ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΕΡΓΑΣΤΗΡΙΟ ΑΚΟΥΣΤΙΚΗΣ, ΕΠΙΚΟΙΝΩΝΙΑΣ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΜΜΜ

Κατασκευή Delta Robot

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ονοματεπώνυμο: Βαγενάς Γεώργιος
Α.Μ.: 03108018

Επιβλέπων: Γεώργιος Καμπουράκης
Επ. καθηγητής Ε.Μ.Π.

Αθήνα, Μάρτιος 2014

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΕΡΓΑΣΤΗΡΙΟ ΑΚΟΥΣΤΙΚΗΣ, ΕΠΙΚΟΙΝΩΝΙΑΣ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΜΜΜ

Κατασκευή Delta Robot

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ονοματεπώνυμο: Βαγενάς Γεώργιος
Α.Μ.: 03108018

Επιβλέπων: Γεώργιος Καμπουράκης
Επ. καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την Τρίτη 18.03.2014.

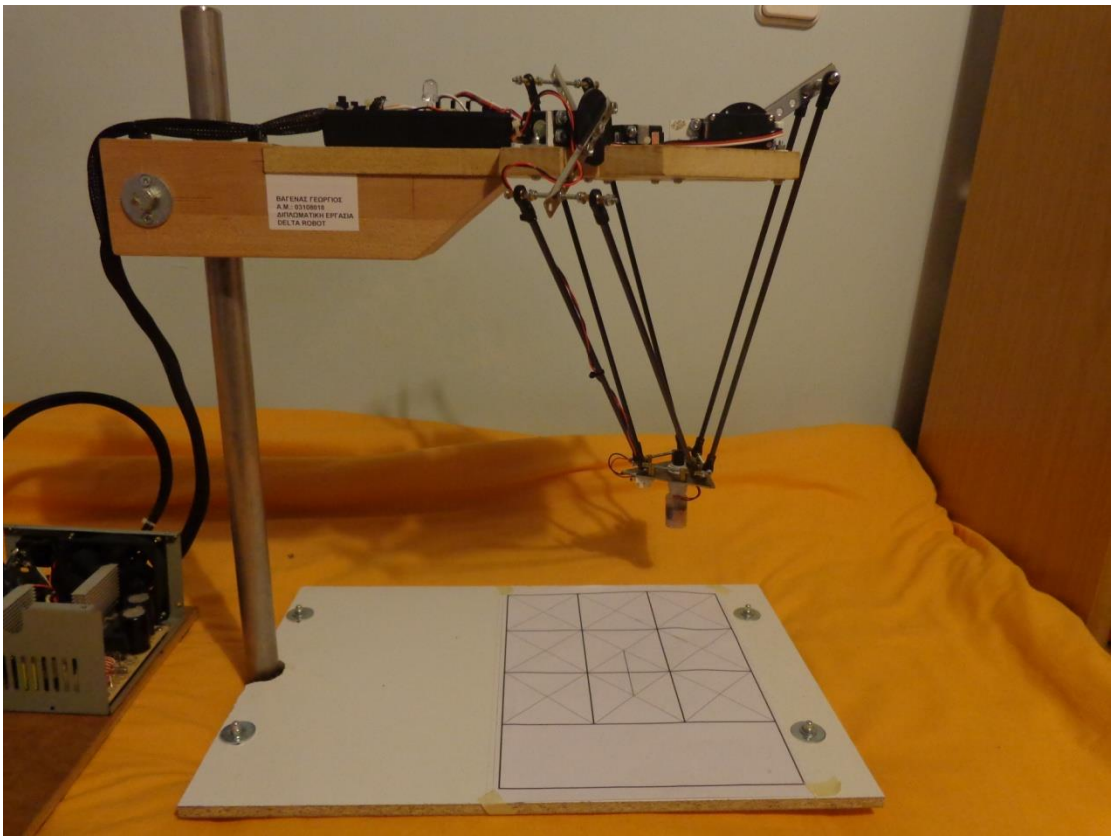
.....	.....	.....
Γ. Καμπουράκης	Β. Λούμος	Ε. Καγιάφας
Επ. Καθηγητής ΕΜΠ	Καθηγητής ΕΜΠ	Καθηγητής ΕΜΠ

Αθήνα, Μάρτιος 2014

.....

Βαγενάς Γεώργιος

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.


ΒΑΓΕΝΑΣ ΓΕΩΡΓΙΟΣ
Α.Μ.: 03108018
ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ
DELTA ROBOT

Περίληψη

Σκοπός της παρούσας διπλωματικής εργασίας ήταν η κατασκευή ενός ρομπότ delta και ο προγραμματισμός του, ώστε να εκτελεί κάποια απλή σχετικά εργασία, που να αποδεικνύει την ορθή λειτουργία του.

Το workspace που ζητήθηκε να καλύπτει το ρομπότ είναι ένα τετράγωνο διαστάσεων 20cm x 20cm. Η λειτουργία που επιλέχθηκε να εκτελεί είναι η συλλογή μεταλλικών κεφαλών πινεζών μέσω ενός ηλεκτρομαγνήτη και η συγκέντρωσή τους σε ένα κουτί.

Στην εργασία αρχικά αναφέρονται κάποια εισαγωγικά στοιχεία σχετικά με τη ρομποτική και πιο συγκεκριμένα σχετικά με το ρομπότ delta, στοιχεία απαραίτητα για τη μελέτη του ρομπότ και τον προγραμματισμό του.

Στη συνέχεια, γίνεται εκτενής προσομοίωση του ρομπότ στο Matlab, ώστε να βρούμε τι διαστάσεις πρέπει να έχει η κατασκευή για να καλύπτει το επιθυμητό workspace.

Το σημαντικότερο μέρος της εργασίας είναι το κατασκευαστικό, όπου επεξηγείται βήμα-βήμα και μέσω της χρήσης πολλών φωτογραφιών ο τρόπος με τον οποίο έγινε η σύνθεση του ρομπότ, τόσο στο μηχανολογικό όσο και στο ηλεκτρονικό τμήμα.

Τέλος, εξηγείται ο τρόπος με τον οποίο προγραμματίστηκε το ρομπότ μέσω του μικροεπεξεργαστή Arduino και προτείνονται κάποια θέματα για περεταίρω μελέτη.

Λέξεις - φράσεις κλειδιά

ρομπότ δέλτα, παράλληλο ρομπότ, e, f, r_e, r_f, σερβοκινητήρας, matlab, arduino, ηλέκτρομαγνήτης

Abstract

The aim of this thesis was to construct a delta robot and program it in order to perform some relatively simple task, demonstrating its proper operation.

The requested workspace to be covered by the robot is a square of dimensions 20cm x 20cm. The function chosen by us to be performed is collecting metal pinheads by an electromagnet and gathering them in a box .

The thesis initially refers to some introductory information about robotics and more specifically about delta robot, which are essential for the study of the robot and its programming.

Then, an extensive simulation of the robot in Matlab was performed, to find out what dimensions our robot should have to cover the desired workspace.

The most important part of the work is the construction, explaining step-by-step, through the use of several photos, the way in which the robot was composed, both in mechanical and electronic section.

Finally, it is explained how the robot was programmed through the Arduino microprocessor and some topics for further study are suggested.

Keywords

delta robot, parallel robot, e, f, r_e, r_f, servomotor (servo), matlab, arduino, electromagnet

Ευχαριστίες…

Αρχικά, θα ήθελα να ευχαριστήσω την οικογένειά μου για τη συμπαράστασή της καθ’ όλη τη διάρκεια των σπουδών μου, αλλά και κατά την εκπόνηση της παρούσας διπλωματικής εργασίας.

Στη συνέχεια θα ήθελα να ευχαριστήσω τον κ. Καμπουράκη που μου έδωσε την ευκαιρία να ασχοληθώ με ένα τόσο ενδιαφέρον θέμα, καθώς και τον κ. Πιπερίδη για τις πολύτιμες συμβουλές και υποδείξεις του.

Τέλος, θα ήθελα να ευχαριστήσω τους φίλους μου, που χωρίς τη βοήθεια και τη στήριξή τους δε θα είχα φτάσει εδώ.

Περιεχόμενα

Α. Θεωρητική ανάλυση

Β. Σχεδίαση - Κατασκευή

Γ. Ανάπτυξη και εφαρμογή software

Α. Θεωρητική ανάλυση

1. Εισαγωγή

1.1. Ρομποτική

Εικόνα 1.1. Ρομποτικός βραχίονας

Ένα ρομπότ είναι μια μηχανική συσκευή η οποία μπορεί να υποκαθιστά τον άνθρωπο σε διάφορες εργασίες. Ένα ρομπότ μπορεί να δράσει κάτω από τον απευθείας έλεγχο ενός ανθρώπου ή αυτόνομα κάτω από τον έλεγχο ενός προγραμματισμένου υπολογιστή.

Τα ρομπότ μπορούν να χρησιμοποιηθούν ώστε να κάνουν εργασίες οι οποίες είτε είναι δύσκολες ή επικίνδυνες για να γίνουν απευθείας από έναν άνθρωπο. Σε άλλες περιπτώσεις, χρησιμοποιούνται για να εκτελέσουν εργασίες ταχύτερα ή φθηνότερα απ' ότι ο άνθρωπος. Έτσι, μπορούν να χρησιμοποιηθούν στην αυτόματη παραγωγή μεγάλων ποσοτήτων κάποιου προϊόντος και με χαμηλότερο κόστος (για παράδειγμα, στις αλυσίδες παραγωγής).

Η λέξη ρομπότ προέρχεται από το σλαβικό ‘robota’ που σημαίνει εργασία. Καθιερώθηκε ως όρος με την σημερινή του έννοια το 1920 από τον Τσέχο θεατρικό συγγραφέα Karel Čapek στο έργο του "R.U.R." (Rossum's Universal Robots), όπου σατιρίζει την εξάρτηση της κοινωνίας από τους μηχανικούς εργάτες (ρομπότ) της τεχνολογικής εξέλιξης και που τελικά εξοντώνουν τους δημιουργούς τους. Σε πολλές σύγχρονες σλαβικές γλώσσες (πχ την πολωνική) χρησιμοποιείται σαν έκφραση της καθημερινότητας με την έννοια της σκληρής δουλειάς.

1.2. Delta Robot

Εικόνα 1.2. Delta Robot

Ένα ρομπότ Delta είναι ένα είδος παράλληλου ρομπότ. Αποτελείται από τρεις βραχίονες που συνδέονται με αρθρώσεις στη βάση. Το βασικό χαρακτηριστικό του σχεδιασμού είναι η χρήση των παραλληλογράμμων στους βραχίονες, το οποίο διατηρεί τον προσανατολισμό του end effector.

Τα Delta ρομπότ έχουν δημοφιλή χρήση στο packaging σε εργοστάσια, επειδή μπορούν να είναι αρκετά γρήγορα, εκτελώντας έως και 300 κινήσεις ανά λεπτό.

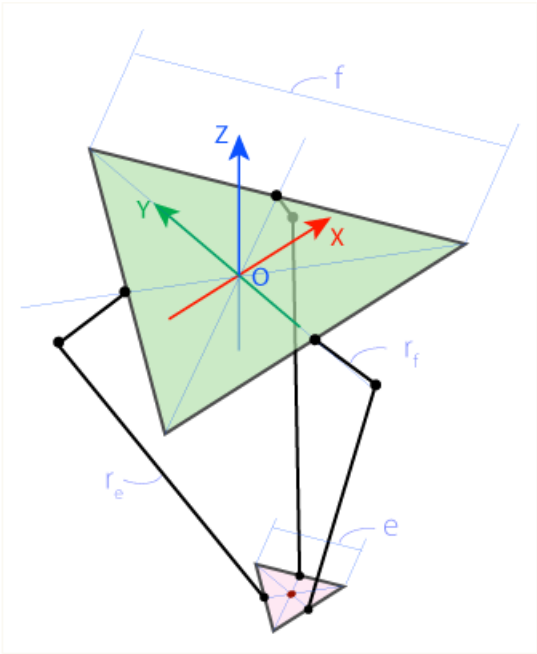
Το δέλτα ρομπότ επινοήθηκε στις αρχές του 1980 από μια ερευνητική ομάδα με επικεφαλής τον καθηγητή Reymond Clavel στο École Polytechnique Fédérale de Lausanne (EPFL , Ελβετία). Ο σκοπός αυτού του νέου τύπου του ρομπότ ήταν να χειριστούν ελαφριά και μικρά αντικείμενα σε μια πολύ υψηλή ταχύτητα, μια βιομηχανική ανάγκη εκείνης της εποχής. Το 1987 , η ελβετική εταιρεία Demaurex αγόρασε μια άδεια χρήσης για το ρομπότ δέλτα και ξεκίνησε την παραγωγή των δέλτα ρομπότ για τη βιομηχανία συσκευασίας. Το 1991 ο Reymond Clavel παρουσίασε τη διδακτορική του διατριβή «Σχεδιασμός ενός γρήγορου παράλληλου ρομπότ 4 βαθμών ελευθερίας» και έλαβε το χρυσό βραβείο ρομποτικής το 1999 για το έργο του και την ανάπτυξη του δέλτα ρομπότ. Επίσης το 1999, η ABB Flexible Automation άρχισε να πουλά το δέλτα ρομπότ της, το FlexPicker. Μέχρι το τέλος του 1999, δέλτα ρομπότ πωλήθηκαν επίσης από τη Sigpack Systems.

Το 2009, η FANUC κυκλοφόρησε την τελευταία έκδοση του Δέλτα ρομπότ, το FANUC M-1iA Robot, και θα κυκλοφορήσει αργότερα παραλλαγές αυτού του Delta ρομπότ για βαρύτερα ωφέλιμα φορτία. Συγκεκριμένα κυκλοφόρησε το M-3iA το 2010 για τα βαρύτερα ωφέλιμα φορτία και πιο πρόσφατα το FANUC M-2iA Robot για μεσαία ωφέλιμα φορτία το 2012.

Απαραίτητος για τη συνέχεια είναι ο προσδιορισμός κάποιον παραμέτρων του ρομπότ δέλτα και η ονομασία τους, οι οποίες θα αναφέρονται συνεχώς στην παρούσα εργασία. Έχουμε λοιπόν τα παρακάτω:

- τρίγωνο βάσης με πλευρά f
- r_f
- r_e
- τρίγωνο end effector με πλευρά e

Οι παράμετροι αυτές φαίνονται και στο Σχήμα 1.1.



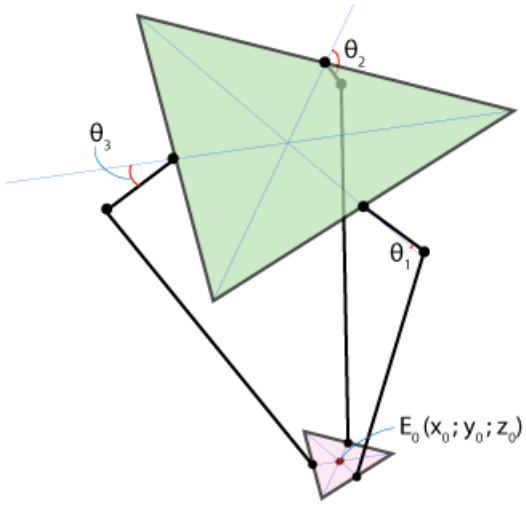
Σχήμα 1.1. Ορισμός των παραμέτρων του delta robot.

1.3. Kinematics

1.3.1. Problem definition

Η ανάλυση του ρομπότ περιλαμβάνει την επίλυση 2 προβλημάτων.
Αρχικά, γνωρίζουμε τις συντεταγμένες του end effector και να ψάχνουμε τις γωνίες των servo. Αυτό είναι το Inverse Kinematics, το οποίο θα χρησιμοποιήσουμε περισσότερο, αφού συνήθως γνωρίζουμε που θέλουμε να κινηθεί το end effector και ψάχνουμε τις κατάλληλες γωνίες των servo για την επίτευξη της κίνησης αυτής.
Επιπλέον, μπορεί να γνωρίζουμε τις γωνίες των servo και να ψάχνουμε τις συντεταγμένες του end effector. Αυτό είναι γνωστό ως Forward Kinematics.

Τα παραπάνω φαίνονται αναλυτικότερα στο Σχήμα 1.2.

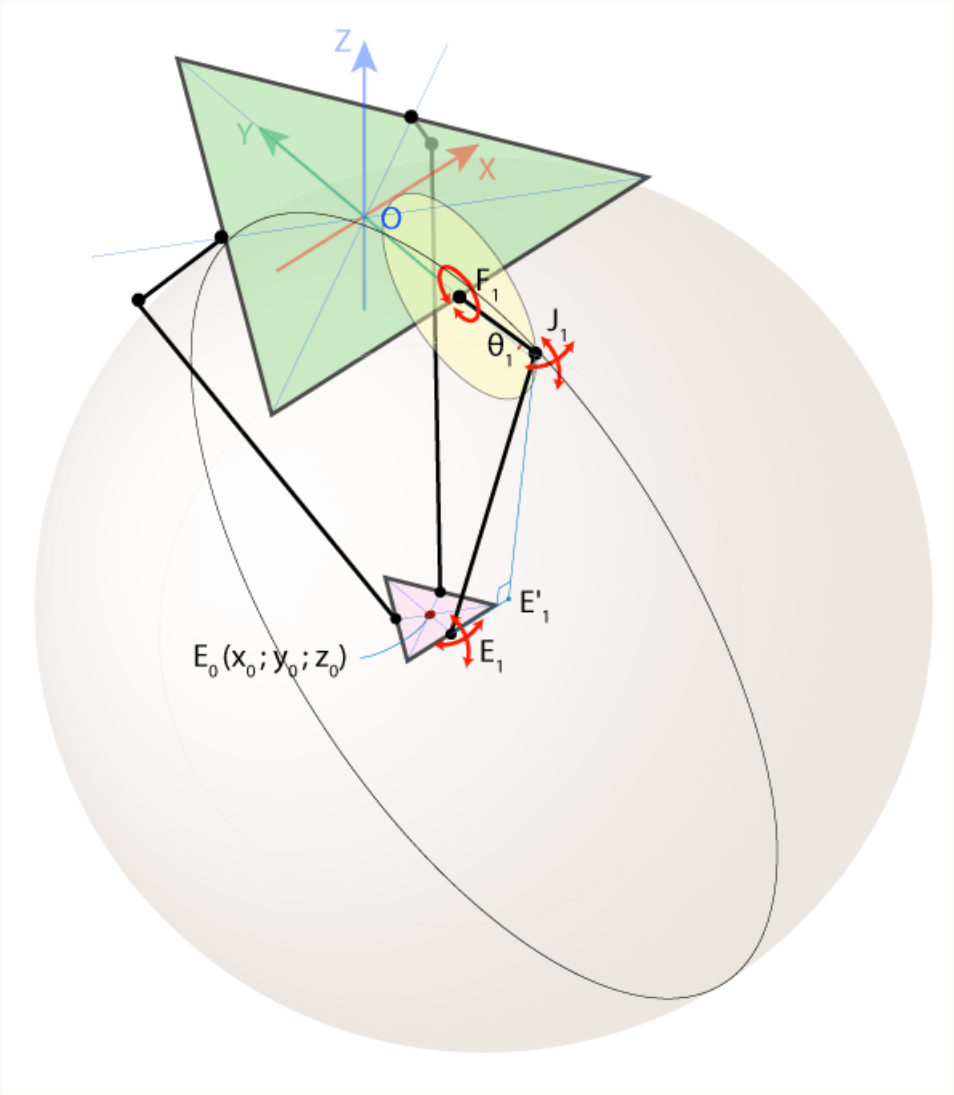


Σχήμα 1.2. Problem definition του delta robot.

Συμβολίζουμε με θ_1, θ_2, θ_3 τις γωνίες των servo και με $E_0(x_0,y_0,z_0)$ τη θέση του end effector. Επομένως στο Forward Kinematics χρειαζόμαστε μια συνάρτηση που να της δίνουμε τα θ_1, θ_2, θ_3 και να μας επιστρέφει τα x_0,y_0,z_0, ενώ για το Inverse Kinematics το αντίστροφο.

1.3.2. Inverse Kinematics

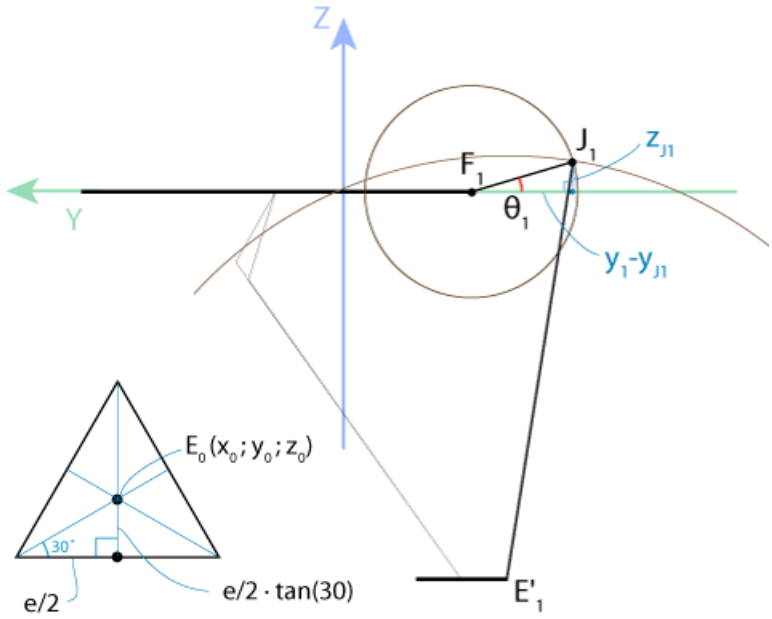
Έχουμε ως δεδομένα τις συντεταγμένες του end effector E_0 (x_0, y_0, z_0) και θέλουμε να βρούμε τις γωνίες θ_1, θ_2, θ_3 των servo.



Σχήμα 1.3. Inverse Kinematics

Έχοντας ως βάση το σχήμα 1.3, βλέπουμε ότι το κομμάτι F_1J_1 (r_f) μπορεί να κινηθεί μόνο στο επίπεδο YZ, σχηματίζοντας έναν κύκλο με κέντρο το F_1 και ακτίνα r_f. Αντίθετα με το F_1, οι σύνδεσμοι J_1 και E_1 είναι ελεύθερες αρθρώσεις, κάτι που σημαίνει ότι το κομμάτι J_1E_1 μπορεί να κινείται ελεύθερο στο χώρο, δημιουργώντας μια σφαίρα με κέντρο E_1 και ακτίνα r_e.

Η τομή αυτής της σφαίρας με το YZ επίπεδο μας δίνει έναν κύκλο με κέντρο E_1' και ακτίνα $E_1'J_1$, όπου E_1' είναι η προβολή του σημείου E_1 πάνω στο YZ επίπεδο. Πλέον το σημείο J_1 μπορεί να βρεθεί ως η τομή των δύο κύκλων (επιλέγουμε το σημείο με το μικρότερο y). Από τη στιγμή που γνωρίζουμε τις συντεταγμένες του σημείου J_1, μπορούμε να υπολογίσουμε τη γωνία θ_1.



Σχήμα 1.4. μαθηματική ανάλυση του Inverse Kinematics

Αναλυτικά, έχουμε:

$E_0(x_0,y_0,z_0)$ η θέση του end effector
$E_0E_1 = \frac{e}{2}\tan 30^o = \frac{e}{2\sqrt{3}}$
Άρα $E_1(x_0, y_0-\frac{e}{2\sqrt{3}}, z_0)$, επομένως $E_1'(0, y_0-\frac{e}{2\sqrt{3}}, z_0)$
Αφού $E_1E_1' = x_0$, $E_1'J_1 = (E_1J_1^2 - E_1E_1'^2)^{1/2} = (r_e^2 - x_0^2)^{1/2}$

$F_1(0, -\frac{f}{2\sqrt{3}}, 0)$

Άρα $(y_{J1} - y_{F1})^2 + (z_{J1} - z_{F1})^2 = r_f^2$
$(y_{J1} - y_{E'1})^2 + (z_{J1} - z_{E'1})^2 = r_e^2 - x_0^2$

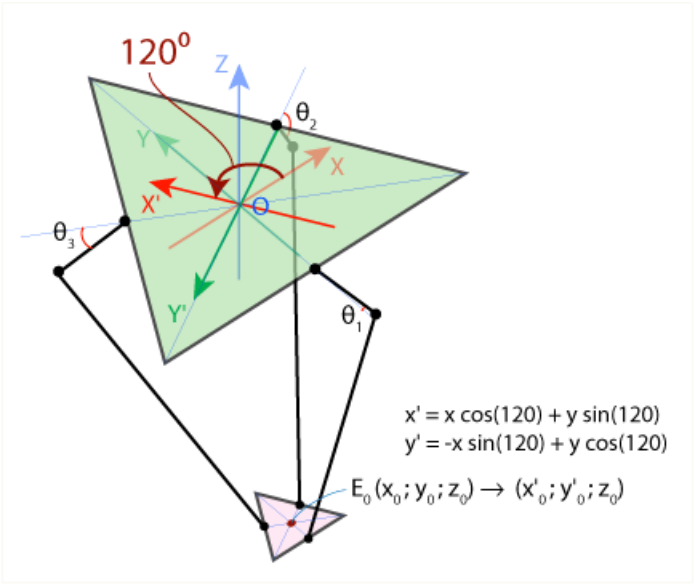
Αντικαθιστώντας τις γνωστές τιμές, έχουμε:

$(y_{J1} + \frac{f}{2\sqrt{3}})^2 + z_{J1}^2 = r_f^2$
$(y_{J1} - y_0 + \frac{e}{2\sqrt{3}})^2 + (z_{J1} - z_0)^2 = r_e^2 - x_0^2$

Λύνοντας αυτό το σύστημα, βρίσκουμε τις συντεταγμένες του σημείου J_1,
άρα έχω $J_1(0, y_{J1}, z_{J1})$.

Επομένως εύκολα προκύπτει ότι $\theta_1 = \arctan(z_{J1}/(y_{F1} - y_{J1}))$.

Έχοντας βρει τη γωνία θ_1, είναι πολύ εύκολο να βρούμε τις θ_2, θ_3 λόγω συμμετρίας του ρομπότ. Για την θ_2, αρκεί να περιστρέψουμε τα αποτελέσματά μας κατά 120^o αντίθετα από τη φορά του ρολογιού, γύρω από τον άξονα z, όπως φαίνεται στο Σχήμα 1.5.

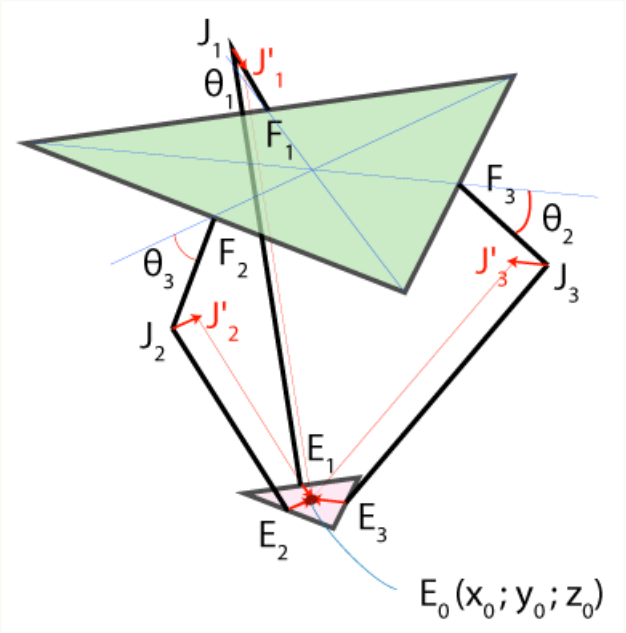


Σχήμα 1.5. Προσδιορισμός θ_2 μέσω συμμετρίας.

Για τη θ_3 ακολουθούμε την ίδια διαδικασία, περιστρέφοντας το σύστημα κατά 120^o κατά τη φορά του ρολογιού γύρω από τον άξονα z, σε σχέση με τη θ_1.

