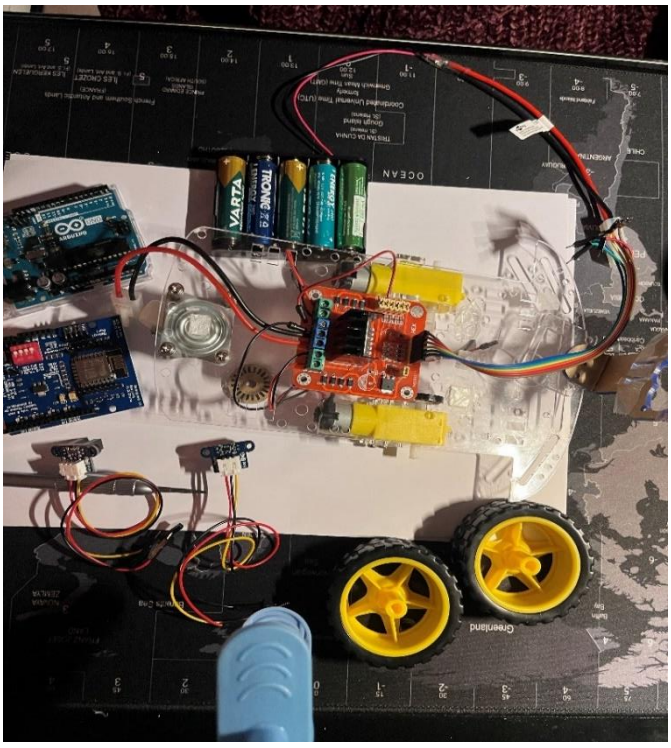
Αρχικά, θα οριστούν ο σχεδιασμός και οι ακριβείς προδιαγραφές του κάθε ρομποτικού οχήματος. Αυτές οι προδιαγραφές θα περιλαμβάνουν τα ακόλουθα:

- Είδος των κινητήρων που θα χρησιμοποιηθούν.
- Τύποι αισθητήρων.
- Απαιτήσεις για την ασύρματη επικοινωνία και τον τηλεχειρισμό.
- Τρόπος τροφοδοσίας των εξαρτημάτων

Οι δύο κατασκευές θα χρησιμοποιήσουν τους ίδιους κινητήρες, το ίδιο σασί και τον ίδιο τρόπο τροφοδοσίας.

Με βάση τις προδιαγραφές, θα πρέπει να συλλεγούν τα αναγκαία υλικά και εξαρτήματα για την κατασκευή τους. Αυτά μπορεί να περιλαμβάνουν:

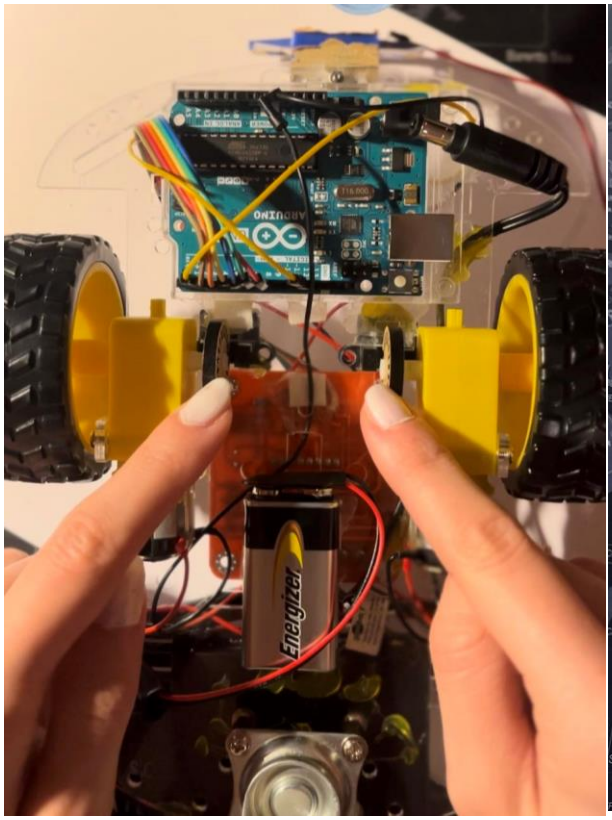
- Ασύρματη μονάδα επικοινωνίας για τον τηλεχειρισμό.
- Μπαταρίες.
- Καλώδια.
- Σασί οχήματος
- Κινητήρες DC και τροχοί.
- Αισθητήρες αποστάσεων.
- Πλατφόρμα Arduino.
- Ρόδες
- Οπτικοί Ελεγκτές



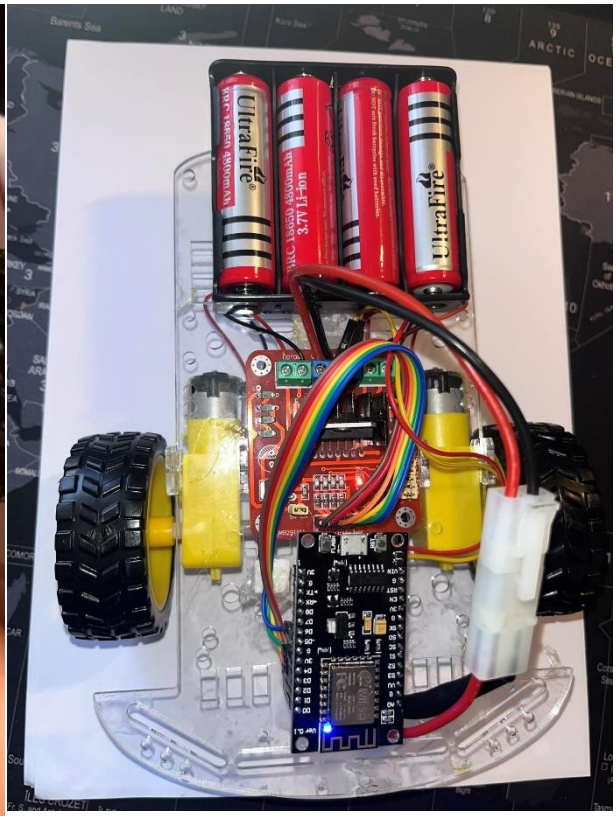
Εικόνα 1.1: Συγκέντρωση των απαραίτητων υλικών

Μετά τη συγκέντρωση των υλικών, θα πρέπει να σχεδιαστεί η φυσική διάταξη των οχημάτων και να συναρμολογηθούν τα εξαρτήματα. Αυτό περιλαμβάνει τη συνδεσμολογία των κινητήρων, των αισθητήρων και τη δημιουργία της μηχανικής δομής του οχήματος.

Προκειμένου να επιτρέπεται η αποδοτική κίνηση τους, θα πρέπει να γίνει σωστή διάταξη των εξαρτημάτων πάνω στο σασί. Η παραμικρή ανισορροπία μπορεί να επηρεάσει την κίνηση των τροχών. Επίσης, θα πρέπει να γίνει σωστή διάταξη των διάφορων καλωδίων ώστε να μην δυσχεραίνουν τους τροχούς και να είναι εύκολα κατανοητές οι συνδεσμολογίες από κάποιον που θα επιχειρήσει να αναπαράγει τον σχεδιασμό .



Εικόνα 1.2: Συναρμολόγηση της πρώτης κατασκευής

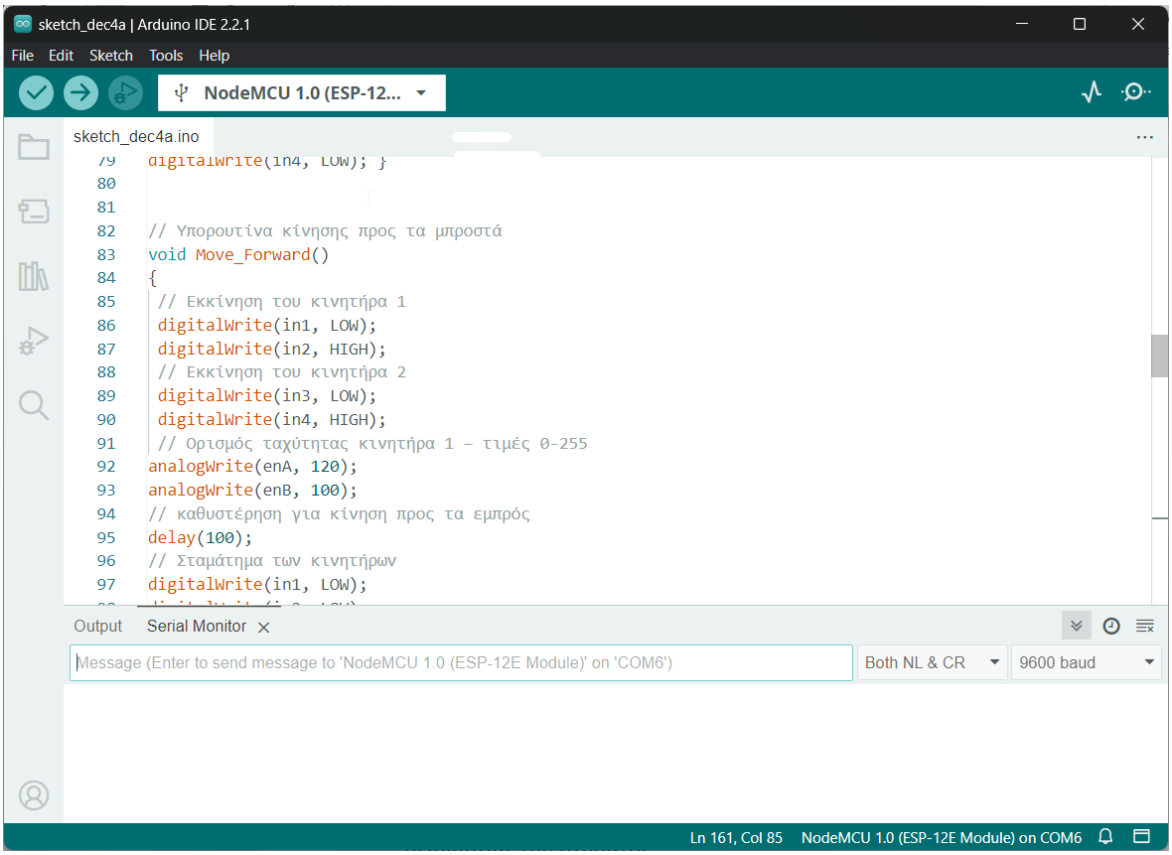


Εικόνα 1.3: Συναρμολόγηση της δεύτερης κατασκευής.

Έπειτα, θα γίνει ο προγραμματισμός των ρομποτικών οχημάτων.

Αυτό περιλαμβάνει τον έλεγχο των κινητήρων και την επικοινωνία με τον Arduino στην πρώτη κατασκευή. Θα γίνει χρήση του λογισμικού Arduino IDE.

Στη δεύτερη κατασκευή περιλαμβάνει τον έλεγχο των κινητήρων και την επικοινωνία με την ασύρματη μονάδα τηλεχειρισμού. Θα γίνει χρήση του λογισμικού Arduino IDE και της πλατφόρμας Blynk.



Εικόνα 1.4: Στιγμιότυπο του κώδικα στο λογισμικό Arduino IDE.

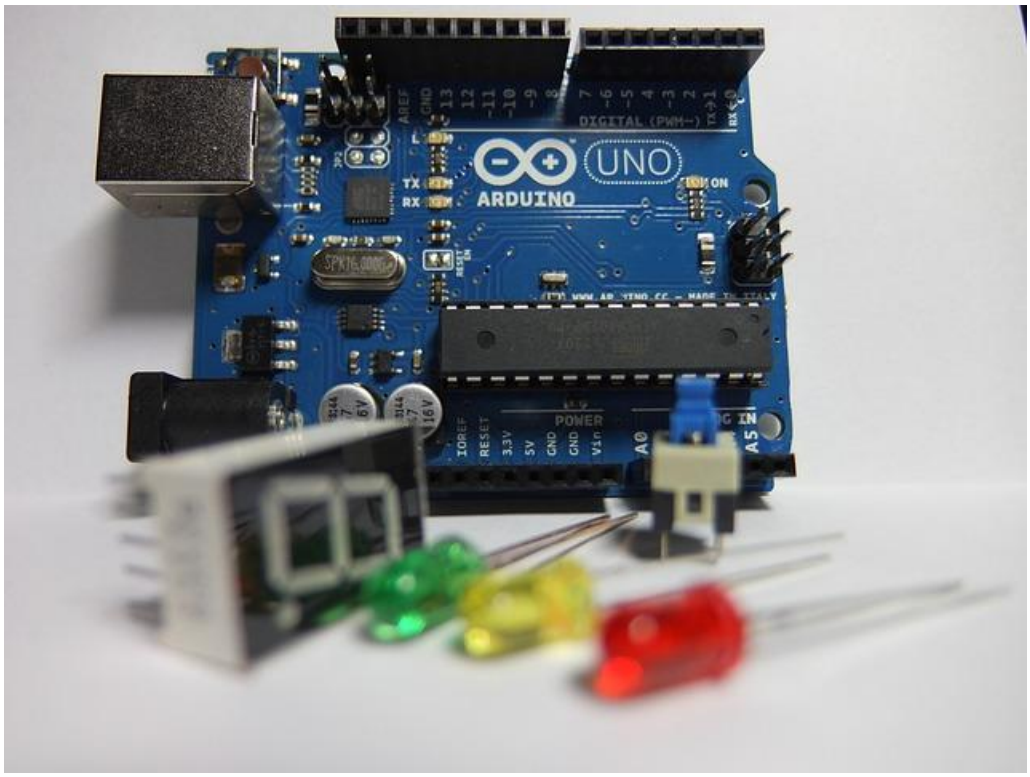
Τέλος, θα γίνουν δοκιμές και βελτιώσεις καθώς και ανάλυση των αποτελεσμάτων. Θα πρέπει να ελεγχθεί η λειτουργία των οχημάτων σε πραγματικές συνθήκες και να προσαρμοστούν τυχόν ανεπάρκειες ή προβλήματα. Οι βελτιώσεις μπορούν να περιλαμβάνουν την βελτίωση των αλγορίθμων ελέγχου, την προσθήκη νέων λειτουργιών ή την αύξηση της αξιοπιστίας του οχήματος.

1.3 Ιστορική Ανασκόπηση

Η πορεία ξεκινά με τις πρώτες προσπάθειες στον χώρο της ρομποτικής και του τηλεκατευθυνόμενου ελέγχου. Αρχικά, ο τομέας αυτός βασιζόταν σε περίπλοκες και ακριβές συσκευές. Ωστόσο, η εμφάνιση της πλατφόρμας Arduino στο τοπίο της ρομποτικής φέρνει μια σημαντική αλλαγή. Οι προγραμματιζόμενες πλακέτες Arduino επιτρέπουν τη δημιουργία προσιτών, ευέλικτων και προσαρμόσιμων ρομποτικών οχημάτων.

Η πλατφόρμα Arduino έχει εξελιχθεί σημαντικά από τον καιρό που πρωτοεμφανίστηκε. Αρχικά, βασιζόταν σε μια απλή πλακέτα μικροελεγκτή, αλλά με τον χρόνο έχουν δημιουργηθεί ποικίλες παραλλαγές και προσθήκες, όπως το Arduino Uno, το Arduino Mega, και πολλά άλλα μοντέλα. Η επέκταση των υλικών και των βιβλιοθηκών προγραμματισμού έχει καταστήσει την Arduino πολύ ισχυρό εργαλείο για τη δημιουργία ρομποτικών οχημάτων.

Πλέον χρησιμοποιείται για την αυτοματοποίηση της παραγωγής στην εκπαίδευση, για την ψυχαγωγία και την έρευνα, οι εφαρμογές της τεχνολογίας Arduino στα ρομποτικά είναι πολλές και ποικίλες.



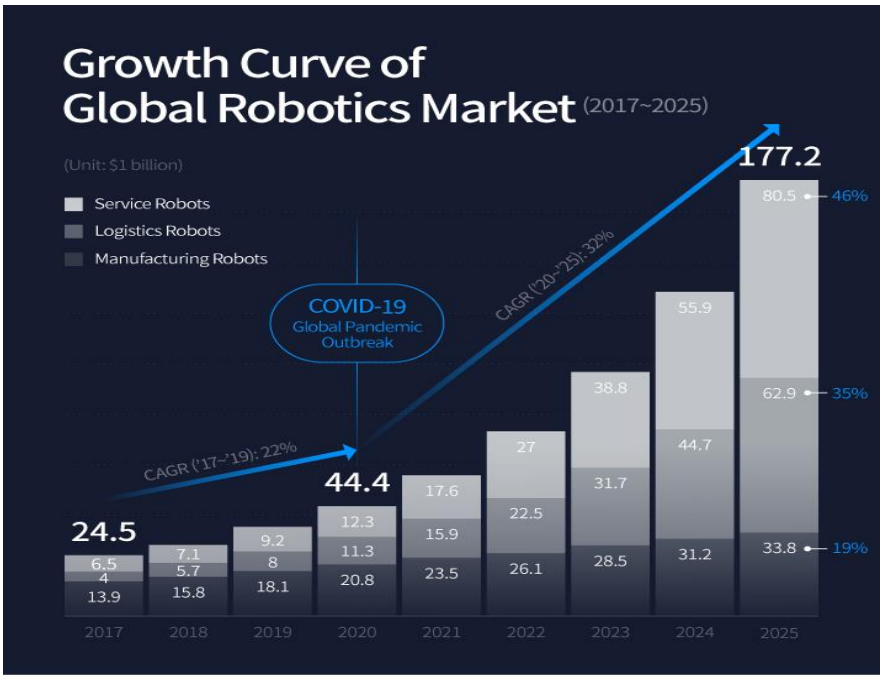
Εικόνα 1.5: Η πλακέτα Arduino Uno.
[Η εικόνα αντλήθηκε από τον διαδικτυακό ιστότοπο https://pixabay.com/photos/arduino-electronics-board-computer-631977/]

1.4 Η κατάσταση της «Τέχνης»

Η κατάσταση των ασύρματα ελεγχόμενων ρομποτικών οχημάτων αντικατοπτρίζει την ταχεία εξέλιξη και τη σημαντική ανάπτυξη που έχει σημειώσει τον τελευταίο καιρό. Η τεχνολογική πρόοδος, καθώς και η διαθεσιμότητα οικονομικών και ευρέως διαθέσιμων υλικών και εργαλείων, όπως ο Arduino, έχουν επιτρέψει στους κατασκευαστές και τους ερασιτέχνες να δημιουργήσουν όλο και πιο εξελιγμένα και διασκεδαστικά ρομποτικά οχήματα.

Αυτά τα ασύρματα ελεγχόμενα ρομποτάκια έχουν εφαρμογές σε ποικίλους τομείς, από την εκπαίδευση και τη διασκέδαση μέχρι τη βιομηχανία και την έρευνα. Το πεδίο αυτό συνεχίζει να εξελίσσεται με την ενσωμάτωση προηγμένων τεχνολογιών, όπως οι αισθητήρες, οι κάμερες, οι αυτόνομοι αλγόριθμοι και τα συστήματα τηλεχειρισμού με τη χρήση smartphones ή ασύρματων ελεγκτών.

Οι ανθρωποκεντρικοί σχεδιασμοί και οι ανοικτοί πόροι, όπως η πλατφόρμα Arduino που χρησιμοποιείται στην πτυχιακή εργασία, επιτρέπουν σε περισσότερους ανθρώπους να δημιουργούν και να συμμετέχουν στον κόσμο των ρομποτικών. Αυτό έχει οδηγήσει σε μια ευρύτερη διάδοση της τεχνολογίας και στη δημιουργία καινοτόμων προτάσεων για τα ασύρματα ελεγχόμενα ρομποτικά οχήματα.



Εικόνα 1.6: Καμπύλη Ανάπτυξης της ρομποτικής αγοράς.
[Η εικόνα αντλήθηκε από τον διαδικτυακό ιστότοπο της Hyundai https://www.hyundaimotorgroup.com/story/CONT0000000000001671]

1.4.1 Εκπαιδευτικά ρομποτικά οχήματα της αγοράς

Παρακάτω θα δούμε κάποια ευρέως διαθέσιμα ρομποτικά οχήματα που χρησιμοποιούν παρόμοιους ή ίδιους ελεγκτές με το ρομπότ της εργασίας.

mBot Robot Kit

Εικόνα 1.7: [Η εικόνα αντλήθηκε από τον διαδικτυακό ιστότοπο https://www.eptecstore.com/product/mbot-v1-2-bluetooth-version/]

Το mBot Robot Kit είναι ένα αριστούργημα δημιουργημένο από παιδιά. Είναι ένα εκπαιδευτικό ρομπότ του οποίου ο σκοπός ήταν να εξοικειώσει τους μαθητές με τον προγραμματισμό του Arduino και την ρομποτική γενικότερα.

