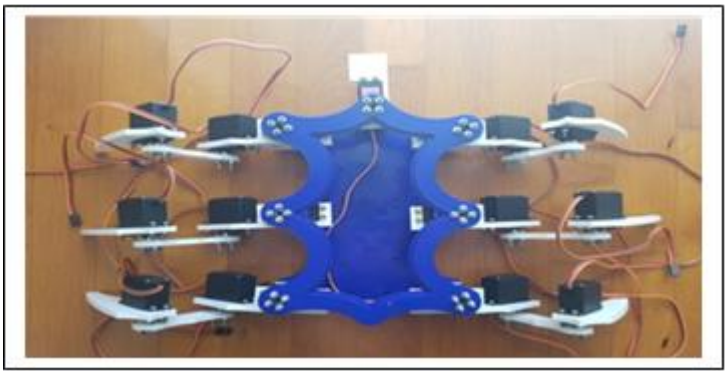
ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΒΙΟΜΗΧΑΝΙΚΗΣ ΣΧΕΔΙΑΣΗΣ ΚΑΙ ΠΑΡΑΓΩΓΗΣ

ΘΕΜΑ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

«ΣΧΕΔΙΑΣΗ ΚΑΙ ΑΝΑΠΤΥΞΗ ΕΝΟΣ ΕΞΑΠΟΔΟΥ ΡΟΜΠΟΤ»



ΟΝΟΜΑΤΑ ΦΟΙΤΗΤΩΝ:

ΧΕΛΗΣ ΑΠΟΣΤΟΛΟΣ - ΤΑΣΙΟΣ ΟΡΕΣΤΗΣ

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ:

ΧΑΤΖΟΠΟΥΛΟΣ ΑΒΡΑΑΜ

ΑΙΓΑΛΕΩ, ΙΟΥΛΙΟΣ 2019

Δήλωση των Συγγραφέων της Πτυχιακής Εργασίας

Ο κάτωθι υπογεγραμμένος Χέλης Απόστολος του Νικολάου, φοιτητής του Τμήματος Μηχανικών Βιομηχανικής Σχεδίασης και Παραγωγής του Πανεπιστημίου Δυτικής Αττικής, πριν αναλάβω την εκπόνηση της Πτυχιακής Εργασίας μας, δηλώνω ότι ενημερώθηκα για τα παρακάτω:

«Η Πτυχιακή Εργασία (Π.Ε) αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο του συγγραφέα, όσο και του Ιδρύματος και θα πρέπει να έχει μοναδικό χαρακτήρα και πρωτότυπο περιεχόμενο.

Απαγορεύεται αυστηρά οποιοδήποτε κομμάτι κειμένου της να εμφανίζεται αυτούσιο ή μεταφρασμένο από κάποια άλλη δημοσιευμένη πηγή. Κάθε τέτοια πράξη αποτελεί προϊόν λογοκλοπής και εγείρει θέμα Ηθικής Τάξης για τα πνευματικά δικαιώματα του άλλου συγγραφέα. Αποκλειστικός υπεύθυνος είναι ο συγγραφέας της Π.Ε, ο οποίος φέρει και την ευθύνη των συνεπειών, ποινικών και άλλων, αυτής της πράξης.

Πέραν των όποιων ποινικών ευθυνών του συγγραφέα, σε περίπτωση που το Ίδρυμα του έχει απονείμει Πτυχίο, αυτό ανακαλείται με απόφαση της Συνέλευσης του Τμήματος. Η Συνέλευση του Τμήματος με νέα απόφασή της, μετά από αίτηση του ενδιαφερόμενου, του αναθέτει εκ νέου την εκπόνηση Π.Ε με άλλο θέμα και διαφορετικό επιβλέποντα καθηγητή. Η εκπόνηση της εν λόγω Π.Ε πρέπει να ολοκληρωθεί εντός τουλάχιστον ενός ημερολογιακού 6μήνου από την ημερομηνία ανάθεσής της.

Ο Δηλών **Ημερομηνία**

Ο κάτωθι υπογεγραμμένος Τάσιος Ορέστης του Παύλου, φοιτητής του Τμήματος Μηχανικών Βιομηχανικής Σχεδίασης και Παραγωγής του Πανεπιστημίου Δυτικής Αττικής, πριν αναλάβω την εκπόνηση της Πτυχιακής Εργασίας μας, δηλώνω ότι ενημερώθηκα για τα παρακάτω:

«Η Πτυχιακή Εργασία (Π.Ε) αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο του συγγραφέα, όσο και του Ιδρύματος και θα πρέπει να έχει μοναδικό χαρακτήρα και πρωτότυπο περιεχόμενο.

Απαγορεύεται αυστηρά οποιοδήποτε κομμάτι κειμένου της να εμφανίζεται αυτούσιο ή μεταφρασμένο από κάποια άλλη δημοσιευμένη πηγή. Κάθε τέτοια πράξη αποτελεί προϊόν λογοκλοπής και εγείρει θέμα Ηθικής Τάξης για τα πνευματικά δικαιώματα του άλλου συγγραφέα. Αποκλειστικός υπεύθυνος είναι ο συγγραφέας της Π.Ε, ο οποίος φέρει και την ευθύνη των συνεπειών, ποινικών και άλλων, αυτής της πράξης.

Πέραν των όποιων ποινικών ευθυνών του συγγραφέα, σε περίπτωση που το Ίδρυμα του έχει απονείμει Πτυχίο, αυτό ανακαλείται με απόφαση της Συνέλευσης του Τμήματος. Η Συνέλευση του Τμήματος με νέα απόφασή της, μετά από αίτηση του ενδιαφερόμενου, του αναθέτει εκ νέου την εκπόνηση Π.Ε με άλλο θέμα και διαφορετικό επιβλέποντα καθηγητή. Η εκπόνηση της εν λόγω Π.Ε πρέπει να ολοκληρωθεί εντός τουλάχιστον ενός ημερολογιακού 6μήνου από την ημερομηνία ανάθεσής της.

Ο Δηλών **Ημερομηνία**

Ευχαριστίες

Η ανά χείρας εργασία αποτελεί το «απαύγασμα» της συνολικής μας προσπάθειας για την ολοκλήρωση των προπτυχιακών μας σπουδών στο τμήμα Βιομηχανικής Σχεδίασης και Παραγωγής του Πανεπιστημίου Δυτικής Αττικής. Κατά τη διάρκεια της φοίτησης μας αποκομίσαμε πλούσια γνωστικά εφόδια που συνέβαλαν καθοριστικά στην ατομική μας εξέλιξη και στην εκπόνηση της παρούσας έρευνας. Στο σημείο αυτό θα θέλαμε να ευχαριστήσουμε θερμά τον επιβλέποντα καθηγητή μας Αβραάμ Χατζόπουλο, ο οποίος με την αδιάκοπη καθοδήγηση και τις πολύτιμες συμβουλές του κατέστησε δυνατή την ολοκλήρωση της εργασίας, καθώς και την οικογένεια μας για την οικονομική και ηθική υποστήριξη καθ' όλη τη διάρκεια των σπουδών μας.

Περίληψη

Σκοπός της παρούσας εργασίας είναι η παρουσίαση της σχεδίασης, της ανάπτυξης και της κατασκευής ενός εξάποδου ρομπότ για την μοντελοποίηση ενός εξάποδου εντόμου. Πιο συγκεκριμένα, με την κατασκευή του συγκεκριμένου ρομπότ επιδιώκεται να προσομοιωθεί ο τρόπος περπατήματος του εντόμου σε δύσβατες και ανώμαλες επιφάνειες, καθώς και γενικότερα η μοντελοποίηση των φυσικών ικανοτήτων κίνησης του στο χώρο. Το ρομπότ απαρτίζεται από ελεγκτή (Arduino Mega), 22 σερβοκινητήρες οι οποίοι ελέγχουν τα πόδια, το κεφάλι και την ουρά του, κεραίες, μάτια, μπαταρία λιθίου για την αυτονομία του, αισθητήρα προσέγγισης, καθώς και όλα τα απαραίτητα ηλεκτρονικά εξαρτήματα που απαιτούνται για τον έλεγχο του εξ' αποστάσεως μέσω εφαρμογής του συστήματος Android. Ακριβώς επειδή η «ανατομία» του ρομπότ προσομοιάζει με αυτή ενός μυρμηγκιού, θα μπορούσε επίσης να ονομαστεί και Arduino Μυρμήγκι Ρομπότ (Arduino Ant Robot). Η ανά χείρας μελέτη παρουσιάζει το θεωρητικό πλαίσιο της κατασκευής (πορίσματα βιβλιογραφικής επισκόπησης, μεθοδολογία, σχεδίαση και κατασκευή υλικού μέρους και λογισμικού κτλ.) συνοδευόμενο από το κατάλληλο φωτογραφικό υλικό που ελήφθη κατά την κατασκευή του ρομπότ.

Λέξεις-Κλειδιά: Ρομπότ, Εξάποδο, Μυρμήγκι, Android, Arduino

Abstract

The purpose of this research is to present the design, development and construction of an hexapode robot for the modeling of an insect. In particular, the construction of this robot is intended to simulate the way the insect walks on rough and uneven surfaces, and the modeling of its natural motion capacities in general. The robot is made up of a controller (Arduino Mega), 22 servos that control the legs, head and tail, antennas, eyes, lithium battery for its autonomy, proximity sensor, and all the necessary electronic components required for remote control via Android application. Just because the "anatomy" of the robot resembles that of an ant, it could also be called the Arduino Ant Robot. This study presents the theoretical framework of construction (bibliographic survey findings, methodology, design and construction of hardware and software, etc.) accompanied by the appropriate photographic material taken during the construction of the robot.

Keywords: Robot, Hexapod, Ant, Android, Arduino

Πίνακας Περιεχομένων

Κατάλογος Πινάκων

Εισαγωγή

Η κατασκευή εξάποδων ρομπότ με ελεγκτή (Arduino) έχει αποτελέσει το αντικείμενο πολυάριθμων ερευνών τις τελευταίες τρεις δεκαετίες, λόγω του εντεινόμενου ενδιαφέροντος της επιστημονικής κοινότητας για την προσομοίωση του τρόπου περπατήματος ορισμένων εντόμων σε ανώμαλες και δύσβατες επιφάνειες. Ωστόσο, η κίνηση των συγκεκριμένων κατασκευών [1] είναι ιδιαίτερα δυσχερής σε ακανόνιστες εκτάσεις [2].

Ειδικότερα, η κίνηση των ρομπότ με τα πόδια επιτρέπει τον συντονισμό των κινήσεων του μηχανισμού για την ασφαλή πλοήγηση του σε ποικίλο έδαφος (πχ ίσιο, ανώμαλο κτλ). Συνεπώς στις συγκεκριμένες περιπτώσεις ο μηχανισμός μπορεί να θεωρηθεί ανεξάρτητος και αυτόνομος, καθώς δεν απαιτείται ανθρώπινη παρέμβαση για την πλοήγηση του [3]. Όπως είναι φυσικό, τα ζώα (πχ έντομα) προσαρμόζονται προκειμένου να μπορούν να περπατούν σε διαφορετικούς τύπους επιφανειών.

Συναφώς, έχοντας ως στόχο την ανάπτυξη ενός παρόμοιου μηχανισμού, η επιστημονική κοινότητα έχει εμπνευστεί από ορισμένα φυσικά χαρακτηριστικά των ζώων-εντόμων [4], προσπαθώντας να τα μιμηθεί στον μηχανικό σχεδιασμό, τον έλεγχο και την πλοήγηση [5]. Αυτές οι έρευνες έχουν ανακινήσει το επιστημονικό ενδιαφέρον για τη χρήση ποδιών ή τροχών για την κίνηση των ρομπότ.

Η κύρια διαφορά μεταξύ τους είναι ότι η χρήση των ποδιών κρίνεται καταλληλότερη για την κίνηση και την προσαρμογή σε ακανόνιστα εδάφη από ότι η χρήση τροχών, οι οποίοι έχουν περιορισμένο εύρος κίνησης στα συγκεκριμένα εδάφη [6]. Επομένως, η κατανόηση της συμπεριφοράς της μετακίνησης των ζώων παρουσιάζει αυξημένο επιστημονικό ενδιαφέρον.

Αναντίρρητα, μία από τις κύριες προκλήσεις στην ανάπτυξη των ρομπότ είναι ο σχεδιασμός του συστήματος μετακίνησης, ο οποίος περιλαμβάνει την αλληλεπίδραση του με δομές που αποτελούνται από πρισματικές ή περιστροφικές αρθρώσεις, οι οποίες μιμούνται τις λειτουργίες κίνησης που υπάρχουν στη φύση, επιτρέποντας την προσαρμογή σε ανώμαλο έδαφος. Επιπροσθέτως, πρέπει να αντιμετωπιστούν προβλήματα όπως η μηχανική πολυπλοκότητα που υπάρχει στα πόδια των ρομπότ, ο μηχανισμός σταθερότητας, η κατανάλωση ενέργειας, ο συγχρονισμός των συνδέσμων σε κάθε μία από τις αρθρώσεις του ρομπότ, καθώς και ο έλεγχος των βαθμών ελευθερίας που απαιτεί η κίνηση του.

Η θέση των ποδιών σχετικά με την επιφάνεια μετατόπισης είναι σημαντική για τη σταθερότητα του ρομπότ, όπως και το κέντρο βάρους του, και τούτο διότι αν δεν έχουν σωστό συγχρονισμό και αν δεν παρέχουν την απαραίτητη υποστήριξη στο σύστημα βάσης, τότε το ρομπότ θα χάσει την ισορροπία του και θα πέσει, ή οι κινήσεις του θα είναι αναποτελεσματικές, γεγονός που θα προκαλέσει μεγαλύτερη κατανάλωση ενέργειας [7]. Αυτός ο συγχρονισμός θα εξαρτηθεί από τον έλεγχο της κινητικότητας των ποδιών για τη μετατόπισή του, διότι μόνο αν το ρομπότ κινείται μέσα στα καθιερωμένα όρια, μπορούν να αποφευχθούν οι συγκρούσεις μεταξύ των συνδέσμων και συνεπώς μόνο με αυτό τον τρόπο θα διασφαλιστεί ότι το σύστημα δεν θα επηρεαστεί.

Επομένως αποδεικνύεται ότι το μήκος και ο σχεδιασμός των ποδιών είναι κρίσιμος για τη μετακίνηση του ρομπότ, καθώς αυτή εξαρτάται αποκλειστικά από την τροχιά που εφαρμόζεται σε κάθε άρθρωση. Άρα εάν η τροχιά που εφαρμόζουμε σε κάθε άρθρωση επιτρέπει μια ομαλή κίνηση, τότε η σταθερότητα του ρομπότ δεν θα επηρεάζεται από κάποια απότομη κίνηση και επιπροσθέτως αυτό θα μας δίνει τη δυνατότητα να καθορίσουμε την πρόοδο της κίνησης του ρομπότ σε δεδομένη χρονική στιγμή. Επίσης, αν το ρομπότ κινείται εντός των καθορισμένων ορίων, ο κίνδυνος σύγκρουσης αποφεύγεται και το σύστημα θα είναι ασφαλές.