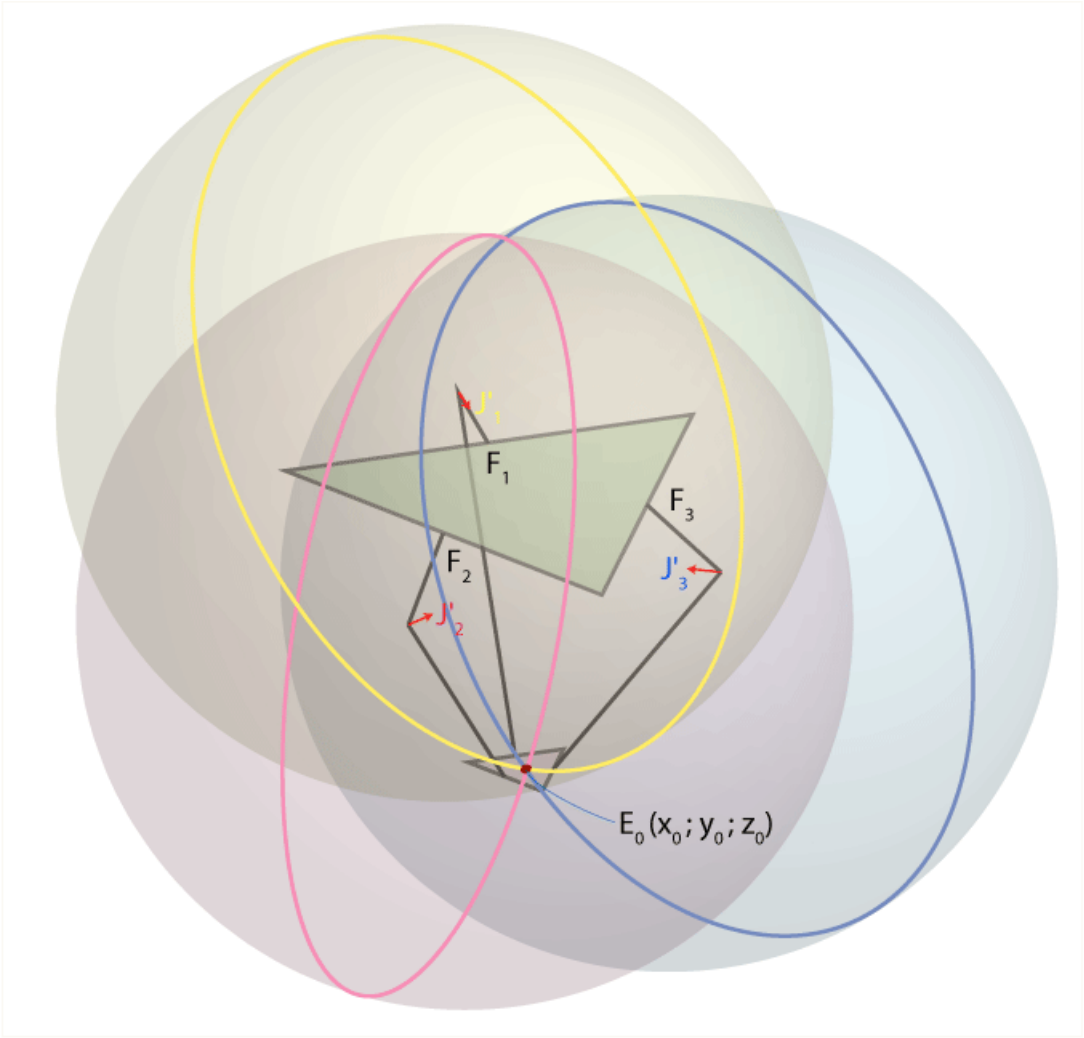
1.3.3. Forward Kinematics

Έχουμε ως δεδομένα τις θ_1, θ_2, θ_3 και θέλουμε να βρούμε τις συντεταγμένες του E_0, δηλαδή τα (x_0, y_0, z_0).



Σχήμα 1.6. Forward Kinematcs.

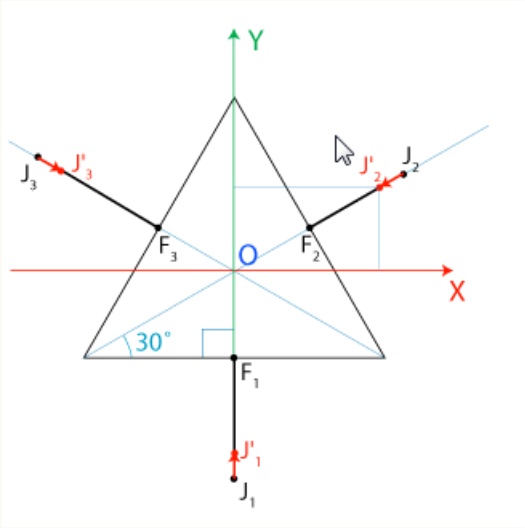
Οι σύνδεσμοι J_1E_1, J_2E_2, J_3E_3 μπορούν να κινούνται ελεύθερα γύρω από τις αρθρώσεις J_1, J_2, J_3 σχηματίζοντας 3 σφαίρες με ακτίνα r_e. Αν μετακινήσουμε τα κέντρα των σφαιρών από τα σημεία J_1, J_2, J_3 στα σημεία J_1', J_2', J_3' χρησιμοποιώντας ως transition vectors τα E_1E_0, E_2E_0, E_3E_0, τότε οι τρεις σφαίρες θα τέμνονται στο σημείο E_0 όπως φαίνεται στο σχήμα.



Σχήμα 1.7. Οι 3 σφαίρες που σχηματίζονται με ακτίνα r_e.

Άρα για την επίλυση του προβλήματος, αρκεί να λύσω 3 εξισώσεις της μορφής $(x-x_i)^2 + (y-y_i)^2 + (z-z_i)^2 = r_e^2$, όπου x_i, y_i, z_i οι γνωστές συντεταγμένες των κέντρων των σφαιρών J_1', J_2', J_3'.

Για την εύρεση των συντεταγμένων των J_1', J_2', J_3' ακολουθούμε τη διαδικασία όπως φαίνεται παρακάτω, με τη βοήθεια του σχήματος 1.8:



Σχήμα 1.8. Μαθηματική ανάλυση του Forward Kinematics.

$OF_1 = OF_2 = OF_3 = \frac{f}{2}\tan 30^o = \frac{f}{2\sqrt{3}}$
$J_1J_1' = J_2J_2' = J_3J_3' = \frac{e}{2}\tan 30^o = \frac{e}{2\sqrt{3}}$
$F_1J_1 = r_f\cos\theta_1$, $F_2J_2 = r_f\cos\theta_2$, $F_3J_3 = r_f\cos\theta_3$

$J_1'\ (0, -\frac{f-e}{2\sqrt{3}} - r_f\cos\theta_1, -r_f\sin\theta_1)$
$J_2'\ ([\frac{f-e}{2\sqrt{3}} + r_f\cos\theta_2]\cos 30^o, [\frac{f-e}{2\sqrt{3}} + r_f\cos\theta_2]\sin 30^o, -r_f\sin\theta_2)$
$J_3'\ (([\frac{f-e}{2\sqrt{3}} + r_f\cos\theta_3]\cos 30^o, [\frac{f-e}{2\sqrt{3}} + r_f\cos\theta_3]\sin 30^o, -r_f\sin\theta_3)$

Για $J_1(0,y_1,z_1)$, $J_2(x_2,y_2,z_2)$, $J_3(x_3,y_3,z_3)$, έχουμε τις παρακάτω εξισώσεις:

$x^2 + (y-y_1)^2 + (z-z_1)^2 = r_e^2$
$(x-x_2)^2 + (y-y_2)^2 + (z-z_2)^2 = r_e^2$
$(x-x_3)^2 + (y-y_3)^2 + (z-z_3)^2 = r_e^2$

$x^2 + y^2 + z^2 - 2y_1y - 2z_1z = r_e^2 - y_1^2 - z_1^2$ (1)
$x^2 + y^2 + z^2 - 2x_2x - 2y_2y - 2z_2z = r_e^2 - x_2^2 - y_2^2 - z_2^2$ (2)
$x^2 + y^2 + z^2 - 2x_3x - 2y_3y - 2z_3z = r_e^2 - x_3^2 - y_3^2 - z_3^2$ (3)

Θέτοντας $w_i = x_i^2 + y_i^2 + z_i^2$ για i = 1,2,3 έχουμε:

$x_2x + (y_1 - y_2)y + (z_1 - z_2)z = (w_1 - w_2)/2$ (4) = (1) - (2)
$x_3x + (y_1 - y_3)y + (z_1 - z_3)z = (w_1 - w_3)/2$ (5) = (1) - (3)
$(x_2 - x_3)x + (y_2 - y_3)y + (z_2 - z_3)z = (w_2 - w_3)/2$ (6) = (2) - (3)

Από τις (4), (5) προκύπτει:

$x = a_1z + b_1$ (7) και $y = a_2z + b_2$ (8)

όπου $a_1 = \frac{1}{d}[(z_2 - z_1)(y_3 - y_1) - (z_3 - z_1)(y_2 - y_1)]$, $a_2 = -\frac{1}{d}[(z_2 - z_1)x_3 - (z_3 - z_1)x_2]$,

$b_1 = -\frac{1}{2d}[(w_2 - w_1)(y_3 - y_1) - (w_3 - w_1)(y_2 - y_1)]$, $b_2 = \frac{1}{2d}[(w_2 - w_1)x_3 - (w_3 - w_1)x_2]$,

$d = (y_2 - y_1)x_3 - (y_3 - y_1)x_2$
Αντικαθιστώντας τις (7), (8) στην (1), παίρνουμε:

$(\alpha_1^2 + \alpha_2^2 + 1)z^2 + 2(a_1 + a_2(b_2 - y_1) - z_1)z + (b_1^2 + (b_2 - y_1)^2 + z_1^2 - r_e^2) = 0$

Λύνοντας αυτή την τελευταία εξίσωση βρίσκουμε το z_0 και μετά αντικαθιστώντας στις (7), (8) βρίσκουμε τα x_0, y_0.

2. Προσομοίωση στο Matlab

Πριν προχωρήσουμε στην κατασκευή, ήταν απαραίτητη η προσομοίωση στο περιβάλλον του Matlab, με τελικό σκοπό τον προσδιορισμό των διαστάσεων του ρομπότ, ώστε να μπορούμε να δουλέψουμε στο επιθυμητό workspace (200x200mm τουλάχιστον). Για το σκοπό αυτό χρησιμοποιήθηκαν οι παρακάτω συναρτήσεις:

2.1. Συναρτήσεις Κινηματικής

Οι συναρτήσεις αυτές προσομοιώνουν την ευθεία και αντίστροφη κινηματική του delta robot, όπως αυτή αναλύθηκε στο προηγούμενο κεφάλαιο.

2.1.1. DeltaInverse

```
function [q1_1,q2_1,q3_1,q1_2,q2_2,q3_2,ws] = DeltaInverse (DeltaRobot,x,y,z)
% Compute inverse kinematics of a DELTA Robot
% [q1,q2] = deltaInverse (DeltaRobot,x,y,z)
% DeltaRobot = Delta Robot Geometry
% x,y,z = Given Cartesian coordinates (you can use vectors with same dimension also)
% q1_1,q2_1,q3_1 = values corresponding joint angles in rad (first solution) for motor 1,2
and 3
% q1_2,q2_2,q3_2 = 3 dimentional vector corresponding joint angles in rad (second
% solution) for motor 1,2 and 3. For real robots this solution can be ignored
% ws = workspace of the robot. 1 = movement allowed, 0 = movement not
% allowed. If x,y,z are arrays (equal size) then ws is array with zeros and
% ones. You can use this array to draw robot's workspace
%
% Created by: Dimitris Piperidis (www.piperidis.co.nr)

% Motor1
[q1_1, q1_2, dr1] = GetAngle (DeltaRobot,x,y,z);
% Motor2
[q2_1, q2_2, dr2] = GetAngle (DeltaRobot,-x*0.5+y*sqrt(3)/2,-x*sqrt(3)/2-y*0.5,z);
% Motor3
[q3_1, q3_2, dr3] = GetAngle (DeltaRobot,-x*0.5-y*sqrt(3)/2 ,x*sqrt(3)/2-y*0.5,z);
% Calculate where dr is not valid and place zero
ws=dr1.*dr2.*dr3;
```

Σε αυτή τη συνάρτηση, βλέπουμε ότι χρησιμοποιήθηκε η βοηθητική συνάρτηση GetAngle, η οποία φαίνεται παρακάτω:

```
function [theta1, theta2, dmask] = GetAngle(DeltaRobot,x,y,z)
% Compute inverse kinematics of a DELTA Robot for one
% motor only.
%*************************************************
% ATTENTION! Do not use this function directly.
% Use DeltaInverse instead
%*************************************************
% [theta1, theta2, dmask] = GetAngle(DeltaRobot,x,y,z)
% DeltaRobot = Delta Robot Geometry
% X,Y,Z = Given Cartesian coordinates of the end effector
```

```
% dmask = This is the discriminant and angles (thetamax and thetamin) map.
% If dmask=0 then movement is not allowed
% If dmask=1 movement is allowed
%
% Created by: Dimitris Piperidis (www.piperidis.co.nr)


% Setup
f  = DeltaRobot(1);
e  = DeltaRobot(2);
rf = DeltaRobot(3);
re = DeltaRobot(4);
tmax = DeltaRobot(5);
tmin = DeltaRobot(6);

% Inverse kinematics equations
y1 = -f/(2*sqrt(3));
e0 = e/(2*sqrt(3));
y0 = y - e0;

a = (x.^2 + y0.^2 + z.^2 + rf.^2 - re.^2 - y1^2)./(2*z);
b = (y1 - y0)./z;

d = -(a + b.*y1).^2 + (rf.^2).*(1+b.^2);
dmask = d>=0;
d = dmask.*d;

yj = (y1 - a.*b - sqrt(d))./(1+b.^2);
zj = a + b.*yj;
theta1 = asin(zj/rf);
%theta1 = atan2(zj,y1-yj);

thetamask = theta1<=tmax & theta1>=tmin;
dmask=thetamask.*dmask;

yj = (y1 - a.*b + sqrt(d))./(1+b.^2);
zj = a + b.*yj;
theta2 = pi-asin(zj/rf);
```

2.1.2.DeltaForward

```
function [x0,y0,z0,ws] = DeltaForward(DeltaRobot,theta1,theta2,theta3)
% Compute Forward kinematics of a DELTA Robot

%
% Created by: Dimitris Piperidis (www.piperidis.co.nr)

% Setup
f  = DeltaRobot(1);
```

```
e  = DeltaRobot(2);
rf = DeltaRobot(3);
re = DeltaRobot(4);
tmax = DeltaRobot(5);
tmin = DeltaRobot(6);
x0=0;
y0=0;
z0=0;

t = (f - e)/(2*sqrt(3));
% Convert degrees to rad
theta1 = theta1*pi/180;
theta2 = theta2*pi/180;
theta3 = theta3*pi/180;

% find coordinates of J'1
y1 = -(t + rf*cos(theta1));
z1 = rf*sin(theta1);

% find coordinates of J'2
y2 = (t + rf*cos(theta2))/2;
x2 = y2*sqrt(3);
z2 = rf*sin(theta2);

% find coordinates of J'3
y3 = (t + rf*cos(theta3))/2;
x3 = -y3*sqrt(3);
z3 = rf*sin(theta3);

dnm = (y2-y1)*x3 - (y3-y1)*x2;
w1 = y1^2 + z1^2;
w2 = x2^2 + y2^2 + z2^2;
w3 = x3^2 + y3^2 + z3^2;

a1 = (z2-z1)*(y3-y1)-(z3-z1)*(y2-y1);
b1 = -((w2-w1)*(y3-y1)-(w3-w1)*(y2-y1))/2;

a2 = (z3-z1)*x2-(z2-z1)*x3;
b2 = ((w2-w1)*x3 - (w3-w1)*x2)/2;

% a*z^2 + b*z + c = 0
a = 4*(a1^2 + a2^2 + dnm^2);
b = 4*(a1*b1 + a2*(b2-2*y1*dnm) - 2*z1*dnm^2);
c = b1^2 + (b2-2*y1*dnm)^2 + 4*(z1^2 - re^2)*dnm^2;

% discriminant
d = b^2 - 4*a*c;
if (d < 0)
   ws=0; % non-existing point
else
  z0 = -(b+sqrt(d))/(2*a);
```

```
 %x = (2*a1*z + b1)/2dnm
 x0 = (2*a1*z0 + b1)/(2*dnm);
 % y = (2*a2*z + b2)/2dnm;
 y0 = (2*a2*z0 + b2)/(2*dnm);

 if (theta1<tmax) && (theta2<tmax) && (theta3<tmax) && (theta1>tmin) && (theta2>tmin)
&& (theta3>tmin)
   ws=1;
 else
   ws=0;
 end
end
```

2.2. Συναρτήσεις προσομοίωσης

2.2.1.DeltaSimulate

```
function [Q1,Q2,Q3,x,y,z,WS] = DeltaSimulate (DeltaRobot,x,y,z,threeD)
% Simulates a DELTA Robot
% deltaSimulate (DeltaRobot,x,y,z)
% DeltaRobot = Delta Robot Geometry
% X,Y,Z = Given Cartesian coordinates of the end effector
%
% Created by: Dimitris Piperidis (www.piperidis.co.nr)

clf
hold on
if (threeD==1)
  view(3)
end
grid on
axis ([-200 200 -200 200 -350 50])


% Setup length (mm)
f  = DeltaRobot(1);
e  = DeltaRobot(2);
rf = DeltaRobot(3);
re = DeltaRobot(4);
step_angle = DeltaRobot(7);


%Draw ChessBoard
chess = [8 8];
step_chess_x = 200/chess(1);
step_chess_y = 200/chess(2);
for j=0:chess(1)
   plot3([-100+j*step_chess_x,-100+j*step_chess_x],[-100,100],[z z],'k-')
end
for j=0:chess(2)
```

```
   plot3([-100,100],[-100+j*step_chess_y,-100+j*step_chess_y],[z z],'k-')
end



[q1,q2,q3,q4,q5,q6, WS] = DeltaInverse (DeltaRobot,x,y,z);
[Q1,Q2,Q3] = AngleToDegrees(q1,q2,q3);

% Keep in mind tha servos have a step angle
Q1 = round(Q1/step_angle)*step_angle;
Q2 = round(Q2/step_angle)*step_angle;
Q3 = round(Q3/step_angle)*step_angle;

[x,y,z,WS] = DeltaForward(DeltaRobot,Q1,Q2,Q3);
if (WS==1)
 % Redo Inverse kinematics
 [q1,q2,q3,q4,q5,q6, WS] = DeltaInverse (DeltaRobot,x,y,z);
 if (WS==1)
   % plot the Base triangle (fixed triangle)
   plot3 ([0,f/2,-f/2,0] , [f/sqrt(3),-f/(2*sqrt(3)),-f/(2*sqrt(3)),f/sqrt(3)] , [0,0,0,0] ,'r');
   % plot the 3 diagonals (this will help to choose the correct angles)
   plot3 ([-f/2,f/4] , [-f/(2*sqrt(3)),f/(4*sqrt(3))] , [0,0] ,'r');
   plot3 ([f/2,-f/4] , [-f/(2*sqrt(3)),f/(4*sqrt(3))] , [0,0] ,'r');
   plot3 ([0,0] , [f/sqrt(3),-f/(2*sqrt(3))] , [0,0] ,'r');
   % plot first arm
   plot3 ([0,0,x] , [-f/(2*sqrt(3)),-rf*cos(q1)-f/(2*sqrt(3)),y-e/(2*sqrt(3))] , [0,rf*sin(q1),z]
,'g');
   % plot second arm
   plot3 ([f/4,f/4+rf*cos(q2)*sqrt(3)/2,x+e/4] ,
[f/(4*sqrt(3)),f/(4*sqrt(3))+rf*cos(q2)/2,y+e/(4*sqrt(3))] , [0,rf*sin(q2),z] ,'g');
   % plot third arm
   plot3 ([-f/4,-f/4-cos(q3)*sqrt(3)*rf/2,x-e/4] ,
[f/(4*sqrt(3)),f/(4*sqrt(3))+rf*cos(q3)/2,y+e/(4*sqrt(3))] , [0,rf*sin(q3),z] ,'g');
   % plot the end effector triangle
   plot3 ([x,x+e/2,x-e/2,x] , [y+e/sqrt(3),y-e/(2*sqrt(3)),y-e/(2*sqrt(3)),y+e/sqrt(3)] , [z,z,z,z]
,'b') ;
   % plot the 3 diagonals
   plot3 ([x-e/2,x+e/4] , [y-e/(2*sqrt(3)),y+e/(4*sqrt(3))] , [z,z] ,'b');
   plot3 ([x+e/2,x-e/4] , [y-e/(2*sqrt(3)),y+e/(4*sqrt(3))] , [z,z] ,'b');
   plot3 ([x,x] , [y+e/sqrt(3),y-e/(2*sqrt(3))] , [z,z] ,'b');

%
**********************************************************************
**********************************
%   Calculate rf. Calculated rf must be equal to rf from geometry (DeltaRobot(3))
%   RF1 =sqrt((-rf*cos(q1)-f/(2*sqrt(3)) + f/(2*sqrt(3)))^2 + (rf*sin(q1))^2);
%   RF2 =sqrt((f/4+rf*cos(q2)*sqrt(3)/2 - f/4)^2 + (f/(4*sqrt(3))+rf*cos(q2)/2 -
f/(4*sqrt(3)))^2 + (rf*sin(q2))^2);
%   RF3 =sqrt((-f/4-cos(q3)*sqrt(3)*rf/2 + f/4)^2 + (f/(4*sqrt(3))+rf*cos(q3)/2 -
f/(4*sqrt(3)))^2 + (rf*sin(q3))^2);
%   RF1 and RF2 and RF3 must be equal to rf = DeltaRobot(3)
```

```
%    *** TESTED OK!!! ***

%
**********************************************************************
************************************
%    Calculate re. Calculated rf must be equal to re from geometry (DeltaRobot(4))
%    RE1 =sqrt(x^2 + (y-e/(2*sqrt(3))+rf*cos(q1)+f/(2*sqrt(3)))^2 + (z-rf*sin(q1))^2);
%    RE2 =sqrt((x+e/4-f/4-rf*cos(q2)*sqrt(3)/2)^2 + (y+e/(4*sqrt(3))-f/(4*sqrt(3))-
rf*cos(q2)/2)^2 + (z-rf*sin(q2))^2);
%    RE3 =sqrt((x-e/4+f/4+cos(q3)*sqrt(3)*rf/2)^2 + (y+e/(4*sqrt(3))-f/(4*sqrt(3))-
rf*cos(q3)/2)^2 + (z-rf*sin(q3))^2);
%    RE1 and RE2 and RE3 must be equal to re = DeltaRobot(4)
%    *** TESTED OK!!! ***

  end
end
```

όπου χρησιμοποιήθηκε η βοηθητική συνάρτηση:

AngleToDegrees

```
function [Q1,Q2,Q3] = AngleToDegrees (q1,q2,q3)
% Convert Angles to degrees
% Use this function after a call to deltaInverse
% to convert the calculated angles to degrees

Q1 = q1*180/pi;
Q2 = q2*180/pi;
Q3 = q3*180/pi;
```

2.2.2. demo - CreateDelta

Με τη συνάρτηση demo, παίρνουμε μια προσομοίωση της τελικής θέσεις του ρομπότ για διάφορες θέσεις.

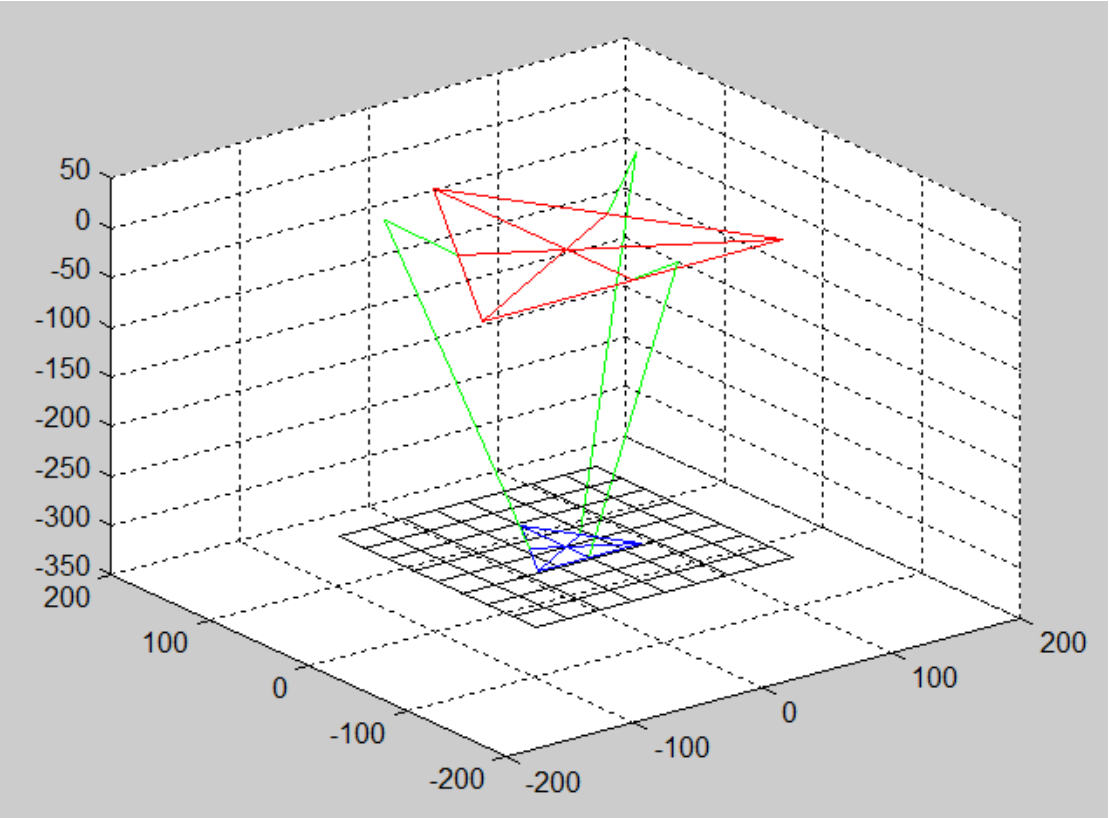
```
% This program will demonstrate motion control of the delta robot in Cartesian mode.
clear all
clc

% Create a Delta Robot with
% f = 233.2 mm
% e = 80 mm (calculated e = e' + d*2sqrt(3), where e' = measured and d the distance of the
joint from e)
% rf = 60 mm
% re = 350 mm
% umax = 80 degrees
% umin = -120 degrees
delta = CreateDelta(233.2,80,60,350,[80 -120],5);
```

```
% Go to zero position
DeltaSimulate(delta,0,0,-300,1);
```

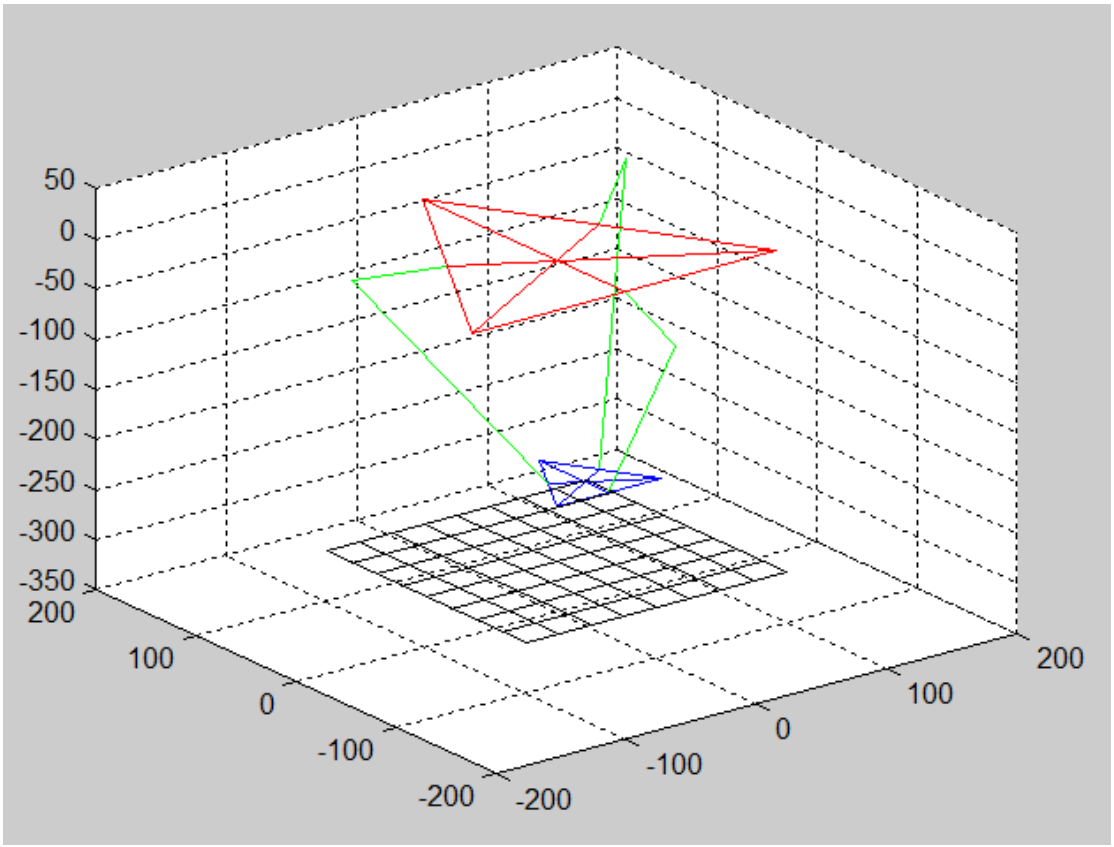
όπου χρησιμοποιήθηκε η βοηθητική συνάρτηση CreateDelta:

```
function delta = CreateDelta (f,e,rf,re,theta,theta_step)
% Creates a DELTA Robot manipulator
% f is the side of the fixed triangle (in mm)
% rf is the length of the upper joint (in mm)
% e is the side of the end effector triangle (in mm)
% re is the length of the parallelogram joint (in mm)
% theta is two element vector. First element is the maximum (positive)
% angle of the motors (in degrees). The second element is the minimum (negative)
% angle of the motors (in degrees)
%
% Created by: Dimitris Piperidis (www.piperidis.co.nr)

% convert angles to rad
thetamax = theta(1)*pi/180;
thetamin = theta(2)*pi/180;
delta = [f e rf re thetamax thetamin theta_step];
```

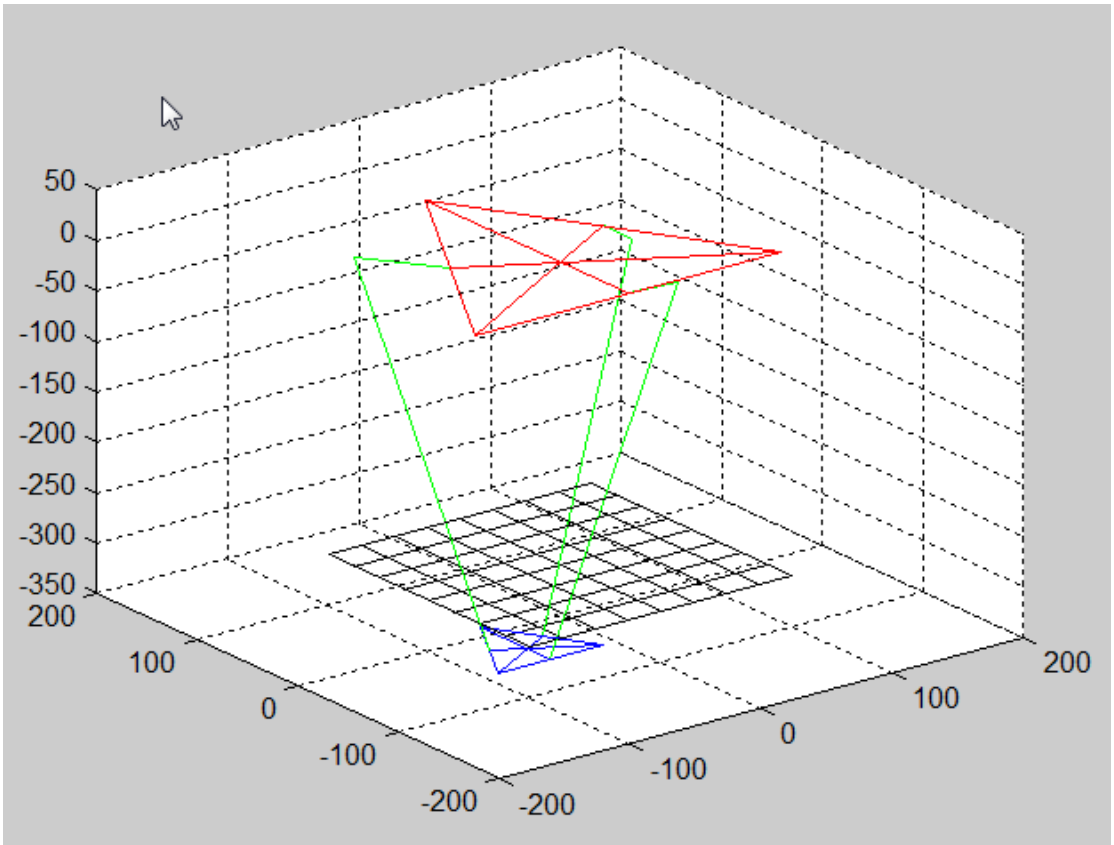
Αλλάζοντας τις παραμέτρους στην τελευταία γραμμή του κώδικα, παίρνουμε την προσομοίωση του ρομπότ για τη θέση που θέλουμε, όπως φαίνεται στις εικόνες 2.1, 2.2, 2.3.