Στο κιτ περιλαμβάνονται:

- Μια έγχρωμη οθόνη, ηχείο, μικρόφωνο, αισθητήρας φωτός, γυροσκόπιο, λωρίδες LED, ενσωματωμένο WiFi, επαναφορτιζόμενη μπαταρία.

Εκτός από τους παραπάνω αισθητήρες και μηχανικά μέρη, περιλαμβάνει και:

- Έναν Ultrasonic αισθητήρα ο οποίος διαθέτει 8 φώτα περιβάλλοντος για να μετρά την απόσταση με ακρίβεια.
- Έναν αισθητήρα Quad RGB ο οποίος μπορεί να ανιχνεύσει ταυτόχρονα γραμμές και χρώματα. Και ο σχεδιασμός τετραπλού καναλιού καθιστά δυνατή την ακριβή ανίχνευση διασταυρώσεων. Τα φώτα του περιβάλλοντος συμβάλλουν στη μείωση του ψευδούς ρυθμού αναγνώρισης.

Δεν έρχεται συναρμολογημένο.

Η τιμή του ανέρχεται στα 130 ευρώ.

Dash &Dot

Εικόνα 1.8 [Η εικόνα αντλήθηκε από τον διαδικτυακό ιστότοπο https://www.normalpl.org/things/dash-dot-robots

Το Dash &Dot είναι ένα ζεύγος ρομπότ σχεδιασμένα τα καταλαβαίνουν τι συμβαίνει γύρω τους και να ανταποκρίνονται. Ακούνε ήχους, ανιχνεύουν αντικείμενα και ξέρουν αν τα έχουμε μετακινήσει. Ο προγραμματισμός τους είναι σχετικά εύκολος καθώς γίνεται με Drag & Drop interface.

Περιλαμβάνουν διάφορους αισθητήρες:

- Αισθητήρες απόστασης.
- Αισθητήρες ήχου για φωνητικές εντολές.
- Αισθητήρες απόστασης.
- Beacon αισθητήρες για την ανίχνευση άλλων ρομπότ
- Επιταχυνσιόμετρο

Το κιτ έρχεται συναρμολογημένο.

Η τιμή τους ανέρχεται στα 40 ευρώ.

Micro: Maqueen

Εικόνα 1.9: [Η εικόνα αντλήθηκε από τον διαδικτυακό ιστότοπο https://grobotronics.com/micro-maqueen-lite-robot-platform.html

Ο Maqueen είναι ένα μεταλλικό μικροσκοπικό ρομπότ γραφικού προγραμματισμού. Είναι εύκολο στη χρήση και στο προγραμματισμό. Υποστηρίζει Makecode, Scratch, Python.

Μέσα στο κιτ εμπεριέχονται τα εξής:

- Αισθητήρας υπέρυθρων
- Servo Interface
- Αισθητήρες βαρύτητας
- Κινητήρας
- Infrared Receiver

Εικόνα 1.10: Διάφορες λειτουργίες του Maqueen
[[Η εικόνα αντλήθηκε από τον διαδικτυακό ιστότοπο https://grobotronics.com/micro-maqueen-lite-robot-platform.html]

Δεν έρχεται προκατασκευασμένο.

Η τιμή του ανέρχεται στα 40 ευρώ.

ELEGOO

Εικόνα 1.11 [Η εικόνα αντλήθηκε από τον διαδικτυακό ιστότοπο https://www.elegoo.com/products/elegoo-smart-robot-car-kit-v-4-0]

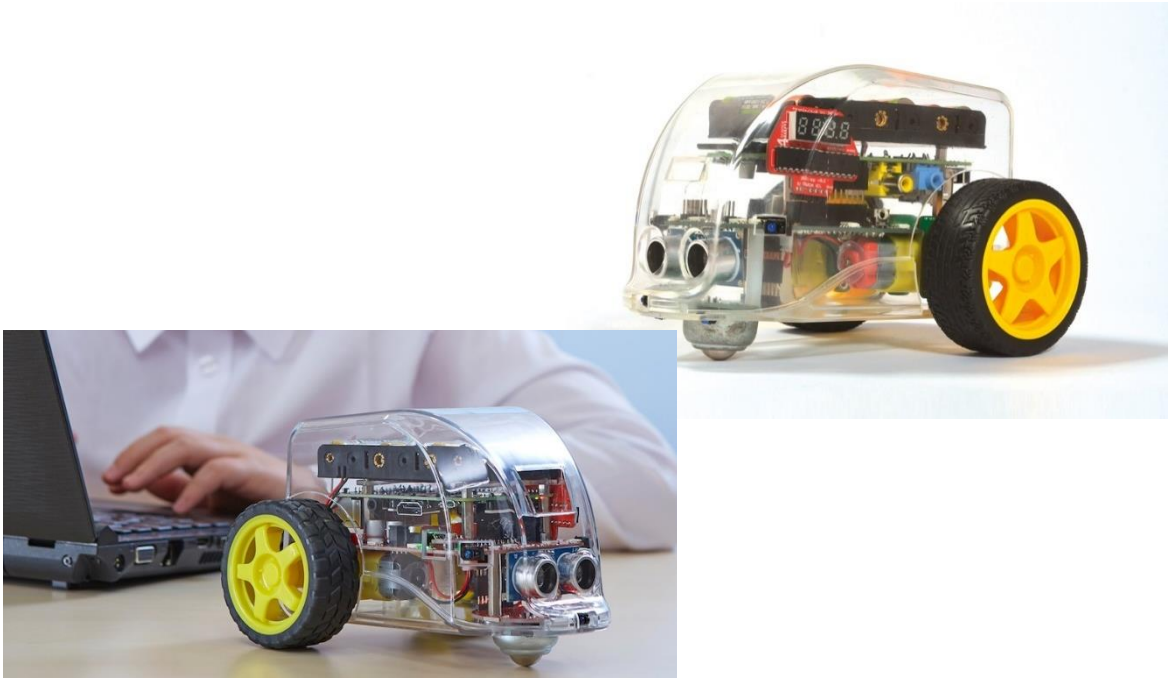
Το Elegoo είναι ένα εκπαιδευτικό κιτ με κάμερα βασισμένο στο ελεγκτή ELEGOO UNO R3 ,συμβατός με τον Arduino. Σχεδιασμένο για ερασιτέχνες και επαγγελματίες ώστε να τους εξοικειώσει με τη ρομποτική, τα ηλεκτρονικά και τη ρομποτική. Περιέχει διάφορους αισθητήρες για αυτόματη κίνηση, έλεγχο υπέρυθρων και αποφυγή εμποδίων.

- FPV Λειτουργία:
  Χρησιμοποιεί WiFi για την μετάδοση του βίντεο και τον έλεγχο των κινήσεων του ρομπότ.
- Λειτουργία αποφυγής εμποδίων:
  Σε αυτή τη λειτουργία το ρομπότ ανιχνεύει αυτόματα τα εμπόδια που βρίσκονται στο δρόμο χάρη στους υπέρυθρους αισθητήρες. Αυτή τη λειτουργία θα τη δούμε αναλυτικά στο ρομπότ της παρούσας εργασίας.
- Line Tracking Λειτουργία
  Σε αυτή τη λειτουργία το ρομπότ ακολουθεί τη γραμμή που του έχουμε σχεδιάσει.

Δεν έρχεται προκατασκευασμένο αλλά η συναρμολόγησή του είναι σχετικά εύκολη διαδικασία.

Το κόστος του ανέρχεται στα 73,80 ευρώ.

Pi2Go Raspberry Pi Programmable Floor Robot



Εικόνα 1.12: [Οι εικόνες αντλήθηκαν από τον διαδικτυακό ιστότοπο https://eduk8.gr/en/product/pi2go-raspberry-pi-programmable-floor-robot/]

Αυτό το ρομπότ είναι συμβατό με διάφορα Raspberry Pi μοντέλα όπως το B και B+. Μπορεί να προγραμματιστεί μέσω Python ή Scratch.

Το κιτ περιλαμβάνει:

- Αισθητήρες εισόδου που επιτρέπουν το χειρισμό μεταβλητών όπως απόσταση και φωτεινότητα.
- Επικοινωνία μέσω Wifi ή απευθείας μέσω USB.
- Προστατευτικό περίβλημα.

Έρχεται προ-κατασκευασμένο και χρειάζεται να το συνδέσετε μόνο με το Raspberry Pi για να λειτουργήσει.

Το Raspberry Pi δεν περιλαμβάνεται στο κιτ και θα πρέπει να προμηθευτεί ξεχωριστά.

Το κόστος του ανέρχεται στα 175 ευρώ.

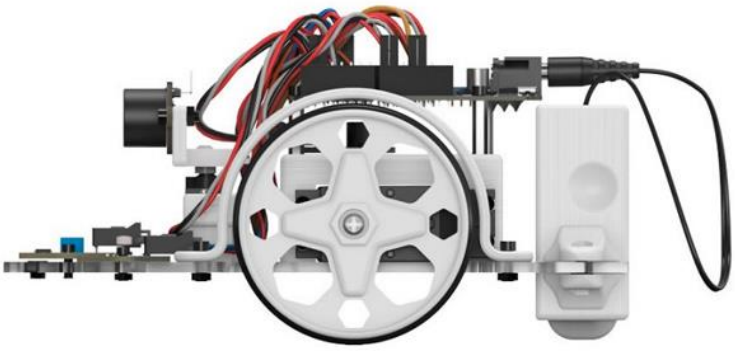
bq PrintBot Evolution

Εικόνα 1.13: [Η εικόνα αντλήθηκε από τον διαδικτυακό ιστότοπο https://www.3dhub.gr/shop/robotics/bq-printbot-evolution/]

Είναι ένα ρομπότ εκτυπωμένο σε 3D εκτυπωτή! Συνδέοντάς το με ένα tablet και την εφαρμογή της BQ, μπορείτε να το ελέγχετε και να το οδηγείτε. Τα ενσωματωμένα ηλεκτρονικά του, δίνουν τη δυνατότητα για διάφορους αυτοματισμούς και τρόπους λειτουργίας του ρομπότ, όπως το να ακολουθεί μια μαύρη γραμμή, να ακολουθεί το φως ή να αποφεύγει εμπόδια.

Το κιτ περιλαμβάνει:

- 3D εκτυπωμένα μέρη.
- Υποστήριξη mini Servo και MiniServo
- Υποστήριξη Ultrasound και αισθητήρα Ultrasound
- ZUM BT-328 πλακέτα ελέγχου
- Υπέρυθρες ZUMbloq



Εικόνα 1.14 Το ρομποτ bq PrintBot Evolution

Δεν έρχεται συναρμολογημένο.

Το κόστος του ανέρχεται στα 99,90 ευρώ.

VEX Super Kit VEX IQ

Εικόνα 1.15: [Η εικόνα αντλήθηκε από τον διαδικτυακό ιστότοπο https://stem-toys.gr/product/vex-super-kit/?lang=en]

Το VEX Super Kit VEX IQ αυξάνει τα στάνταρ των εκπαιδευτικών ρομπότ κατά πολύ! Οι αγοραστές μπορούν να χρησιμοποιήσουν το σετ με τον προσχεδιασμένο κώδικα από την πρώτη στιγμή ή μπορούν να προγραμματίσουν αυτόνομες λειτουργίες χρησιμοποιώντας επιπρόσθετους έξυπνους αισθητήρες.

Περιλαμβάνει πάνω από 800 μηχανικά και κατασκευαστικά μέρη. Η ευρεία ποικιλία των επιπρόσθετων εξαρτημάτων που περιλαμβάνει επιτρέπει την κατασκευή ενός ρομπότ μεγαλύτερου, πιο δυνατού και πιο λειτουργικού

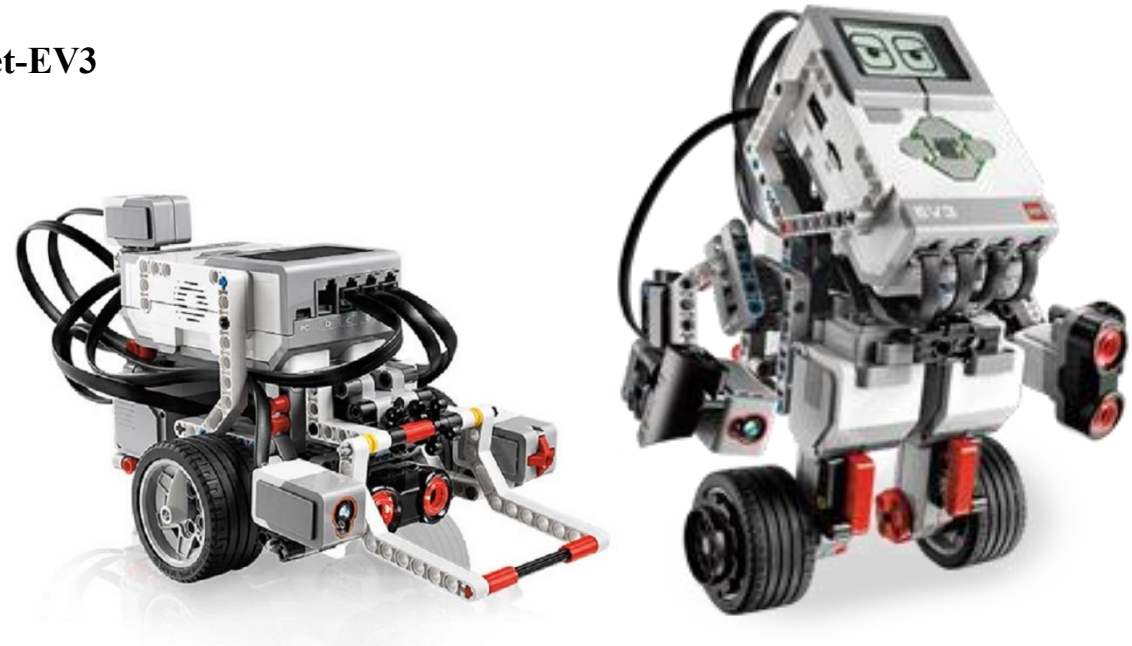
Οι αισθητήρες που εμπεριέχονται στο κιτ είναι:

- Ρομποτικός εγκέφαλος
- VEX IQ controller
- 2.4 GHZ radio
- Smart engine
- Γυροσκοπικός αισθητήρας
- Αισθητήρας αφής
- Αισθητήρας πίεσης
- Αισθητήρας απόστασης
- Αισθητήρας χρωμάτων

Δεν έρχεται συναρμολογημένο.

Το κόστος του ανέρχεται στα 495,88 ευρώ

Core Set-EV3



Εικόνα 1.16 [Η εικόνα αντλήθηκε από τον διαδικτυακό ιστότοπο https://stemcs.com/product/lego-mindstorms-education-ev3-core-set/

Το Core Set-EV3 επιτρέπει τον προγραμματισμό και τη δοκιμή πραγματικών προβλημάτων της ρομποτικής τεχνολογίας. Περιέχει το έξυπνο τούβλο, το οποίο είναι ένας **ισχυρός μικρός υπολογιστής** που κάνει δυνατό τον έλεγχο μοτέρ και την συλλογή δεδομένων από τους αισθητήρες. Επιτρέπει Bluetooth και Wi-Fi επικοινωνία και παρέχει συλλογή δεδομένων και προγραμματισμό.

Το κιτ περιλαμβάνει:

- 3 σερβο-μοτέρ με ενσωματωμένους αισθητήρες περιστροφής
- Αισθητήρα χρώματος, γυροσκόπιο, αισθητήρα υπερήχων (απόστασης) και 2 αισθητήρες αφής
- Τροχό με σφαιρίδιο
- Επαναφορτιζόμενη μπαταρία DC
- Καλώδια σύνδεσης
- Τουβλάκια κατασκευής για την δημιουργία μεγάλου αριθμού μοντέλων

Δεν έρχεται συναρμολογημένο αλλά περιλαμβάνει γραπτές οδηγίες κατασκευής του βασικού μοντέλου και 4 επιπλέον μοντέλων στο λογισμικό.

Είναι συμβατό με τους αισθητήρες Hi-Technic.

Το κόστος του ανέρχεται στα 550,00 ευρώ.

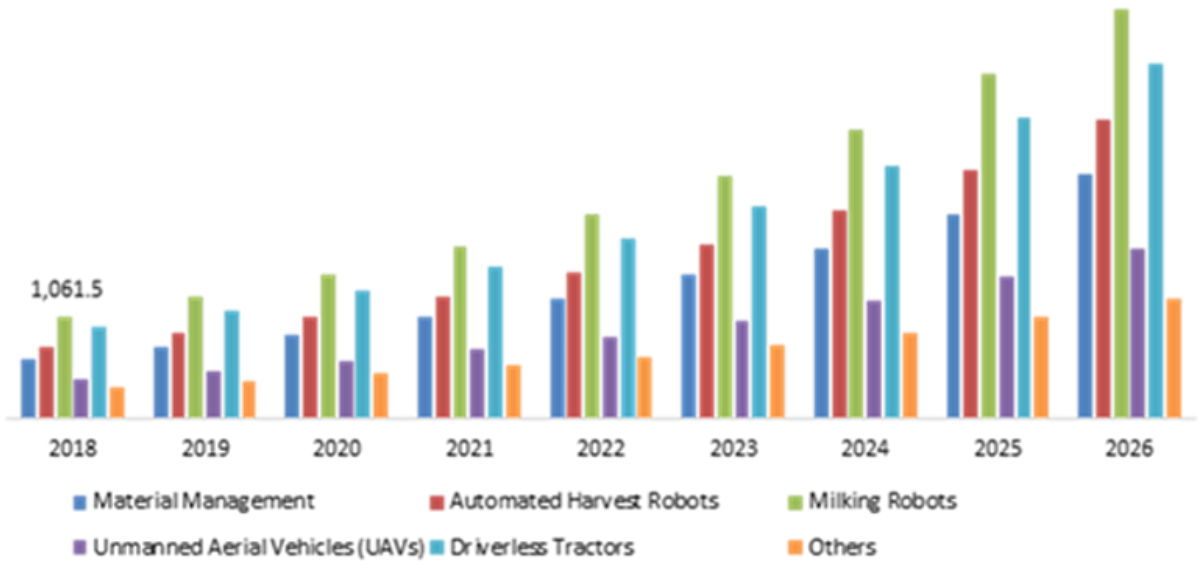
1.4.2 Γενικά συμπεράσματα για την κατάσταση της «Τέχνης»

Όπως διαπιστώνουμε από την παραπάνω παρουσίαση, η κατάλογος των ρομποτικών οχημάτων είναι πολύ μεγάλος και περιλαμβάνει εκατοντάδες άλλα οχήματα. Το εύρος τιμών βρίσκεται μεταξύ 30-600 ευρώ με ρομποτάκια να ξεπερνάνε ακόμα και τα 1000 ευρώ. Οι λειτουργίες που προσφέρουν είναι ποικίλες, αλλά οι αισθητήρες, ιδίως στα πιο οικονομικά ρομποτάκια, παραμένουν οι ίδιοι με μικρές προσθήκες και παραλλαγές.

Εύκολα καταλήγει κάποιος στο συμπέρασμα πως η κατασκευή και ο πειραματισμός με την ρομποτική τεχνολογία είναι πολύ προσιτή και εύκολη. Η αγορά μπορεί να καλύψει τις ανάγκες για κάθε οικονομική δυνατότητα από μαθητές, ερασιτέχνες, μέχρι επαγγελματίες που θέλουν να δημιουργήσουν.

Η ανάπτυξη της ρομποτικής αναμένεται να είναι πολύ μεγάλη τα επόμενα χρόνια καθώς όλο και περισσότερα σχολεία στρέφονται στην κατάρτιση των μαθητών στον παραπάνω τομέα. Η «τέχνη» αυτή γίνεται αναπόσπαστο κομμάτι της καθημερινότητας καθώς κάνει την εμφάνιση της σε οικιακές συσκευές όλο και πιο συχνά, σε ιατρικές εφαρμογές, σε εργοστάσια, στην αυτοκινητιστική βιομηχανία κα.