Τα ανωτέρω στοιχεία αποτελούν σημαντικά πορίσματα της βιβλιογραφικής επισκόπησης, τα οποία ουσιαστικά αποδεικνύουν την αυξημένη επιστημονική αξία της παρούσας πτυχιακής. Ειδικότερα, σκοπός αυτής της εργασίας είναι η παρουσίαση της σχεδίασης, της ανάπτυξης και της κατασκευής ενός εξάποδου ρομπότ μυρμηγκιού με μικροελεγκτή (Arduino).

Όπως αναδείχθηκε παραπάνω, το θέμα της πτυχιακής παρουσιάζει αυξημένο ερευνητικό ενδιαφέρον, λόγω της εντεινόμενης προσπάθειας της επιστημονικής κοινότητας για κατασκευή μηχανισμών που προσομοιάζουν-μοντελοποιούν τον τρόπο περπατήματος των εντόμων σε ανώμαλες και δύσβατες επιφάνειες. Η πρωτοτυπία της εργασίας έγκειται αρχικώς στην επιλογή του είδους του ρομπότ (δηλαδή εξάποδο που προσομοιάζει με ένα μυρμήγκι) και κατά δεύτερον στην επιλογή ποδιών έναντι τροχών για την κίνηση του ρομπότ, που όπως διαπιστώσαμε παραπάνω παρουσιάζουν ως συγκριτικό πλεονέκτημα το μεγαλύτερο εύρος κίνησης στις ανώμαλες επιφάνειες. Περαιτέρω, στους στόχους της εργασίας συγκαταλέγεται και η προσπάθεια να αναδειχθεί ότι εν αντιθέσει με αυτά που υποστηρίζονται στη θεωρία, η κατασκευή και λειτουργία των ποδιών στα Arduino ρομπότ δεν παρουσιάζει τόσο μεγάλη πολυπλοκότητα, υπό την προϋπόθεση ότι δίνεται η δέουσα προσοχή στο μήκος και στο σχεδιασμό των ποδιών, στο συντονισμό τους καθώς και στη σωστή τους τοποθέτηση, ώστε να διασφαλίζεται η σταθερότητα του ρομπότ.

Για τους σκοπούς αυτούς η εργασία διαρθρώνεται σε οκτώ κεφάλαια. Πιο συγκεκριμένα, στο πρώτο κεφάλαιο αναλύεται το θεωρητικό πλαίσιο της εργασίας όπου παρουσιάζονται οι ορισμοί των επιστημονικών εννοιών που χρησιμοποιούνται, τα αποτελέσματα παλαιότερων ερευνών, καθώς και οι υποθέσεις και προσδοκίες σχετικά με τα αποτελέσματα της παρούσας έρευνας. Έπειτα, στο κεφάλαιο 2 παρουσιάζεται ο ορισμός του προβλήματος της εργασίας δηλαδή η κατασκευή του εξάποδου μυρμηγκιού ρομπότ, ενώ στο κεφάλαιο 3 παρουσιάζεται η μεθοδολογία που χρησιμοποιείται. Περαιτέρω, στο κεφάλαιο 4 εξετάζεται η σχεδίαση και η κατασκευή του υλικού μέρους του ρομπότ και ειδικότερα τα λειτουργικά του μέρη, το μηχανολογικό σχέδιο, το τυπωμένο και το ηλεκτρονικό κύκλωμα, καθώς και της λίστας των υλικών που χρησιμοποιήθηκαν. Τέλος, στο κεφάλαιο 5 παρουσιάζεται η ανάλυση και η συγγραφή του λογισμικού της κατασκευής, στο κεφάλαιο 6 αναλύονται τα αποτελέσματα της εργασίας, στο κεφάλαιο 7 εξετάζονται τα συμπεράσματα της έρευνας και στο κεφάλαιο 8 παρατίθεται η βιβλιογραφία της εργασίας.

Κεφάλαιο 1: Θεωρητικό Υπόβαθρο

Στο παρόν κεφάλαιο παρουσιάζεται το θεωρητικό πλαίσιο της κατασκευής. Ειδικότερα, στο πρώτο υποκεφάλαιο αναλύονται οι βασικοί ορισμοί των επιστημονικών εννοιών που χρησιμοποιούνται στην εργασία (δηλαδή του εξάποδου ρομπότ-hexapod robot, του Arduino, των σερβοκινητήρων και της πλακέτας τυπωμένου κυκλώματος PCB), στο δεύτερο υποκεφάλαιο παρουσιάζονται συνοπτικά οι λόγοι για τους οποίους κρίνεται επιβεβλημένη η συγκεκριμένη κατασκευή και στο τρίτο υποκεφάλαιο εξετάζονται τα πορίσματα παρόμοιων ερευνών που έχουν ήδη πραγματοποιηθεί και παρατίθενται οι υποθέσεις και προσδοκίες σχετικά με τα αποτελέσματα της κατασκευής μας.

1.1: Ορισμοί επιστημονικών εννοιών

Το εξάποδο ρομπότ είναι ένα μηχανικό όχημα που περπατά σε έξι πόδια. Δεδομένου ότι ένα ρομπότ μπορεί να είναι στατικά σταθερό σε τρία ή περισσότερα πόδια, ένα εξάποδο ρομπότ έχει μεγάλη ευελιξία ως προς το πώς μπορεί να κινηθεί. Περαιτέρω, εάν τα πόδια απενεργοποιηθούν, το ρομπότ μπορεί ακόμα να περπατήσει. Αξίζει μάλιστα να σημειωθεί ότι δεν χρειάζονται όλα τα πόδια του ρομπότ για να διασφαλιστεί η σταθερότητα του. Τα περισσότερα εξάποδα ρομπότ εμπνέονται βιολογικά από τη μετακίνηση των εντόμων.

Το Arduino είναι ένας μικροελεγκτής μονής πλακέτας, δηλαδή μια απλή μητρική πλακέτα ανοικτού κώδικα με ενσωματωμένο μικροελεγκτή και εισόδους/εξόδους, η οποία μπορεί να προγραμματιστεί με τη γλώσσα Wiring[1].

Οι σερβοκινητήρες ή σερβομηχανές (servos) είναι αυτόνομες ηλεκτρικές συσκευές οι οποίες περιστρέφονται ή ωθούν τμήματα μιας μηχανής με μεγάλη ακρίβεια. Όπως είναι προφανές και όπως άλλωστε θα φανεί και στη συνέχεια στην αναλυτική παρουσίαση της κατασκευής μας, οι σερβοκινητήρες διαδραματίζουν καθοριστικό ρόλο στην κίνηση των ρομπότ (πχ κάθε σε κάθε πόδι της κατασκευής μας υπάρχουν τρεις σερβοκινητήρες που το καθοδηγούν).

Τέλος, η πλακέτα τυπωμένου κυκλώματος (PCB) υποστηρίζει μηχανικά και συνδέει ηλεκτρικά ηλεκτρονικά εξαρτήματα χρησιμοποιώντας αγώγιμες τροχιές, ή περισσότερα στρώματα φύλλου από χαλκό που είναι ελασματοποιημένος επάνω ή / και μεταξύ στρωμάτων φύλλου ενός μη αγώγιμου υποστρώματος. Τα συγκεκριμένα εξαρτήματα είναι συγκολλημένα επάνω στην πλακέτα PCB τόσο για να συνδεθούν ηλεκτρικά, όσο και για να τα «στερεωθούν» μηχανικά σε αυτά. Τα συγκεκριμένα στοιχεία θα μας απασχολήσουν ιδιαίτερα στο υποκεφάλαιο 4.4.

1.2: Οι λόγοι εξέτασης του προβλήματος

Η κατασκευή του εξάποδου ρομπότ-μυρμηγκιού κρίνεται επιβεβλημένη ούτως ώστε να προσομοιωθεί ο τρόπος περπατήματος των μυρμηγκιών σε ανώμαλες και δύσβατες επιφάνειες. Επίσης σημαντικό είναι να μοντελοποιηθούν και οι υπόλοιπες ενστικτώδεις κινήσεις του μυρμηγκιού, όπως οι αντανακλαστικές κινήσεις του κεφαλιού του όταν κάποιος πάει να το ακουμπήσει (με τη χρήση αισθητήρα), οι κινήσεις των κεραιών του, αλλά και της ουράς του. Επομένως, ο βασικός λόγος για τον οποίο κρίνεται επιβεβλημένη αυτή η κατασκευή είναι προκειμένου να μοντελοποιηθούν οι πολύ ενδιαφέρουσες φυσικές ικανότητες κίνησης του μυρμηγκιού.

[1] Ουσιαστικά πρόκειται για τη γλώσσα προγραμματισμού C++

1.3: Προηγούμενες έρευνες και ερευνητική υπόθεση της εργασίας

Στη διεθνή βιβλιογραφία έχουν πραγματοποιηθεί πολυάριθμες έρευνες με αντικείμενο την κατασκευή εξάποδων ρομπότ για την μοντελοποίηση του περπατήματος των εντόμων. Πιο συγκεκριμένα, σε έρευνα των A. Mojdehi, M. Alitavoli, A. Darvizeh, H. Rajabi & H. Larijani [8] επιδιώχθηκε η προσομοίωση του περπατήματος της αράχνης Ειδικότερα, με αυτή την κατασκευή επετεύχθη το ρομπότ να μπορεί να διασχίσει τη διαδρομή παρόμοια με το περπάτημα της αράχνης. Η ακριβής μοντελοποίηση του περπατήματος της αράχνης επετεύχθη μέσω της επιλογής των καλύτερων ενεργοποιητών στο ισοδύναμο μοντέλο σύνδεσης των ποδιών της αράχνης και μέσω της σχεδίασης σωστών ελεγκτών για αυτούς τους ενεργοποιητές.

Έπειτα, στην έρευνα των F. Delcomyn & M. Nelson [9] επιδιώχθηκε η κατασκευή ενός ρομπότ για τη μοντελοποίηση του περπατήματος της κατσαρίδας, η οποία φανέρωσε αξιόλογες βελτιώσεις στο σχεδιασμό και στην πραγματική απόδοση των ρομπότ. Τέλος, με την έρευνα των M. Agheli, L. Qu & S. Nestinger [10] κατασκευάστηκε ένα εξάποδο ρομπότ για τη συντήρηση, την επισκευή και τις λειτουργίες σε απομακρυσμένα, ακανόνιστα και επικίνδυνα περιβάλλοντα. Με το συγκεκριμένο ρομπότ ενισχύθηκε σημαντικά ο χώρος λειτουργίας του παραδοσιακού εξάποδου ρομπότ, με την ενσωμάτωση δύο πρόσθετων πρισματικών αρθρώσεων σε κάθε πόδι του συστήματος. Ο κλιμακωτός σχεδιασμός του ρομπότ ενίσχυσε τη διασταύρωση του ακανόνιστου εδάφους με επέκταση του επιτρεπόμενου εύρους των ποδιών του ρομπότ, επιτρέποντας στο ρομπότ να διασχίσει μεγαλύτερα κενά και εμπόδια ή να περάσει από στενά ανοίγματα. Επομένως, αυτή η κατασκευή απέδειξε την ικανότητα των εξάποδων ρομπότ να προσανατολιστούν σε πολλαπλά σημεία κατά μήκος μιας διαδρομής.

Όπως αναδείχθηκε παραπάνω δεν έχει πραγματοποιηθεί κατασκευή ρομπότ που να προσομοιάζει σε μυρμήγκι. Στο σημείο αυτό έγκειται η πρωτοτυπία της εργασίας, η οποία επιδιώκει την κατασκευή ρομπότ-μυρμηγκιού προκειμένου να μιμηθεί τις αξιόλογες ικανότητες του (περπάτημα σε δύσβατες και ανώμαλες επιφάνειες, ισορροπία, περιστροφή κεφαλιού, κίνηση ουράς και κεραιών, δαγκάνες κτλ). Επιπροσθέτως, σημαντική καινοτομία της εργασίας είναι η κατασκευή και χρήση για τον έλεγχο του ρομπότ εφαρμογής στο λογισμικό Android. Τέλος, βασική υπόθεση της εργασίας είναι ότι το κατάλληλο μήκος, ο κατάλληλος σχεδιασμός και η κατάλληλη τοποθέτηση των ποδιών στο κύριο σώμα του ρομπότ (τοποθέτηση σε ειδικό σημείο ώστε να μην περιορίζεται το εύρος κίνησης του ρομπότ) αφ’ ενός δεν παρουσιάζει τόσο μεγάλη μηχανική πολυπλοκότητα και αφ’ ετέρου διασφαλίζει τη σταθερότητα και την αδιάλειπτη κίνηση του μηχανισμού.

Κεφάλαιο 2: Ορισμός Προβλήματος

Με αυτή την εργασία επιδιώκεται η κατασκευή ενός εξάποδου Arduino ρομπότ (δηλαδή ρομπότ με ελεγκτή) που λόγω της «ανατομίας» του προσομοιάζει με ένα μυρμήγκι. Ειδικότερα, το συγκεκριμένο ρομπότ έχει 6 πόδια (κάθε ένα από τα οποία έχει 3 σερβοκινητήρες), ουρά, κεφάλι, κεραίες, δαγκάνες, αλλά και λειτουργικά μάτια. Σκοπός είναι να μοντελοποιηθεί το σύνολο των κινήσεων του μυρμηγκιού, από τον τρόπο περπατήματος μέχρι και τις αντανακλαστικές κινήσεις του όταν κάποιος πλησιάζει το κεφάλι του ή τις δαγκάνες του.

Κεφάλαιο 3: Μεθοδολογία

Για τον έλεγχο του ρομπότ δημιουργήθηκε μία εφαρμογή Android η οποία διαθέτει τέσσερα κουμπιά, μέσω των οποίων μπορούν να δοθούν εντολές στο ρομπότ είτε να προχωρήσει προς τα εμπρός ή προς τα πίσω, καθώς και να στρίψει αριστερά ή δεξιά. Έκτος από αυτές τις βασικές κινήσεις, το ρομπότ μπορεί επίσης να κινήσει το κεφάλι αλλα΄και την ουρά του, να δαγκώσει, να «μαγκώσει» με τις δαγκάνες του, να αρπάξει αλλά ακόμη και να ρίξει πράγματα, καθώς και να επιτεθεί. Η επιλογή της συγκεκριμένης μεθόδου επελέγη καθώς κρίθηκε κατάλληλη ώστε να μπορούν να δίνεται με μεγάλη ευκολία και αποτελεσματικότητα ένας όγκος διαφορετικών κινήσεων (κίνηση εμπρός, πίσω, δεξιά, αριστερά, επίθεση, δάγκωμα κτλ.).