Εικόνα 2.1. Προσομοίωση του ρομπότ στη θέση (0,0,-300).



Εικόνα 2.2. Προσομοίωση του ρομπότ στη θέση (100,100,-300).



Εικόνα 2.3. Προσομοίωση του ρομπότ στη θέση (-100,-100,-300).

2.3. Συναρτήσεις πεδίου δράσης

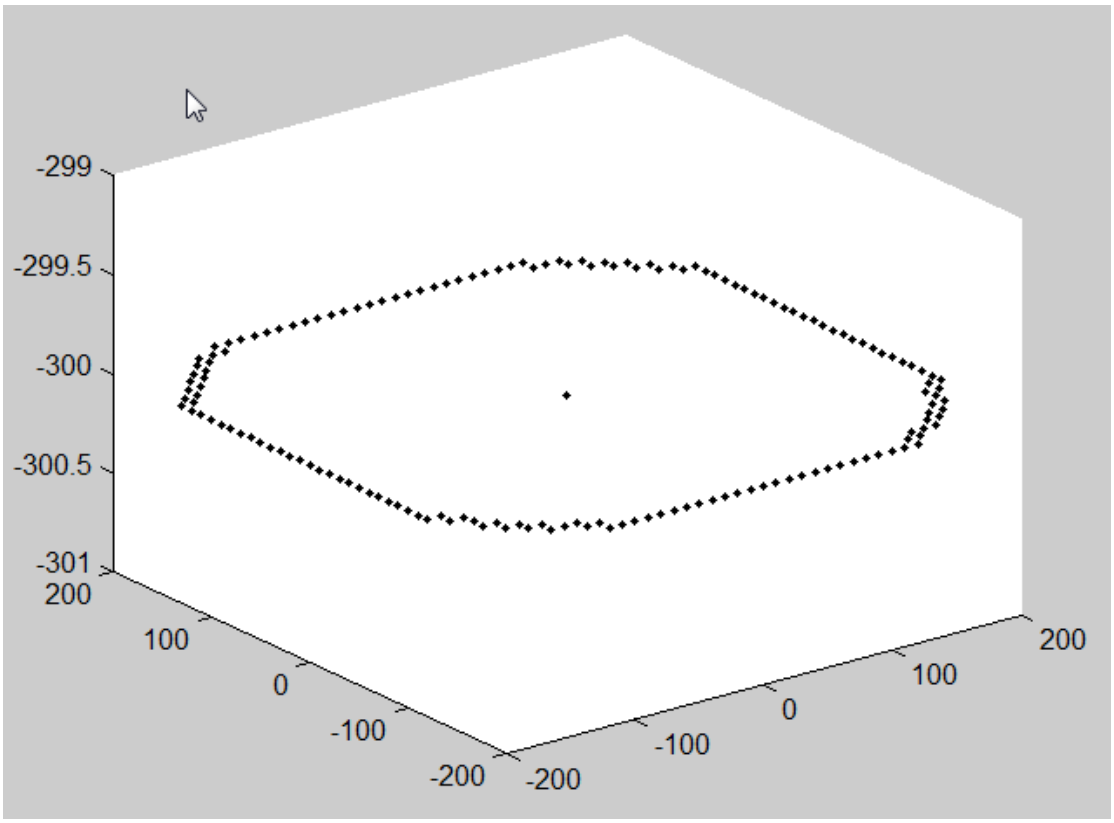
2.3.1. region

Με αυτή τη συνάρτηση βλέπουμε μια πρώτη εικόνα της περιοχής που μπορεί να καλύψει το ρομπότ με τις συγκεκριμένες διαστάσεις.

```
% This program will demonstrate delta's workspace (3D).
clear all
clc
% Create a Delta Robot with
% f = 233.2 mm
% e = 80 mm (calculated e = e' + d*2sqrt(3), where e' = measured and d the distance of the
joint from e)
% rf = 60 mm
% re = 350 mm
% umax = 80 degrees
% umin = -120 degrees
delta = CreateDelta(233.2,80,60,350,[80 -120],5);

% Conclusion: Even when oΩ½max =80 and oΩ½min=-120
% we can cover Desired xy area 200mm x 200mm (meaning -100=<x<=100
% and -100=<y<=100
% For this desired plane z = -300mm

% Create arrays
xx=-200:10:200;
M=length(xx);

x=ones(M);

for i=1:M
 x(:,i)=x(:,i)'.*xx;
end
y=x';
z=-300*ones(M);


grid on
xlabel('x axis (mm)');
ylabel('y axis (mm)');
zlabel('z axis (mm)');

for i=1:1
  z=z+10;
  [Q1 Q2 Q3 Q4 Q5 Q6 BW]=DeltaInverse(delta,x,y,z);

  BW = bwperim(BW,8);
  X=x.*BW;
  Y=y.*BW;
end
```

```
axis ([-200 200 -200 200 -350 0]);
plot3(X,Y,z,'k.');
```

Η περιοχή για τις διαστάσεις του δικού μας ρομπότ φαίνεται στην εικόνα 2.4.



Εικόνα 2.4. Περιοχή που καλύπτει το δικό μας delta robot

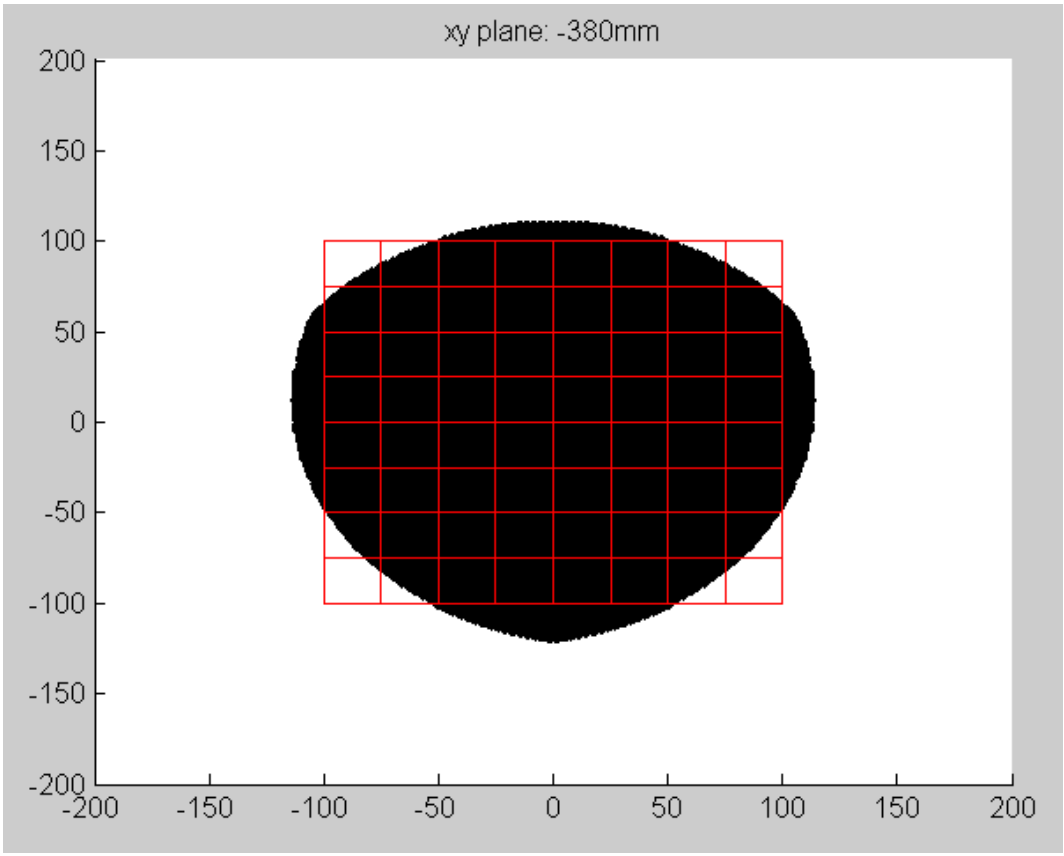
2.3.1. Workspace

Αυτή είναι και η κυριότερη συνάρτηση που χρησιμοποιήσαμε για να καταλήξουμε στις διαστάσεις του ρομπότ. Για διαφορετικά z, βλέπουμε το εύρος που μπορεί να καλύψει το ρομπότ.

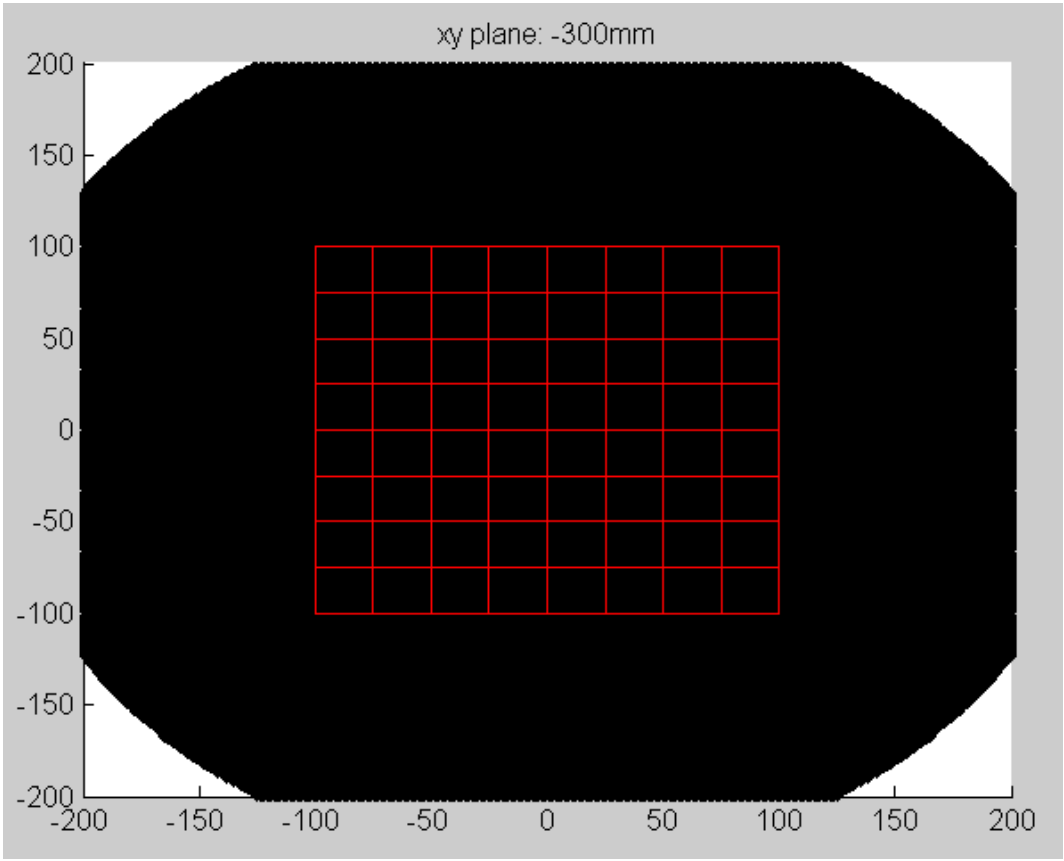
```
% This program will demonstrate delta's workspace.
clear all
clc

% Create a Delta Robot with
% f = 233.2 mm
% e = 80 mm (calculated e = e' + d*2sqrt(3), where e' = measured and d the distance of the
joint from e)
% rf = 60 mm
% re = 350 mm
% umax = 80 degrees
% umin = -120 degrees
delta = CreateDelta(233.2,80,60,350,[80 -120],5);
```

```
% Conclution: Even when oΩ½max =80 and oΩ½min=-120
% we can cover Desired xy area 200mm x 200mm (meaning -100=<x<=100
% and -100=<y<=100
% For this desired plane z = -300mm

% Create arrays
M=400;
x=ones(M);
for i=1:M
 x(:,i)=x(:,i)'.*linspace(-200,200,M);
end
y=x';

% Run the loop
% Each plane is 10mm higher than the previous one
% We start at 250 mm
z=-400*ones(M);
grid on
xlabel('x axis');
ylabel('y axis');

chess = 8;
step = 200/chess;
for i=1:40
  clf
  z=z+10;
  [Q1 Q2 Q3 Q4 Q5 Q6 BW]=DeltaInverse(delta,x,y,z);
  axis ([-200 200 -200 200]);
  hold on;
  title(['xy plane: ' int2str(z(1,1)) 'mm']);
  X=x.*BW;
  Y=y.*BW;
  plot (X ,Y, 'k.');
  % draw desired working rectangle
  % Desired xy area 200mm x 200mm
  plot([-100 -100 100 100 -100],[-100 100 100 -100 -100],'r-')
  for j=0:chess-1
    plot([-100+j*step -100+j*step],[-100 100],'r-')
  end
  for j=0:chess-1
    plot([-100 100],[-100+j*step -100+j*step],'r-')
  end
  % pause for a little while to see the result
  pause(0.5);
end
```

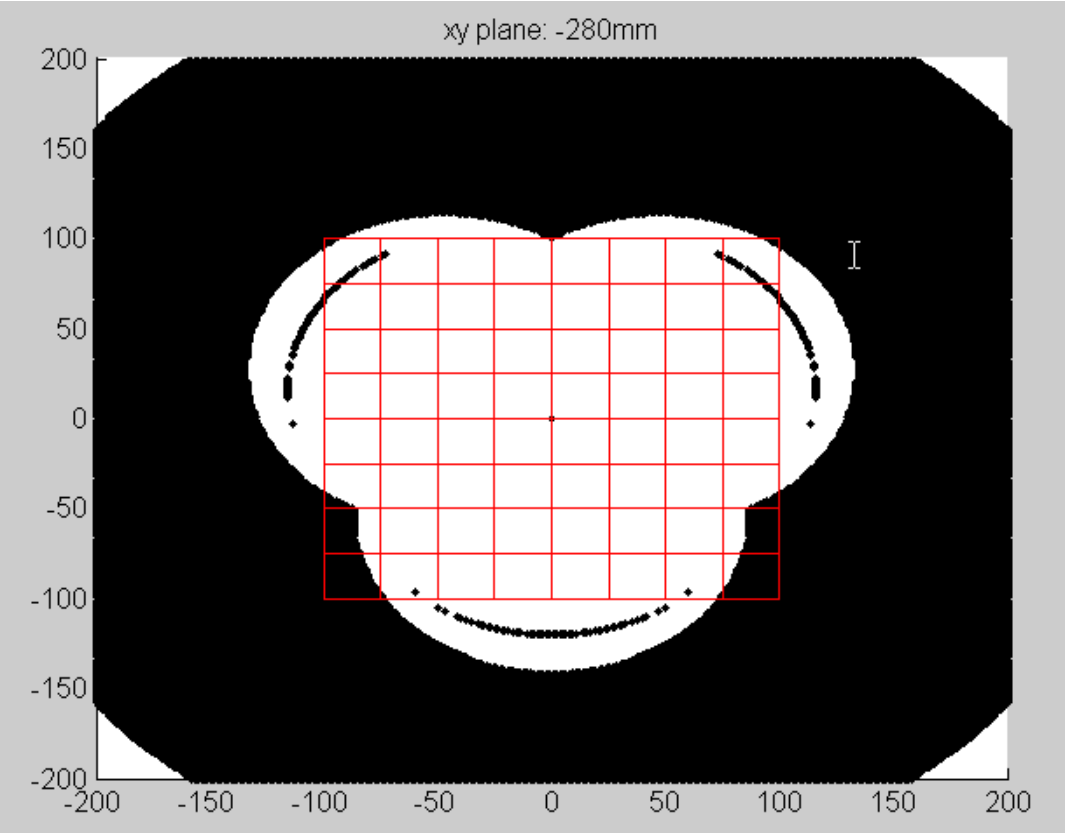
Στις εικόνες 2.5-2.7 φαίνεται το εύρος που καλύπτει το ρομπότ για z = -380, z = -300 και z = -280.



Εικόνα 2.5. Εύρος που καλύπτει το ρομπότ για z = -380.



Εικόνα 2.6. Εύρος που καλύπτει το ρομπότ για z = -300.



Εικόνα 2.7. Εύρος που καλύπτει το ρομπότ για z = -280.

Φυσικά, δεν αρκεί το ρομπότ να καλύπτει ακριβώς το χώρο 200mmx200mm. Πρώτον γιατί πιθανώς να χρειαστεί και λίγο χώρο έξω από αυτό, για να παίρνει και να αφήνει αντικείμενα και δεύτερον, γιατί, λόγω του ότι είναι μια προσομοίωση και όχι η πραγματικότητα, ήταν απαραίτητο να διασφαλίσουμε ότι δε θα έχουμε δυσάρεστες εκπλήξεις λόγω κατασκευαστικών και υπολογιστικών ατελειών.

Επομένως καταλήξαμε οι εργασίες του ρομπότ να γίνονται στο επίπεδο z=-300mm.

Τέλος, συμπληρωματικά, χρησιμοποιήσαμε και τη συνάρτηση για το 3D πεδίο δράσης.

2.3.3. Workspace 3d

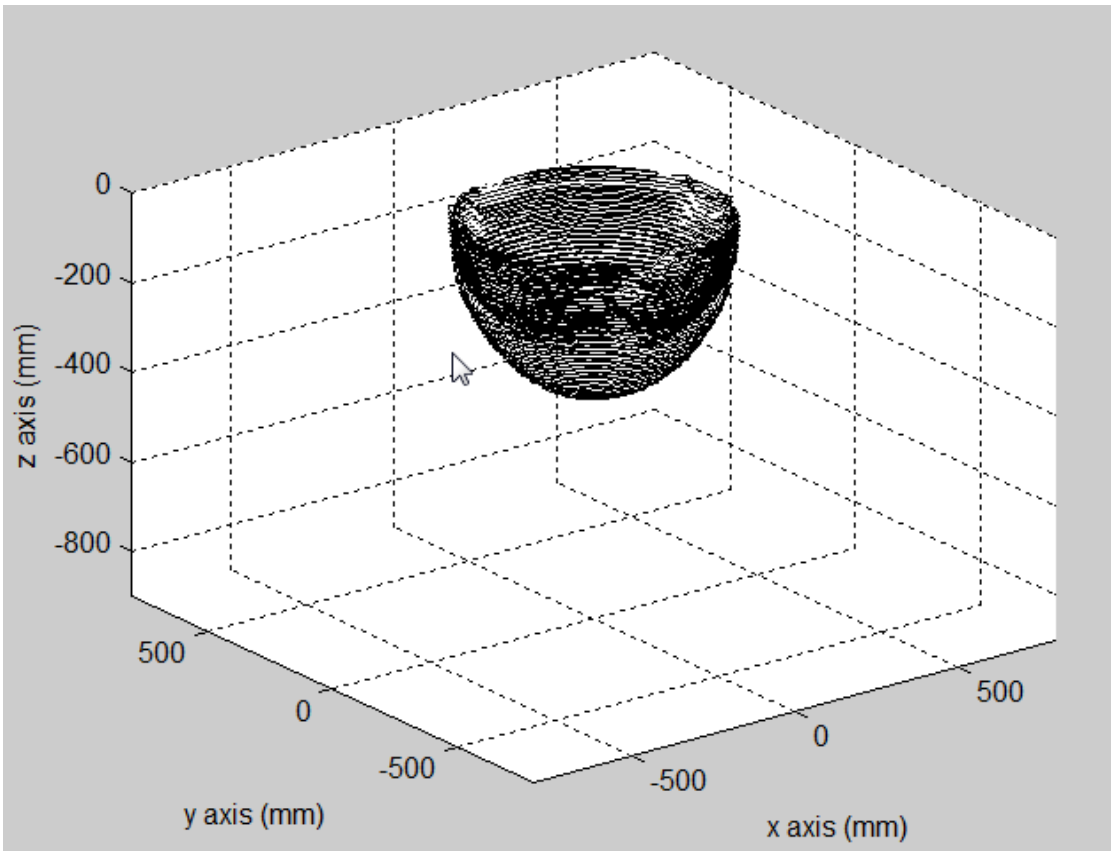
Αυτή η συνάρτηση μας δίνει το εύρος που καλύπτει το ρομπότ σε 3D απεικόνιση.

```
% This program will demonstrate delta's workspace (3D).
clear all
clc

% Create a Delta Robot with
% f = 233.2 mm
% e = 80 mm (calculated e = e' + d*2sqrt(3), where e' = measured and d the distance of the joint from e)
% rf = 60 mm
```

```
% re = 350 mm
% umax = 80 degrees
% umin = -120 degrees
delta = CreateDelta(233.2,80,60,350,[80 -120],5);

% Conclution: Even when oΩ½max =80 and oΩ½min=-120
% we can cover Desired xy area 200mm x 200mm (meaning -100=<x<=100
% and -100=<y<=100
% For this desired plane z = -300mm

% Create arrays
xypoints = 10;
zpoints = 10;
zpanel_start=-900;
xyaxis_min = -800;
xyaxis_max = 800;
zaxis_min =-900;
zaxis_max =0;


xx=xyaxis_min:xypoints:xyaxis_max;
M=length(xx);
x=ones(M);

for i=1:M
 x(:,i)=x(:,i)'.*xx;
end
y=x';

z=zpanel_start*ones(M);

% Run the loop
% Each plane is 10mm higher than the previous one
% We start at 250 mm
colors=['b' 'g' 'r' 'c' 'm' 'y'];
grid on
xlabel('x axis (mm)');
ylabel('y axis (mm)');
zlabel('z axis (mm)');

i=zpanel_start;
while (i<0)
  z=z+zpoints;
  [Q1 Q2 Q3 Q4 Q5 Q6 BW]=DeltaInverse(delta,x,y,z);

  [B,L] = bwboundaries(BW,'noholes');

%    cidx = mod(i,length(colors))+1;
  for k = 1:length(B)
    boundary = B{k};
    axis ([xyaxis_min xyaxis_max xyaxis_min xyaxis_max zaxis_min zaxis_max]);
```

```
    hold on
    XX=boundary(:,2)*xypoints-xyaxis_max;
    YY=boundary(:,1)*xypoints-xyaxis_max;
    ZZ=z(1,1)*ones(1,length(boundary(:,1)));
    plot3(XX, YY,ZZ, 'k')
  end

  i=i+zpoints;
end
```

Στην εικόνα 2.8 βλέπουμε τη 3D απεικόνιση.



Εικόνα 2.8. 3D απεικόνιση του πεδίου δράσης του ρομπότ.

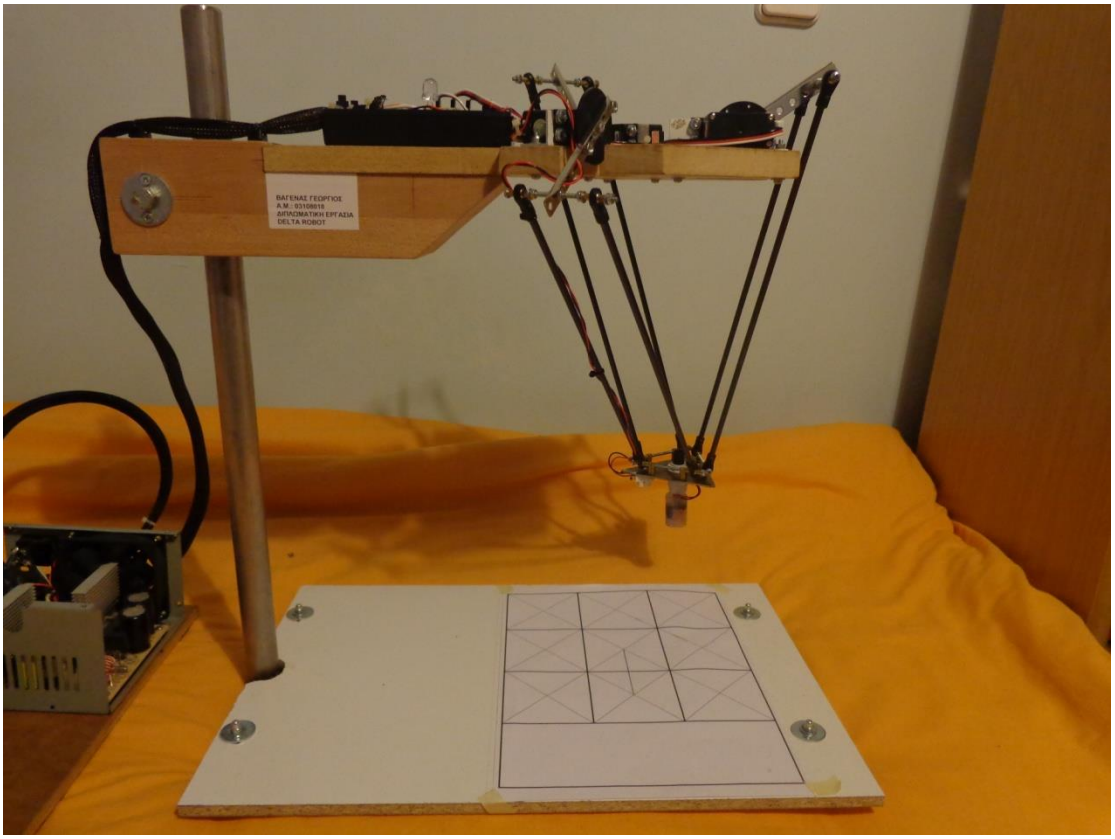
2.4. Συμπεράσματα

Με βάση λοιπόν αυτές τις προσομοιώσεις και με δεδομένο ότι θέλουμε να δουλέψουμε σε ένα workspace τουλάχιστον 200x200mm, δοκιμάσαμε διάφορες τιμές για τις διαστάσεις του ρομπότ. Καταλήξαμε σε διαστάσεις e = 80mm, f = 233.2mm, r_e = 350mm, r_f = 60mm, οι οποίες μας προσφέρουν αρκετά μεγαλύτερο εύρος από το επιθυμητό, χωρίς ωστόσο να μεγαλώνουν ανησυχητικά το συνολικό όγκο του ρομπότ, κάτι που μπορεί να φανεί ιδιαίτερα χρήσιμο σε μελλοντικές εφαρμογές, πέρα από τα πλαίσια αυτής της διπλωματικής εργασίας. Βλέπουμε δηλαδή ότι η επιφάνεια 200x200 καλύπτεται και για "ψηλότερα z" (z = -290, z = -280), ωστόσο για τους λόγους που αναφέρθηκαν και παραπάνω, προτιμήσαμε να χάσουμε ελαφρώς σε όγκο, κάνοντας το ρομπότ μεγαλύτερο, αλλά να κερδίσουμε σε ευελιξία και προσαρμοστικότητα σε διάφορες εφαρμογές.