Παρακάτω βλέπουμε πώς η ρομποτική έχει διεισδύσει στην αγροτική αγορά και πως αναμένεται να αυξηθεί τα επόμενα δύο χρόνια.



Εικόνα 1.17: Η πρόβλεψη της ρομποτικής αγοράς στον αγροτικό τομέα.
[Η εικόνα αντλήθηκε από τον διαδικτυακό ιστότοποhttps://roboticsandautomationnews.com/2021/06/14/global-agriculture-robot-market-expected-to-quadruple-in-size-by-2026/43841/]

2ο ΚΕΦΑΛΑΙΟ: Εργαλεία, Μέθοδοι και Υλικά

Όπως είδαμε στο προηγούμενο κεφάλαιο υπάρχουν πολλές κατηγορίες ρομποτικών οχημάτων. Στην παρούσα πτυχιακή θα ασχοληθούμε με το σχεδιασμό δυο διαφορετικών οχημάτων. Στο πρώτο θα εξετάσουμε την επεκτασιμότητα που μας παρέχει η εκπαιδευτική πλακέτα Arduino Uno σε συνδυασμό με τη γέφυρα L298N και τους αισθητήρες απόστασης και μέτρησης στροφών. Στο δεύτερο θα εξετάσουμε την επεκτασιμότητα που μας προσφέρει και η εκπαιδευτική πλακέτα ESP8266 σε συνδυασμό με τη γέφυρα L298N που θα επιτρέψει την ασύρματη επικοινωνία μέσω τοπικού δικτύου.

2.1 Μέθοδοι οργάνωσης και προγραμματισμού

Για τον σχεδιασμό, τον προγραμματισμό και την οργάνωση της πτυχιακής εργασίας, που αφορά τη δημιουργία δύο ρομποτικών οχημάτων με τη χρήση εκπαιδευτικών πλακετών ακολουθήσαμε τις παρακάτω μεθόδους:

Δημιουργία ενός πλάνου εργασίας που θα περιλαμβάνει τις διάφορες φάσεις του έργου, όπως τη σχεδίαση, τη συναρμολόγηση, τον προγραμματισμό, τα πειράματα και τη συγγραφή της πτυχιακής.

Κατανόηση των απαιτήσεων και των προδιαγραφών των οχημάτων. Αυτά μπορεί να περιλαμβάνουν τους περιορισμούς στην αυτονομία και τις απαιτήσεις ρεύματος ώστε να λειτουργούν αποδοτικά για μεγάλο χρονικό διάστημα.

Επιλογή τύπων μοντέλων κινητήρων. Υπήρχαν δύο κατηγορίες προς επιλογή: Οι κινητήρες servo που λειτουργούν με ειδικό μηχανισμό που ελέγχει τη θέση του άξονα και είναι κατάλληλοι για εφαρμογές που απαιτούν ακριβή έλεγχο θέσεις όπως ρομποτικά χέρια. Η άλλη επιλογή είναι οι DC κινητήρες συνεχούς ρεύματος και περιστροφικής κίνησης, κατάλληλοι για μηχανικές μετακινήσεις. Για τις ανάγκες της πτυχιακής επιλέχτηκε ο 2ος.

Ορισμός της συνδεσμολογίας των εξαρτημάτων, μια δύσκολη διαδικασία, καθώς είναι πολύ εύκολο να χαθεί κάποιος με τον αριθμό των καλωδίων. Θα πρέπει να γίνει σωστή διάταξη τους.

Στη συνέχεια, ακολουθεί η ανάπτυξη κώδικα για τον έλεγχο των εξαρτημάτων. Το λογισμικό που θα χρησιμοποιηθεί και στις δυο κατασκευές είναι το Arduino IDE.

Θα πρέπει να γίνει μελέτη των μεθόδων και βιβλιοθηκών του λογισμικού ώστε να διευκολυνθεί η συγγραφή του κώδικα.

Έπειτα πραγματοποιούμε δοκιμή του κώδικα για να βεβαιωθούμε ότι λειτουργεί σωστά. Διόρθωση τυχόν σφαλμάτων μέσω της χρήσης του Serial Monitor του Arduino IDE.

Τέλος, δημιουργία εφαρμογής για την δεύτερη κατασκευή ώστε να δίνουμε εντολές κίνησης στο όχημα μέσω της φορητής συσκευής μας.

2.2 Υλικά

Για την πρώτη κατασκευή τα υλικά που χρησιμοποιήσαμε είναι: Η πλακέτα Arduino UNO, η γέφυρα L298N, το αποστασιόμετρο HC-SR04, οι οπτικοί αισθητήρες στροφών, το σασί, 2 DC κινητήρες συνεχούς ρεύματος, 5 * 1.2V επαναφορτιζόμενες μπαταρίες, μια 9V μπαταρία και καλώδια.

Για την δεύτερη κατασκευή τα υλικά που χρειαστήκαμε είναι: Η πλακέτα Node Mcu Lua ESP8266, η γέφυρα L298N, 4 * 3.7V επαναφορτιζόμενες μπαταρίες, το σασί, 2 DC κινητήρες και καλώδια.

Παρακάτω θα δούμε αναλυτικά όλα τα υλικά που χρησιμοποιήθηκαν στο σύνολο.

2.2.1 Arduino Uno

Η πλακέτα Arduino Uno είναι μια από τις πιο δημοφιλείς και ευρέως χρησιμοποιούμενες πλακέτες της οικογένειας Arduino. Πρόκειται για μια προγραμματιζόμενη μικροελεγκτική πλακέτα που δημιουργήθηκε με σκοπό να είναι ευέλικτη, εύκολη στη χρήση και προσιτή για ερασιτέχνες, εκπαιδευτικούς και επαγγελματίες μηχανικούς.

Ο χρήστης μπορεί χρησιμοποιήσει την πλακέτα με όποιο τρόπο θέλει, απλά στέλνοντας τις κατάλληλες οδηγίες στον μικροελεγκτή μέσω του Arduino IDE. Ο μικροελεγκτής που χρησιμοποιεί είναι ο ATmega328P, ο οποίος διαθέτει αρκετή μνήμη και ισχύ υπολογισμού για την εκτέλεση ποικίλων εργασιών.

Διαθέτει πολλές ψηφιακές και αναλογικές εισόδους και εξόδους, που επιτρέπουν τη σύνδεση με αισθητήρες, ενεργοποίηση/ απενεργοποίηση συσκευών και την αλληλεπίδραση με το περιβάλλον.

Η πλακέτα διαθέτει USB θύρα για σύνδεση με υπολογιστή, πράγμα που διευκολύνει τον προγραμματισμό της και την επικοινωνία με άλλες συσκευές.

Η Arduino Uno χρησιμοποιεί το Arduino IDE, ένα προγραμματιστικό περιβάλλον που καθιστά εύκολο τον προγραμματισμό της πλακέτας μέσω γραφικής διασύνδεσης.

Τα αριθμημένα pin της πλακέτας Arduino Uno αντιπροσωπεύουν τις επαφές που επιτρέπουν την εισαγωγή και την έξοδο σημάτων από και προς την πλακέτα. Είναι απαραίτητα για τη σύνδεση αισθητήρων, εξόδων, οθονών, κινητήρων και πολλών άλλων συσκευών με την πλακέτα για την αλληλεπίδραση με το περιβάλλον και την εκτέλεση διαφορετικών λειτουργιών.

Εικόνα 2.1: Η πλακέτα Arduino UNO που χρησιμοποιήθηκε στο 1ο όχημα. [Η εικόνα αντλήθηκε από τον ιστότοπο https://www.cleanpng.com/png-arduino-uno-atmega328-single-board-microcontroller-2899186/]

Οι βασικές κατηγορίες pin στην πλακέτα Arduino Uno είναι:

1.Ψηφιακές είσοδοι/έξοδοι: Αυτές οι επαφές μπορούν να ρυθμιστούν ως είσοδοι (για την ανάγνωση ψηφιακών σημάτων, όπως διακόπτες) ή ως έξοδοι (για την αποστολή ψηφιακών σημάτων, όπως τον έλεγχο των LED).

2.Αναλογικές είσοδοι/έξοδοι: Οι επαφές αυτές επιτρέπουν την ανάγνωση αναλογικών τιμών, όπως αυξομείωση τάσης από αισθητήρες (π.χ., αισθητήρες φωτός, θερμοκρασίας).

3.Ειδικές Επαφές: Στην πλακέτα Arduino Uno, υπάρχουν επίσης ειδικές επαφές, όπως οι επαφές τροφοδοσίας (5V και GND) και επαφές ειδικών λειτουργιών (όπως οι επαφές του διακόπτη επαναφοράς - Reset).

4.Επαφές Επικοινωνίας: Στην πλακέτα περιλαμβάνονται επίσης επαφές για σύνδεση με τον υπολογιστή μέσω USB και επαφές για εξωτερικές συσκευές επικοινωνίας, όπως τηλεμετρία ή ασύρματη επικοινωνία.

2.2.2 DC Κινητήρες με γρανάζωμα υποπολλαπλασιασμού στροφών

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

Τα χαρακτηριστικά τους είναι τα εξής:

Τάση λειτουργίας: 3V - 12VDC (προτεινόμενη 6 - 8V)

Μέγιστη ροπή: 0.8 KgrXcm

Σχέση γραναζιών: 1:48

Ταχύτητα περιστροφής: μέγιστη 170 rpm

Ρεύμα: τυπικό 70mA (μέγιστο 250mA)

Μέγεθος: 7x2.2x1.8cm

Εικόνα 2.2: Ένας κινητήρας DC συνεχούς ρεύματος. Χρησιμοποιήθηκε και στις δυο κατασκευές

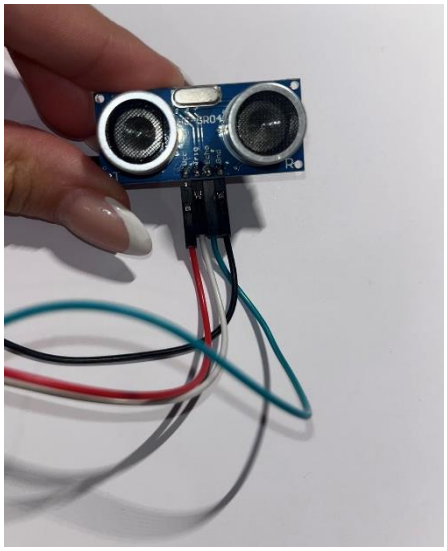
2.2.3 Γέφυρα L298N

Η γέφυρα L298N είναι ένα ολοκληρωμένο κύκλωμα που χρησιμοποιείται ευρέως για τον έλεγχο κινητήρων σε ποικίλες εφαρμογές, όπως ρομποτικά οχήματα, συστήματα αυτοματισμού και άλλες συσκευές που απαιτούν την κίνηση μηχανικών εξαρτημάτων. Η πλακέτα με το IC L298N έχει τις εξής συνδέσεις:

Εικόνα 2.3: Η γέφυρα που χρησιμοποιήσαμε και στις δυο κατασκευές

1.DC Κινητήρας Α παροχή ρεύματος +
2.DC Κινητήρας Α παροχή ρεύματος -
3.Τροφοδοσία 5V
4. Γείωση
5. Σύνδεση της τάσης τροφοδοσίας των κινητήρων με μέγιστο τα 35V.
6. DC Κινητήρας Β παροχή ρεύματος -
7 πάνω. DC Κινητήρας Β παροχή ρεύματος -
7 κάτω. Σύνδεση της PWM εξόδου για έλεγχο της ταχύτητας του DC κινητήρα Α
8. IN1: Είσοδος 1 η οποία καθορίζει τη φορά κίνησης του κινητήρα Α
9. IN2: Είσοδος 2 η οποία καθορίζει τη φορά κίνησης του κινητήρα Α
10. IN3: Είσοδος 3 η οποία καθορίζει τη φορά κίνησης του κινητήρα Β
11. IN4: Είσοδος 4 η οποία καθορίζει τη φορά κίνησης του κινητήρα Β
12. Σύνδεση της PWM εξόδου για έλεγχο της ταχύτητας του DC κινητήρα Β

2.2.4 Αποστασιόμετρο HC-SR04



Εικόνα 2.4.1: Ο αισθητήρας υπερύθρων για μέτρηση αποστάσεων από εμπόδια

Εικόνα 2.4.2: Τα pin του αισθητήρα υπερύθρων

Ο αισθητήρας υπέρυθρων HC-SR04 από 2εκ. εως 400εκ. ανέπαφες μετρήσεις με ακρίβεια 3 χιλιοστών. Η λειτουργία του αισθητήρα αυτού βασίζεται στην αρχή «time of flight». Η τεχνική αυτή συνίσταται στην αποστολή από τον αισθητήρα προς το αντικείμενο μίας δέσμης ηλεκτρομαγνητικού ή άλλου κύματος (π.χ. υπερήχων), την ανάκλαση της δέσμης από το αντικείμενο, την λήψη της δέσμης από τον αισθητήρα και την μέτρηση του χρόνου που χρειάστηκε για να «ταξιδέψει» η δέσμη από τον αισθητήρα προς το αντικείμενο και ξανά πίσω. Ο αισθητήρας HC-SR04 χρησιμοποιεί

έναν πομπό και έναν δέκτη υπερήχων (δονήσεις αέρα με συχνότητα > 20KHz). Ο πομπός στέλνει μία δέσμη υπερήχων προς το αντικείμενο π.χ. τοίχος. Η δέσμη αυτή ανακλάται στο αντικείμενο και επιστρέφει πίσω όπου λαμβάνεται από τον δέκτη υπερήχων. Ο αισθητήρας μετράει τον χρόνο που χρειάστηκε η δέσμη υπερήχων για να ταξιδέψει έως το αντικείμενο και πίσω και βάσει αυτού υπολογίζεται η απόσταση του αντικειμένου.

Ο αισθητήρας αυτός έχει 4 pin που είναι:

VCC: Σύνδεση της τάσης τροφοδοσίας που είναι τυπικά 5V

Trig: Στο pin αυτό παρέχουμε σύντομο παλμό HIGH διάρκειας τυπικά 10μsec για να παραχθεί ο υπερηχητικός παλμός και να ξεκινήσει η μέτρηση χρόνου επιστροφής (και άρα απόστασης από το αντικείμενο που ανακλά την δέσμη)

Echo: Στο pin αυτό το HC-SR04 παράγει έναν θετικό παλμό (HIGH) από την στιγμή αποστολής του υπερηχητικού παλμού έως και την λήψη της ανάκλασης, οπότε το pin αυτό γίνεται LOW. Με μέτρηση της διάρκειας αυτού του παλμού μπορούμε να υπολογίσουμε τον συνολικό χρόνο αποστολής, ανάκλασης και λήψης του υπερηχητικού παλμού. Επειδή ο υπερηχητικός παλμός ταξιδεύει με την ταχύτητα του ήχου (που είναι 340 m/s ή 29 microseconds ανά centimeter) μπορούμε να υπολογίσουμε την απόσταση του εμποδίου ως εξής: Απόσταση = χρόνος παλμού X ταχύτητα ήχου / 2. Η διαίρεση δια 2 επιβάλλεται επειδή ο χρόνος παλμού είναι συνολικός, δηλαδή χρόνος αποστολής + χρόνος επιστροφής.

GND: Η γείωση για την τροφοδοσία του αισθητήρα.

Η ανάγνωση της διάρκειας του παλμού στο pin Echo μπορεί να γίνει με την συνάρτηση pulseIn() που συντάσσεται ως εξής:

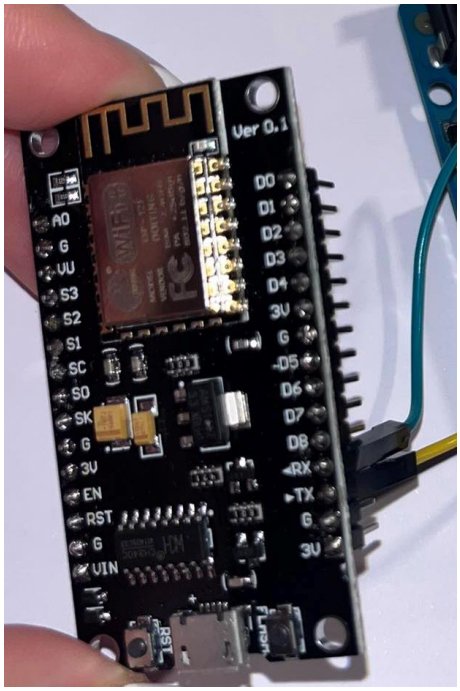
duration = pulseIn (echoPin , HIGH ή LOW); ή

duration = pulseIn (echoPin , HIGH ή LOW, περίοδος timeout);

Η συνάρτηση επιστρέφει τον χρόνο σε μsec της διάρκειας ενός παλμού HIGH ή LOW που διαβάζεται στο pin «echoPin». Το echoPin είναι η ψηφιακή είσοδος του Arduino όπου θα γίνει η μέτρηση της διάρκειας του παλμού σε μsec. Η τιμή HIGH ή LOW καθορίζει αν ο παλμός που θα μετρηθεί είναι HIGH ή LOW αντίστοιχα. Για παράδειγμα αν η τιμή αυτή είναι HIGH τότε η είσοδος αυτή είναι κανονικά σε κατάσταση LOW και η ρουτίνα περιμένει έως ότου γίνει HIGH. Από την στιγμή εκείνη αρχίζει να μετράει τον χρόνο και σταματάει όταν ο παλμός ξαναγίνει LOW.

2.2.5 Node ESP8266

Με αυτό το WiFimodule θα πραγματοποιηθεί η ασύρματη επικοινωνία του οχήματος με την εφαρμογή μας στη 2η κατασκευή.
Υποστηρίζει TCP/IP στιβαρές συνδέσεις και μπορεί να λειτουργήσει ως εξυπηρετητής (server) ή ως πελάτης (client) σε ασύρματη επικοινωνία.
Το ESP8266 μπορεί να προγραμματιστεί χρησιμοποιώντας το Arduino IDE με τη χρήση των κατάλληλων εντολών. Στη συνέχεια μέσω της βιβλιοθήκης της εφαρμογής Blynk μπορούμε να πραγματοποιήσουμε την επικοινωνία του τσιπ με το κινητό μας. Ο κώδικας θα εξεταστεί στα επόμενα κεφάλαια.



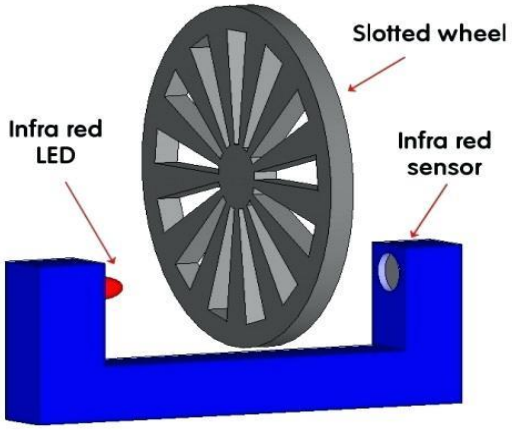
Εικόνα 2.5: Η πλακέτα ESP8266 για την ασύρματη επικοινωνία

Αυτός ο ελεγκτής καθιστά τον προγραμματισμό και την ανάπτυξη εφαρμογών εξαιρετικά εύκολη.

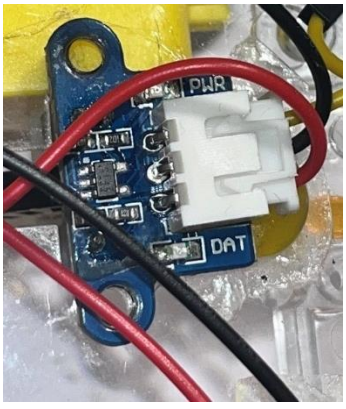
Τα pins D0 έως D8 είναι ψηφιακά pins εισόδου εξόδου και είναι αυτά που μας ενδιαφέρουν στη συγκεκριμένη κατασκευή.