Περαιτέρω το ρομπότ έχει λειτουργικά μάτια (τα λειτουργικά μέρη της κατασκευής παρουσιάζονται αναλυτικά στο αμέσως επόμενο κεφάλαιο), καθώς και ειδικώς σχεδιασμένο κεφάλι ώστε να «χωράει» ένας υπερηχητικός αισθητήρας. Μέσω αυτής της μεθόδου επιτυγχάνεται η αντίδραση του ρομπότ, το οποίο αν προσπαθήσουμε να αγγίξουμε το κεφάλι του ή να βγάλουμε το χέρι μας κοντά στον αισθητήρα, αυτό να προετοιμαστεί για επίθεση. Σε περίπτωση που υποχωρήσουμε τότε το ρομπότ θα απορρίψει την επίθεση, ενώ αν πλησιάσουμε το χέρι μας το ρομπό θα επιτεθεί και θα μας δαγκώσει. Με τη μεθοδολογία που προεξετέθη επετεύχθη η κατασκευή να μπορεί να εκτελέσει τις ανωτέρω κινήσεις. Τέλος, ο σχεδιασμός του εξάποδου μυρμηγκιού-ρομπότ έγινε με τη χρήση τρισδιάστατου λογισμικού μοντελοποίησης (3D Modeling Software), ενώ για την εκτύπωση των εξαρτημάτων της κατασκευής χρησιμοποιήθηκε ο εκτυπωτής 3D Creal CR-10.

Κεφάλαιο 4: Σχεδίαση και Κατασκευή Υλικού Μέρους (Hardware)

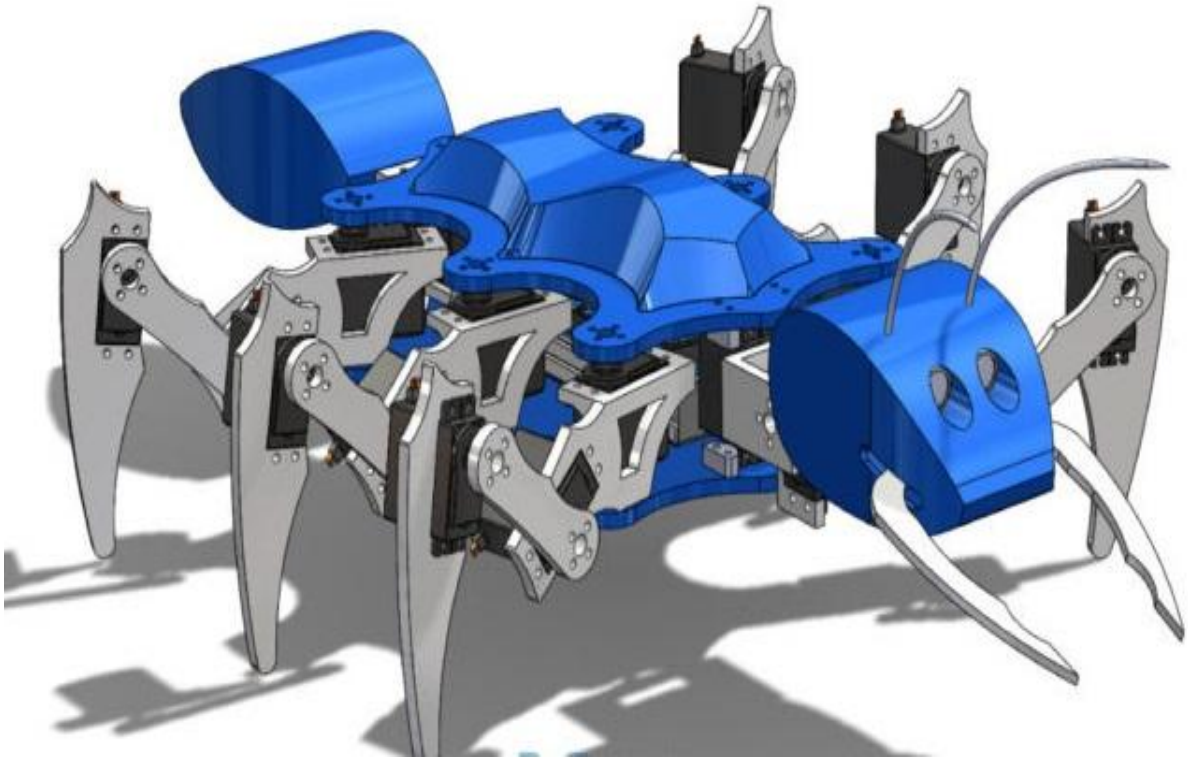
Στο συγκεκριμένο κεφάλαιο παρουσιάζεται η σχεδιάση και κατασκευή του Υλικού Μέρους του ρομπότ. Ειδικότερα, στο πρώτο υποκεφάλαιο παρουσιάζονται τα λειτουργικά μέρη της κατασκευής τα οποία συνοδεύονται με το απαραίτητο φωτογραφικό υλικό, καθώς και την περιγραφή της λειτουργίας και αλληλεπίδρασης τους, στο δεύτερο υποκεφάλαιο παρουσιάζεται το μηχανολογικό σχέδιο της κατασκευής, στο τρίτο υποκεφάλαιο το ηλεκτρονικό κύκλωμα, στο τέταρτο υποκεφάλαιο το τυπωμένο κύκλωμα, στο πέμπτο υποκεφάλαιο η λίστα των υλικών που χρησιμοποιήθηκαν και στο έκτο υποκεφάλαιο η διαδικασία εκτύπωσης και συναρμολόγησης των εξαρτημάτων.

4.1: Λειτουργικά Μέρη Κατασκευής

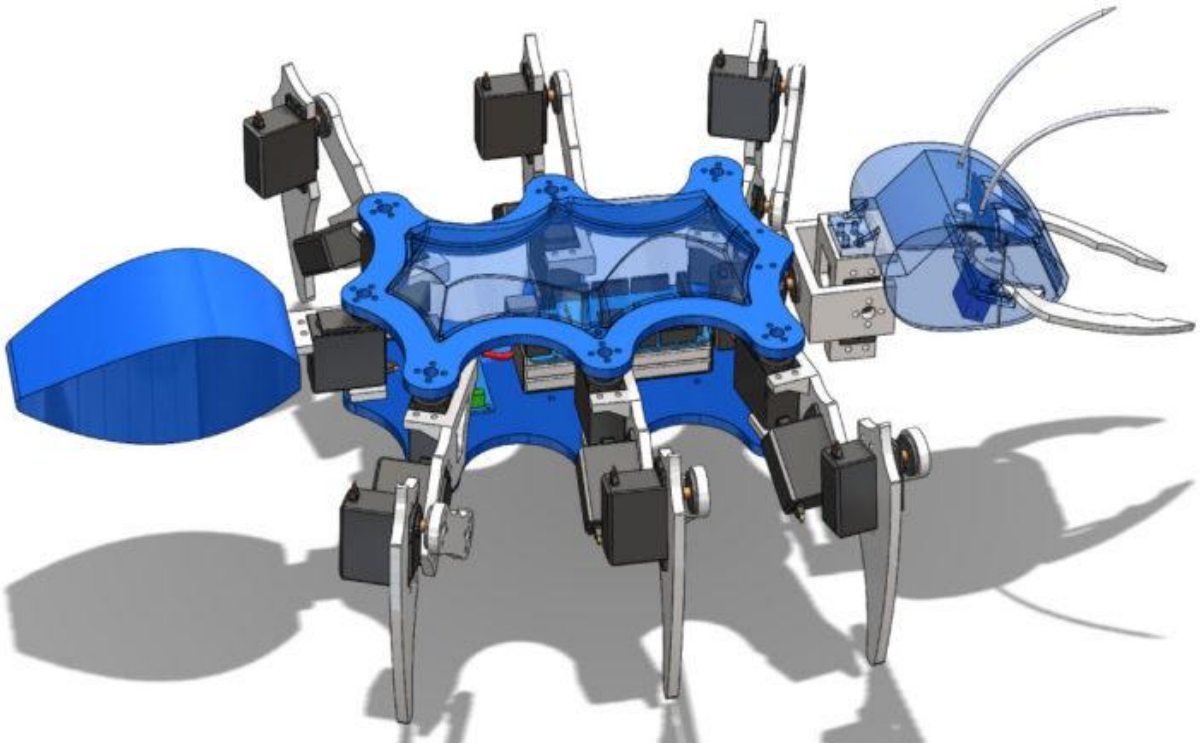
Το ρομπότ έχει έξι πόδια και κάθε ένα από αυτά αποτελείται από τρεις σερβοκινητήρες. Κατά συνέπεια για τη συγκεκριμένη κατασκευή χρειάστηκαν 18 σερβοκινητήρες. Εν προκειμένω χρησιμοποιήθηκαν σερβοκινητήρες τύπου MG996R. Επιπροσθέτως, στο πίσω μέρος του ρομπότ υπάρχει η ουρά η οποία καθοδηγείται-ελέγχεται επίσης από σερβοκινητήρα MG996R. Το τελευταίο λειτουργικό μέρος του ρομπότ είναι το κεφάλι του μυρμηγκιού, το οποίο μπορεί να πραγματοποιήσει δύο κινήσεις, δηλαδή περιστροφή ή κλίση και αυτές οι κινήσεις καθοδηγούνται-ελέγχονται επίσης από δύο σερβοκινητήρες. Επομένως για τη συγκεκριμένη κατασκευή χρειαστήκαμε συνολικά 21 σερβοκινητήρες τύπου MG996R, καθώς και ένα μικρό σερβοκινητήρα τύπου SG90 προκειμένου να ελέγχονται οι δαγκάνες του μυρμηγκιού. Στο κεφάλι του ρομπότ περιλαμβάνονται και τα μάτια του μυρμηγκιού, που σχεδιάστηκαν κατάλληλα ούτως ώστε να χωρέσουν έναν υπερηχητικό αισθητήρα HC-SR04.

4.2: Μηχανολογικό Σχέδιο

Στο παρόν υποκεφάλαιο παρουσιάζεται το μηχανολογικό σχέδιο που χρησιμοποιήθηκε για την κατασκευή.

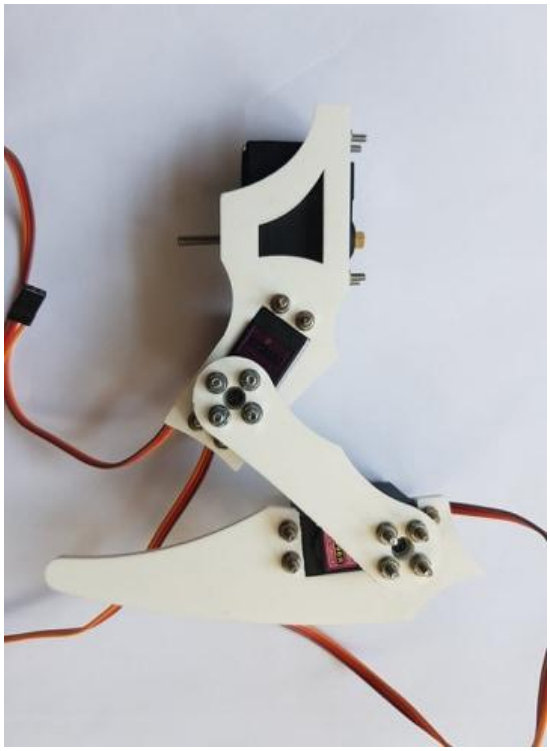


Εικόνα 1: Μηχανολογικό σχέδιο [11]

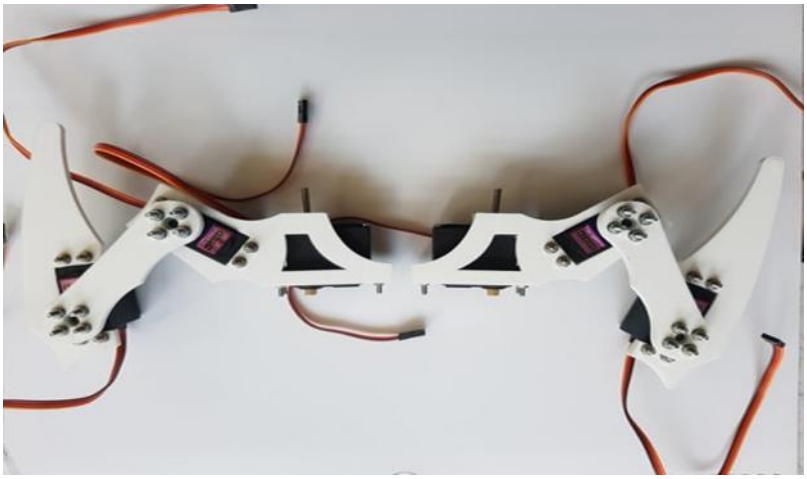


Εικόνα 2: Μηχανολογικό σχέδιο [11]

Στις παρακάτω εικόνες που ελήφθησαν κατά τα πρώτα στάδια της συνδεσμολογίας του ρομπότ διακρίνονται τα πόδια του ρομπότ, το είδος του χρησιμοποιούμενου σερβοκινητήρα, η σύνδεση των «αρθρώσεων» των ποδιών, η βάση αλλά και το κεφάλι του μυρμηγκιού.



Εικόνα 3: Όψη ενός ποδιού του ρομπότ



Εικόνα 4: Ολοκληρωμένη όψη δύο ποδιών του ρομπότ

Σε αυτή την εικόνα διακρίνονται οι τρεις σερβοκινητήρες του ποδιού.

Σε αυτή την εικόνα διακρίνονται συνδεδεμένα τα δύο πόδια του ρομπότ.

Εικόνα 5: Σερβοκινητήρας του ποδιού

Εικόνα 6: Κάτω όψη της βάσης

Σε αυτή την εικόνα διακρίνεται εναργέστερα ο τύπος του σερβοκινητήρα που χρησιμοποιήθηκε, αλλά και ο τρόπος τοποθέτησης του στο πόδι.

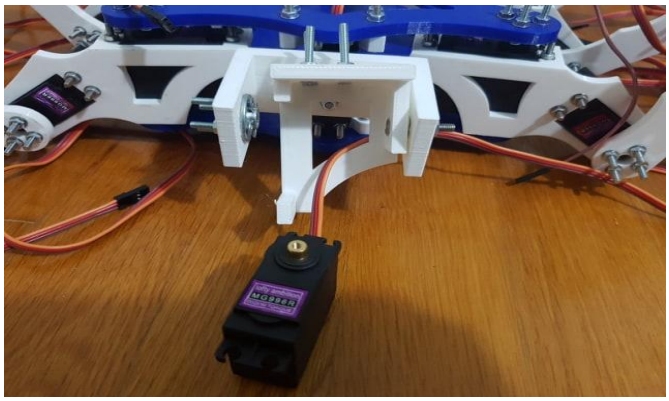
Ακολουθεί η βάση του ρομπότ που όπως διευκρινίστηκε διαδραματίζει καθοριστικό ρόλο για την ισορροπία του μηχανισμού.