Σημείωση: Όλες οι συναρτήσεις δημιουργήθηκαν από τον κύριο Πιπερίδη, όπως αναφέρεται σε σχόλια στον κώδικα της κάθε μίας. Αυτό που έγινε στα πλαίσια αυτής της διπλωματικής ήταν η κατάλληλη τροποποίησή τους, κυρίως δοκιμάζοντας διάφορες παραμέτρους, ώστε να εξυπηρετηθεί ο σκοπός μας.

Β. Σχεδίαση - Κατασκευή

3. Μηχανολογικό μέρος



Εικόνα 3.1. Μηχανολογικό μέρος του ρομπότ.

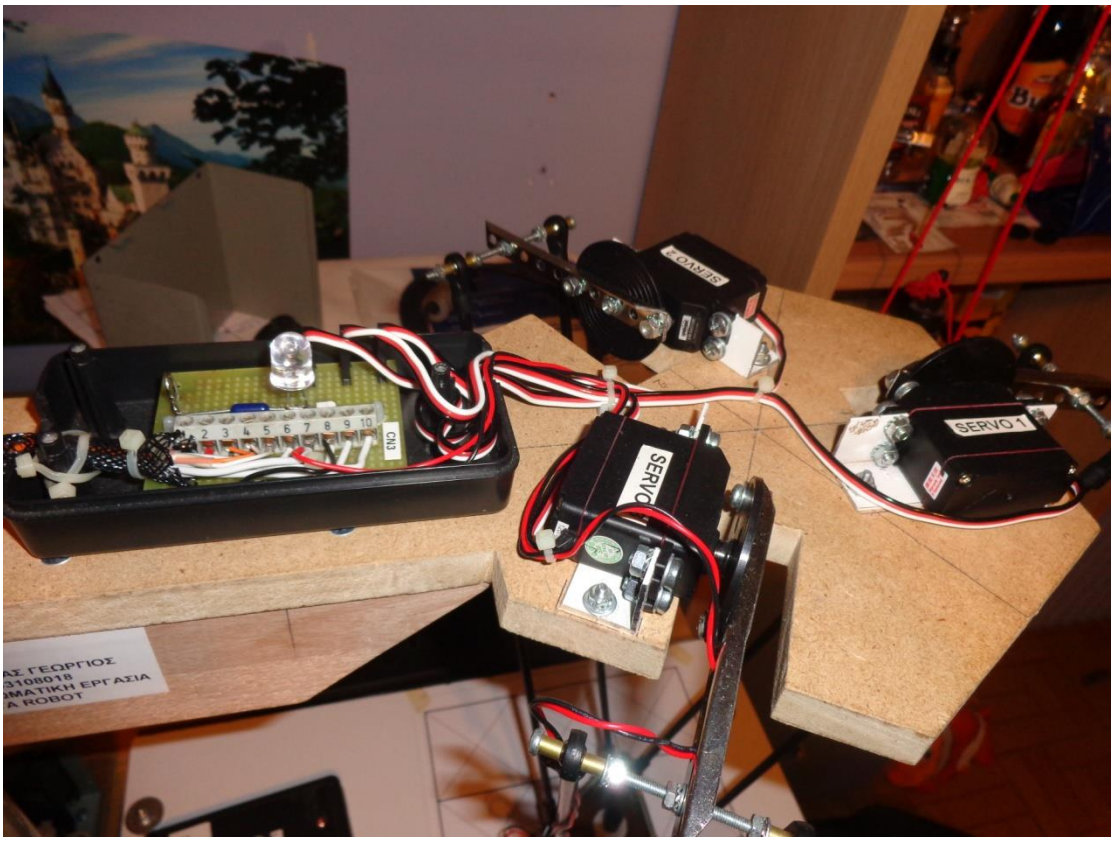
Για να γίνει ευκολότερη η επεξήγηση της κατασκευής, το ρομπότ έχει χωριστεί στα εξής τμήματα:

- βάση στήριξης σέρβο
- r_f - r_e - end effector
- φορέας
- οδηγός ύψους
- καστάνια ακινητοποίησης φορέα
- μεταλλική βάση
- τραπέζι εργασίας

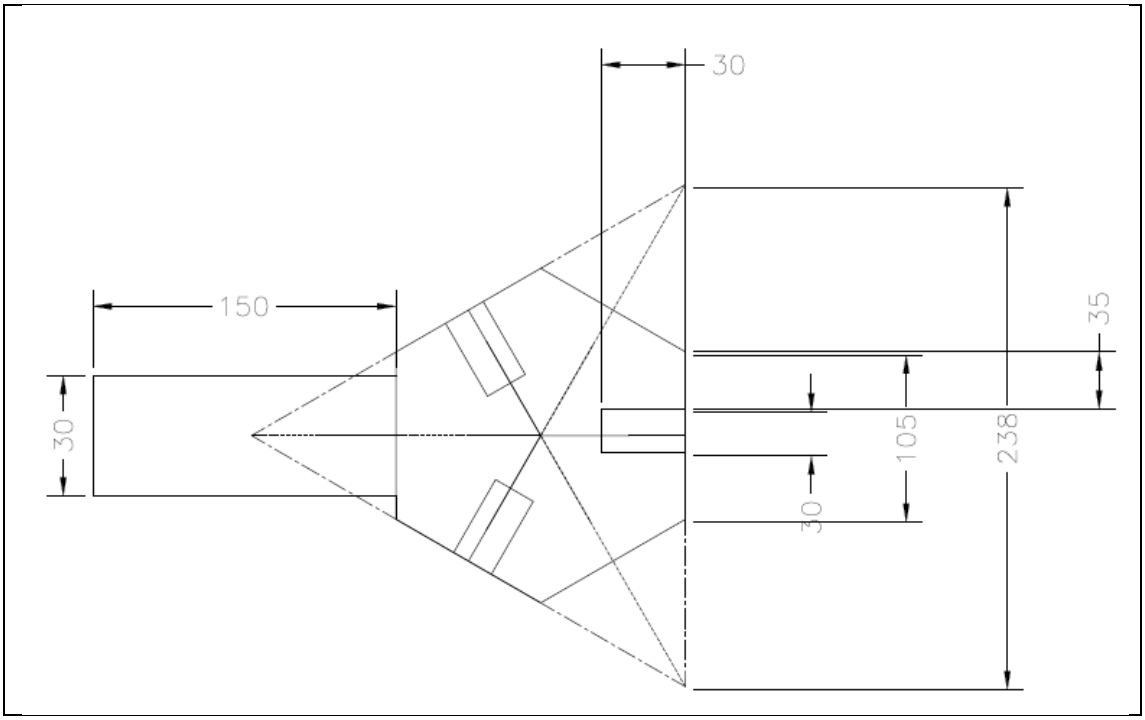
Τα τμήματα αυτά και ο τρόπος σύνδεσής τους αναλύονται λεπτομερώς σε αυτό το κεφάλαιο.

3.1. Βάση στήριξης servo

Η διαδικασία της κατασκευής ξεκίνησε με τη βάση στήριξης των servo (Εικόνα 3.2, Σχέδιο 3.1). Αυτό ήταν ίσως το σημαντικότερο βήμα της κατασκευής, αφού η σωστή τοποθέτηση των σέρβο καθορίζει σε μεγάλο βαθμό την ακρίβεια που θα έχει τελικά το ρομπότ.
Το υλικό που χρησιμοποιήθηκε είναι MDF πάχους 16mm.



Εικόνα 3.2. Βάση στήριξης των servo.



Σχέδιο 3.3. Μηχανολογικό σχέδιο της βάσης στήριξης των servo (κάτοψη).

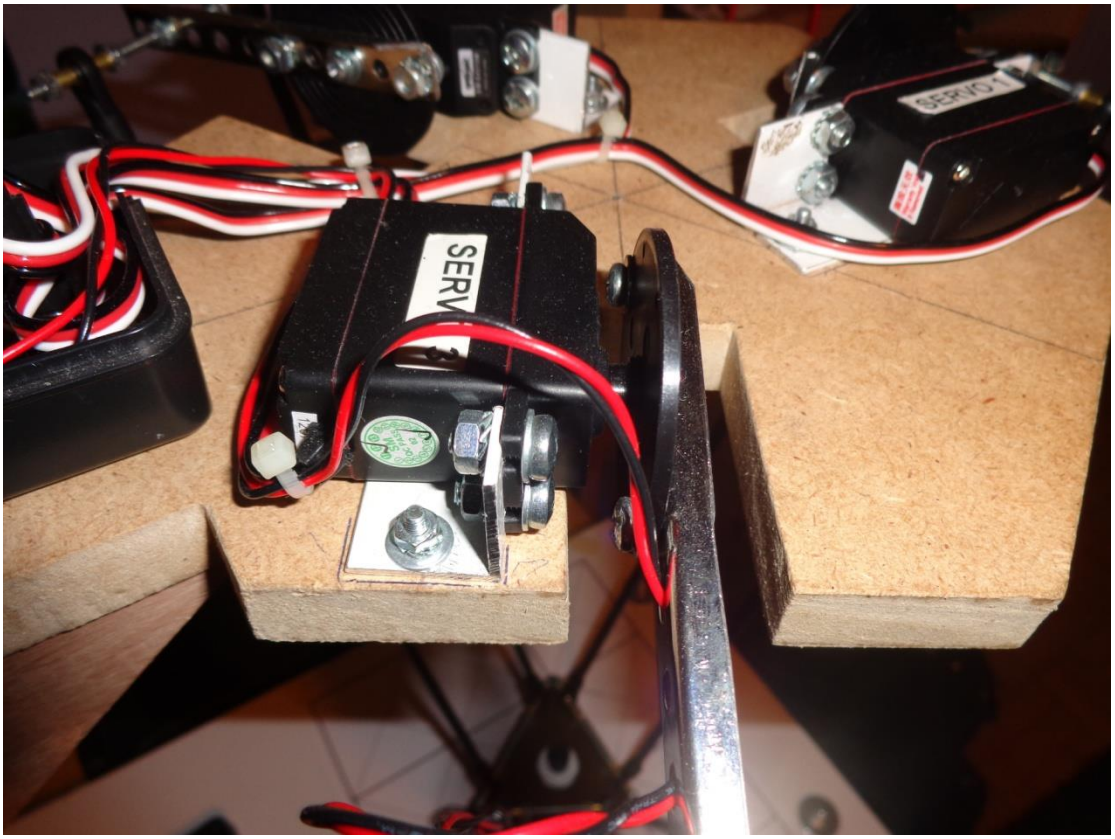
Για να τοποθετήσουμε σωστά τα σέρβο, ξεκινήσαμε από ένα σημείο, το οποίο είναι το κέντρο του τριγώνου f και με τριγωνομετρικούς υπολογισμούς βρήκαμε σε ποια απόσταση από αυτό το σημείο πρέπει να τοποθετηθεί το πρώτο σέρβο. Απαιτείται μεγάλη προσοχή ώστε ο άξονας περιστροφής του να συμπίπτει με την πλευρά f του τριγώνου της βάσης και το σημείο της άρθρωσης μεταξύ μηχανικού άξονα και r_f να βρίσκεται στο μέσο της πλευράς f. Θεωρούμε ότι όλα τα σέρβο βρίσκονται στο ίδιο επίπεδο, επειδή στερεώνονται με τον ίδιο ακριβώς τρόπο πάνω στη βάση στήριξής τους.

Τα άλλα δύο σέρβο θα τοποθετηθούν στην ίδια απόσταση, αλλά με μια περιστροφή $+120^o$ και -120^o σε σχέση με το πρώτο σέρβο.

Για την εύρεση της απόστασης του πρώτου σέρβο από το κέντρο του τριγώνου, ακολουθήσαμε την εξής διαδικασία:

$\tan 30^o$ = επιθυμητή απόσταση / (f/2) ⇔ επιθυμητή απόσταση = 6.73cm

Στη συνέχεια, με ένα παχύμετρο μετρήθηκαν οι διαστάσεις των σέρβο και των μεταλλικών γωνιών στηρίξεώς τους, καθώς και των γωνιακών συνδέσμων και προέκυψε το Σχέδιο 3.1. Με βάση αυτό το σχέδιο, έγιναν η κατάλληλες κοπές στο MDF και τοποθετήθηκαν τα σέρβο, όπως φαίνεται στην εικόνα 3.3.

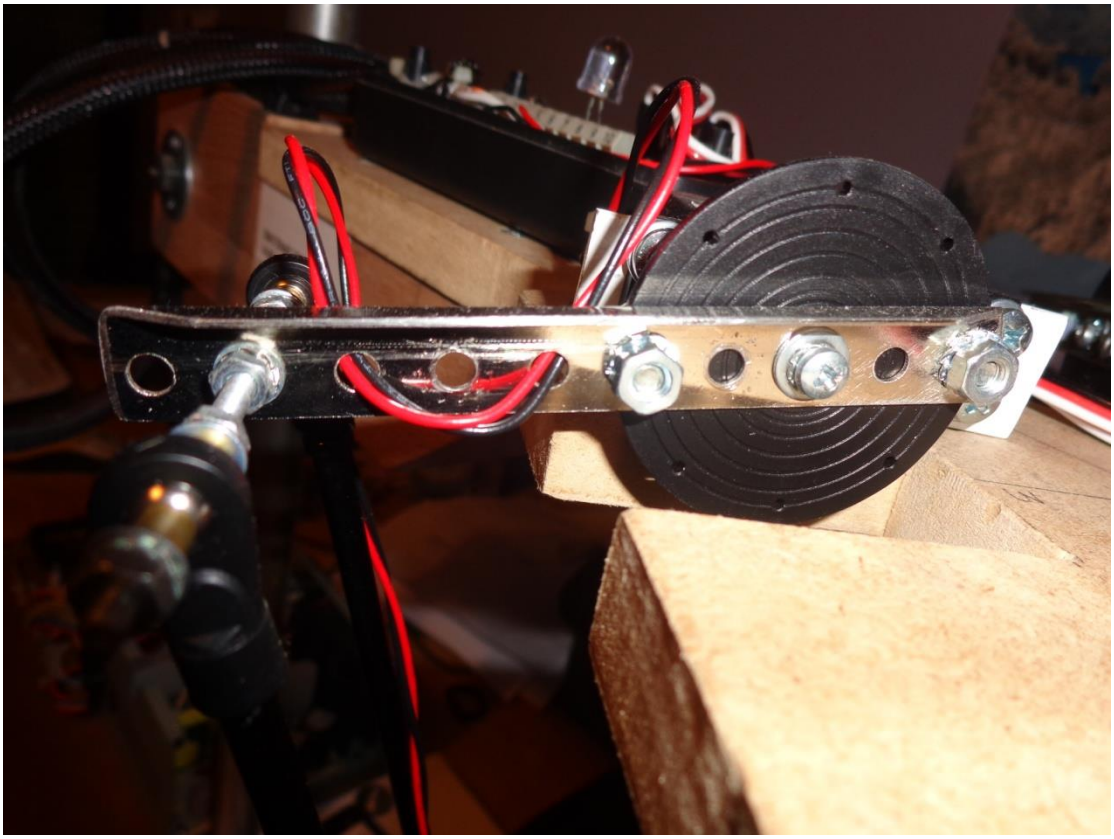


Εικόνα 3.3. Ακριβής τρόπος σύνδεσης των servo πάνω στη βάση στήριξής τους.

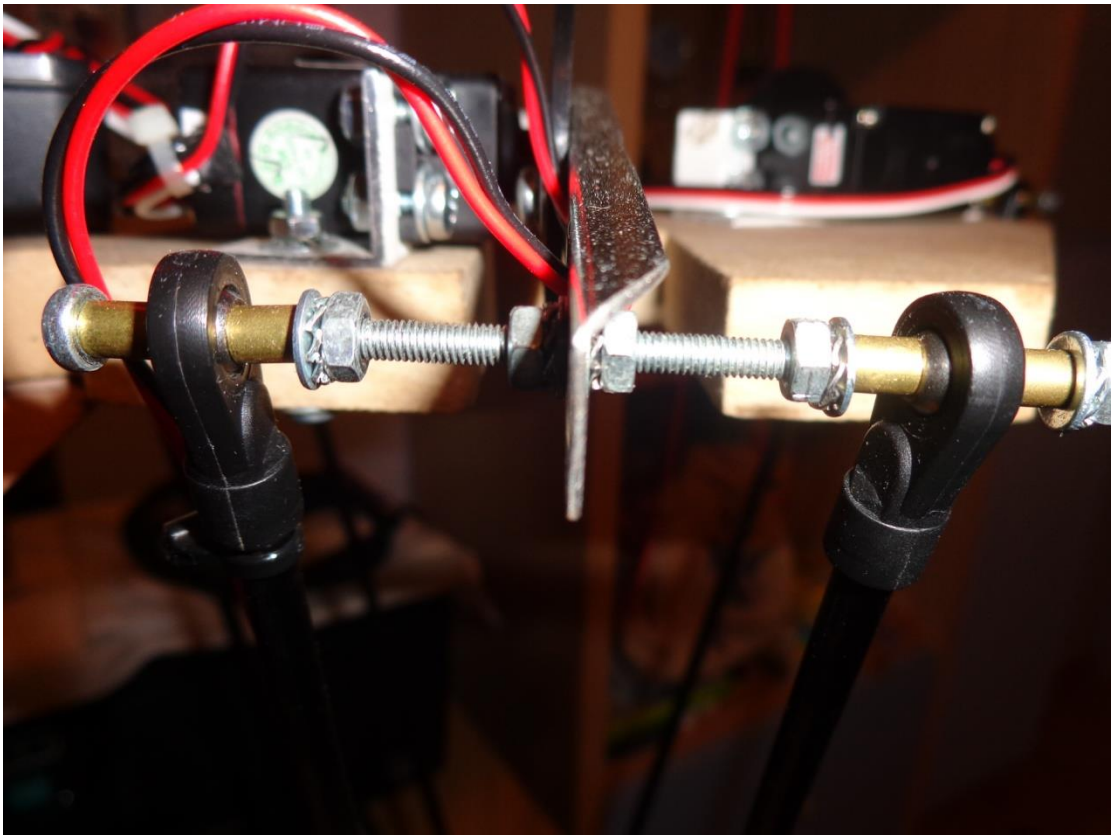
3.2. Υλοποίηση r_f - r_e - end effector

Για την υλοποίηση της διάστασης r_f χρησιμοποιήθηκαν γωνιακοί σύνδεσμοι 6mm x 12mm με 10 τρύπες, όπως φαίνεται στην Εικόνα 3.4. Το πλεονέκτημά τους είναι ότι εύκολα μπορεί να αλλάξει το μήκος r_f, συνδέοντας το r_e σε διαφορετική τρύπα. Για την υλοποίηση του r_e χρησιμοποιήθηκε σωλήνας ανθρακονημάτων εξωτερικής διαμέτρου Φ4 και εσωτερικής Φ3. Αυτό το υλικό το χαρακτηρίζει μεγάλη αντοχή, μεγάλη ακαμψία και ελάχιστο βάρος, συνδυασμός στοιχείων απαραιτήτων για την εφαρμογή μας. Κάθε ζεύγος σωλήνων τερματίζεται σε δύο ball joints. Επιλέχθηκε αυτός ο τύπος άρθρωσης, γιατί κατά τη λειτουργία της παρουσιάζει ελάχιστη αντίσταση και πολύ μικρές ανοχές (τζόγο). Βέβαια, η συγκεκριμένη άρθρωση δεν έχει κανέναν περιορισμό κατά τον έναν άξονα λειτουργίας της, ενώ κατά τον άλλο άξονα έχει ένα εύρος κίνησης $\pm 10^o$ (Εικόνα 3.5). Το r_e συνδέεται με το r_f με μία βίδα M3x80mm, περικόχλια, ροδέλες, γκρόβερ και αποστατικά σωληνάκια. Ο ακριβής τρόπος σύνδεσης φαίνεται στη Εικόνα 3.5. Αξίζει να σημειωθεί ότι ο ρόλος των γκρόβερ είναι να μην επιτρέπουν τη χαλάρωση των περικοχλίων λόγω των κραδασμών της κατασκευής. Στο συγκεκριμένο σημείο απαιτείται ρύθμιση, ώστε τα σημεία σύνδεσης του ζεύγους των σωλήνων να ισαπέχουν από το γωνιακό σύνδεσμο r_f.

Η άλλη άκρη του r_e συνδέεται με το end effector.



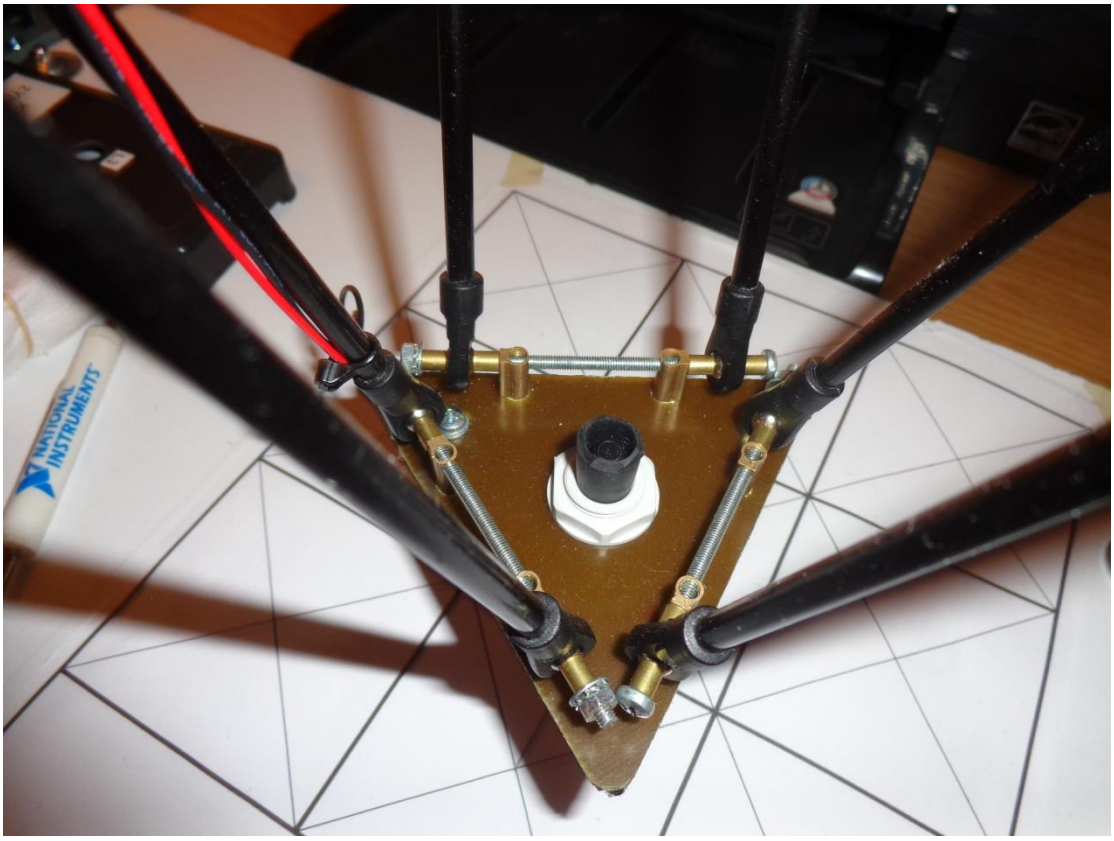
Εικόνα 3.4. Γωνιακοί σύνδεσμοι για την υλοποίηση της διάστασης r_f.



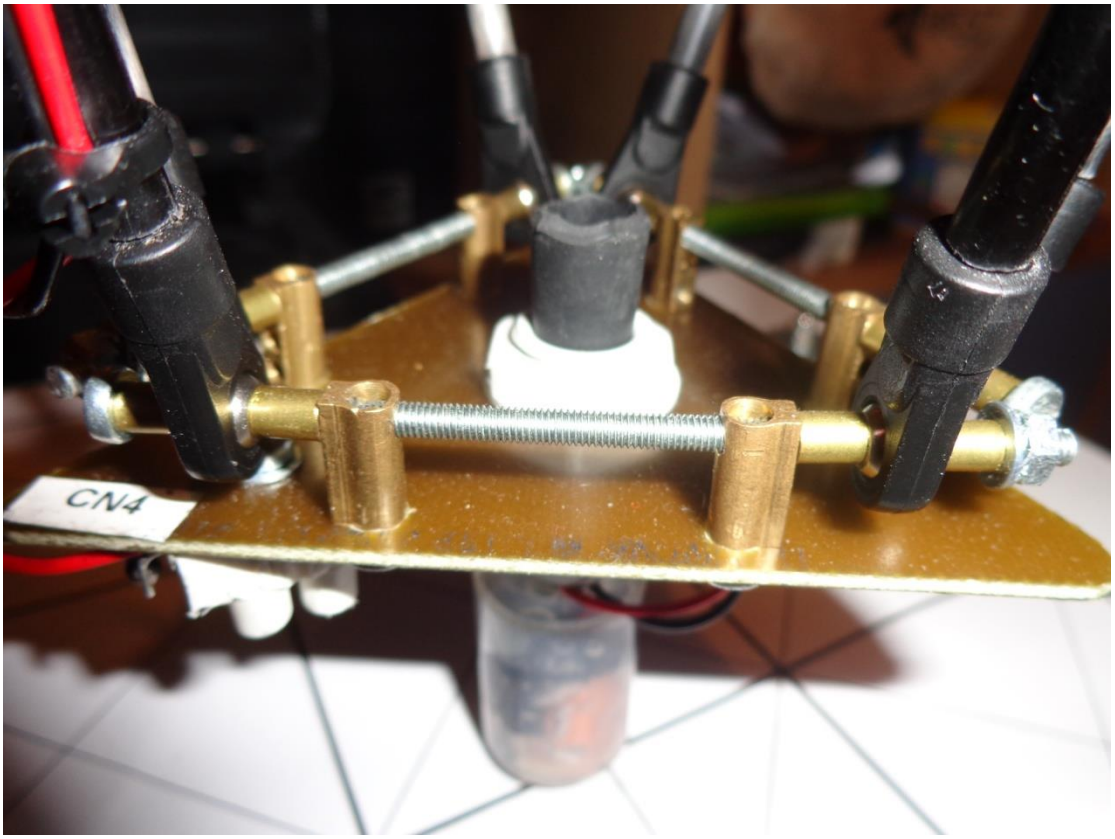
Εικόνα 3.5. Άρθρωση r_f - r_e και ακριβής τρόπος σύνδεσής τους.

Η υλοποίηση του end effector ξεκίνησε κατασκευάζοντας ένα ισόπλευρο τρίγωνο εργασίας από βακελίτη πλευράς 8.5cm. Στη συνέχεια σχεδιάστηκε στο εσωτερικό του ένα ισόπλευρο τρίγωνο πλευράς 8cm. Έπειτα, εκατέρωθεν του μέσου κάθε πλευράς και σε απόσταση 1.5cm από αυτό ανοίχθηκαν τρύπες Φ3mm (συνολικά 6 τρύπες).