2.2.6 Οπτικός αισθητήρας φωτός

Χάρη σε αυτόν τον αισθητήρα θα μπορούμε να μετράμε την πραγματική ταχύτητα των τροχών. Μεταξύ των εγκοπών του αισθητήρα βρίσκεται ένα LED υπερύθρων και ένας receiver. Όσο περιστρέφεται ο τροχός με τις εγκοπές, ο αισθητήρας μετράει πόσα «ανοίγματα» έχουν περάσει και έτσι υπολογίζει πόσες στροφές πραγματοποίησε ο τροχός.



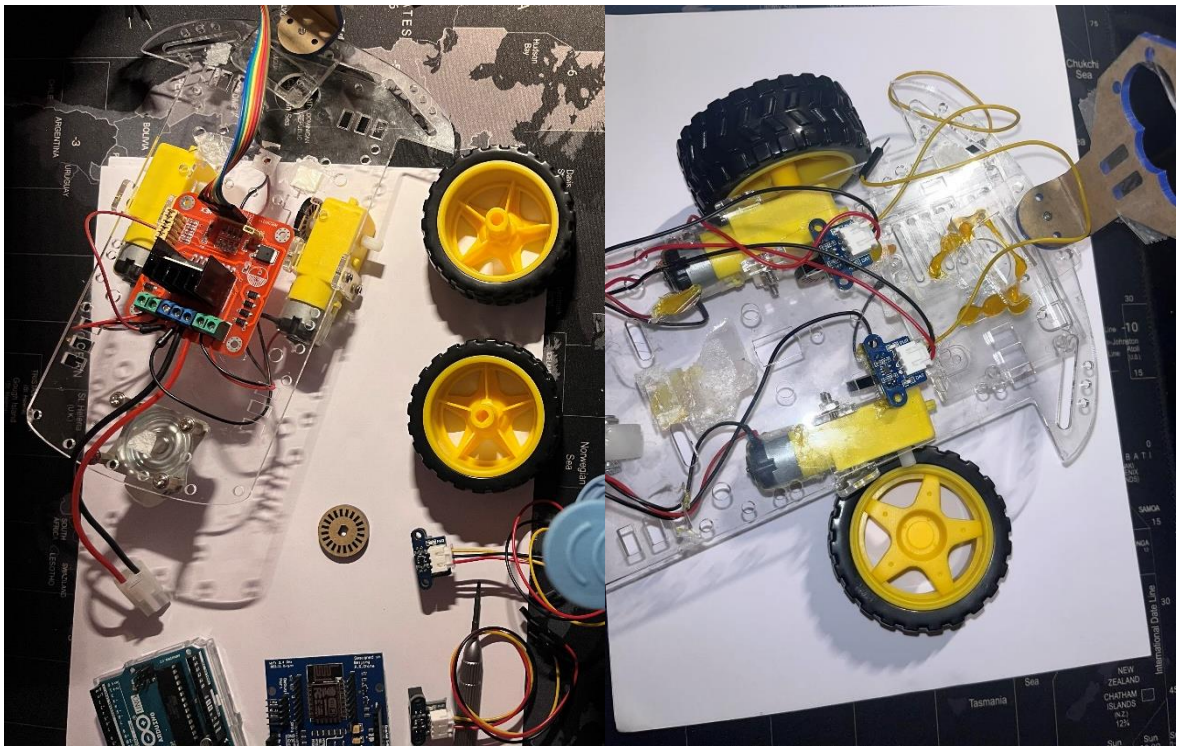
Εικόνα 2.6.1: Περιγραφή του τρόπου λειτουργίας του οπτικού αισθητήρα στροφών. [Η εικόνα αντλήθηκε από τον ιστότοπο https://grobotronics.com/waveshare-photo-interrupter-sensor.html



Εικόνα 2.6.2 Το ηλεκτρονικό εξάρτημα του αισθητήρα στροφών.

2.2.7 Σασί

Μια απλή μηχανική κατασκευή για να τοποθετηθούν τα εξαρτήματα.



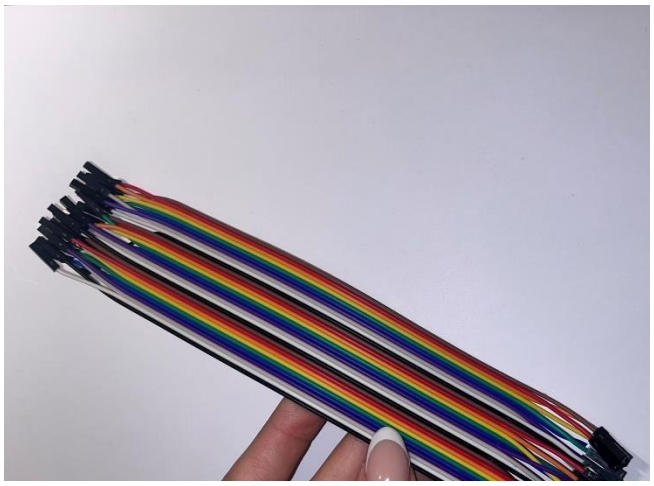
Εικόνα 2.7: Το σασί που θα χρησιμοποιηθεί στις 2 κατασκευές

2.2.8 Επαναφορτιζόμενες μπαταρίες

Όπως είδαμε παραπάνω, η γέφυρα L298N μπορεί να τροφοδοτηθεί με μέγιστο 35V.

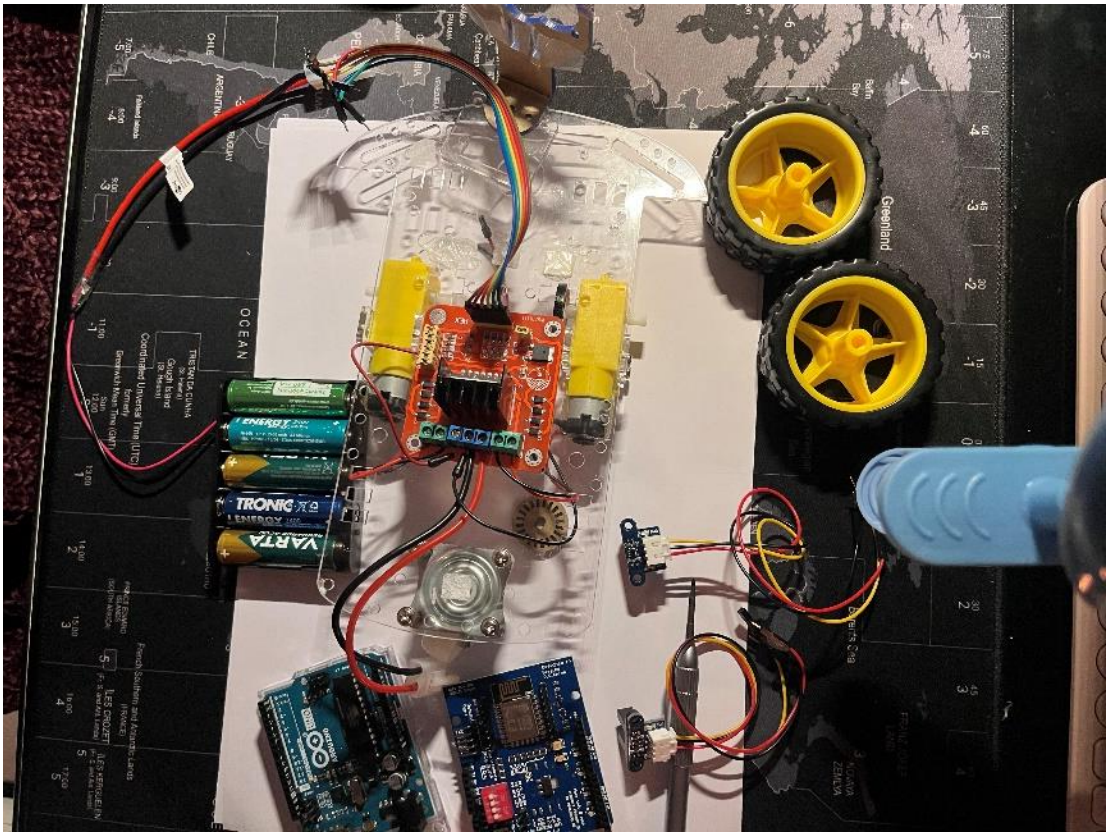
Για την πρώτη κατασκευή θα χρειαστούμε 5*1.3V επαναφορτιζόμενες μπαταρίες και μια των 9V. Για την δεύτερη κατασκευή θα χρειαστούμε 4*3.7V επαναφορτιζόμενες μπαταρίες.

2.2.9 Καλώδια



Εικόνα 2.8: Καλώδια που χρησιμοποιήθηκαν στις 2 κατασκευές

Παρακάτω βλέπουμε τη συγκέντρωση όλων των υλικών για το πρώτο όχημα :



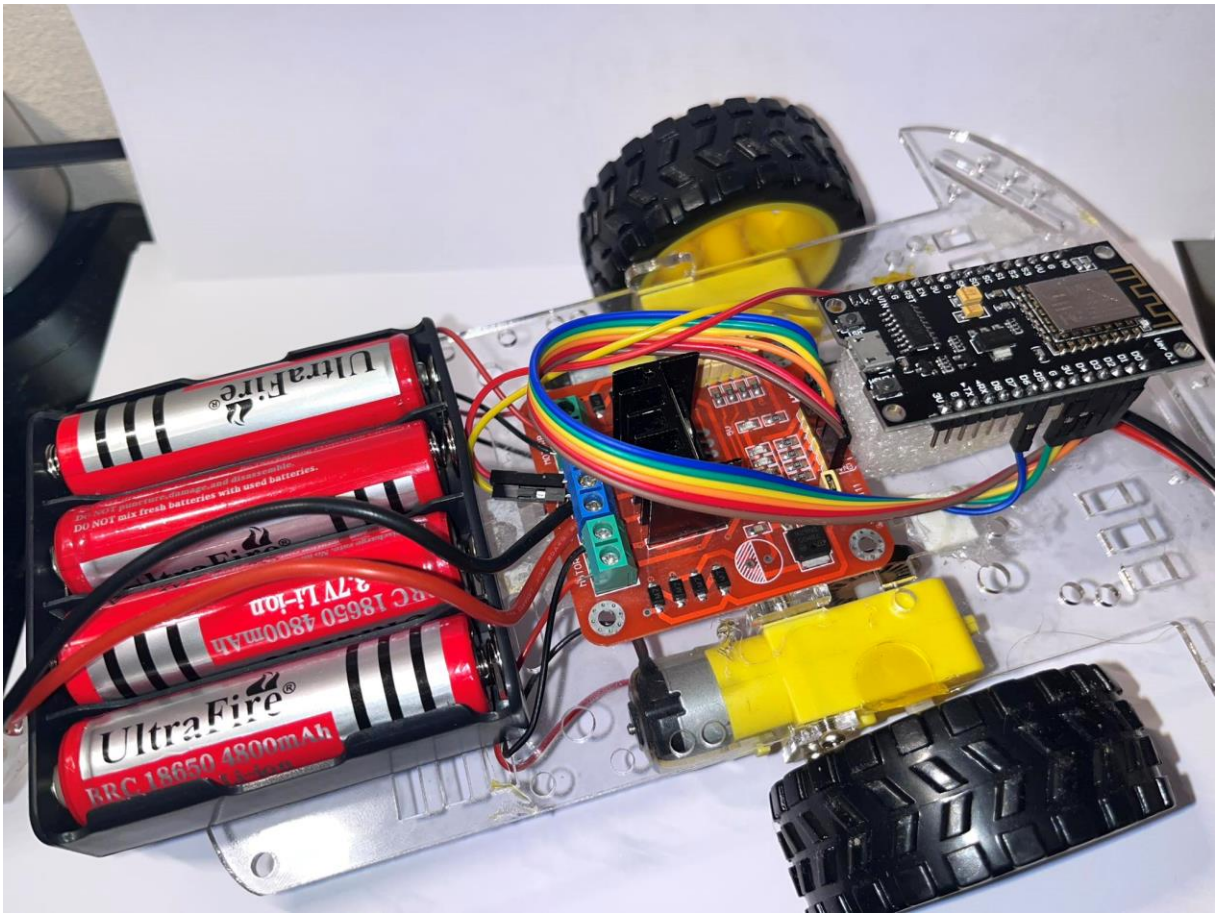
Εικόνα 2.9: Τα υλικά της πρώτης κατασκευής

2.2.10 Κοστολόγιο πρώτης κατασκευής

Εξάρτημα	**Κόστος σε ευρώ**
Πλακέτα Arduino Uno	46.95
DC κινητήρες και σασί	12.90
Γέφυρα L298N	4.20
Αποστασιόμετρο HC-SR04	2.50
2 Photo Interrupter Sensors	8.80
2 plastic Encoder Wheels – 24mm	0.80
5 επαναφορτιζόμενες μπαταρίες 1.2 V	10
1 μπαταρία 9 V	5
Καλώδια	7.20
Θήκη μπαταριών	0.80
Σύνολο	**99.15**

Πίνακας 1. Αναλυτικό Κοστολόγιο της 1ης κατασκευής

Παρακάτω βλέπουμε τη συγκέντρωση των υλικών του δεύτερου οχήματος:



Εικόνα 2.10: Τα υλικά της 2ης κατασκευής

2.2.11 Κοστολόγιο της δεύτερης κατασκευής

Εξάρτημα	**Κόστος σε ευρώ**
DC κινητήρες και σασί	12,90
Γέφυρα L298N	4,20
ESP8266 WiFi Module	5,80
4 επαναφορτιζόμενες μπαταρίες	30
Καλώδια	7,20
Θήκη μπαταριών	2
Σύνολο	**62,1**

Πίνακας 2: Αναλυτικό Κοστολόγιο του 2ου συστήματος

2.3 Εργαλεία

2.3.1 Περιβάλλον Προγραμματισμού Arduino

Το λογισμικό Arduino IDE θα χρησιμοποιηθεί τόσο για τον προγραμματισμό της πλατφόρμας Arduino UNO της πρώτης κατασκευής, όσο και για τον προγραμματισμό της επικοινωνίας του ESP8266 με την εφαρμογή Blynk μέσω τοπικού δικτύου της 2ης κατασκευής.

Εικόνα 2.11: [Η εικόνα αντλήθηκε από τον ιστότοπο https://www.arduino.cc/en/software]

Το Arduino IDE παρέχει έναν απλό επεξεργαστή κώδικα στον οποίο οι χρήστες μπορούν να γράψουν, να τροποποιήσουν ή να οργανώσουν τον κώδικά τους. Υποστηρίζει τη γλώσσα προγραμματισμού του Arduino, ένα υποσύνολο της C και C++. Το IDE παρέχει διάφορα χαρακτηριστικά, όπως, αυτόματη συμπλήρωση, προειδοποίηση λαθών και προτάσεις για αντικατάσταση κώδικα, όλα αυτά για να μπορεί ο προγραμματιστής να έχει έναν καθαρό και χωρίς λάθη κώδικα.

Παρέχει βιβλιοθήκες και προκατασκευασμένες μεθόδους η οποίες εξοικονομούν χρόνο και προσπάθεια, απλοποιώντας τη διαδικασία ενσωμάτωσης περίπλοκων λειτουργιών στα Arduino projects.

Είναι ένα θεμελιώδες εργαλείο προγραμματισμού, εντελώς δωρεάν και πολύ εύκολο στην εγκατάστασή του.

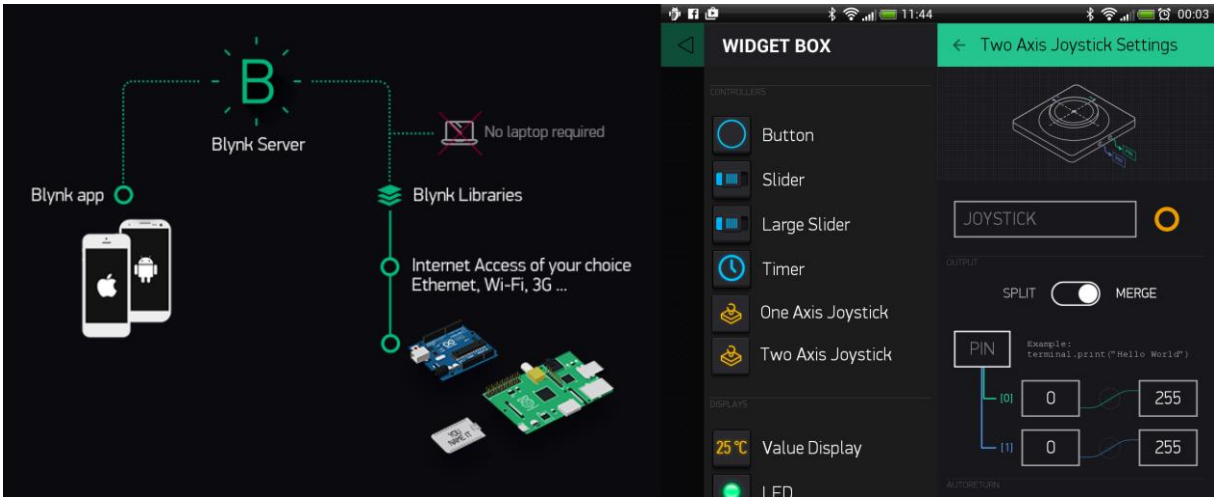
Το Arduino IDE περιλαμβάνει ένα "Serial Monitor" που επιτρέπει την ανάγνωση και αποστολή δεδομένων σε πραγματικό χρόνο μέσω της θύρας

σειριακής επικοινωνίας, βοηθώντας στην αντιμετώπιση σφαλμάτων και την αποσφαλμάτωση του κώδικα.

Το Arduino IDE είναι ελεύθερο λογισμικό και είναι διαθέσιμο για Windows, macOS και Linux, καθιστώντας το ευρέως προσβάσιμο για όλους όσους θέλουν να ασχοληθούν με τον προγραμματισμό των πλακετών Arduino.

2.3.2 Εφαρμογή Blynk

Η εφαρμογή αυτή θα χρησιμοποιηθεί αποκλειστικά για το σχεδιασμό λογισμικού του 2ου οχήματος και θα μας εξυπηρετήσει στην ασύρματη επικοινωνία.



Εικόνα 2.12: Σχεδιάγραμμα επικοινωνίας της εφαρμογής με τα ηλεκτρονικά εξαρτήματα
[Η εικόνα αντλήθηκε από τον ιστότοπο https://hackaday.com/2016/03/10/app-control-with-ease-using-blynk/]

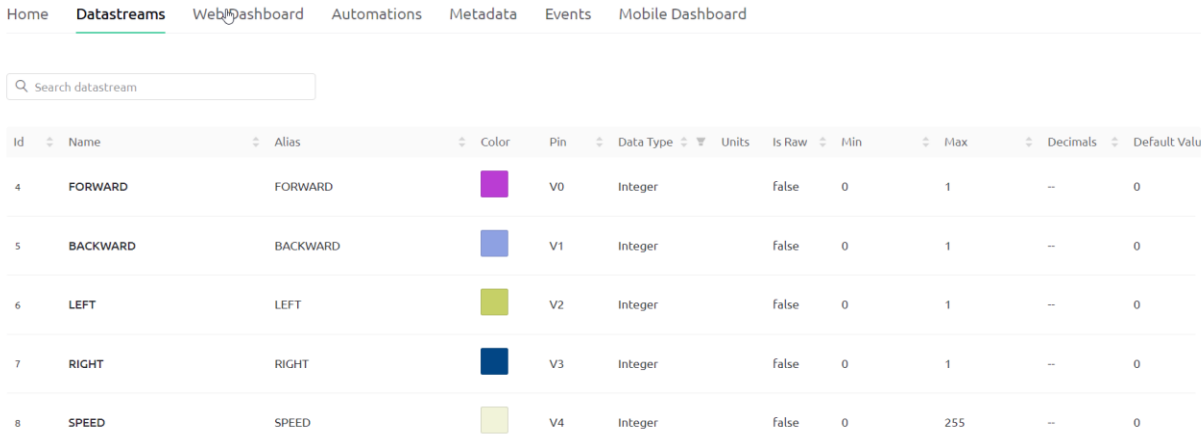
Αυτό το εργαλείο επιτρέπει σε χιλιάδες χρήστες τη δημιουργία εφαρμογών ελέγχου και παρακολούθησης συσκευών μέσω φορητών συσκευών όπως κινητά και τάμπλετ. Η εφαρμογή είναι συμβατή τόσο με Android όσο και με iOS.