Εικόνα 7: Σύνδεση βάσης-ποδιών

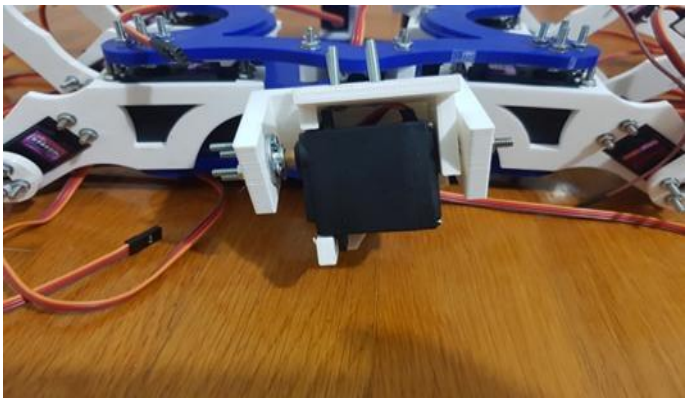
Εικόνα 8: Όψη της βάσης

Εικόνα 9: Εμπρόσθια όψη της βάσης

Εικόνα 10: Ουρά του ρομπότ

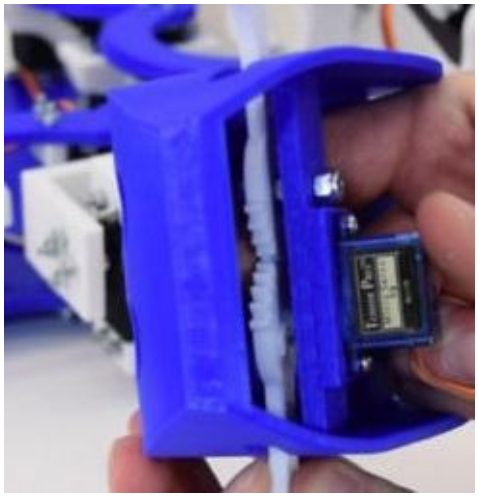


Εικόνα 11: Η σύνδεση του σερβοκινητήρα με την ουρά

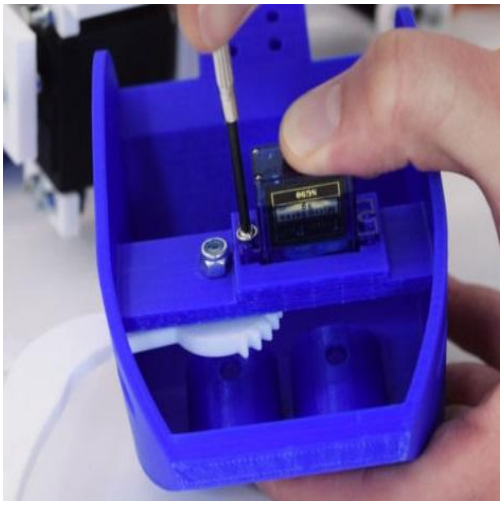


Εικόνα 12: Ο σερβοκινητήρας της ουράς

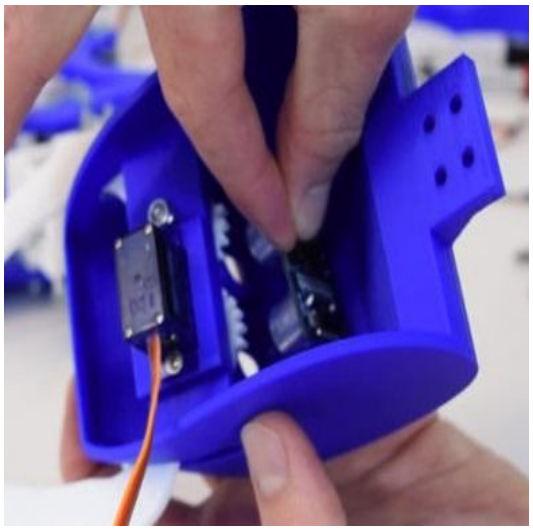
Στις κατωτέρω εικόνες παρουσιάζονται το κεφάλι, τα μάτια και οι διαγκάνες του μυρμηγκιού.



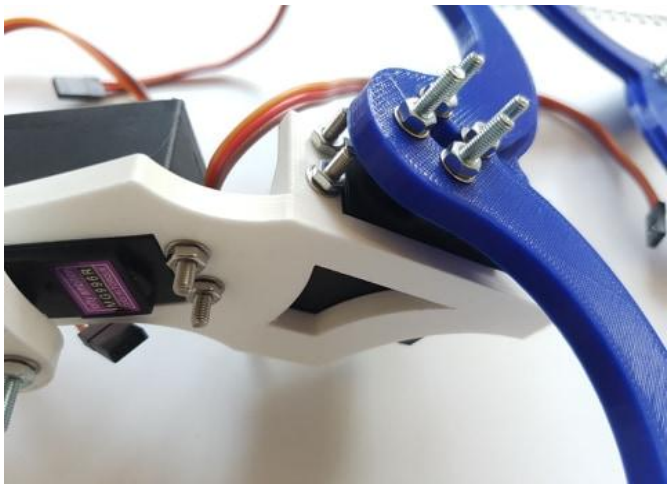
Εικόνα 13: Το κεφάλι του ρομπότ



Εικόνα 14: Η τοποθέτηση της δαγκάνας στο κεφάλι

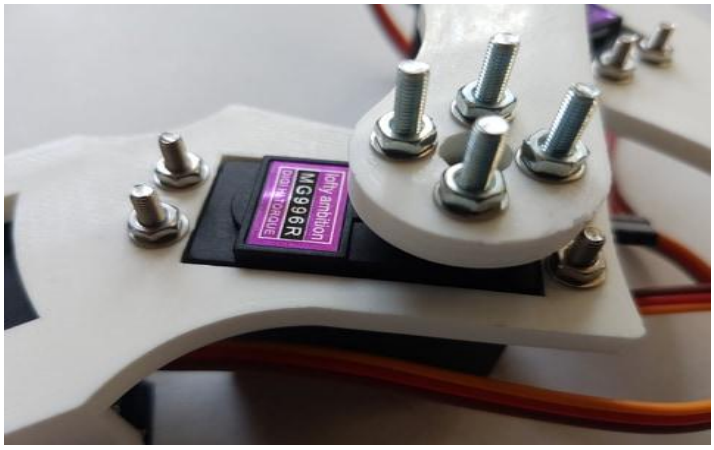


Εικόνα 15: Ο σερβοκινητήρας του κεφαλιού



Εικόνα 16: Η σύνδεση των ποδιών με τη βάση του ρομπότ

Η σύνδεση των ποδιών με τη βάση του ρομπότ αποτελεί ένα από τα πιο κρίσιμα στάδια της κατασκευής, για τη διασφάλιση της ισορροπίας του μηχανισμού αλλά και του κατάλληλου εύρους κίνησης.

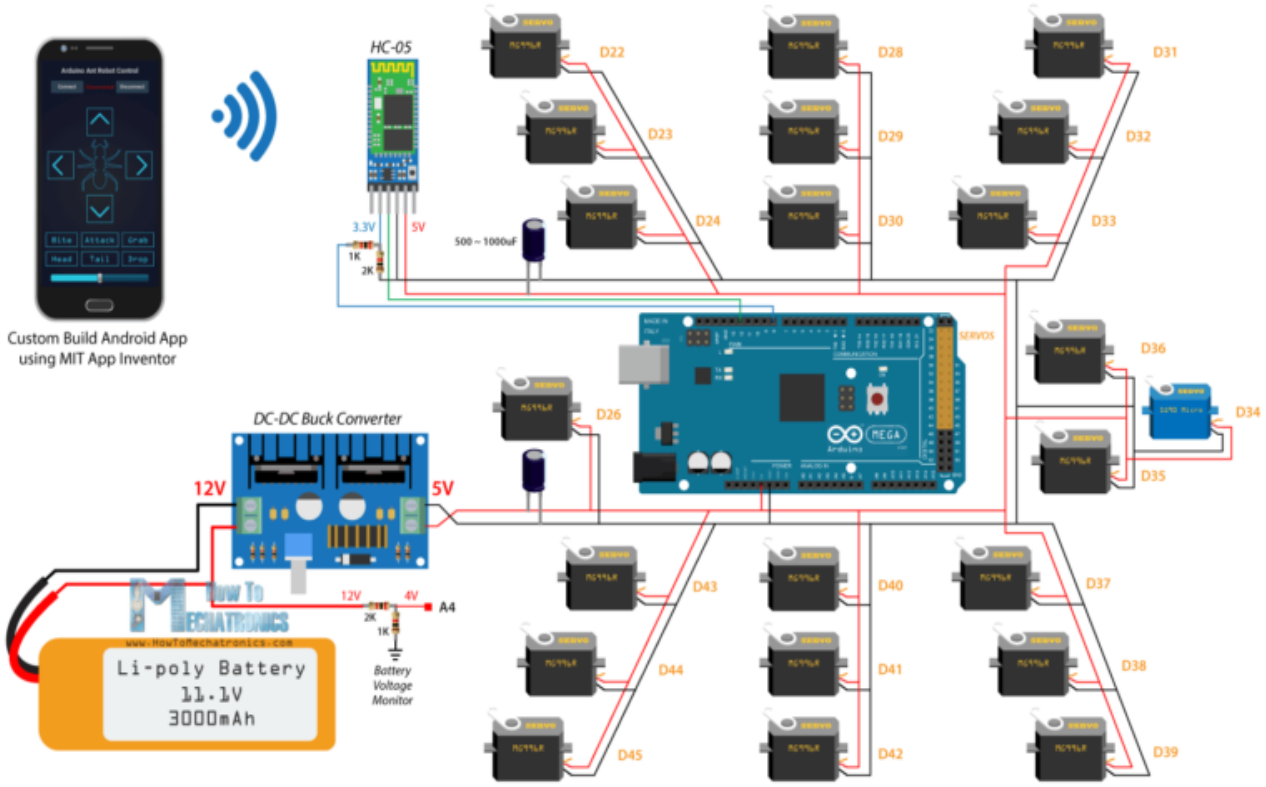


Εικόνα 17: Η σύνδεση των αρθρώσεων των ποδιών και ο χρησιμοποιούμενος σερβοκινητήρας

4.3: Ηλεκτρονικό κύκλωμα

Εκτός από τους 22 προαναφερθέντες σερβοκινητήρες, χρησιμοποιήθηκε μια μονάδα Bluetooth HC-05 για την επικοινωνία του smartphone με πυκνωτές και αντιστάτες. Ο εγκέφαλος του ρομπότ είναι ο ελεγκτής Arduino που μπορεί να ελέγξει περισσότερους από 12 σερβοκινητήρες.

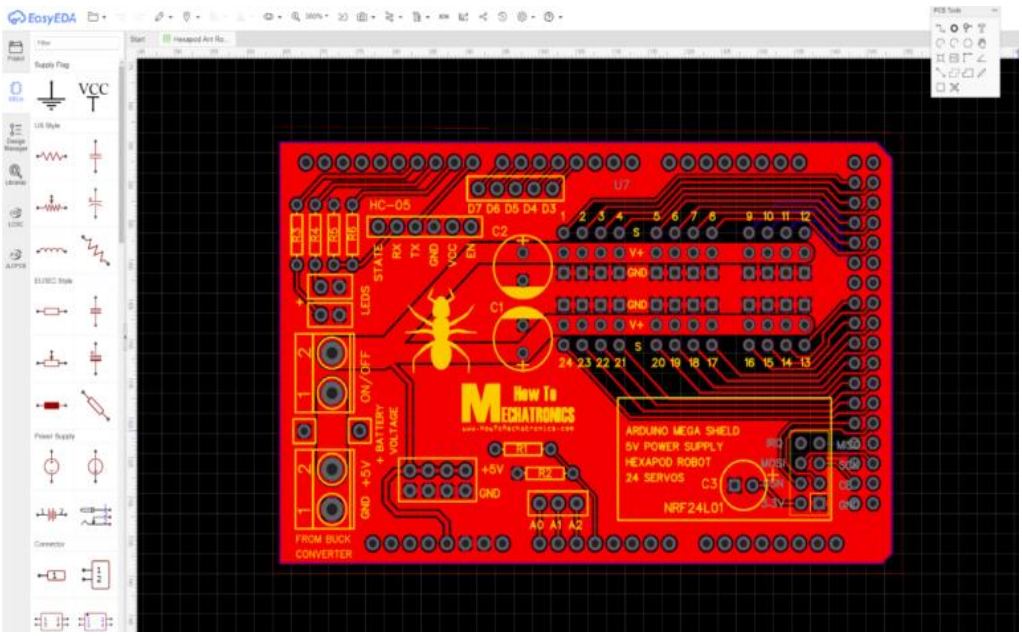
Περαιτέρω, για την τροφοδοσία του ρομπότ χρησιμοποιήθηκε μια μπαταρία λιθίου 3S LiPo που έχει τάση περίπου 12V. Και τούτο διότι οι μπαταρίες LiPo μπορούν να χειριστούν υψηλότερη ποσότητα ρεύματος, γεγονός που τις καθιστά κατάλληλες για την παρούσα κατασκευή, διότι αν όλοι οι σερβοκινητήρες είναι ενεργοποιημένοι ταυτόχρονα με πλήρες φορτίο, μπορούν να αντλήσουν περίπου 10 αμπέρ ρεύματος. Ωστόσο, η τάση λειτουργίας του σερβοκινητήρα είναι περιορισμένη στους 4,8-7,2V, πράγμα που σημαίνει ότι έπρεπε να χρησιμοποιηθεί μετατροπέας DC για να μετατραπούν τα 12V σε 5V. Ακόμα και σε περίπτωση που χρησιμοποιηθεί μπαταρία 2S LiPo που έχει τάση περίπου 7,4V-8,4V, όταν φορτωθεί πλήρως απαιτείται να χρησιμοποιηθεί μετατροπέας. Ο μετατροπέας που χρησιμοποιήθηκε μπορεί να χειριστεί μέχρι και 8 ενισχυτές ρεύματος.