Σε κάθε τρύπα βιδώθηκε ένα αποστατικό και στη συνέχεια χρησιμοποιώντας τρεις βίδες M3x80mm, περικόχλια, ροδέλες, γκρόβερ και αποστατικά σωληνάκια συνδέθηκαν οι άκρες των r_e (Εικόνα 3.7). Οι άξονες των βιδών αυτών συμπίπτουν με τις πλευρές του τριγώνου του end effector, e. Στο συγκεκριμένο σημείο απαιτείται ρύθμιση, ώστε τα σημεία σύνδεσης των ζευγών των σωληνών να ισαπέχουν από το κέντρο της αντίστοιχης πλευράς e.
Οι συνδέσεις r_f - r_e και r_e-end effector θέλουν ιδιαίτερη προσοχή, ώστε οι σωλήνες κάθε ζεύγους να είναι παράλληλες.



Εικόνα 3.6. End effector.



Εικόνα 3.7. Σύνδεση end effector - r_e.

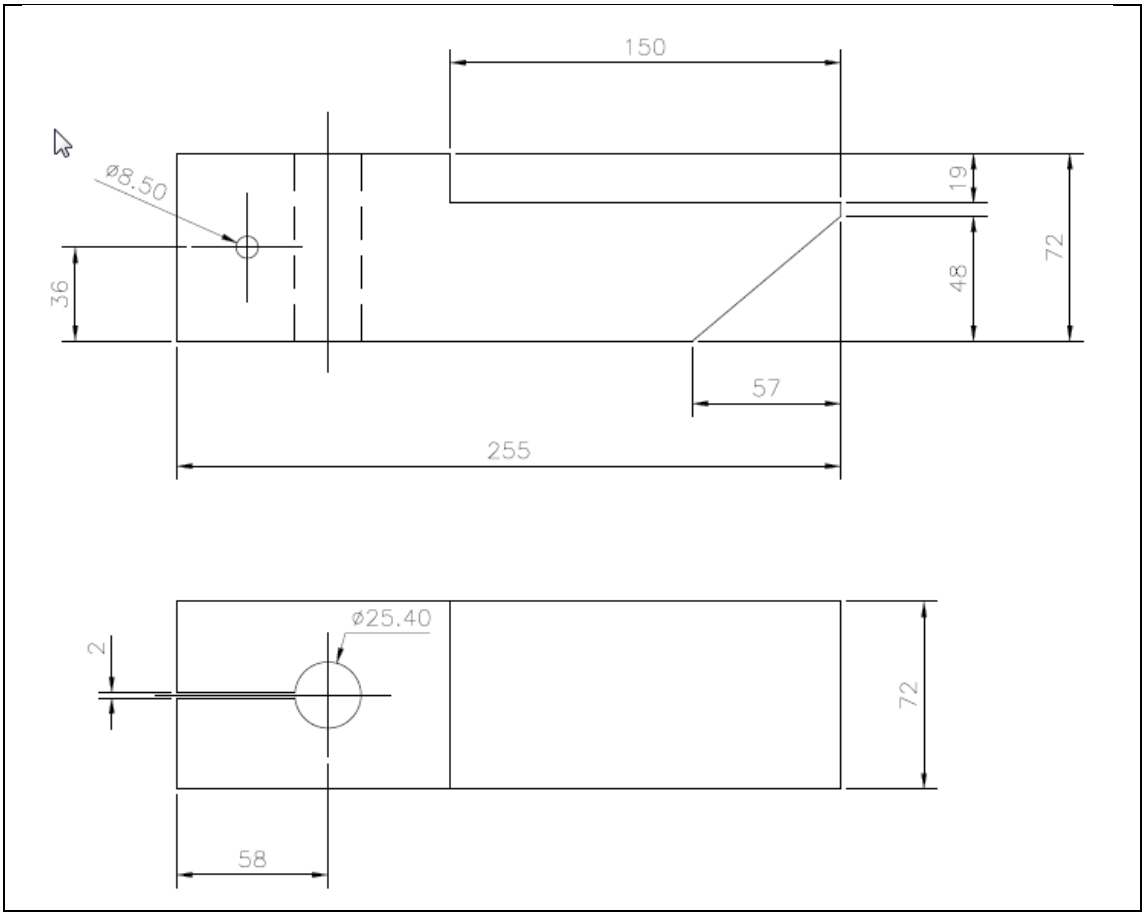
Στο κέντρο του τριγωνικού βακελίτη ανοίχτηκε μία τρύπα Φ 15mm και στη θέση αυτή βιδώθηκε ένας πλαστικός στυπιοθλήπτης μεγέθους PG9 (Εικόνα 3.8). Η επιλογή του στυπιοθλήπτη έγινε επειδή έχει τη δυνατότητα να στηριχθεί σωστά και εύκολα στην πλακέτα και επειδή μπορούμε να συσφίξουμε στο κέντρο του οποιοδήποτε εργαλείο με άξονα Φ περίπου 6mm.

Εικόνα 3.8. Στυπιοθλήπτης PG9.

3.3. Φορέας

Η μέχρι τώρα κατασκευή στηρίζεται σε έναν ξύλινο φορέα (Εικόνα 3.9, Σχέδιο 3.2). Η χρήση του φορέα είναι να προσαρμόζει τη βάση των σέρβο στο επόμενο κομμάτι της κατασκευής, στον οδηγό ύψους. Ο φορέας κατασκευάστηκε από σκληρό ξύλο, για να μπορέσει η τρύπα ολίσθησης να ανοιχτεί στις σωστές διαστάσεις και να παραμένει όσο το δυνατόν αναλλοίωτη μετά από πολλές χρήσεις.

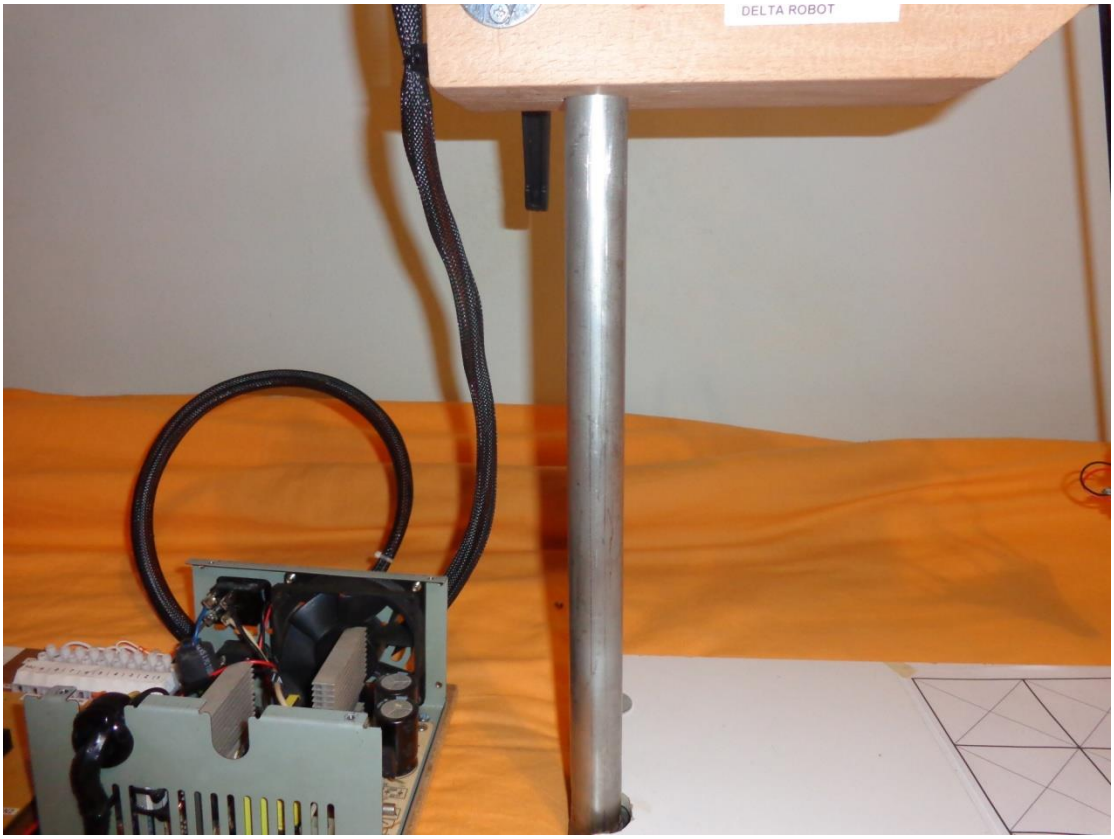
Εικόνα 3.9. Ξύλινος φορέας.



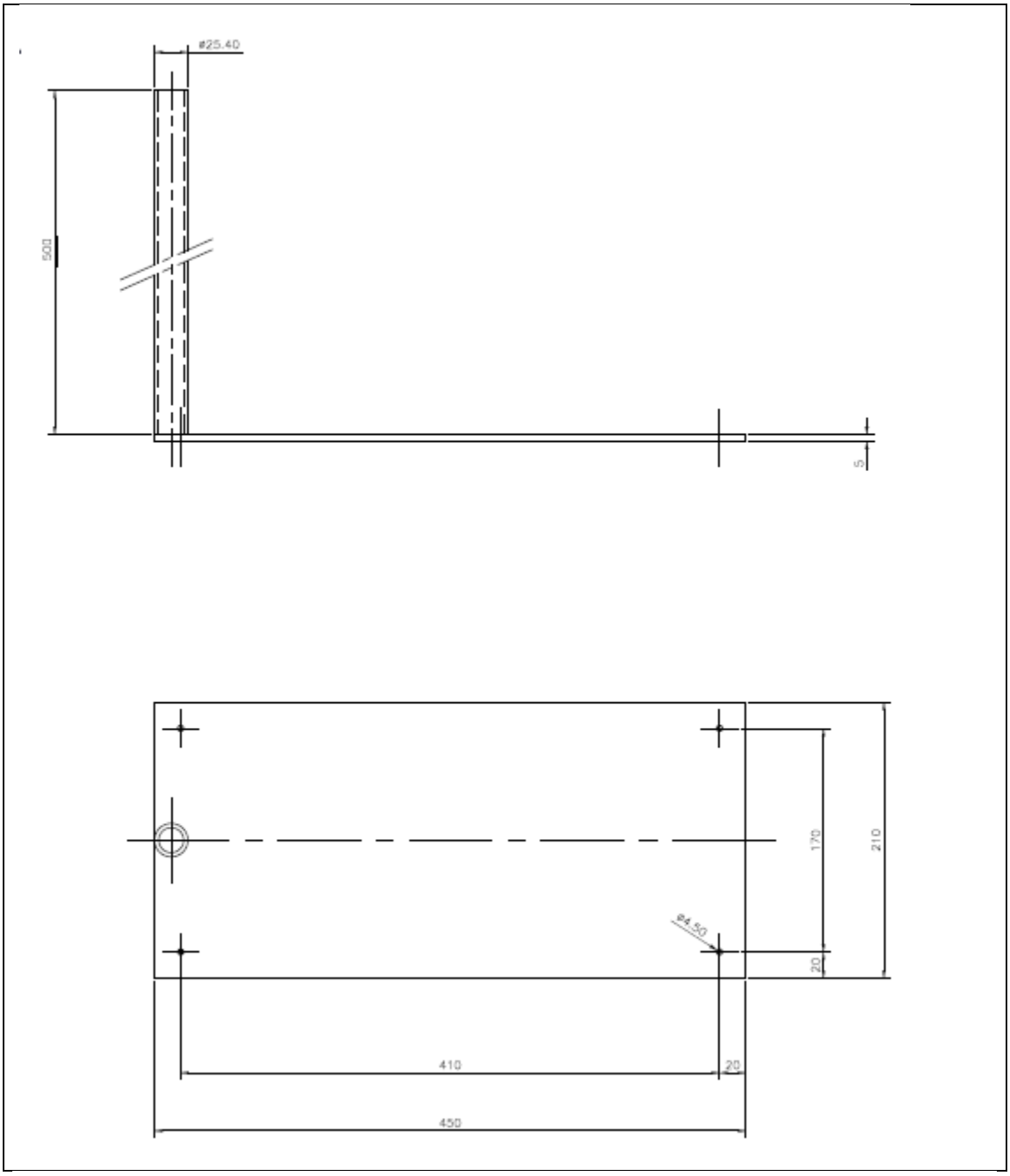
Σχέδιο 3.2. Μηχανολογικό σχέδιο (πλάγια όψη και κάτοψη) φορέα.

3.4. Οδηγός ύψους - καστάνια ακινητοποίησης φορέα

Ο οδηγός ύψους (Εικόνα 3.10, Σχέδιο 3.3) μας δίνει τη δυνατότητα να ολισθαίνουμε καθ' ύψος του το φορέα και έτσι να ρυθμίζουμε το ύψος εργασίας του ρομπότ, όσον αφορά τη συντεταγμένη z, δεδομένου ότι το τραπέζι εργασίας (βλ. παρακάτω) βρίσκεται σε σταθερό ύψος. Αυτό επιτυγχάνεται χαλαρώνοντας την καστάνια ακινητοποίησης φορέα, που φαίνεται στην εικόνα 3.11, ολισθαίνοντας το φορέα στο επιθυμητό ύψος και σφίγγοντας πάλι την καστάνια. Να σημειωθεί ότι κατά τη χρήση της καστάνιας δεν απαιτείται συγκράτηση της κεφαλής της βίδας σύσφιξης, γιατί όπως φαίνεται στην εικόνα 3.12, είναι μονίμως ακινητοποιημένη πάνω στο φορέα.
Το υλικό που επιλέχθηκε για την κατασκευή του οδηγού ύψους είναι ανοξείδωτος χάλυβας που δεν απαιτεί βάψιμο και η λεία επιφάνειά του επιτρέπει την εύκολη ολίσθηση του φορέα.



Εικόνα 3.10. Οδηγός ύψους.



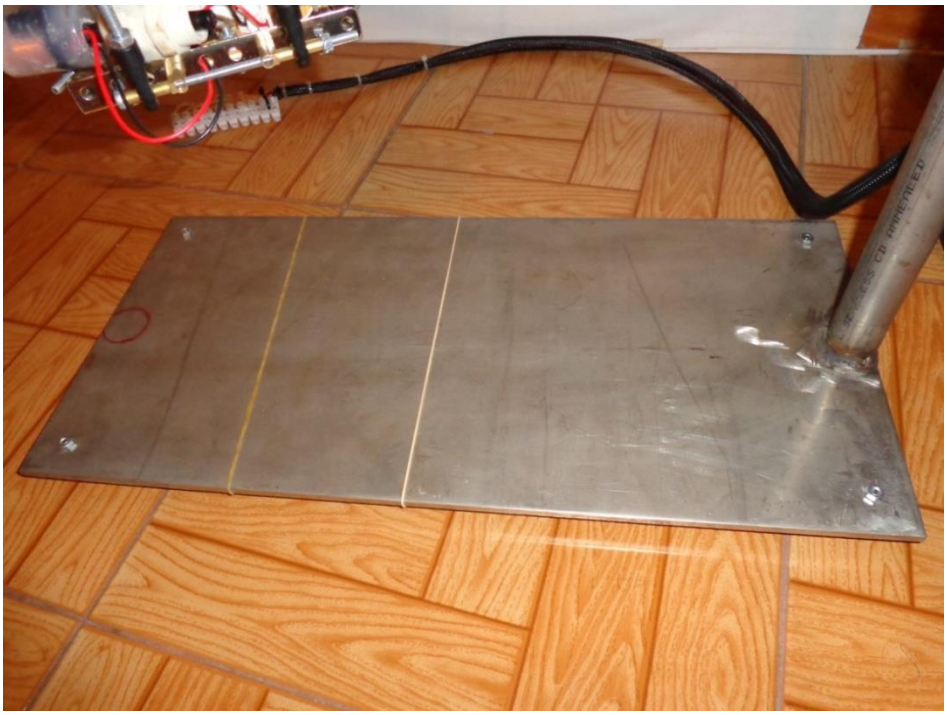
Σχέδιο 3.3. Μηχανολογικό σχέδιο (πλάγια όψη και κάτοψη) οδηγού ύψους και μεταλλικής βάσης.

3.11. Καστάνια ακινητοποίησης φορέα.

Εικόνα 3.12. Σύνδεση καστάνιας πάνω στο φορέα.

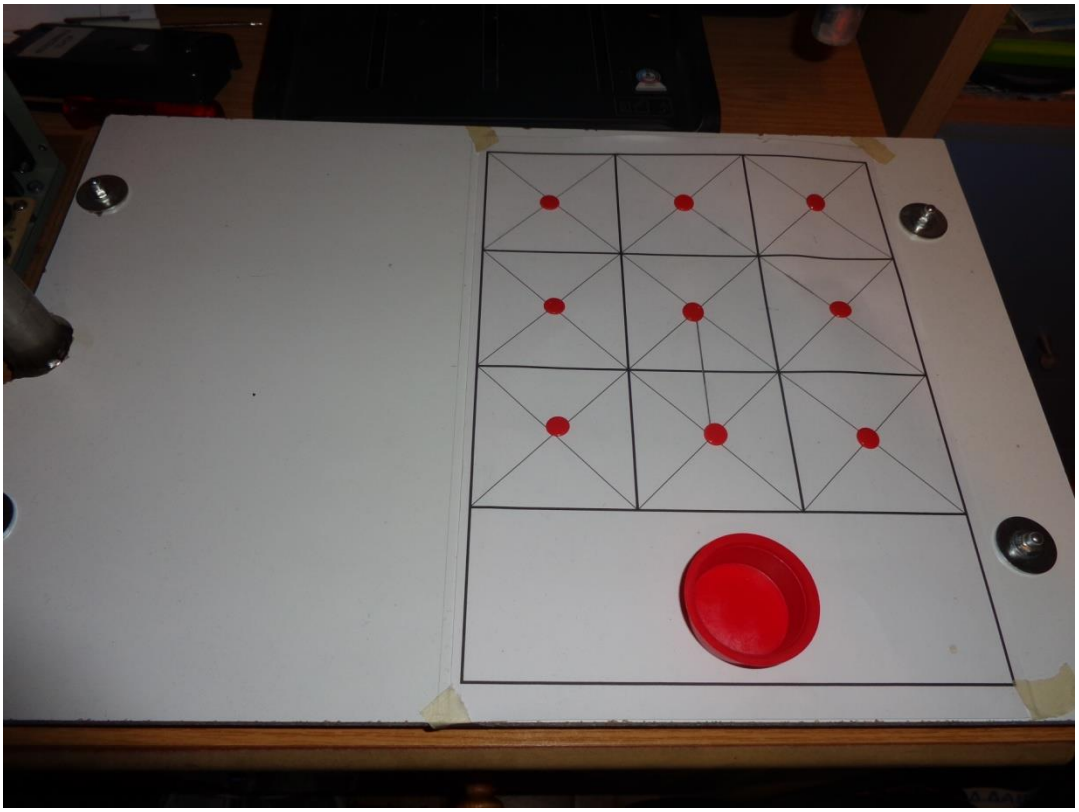
3.5. Μεταλλική βάση - Τραπέζι εργασίας

Η μεταλλική βάση του ρομπότ (Εικόνα 3.13, Σχέδιο 3.3), στην οποία έχει συγκολληθεί ο οδηγός ύψους, είναι η επαφή του ρομπότ μας με τον πάγκο εργασίας. Είναι και αυτή από ανοξείδωτο χάλυβα, άρα αρκετά βαριά, ώστε το ρομπότ να έχει σταθερότητα και να μην υπάρχει ο κίνδυνος να κουνηθεί σε κάποια απότομη κίνησή του.



Εικόνα 3.13. Μεταλλική βάση ρομπότ.

Πάνω σε αυτή τη μεταλλική βάση έχει τοποθετηθεί το τραπέζι εργασίας (Εικόνα 14).



Εικόνα 3.14. Τραπέζι εργασίας

Όλη η κατασκευή στηρίζεται σε 4 ελαστικά στηρίγματα για καλή επαφή με τον πάγκο εργασίας και για απόσβεση κραδασμών (Εικόνα 3.15).

Εικόνα 3.15. Ελαστικά στηρίγματα μεταλλικής βάσης.

Είναι βέβαιο ότι ορισμένα σημεία της κατασκευής θα μπορούσαν να είναι πιο απλά. Επιλέχθηκε όμως να αποτελείται από περισσότερα του ενός συνεργαζόμενων κομματιών, από τα οποία κάποια να είναι μεταβαλλόμενα/ρυθμιζόμενα, ούτως ώστε να δίνεται η δυνατότητα τροποποίησης αυτής της κατασκευής εύκολα και χωρίς να απαιτείται σε μεγάλο βαθμό η αντικατάστασή τους.

4. Ηλεκτρονικό μέρος

Το ηλεκτρονικό τμήμα της κατασκευής αποτελείται από δύο κύρια τμήματα:

1. το κέντρο λειτουργίας, το οποίο αποτελείται από:

    - τον κονέκτορα διασύνδεσης CN1α
    - το τροφοδοτικό
    - την πλακέτα εφαρμογών
    - το arduino mega

2. το τμήμα που βρίσκεται εγκατεστημένο πάνω στη βάση στήριξης των σέρβο, που αποτελείται από:

- τα τρία servo
- το κουτί συνδέσεων
- τον ηλεκτρομαγνήτη
- το καλώδιο διασύνδεσης

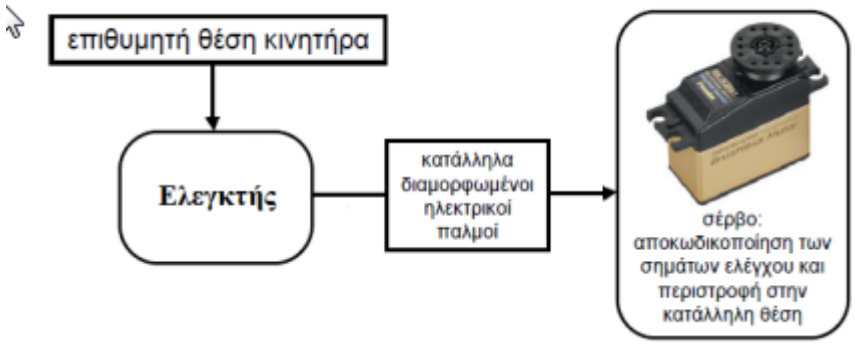
4.1. Servo

Πολύ σημαντικό κομμάτι της κατασκευής μας είναι οι σερβοκινητήρες του ρομπότ.

Το σύστημα των σέρβο αποτελείται από τον κινητήρα, το κιβώτιο ταχυτήτων, τη συσκευή ανάδρασης και το κύκλωμα ελέγχου και οδήγησης. Τα σέρβο περιλαμβάνουν 3 αγωγούς: τροφοδοσίας, γείωσης και ελέγχου. Το σήμα ελέγχου είναι ένας παλμός συχνότητας 50 Hz. Το πλάτος του παλμού καθορίζει τη θέση στην έξοδο του σερβοκινητήρα. Ο έλεγχός τους μπορεί να υλοποιηθεί αρκετά εύκολα μέσω ενός ψηφιακού ελεγκτή (π.χ. arduino). Τα βασικά μέρη από τα οποία αποτελείται ένα σέρβο είναι: ένας ηλεκτροκινητήρας συνεχούς ρεύματος, ένα ηλεκτρονικό κύκλωμα που ελέγχει τη θέση του τελικού άξονα κίνησης και ένα κιβώτιο υποβιβασμού της σχέσης μετάδοσης του κινητήρα. Ο τελικός άξονας κίνησης δεν εκτελεί πλήρεις περιστροφές, αλλά περιστρέφεται μεταξύ δύο ακραίων θέσεων.

Για τον έλεγχο του σέρβο απαιτείται εξειδικευμένος ελεγκτής και χρησιμοποιείται η μέθοδος ελέγχου ανοιχτού βρόχου. Πιο συγκεκριμένα, ο ελεγκτής διαμορφώνει και μεταδίδει στο σέρβο ηλεκτρικούς παλμούς, ανάλογα με τη θέση στην οποία πρέπει να περιστραφεί ο άξονας του σέρβο. Οι ηλεκτρικοί παλμοί λαμβάνονται και αποκωδικοποιούνται από το σέρβο, με τη βοήθεια του κυκλώματος ελέγχου που περιλαμβάνεται σε αυτό. Στη συνέχεια, μετά την αποκωδικοποίηση, το κύκλωμα ελέγχου του σέρβο οδηγεί τον κινητήρα του στην κατάλληλη θέση.

Τα παραπάνω φαίνονται σχηματικά στην εικόνα 4.1:



Εικόνα 4.1. Λειτουργία servo.

Στη συγκεκριμένη κατασκευή χρησιμοποιήθηκαν τρία SPRINGRC SM-S4315M. Κάποια χαρακτηριστικά τους φαίνονται στις εικόνες 4.2 και 4.3:

Motor Properties

Motor Type	Limited Rotation Servo
Range of Rotation	180°
Maximum Speed at Rated Voltage	285°/s
Rated Torque	13 kg·cm

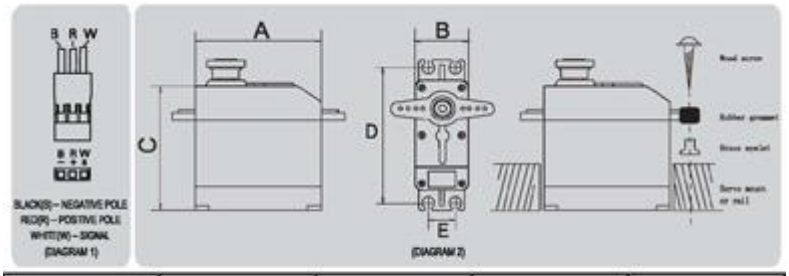
Physical Properties

Gear Train Material	Metal
Bearing Type	Double Ball Bearing
Motor Length	40 mm
Motor Width	41.3 mm
Motor Depth	20.7 mm
Wire Length	280 mm
Weight	60 g

Electrical Properties

Rated Voltage	5 V DC
Rated Current (on 1061 controller)	350 mA
Rated Current (on 1066 controller)	370 mA
Stall Current (on 1061 controller)	1.8 A
Stall Current (on 1066 controller)	450 mA

Εικόνα 4.2. Ιδιότητες των servo SPRINGRC SM-S4315M.