Υποστηρίζει διάφορες πλατφόρμες όπως Arduino, Raspberry Pi, ESP8266, Particle, και πολλές άλλες. Οι χρήστες μπορούν να συνδέσουν συσκευές IoT(Internet of Things) από διάφορους κατασκευαστές. Δεν απαιτεί πολύπλοκο προγραμματισμό. Αρκεί να ξέρει κάποιος πως να κάνει τις προ-απαιτούμενες συνδεσμολογίες στο κύκλωμα του. Έπειτα κατεβάζουμε την εφαρμογή στο κινητό μας και ταυτόχρονα συνδεόμαστε στην διαδικτυακή της έκδοση για τον πιο εύκολο σχεδιασμό της εφαρμογής που θα κατασκευάσουμε.

Αφού συνδεθούμε στην διαδικτυακή εφαρμογή από την μηχανή αναζήτησής μας, επιλέγουμε από το μενού Templates -> New Template.

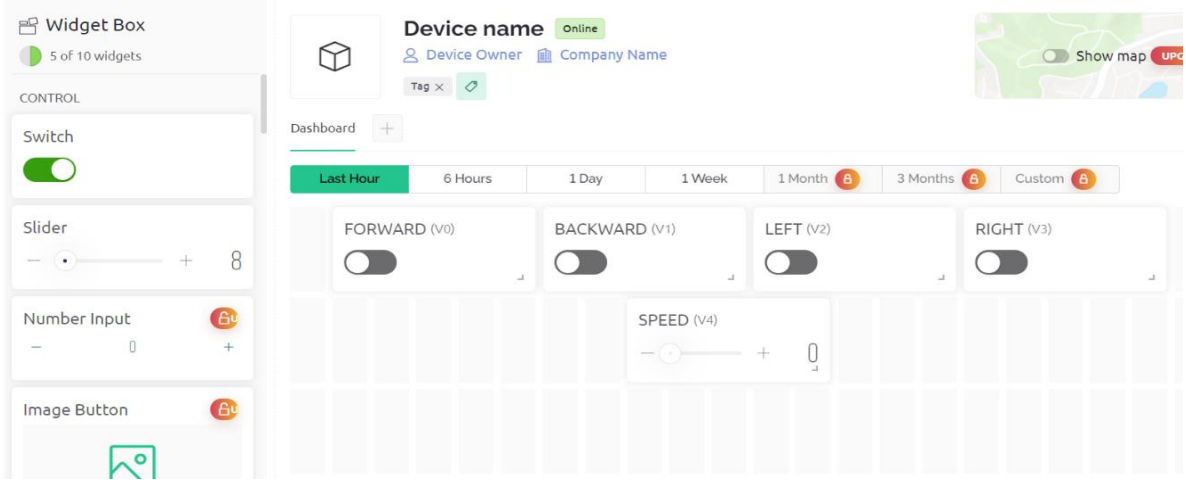
Στο νέο template θα δώσουμε το Όνομα: Arduino Car και στο Υλικό θα επιλέξουμε ESP8266. Έπειτα πατάμε Create και μεταφερόμαστε στο μενού Datastreams-> New Datastream-> Virtual Pin. Στο νέο εικονικό pin θα δώσουμε το όνομα MoveForward και θα το αντιστοιχήσουμε στο pin V0. Η ίδια διαδικασία θα επαναληφθεί με τα εξής δεδομένα:

- MoveBackward -> V1
- TurnLeft -> V2
- TurnRight -> V3
- Velocity -> V4 (0 εως 255)



Εικόνα 2.13: Στιγμιότυπο από την εφαρμογή Blynk κατά τη δημιουργία των εικονικών pins

Στη συνέχεια, θα επιλέξουμε το μενού Web Dashboard και θα προσθέσουμε 4 διακόπτες και ένα Slider όπως την εικόνα παρακάτω. Θα πρέπει να αντιστοιχήσουμε κάθε ένα αντικείμενο με ένα από τα datastreams που δημιουργήσαμε προηγουμένως.



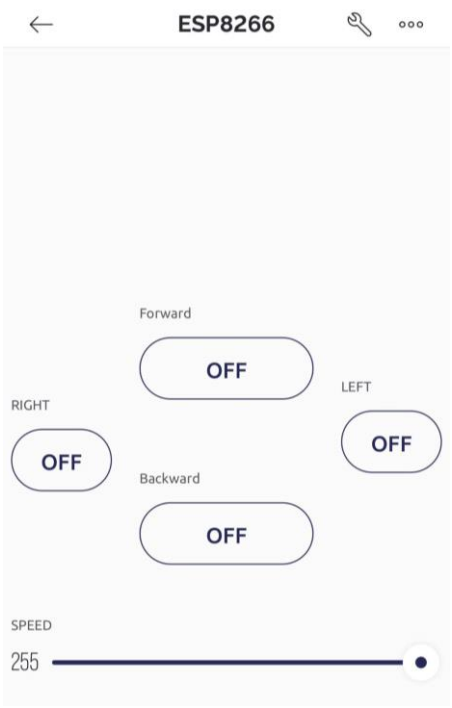
Εικόνα 2.14: Στιγμιότυπο από την εφαρμογή Blynk κατά τη δημιουργία του Interface της εφαρμογής

Τέλος, θα συνδέσουμε τη συσκευή μας με την εφαρμογή. Αυτό το βήμα μπορεί να πραγματοποιηθεί με διάφορους τρόπους όπως με αναγνωριστικό QR , χειροκίνητα, ή μέσω του template που δημιουργήσαμε παραπάνω. Θα ακολουθήσουμε την 3η μεθοδολογία επιλέγοντας “Create from template”. Έπειτα θα μας εμφανιστεί ένα μήνυμα με τον παρακάτω κώδικα :

```
#define BLYNK_TEMPLATE_ID "TMPL4VcpINbQa"

#define BLYNK_TEMPLATE_NAME "Arduino Car"

#define BLYNK_AUTH_TOKEN "mwPLN7lIfGkg2Y0HD4odZ6lObFKHG8-v"
```

Αυτά τα αναγνωριστικά είναι μοναδικά για τον καθένα και για κάθε tamplate που δημιουργούμε. Θα τα χρησιμοποιήσουμε στον κώδικα του Arduino IDE για την επικοινωνία του Blynk με το ESP8266 όπως θα δούμε στο κεφάλαιο 3.Προς το παρόν τα αποθηκεύουμε σε ένα πρόχειρο.

Τώρα μπορούμε να συνδεθούμε στην εφαρμογή Blynk που εγκαταστήσαμε στο κινητό μας και θα δούμε την ταμπλέτα που δημιουργήσαμε προηγουμένως στην αρχική σελίδα. Θα πατήσουμε πάνω και θα φτιάξουμε ένα user interface όπως αυτό που ακολουθεί:



Εικόνα 2.15 Σχεδίαση της εφαρμογής στο κινητό

Αφού προσθέσουμε τα απαραίτητα κουμπιά για τον έλεγχο των κινήσεων των τροχών, θα αντιστοιχήσουμε αυτά τα κουμπιά στα εικονικά pins που δημιουργήσαμε στο προηγούμενο στάδιο.

Πλέον είμαστε έτοιμοι να γράψουμε τον κώδικα της εφαρμογής όπως θα αναλύσουμε στο επόμενο κεφάλαιο.

3ο ΚΕΦΑΛΑΙΟ Ανάλυση κώδικα και συνδεσμολογίας 1ης κατασκευής

Η κατασκευή και συνδεσμολογία του πρώτου ρομποτικού οχήματος θα εξεταστεί βήμα βήμα στις επόμενες ενότητες και έπειτα θα δούμε το τελικό σύστημα συναρμολογημένο μαζί με τον κώδικα για τον προγραμματισμό του.

3.1 Σύνδεση γέφυρας με κινητήρες

Για να θέσουμε σε κίνηση τους τροχούς του οχήματος θα πρέπει αρχικά να υλοποιήσουμε τις κατάλληλες συνδέσεις τους με την γέφυρα WB291111. Κάθε ένας από τους δύο κινητήρες έχει δυο καλώδια, τα οποία ανάλογα με το ρεύμα που παρέχουμε, θα κινούν τους τροχούς μπροστά ή πίσω. Τα καλώδια που προέρχονται από τον πρώτο κινητήρα (motorA) θα τα συνδέσουμε στα pins 1 και 2 της εικόνας 3.1 ενώ, τα καλώδια του δεύτερου κινητήρα (motorB) θα τα συνδέσουμε στα pins 6 και 7.
Τα pins 3 και 4 είναι υπεύθυνα για την τροφοδοσία της γέφυρας (6V και Ground) ενώ το pin 5 είναι ικανό να τροφοδοτήσει την πλακέτα Arduino με τον τρόπο που θα δούμε παρακάτω.
Η τροφοδοσία της γέφυρας γίνεται με 5 επαναφορτιζόμενες μπαταρίες 1.2V σε ειδική θήκη μπαταριών την οποία θα συνδέσουμε στο pin 4 (GND) και στο pin 3(VMS). Αν οι μπαταρίες δεν είναι πλήρως φορτισμένες και δίνουν κάτω από 6 V, το σύστημα θα υπολειτουργεί και το pin 5 δεν θα μπορεί παρέχει αρκετή τάση για τη πλακέτα Arduino ή κάποιον αισθητήρα που πιθανώς να συνδέσουμε.
Αυτή, λοιπόν, είναι η συνδεσμολογία για τη σύνδεση των δυο κινητήρων με τη γέφυρα WB291111.

Εικόνα 3.1: Η Γέφυρα WB291111

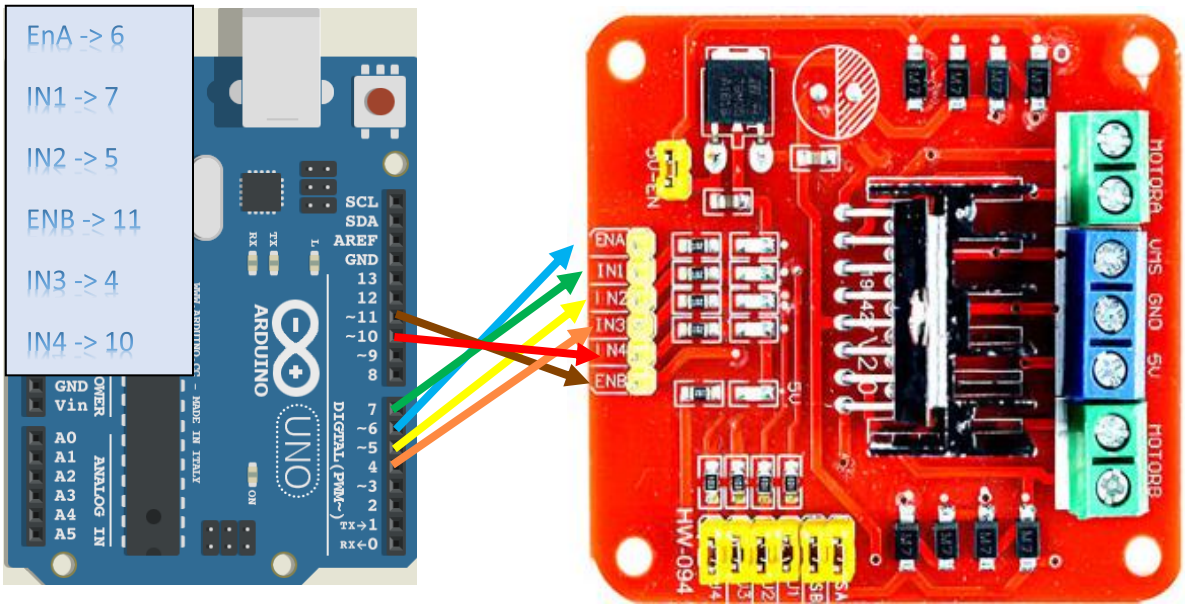
3.2 Σύνδεση Arduino με τη γέφυρα

Υπάρχουν δυο περιπτώσεις τροφοδοσίας του Arduino.

Η πρώτη είναι η τροφοδοσία του μέσω της γέφυρας WB 291111.Συνδέουμε την τροφοδοσία 5V της γέφυρας (pin 5) στο pin Vin του Arduino, αλλά προσοχή, το συγκεκριμένο pin του Arduino δεν έχει ρυθμιστή τάσης και έτσι θα πρέπει να διασφαλίσουμε ότι η τάση που διέρχεται από εκεί είναι σταθερή και όχι μεγαλύτερη των 5V. Από τη στιγμή που η έξοδος 5V της γέφυρας (pin 5) είναι ακριβώς 5V, δεν θα έχουμε κανένα πρόβλημα.

Η δεύτερη περίπτωση, την οποία και εφαρμόσαμε στην κατασκευή του οχήματος, είναι η σύνδεση μιας 9V μπαταρίας με την είσοδο DC της πλακέτας Arduino. Με αυτό τον τρόπο δεν θα επιβαρυνθεί η γέφυρα προσφέροντας ρεύμα σε μια απαιτητική πλακέτα και ο Arduino θα είναι πιο αποδοτικός.

Συνεπώς, η τροφοδοσία του Arduino δεν χρειάζεται να γίνει μέσω υπολογιστή, αρα το σύστημά μας αποκτά αυτονομία και μπορεί να κινείται ελεύθερα στο χώρο. Τις εντολές για τις κινήσεις και τις ταχύτητες των κινητήρων θα αναλάβουν να μεταφέρουν τα pins 8-13 της γέφυρας που θα συνδεθούν με τα pins του Arduino όπως παρακάτω:



Εικόνα 3.2: Συνδεσμολογία της γέφυρας με το Arduino UNO

ΠΡΟΣΟΧΗ! Θα πρέπει να φροντίσουμε η γέφυρα και ο Arduino να έχουν κοινή γείωση! Άρα, συνδέουμε το GND της γέφυρας με το GND του Arduino και το GND όλων των μπαταριών.

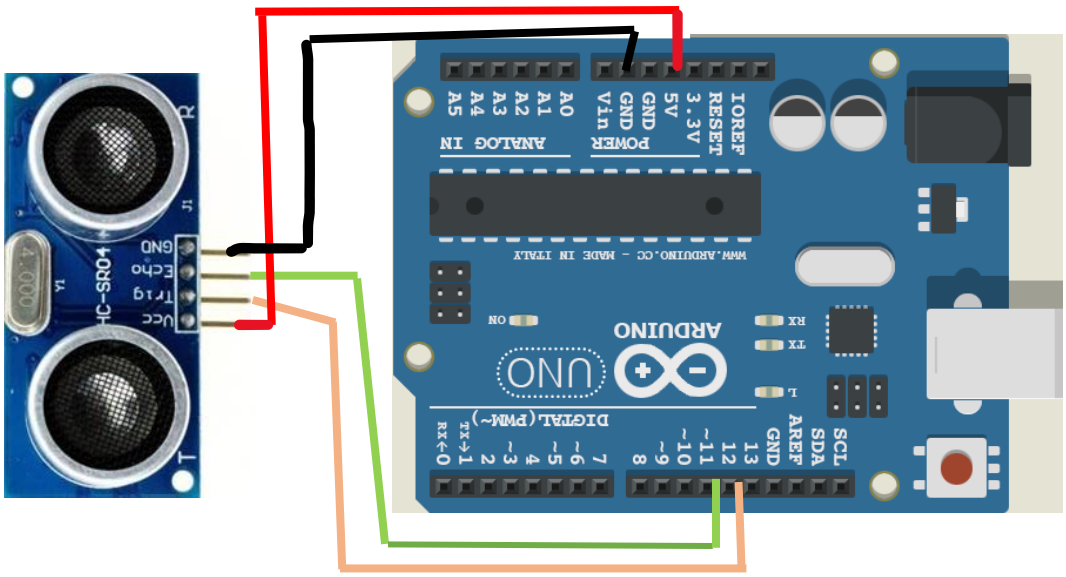
3.3 Σύνδεση Arduino με αποστασιόμετρο

Ο αισθητήρας HC-SR04 αποτελείται από 4 pins.
Το Vcc pin τροφοδοτεί το κύκλωμα με 5V και συνδέεται στο pin του Arduino με την ένδειξη 5V.

Το trig pin παρέχει σύντομο παλμό HIGH για να παραχθεί ο υπερηχητικός παλμός και να ξεκινήσει η μέτρηση χρόνου επιστροφής και άρα η απόσταση από το αντικείμενο που ανακλά τη δέσμη. Συνδέεται στο pin 10.

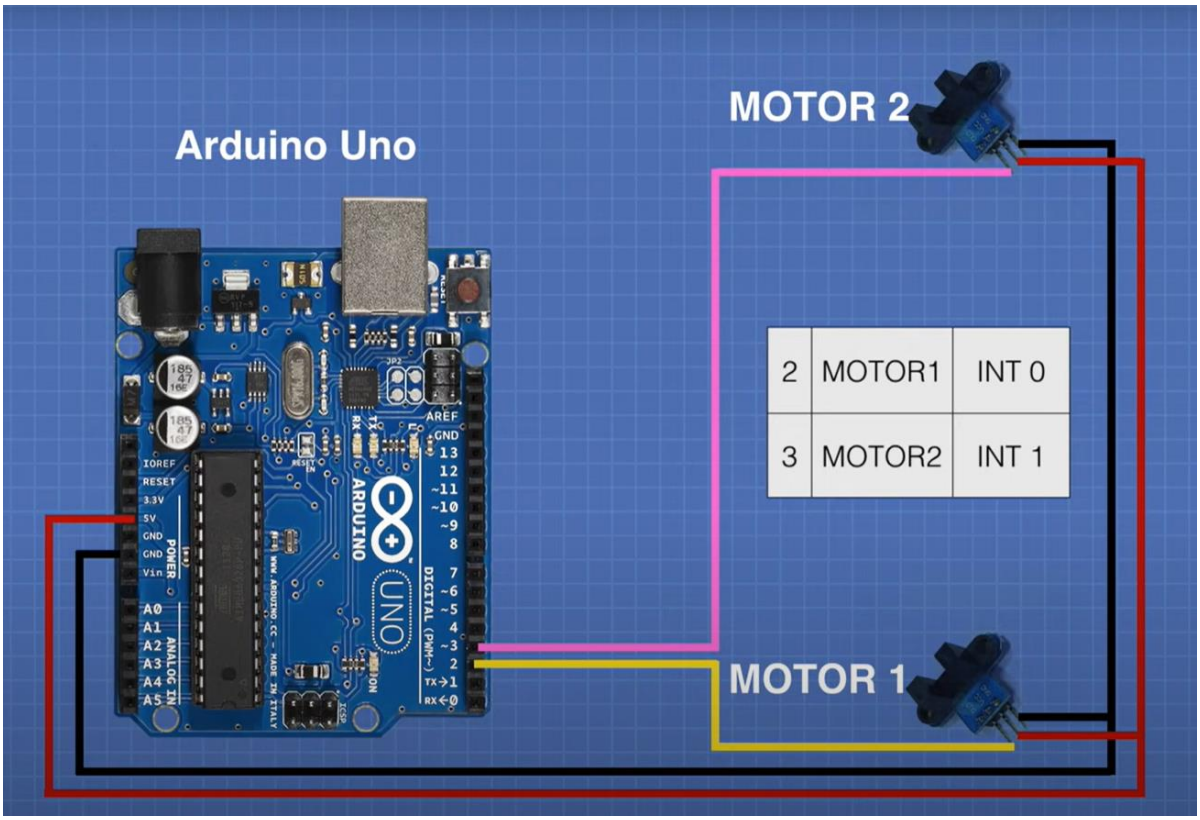
Το echo pin παράγει έναν θετικό παλμό (HIGH) από την στιγμή αποστολής του υπερηχητικού παλμού έως και την λήψη της ανάκλασης, οπότε το pin αυτό γίνεται LOW. Με μέτρηση της διάρκειας αυτού του παλμού μπορούμε να υπολογίσουμε τον συνολικό χρόνο αποστολής, ανάκλασης και λήψης του υπερηχητικού παλμού. Συνδέεται στο pin 12.

Το GND είναι η γείωση του αισθητήρα.



Εικόνα 3.3: Συνδεσμολογία του αποστασιόμετρου με τον Arduino.
[Η εικόνα αντλήθηκε από τον ιστότοπο https://stock.adobe.com/gr_en/search/images?k=arduino+uno]

3.4 Σύνδεση Arduino με Photo Interrupter Sensors



Εικόνα 3.4: Συνδεσμολογία του Arduino με τους οπτικούς αισθητήρες φωτός.
[Η εικόνα αντλήθηκε από το Youtube https://www.youtube.com/watch?v=oQQpAACa3ac&list=LL&index=5&t=2009s]

Η συνδεσμολογία τους είναι πολύ απλή. Κάθε αισθητήρας έχει 3 pin.