Εικόνα 18: Ηλεκτρονικό κύκλωμα [11]

4.4: Τυπωμένο κύκλωμα

Για το σχεδιασμό του τυπωμένου κυκλώματος χρησιμοποιήθηκε το δωρεάν λογισμικό σχεδιασμού ηλεκτρονικών κυκλωμάτων EasyEDA. Το συγκεκριμένο κύκλωμα συνδέεται απευθείας με την πλάκα Arduino. Στο κύκλωμα υπάρχουν πολλές συνδέσεις ψηφιακών και αναλογικών ακίδων, συνδέσεις 5V και γείωσης, δύο συνδέσεις LED, καθώς και μια σύνδεση για την παρακολούθηση της τάσης της μπαταρίας. Η τάση μπαταρίας 12V θα περάσει από ένα διαιρέτη τάσης αποτελούμενο από δύο αντιστάσεις R1 και R2, οι οποίες θα μειώσουν την τάση κάτω από 5V, έτσι ώστε ο αναλογικός πείρος να μπορεί να το διαβάσει με ασφάλεια. Με αυτόν τον τρόπο θα γνωρίζουμε πότε θα χρειαστεί να επαναφορτιστεί η μπαταρία.



Εικόνα 19: Τυπωμένο κύκλωμα PCB [11]

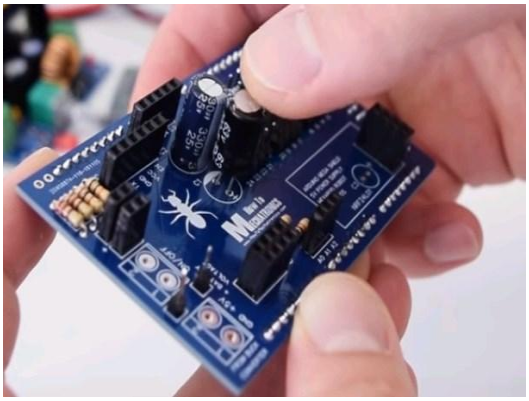
4.4.1: Συναρμολόγηση του PCB

Η συναρμολόγηση ξεκίνησε με τη συγκόλληση αρσενικών κεφαλίδων ακροδεκτών στο PCB με την πλάκα του ελεγκτή (Arduino). Αφού τοποθετηθούν οι κεφαλίδες των ακίδων στην κάτω πλευρά έγινε η σύνδεση με το PCB. Έπειτα, εγκαταστάθηκαν οι κεφαλίδες για όλες τις συνδέσεις και χρησιμοποιήθηκε η ίδια μέθοδος για να συγκολληθούν όλες οι καρφίτσες στο PCB. Τέλος, εγκαταστάθηκαν οι αντιστάτες, οι πυκνωτές και τα τερματικά.

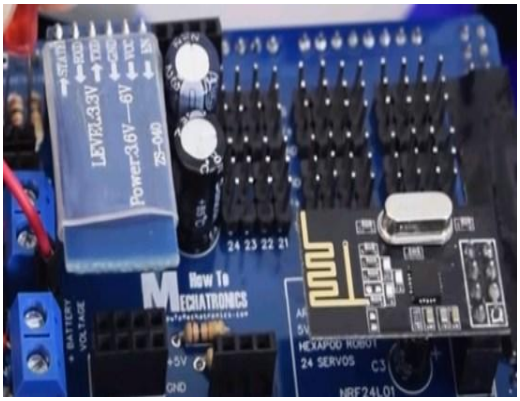
Εν συνεχεία ρυθμίστηκε η τάση εξόδου του μετατροπέα σε 5V. Περαιτέρω, προσετέθη ένας διακόπτης τροφοδοσίας στην είσοδο και συνδέθηκε η μπαταρία 12V με τον ακροδέκτη 12V του PCB που θα χρησιμοποιηθεί για την παρακολούθηση της τάσης της μπαταρίας.

Το επόμενο στάδιο περιελάμβανε την εισαγωγή των ηλεκτρονικών εξαρτημάτων μεταξύ των δύο πλακών. Αρχικώς μπήκε η μπαταρία δένοντας τη με μία ταινία, ενώ στην κορυφή της τοποθετήθηκε η πλάκα του ελεγκτή μαζί με το PCB. Εν συνεχεία, τοποθετήθηκε μία ενδεικτική λυχνία-ενδείκτη για το πότε πρέπει να επαναφορτιστεί η μπαταρία ή σε περίπτωση που η τάση πέσει κάτω από τα 11V.

Οι παρακάτω φωτογραφίες ελήφθησαν κατά τη συναρμολόγηση του PCB:



Εικόνα 20: Συναρμολόγηση PCB (1)



Εικόνα 21: Συναρμολόγηση PCB (2)

4.5: Λίστα υλικών

Για τη συγκεκριμένη κατασκευή χρησιμοποιήθηκαν 21 σερβοκινητήρες τύπου MG996R, καθώς και ένας μίκρο-σερβοκινητήρας SG90 για τον έλεγχο των δαγκάνων του ρομπότ. Επιπροσθέτως, χρησιμοποιήθηκε ένας υπερηχητικός αισθητήρας HC-SR04, ο οποίος τοποθετήθηκε στο κεφάλαι του ρομπότ ώστε να «αισθάνεται» τις εχθρικές κινήσεις και να επιτίθεται. Όλα τα μέρη του ρομπότ (δηλαδή τα πόδια, η ουρά και το κεφάλι συνδέθηκαν σε δύο πλάκες που αποτέλεσαν τη βάση του μηχανισμού, ενώ στην κορυφή χρησιμοποιήθηκε ένα ομοιόμορφο «κάλυμμα» προκειμένου να κρύβει τις καλωδιώσεις και τις μπαταρίες μεταξύ των δύο πλακών.

Έπειτα, χρησιμοποιήθηκε ο εκτυπωτής Creality CR-10 3D για την εκτύπωση των εξαρτημάτων της κατασκευής και βέβαια το τρισδιάστατο λογισμικό μοντελοποίησης 3D Modeling Software. Όσον αφορά τώρα τη συναρμολόγηση των μερών του ρομπότ χρησιμοποιήθηκαν μπουλόνια και «παξιμάδια» M3, καθώς και ελατηριοειδείς ροδέλες.

4.6: Εκτύπωση και συναρμολόγηση των εξαρτημάτων

Για την εκτύπωση των εξαρτημάτων του εξάποδου ρομπότ χρησιμοποιήθηκε το τρισδιάστατο λογισμικό μοντελοποίησης (3D Modeling Software) και ο εκτυπωτής 3D Creal CR-10. Αναμφισβήτητα, το πιο δύσκολο μέρος για εκτύπωση ήταν το κεφάλι του μυρμηγκιού, και τούτο διότι έπρεπε να εκτυπωθεί σε ένα ενιαίο έντυπο. Για το σκοπό αυτό χρησιμοποιήθηκε μία γωνία υποστήριξης 60 μοιρών, καθώς και ορισμένοι επιπρόσθετοι μηχανισμοί υποστήριξης. Σε κάθε περίπτωση το κεφάλι του μυρμηγκιού εκτυπώθηκε ακριβώς όπως χρειαζόταν και γενικότερα ο εκτυπωτής 3D Creal CR-10 έκανε εξαιρετική δουλειά.

Έπειτα από την εκτύπωση των μερών του Ρομπότ ακολούθησε η συναρμολόγηση του Εξάποδου Μυρμηγκιού. Ειδικότερα, αρχικά συναρμολογήθηκαν τα πόδια. Προκειμένου οι σερβοκινητήρες να «στερεωθούν» στα τυπωμένα μέρη χρησιμοποιήθηκαν μπουλόνια και «παξιμάδια» Μ3, καθώς και ελατηριοειδείς ροδέλες. Το μήκος των βιδών που χρησιμοποιήθηκαν ήταν τουλάχιστον 12 και των μπουλονιών 16 mm.

Για ολόκληρη την κατασκευή χρειάστηκαν συνολικά 200 βίδες. Περαιτέρω, για τη σύνδεση των ζεύξεων μεταξύ τους χρησιμοποιήθηκαν τα στρογγυλά κέρατα που έρχονται μαζί με το πακέτο των σερβοκινητήρων ως αξεσουάρ. Ωστόσο, χρειάστηκε να τρυπήσουμε με τρυπάνι τρύπες μήκους 3 mm σε κάθε μία από αυτές, ούτως ώστε να μπορούν να διέλθουν τα μπουλόνια. Εναλλάκτικα, μπορούν να χρησιμοποιηθούν μεταλλικά στρογγυλά κέρατα τα οποία έχουν σπειρώματα Μ3 τα οποία μπορούν να χρησιμοποιηθούν χωριστά.

Κατά τη σύνδεση των σερβοκινητήρων ήταν πολύ σημαντικό να διασφαλίσουμε ότι όλοι οι σερβοκινητήρες συνδέονται ακριβώς στην ίδια θέση, ούτως ώστε να έχουν όλο το εύρος κίνησης. Στο συγκεκριμένο σημείο πρέπει να επισημανθεί ότι οι σερβοκινητήρες έχουν στην κορυφή τους μία τριγωνική υποστήριξη η οποία για τους σκοπούς της κατασκευής αφαιρέθηκε. Για το σκοπό αυτό χρησιμοποιήθηκε σουγιάς. Επίσης κατ' αυτό τον τρόπο κατέστη δυνατό να τοποθετηθούν φλας με τα τυπωμένα μέρη στους σερβοκινητήρες. Αξίζει τέλος να σημειωθεί ότι πριν την τοποθέτηση του τρίτου σερβοκινητήρα στη θέση του, αρχικώς τοποθετήθηκε ένα μπουλόνι Μ4 για τη σύνδεσης του ποδιού με την πλάκα βάσης.

Αφότου ολοκληρώθηκε η κατασκευή των ποδιών του ρομπότ ακολούθησε η τοποθέτηση τους στο σώμα του μυρμηγκιού. Απαραίτητο κρίθηκε να εξασφαλίσουμε τα προαναφερθέντα «στρογγυλά κέρατα» χρησιμοποιώντας την ίδια μέθοδο με προηγουμένως, δηλαδή με βίδες και «παξιμάδια» Μ3. Εν συνεχεία συνδέθηκαν με τους άξονες των σερβοκινητήρων με τη βοήθεια μπουλονιών, ρυθμίζοντας τη θέση τους ακριβώς στη μέση. Αυτό κρίθηκε απαραίτητο ούτως ώστε αφ' ενός να διασφαλιστεί όλο το εύρος της κίνησης των σερβοκινητήρων και αφ' ετέρου να διευκολύνει τον κατοπινό προγραμματισμό του Arduino.

Κεφάλαιο 5: Ανάλυση Και Συγγραφή Λογισμικού (Software)

Στο συγκεκριμένο κεφάλαιο παρουσιάζεται ο κώδικας του προγράμματος που χρησιμοποιήθηκε στην κατασκευή του λογισμικού (software).

5.1: Επεξήγηση Λογισμικού

Στο πρόγραμμα εγκατάστασης προετοιμάστηκε το σύστημα επικοινωνίας Bluetooth, οι λειτουργίες των ακίδων για τον υπερηχητικό αισθητήρα, και καθορίστηκαν οι ακροδέκτες που συνδέθηκαν με τους σερβοκινητήρες.

```
#include <Servo.h>
#include <SoftwareSerial.h>

#define trigPin 7
#define echoPin 6
#define ledB 10

SoftwareSerial Bluetooth(12, 9); // Arduino(RX, TX) - HC-05 Bluetooth (TX, RX)

// Create servo object
Servo s24;
Servo s23;
Servo s22;
```

```
void setup() {
  Serial.begin(38400);
  Bluetooth.begin(38400); // Default baud rate of the Bluetooth module
  Bluetooth.setTimeout(1);
  delay(20);
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
  pinMode(ledB, OUTPUT);
  // Head
  s15.attach(36, 600, 2400);
  s14.attach(35, 600, 2400);
  s13.attach(34, 600, 2400); //grip
  // Tail
  s5.attach(26, 600, 2400); // Tail
  // Leg 4
  s10.attach(31, 600, 2400);
  s11.attach(32, 600, 2400);
  s12.attach(33, 600, 2400); //rot
  // Leg 5
  s7.attach(28, 600, 2400);
  s8.attach(29, 600, 2400);
  s9.attach(30, 600, 2400); //rot
  // Leg 6
  s1.attach(22, 600, 2400);
  s2.attach(23, 600, 2400);
  s3.attach(24, 600, 2400); //rot
  // Leg 1
  s18.attach(39, 600, 2400);
  s17.attach(38, 600, 2400);
  s16.attach(37, 600, 2400); //rot
  // Leg 2
  s21.attach(42, 600, 2400);
  s20.attach(41, 600, 2400);
  s19.attach(40, 600, 2400); //rot
  // Leg 3
  s24.attach(45, 600, 2400);
  s23.attach(44, 600, 2400);
  s22.attach(43, 600, 2400); //rot
```

Στη συνέχεια, χρησιμοποιώντας τις λειτουργίες write (), μεταφέραμε τους σερβοκινητήρες στην αρχική τους θέση. Εδώ μπορεί να συντελεστεί η βαθμονόμηση του σερβοκινητήρα. Και τούτο διότι δεν θα είναι εύκολο να ρυθμιστεί ο σερβοκινητήρας στην ακριβή του θέση κατά τη συναρμολόγηση του ρομπότ, ωστόσο σε αυτό το σημείο να είναι δυνατό να γίνουν προσαρμογές και να προγραμματιστεί η κίνηση του ρομπότ.

```
// == Move to initial position
  // Head
  s15.write(72);
  s14.write(50);
  s13.write(90); // Grip

  s5.write(65); // Tail

  // Leg 4
  s10.write(65);
  s11.write(35);
  s12.write(40);
  // Leg 5
  s7.write(80);
  s8.write(50);
  s9.write(25);
  // Leg 6
  s1.write(90);
  s2.write(45);
  s3.write(60);

  // Leg 1
  s18.write(60);
  s17.write(90);
  s16.write(100);
  // Leg 2
  s21.write(50);
  s20.write(85);
  s19.write(75);
  // Leg 3
  s24.write(50);
  s23.write(80);
```

Όταν πρόκειται για το προγραμματισμό ενός εξάποδου ρομπότ, υπάρχουν διάφοροι τρόποι, όπως η χρήση μίας κινήτριας προς τα εμπρός ή αντίστροφα. Αυτές οι μέθοδοι περιλαμβάνουν μαθηματικούς τύπους όπου η θέση κάθε άρθρωσης υπολογίζεται με βάση τις εισόδους για την επιθυμητή τελική θέση του ρομπότ.

```
void moveLeg1() {
  // Swign phase - move leg though air - from initial to final position
  // Rise the leg
  if (i1L1 <= 10) {
    s18.write(60 - i1L1 * 2);
    s17.write(90 - i1L1 * 3);
    i1L1++;
  }
  // Rotate the leg
  if (i2L1 <= 30) {
    s16.write(100 - i2L1);
    i2L1++;

  }
  // Move back to touch the ground
  if (i2L1 > 20 & i3L1 <= 10) {
    s18.write(40 + i3L1 * 2);
    s17.write(60 + i3L1 * 3);
    i3L1++;
  }
  // Stance phase - move leg while touching the ground
  // Rotate back to initial position
  if (i2L1 >= 30) {
    s16.write(70 + i4L1);
    i4L1++;
    l1status = HIGH;
  }
  // Reset the counters for repeating the process
  if (i4L1 >= 30) {
    i1L1 = 0;
    i2L1 = 0;
    i3L1 = 0;
```

```
// Move forward
  if (m == 2) {
    moveLeg1();
    moveLeg3();
    moveLeg5();
    if (l1status == HIGH) {
      moveLeg2();
      moveLeg4();
      moveLeg6();
    }
  }
```

Όπως διαπιστώνεται και ανωτέρω, δημιουργήθηκε μία ξεχωριστή, προσαρμοσμένη λειτουργία για τη μετακίνηση κάθε σκέλους. Ειδικότερα, η κίνηση του ποδιού περιλαμβάνει δύο φάσεις, την ταλάντευση και στάση. Πιο συγκεκριμένα, στη φάση της ταλάντωσης το πόδι μετακινείτα από την αρχική σε μία τελική θέση, ενώ στη φάση της στάσης το πόδι μετακινείται από την τελική θέση πίσω στην αρχική. Με αυτόν τον τρόπο το σώμα του ρομπότ προχωράει.