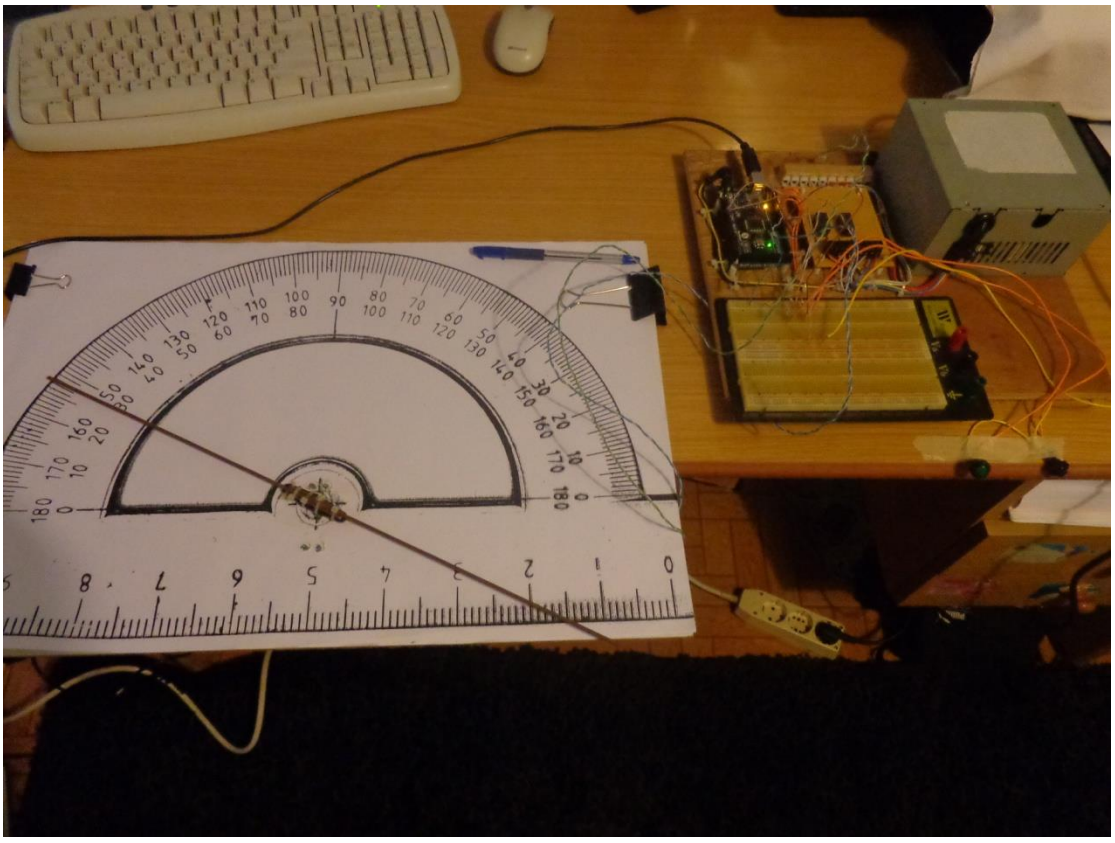


A	B	C	D	E
41.3mm	20.7mm	40.0mm	50.3mm	9.9mm

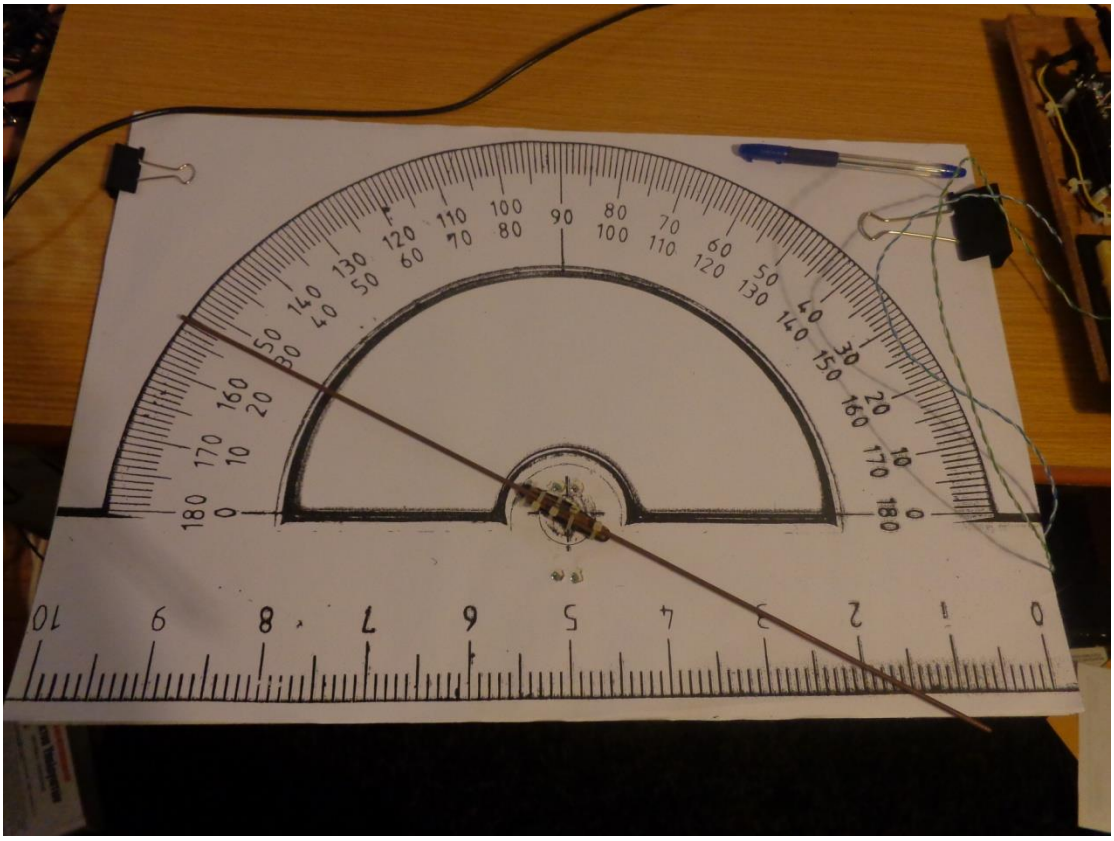
Εικόνα 4.3. Διαστάσεις των servo SPRINGRC SM-S4315M.

Γενικά δεν έχουμε μεγάλες απαιτήσεις ισχύος από τα σέρβο, αφού τα αντικείμενα που θα χρειαστεί να σηκώσουν είναι αρκετά ελαφριά. Σχετικά με την ακρίβειά τους, δεν είναι τα καλύτερα δυνατά, γι αυτό αφιερώσαμε πολύ χρόνο στις μετρήσεις πάνω σε αυτά, ώστε να πάρουμε το μέγιστο δυνατό αποτέλεσμα στον τομέα της ακρίβειας.

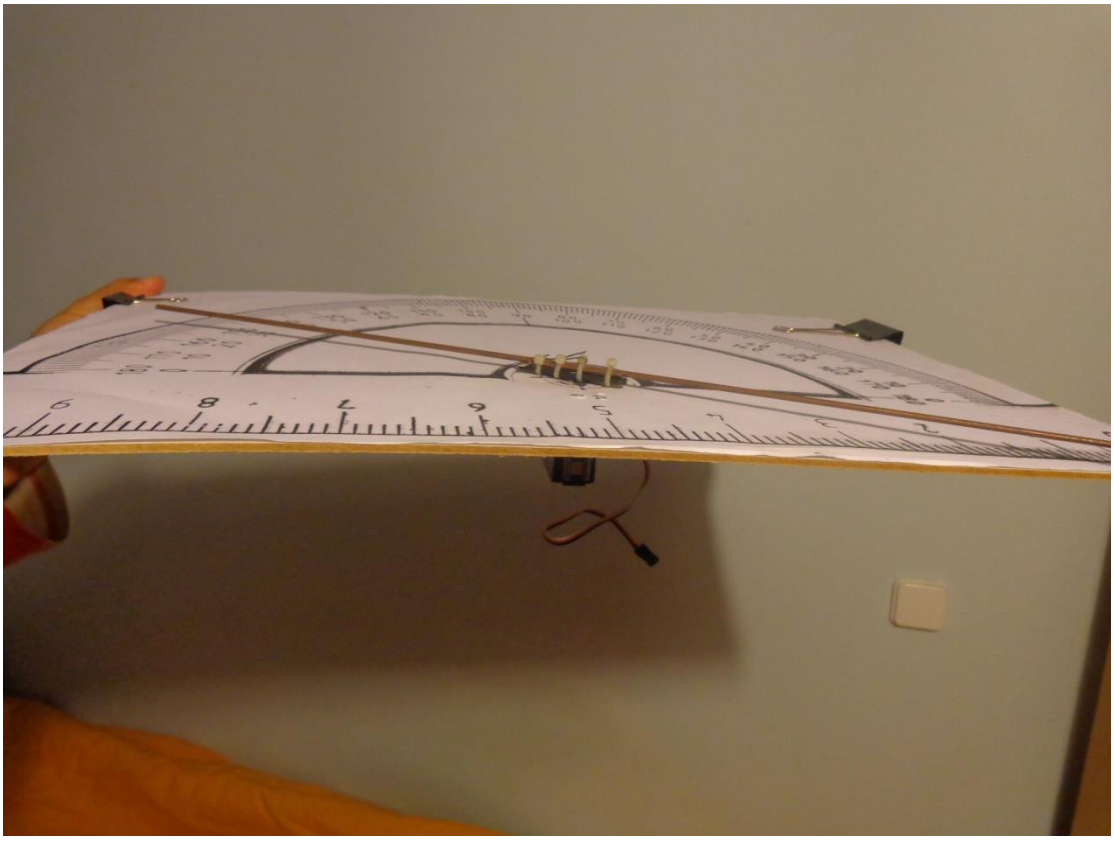
Για τις μετρήσεις στα servo χρησιμοποιήθηκε η διάταξη των εικόνων 4.4, 4.5, 4.6, 4.7. Συγκεκριμένα, χρησιμοποιήθηκαν το arduino με 2 button, ένα μεγενθυμένο μοιρογνωμόνιο και το προς έλεγχο σέρβο.



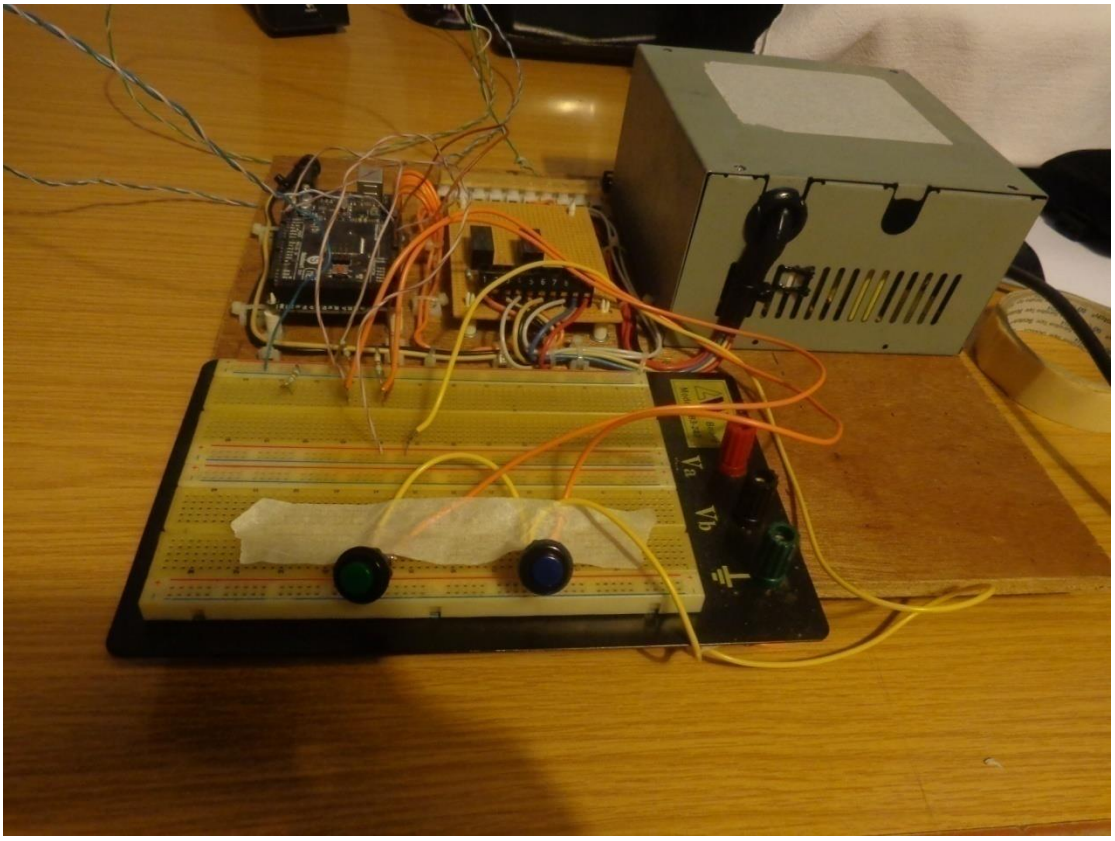
Εικόνα 4.4. Συνολική διάταξη μέτρησης των servo.



Εικόνα 4.5. Διάταξη με μεγεθυμένο μοιρογνωμόνιο για μετρήσεις των servo.



Εικόνα 4.6. Τρόπος σύνδεσης του servo πάνω στο μεγεθυμένο μοιρογνωμόνιο.



Εικόνα 4.7. Μπουτόν και κύκλωμα λειτουργίας τους.

Μέσω των 2 button, αυξάναμε ή μειώναμε, ανάλογα με το ποια μέτρηση θέλαμε να πάρουμε, το πλάτος παλμού (microseconds) στην έξοδο του arduino, ώστε να δημιουργήσουμε τον πίνακα 4.1. Ξεκινώντας από τις 0^{o} και με βήμα 1^{o} σημειώθηκαν τα μικροseconds που αντιστοιχούν σε κάθε μοίρα ($1^{η}$ μέτρηση). Στη συνέχεια, λόγω έλλειψης επαναλειψημότητας των servo, ακολουθήσαμε την ίδια διαδικασία, αλλά με αντίθετη φορά ($2^{η}$ μέτρηση). Τέλος βγάλαμε το μέσο όρο των δύο αυτών μετρήσεων.

Ακολουθεί ο πίνακας των μετρήσεων για τους 3 σερβοκινητήρες.

	Σερβο 1			**Σερβο 2**			**Σερβο 3**		
Μοίρες	**$1^{η}$**	**$2^{η}$**	**Μ. Ό.**	**$1^{η}$**	**$2^{η}$**	**Μ. Ό.**	**$1^{η}$**	**$2^{η}$**	**Μ. Ό.**
0	643	639	641	590	584	587	590	584	587
1	653	648	651	593	594	594	598	593	596
2	660	658	659	603	597	600	604	601	603
3	668	667	668	608	608	608	610	608	609
4	678	673	676	610	613	612	617	615	616
5	681	679	680	623	617	620	629	623	626
6	693	687	690	627	626	627	635	629	632
7	701	699	700	631	634	633	642	637	640
8	707	705	706	642	640	641	650	645	648
9	716	712	714	654	648	651	659	654	657
10	723	721	722	660	657	659	668	662	665
11	731	729	730	663	661	662	674	668	671
12	737	736	737	675	669	672	681	676	679
13	748	743	746	681	679	680	690	685	688
14	755	752	754	690	686	688	699	693	696
15	765	761	763	697	693	695	704	698	701
16	772	771	772	701	705	703	710	705	708
17	781	777	779	712	713	713	718	713	716

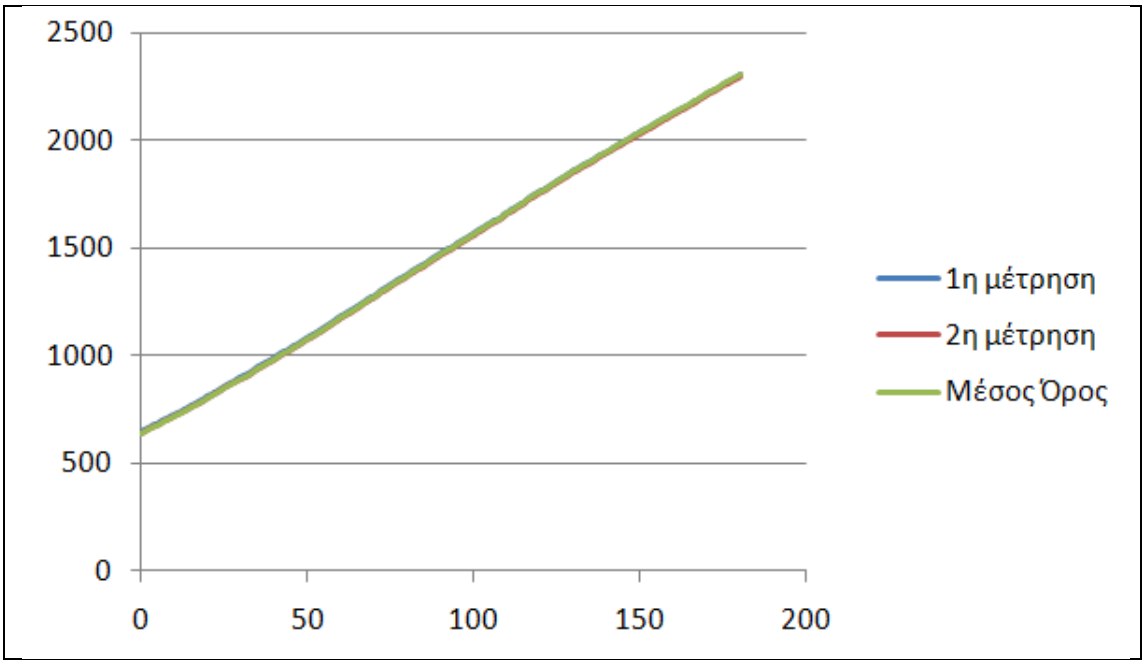
18	790	785	788	721	720	721	724	719	722
19	798	794	796	731	729	730	731	727	729
20	811	805	808	740	735	738	741	737	739
21	816	814	815	745	743	744	751	746	749
22	826	824	825	753	751	752	758	752	755
23	833	832	833	764	758	761	766	761	764
24	844	841	843	770	766	768	774	768	771
25	854	853	854	779	775	777	782	777	780
26	863	861	862	784	781	783	790	784	787
27	872	870	871	793	789	791	796	791	794
28	878	877	878	802	802	802	808	803	806
29	890	886	888	811	806	809	817	811	814
30	899	893	896	820	818	819	822	817	820
31	906	900	903	828	826	827	830	824	827
32	913	907	910	837	833	835	837	831	834
33	921	917	919	850	844	847	845	840	843
34	933	930	932	856	855	856	854	849	852
35	946	940	943	864	858	861	864	858	861
36	954	948	951	872	873	873	872	867	870
37	963	957	960	884	881	883	880	876	878
38	971	967	969	891	887	889	890	885	888
39	978	974	976	898	894	896	898	893	896
40	989	983	986	908	906	907	907	901	904
41	999	993	996	917	915	916	914	908	911
42	1010	1004	1007	926	924	925	925	920	923
43	1019	1013	1016	935	935	935	934	928	931
44	1023	1020	1022	942	940	941	944	938	941
45	1035	1029	1032	956	950	953	955	949	952
46	1045	1040	1043	962	959	961	962	956	959
47	1051	1048	1050	972	967	970	970	965	968
48	1062	1058	1060	979	973	976	981	978	980
49	1073	1069	1071	984	984	984	990	984	987
50	1082	1079	1081	995	995	995	998	992	995
51	1092	1086	1089	1005	1001	1003	1006	1000	1003
52	1099	1095	1097	1015	1009	1012	1016	1011	1014
53	1109	1106	1108	1024	1023	1024	1025	1020	1023
54	1119	1113	1116	1034	1032	1033	1036	1031	1034
55	1129	1124	1127	1044	1042	1043	1045	1039	1042
56	1139	1135	1137	1052	1048	1050	1054	1050	1052
57	1149	1145	1147	1060	1057	1059	1063	1057	1060
58	1159	1156	1158	1069	1068	1069	1073	1068	1071
59	1172	1166	1169	1076	1078	1077	1082	1077	1080
60	1183	1177	1180	1087	1084	1086	1091	1085	1088
61	1190	1187	1189	1102	1098	1100	1099	1093	1096
62	1201	1195	1198	1107	1103	1105	1111	1105	1108
63	1209	1203	1206	1119	1115	1117	1120	1115	1118
64	1218	1214	1216	1127	1128	1128	1131	1125	1128
65	1228	1223	1226	1138	1133	1136	1140	1136	1138
66	1238	1234	1236	1148	1143	1146	1149	1143	1146
67	1248	1247	1248	1155	1156	1156	1160	1156	1158
68	1260	1255	1258	1167	1163	1165	1170	1164	1167
69	1270	1265	1268	1181	1175	1178	1178	1172	1175
70	1275	1272	1274	1188	1182	1185	1188	1182	1185
71	1287	1283	1285	1195	1193	1194	1199	1193	1196
72	1301	1298	1300	1209	1203	1206	1208	1203	1206
73	1310	1304	1307	1216	1210	1213	1217	1212	1215
74	1319	1314	1317	1228	1222	1225	1226	1222	1224
75	1330	1325	1328	1233	1230	1232	1237	1233	1235
76	1339	1336	1338	1240	1242	1241	1246	1240	1243
77	1351	1345	1348	1251	1250	1251	1257	1251	1254
78	1359	1354	1357	1261	1261	1261	1265	1260	1263
79	1367	1361	1364	1266	1269	1268	1274	1269	1272
80	1375	1373	1374	1275	1274	1275	1285	1282	1284
81	1388	1384	1386	1289	1286	1288	1296	1291	1294
82	1398	1392	1395	1300	1295	1298	1305	1300	1303
83	1408	1404	1406	1308	1306	1307	1317	1313	1315

84	1416	1410	1413	1323	1318	1321	1328	1322	1325
85	1424	1418	1421	1332	1328	1330	1336	1330	1333
86	1433	1431	1432	1336	1334	1335	1348	1343	1346
87	1446	1440	1443	1350	1345	1348	1358	1353	1356
88	1457	1451	1454	1359	1353	1356	1367	1362	1365
89	1466	1460	1463	1370	1367	1369	1379	1373	1376
90	1475	1472	1474	1374	1371	1373	1388	1384	1386
91	1486	1480	1483	1383	1385	1384	1396	1393	1395
92	1495	1489	1492	1397	1395	1396	1405	1399	1402
93	1500	1496	1498	1411	1405	1408	1415	1411	1413
94	1508	1507	1508	1419	1416	1418	1426	1420	1423
95	1522	1517	1520	1428	1424	1426	1435	1429	1432
96	1531	1527	1529	1439	1434	1437	1443	1437	1440
97	1540	1536	1538	1451	1446	1449	1453	1448	1451
98	1548	1545	1547	1462	1458	1460	1461	1458	1460
99	1558	1556	1557	1471	1467	1469	1472	1466	1469
100	1569	1563	1566	1477	1475	1476	1480	1474	1477
101	1580	1576	1578	1489	1487	1488	1488	1484	1486
102	1589	1583	1586	1499	1496	1498	1498	1494	1496
103	1598	1593	1596	1507	1506	1507	1510	1505	1508
104	1608	1605	1607	1521	1519	1520	1521	1515	1518
105	1616	1614	1615	1530	1531	1531	1527	1521	1524
106	1628	1622	1625	1540	1539	1540	1537	1531	1534
107	1632	1630	1631	1548	1550	1549	1547	1541	1544
108	1639	1639	1639	1557	1562	1560	1555	1552	1554
109	1657	1652	1655	1575	1569	1572	1566	1560	1563
110	1663	1665	1664	1586	1580	1583	1575	1570	1573
111	1677	1672	1675	1594	1592	1593	1587	1581	1584
112	1685	1680	1683	1601	1600	1601	1598	1593	1596
113	1694	1693	1694	1611	1616	1614	1606	1600	1603
114	1705	1701	1703	1622	1624	1623	1613	1607	1610
115	1711	1710	1711	1634	1639	1637	1623	1618	1621
116	1726	1726	1726	1644	1645	1645	1632	1628	1630
117	1739	1735	1737	1657	1655	1656	1638	1636	1637
118	1748	1743	1746	1674	1669	1672	1654	1650	1652
119	1757	1754	1756	1684	1680	1682	1663	1659	1661
120	1766	1763	1765	1693	1688	1691	1672	1668	1670
121	1771	1769	1770	1702	1698	1700	1681	1678	1680
122	1781	1779	1780	1711	1709	1710	1690	1685	1688
123	1791	1791	1791	1717	1717	1717	1699	1697	1698
124	1802	1800	1801	1730	1728	1729	1708	1703	1706
125	1813	1810	1812	1740	1745	1743	1720	1718	1719
126	1822	1819	1821	1756	1751	1754	1730	1726	1728
127	1831	1829	1830	1765	1763	1764	1738	1738	1738
128	1841	1837	1839	1776	1776	1776	1746	1746	1746
129	1851	1850	1851	1790	1788	1789	1759	1755	1757
130	1863	1859	1861	1800	1799	1800	1771	1766	1769
131	1868	1865	1867	1813	1811	1812	1781	1776	1779
132	1878	1876	1877	1820	1816	1818	1790	1786	1788
133	1889	1885	1887	1829	1829	1829	1795	1791	1793
134	1896	1892	1894	1839	1843	1841	1804	1799	1802
135	1903	1900	1902	1853	1849	1851	1815	1811	1813
136	1912	1910	1911	1863	1860	1862	1823	1819	1821
137	1925	1922	1924	1869	1869	1869	1832	1827	1830
138	1933	1932	1933	1883	1880	1882	1844	1841	1843
139	1940	1940	1940	1889	1887	1888	1854	1849	1852
140	1947	1949	1948	1898	1898	1898	1864	1860	1862
141	1956	1956	1956	1912	1909	1911	1875	1872	1874
142	1969	1967	1968	1921	1919	1920	1884	1880	1882
143	1977	1976	1977	1932	1930	1931	1895	1891	1893
144	1988	1982	1985	1943	1942	1943	1904	1899	1902
145	1995	1993	1994	1954	1953	1954	1913	1910	1912
146	2004	2003	2004	1962	1959	1961	1922	1917	1920
147	2012	2010	2011	1970	1969	1970	1933	1928	1931
148	2025	2021	2023	1979	1973	1976	1940	1935	1938
149	2031	2029	2030	1988	1987	1988	1948	1944	1946

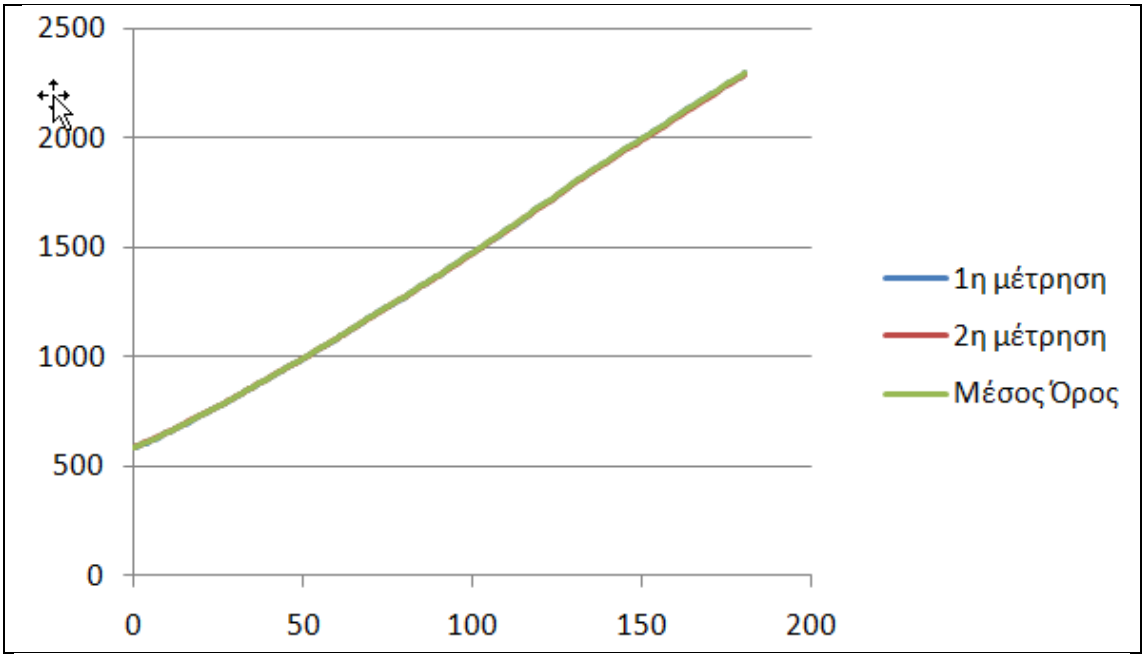
150	2042	2037	2040	2000	1999	2000	1958	1955	1957
151	2051	2046	2049	2010	2005	2008	1966	1961	1964
152	2057	2056	2057	2016	2016	2016	1974	1970	1972
153	2067	2062	2065	2031	2026	2029	1983	1979	1981
154	2077	2072	2075	2040	2035	2038	1993	1988	1991
155	2086	2082	2084	2047	2046	2047	2000	1995	1998
156	2095	2091	2093	2058	2053	2056	2012	2008	2010
157	2104	2101	2103	2064	2064	2064	2019	2015	2017
158	2110	2107	2109	2081	2078	2080	2028	2026	2027
159	2120	2118	2119	2090	2089	2090	2038	2033	2036
160	2130	2127	2129	2100	2098	2099	2044	2039	2042
161	2138	2135	2137	2111	2106	2109	2055	2050	2053
162	2146	2145	2146	2122	2117	2120	2061	2058	2060
163	2154	2151	2153	2134	2129	2132	2072	2067	2070
164	2161	2161	2161	2142	2136	2139	2081	2076	2079
165	2170	2170	2170	2149	2145	2147	2088	2083	2086
166	2183	2177	2180	2160	2157	2159	2097	2092	2095
167	2188	2188	2188	2172	2167	2170	2104	2100	2102
168	2199	2199	2199	2179	2177	2178	2113	2108	2111
169	2209	2207	2208	2190	2184	2187	2119	2118	2119
170	2221	2218	2220	2201	2195	2198	2130	2125	2128
171	2228	2227	2228	2207	2203	2205	2139	2134	2137
172	2235	2233	2234	2214	2215	2215	2148	2143	2146
173	2242	2242	2242	2227	2228	2228	2155	2151	2153
174	2252	2252	2252	2236	2241	2239	2164	2159	2162
175	2265	2263	2264	2251	2247	2249	2170	2168	2169
176	2274	2268	2271	2257	2255	2256	2177	2172	2175
177	2281	2277	2279	2263	2266	2265	2188	2180	2184
178	2289	2287	2288	2274	2277	2276	2194	2191	2193
179	2298	2293	2296	2280	2283	2282	2204	2200	2202
180	2307	2303	2305	2298	2294	2296	2213	2203	2208

Πίνακας 4.1. Μετρήσεις των 3 σερβοκινητήρων.