Το pin GND είναι το pin της γείωσης και βρίσκεται στη μέση των pins. Θα συνδεθεί με την αντίστοιχη γείωση στον Arduino ή στην γέφυρα. Η γείωση είναι κοινή για όλο το σύστημα οπότε δεν έχει σημασία σε ποιο από τα δύο θα είναι.

Το pin DOUT είναι η έξοδος του pin (κίτρινο και ροζ καλώδιο) και με αυτό μεταδίδει τα δεδομένα στον Arduino. Θα τα συνδέσουμε με τα pin 2 και 3 του Arduino καθώς αυτά τα δυο είναι τα pin διακοπών. Αν τοποθετηθούν σε άλλα pin το κύκλωμα δεν θα δουλέψει.

Το pin Vcc είναι η τροφοδοσία του αισθητήρα και θα συνδεθεί στα 5V του Arduino ή αντίστοιχα στα 5V της γέφυρας.

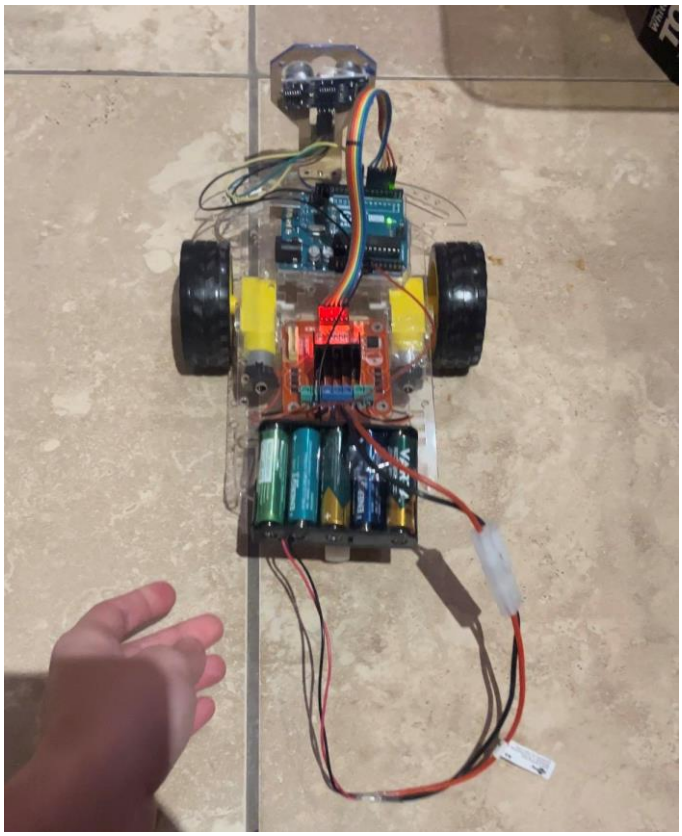
3.5 Τροφοδοσία του 1ου ρομποτικού οχήματος

Για την τροφοδοσία της γέφυρας και των κινητήρων θα χρησιμοποιήσουμε 5 επαναφορτιζόμενες μπαταρίες των 1.2V συνδεμένες σε σειρά οι οποίες θα αποδίδουν περίπου 6V. Θα τις τοποθετήσουμε σε μια θήκη μπαταριών και θα συνδέσουμε τον θετικό πόλο στην είσοδο VMS της γέφυρας. Αυτή η είσοδος μπορεί να δεχτεί μέχρι 35V.

Για την τροφοδοσία του Arduino και των αισθητήρων που θα συνδέσουμε στα pins του θα χρησιμοποιήσουμε μια μπαταρία των 9V στην είσοδο DC του Arduino. Έτσι, θα αποκτήσει το σύστημα μας μεγαλύτερη αυτονομία και δεν θα επιβαρυνθεί η γέφυρα L298N με την τροφοδοσία του Arduino UNO.

3.6 Το όχημα ολοκληρωμένο

Στην εικόνα 3.5 βλέπουμε να έχουν γίνει όλες οι απαραίτητες συνδέσεις με το Arduino, τη γέφυρα και τους αισθητήρες όπως περιγράψαμε στις παραπάνω ενότητες. Αφού φορτώσουμε τον κώδικα στο Arduino, το όχημα είναι έτοιμο να λειτουργήσει αυτόνομα αποφεύγοντας εμπόδια.



Εικόνα 3.5: Η πρώτη κατασκευή συναρμολογημένη και έτοιμη για λειτουργία.

3.7 Ανάλυση Κώδικα της 1ης κατασκευής

Παρακάτω αναλύσουμε τον κώδικα για τη λειτουργία του οχήματος μέσω του Arduino χωρίς ασύρματο έλεγχο δικτύου.

Οι δύο βασικές μέθοδοι που θα μας απασχολήσουν κατά τη διάρκεια της εργασίας, αλλά είναι απαραίτητες και για οποιοδήποτε project, είναι η setup() και η loop().

- Setup() – εδώ βάζουμε όλες τις εντολές που πρέπει να τρέξουν μια φορά στην αρχή του προγράμματος, όταν δηλαδή δοθεί ρεύμα για πρώτη φορά στη πλακέτα. Συνήθως, σε αυτή τη μέθοδο θα δούμε τη δήλωση μεταβλητών ως είσοδοι ή έξοδοι. Δηλώνουμε ποια pin θα οριστούν ως είσοδοι και θα λαμβάνουν εντολές από το IDE και ποια θα οριστούν ως έξοδοι και τα στέλνουν δεδομένα από το IDE προς το εξάρτημα που μας ενδιαφέρει.
- Loop() – εδώ γράφουμε τις εντολές που θα τρέξουν σε μια επαναλαμβανόμενη διαδικασία για όσο θέλουμε εμείς. Συνήθως καλούμε μεθόδους που έχουμε δημιουργήσει ή προϋπάρχουν σε βιβλιοθήκες και συμπεριλάβαμε στην αρχή του κώδικα.

Παράδειγμα

```
void setup() {
  /* Οι εντολές εδώ θα τρέξουν μόνο στην ενεργοποίηση ή μετά το Reset */
}
void setup() {
  /* Οι εντολές εδώ θα τρέχουν συνεχώς μέχρι να απενεργοποιηθεί το πρόγραμμα */
}
```

Όπως σε κάθε γλώσσα προγραμματισμού, έτσι και στη γλώσσα του Arduino, πρέπει να γίνονται οι δηλώσεις μεταβλητών με τα ονόματα και τους τύπους τους.

Στην εργασία αυτή, θέλουμε αρχικά να αντιστοιχήσουμε τα pin του Arduino με τις μεταβλητές. Όπως έχουμε δει παραπάνω στις συνδεσμολογίες, οι κινητήρες του οχήματος θα συνδεθούν με τη γέφυρα και αυτή με το Arduino.Έπειτα θα πρέπει να δηλώσουμε τα pin του Arduino με τις μεταβλητές στον κώδικα:

```
//Κινητήρας A
int enA = 6;
int in1 = 7;
int in2 = 5;

//Κινητήρας B
int enB = 11;
int in3 = 4;
int in4 = 10;
```

Στη συνέχεια, πρέπει να δηλώσουμε τα pins των οπτικών αισθητήρων στροφών. Επειδή αυτοί οι αισθητήρες λειτουργούν με διακοπές, τα κατάλληλα pin του Arduino για αυτή τη λειτουργία είναι τα 2,3.

```
//Οπτικοί αισθητήρες στροφών
const byte MOTOR1= 2; //Κινητήρας A Interrupt Pin - INT 0
const byte MOTOR2= 3; //Κινητήρας B Interrupt Pin - INT 1
```

Έπειτα θα πρέπει θα δηλωθούν τα pins του αποστασιόμετρου, δηλαδή το pin που θα λαμβάνει δεδομένα και το pin που θα στέλνει δεδομένα μέσω της σειριακής επικοινωνίας καθώς και οι μεταβλητές για τον υπολογισμό της απόστασης του οχήματος από το εμπόδιο. Επίσης θα πρέπει να αρχικοποιήσουμε δυο μεταβλητές για να μετράμε τους παλμούς. Τέλος, χρειαζόμαστε και μια μεταβλητή με τον αριθμό των εγκοπών του οπτικού δίσκου(βλ. εικ 20).

```
//Αποστασιομετρο
const int trigPin = 9;
const int echoPin = 8;
// Απόσταση σε εκατοστά
long cm_dist;
//για να μετράμε τους παλμούς
unsigned int counter1 = 0;
unsigned int counter2 = 0;
float diskslots= 20.00;
```

Εικόνα 3.6 Τροχός για τη μέτρηση των εγκοπών από τον οπτικό αισθητήρα

Οι επόμενες δύο μέθοδοι θα χρειαστούν για τον υπολογισμό των παλμών του κινητήρα Α και Β αντίστοιχα.

```
void ISR_count1(){
  counter1++;
}
//Motor2 pulse count
void ISR_count2(){
  counter2++;
}
```

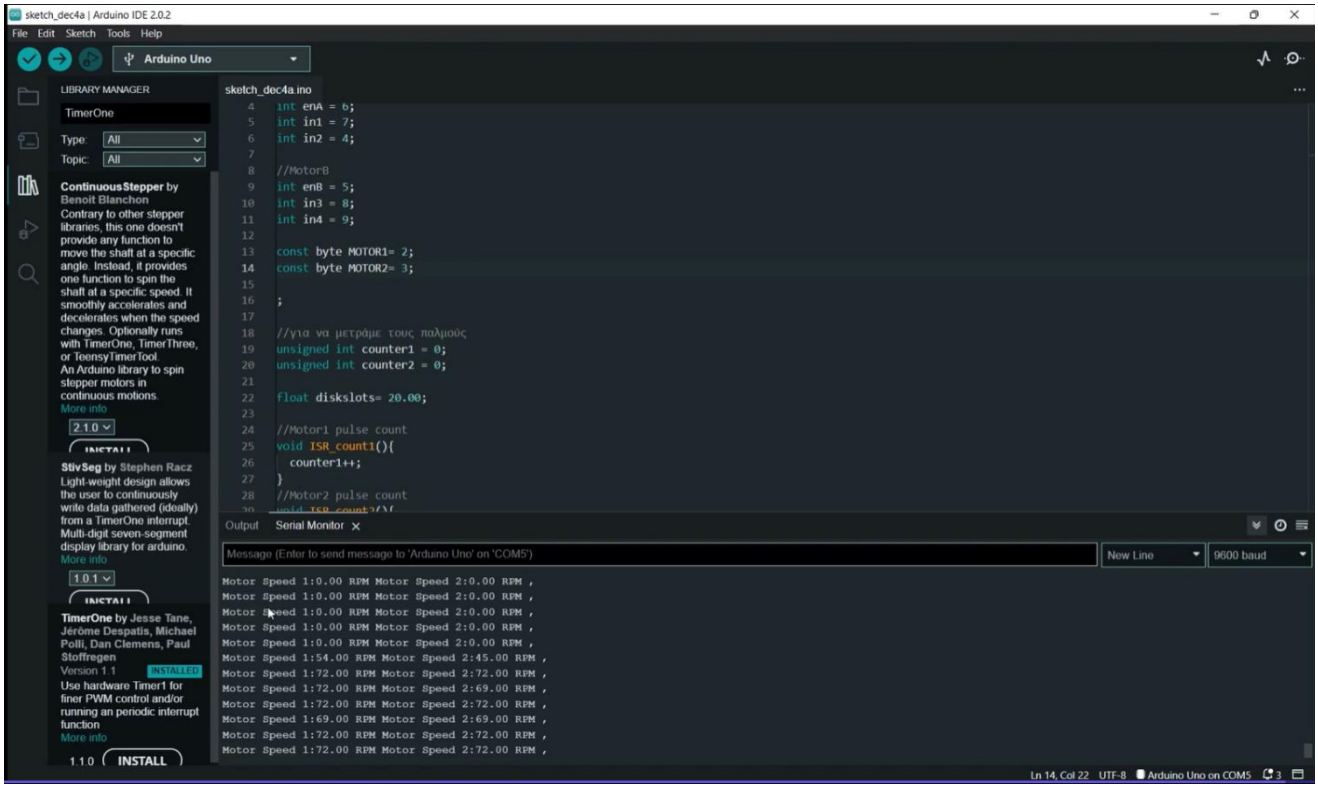
Η μέθοδος ISR_timerone() είναι αυτή που θα μετατρέπει τα δεδομένα που λαμβάνει από τους οπτικούς αισθητήρες και θα τα τυπώνει στην κονσόλα. Θα μας ενημερώνει με πόσες στροφές το λεπτό περιστρέφεται ο κάθε τροχός ξεχωριστά. Ο υπολογισμός των στροφών γίνεται με την εξής εντολή:

```
float rotation1= (counter1/ diskslots) * 60.00;
float rotation2= (counter2/ diskslots) * 60.00;
```

όπου counter είναι η μεταβλητή που μετρά τους παλμούς δια των 20 εγκοπών του δίσκου και όλο αυτό επί τα 60 δευτερόλεπτα του λεπτού. Αυτή η πράξη θα μας δώσει τις στροφές ανά λεπτό. Η εκτύπωση του αποτελέσματος στην κονσόλα γίνεται με τις εντολές:

```
Serial.print(rotation1);
Serial.print(" RPM ");
Serial.print(rotation2);
Serial.print(" RPM ,\n");
```

Τα αποτελέσματα των παραπάνω εντολών θα εμφανίζονται στην κονσόλα μας όπως στην εικόνα που ακολουθεί:



Εικόνα 3.7 Στιγμιότυπο του κώδικα κατά τη λειτουργία των οπτικών αισθητήρων στροφών.

Όταν καλούμε την Timer1.detachInterrupt(); Παύουμε οποιαδήποτε διαδικασία μέτρησης ή ρύθμισης και ο μόνος τρόπος επαναφοράς της διακοπής είναι μέσω της εντολής Timer1.attachInterrupt(ISR_timerone);

Στη συνέχεια θα καλέσουμε την μέθοδο void setup() {} στην οποία έγινε ήδη αναφορά. Σε αυτήν θα ορίσουμε αν τα δηλωμένα pin του Arduino θα είναι pin εισόδου ή εξόδου. Θα αρχικοποιήσουμε τον Timer.attachInterrupt και τα pins του οπτικού αισθητήρα φωτός με τον παρακάτω κώδικα:

```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  Timer1.initialize(1000000);
  attachInterrupt(digitalPinToInterrupt (MOTOR1), ISR_count1,RISING);
  attachInterrupt(digitalPinToInterrupt (MOTOR2), ISR_count2,RISING);
  Timer1.attachInterrupt(ISR_timerone);//Enable the timer
  pinMode(enA, OUTPUT);
  pinMode(enB, OUTPUT);
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);

}
```

Η εντολή Serial.begin(9600); Προετοιμάζει τον Arduino για την ανταλλαγή μηνυμάτων με την σειριακή πύλη σε ρυθμό δεδομένων 9600 bits το δευτερόλεπτο.

Για την κίνηση των τροχών θα χρειαστούμε 4ς υπορουτίνες που να προγραμματίζουν την κίνηση εμπρός, πίσω, δεξιά και αριστερά.

Στην κίνηση εμπρός θέλουμε οι κινητήρες να δέχονται παλμο HIGH στα pins 2 και 3 αντίστοιχα, ενώ στα άλλα δύο pins παλμό LOW.

Επίσης θα πρέπει να δοθεί και μια ταχύτητα στους κινητήρες. Μετά από διάφορους υπολογισμούς και πειράματα που έγιναν και σε συνεργασία με τα στροφόμετρα, διαπιστώθηκε ότι η σωστή ταχύτητα για να πηγαίνουν ευθεία οι κινητήρες είναι 120 για τον A και 100 για τον B. Έτσι προκύπτει ο εξής κώδικας:

```
// Υπορουτίνα κίνησης προς τα μπροστά
void Move_Forward()
{
 // Εκκίνηση του κινητήρα 1
 digitalWrite(in1, LOW);
```

```
 digitalWrite(in2, HIGH);
 // Εκκίνηση του κινητήρα 2
 digitalWrite(in3, LOW);
 digitalWrite(in4, HIGH);
 // Ορισμός ταχύτητας κινητήρα 1 – τιμές 0-255
analogWrite(enA, 120);
analogWrite(enB, 100);
// καθυστέρηση για κίνηση προς τα εμπρός
delay(100);
// Σταμάτημα των κινητήρων
digitalWrite(in1, LOW);
digitalWrite(in2, LOW);
digitalWrite(in3, LOW);
digitalWrite(in4, LOW); }
```

Για τις υπόλοιπες υπορουτίνες θα ακολουθήσουμε την ίδια διαδικασία με μόνη διαφορά τους παλμούς που θα δίνουμε στους κινητήρες.

Για το σταμάτημα των τροχών θα κατασκευάσουμε μια υπορουτίνα στην οποία όλοι οι παλμοί θα είναι LOW όπως παρακάτω:

```
void MotorStop(void)
{
digitalWrite(enB,LOW);
digitalWrite(enA,LOW);
digitalWrite(in1,LOW);
digitalWrite(in2,LOW);
digitalWrite(in3,LOW);
digitalWrite(in4,LOW);
}
```

Όσον αφορά το αποστασιόμετρο, θα χρειαστούμε μια υπορουτίνα ανάγνωσης της απόστασης. Ο τρόπος που θα λειτουργεί είναι να λαμβάνει δεδομένα από τον αισθητήρα σχετικά με το πόσα δευτερόλεπτα κάνει να επιστρέψει το ηλεκτρομαγνητικό κύμα και έτσι θα μετατρέπει τα δευτερόλεπτα σε απόσταση και θα μας επιστρέφει τον αριθμό σε εκατοστά.

Αρχικά, στην υπορουτίνα πρέπει να δηλώσουμε τις μεταβλητές που θα χρησιμοποιήσουμε, δηλαδή: `long duration,cm;`

Πρώτα παρέχουμε έναν σύντομο παλμό LOW για να αποσταλεί ένας καθαρός παλμός HIGH από τον αισθητήρα.

```
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
```

```
digitalWrite(trigPin, LOW);
```
Έπειτα θα γίνει ανάγνωση του σήματος όταν επιστρέψει από τον αισθητήρα. Ένας παλμός HIGH έχει διάρκεια από τη στιγμή αποστολής του υπερηχητικού παλμού έως την επιστροφή του μετά από ανάκλαση σε κάποιο αντικείμενο. Η εντολή για την ανάγνωση του σήματος είναι
```
duration = pulseIn(echoPin, HIGH,30000);
```
Αν λάβουμε την τιμή μηδέν τότε πρέπει να επαναφέρουμε την στάθμη του echoPin με τις εντολές:

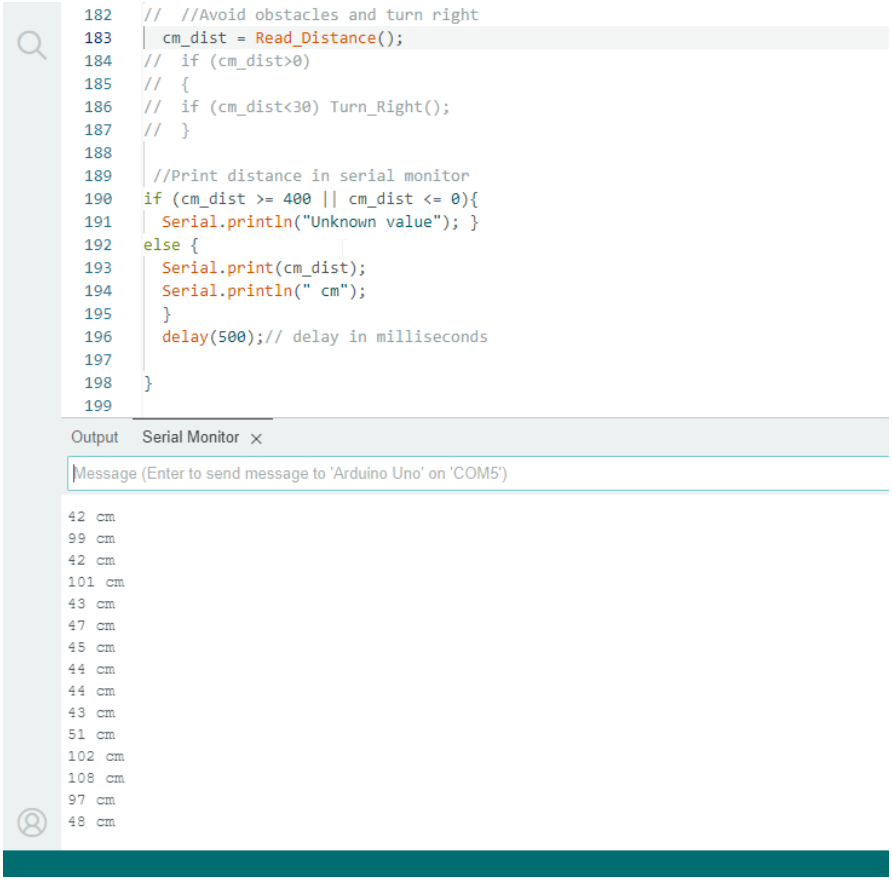
```
if (duration==0)
 {
 pinMode(echoPin, OUTPUT);
 delay(10);
 digitalWrite(echoPin, LOW);
 delay(10);
 pinMode(echoPin, INPUT);
 delay(10);
 }
```
Επίσης, πρέπει να τοποθετήσουμε μέσα στη μεταβλητή cm πόσα είναι τα εκατοστά από το αντικείμενο. Θα χρειαστούμε μια συνάρτηση μετατροπής των δευτερολέπτων σε εκατοστά. Ας την ονομάσουμε microsecondsToCentimeters. Εφόσον η ταχύτητα του ήχου είναι 340 m/s ή 29 microseconds ανά εκατοστό, και εφόσον ο υπερηχητικός παλμός ταξιδεύει εμπρός και πίσω, για να υπολογίσουμε την απόσταση του αντικειμένου λαμβάνουμε το ήμισυ της διανυόμενης απόστασης.Οπότε:

```
long microsecondsToCentimeters(long microseconds){
  return microseconds/58;}
```
Αύτη τη μέθοδο θα την καλέσουμε μέσα στη μέθοδο που υπολογίζει τη διάρκεια για να μας δώσει εν τέλει την απόσταση από το αντικείμενο.