Έτσι, αρχικώς οι δύο εξωτερικοί σερβοκινητήρες ανυψώνουν το πόδι, ενώ ο τρίτος σερβοκινητήρας που συνδέεται με το σώμα αρχίζει να περιστρέφεται σε μία συγκεκριμένη κατεύθυνση. Όταν δε ο τρίτος σερβοκινητήρας βρίσκεται στα 10 βήματα πριν σταματήσει να περιστρέφεται, αρχίζουν να μετακινούνται οι δύο εξωτερικοί σερβοκινητήρες (στην πλάτη) στην ίδια θέση, ούτως ώστε να ακουμπήσει στο έδαφος.

Αυτό ολοκληρώνει τη φάση της ταλάντευσης που αναφέρθηκε παραπάνω ή το πόδι μετακινείται από την αρχική στην τελική θέση. Εν συνεχεία, περιστρέφεται ο τρίτος σερβοκινητήρας από την τελική στην αρχική θέση ώστε να ολοκληρωθεί η φάση της στάσης. Αφού το πόδι του ρομπότ εκτελέσει έναν κύκλο, τότε οι μετρητές επαναφέρονται και το πόδι θα επαναλάβει τον κύκλο ξανά και ξανά. Κάθε επανάληψη ή βήμα του ρομπότ εκτελείται με ένα βήμα καθυστέρησης ο οποίος ελέγχει την ταχύτητα των σερβοκινητήρων. Τέλος, με τον ίδιο τρόπο, δηλαδή με τη χρήση μετρητών για την παρακολούθηση των βημάτων, προγραμματίστηκαν και οι υπόλοιπες λειτουργίες, όπως επί παραδείγματι η μετακίνηση του κεφαλιού, η κίνηση της ουράς κτλ.