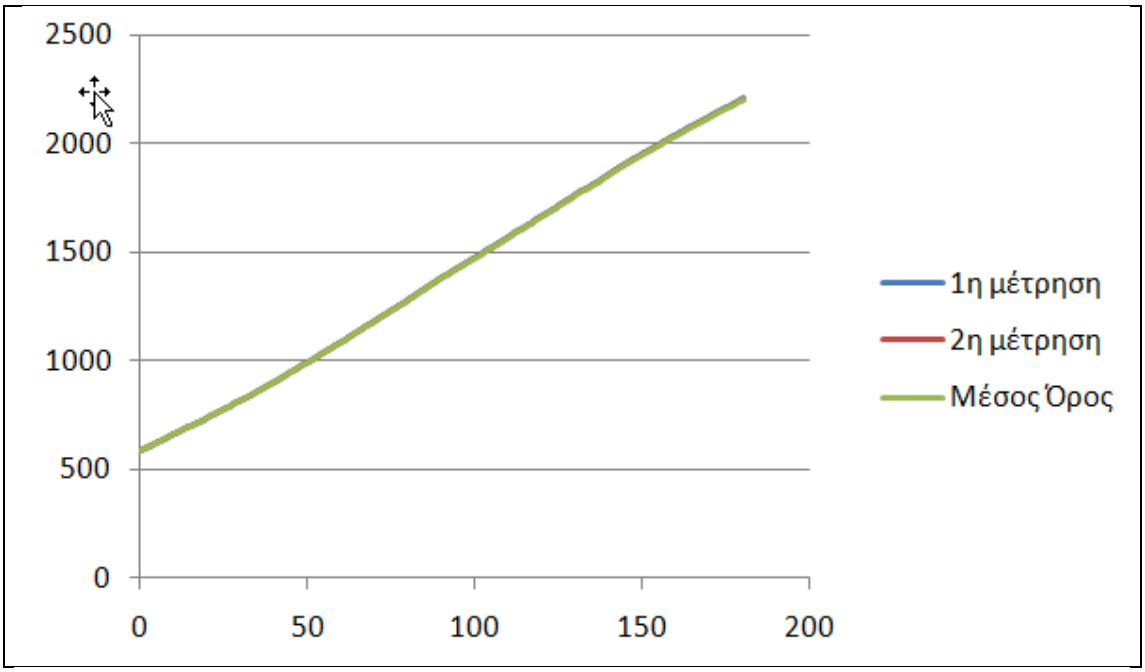
Στη συνέχεια, παρατίθενται τα γραφήματα με τις γραφικές παραστάσεις μοιρών - microseconds για κάθε servo:



Γράφημα 4.1. Χαρακτηριστική ευθεία του servo 1.



Γράφημα 4.2. Χαρακτηριστική ευθεία του servo 2.

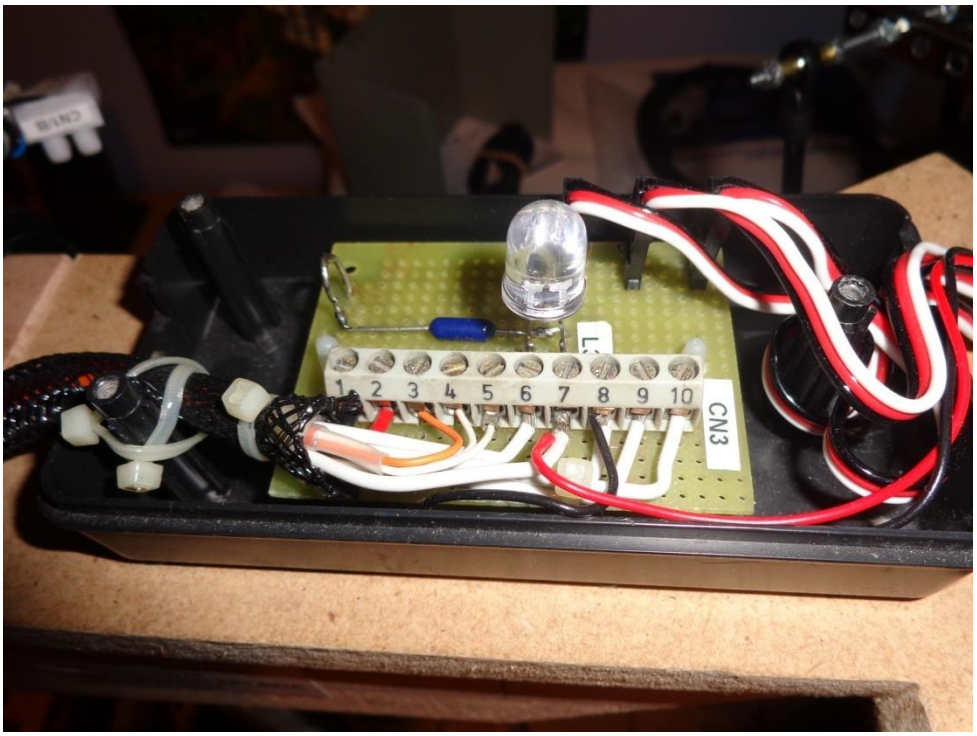


Γράφημα 4.3. Χαρακτηριστική ευθεία του servo 3.

4.2. Κουτί συνδέσεων - Ηλεκτρομαγνήτης - Καλώδιο διασύνδεσης

Όπως έχουμε αναφέρει παραπάνω, το κάθε σέρβο διαθέτει μία καλωδιοταινία τριών αγωγών για τη σύνδεσή του με το κύκλωμα οδήγησής του. Αυτές τις τρεις καλωδιοταινίες τις οδηγούμε στο κουτί συνδέσεων.

Εικόνα 4.8. Κουτί συνδέσεων (κλειστό).

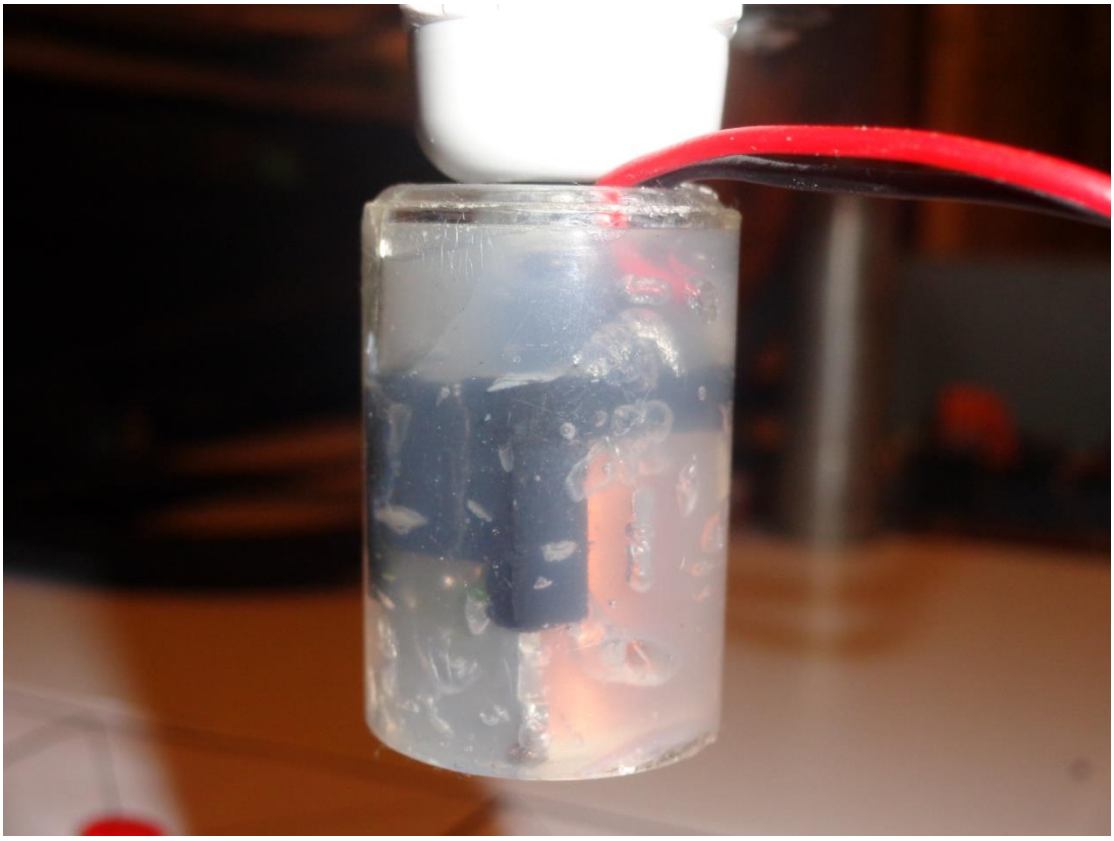


Εικόνα 4.9. Κουτί συνδέσεων (ανοιχτό) και το κύκλωμά του.

Από το κουτί συνδέσεων ξεκινούν δύο αγωγοί απαραίτητοι για την τροφοδοσία του ηλεκτρομαγνήτη που έχουμε προσαρμόσει στο end effector (Εικόνα 4.10). Συγκεκριμένα, οι δύο αγωγοί, αφού στερεωθούν πάνω στο r_e και r_f, συνδέονται σε μια κλέμα δύο πόλων πάνω στο end effector, παράλληλα με μια δίοδο σβέσης ανάστροφα πολωμένη. Αυτό γίνεται, ώστε να αποφύγουμε ζημιά στο υπόλοιπο κύκλωμα από την ανάστροφη τάση που εμφανίζεται στα άκρα του πηνίου τη στιγμή της διακοπής της τροφοδοσίας του. Στην άλλη πλευρά της κλέμας συνδέεται ο ηλεκτρομαγνήτης. Όπως

φαίνεται και στο ηλεκτρονικό σχέδιο 4.2, μέσω του ρελέ RL1 ελέγχουμε την σύνδεση των 12V στο ένα άκρο του πηνίου, ενώ το άλλο του άκρο είναι μόνιμα συνδεδεμένο στα -5V. Έτσι η τελική διαφορά δυναμικού που εφαρμόζεται πάνω στο πηνίο είναι 17V. Η συγκεκριμένη τροφοδοσία των 17V στο end effector δεν αφορά αποκλειστικά την τροφοδοσία του ηλεκτρομαγνήτη, αλλά και οποιασδήποτε άλλης μελλοντικής εφαρμογής, δεδομένου ότι αυτή ελέγχεται από το arduino μέσω του ρελέ RL1.

Για την κατασκευή του ηλεκτρομαγνήτη χρησιμοποιήθηκε το πηνίο ενός ρελέ με τάση διέγερσης 17V. Στο πηνίο αυτό κολλήθηκε ένας μεγάλου μήκους άξονας ποτενσιομέτρου. Στη συνέχεια, το πηνίο με τον άξονα τοποθετήθηκαν μέσα σε μια διαφανή θήκη, την οποία γεμίσαμε με σιλικόνη, ώστε να μένουν σταθερά. Ο ρόλος του άξονα ποτενσιομέτρου είναι η προσαρμογή του ηλεκτρομαγνήτη πάνω στο end effector, με τη βοήθεια ενός στυπιοθλήπτη (Εικόνα 3.8). Να σημειωθεί ότι η επιλογή του συγκεκριμένου πηνίου έγινε με δοκιμές, ελέγχοντας κατά πόσο μπορεί να σηκώνει τα αντικείμενα που απαιτούνται για τη συγκεκριμένη εφαρμογή χωρίς να προσθέτει μεγάλο βάρος και όγκο στην κατασκευή.



Εικόνα 4.10. Ηλεκτρομαγνήτης

Επανερχόμενοι στο κουτί συνδέσεων, πρέπει να αναφερθεί ότι έχει ενσωματωθεί σε αυτό και ένα διαφανές/κόκκινο flashing LED Φ 10mm (L3) σε σειρά με μια αντίσταση R3 = 330Ω, το οποίο θα αναβοσβήνει όταν τρέχει κάποιο πρόγραμμα λειτουργίας και θα προειδοποιεί για το ενδεχόμενο κίνησης του ρομπότ.

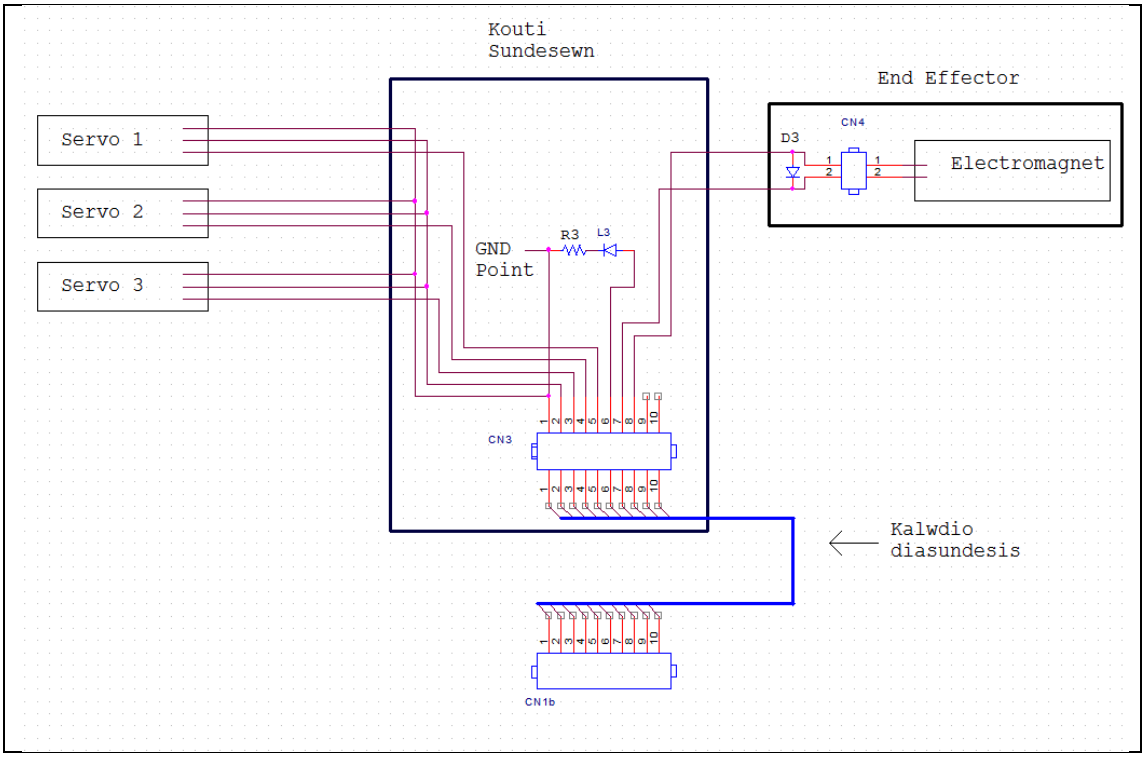
Θεωρήσαμε αναγκαία τη χρήση του συγκεκριμένη κουτιού πρώτον για να γίνεται εύκολα η σύνδεση και, αν χρειαστεί, η αποσύνδεση των εξόδων που ελέγχουμε (σέρβο, ηλεκτρομαγνήτης, L3) και δεύτερον για να γίνει ομαδοποίηση αγωγών οι οποίοι χρειάζονται κοινό σημείο σύνδεσης (τροφοδοσία, γείωση), έτσι ώστε να απαιτείται μικρότερος αριθμός αγωγών στο καλώδιο διασύνδεσης (Εικόνα 4.11) του κουτιού συνδέσεων με το κέντρο λειτουργίας.

Εικόνα 4.11. Καλώδιο διασύνδεσης.

Όπως αναφέραμε και στην περιγραφή του κατασκευαστικού κομματιού, θέλοντας να δώσουμε τη δυνατότητα επέκτασης της κατασκευής, στο καλώδιο διασύνδεσης υπάρχουν αγωγοί, οι οποίοι έχουν τερματιστεί στο κουτί συνδέσεων και συγκεκριμένα στις θέσεις της κλεμοσειράς CN3 / 9, 10. Αυτοί οι αγωγοί δε χρησιμοποιούνται στην παρούσα εφαρμογή, αλλά είναι διαθέσιμοι για οποιαδήποτε επέκτασή της.

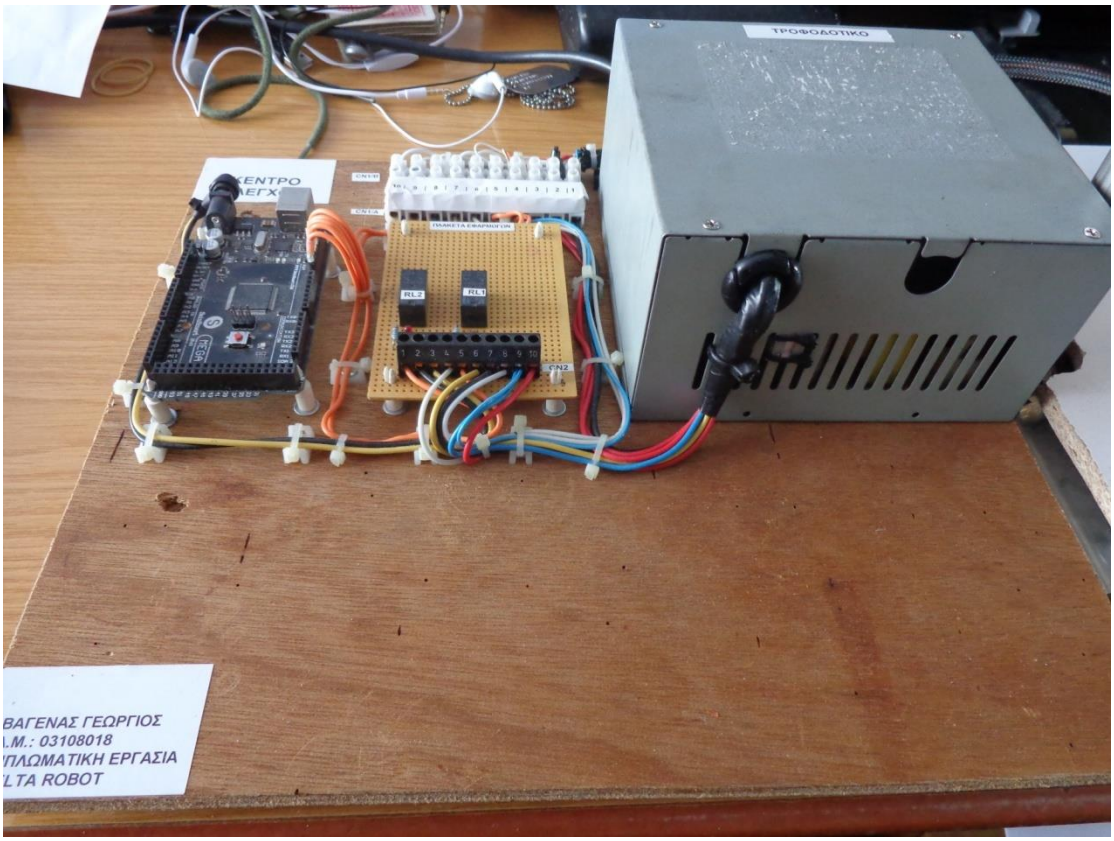
Το είδος και ο αριθμός των αγωγών που χρησιμοποιούμε (θωρακισμένα (shield) καλώδια και αγωγοί διαφόρων διατομών) για τη διασύνδεση δεν υπάρχει σε καλώδιο έτοιμο στο εμπόριο. Για το λόγο αυτό επιλέξαμε να οδηγήσουμε όλους αυτούς τους αγωγούς μέσα από ένα δίχτυ ομαδοποίησης/προστασίας, το οποίο μας προσέφερε και την επιθυμητή ευκαμψία.

Όλες αυτές οι συνδέσεις που αναφέρονται παραπάνω φαίνονται στο Σχέδιο 4.1.



4.1. Ηλεκτρονικό σχέδιο του κυρίως σώματος του ρομπότ.

4.3. Κέντρο λειτουργίας



Εικόνα 4.12. Κέντρο λειτουργίας.

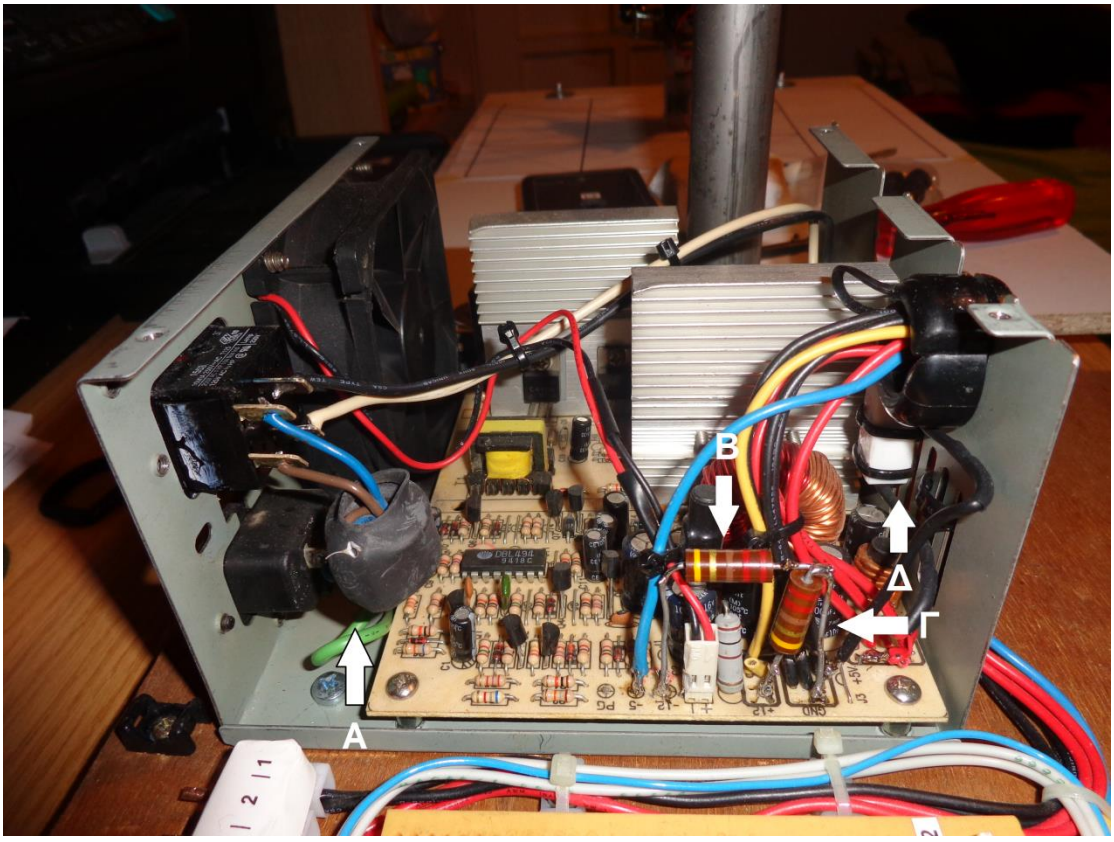
Τροφοδοτικό

Η πιο συμφέρουσα και σωστή επιλογή για την τροφοδοσία των ηλεκτρονικών κυκλωμάτων της κατασκευής είναι η χρήση ενός τροφοδοτικού από PC (Εικόνα 4.13), επειδή μπορεί να μας παρέχει όλες τις τάσεις που χρειαζόμαστε στην ισχύ που απαιτείται. Ωστόσο, ήταν αναγκαίο να γίνουν κάποιες τροποποιήσεις, ώστε να προσαρμόσουμε τη λειτουργικότητά του έξω από το περιβάλλον ενός PC.

Εικόνα 4.13. Τροφοδοτικό (εξωτερικά).

Αρχικά, αφαιρέσαμε τον διακόπτη ON/OFF, το διακόπτη μεταγωγής 220V/110V και την πρίζα που έδινε τάση στην οθόνη (Εικόνα 4.14). Στη θέση της πρίζας αυτής ενσωματώσαμε έναν νέο φωτιζόμενο διακόπτη ON/OFF. Στη συνέχεια τοποθετήθηκε ένα φίλτρο στην είσοδο τροφοδοσίας (Εικόνα 4.15 Α), για την προστασία από παράσιτα που μπορεί να μας φέρει το δίκτυο και τα οποία επηρεάζουν σε πολύ μεγάλο βαθμό τη λειτουργία του συγκεκριμένου είδους παλμοτροφοδοτικών (switching).

Εικόνα 4.14. Υλικά που αφαιρέθηκαν από το αρχικό τροφοδοτικό.



Εικόνα 4.15. Τροφοδοτικό (εσωτερικό), όπου φαίνονται τα υλικά που προστέθηκαν.

Τα παραπάνω τροφοδοτικά είναι κατασκευασμένα για να λειτουργούν πάντα υπό φορτίο, ακριβώς όπως συμβαίνει κατά τη λειτουργία τους σε ένα PC. Στη δική μας εφαρμογή όμως, επειδή το βασικό φορτίο μας είναι πολύ μικρό, το τροφοδοτικό μετά από μερικά δευτερόλεπτα λειτουργίας, «έκοβε» την έξοδό του. Για το λόγο αυτό, συνδέσαμε εσωτερικά στο τροφοδοτικό αντιστάσεις, ούτως ώστε το τροφοδοτικό να βλέπει πάντα ικανοποιητικό φορτίο και να μην διακόπτει την έξοδό του.

Συγκεκριμένα, τοποθετήσαμε:

- μία αντίσταση 220Ω/2W μεταξύ -12V και γης, που δημιουργεί ρεύμα I = V/R = 12V/220Ω = 54mA και καταναλώνει ισχύ $P = V^2/R = (12V)^2/220Ω = 0.65W$, οπότε και η αντίσταση των 2W που βάλαμε υπερκαλύπτει την καταναλισκόμενη ισχύ. (Εικόνα 4.15 Β)
- μία αντίσταση 220Ω/2W μεταξύ +12V και γης, που δημιουργεί ρεύμα I = V/R = 12V/220Ω = 54mA και καταναλώνει ισχύ $P = V^2/R = (12V)^2/220Ω = 0.65W$, οπότε και η αντίσταση των 2W που βάλαμε υπερκαλύπτει την καταναλισκόμενη ισχύ. (Εικόνα 4.15 Γ)
- μία αντίσταση 18Ω/5W μεταξύ +5V και γης, που δημιουργεί ρεύμα I = V/R = 5V/18Ω = 278mA και καταναλώνει ισχύ $P = V^2/R = (5V)^2/18Ω = 1.4W$, οπότε και η αντίσταση των 5W που βάλαμε υπερκαλύπτει την καταναλισκόμενη ισχύ. (Εικόνα 4.15 Δ)

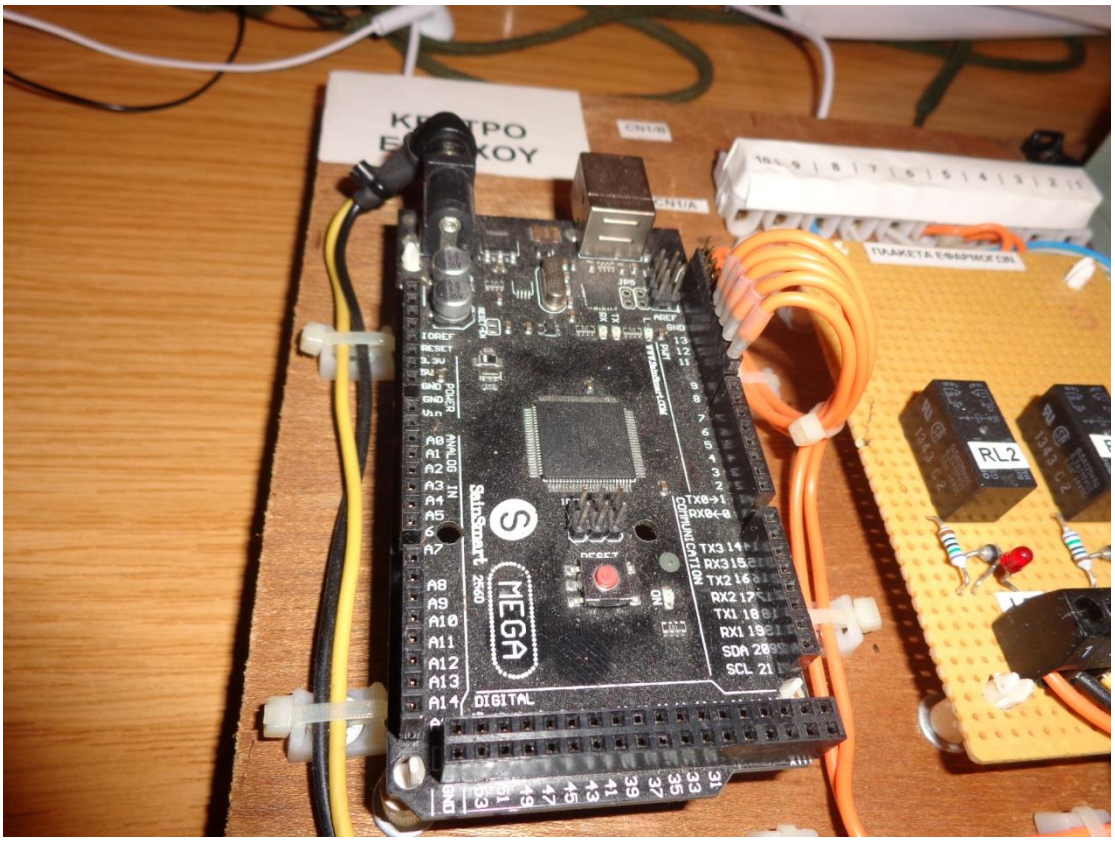
Τα ανωτέρω φορτία είναι τα μικρότερα με τα οποία διαπιστώθηκε η σωστή και χωρίς διακοπές λειτουργία του τροφοδοτικού.