Τέλος, θα καλέσουμε τη συνάρτηση loop() στην οποία θα γράψουμε όλες τις εντολές που θέλουμε να εκτελούνται ατέρμονα για όση ώρα τροφοδοτούμε το όχημα.

Για παράδειγμα, θέλουμε το όχημα να κινείται μπροστά και να υπολογίζει την απόσταση από τα αντικείμενα που συναντά. Αν αυτή η απόσταση είναι μικρότερη των 30 εκατοστών τότε θα στρίβει δεξιά.

Επίσης θέλουμε να μας τυπώνει στην κονσόλα πόση είναι η απόσταση από το αντικείμενο που συναντά. Αυτό δεν μπορεί να υπολογιστεί όσο το όχημα είναι αυτόνομο, αλλά μέσω της σύνδεσης USB μπορούμε να δούμε στην κονσόλα τα αποτελέσματα που λαμβάνει ο αισθητήρας όπως παρακάτω:



Εικόνα 3.8: Στιγμιότυπο κατά τη λειτουργία του αποστασιόμετρου

Ο κώδικας για να εκτελεστούν οι παραπάνω εντολές στην loop() είναι ο εξής:

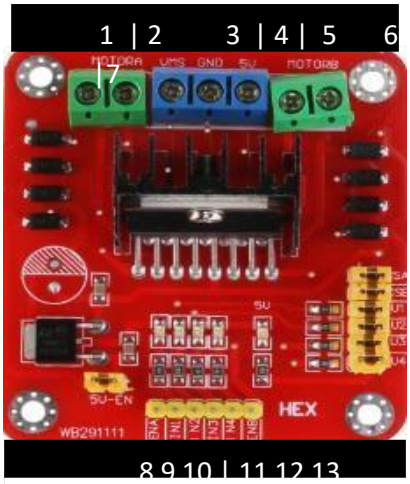
```
void loop()
{
 Move_Forward();

 //Avoid obstacles and turn right
 cm_dist = Read_Distance();
 if (cm_dist>0)
 {
 if (cm_dist<30) Turn_Right(); }
 //Print distance in serial monitor
if (cm_dist >= 400 || cm_dist <= 0){
  Serial.println("Unknown value"); }
else {
  Serial.print(cm_dist);
  Serial.println(" cm");
  }
  delay(500);// delay in milliseconds
}
```

4ο ΚΕΦΑΛΑΙΟ Ανάλυση κώδικα και συνδεσμολογίας 2ης κατασκευής

Η κατασκευή και συνδεσμολογία του δεύτερου ρομποτικού οχήματος θα εξεταστεί αναλυτικά και έπειτα θα δούμε το τελικό σύστημα συναρμολογημένο μαζί με τον κώδικα για τον προγραμματισμό του και το εργαλείο Blynk για τον ασύρματο έλεγχο του. Για τις πληροφορίες του σχεδιασμού της εφαρμογής μέσω Blynk μπορείτε να ανατρέξετε στο κεφάλαιο 2.3 Εργαλεία: Εφαρμογή Blynk.

4.1 Σύνδεση γέφυρας με κινητήρες

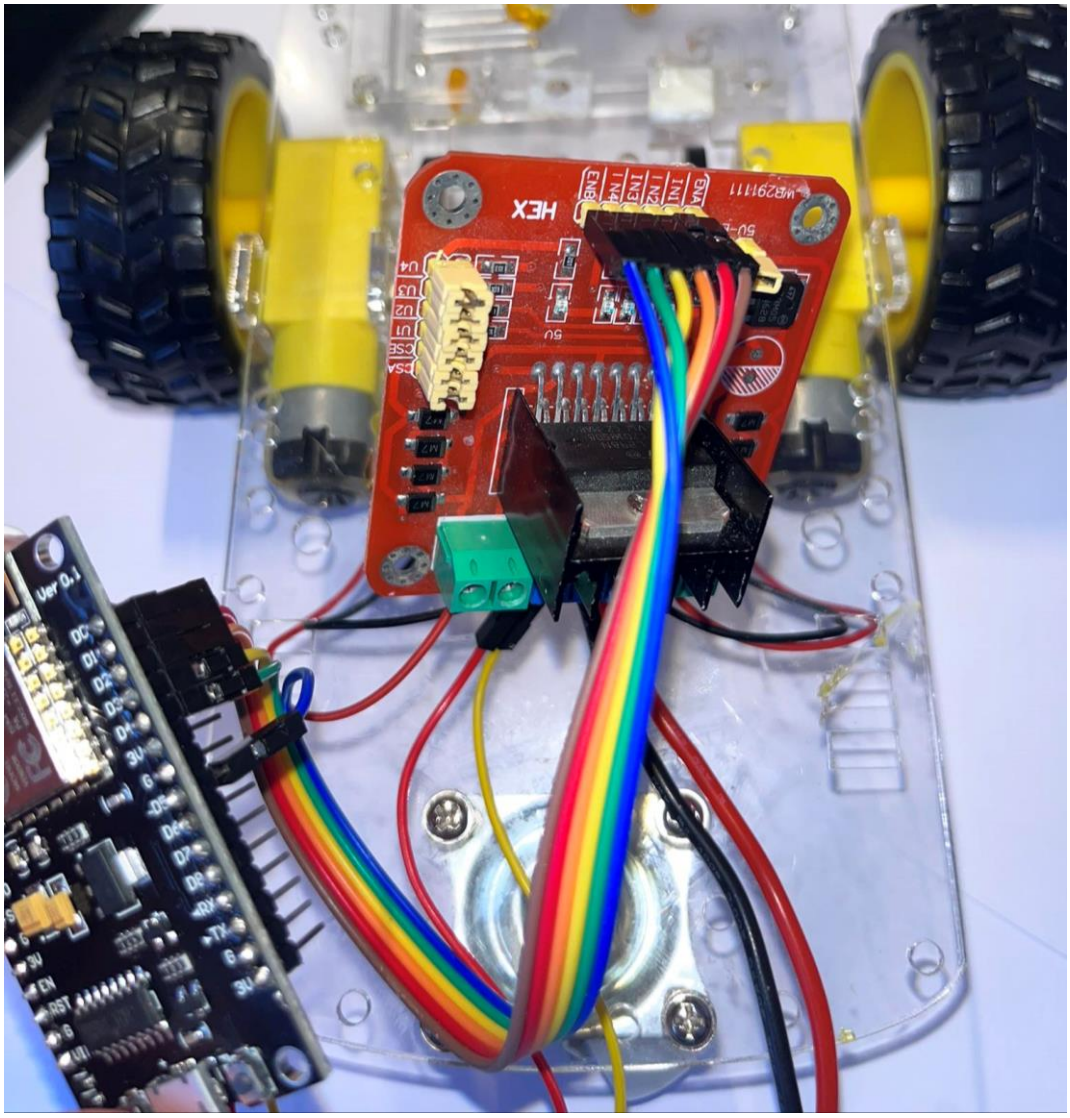
Η σύνδεση αυτών των δυο ακολουθεί την λογική της 1ης κατασκευής.
Για να θέσουμε σε κίνηση τους τροχούς του οχήματος θα πρέπει αρχικά να υλοποιήσουμε τις κατάλληλες συνδέσεις τους με την γέφυρα WB291111. Κάθε ένας από τους δύο κινητήρες έχει δυο καλώδια, τα οποία ανάλογα με το ρεύμα που παρέχουμε, θα κινούν τους τροχούς μπροστά ή πίσω. Τα καλώδια που προέρχονται από τον πρώτο κινητήρα (motorA) θα τα συνδέσουμε στα pins 1 και 2 της εικόνας ενώ, τα καλώδια του δεύτερου κινητήρα (motorB) θα τα συνδέσουμε στα pins 6 και 7.
Τα pins 3 και 4 είναι υπεύθυνα για την τροφοδοσία της γέφυρας (6V και Ground) ενώ το pin 5 είναι ικανό να τροφοδοτήσει την πλακέτα Arduino με τον τρόπο που θα δούμε παρακάτω.
Η τροφοδοσία της γέφυρας γίνεται με 5 επαναφορτιζόμενες μπαταρίες 1.2V σε ειδική θήκη μπαταριών την οποία θα συνδέσουμε στο pin 4 (GND) και στο pin 3(VMS). Αν οι μπαταρίες δεν είναι πλήρως φορτισμένες και δίνουν κάτω από 6 V, το σύστημα θα υπολειτουργεί και το pin 5 δεν θα μπορεί παρέχει αρκετή τάση για τη πλακέτα Arduino ή κάποιον αισθητήρα που πιθανώς να συνδέσουμε.
Αυτή, λοιπόν, είναι η συνδεσμολογία για τη σύνδεση των δυο κινητήρων με τη γέφυρα WB291111.



Εικόνα 4.1: Η Γέφυρα WB291111

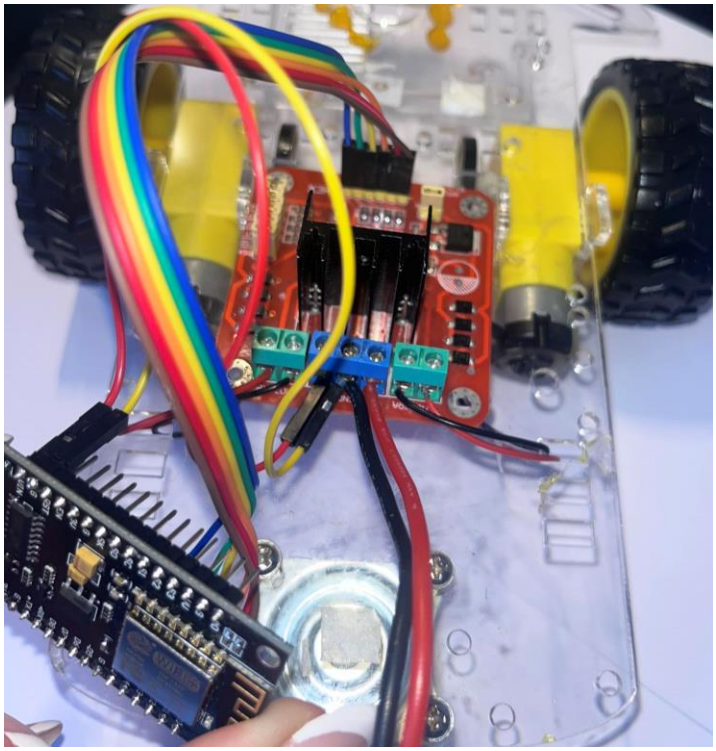
4.2 Σύνδεση γέφυρας με ESP8266

Σε αυτή την κατασκευή, αφαιρέσαμε την πλακέτα Arduino UNO, το αποστασιόμετρο και τους οπτικούς αισθητήρες φωτός. Ακολουθεί η συνδεσμολογία του ESP8266 με τη γέφυρα, το στάδιο στο οποίο θα δούμε τον ασύρματο έλεγχο του οχήματος



Εικόνα 4.2 Σύνδεση της γέφυρας με το ESP8266

Όπως φαίνεται στην παραπάνω εικόνα, έχουμε συνδέσει το D0 pin του ESP8266 με το ENA, το D1 με το IN1, το D2 με το IN2, το D3 με το IN3, το D4 με το IN4 και το D5 με το ENB.



Εικόνα 4.3 Σύνδεση της γείωσης και του ρεύματος του ESP8266 με τη γέφυρα

Επίσης, συνδέουμε το GND του ESP με το GND της γέφυρας και την τροφοδοσία Vin με την έξοδο 5V της γέφυρας.

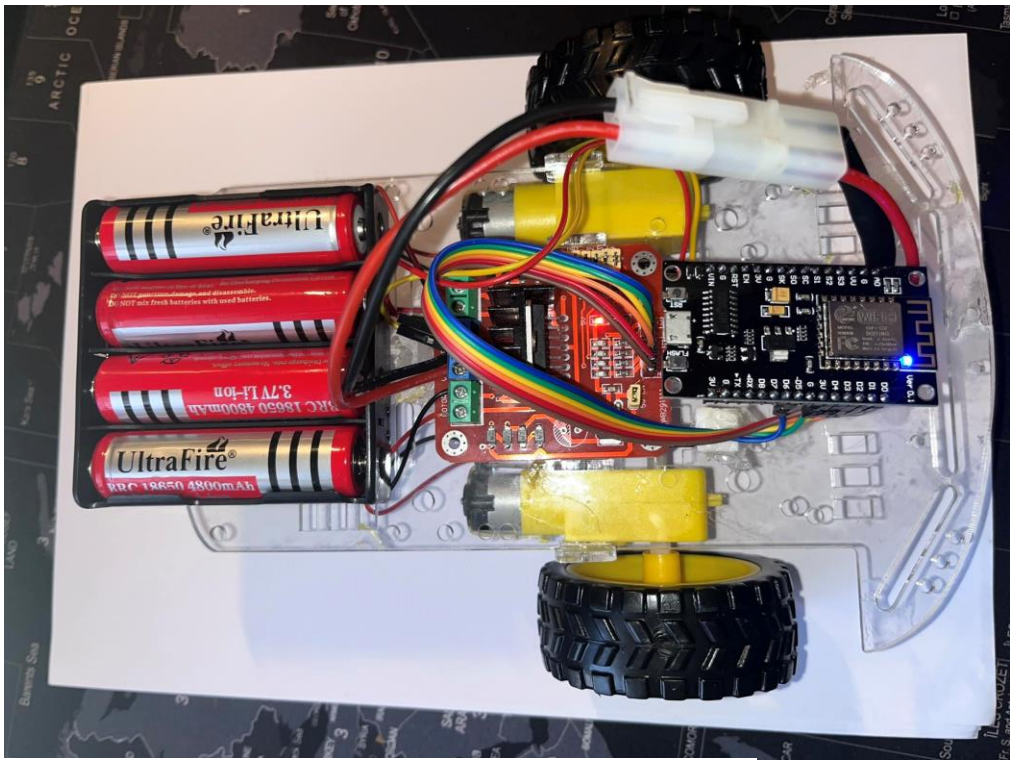
4.3 Τροφοδοσία του 2ου ρομποτικού οχήματος

Η τροφοδοσία όλου του συστήματος θα γίνει με 4ς μπαταρίες 3.7 V οι οποίες θα συνδεθούν στην Vin είσοδο της γέφυρας. Θα τις τοποθετήσουμε σε σειρά και θα αποδίδουν περίπου 16 V.

Εικόνα 4.4 Μέτρηση τροφοδοσίας με βολτόμετρο.

4.4 Το όχημα ολοκληρωμένο

Στην εικόνα βλέπουμε την συναρμολόγηση όλου του οχήματος.



Εικόνα 4.5 Το 2ο όχημα συναρμολογημένο

Αφού γίνουν οι κατάλληλες συνδέσεις μεταξύ τους, μπορούμε να συνδέσουμε το ESP8266 στον υπολογιστή για να φορτώσουμε τον κώδικα που ετοιμάσαμε. Έπειτα ανοίγουμε την εφαρμογή στο κινητό μας και το όχημα είναι έτοιμο να ελεγχθεί ασύρματα.

4.5 Ανάλυση Κώδικα της 2ης κατασκευής

Όπως είχαμε δει στο κεφάλαιο 2.3.2 Εφαρμογή Blynk, κατά τη δημιουργία του template της εφαρμογής και της σύνδεσης της συσκευής ESP8266 με αυτό το template, η εφαρμογή παράγει αυτόματα κάποια μοναδικά αναγνωριστικά. Αυτά θα πρέπει να τοποθετηθούν οπωσδήποτε στην αρχή του κώδικα όπως παρακάτω:

```
#define BLYNK_TEMPLATE_ID "TMPL4VcpINbQa"
#define BLYNK_TEMPLATE_NAME "Arduino Car"
#define BLYNK_AUTH_TOKEN "gInjf9ME9sWmSzzYAF2tKE7_AKGi5Br3"
```

Έπειτα, θα πρέπει να συμπεριλάβουμε τις εξής βιβλιοθήκες:

```
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
```

Η βιβλιοθήκη `ESP8266WiFi.h` είναι σημαντική για τον προγραμματισμό των μικροελεγκτών ESP8266 και τη σύνδεσή τους στο δίκτυο. Παρέχει μεθόδους για τη σύνδεση των μικροελεγκτών στο τοπικό δίκτυο με τη χρήση του SSID (όνομα δικτύου) και τον κωδικό.

Η βιβλιοθήκη `BlynkSimpleEsp8266.h` είναι απαραίτητη όταν εργαζόμαστε με την πλατφόρμα Blynk και μικροελεγκτές όπως ο ESP8266. Απλοποιεί κατά πολύ τη διαδικασία σύνδεσης μιας συσκευής με τις υπηρεσίες cloud του Blynk.Μας παρέχει μεθόδους για τη σύνδεση στο τοπικό δίκτυο, όπως η `Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);`

Στη συνέχεια πρέπει να καθορίσουμε τα pin που θα χρησιμοποιήσουμε στην πλακέτα ESP8266. Όπως αναφέραμε και σε προηγούμενο κεφάλαιο, τα pins D0-D8 είναι ψηφιακές είσοδοι-έξοδοι. Δεν έχει σημασία σε ποια σειρά θα τα τοποθετήσουμε. Στα πλαίσια της εργασίας οι συνδέσεις έγιναν ως εξής:

```
//MOTOR PINS
#define ENA D0
#define ENB D5
#define IN1 D1
#define IN2 D2
#define IN3 D3
#define IN4 D4
```

Επίσης, πρέπει να γίνει αρχικοποίηση των μεταβλητών που θα περιγράφουν τις 4ς κινήσεις των τροχών, καθώς και η ταχύτητα αυτών. Οι τιμές που θέλουμε να λαμβάνουν οι μεταβλητές είναι 0 και 1, όπου μηδέν σημαίνει ότι δεν λαμβάνει σήμα από την εφαρμογή Blynk και 1 σημαίνει ότι λαμβάνει.

Η ταχύτητα μπορεί να πάρει τιμές από 0 έως 255.