5.2: Κώδικας C/C++ Λογισμικού

Παρακάτω παρατίθεται ο κώδικας C/C++ του λογισμικού του εξάποδου ρομπότ για τον μικροελεγκτή Arduino Mega.

```
/*
    === Arduino Ant Robot / Hexapod ===
    by Dejan, https://howtomechatronics.com
*/

#include <Servo.h>
#include <SoftwareSerial.h>

#define trigPin 7
#define echoPin 6
#define ledB 10

SoftwareSerial Bluetooth(12, 9); // Arduino(RX, TX) - HC-05 Bluetooth (TX, RX)

// Create servo object
Servo s24;
Servo s23;
Servo s22;

Servo s21;
Servo s20;
Servo s19;

Servo s18;
Servo s17;
Servo s16;

Servo s7;
Servo s8;
Servo s9;

Servo s10;
Servo s11;
Servo s12;
```

```
Servo s1;
Servo s2;
Servo s3;

Servo s15;
Servo s14;
Servo s13;

Servo s5;

int i0H1 = 0;
int i1H1 = 0;
int i2H1 = 0;
int i3H1 = 0;
int i4H1 = 0;
int i5H1 = 0;
int i6H1 = 0;
int i7H1 = 0;
int i8H1 = 0;

int i0T1 = 0;
int i1T1 = 0;
int i2T1 = 0;
int i3T1 = 0;
int i4T1 = 0;
int i5T1 = 0;
int i6T1 = 0;

int i1L1 = 0;
int i2L1 = 0;
int i3L1 = 0;
int i4L1 = 0;
int i5L1 = 0;
int i6L1 = 0;

int i1L2 = 0;
int i2L2 = 0;
int i3L2 = 0;
int i4L2 = 0;
int i5L2 = 0;
int i6L2 = 0;
boolean l1status = LOW;
boolean l2status = LOW;
boolean aStatus = LOW;
boolean attStatus = LOW;
int k = 0;
int a = 0;
int aa = 0;
int period = 1000;
unsigned long time_now = 0;

float distance;
long duration;
int dataIn;
int m = 0;
int h = 0;
int t = 0;
int att = 0;
int speedV = 30;

void setup() {
  Serial.begin(38400);
  Bluetooth.begin(38400); // Default baud rate of the Bluetooth module
```

```
Bluetooth.setTimeout(1);
delay(20);
pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
pinMode(echoPin, INPUT); // Sets the echoPin as an Input
pinMode(ledB, OUTPUT);
// Head
s15.attach(36, 600, 2400);
s14.attach(35, 600, 2400);
s13.attach(34, 600, 2400); //grip
// Tail
s5.attach(26, 600, 2400); // Tail
// Leg 4
s10.attach(31, 600, 2400);
s11.attach(32, 600, 2400);
s12.attach(33, 600, 2400); //rot
// Leg 5
s7.attach(28, 600, 2400);
s8.attach(29, 600, 2400);
s9.attach(30, 600, 2400); //rot
// Leg 6
s1.attach(22, 600, 2400);
s2.attach(23, 600, 2400);
s3.attach(24, 600, 2400); //rot
// Leg 1
s18.attach(39, 600, 2400);
s17.attach(38, 600, 2400);
s16.attach(37, 600, 2400); //rot
// Leg 2
s21.attach(42, 600, 2400);
s20.attach(41, 600, 2400);
s19.attach(40, 600, 2400); //rot
// Leg 3
s24.attach(45, 600, 2400);
s23.attach(44, 600, 2400);
s22.attach(43, 600, 2400); //rot

// == Move to initial position
// Head
s15.write(72);
s14.write(50);
s13.write(90); // Grip

s5.write(65); // Tail

// Leg 4
s10.write(65);
s11.write(35);
s12.write(40);
// Leg 5
s7.write(80);
s8.write(50);
s9.write(25);
// Leg 6
s1.write(90);
s2.write(45);
s3.write(60);

// Leg 1
s18.write(60);
s17.write(90);
s16.write(100);
// Leg 2
s21.write(50);
s20.write(85);
```

```
  s19.write(75);
  // Leg 3
  s24.write(50);
  s23.write(80);
  s22.write(80);

  delay(3000);
}
void loop() {

  // Check for incoming data
  if (Bluetooth.available() > 0) {
    dataIn = Bluetooth.read();  // Read the data
    if (dataIn == 2) {
      m = 2;
    }
    if (dataIn == 3) {
      m = 3;
    }
    if (dataIn == 4) {
      m = 4;
    }
    if (dataIn == 5) {
      m = 5;
    }
    if (dataIn == 10) {
      t = 10;
    }
    if (dataIn == 11) {
      h = 11;
    }
    if (dataIn >= 15) {
      speedV = dataIn;
    }
  }
  // Move forward
  if (m == 2) {
    moveLeg1();
    moveLeg3();
    moveLeg5();
    if (l1status == HIGH) {
      moveLeg2();
      moveLeg4();
      moveLeg6();
    }
  }
  // Rotate left
  if (m == 3) {
    moveHeadLeft();
    moveLeg1();
    moveLeg3();
    moveLeg5Left();
    if (l1status == HIGH) {
      moveLeg2();
      moveLeg4Left();
      moveLeg6Left();
    }
  }
  // Rotate right
  if (m == 4) {
    moveHeadRight();
    moveLeg1Right();
    moveLeg3Right();
    moveLeg5();
```

```
  if (l1status == HIGH) {
    moveLeg2Right();
    moveLeg4();
    moveLeg6();
  }
}
// Move reverse
if (m == 5) {
  moveLeg1Rev();
  moveLeg3Rev();
  moveLeg5Rev();
  if (l1status == HIGH) {
    moveLeg2Rev();
    moveLeg4Rev();
    moveLeg6Rev();
  }
}
// Bite
if (dataIn == 6) {
  bite();
}
// Attack
if (dataIn == 7) {
  prepareAttack();
  if (aStatus == HIGH) {
    while (a == 0) {
      delay(1000);
      a = 1;
    }
    attack();
    if (attStatus == HIGH) {
      while (aa == 0) {
        delay(2000);
        aa = 1;
      } attStatus = LOW;
    }
  }
}
// Grab
if (dataIn == 8) {
  grab();
}
// Drop
if (dataIn == 9) {
  drop();
}

// Tail
if (t == 10) {
  tail();
}
// Head
if (h == 11) {
  moveHead();
}

if (dataIn == 12) {
  initialPosTail();
}
if (dataIn == 13) {
  initialPosHead();
}
// Initial and resting position
if (dataIn == 0) {
```

```
    initialPosition();

    // Get the distance from the ultrasonic sensor
    if (getDistance() > 40) {
      att = 0;
    }
    if (getDistance() <= 40) {
      att = 1;
      dataIn = 99;
    }
    // Monitor the battery voltage
    int sensorValue = analogRead(A3);
    float voltage = sensorValue * (5.00 / 1023.00) * 2.9; // Convert the reading values
from 5v to suitable 12V i
    Serial.println(voltage);
    // If voltage is below 11V turn on the LED
    if (voltage < 11) {
      digitalWrite(ledB, HIGH);
    }
    else {
      digitalWrite(ledB, LOW);
    }
  }
  // If there is an object in front of the sensor prepare for attack
  if (att == 1) {
    prepareAttack();
    if (aStatus == HIGH) {
      while (a == 0) {
        delay(2000);
        a = 1;
      }
      if (getDistance() > 30) {
        att = 2;
        a = 0;
        aStatus = LOW;
        initialPosHead();
      }
      if (getDistance() < 30) {
        att = 3;
        a = 0;
        aStatus = LOW;
        initialPosHead();
      }
    }
  }
  // If there is no longer object in front, dismiss the attack
  if (att == 2) {
    dismissAttack();
    if (aStatus == HIGH) {
      dataIn = 0;
      att = 0;
    }
  }
  // If there is closer to the sensor attack
  if (att == 3) {
    attack();
    if (attStatus == HIGH) {
      while (aa == 0) {
        delay(2000);
        aa = 1;
      } attStatus = LOW;
    }
    if (aStatus == HIGH) {
      while (a == 0) {
```

```
        delay(2000);
        a = 1;
      }
      dataIn = 0;
      att = 0;
      initialPosHead();
    }
  }
  delay(speedV);
}

// === ATTACK === //

void prepareAttack() {
  // LEG 1
  if (i1H1 <= 15) {
    //Leg 1
    s18.write(60 - i1H1);
    s17.write(90 - i1H1);
    // Leg 3
    s24.write(50 + i1H1 / 2);
    s23.write(80 + i1H1);
    // Leg 4
    s10.write(65 + i1H1);
    s11.write(35 + i1H1);
    // Leg 6
    s1.write(90 - i1H1);
    s2.write(45 - i1H1);
    // Head
    s14.write(50 - i1H1 * 2);
    s13.write(90 - i1H1);

  }
  if (i1H1 <= 30) {
    s16.write(100 - i1H1);
    s19.write(75 - i1H1);
    s22.write(80 - i1H1);
    s12.write(35 + i1H1);
    s9.write(30 + i1H1);
    s3.write(60 + i1H1);
    i1H1++;
  }
  if (i1H1 >= 30) {
    aStatus = HIGH;
  }
}

void dismissAttack() {
  // LEG 1
  if (i2H1 <= 15) {
    //Leg 1
    s18.write(45 + i2H1);
    s17.write(75 + i2H1);
    // Leg 3
    s24.write(57 - i2H1 / 2);
    s23.write(95 - i2H1);
    // Leg 4
    s10.write(80 - i2H1);
    s11.write(50 - i2H1);
    // Leg 6
    s1.write(75 + i2H1);
    s2.write(30 + i2H1);
    // Head
    s14.write(20 + i2H1 * 2);
```

```
    s13.write(75 + i2H1);
  }
  if (i2H1 <= 30) {
    s16.write(70 + i2H1);
    s19.write(45 + i2H1);
    s22.write(50 + i2H1);
    s12.write(65 - i2H1);
    s9.write(60 - i2H1);
    s3.write(90 - i2H1);
    i2H1++;
  }
  if (i2H1 >= 30) {
    aStatus = HIGH;
  }
}

void attack() {
  // LEG 1
  if (i3H1 <= 10) {
    //Leg 1
    s18.write(45 + i3H1 * 2);
    s17.write(75 + i3H1 * 3);
    // Leg 3
    s24.write(57 - i3H1 / 2);
    s23.write(95 - i3H1 * 3);
    // Leg 4
    s10.write(80 - i3H1 * 2);
    s11.write(50 - i3H1 * 3);
    // Leg 6
    s1.write(75 + i3H1 * 2);
    s2.write(30 + i3H1 * 3);
    // Head
    s14.write(20 + i3H1 * 2);
    s13.write(75 + i3H1 * 3);
  }
  if (i3H1 <= 16) {
    s16.write(70 + i3H1 * 3);
    s19.write(45 + i3H1 * 3);
    s22.write(50 + i3H1 * 3);
    s12.write(65 - i3H1 * 3);
    s9.write(60 - i3H1 * 3);
    s3.write(90 - i3H1 * 3);
    i3H1++;
  }
  if (i3H1 >= 16) {
    attStatus = HIGH;
  }
  if (i3H1 >= 16 & i4H1 < 15) {
    //Leg 1
    s18.write(65 - i4H1 / 3);
    s17.write(105 - i4H1);
    // Leg 3
    //s24.write(50 + i4H1 / 2);
    s23.write(65 + i4H1);
    // Leg 4
    s10.write(60 + i4H1 / 3);
    s11.write(20 + i4H1);
    // Leg 6
    s1.write(95 - i4H1 / 5);
    s2.write(60 - i4H1);
    // Head
    s14.write(40 + i4H1 / 2);
    s13.write(105 - i4H1);
  }
```

```
  if (i3H1 >= 16 & i4H1 <= 18) {
    s16.write(118 - i4H1);
    s19.write(93 - i4H1);
    s22.write(98 - i4H1);
    s12.write(17 + i4H1);
    s9.write(12 + i4H1);
    s3.write(42 + i4H1);
    i4H1++;
  }
  if (i4H1 >= 18) {
    aStatus = HIGH;
  }
}

void moveHead() {
  if (i0H1 <= 40) {
    s15.write(72 + i0H1);
    i0H1++;
  }
  if (i0H1 >= 40 & i1H1 <= 40) {
    s14.write(50 - i1H1 / 2);
    s15.write(112 - i1H1);
    i1H1++;
  }
  if (i1H1 >= 40 & i2H1 <= 20) {
    s13.write(90 - i2H1);
    i2H1++;
  }
  if (i2H1 >= 20 & i3H1 <= 10) {
    s13.write(70 + i3H1 * 4);
    i3H1++;
  }
  if (i3H1 >= 10 & i4H1 <= 40) {
    s13.write(110 - i4H1);
    i4H1++;
  }
  if (i4H1 >= 40 & i5H1 <= 10) {
    s13.write(70 + i5H1 * 4);
    i5H1++;
  }
  if (i5H1 >= 10 & i6H1 <= 20) {
    s13.write(110 - i6H1);
    i6H1++;
  }
  if (i6H1 >= 20 & i7H1 <= 40) {
    s14.write(30 + i7H1 / 2);
    s15.write(72 - i7H1);
    i7H1++;
  }
  if (i7H1 >= 40 & i8H1 <= 40) {
    s15.write(32 + i8H1);
    i8H1++;
  }
}

void bite() {
  if (i1H1 <= 20) {
    s13.write(90 - i1H1);
    i1H1++;
  }
  if (i1H1 >= 20 & i2H1 <= 10) {
    s13.write(70 + i2H1 * 4);
    i2H1++;
  }
```

```
  if (i2H1 >= 10 & i3H1 <= 40) {
    s13.write(110 - i3H1);
    i3H1++;
  }
  if (i3H1 >= 40 & i4H1 <= 10) {
    s13.write(70 + i4H1 * 4);
    i4H1++;
  }
  if (i4H1 >= 10 & i5H1 <= 40) {
    s13.write(110 - i5H1);
    i5H1++;
  }
  if (i5H1 >= 40 & i6H1 <= 20) {
    s13.write(70 + i6H1);
    i6H1++;
  }
}

void moveHeadLeft() {
  if (i0H1 <= 25) {
    s14.write(50 - i0H1);
    s15.write(72 - i0H1);
    s5.write(65 + i0H1); // Tail
    i0H1++;
  }
}
void moveHeadRight() {
  if (i2H1 <= 25) {
    s14.write(50 - i2H1);
    s15.write(72 + i2H1);
    s5.write(65 - i2H1); // Tail
    i2H1++;
  }
}

void grab() {
  if (i1H1 <= 20) {
    s13.write(90 + i1H1);
    i1H1++;
  }
}
void drop() {
  if (i2H1 <= 20) {
    s13.write(110 - i2H1);
    i2H1++;
  }
}

void tail() {
  if (i0T1 <= 25) {
    s5.write(65 - i0T1);
    i0T1++;
  }
  if ( i0T1 >= 25 & i1T1 <= 50) {
    s5.write(40 + i1T1);
    i1T1++;
  }
  if ( i1T1 >= 50 & i2T1 <= 50) {
    s5.write(90 - i2T1);
    i2T1++;
  }
  if ( i2T1 >= 50 & i3T1 <= 50) {
    s5.write(40 + i3T1);
    i3T1++;
```

```
  }
  if ( i3T1 >= 50 & i4T1 <= 50) {
    s5.write(90 - i4T1);
    i4T1++;
  }
  if ( i4T1 >= 50 & i5T1 <= 50) {
    s5.write(40 + i5T1);
    i5T1++;
  }
  if ( i5T1 >= 50 & i6T1 <= 25) {
    s5.write(90 - i6T1);
    i6T1++;
  }
}

void moveLeg1() {
  // Swign phase - move leg though air - from initial to final position
  // Rise the leg
  if (i1L1 <= 10) {
    s18.write(60 - i1L1 * 2);
    s17.write(90 - i1L1 * 3);
    i1L1++;
  }
  // Rotate the leg
  if (i2L1 <= 30) {
    s16.write(100 - i2L1);
    i2L1++;

  }
  // Move back to touch the ground
  if (i2L1 > 20 & i3L1 <= 10) {
    s18.write(40 + i3L1 * 2);
    s17.write(60 + i3L1 * 3);
    i3L1++;
  }
  // Stance phase - move leg while touching the ground
  // Rotate back to initial position
  if (i2L1 >= 30) {
    s16.write(70 + i4L1);
    i4L1++;
    l1status = HIGH;
  }
  // Reset the counters for repeating the process
  if (i4L1 >= 30) {
    i1L1 = 0;
    i2L1 = 0;
    i3L1 = 0;
    i4L1 = 0;
    i5L1 = 0;
  }
  // Each iteration or step is executed in the main loop section where there is also a
delay time for controlling the speed of movement
}
void moveLeg2() {
  if (i1L2 <= 10) {
    s21.write(50 - i1L2 * 2);
    s20.write(80 - i1L2 * 3);
    i1L2++;
  }
  if (i2L2 <= 30) {
    s19.write(75 - i2L2);
    i2L2++;

  }
```

```
  if (i2L2 > 20 & i3L2 <= 10) {
    s21.write(30 + i3L2 * 2);
    s20.write(50 + i3L2 * 3);
    i3L2++;
  }
  if (i2L2 >= 30) {
    s19.write(45 + i4L2);
    i4L2++;
  }
  if (i4L2 >= 30) {
    i1L2 = 0;
    i2L2 = 0;
    i3L2 = 0;
    i4L2 = 0;
    i5L2 = 0;
  }
}

void moveLeg3() {
  if (i1L1 <= 10) {
    s24.write(50 - i1L1 * 2);
    s23.write(80 - i1L1 * 3);
  }
  if (i2L1 <= 30) {
    s22.write(80 - i2L1);

  }
  if (i2L1 > 20 & i3L1 <= 10) {
    s24.write(30 + i3L1 * 2);
    s23.write(50 + i3L1 * 3);
  }
  if (i2L1 >= 30) {
    s22.write(50 + i4L1);
  }
}

void moveLeg4() {
  if (i1L2 <= 10) {
    s10.write(65 + i1L2 * 2);
    s11.write(35 + i1L2 * 3);
  }
  if (i2L2 <= 30) {
    s12.write(35 + i2L2);

  }
  if (i2L2 > 20 & i3L2 <= 10) {
    s10.write(85 - i3L2 * 2);
    s11.write(65 - i3L2 * 3);
  }
  if (i2L2 >= 30) {
    s12.write(65 - i4L2);
  }
}

void moveLeg5() {
  if (i1L1 <= 10) {
    s7.write(80 + i1L1 * 2);
    s8.write(50 + i1L1 * 3);
  }
  if (i2L1 <= 30) {
    s9.write(30 + i2L1);

  }
  if (i2L1 > 20 & i3L1 <= 10) {
```

```
    s7.write(100 - i3L1 * 2);
    s8.write(80 - i3L1 * 3);
  }
  if (i2L1 >= 30) {
    s9.write(60 - i4L1);
  }
}

void moveLeg6() {
  if (i1L2 <= 10) {
    s1.write(90 + i1L2 * 2);
    s2.write(45 + i1L2 * 3);
  }
  if (i2L2 <= 30) {
    s3.write(60 + i2L2);
  }
  if (i2L2 > 20 & i3L2 <= 10) {
    s1.write(110 - i3L2 * 2);
    s2.write(75 - i3L2 * 3);
  }
  if (i2L2 >= 30) {
    s3.write(90 - i4L2);
  }
}

void moveLeg1Rev() {
  if (i1L1 <= 10) {
    s18.write(60 - i1L1 * 2);
    s17.write(90 - i1L1 * 3);
    Serial.println(s17.read());
    i1L1++;
  }
  if (i2L1 <= 30) {
    s16.write(70 + i2L1);
    i2L1++;

  }
  if (i2L1 > 20 & i3L1 <= 10) {
    s18.write(40 + i3L1 * 2);
    s17.write(60 + i3L1 * 3);
    Serial.println(s17.read());
    i3L1++;
  }
  if (i2L1 >= 30) {
    s16.write(100 - i4L1);
    i4L1++;
    l1status = HIGH;
  }
  if (i4L1 >= 30) {
    i1L1 = 0;
    i2L1 = 0;
    i3L1 = 0;
    i4L1 = 0;
    i5L1 = 0;
  }
}
void moveLeg2Rev() {
  if (i1L2 <= 10) {
    s21.write(50 - i1L2 * 2);
    s20.write(80 - i1L2 * 3);
    i1L2++;
  }
  if (i2L2 <= 30) {
    s19.write(45 + i2L2);
```

```
    i2L2++;

  }
  if (i2L2 > 20 & i3L2 <= 10) {
    s21.write(30 + i3L2 * 2);
    s20.write(50 + i3L2 * 3);
    i3L2++;
  }
  if (i2L2 >= 30) {
    s19.write(75 - i4L2);
    i4L2++;
  }
  if (i4L2 >= 30) {
    i1L2 = 0;
    i2L2 = 0;
    i3L2 = 0;
    i4L2 = 0;
    i5L2 = 0;
  }
}

void moveLeg3Rev() {
  if (i1L1 <= 10) {
    s24.write(50 - i1L1 * 2);
    s23.write(80 - i1L1 * 3);
  }
  if (i2L1 <= 30) {
    s22.write(50 + i2L1);

  }
  if (i2L1 > 20 & i3L1 <= 10) {
    s24.write(30 + i3L1 * 2);
    s23.write(50 + i3L1 * 3);
  }
  if (i2L1 >= 30) {
    s22.write(80 - i4L1);
  }
}

void moveLeg4Rev() {
  if (i1L2 <= 10) {
    s10.write(65 + i1L2 * 2);
    s11.write(35 + i1L2 * 3);
  }
  if (i2L2 <= 30) {
    s12.write(65 - i2L2);

  }
  if (i2L2 > 20 & i3L2 <= 10) {
    s10.write(85 - i3L2 * 2);
    s11.write(65 - i3L2 * 3);
  }
  if (i2L2 >= 30) {
    s12.write(35 + i4L2);
  }
}

void moveLeg5Rev() {
  if (i1L1 <= 10) {
    s7.write(80 + i1L1 * 2);
    s8.write(50 + i1L1 * 3);
  }
  if (i2L1 <= 30) {
    s9.write(60 - i2L1);
```

```
  }
  if (i2L1 > 20 & i3L1 <= 10) {
    s7.write(100 - i3L1 * 2);
    s8.write(80 - i3L1 * 3);
  }
  if (i2L1 >= 30) {
    s9.write(30 + i4L1);
  }
}

void moveLeg6Rev() {
  if (i1L2 <= 10) {
    s1.write(90 + i1L2 * 2);
    s2.write(45 + i1L2 * 3);
  }
  if (i2L2 <= 30) {
    s3.write(90 - i2L2);
  }
  if (i2L2 > 20 & i3L2 <= 10) {
    s1.write(110 - i3L2 * 2);
    s2.write(75 - i3L2 * 3);
  }
  if (i2L2 >= 30) {
    s3.write(60 + i4L2);
  }
}

void moveLeg1Right() {
  if (i1L1 <= 10) {
    s18.write(60 - i1L1 * 2);
    s17.write(90 - i1L1 * 3);
    i1L1++;
  }
  if (i2L1 <= 30) {
    s16.write(70 + i2L1);
    i2L1++;

  }
  if (i2L1 > 20 & i3L1 <= 10) {
    s18.write(40 + i3L1 * 2);
    s17.write(60 + i3L1 * 3);
    i3L1++;
  }
  if (i2L1 >= 30) {
    s16.write(100 - i4L1);
    i4L1++;
    l1status = HIGH;
  }
  if (i4L1 >= 30) {
    i1L1 = 0;
    i2L1 = 0;
    i3L1 = 0;
    i4L1 = 0;
    i5L1 = 0;
  }
}
void moveLeg2Right() {
  if (i1L2 <= 10) {
    s21.write(50 - i1L2 * 2);
    s20.write(80 - i1L2 * 3);
    i1L2++;
  }
  if (i2L2 <= 30) {
```

```
    s19.write(45 + i2L2);
    i2L2++;

  }
  if (i2L2 > 20 & i3L2 <= 10) {
    s21.write(30 + i3L2 * 2);
    s20.write(50 + i3L2 * 3);
    i3L2++;
  }
  if (i2L2 >= 30) {
    s19.write(75 - i4L2);
    i4L2++;
  }
  if (i4L2 >= 30) {
    i1L2 = 0;
    i2L2 = 0;
    i3L2 = 0;
    i4L2 = 0;
    i5L2 = 0;
  }
}

void moveLeg3Right() {
  if (i1L1 <= 10) {
    s24.write(50 - i1L1 * 2);
    s23.write(80 - i1L1 * 3);
  }
  if (i2L1 <= 30) {
    s22.write(50 + i2L1);

  }
  if (i2L1 > 20 & i3L1 <= 10) {
    s24.write(30 + i3L1 * 2);
    s23.write(50 + i3L1 * 3);
  }
  if (i2L1 >= 30) {
    s22.write(80 - i4L1);
  }
}

void moveLeg4Left() {
  if (i1L2 <= 10) {
    s10.write(65 + i1L2 * 2);
    s11.write(35 + i1L2 * 3);
  }
  if (i2L2 <= 30) {
    s12.write(60 - i2L2);

  }
  if (i2L2 > 20 & i3L2 <= 10) {
    s10.write(85 - i3L2 * 2);
    s11.write(65 - i3L2 * 3);
  }
  if (i2L2 >= 30) {
    s12.write(30 + i4L2);
  }
}

void moveLeg5Left() {
  if (i1L1 <= 10) {
    s7.write(80 + i1L1 * 2);
    s8.write(50 + i1L1 * 3);
  }
  if (i2L1 <= 30) {
```

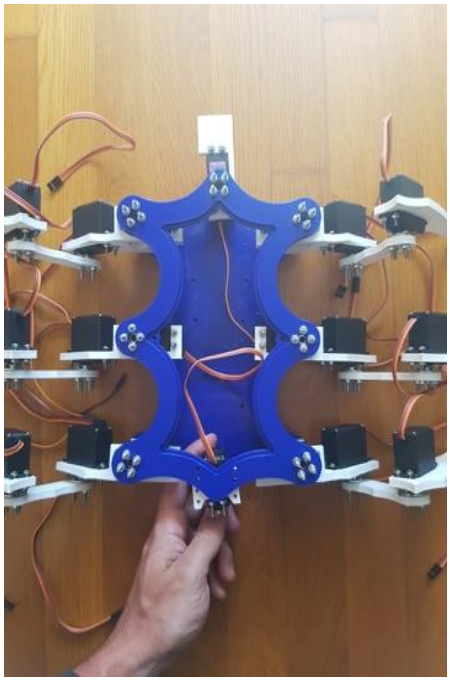
```
    s9.write(60 - i2L1);

  }
  if (i2L1 > 20 & i3L1 <= 10) {
    s7.write(100 - i3L1 * 2);
    s8.write(80 - i3L1 * 3);
  }
  if (i2L1 >= 30) {
    s9.write(30 + i4L1);
  }
}

void moveLeg6Left() {
  if (i1L2 <= 10) {
    s1.write(90 + i1L2 * 2);
    s2.write(45 + i1L2 * 3);
  }
  if (i2L2 <= 30) {
    s3.write(90 - i2L2);
  }
  if (i2L2 > 20 & i3L2 <= 10) {
    s1.write(110 - i3L2 * 2);
    s2.write(75 - i3L2 * 3);
  }
  if (i2L2 >= 30) {
    s3.write(60 + i4L2);
  }
}

void initialPosTail() {
  i0T1 = 0;
  i1T1 = 0;
  i2T1 = 0;
  i3T1 = 0;
  i4T1 = 0;
  i5T1 = 0;
  i6T1 = 0;
  t = 0;
}

void initialPosHead() {
  attStatus = LOW;
  aStatus = LOW;
  i0H1 = 0;
  i1H1 = 0;
  i2H1 = 0;
  i3H1 = 0;
  i4H1 = 0;
  i5H1 = 0;
  i6H1 = 0;
  i7H1 = 0;
  i8H1 = 0;
  h = 0;
  aa = 0;
}

void initialPosition() {
  a = 0;
  aa = 0;
  m = 0;
  l1status = LOW;
  l2status = LOW;
  aStatus = LOW;
  attStatus = LOW;
```

```
  // Head
  s15.write(72);
  s14.write(55);
  s13.write(90); // Grip

  s5.write(65); // Tail

  // Leg 4
  s10.write(65);
  s11.write(35);
  s12.write(40);
  // Leg 5
  s7.write(80);
  s8.write(50);
  s9.write(25);
  // Leg 6
  s1.write(90);
  s2.write(45);
  s3.write(60);

  // Leg 1
  s18.write(60);
  s17.write(90);
  s16.write(100);
  // Leg 2
  s21.write(50);
  s20.write(80);
  s19.write(75);
  // Leg 3
  s24.write(50);
  s23.write(80);
  s22.write(80);
  i0H1 = 0;
  i1H1 = 0;
  i2H1 = 0;
  i3H1 = 0;
  i4H1 = 0;
  i5H1 = 0;
  i6H1 = 0;
  i7H1 = 0;
  i8H1 = 0;

  i0T1 = 0;
  i1T1 = 0;
  i2T1 = 0;
  i3T1 = 0;
  i4T1 = 0;
  i5T1 = 0;
  i6T1 = 0;

  i1L1 = 0;
  i2L1 = 0;
  i3L1 = 0;
  i4L1 = 0;
  i5L1 = 0;
  i6L1 = 0;

  i1L2 = 0;
  i2L2 = 0;
  i3L2 = 0;
  i4L2 = 0;
  i5L2 = 0;
  i6L2 = 0;
}
```

```
//===== getDistance - Custom Function
int getDistance() {
  // Clears the trigPin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  // Reads the echoPin, returns the sound wave travel time in microseconds
  duration = pulseIn(echoPin, HIGH);
  // Calculating the distance
  distance = duration * 0.034 / 2; // distance in cm
  return distance;
}
```

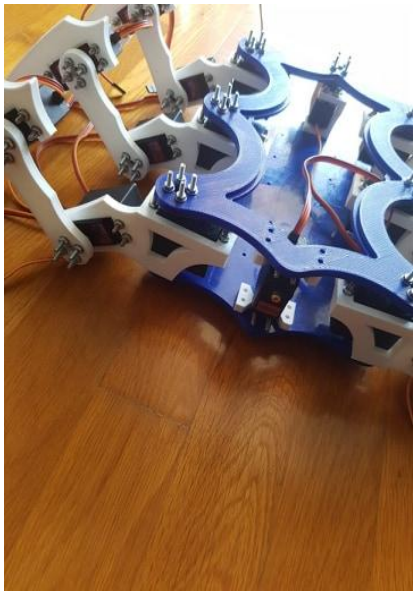
Κεφάλαιο 6: Αποτελέσματα

Οι ανωτέρω διαδικασίες οδήγησαν στην κατασκευή ενός εξάποδου ρομπότ-μυρμηγκιού το οποίο ελέγχεται μέσω εφαρμογής Android. Ο μηχανισμός μπορεί αφού του δοθούν οι κατάλληλες εντολές να προχωρήσει μπροστά, πίσω και πλάγια, να επιτεθεί και να αρπάξει αντικείμενα με τις εμπρόσθιες δαγκάνες του. Επιπροσθέτως, μπορεί να κινήσει την ουρά του. Σημαντική είναι επίσης η δυνατότητα του μυρμηγκιού, μέσω των αισθητήρων που έχει στο κεφάλι του, να αντιλαμβάνεται την απειλητικές κινήσεις που κατευθύνονται στο κεφάλι του και να προετοιμάζεται για επίθεση, αλλά και να εγκαταλείπει την προετοιμασία για επίθεση όταν το χέρι απομακρύνεται. Στις παρακάτω εικόνες παρουσιάζεται το ρομπότ κατά τα τελευταία στάδια της κατασκευής όπου αρχίζει να παίρνει την τελική του μορφή.

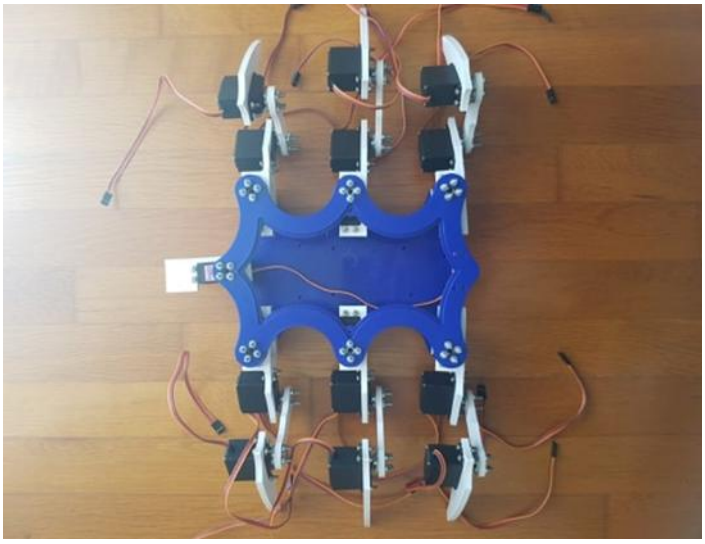


Εικόνα 22: Τελικά στάδια κατασκευής (1)

Εικόνα 23: Η «στερέωση» της κάτω βάσης του ρομπότ



Εικόνα 24: Τελικά στάδια της κατασκευής (2)



Εικόνα 25: Πανοραμική λήψη του ρομπότ χωρίς το σκέπαστρο

Κεφάλαιο 7: Συμπεράσματα

Με την παρούσα εργασία επιχειρήθηκε να παρουσιαστεί το θεωρητικό πλαίσιο της κατασκευής ενός εξάποδου Arduino-ρομπότ για την προσομοίωση του τρόπου περπατήματος και δράσης του μυρμηγκιού. Το περιεχόμενο της εργασίας, η λογική διάρθρωση και η διατύπωση της ήταν σκοίμως περιεκτική και όσο το δυνατόν απλουστευμένη, ούτως ώστε η εργασία να είναι εύληπτη ακόμη και σε αναγνωστικό κοινό που δεν έχει ασχοληθεί με τον προγραμματισμό.

Είναι βέβαια προφανές ότι κάθε περίπτωση αυτή η κατασκευή γίνεται στα πλαίσια μίας πτυχιακής εργασίας και ως εκ τούτου δεν μπορεί να συγκριθεί με μια επαγγελματική κατασκευή εμπορίου. Αναντίλλεκτα, η συγκεκριμένη κατασκευή χρήζει περαιτέρω διερεύνησης και βελτίωσης, ωστόσο θεωρείται ιδανικό «ερέθισμα» για οποιονδήποτε επιθυμεί να κάνει τα πρώτα βήματα ενασχόλησης με τον μικροελεγκτή Arduino και γενικότερα με το περιβάλλον της μηχατρονικής.

Η κατανόηση των ενσωματωμένων κυκλωμάτων, δηλαδή των μικρών αυτών ηλεκτρονικών συστημάτων που επεξεργάζονται δεδομένα παρουσιάζει αυξημένο επιστημονικό ενδιαφέρον και είναι μία θεματική που διαρκώς μελετάται και εξελίσσεται από τις βιομηχανίες. Σημαντικό πλεονέκτημα των ενσωματωμένων κυκλωμάτων σε συνδυασμό και με τους μικροελεκτές είναι ότι μπορούν να πραγματοποιηθούν κατασκευές με χαμηλό κόστος οι οποίες μπορούν να καλύψουν πληθώρα αναγκών.

Η συγκεκριμένη εργασία επιδιώκει να αποτελέσει το «έναυσμα» για τη δημιουργία παρόμοιων κυκλωμάτων και για περαιτέρω έρευνα που θα κατευθύνεται στη βελτίωση παρόμοιων μηχανισμών. Η χρησιμότητα αυτών των μηχανισμών είναι αδιαμφισβήτητη. Επί παραδείγματι, το ρομπότ-μυρμήγκι που κατασκευάστηκε με αυτή την εργασία θα μπορούσε να χρησιμεύσει από την παραγωγή (εκτελώντας κινήσεις, μετακινήσεις και μεταφορές σε δύσβατες επιφάνειες) μέχρι και να διευκολύνει την καθημερινή ζωή των αμεα, μεταφέροντας υπό τις κατάλληλες οδηγίες με τις δαγκάνες τους απομακρυσμένα αντικείμενα κτλ.

Η σημαντική συνεισφορά της εργασίας είναι ότι απέδειξε πως η κατασκευή και ρύθμιση των ποδιών του ρομπότ, πέραν του ότι είναι ασύγκριτα πιο ευχερής από ό, τι η επιλογή τροχών, είναι αρκετά πιο εύκολη, αν γίνει ο κατάλληλος σχεδιασμός, ρύθμιση και σωστή τοποθέτηση τους στο σώμα του μηχανισμού. Δηλαδή, νευραλγικής σημασίας είναι εν πρώτοις το μήκος των ποδιών, αλλά και η ακριβής τοποθέτηση τους στο σώμα του ρομπότ, ώστε να μη μειώνεται το εύρος της κίνησης του.

Σημαντικές προτάσεις προς περαιτέρω διερεύνηση και βελτίωση της συγκεκριμένης κατασκευής περιλαμβάνουν τον εμπλουτισμό των εντολών που θα μπορεί να δεχθεί ένας τέτοιος μηχανισμός, δηλαδή να μπορεί να εκτελέσει πιο σύνθετες κινήσεις από αυτές τις εμπρόσθιας και οπίσθιας μετακίνησης, της επίθεσης και του δαγκώματος. Απαραίτητος κρίνεται επίσης ο εκτενέστερος περιαματισμός με την προσαρμογή αισθητήρα στο σώμα του ρομπότ, ούτως ώστε να αντιλαμβάνεται επικίνδυνες επιθέσεις-κινήσεις όχι μόνο από το κεφάλι και τα μάτια του, αλλά και πίσω από την ουρά του ή και πάνω από τη βάση του. Βέβαια είναι προφανές ότι το συγκεκριμένο εγχείρημα παρουσιάζει πρόσθετες δυσκολίες διότι είναι πολύ δυσχερές να «χωρέσει» ένας παρόμοιος αισθητήρας τόσο στην ουρά, όσο και στο επάνω μέρος της βάσης του ρομπότ.

Κεφάλαιο 8: Βιβλιογραφία

[1] M. Erden, «Optimal Protraction of a Biologically Inspired Robot Leg», Journal of Intelligent and Robotic Systems, v. 64, 2011, σ. 301-322

[2] M. Silva & J. Machado, «A Historical Perspective of Legged Robots», Journal of Vibration and Control, v. 13, 2007, σ. 1447 καθώς και J. Estremera & P. de Santos, Generating Continuous Free Crab Gaits for Quadruped Robots on Irregular Terrain», Transactions on Robotics, v. 21, No 6, 2005

[3] Zu Guang Zhang & Hiroshi Kimura, «Adaptive Running of a Quadruped Robot Using Forced Vibration and Synchronization», Proceeding of the Journal of Vibration and Control, 4/5/2006, σ. 1361–1383

[4] U. Saranli & M. Buehler, «A Simple and Highly Mobile Hexapod Robot», The International Journal of Robotics Research, Ιούλιος 2001, v. 20, 2001, σ. 616

[5] H. Kimura, «Adaptive Dynamic Walking of a Quadruped Robot on Natural Ground Based on Biological Concepts», The International Journal of Robotics Research, v. 26, 2007, σ. 475

[6] S. Chu & G. Kwok, «Comparison Between Different Model of Hexapod Robot in Fault-Tolerant Gait», Transactions On Systems, Man, And Cybernetics—Part A: Systems And Humans, v. 32, N. 6, Νοέμβριος 2002

[7] J. Liu & X. Zhao, «Legged robots – an overview», Transactions of the Institute of Measurement and Control, v. 29, 2007, σ. 185

[8] A. Mojdehi, M. Alitavoli, A. Darvizeh, H. Rajabi & H. Larijani, «Modeling and Simulation of Spider's Walking», International Journal of Design & Nature and Ecodynamics, v. 6(2), Ιούνιος 2011, σ. 83-96

[9] F. Delcomyn & M. Nelson, «Architectures for a biomimetic hexapod robot», Robotics and Autonomous Systems, v. 30, 2000, Διαθέσιμο σε: https://experts.illinois.edu/en/publications/architectures-for-a-biomimetic-hexapod-robot

[10] M. Agheli, L. Qu & S. Nestinger, «SHeRo: Scalable hexapod robot for maintenance, repair, and operations», Robotics and Computer-Integrated Manufacturing v. 30(5), Οκτώβριος 2014, σ. 478–488, Διαθέσιμο:https://www.google.com/amp/s/www.researchgate.net/publication/261714501_SHeRo_Scalable_hexapod_robot_for_maintenance_repair_and_operations/amp

[11] https://howtomechatronics.com/projects/arduino-ant-hexapod-robot/