Επομένως, οι μεταβλητές θα δηλωθούν ως εξής:

```
bool forward = 0;
bool backward = 0;
bool left = 0;
bool right = 0;
int Speed ;
```

Αφού δηλωθούν τα στοιχεία του τοπικού δικτύου μας ώστε να γίνει η επικοινωνία με το Wifi θα προσθέσουμε την setup(). Εδώ βάζουμε όλες τις εντολές που πρέπει να τρέξουν μια φορά στην αρχή του προγράμματος, όταν δηλαδή δοθεί ρεύμα για πρώτη φορά στη πλακέτα. Συνήθως, σε αυτή τη μέθοδο θα δούμε τη δήλωση μεταβλητών ως είσοδοι ή έξοδοι. Δηλώνουμε ποια pin θα οριστούν ως είσοδοι και θα λαμβάνουν εντολές από το IDE και ποια θα οριστούν ως έξοδοι και τα στέλνουν δεδομένα από το IDE προς το εξάρτημα που μας ενδιαφέρει.

```
pinMode(ENA,OUTPUT);
pinMode(IN1,OUTPUT);
pinMode(IN2,OUTPUT);
```

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

Με την εντολή `Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);` γίνεται η σύνδεση στο δίκτυο, καλείται η Blynk.config() ώστε να δώσει το Auth αναγνωριστικό, τη διεύθυνση του server κτλ.

Στη συνέχεια καλούμε την εντολή `BLYNK_WRITE` 5 φορές για κάθε ένα από τα εικονικά pin που είχαμε κατασκευάσει στην διαδικτυακή εφαρμογή (βλ. ενότητα 2.3.2). Σε κάθε pin θα αντιστοιχήσουμε και την μεταβλητή κίνησης που μας ενδιαφέρει. Παράδειγμα:

```
BLYNK_WRITE(V0)
{
  // Set incoming value from pin V0 to a variable
  forward = param.asInt();
}
```

Η μεταβλητή param είναι μια προκατασκευασμένη μεταβλητή της βιβλιοθήκης BlynkHandlers.h και η μέθοδος asInt() είναι μια μέθοδος της βιβλιοθήκης BlynkParam.h.

Για τη χρήση των μεθόδων της βιβλιοθήκης `BlynkSimpleEsp8266` συμβουλεύτηκα το documentation στον ακόλουθο σύνδεσμο: https://docs.blynk.io/en/.

Στη συνέχεια, κατασκευάσαμε 5 μεθόδους που να περιγράφουν την κίνηση των τροχών δίνοντας τους αντίστοιχους παλμούς. Επειδή όμως, στην εμπρός και πίσω κίνηση, οι δυο τροχοί δεν μετακινούνταν με τις ίδιες ταχύτητες παρόλο που ο Slider της εφαρμογής παρείχε ακριβώς την ίδια ταχύτητα, έπρεπε να δώσουμε διαφορετική ταχύτητα σε κάθε τροχό ώστε η κίνηση να είναι πραγματικά ευθεία.

```
void Move_Forward() {
  analogWrite(ENA, 247);
  analogWrite(ENB, 240);
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, HIGH);
}
void Move_Backward() {
  analogWrite(ENA, 240);
  analogWrite(ENB, 247);
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
}
```

Τέλος, δημιουργήσαμε μια μέθοδο που θα ελέγχει τις εντολές που λαμβάνει από την εφαρμογή και θα κινεί τους τροχούς. Αναλυτικά, αν η μεταβλητή forward λάβει την τιμή 1,

σημαίνει ότι ο χρήστης πάτησε το κουμπί forward. Επομένως, πρέπει να καλέσουμε την μέθοδο Move_Forward().Το ίδιο ισχύει για τις μεταβλητές backward, right, left. Αν καμία μεταβλητή δεν έχει την τιμή 1, τότε το όχημα βρίσκεται σε ακινησία. Θα ονομάσουμε τη μέθοδο :

```
void remoteControl(){
  if (forward == 1){
    Move_Forward();
    Serial.println("carforward");
  }else if (backward == 1){
    Move_Backward();
    Serial.println("carbackward");
  }else if (left == 1){
    Move_Left();
    Serial.println("carleft");
  }else if (right == 1){
    Move_Right();
    Serial.println("carright");
  }else if (forward == 0 && backward == 0 && left == 0 && right == 0){
    Stop();
    Serial.println("carstop");
  }
}
```

Η μέθοδος loop() είναι αυτή που θα εκτελέσει το πρόγραμμα μας ατέρμονα.

Η εντολή `Blynk.run();` είναι μια βασική Blynk ρουτίνα, υπεύθυνη για τη διατήρηση της σύνδεσης, την αποστολή δεδομένων και την λήψη δεδομένων.

Η εντολή `remoteControl();` μέσα στην loop() καλεί τη μέθοδο που κατασκευάσαμε πιο πάνω.

```
void loop()
{
 Blynk.run();
 remoteControl();
}
```

5ο ΚΕΦΑΛΑΙΟ Συμπεράσματα και Προοπτικές Εξέλιξης

Η παρούσα πτυχιακή εργασία αναλύει τη διαδικασία σχεδιασμού και υλοποίησης δύο ρομποτικών οχημάτων με τη χρήση απλών ελεγκτών χαμηλού κόστους. Τα δύο ρομποτικά οχήματα, το αυτόνομο που χρησιμοποιεί την πλακέτα Arduino και το τηλεκατευθυνόμενο που βασίζεται στην πλατφόρμα NodeMcu Lua ESP8266, προσφέρουν διαφορετικές λύσεις για τη ρομποτική και την αυτοματοποίηση.

Οι προοπτικές εξέλιξης της εργασίας αυτής περιλαμβάνουν βελτιώσεις στους αλγορίθμους ελέγχου για ακόμη αποτελεσματικότερη λειτουργία.

Για παράδειγμα στο 2ο όχημα μπορούμε να ενσωματώσουμε το αποστασιόμετρο και να λαμβάνουμε ασύρματα τα δεδομένα για τις αποστάσεις των αντικειμένων.

Επιπλέον, μπορεί να εξεταστεί η αύξηση της αυτονομίας των οχημάτων με τη χρήση φωτοβολταικών συστημάτων.

Μπορεί να γίνει προσπάθεια ενσωμάτωσης των οχημάτων σε μεγαλύτερα δίκτυα και συστήματα επικοινωνίας.

Επίσης, οι ρομποτικές κατασκευές μπορούν να χρησιμοποιηθούν ως εκπαιδευτικά εργαλεία για την εκμάθηση των βασικών αρχών της ρομποτικής και την πρακτική εξάσκηση.

Συνοψίζοντας, η παρούσα εργασία αποτελεί μια εξαιρετική αφετηρία για τη συνεχιζόμενη έρευνα και ανάπτυξη στον τομέα της ασύρματης ρομποτικής και αυτοματοποίησης και ειδικότερα σε πολύ γνωστές πλατφόρμες όπως ο Arduino και ο μικροελεγκτής ESP8266. Επίσης είναι ένας πολύ καλός πρώτος οδηγός για την εξοικείωση με την ηλεκτρονική πλατφόρμα Blynk της οποίας οι λειτουργίες δεν περιορίζονται μόνο στο ESP8266 αλλά σε πολλές ακόμα τεχνολογίες. Οι προοπτικές εξέλιξης ανοίγουν νέους ορίζοντες για την εφαρμογή της τεχνολογίας σε διάφορους τομείς, ενώ παράλληλα ενισχύουν τη σημασία της τεχνολογίας στην σύγχρονη κοινωνία.

Βιβλιογραφία

[1] Article - EUROPA: A Case Study for Teaching Sensors, Data Acquisition and Robotics via a ROS-Based Educational Robot, by Georgios Karalekas, Stavros Vologiannidis and John Kalomoiros. *Published 27 April 2020 [από τον διαδικτυακό ιστότοπο https://www.mdpi.com/1424-8220/20/9/2469]*

[2] Εγχειρίδιο: Προγραμματίζοντας με τον μικροελεγκτή Arduino, Εμμανουήλ Πουλάκης *Ηράκλειο Ιανουάριος 2015* [από τον διαδικτυακό ιστότοπο https://users.sch.gr/manpoul/docs/arduino/ProgrammingArduino.pdf]

[3] «Δημιουργία εκπαιδευτικής ρομποτικής πλατφόρμας για εισαγωγή της ρομποτικής στη δευτεροβάθμια εκπαίδευση», Καραλέκας Γεώργιος, *Μεταπτυχιακή Εργασία του Μεταπτυχιακού Ρομποτικής του τμήματος Μηχανικών Πληροφορικής, ΤΕΙ Κεντρικής Μακεδονίας [από τον ιστότοπο http://apothesis.teicm.gr/xmlui/handle/123456789/3979]*

[4] «Η ρομποτική στην εκπαίδευση: Robotics in education», Paraskeva Evangelia Mountridou Maria, *Μεταπτυχιακή Διπλωματική Εργασία του Διιδρυματικού Προγράμματος Σπουδών του Παιδαγωγικού Τμήματος, ΑΙΓΑΛΕΩ 2020 [από τον διαδικτυακό ιστότοπο https://www.ebooks4greeks.gr/h-rompotikh-sthn-ekpaideysh]*

[5] «Ανάπτυξη Εφαρμογών με το Arduino: Ένας πλήρης οδηγός για το Arduino και ένα χρήσιμο εργαλείο για την ανάπτυξη εφαρμογών STEM», Παναγιώτης Παπάζογλου M.Sc., Ph.D., Σπύρος-Πολυχρόνης Λιώνης M.Sc., *Εκδόσεις ΤΖΙΟΛΑ 2η Έκδοση*

[6] "Εισαγωγή στη ρομποτική: Μηχανική και Αυτόματος Έλεγχος", John J. Craig, *Εκδόσεις ΤΖΙΟΛΑ 3η Έκδοση*

Παράρτημα Κώδικα

1. Κώδικας 1ης Κατασκευής

```
#include <TimerOne.h>
#include <ESP8266WiFi.h>

//MotorA
int enA = 6;
int in1 = 7;
int in2 = 5;
//MotorB
int enB = 11;
int in3 = 4;
int in4 = 10;
//RPM SENSORS
const byte MOTOR1= 2; //Motor 1 Interrupt Pin - INT 0
const byte MOTOR2= 3; //Motor 2 Interrupt Pin - INT 1
//Αποστασιομετρο
const int trigPin = 9;
const int echoPin = 8;
// Απόσταση σε εκατοστά
long cm_dist;
//για να μετράμε τους παλμούς
unsigned int counter1 = 0;
unsigned int counter2 = 0;
float diskslots= 20.00;
//Motor1 pulse count
void ISR_count1(){
  counter1++;
}
//Motor2 pulse count
void ISR_count2(){
  counter2++;
}
void ISR_timerone(){
  Timer1.detachInterrupt(); //stop the timer
  Serial.print("Motor Speed 1: ");
  float rotation1= (counter1/ diskslots) * 60.00; //calculate the value of
  its rotation
  Serial.print(rotation1);
  Serial.print(" RPM ");
  counter1 = 0;
  Serial.print("Motor Speed 2: ");
  float rotation2= (counter2/ diskslots) * 60.00; //calculate the value of
  its rotation
```

```
  Serial.print(rotation2);
  Serial.print(" RPM ,\n");
  counter2 = 0;
  Timer1.attachInterrupt(ISR_timerone); //Enable the timer
}
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  Timer1.initialize(1000000);
  attachInterrupt(digitalPinToInterrupt (MOTOR1), ISR_count1,RISING);
  attachInterrupt(digitalPinToInterrupt (MOTOR2), ISR_count2,RISING);
  Timer1.attachInterrupt(ISR_timerone);//Enable the timer
  pinMode(enA, OUTPUT);
  pinMode(enB, OUTPUT);
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);

  /* start server communication */
  server.begin();
}

// Υπορουτίνα κίνησης προς τα πίσω
void Move_Back() {
// Εκκίνηση του κινητήρα 1
digitalWrite(in1, HIGH);
digitalWrite(in2, LOW);
// Εκκίνηση του κινητήρα 2
digitalWrite(in3, HIGH);
digitalWrite(in4, LOW);
analogWrite(enA, 100);
analogWrite(enB, 100);
delay(100);
digitalWrite(in1, LOW);
digitalWrite(in2, LOW);
digitalWrite(in3, LOW);
digitalWrite(in4, LOW); }

// Υπορουτίνα κίνησης προς τα μπροστά
void Move_Forward()
{
 // Εκκίνηση του κινητήρα 1
```

```
87. digitalWrite(in1, LOW);
88. digitalWrite(in2, HIGH);
89. // Εκκίνηση του κινητήρα 2
90. digitalWrite(in3, LOW);
91. digitalWrite(in4, HIGH);
92. // Ορισμός ταχύτητας κινητήρα 1 – τιμές 0-255
93.analogWrite(enA, 120);
94.analogWrite(enB, 100);
95.// καθυστέρηση για κίνηση προς τα εμπρός
96.delay(100);
97.// Σταμάτημα των κινητήρων
98.digitalWrite(in1, LOW);
99.digitalWrite(in2, LOW);
100.     digitalWrite(in3, LOW);
101.     digitalWrite(in4, LOW); }
102.
103.     void Turn_Right() {
104.     // Εκκίνηση του κινητήρα 1
105.     digitalWrite(in1, LOW);
106.     digitalWrite(in2, HIGH);
107.     // Εκκίνηση του κινητήρα 2
108.     digitalWrite(in3, HIGH);
109.     digitalWrite(in4, LOW);
110.     analogWrite(enA, 100);
111.     analogWrite(enB, 100);
112.     // καθυστέρηση για υλοποίηση στροφής
113.     delay(500);
114.     digitalWrite(in1, LOW);
115.     digitalWrite(in2, LOW);
116.     digitalWrite(in3, LOW);
117.     digitalWrite(in4, LOW); }
118.
119.     void Turn_Left() {
120.     // Εκκίνηση του κινητήρα 1
121.     digitalWrite(in1, HIGH);
122.     digitalWrite(in2, LOW);
123.     // Εκκίνηση του κινητήρα 2
124.     digitalWrite(in3, LOW);
125.     digitalWrite(in4, HIGH);
126.     analogWrite(enA, 100);
127.     analogWrite(enB, 100);
128.     // καθυστέρηση για υλοποίηση στροφής
129.     delay(500);
130.     digitalWrite(in1, LOW);
131.     digitalWrite(in2, LOW);
```

```
132.      digitalWrite(in3, LOW);
133.      digitalWrite(in4, LOW); }
134.
135.      void MotorStop(void)
136.      {
137.      digitalWrite(enB,LOW);
138.      digitalWrite(enA,LOW);
139.      digitalWrite(in1,LOW);
140.      digitalWrite(in2,LOW);
141.      digitalWrite(in3,LOW);
142.      digitalWrite(in4,LOW);
143.      }
144.
145.      //Υπορουτίνα μετατροπής μικροδευτερολέπτων σε εκατοστά
146.      long microsecondsToCentimeters(long microseconds){
147.        return microseconds/58;
148.      }
149.
150.      //Υπορουτίνα ανάγνωσης απόστασης
151.      long Read_Distance()
152.      {
153.      long duration,cm;
154.      // Ο αισθητήρας εκκινεί με παλμό HIGH διάρκειας 10 ή περισσότερων
   microseconds.
155.      // Παρέχουμε έναν σύντομο παλμό LOW για να αποσταλεί ένας καθαρός
   παλμός HIGH:
156.      digitalWrite(trigPin, LOW);
157.      delayMicroseconds(2);
158.      digitalWrite(trigPin, HIGH);
159.      delayMicroseconds(10);
160.      digitalWrite(trigPin, LOW);
161.      // Ανάγνωση του σήματος από τον αισθητήρα: ένας παλμός HIGH που η
   διάρκειά του είναι από την στιγμή
162.      // αποστολής του υπερηχητικού παλμού έως την επιστροφή του μετά από
   ανάκλαση σε κάποιο αντικείμενο
163.      duration = pulseIn(echoPin, HIGH,30000);
164.      // Επαναφορά της στάθμης του echoPin αν λαμβάνεται τιμή 0
165.       if (duration==0)
166.       {
167.       pinMode(echoPin, OUTPUT);
168.       delay(10);
169.       digitalWrite(echoPin, LOW);
170.       delay(10);
171.       pinMode(echoPin, INPUT);
172.       delay(10);
```

```
 }
 cm = microsecondsToCentimeters(duration);
 return cm;
}

void loop()
{
 Move_Forward();

 //Avoid obstacles and turn right
 cm_dist = Read_Distance();
 if (cm_dist>0)
 {
 if (cm_dist<30) Turn_Right();
 }

 //Print distance in serial monitor
if (cm_dist >= 400 || cm_dist <= 0){
  Serial.println("Unknown value"); }
else {
  Serial.print(cm_dist);
  Serial.println(" cm");
  }
  delay(500);// delay in milliseconds

}
```

2. Κώδικας 2ης Κατασκευής

```
#define BLYNK_TEMPLATE_ID "TMPL4VcpINbQa"
#define BLYNK_TEMPLATE_NAME "Arduino Car"
#define BLYNK_AUTH_TOKEN "gInjf9ME9sWmSzzYAF2tKE7_AKGi5Br3"

#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>

//MOTOR PINS
#define ENA D0
#define ENB D5
#define IN1 D1
#define IN2 D2
#define IN3 D3
#define IN4 D4

bool forward = 0;
```

```
bool backward = 0;
bool left = 0;
bool right = 0;
int Speed ;

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "YOUR_SSID";
char pass[] = "YOUR_PASSWORD";

void setup()
{
  // Debug console
  Serial.begin(115200);
  pinMode(ENA,OUTPUT);
  pinMode(IN1,OUTPUT);
  pinMode(IN2,OUTPUT);
  pinMode(IN3,OUTPUT);
  pinMode(IN4,OUTPUT);
  pinMode(ENB,OUTPUT);

  Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);
}

BLYNK_WRITE(V0)
{
  // Set incoming value from pin V0 to a variable
  forward = param.asInt();
}
BLYNK_WRITE(V1)
{
  // Set incoming value from pin V0 to a variable
  backward = param.asInt();
}
BLYNK_WRITE(V2)
{
  // Set incoming value from pin V0 to a variable
  left = param.asInt();
}
BLYNK_WRITE(V3)
{
  // Set incoming value from pin V0 to a variable
  right = param.asInt();
}
BLYNK_WRITE(V4)
```

```
{
  // Set incoming value from pin V0 to a variable
  Speed = param.asInt();
}

void remoteControl(){
  if (forward == 1){
    Move_Forward();
    Serial.println("carforward");
  }else if (backward == 1){
    Move_Backward();
    Serial.println("carbackward");
  }else if (left == 1){
    Move_Left();
    Serial.println("carleft");
  }else if (right == 1){
    Move_Right();
    Serial.println("carright");
  }else if (forward == 0 && backward == 0 && left == 0 && right == 0){
    Stop();
    Serial.println("carstop");
  }
}

void loop()
{
  Blynk.run();
  remoteControl();
}

void Move_Forward(){
  analogWrite(ENA,Speed);
  analogWrite(ENB,Speed);
  digitalWrite(IN1,LOW);
  digitalWrite(IN2,HIGH);
  digitalWrite(IN3,LOW);
  digitalWrite(IN4,HIGH);

      }
      void Move_Backward(){
        analogWrite(ENA,Speed);
        analogWrite(ENB,Speed);
        digitalWrite(IN1,HIGH);
        digitalWrite(IN2,LOW);
        digitalWrite(IN3,HIGH);
```

```
107.       digitalWrite(IN4,LOW);
108.
109.     }
110.     void Move_Left(){
111.       analogWrite(ENA,Speed);
112.       analogWrite(ENB,Speed);
113.       digitalWrite(IN1,HIGH);
114.       digitalWrite(IN2,LOW);
115.       digitalWrite(IN3,LOW);
116.       digitalWrite(IN4,HIGH);
117.
118.     }
119.     void Move_Right(){
120.       analogWrite(ENA,Speed);
121.       analogWrite(ENB,Speed);
122.       digitalWrite(IN1,LOW);
123.       digitalWrite(IN2,HIGH);
124.       digitalWrite(IN3,HIGH);
125.       digitalWrite(IN4,LOW);
126.
127.     }
128.     void Stop(){
129.       digitalWrite(IN1,LOW);
130.       digitalWrite(IN2,LOW);
131.       digitalWrite(IN3,LOW);
132.       digitalWrite(IN4,LOW);
133.
134.     }
```