Στη συνέχεια, αφαιρέθηκαν αρκετά καλώδια από τις τάσεις εξόδου και κρατήσαμε μόνο αυτά που είναι απαραίτητα για τη συνδεσμολογία του τροφοδοτικού στο κέντρο λειτουργίας.

Τέλος, είναι σημαντικό να αναφερθεί η αντιστοιχία χρώματος καλωδίου και τάσης:

- Το κόκκινο χρώμα αντιστοιχεί στα +5V
- Το κίτρινο χρώμα αντιστοιχεί στα +12V
- Το μπλε χρώμα αντιστοιχεί στα -5V
- Το μαύρο χρώμα αντιστοιχεί στη γη (GND)

Arduino



Εικόνα 4.16. Arduino.

Το Arduino είναι μια υπολογιστική πλατφόρμα βασισμένη σε μια απλή μητρική πλακέτα ανοικτού κώδικα, με ενσωματωμένο μικροελεγκτή και εισόδους/εξόδους, και η οποία μπορεί να προγραμματιστεί με τη γλώσσα Wiring. Στη συγκεκριμένη εφαρμογή χρησιμοποιήθηκε το Arduino Mega. Η πλακέτα αποτελείται από ένα μικροελεγκτή Atmel ATmega1280 και συμπληρωματικά εξαρτήματα για την διευκόλυνση του χρήστη στον προγραμματισμό και την ενσωμάτωση του σε άλλα κυκλώματα. Όλες οι πλακέτες περιλαμβάνουν ένα γραμμικό ρυθμιστή τάσης 5V και έναν κρυσταλλικό ταλαντωτή 16MHz (ή κεραμικό αντηχητή σε κάποιες παραλλαγές). Ο μικροελεγκτής είναι από κατασκευής προγραμματισμένος με ένα bootloader, έτσι ώστε να μην χρειάζεται εξωτερικός προγραμματιστής.

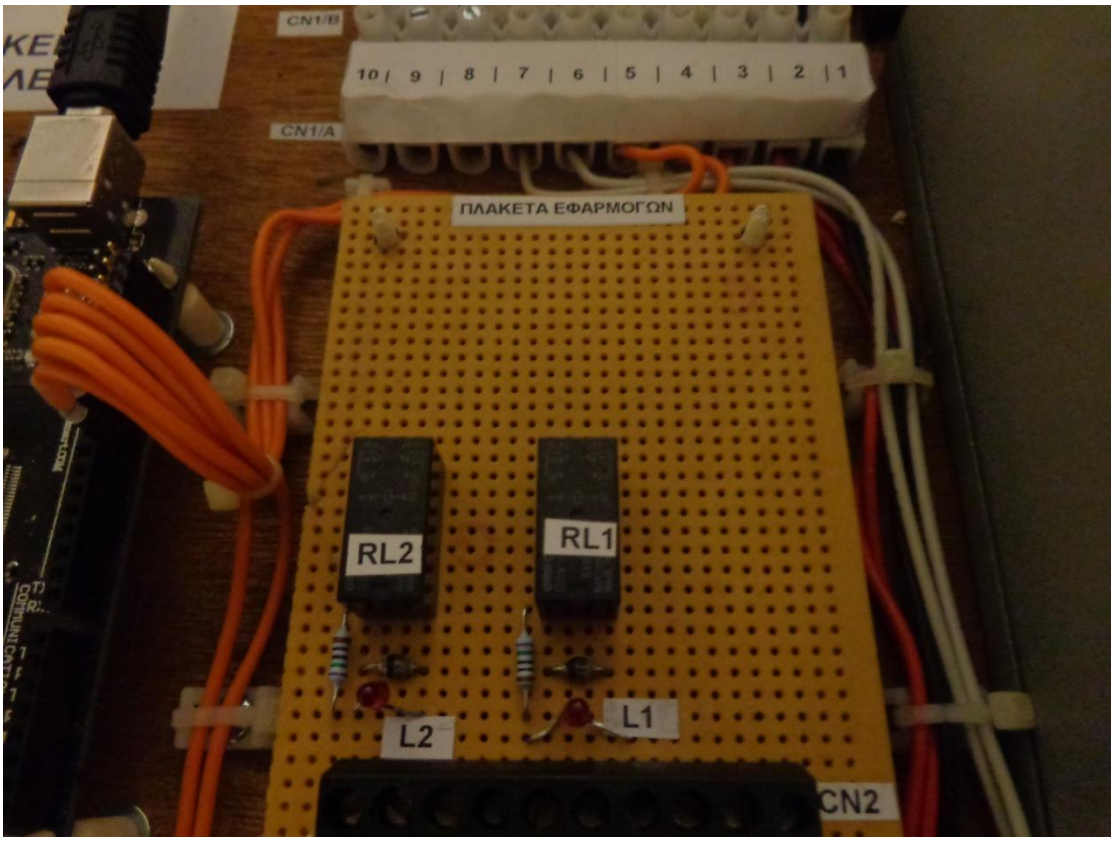
Πλακέτα εφαρμογών

Στο κέντρο λειτουργίας έχουμε τοποθετήσει μία διάτρητη πλακέτα (Εικόνα 4.17), για να κατασκευάσουμε όλα τα κυκλώματα που είναι απαραίτητα για τη λειτουργία της εφαρμογής μας.

Για να τροφοδοτήσουμε τον ηλεκτρομαγνήτη που έχουμε στο end effector, ο οποίος απαιτεί τάση 17V και ρεύμα περίπου 60mA, τοποθετήσαμε στην πλακέτα εφαρμογών ένα ρελέ (Εικόνα 4.17 Α), το οποίο ελέγχεται από το arduino (pin12 - CN2/pin5) και αυτό με τη σειρά του ελέγχει τα +12V προς τον ηλεκτρομαγνήτη (CN2/pin8 - CN1a/pin7). Παράλληλα στο πηνίο του ρελέ έχουμε τοποθετήσει ένα LED (σε σειρά με μία αντίσταση), ώστε να έχουμε μια οπτική ένδειξη ενεργοποίησης του ρελέ. Επιπλέον, έχουμε συνδέσει παράλληλα και μία δίοδο σβέσης, ώστε να αποφύγουμε ζημιά στο υπόλοιπο κύκλωμα από την

ανάστροφη τάση που εμφανίζεται στα άκρα του πηνίου τη στιγμή της διακοπής της τροφοδοσίας του.

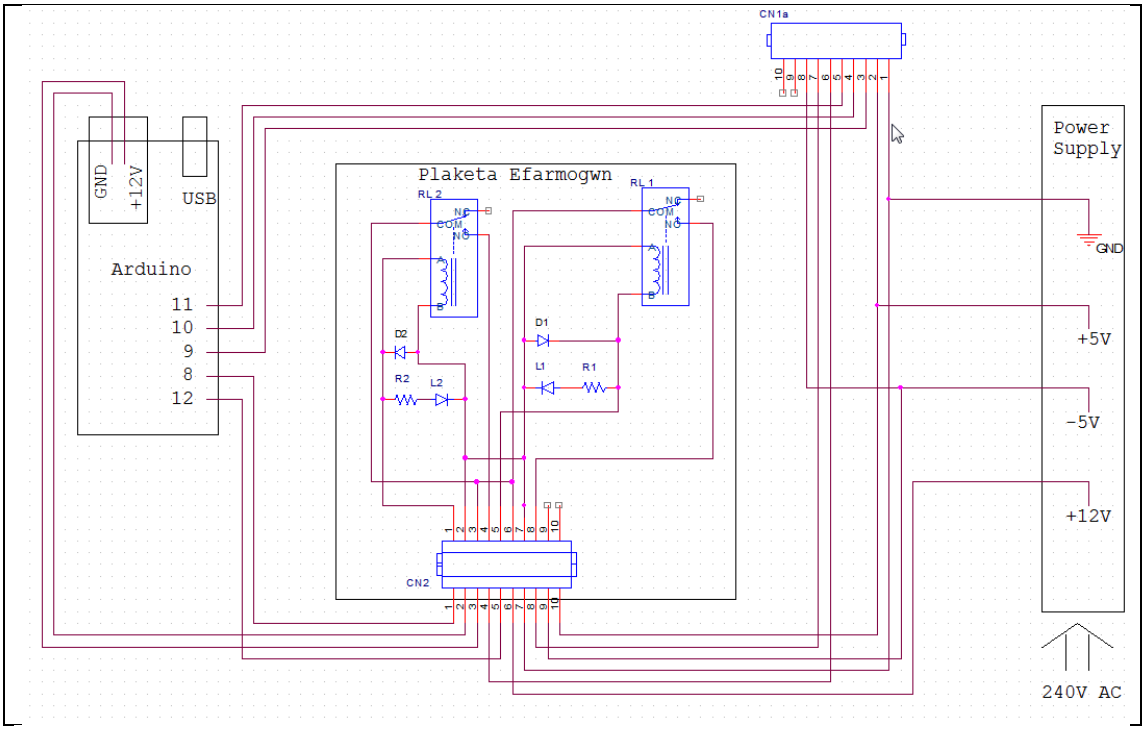
Με τον ίδιο ακριβώς τρόπο οδηγούμε από το arduino (pin8 - CN2/pin1) το δεύτερο ρελέ που βρίσκεται στην πλακέτα εφαρμογών (CN2/pin7 - CN1a/pin1) (Εικόνα 4.17 Β), ώστε να ελέγχουμε το LED εργασίας που βρίσκεται στο κουτί συνδέσεων πάνω στη βάση στήριξης των σέρβο.



Εικόνα 4.17. Πλακέτα εφαρμογών.

Στην πλακέτα εφαρμογών οδηγούνται όλες οι τάσεις του τροφοδοτικού και υπάρχει αρκετός χώρος για κατασκευή νέων κυκλωμάτων για διαφορετικές εφαρμογές.

Όλες οι συνδέσεις του κέντρου λειτουργίας φαίνονται αναλυτικά στο Σχέδιο 4.2.



Σχέδιο 4.2. Ηλεκτρονικό σχέδιο του κέντρου λειτουργίας.

5. Λίστα υλικών

Τροφοδοτικό

τροφοδοτικό PC
καλώδιο τροφοδοσίας AC
αντιπαρασιτικό φίλτρο
φωτιζόμενος διακόπτης ON/OFF
2 αντιστάσεις 220Ω/2W
1 αντίσταση 18Ω/5W
2 βίδες M4x15mm
2 περικόχλια M4mm
2 ροδέλες Φ4mm
2 γκρόβερ Φ4mm

Πλακέτα Εφαρμογών

διάτρητη πλακέτα 100mm x 70mm
κλεμοσειρά πλακέτας 2.5mm^2 10 θέσεων
2 κόκκινα LED 3mm
2 δίοδοι 127-150
2 αντιστάσεις 750Ω/0.25W
2 ρελέ 5V μιας μεταγωγικής επαφής
αγωγός ενσυρμάτωσης πλακέτας
4 πλαστικά αποστατικά
4 βίδες M4mm x 10mm
8 ροδέλες Φ4mm

Arduino

Arduino Mega 2560

4 πλαστικά αποστατικά
4 βίδες M4mm x 10mm
8 ροδέλες Φ4mm
DC κονέκτορας τροφοδοσίας Φ5.5mm

Υπόλοιπα υλικά κέντρου ελέγχου
Κόντρα-πλακέ 340mm x 300mm
4 ελαστικά πατήματα Φ30mm x 12mm
4 νοβοπανόβιδες M4mm x 10mm
4 ροδέλες Φ4mm
1 κλεμοσειρά καλωδίου 2.5mm^2 10 θέσεων
12 βάσεις συγκράτησης καλωδίων μεσαίου μεγέθους
12 νοβοπανόβιδες M4mm x 10mm
12 ροδέλες Φ4mm
15 μικρά δεματικά καλωδίων
5 διαφανή θερμοσυστελλόμενα Φ3mm μήκους 10mm
2 m καλώδια διαφόρων χρωμάτων 0.5mm^2

Μεταλλική βάση ρομπότ - οδηγός ύψους (Σχέδιο 3.3)
Τραπέζι εργασίας από μελαμίνη 450mm x 300mm x 8mm
4 ελαστικά πατήματα Φ30mm x 12mm
4 βίδες M4mm x 15mm
4 ροδέλες Φ4mm
4 περικόχλια M4mm
Φορέας (Σχέδιο 3.2)
1 βίδα M8mm x 100mm
2 ροδέλες Φ8mm
2 νοβοπανόβιδες M3mm x 8mm
1 βίδα M2mm x 8mm
καστάνια ακινητοποίησης φορέα
3 βάσεις στήριξης καλωδίων

Βάση στήριξης σέρβο (Σχέδιο 3.1)
3 νοβοπανόβιδες M3mm x 10mm
24 βίδες M4mm x 8mm
6 βίδες M3mm x 25mm
6 βίδες M3mm x 80mm
24 περικόχλια M4mm
18 γκρόβερ Φ4mm
18 περικόχλια M3mm
24 ροδέλες Φ3mm
27 γκρόβερ Φ3mm
3 γωνιακοί σύνδεσμοι 6mm x 12mm με 10 τρύπες
6 σωληνάκια carbon Φ4mm/Φ3mm x 330mm
6 μπρούτζινα αποστατικά
24 αποστατικά σωληνάκια
1 βακελίτης σε σχήμα ισόπλευρου τριγώνου πλευράς 85mm
1 στυπιοθλήπτης
12 ball joints
3 γωνιακοί σύνδεσμοι (για τη σύνδεση των servo)
3 servo SM-S4315M

1 κλέμα 1.5mm^2 2 θέσεων
1 δίοδος 127-150

1 κουτί συνδέσεων με πλακέτα
1 κλεμοσειρά πλακέτας 1.5mm^2 10 θέσεων
1 διαφανές/κόκκινο flashing LED Φ10mm
1 αντίσταση 330Ω
1 βάση στήριξης καλωδίου
1 νοβοπανόβιδα M3mm x 8mm
5 δεματικά
4 νοβοπανίδες M4mm x 30mm
7 αγωγοί 1mm^2 1300mm
3 αγωγοί με μπλεντάζ 0.25mm^2 1300mm
πλαστικό δίχτυ προστασίας αγωγών 1300mm
14 δεματικά
1 κλεμοσειρά 2.5mm^2 10 θέσεων
2 αποστατικά πλακέτας
1 ηλεκτρομαγνήτης 17V DC
καλώδιο ενός ζεύγους 0.3mm^2 700mm

Γ. Ανάπτυξη και εφαρμογή software

6. Εφαρμογή του ρομπότ

6.1. Προγραμματίζοντας στο Arduino

Για τον προγραμματισμό του ρομπότ μας χρησιμοποιήθηκε ο μικροεπεξεργαστής Arduino Mega και η γλώσσα Wiring, όπως έχει αναφερθεί και παραπάνω.

Κάθε πρόγραμμα χωρίζεται σε 3 μέρη:

- τμήμα δηλώσεων, όπου δηλώνονται οι μεταβλητές, οι σταθερές και οι βιβλιοθήκες που πιθανόν να χρησιμοποιηθούν.
- τμήμα αρχικοποιήσεων, όπου ορίζεται ποια pins του arduino θα χρησιμοποιήσουμε και αρχικοποιείται η σειριακή οθόνη σε περίπτωση που θέλουμε να βλέπουμε κάτι σε αυτή. Το συγκεκριμένο κομμάτι ξεκινάει με την εντολή void setup() και το περιεχόμενό του περιέχεται μέσα σε αγκύλες ({περιεχόμενο}).
- κυρίως τμήμα, όπου γράφουμε το κυρίως πρόγραμμα. Το τμήμα αυτό ξεκινάει με την εντολή void loop() και το περιεχόμενό του περιέχεται μέσα σε αγκύλες ({περιεχόμενο}).

6.2. Η εφαρμογή του δικού μας ρομπότ

Σκοπός της εφαρμογής μας είναι το ρομπότ να μαζεύει 9 κεφαλές από πινέζες και να τις τοποθετεί μέσα σε ένα κουτάκι. (Εικόνα 3.14)

Στο δικό μας πρόγραμμα, στο τμήμα δηλώσεων δηλώνουμε τα 3 servo που θα χρησιμοποιήσουμε, ορίζουμε ως σταθερές τις 4 παραμέτρους του ρομπότ και ορίζουμε κάποιες βοηθητικές μαθηματικές σταθερές. Στη συνέχεια ορίζουμε τις συναρτήσεις που χρειάζονται για να «τρέχει» το ευθύ και αντίστροφο κινηματικό πρόβλημα. Είναι σημαντικό να αναφερθεί ότι μέσα από το πρόγραμμα ρυθμίζουμε τα microseconds του παλμού που στέλνουμε στα servo και όχι τις μοίρες στις οποίες θέλουμε να πάει αυτό, ώστε να είμαστε πιο ακριβείς. Επομένως, εκμεταλλευόμενοι τις μετρήσεις των servo, έχουμε 3 πίνακες 181 θέσεων (0 - 180 μοίρες), όπου γίνεται η αντιστοίχηση μοιρών - microseconds για κάθε servo. Τέλος, ορίζουμε από ποιο pin του arduino ελέγχεται ο ηλεκτρομαγνήτης.

Στο τμήμα αρχικοποιήσεων, ορίζουμε ως output το pin που ελέγχει τον ηλεκτρομαγνήτη και ορίζουμε για κάθε servo το pin από το οποίο ελέγχεται, καθώς και το εύρος των microseconds του παλμού που το στέλνουμε.

Στο κυρίως τμήμα του προγράμματος, μετά τον ορισμό κάποιων μεταβλητών, ορίζουμε έναν πίνακα πολύ σημαντικό για τη λειτουργία της εφαρμογής. Αυτός ο πίνακας περιλαμβάνει την ακολουθία των θέσεων στις οποίες πηγαίνει το ρομπότ κατά την εκτέλεση του προγράμματος. Οι στήλες δηλώνουν το x, y, z, πόσα microseconds θα παραμένει σε εκείνη τη θέση και αν ο ηλεκτρομαγνήτης είναι ενεργοποιημένος (1) ή όχι (0). Τέλος, ακολουθεί το κομμάτι εκείνο που δίνει τους παλμούς στα servo και ορίζει την ενεργοποίηση ή μη του ηλεκτρομαγνήτη, για κάθε θέση.

Το πρόγραμμα στο σύνολό του ακολουθεί παρακάτω. Έχουν τοποθετηθεί σχόλια σε οποιοδήποτε σημείο κρίνεται απαραίτητο για την ευκολότερη κατανόησή του.

```
#include <Servo.h> // Servo library
Servo servo_1;  // create servo object to control a servo
Servo servo_2;  // a maximum of eight servo objects can be created
Servo servo_3;

// robot geometry
const float e = 8.0;    // end effector
const float f = 23.32;      // base
const float re = 35.0;
const float rf = 6.0;

// trigonometric constants
const float sqrt3 = sqrt(3.0);
const float pi = 3.141592653;    // PI
const float sin120 = sqrt3/2.0;
const float cos120 = -0.5;
const float tan60 = sqrt3;
const float sin30 = 0.5;
const float tan30 = 1.0/sqrt3;

// forward kinematics: (theta1, theta2, theta3) -> (x0, y0, z0)
// returned status: 0=OK, -1=non-existing position
int delta_calcForward(float theta1, float theta2, float theta3, float &x0, float &y0, float &z0) {
   float t = (f-e)*tan30/2;
   float dtr = pi/180.0;

   theta1 *= dtr; //μετατροπή μοιρών σε ακτίνια
   theta2 *= dtr;
   theta3 *= dtr;

   float y1 = -(t + rf*cos(theta1));
   float z1 = -rf*sin(theta1);

   float y2 = (t + rf*cos(theta2))*sin30;
   float x2 = y2*tan60;
   float z2 = -rf*sin(theta2);

   float y3 = (t + rf*cos(theta3))*sin30;
   float x3 = -y3*tan60;
   float z3 = -rf*sin(theta3);

   float dnm = (y2-y1)*x3-(y3-y1)*x2;

   float w1 = y1*y1 + z1*z1;
   float w2 = x2*x2 + y2*y2 + z2*z2;
   float w3 = x3*x3 + y3*y3 + z3*z3;

   // x = (a1*z + b1)/dnm
   float a1 = (z2-z1)*(y3-y1)-(z3-z1)*(y2-y1);
   float b1 = -((w2-w1)*(y3-y1)-(w3-w1)*(y2-y1))/2.0;

   // y = (a2*z + b2)/dnm;
   float a2 = -(z2-z1)*x3+(z3-z1)*x2;
   float b2 = ((w2-w1)*x3 - (w3-w1)*x2)/2.0;

   // a*z^2 + b*z + c = 0
   float a = a1*a1 + a2*a2 + dnm*dnm;
```

```
   float b = 2*(a1*b1 + a2*(b2-y1*dnm) - z1*dnm*dnm);
   float c = (b2-y1*dnm)*(b2-y1*dnm) + b1*b1 + dnm*dnm*(z1*z1 - re*re);

   // discriminant
   float d = b*b - 4.0*a*c;
   if (d < 0) return -1; // non-existing point

   z0 = -0.5*(b+sqrt(d))/a;
   x0 = (a1*z0 + b1)/dnm;
   y0 = (a2*z0 + b2)/dnm;
   return 0;
}

int delta_calcAngleYZ(float x0, float y0, float z0, float &theta) {
   float y1 = -0.5 * 0.57735 * f; // f/2 * tg 30
   //float y1 = yy1;
   y0 -= 0.5 * 0.57735    * e;    // shift center to edge
   // z = a + b*y
   float a = (x0*x0 + y0*y0 + z0*z0 +rf*rf - re*re - y1*y1)/(2*z0);
   float b = (y1-y0)/z0;
   // discriminant
   float d = -(a+b*y1)*(a+b*y1)+rf*(b*b*rf+rf);
   if (d < 0) return -1; // non-existing point
   float yj = (y1 - a*b - sqrt(d))/(b*b + 1); // choosing outer point
   float zj = a + b*yj;
   theta = 180.0*atan(-zj/(y1 - yj))/pi + ((yj>y1)?180.0:0.0);
   if ((theta < 0) || (theta > 180))
      return -1;
   return 0;
}

// inverse kinematics: (x0, y0, z0) -> (theta1, theta2, theta3)
// returned status: 0=OK, -1=non-existing position
int delta_calcInverse(float x0, float y0, float z0, float &theta1, float &theta2, float &theta3) {
   theta1 = theta2 = theta3 = 0;
   int status = delta_calcAngleYZ(x0, y0, z0, theta1);
    status = delta_calcAngleYZ(x0*cos120 + y0*sin120, y0*cos120-x0*sin120, z0, theta2);  // rotate
coords to +120 deg
    status = delta_calcAngleYZ(x0*cos120 - y0*sin120, y0*cos120+x0*sin120, z0, theta3);  // rotate
coords to -120 deg
   //return status;
}

int servo1[] =
{641,651,659,668,676,680,690,700,706,714,722,730,737,746,754,763,772,779,788,796,808,815,825,
833,843,854,862,871,878,888,896,

903,910,919,932,943,951,960,969,976,986,996,1007,1016,1022,1032,1043,1050,1060,1071,1081,108
9,1097,1108,1116,1127,1137,1147,

1158,1169,1180,1189,1198,1206,1216,1226,1236,1248,1258,1268,1274,1285,1300,1307,1317,1328,1
338,1348,1357,1364,1374,1386,1395,

1406,1413,1421,1432,1443,1454,1463,1474,1483,1492,1498,1508,1520,1529,1538,1547,1557,1566,1
578,1586,1596,1607,1615,1625,1631,
```

```
1639,1655,1664,1675,1683,1694,1703,1711,1726,1737,1746,1756,1765,1770,1780,1791,1801,1812,1
821,1830,1839,1851,1861,1867,1877,

1887,1894,1902,1911,1924,1933,1940,1948,1956,1968,1977,1985,1994,2004,2011,2023,2030,2040,2
049,2057,2065,2075,2084,2093,2103,

2109,2119,2129,2137,2146,2153,2161,2170,2180,2188,2199,2208,2220,2228,2234,2242,2252,2264,2
271,2279,2288,2296,2305};

int servo2[] =
{587,594,600,608,612,620,627,633,641,651,659,662,672,680,688,695,703,713,721,730,738,744,752,
761,768,777,783,791,802,809,819,

827,835,847,856,861,873,883,889,896,907,916,925,935,941,953,961,970,976,984,995,1003,1012,102
4,1033,1043,1050,1059,1069,1077,

1086,1100,1105,1117,1128,1136,1146,1156,1165,1178,1185,1194,1206,1213,1225,1232,1241,1251,1
261,1268,1275,1288,1298,1307,1321,

1330,1335,1348,1356,1369,1373,1384,1396,1408,1418,1426,1437,1449,1460,1469,1476,1488,1498,1
507,1520,1531,1540,1549,1560,1572,

1583,1593,1601,1614,1623,1637,1645,1656,1672,1682,1691,1700,1710,1717,1729,1743,1754,1764,1
776,1789,1800,1812,1818,1829,1841,

1851,1862,1869,1882,1888,1898,1911,1920,1931,1943,1954,1961,1970,1976,1988,2000,2008,2016,2
029,2038,2047,2056,2064,2080,2090,

2099,2109,2120,2132,2139,2147,2159,2170,2178,2187,2198,2205,2215,2228,2239,2249,2256,2265,2
276,2282,2296};

int servo3[] =
{587,596,603,609,616,626,632,640,648,657,665,671,679,688,696,701,708,716,722,729,739,749,755,
764,771,780,787,794,806,814,820,

827,834,843,852,861,870,878,888,896,904,911,923,931,941,952,959,968,980,987,995,1003,1014,102
3,1034,1042,1052,1060,1071,1080,

1088,1096,1108,1118,1128,1138,1146,1158,1167,1175,1185,1196,1206,1215,1224,1235,1243,1254,1
263,1272,1284,1294,1303,1315,1325,

1333,1346,1356,1365,1376,1386,1395,1402,1413,1423,1432,1440,1451,1460,1469,1477,1486,1496,1
508,1518,1524,1534,1544,1554,1563,

1573,1584,1596,1603,1610,1621,1630,1637,1652,1661,1670,1680,1688,1698,1706,1719,1728,1738,1
746,1757,1769,1779,1788,1793,1802,

1813,1821,1830,1843,1852,1862,1874,1882,1893,1902,1912,1920,1931,1938,1946,1957,1964,1972,1
981,1991,1998,2010,2017,2027,2036,

2042,2053,2060,2070,2079,2086,2095,2102,2111,2119,2128,2137,2146,2153,2162,2169,2175,2184,2
193,2202,2208};

int em = 12; //ο ηλεκτρομαγνήτης ελέγχεται από το pin 12 του arduino
int led = 8; //το LED του κουτιού συνδέσεων ελέγχεται από το pin 8 του arduino
```

```
void setup()

{  Serial.begin(9600);
  pinMode(13, OUTPUT);
  pinMode(em, OUTPUT);
  pinMode(led, OUTPUT);
  servo_1.attach(9,500,2500);   // attaches the servo on pin 9  to the servo object
  servo_2.attach(10,400,2500);  // attaches the servo on pin 10 to the servo object
  servo_3.attach(11,400,2500);  // attaches the servo on pin 11 to the servo object
}



void loop()

{  Serial.begin(9600);
    float t1;
    float t2;
    float t3;

    int angle_1,angle_2,angle_3;
    int offset_1 =111; // σε αυτές τις θέσεις τα rf είναι οριζόντια
    int offset_2 =110;
    int offset_3 =116;


    int i;



    //ορισμός πίνακα θέσεων
   float positionMatrix[][5] = {
  {0,0,-31.0,1000,0},
  {-3.3,-6.7,-31.0,1000,0},
  {-3.3,-6.7,-34.0,1000,0},
  {-3.3,-6.7,-34.0,1000,1},
  {-3.3,-6.7,-31.0,1000,1},
  {-10,0,-31.0,1000,1},
  {-10,0,-32.0,1000,1},
  {-10,0,-32.0,1000,0},  //end of step 1

  {-10,0,-31.0,1000,0},
  {-3.3,0,-31.0,1000,0},
  {-3.3,0,-34.0,1000,0},
  {-3.3,0,-34.0,1000,1},
  {-3.3,0,-31.0,1000,1},
  {-10,0,-31.0,1000,1},
  {-10,0,-32.0,1000,1},
  {-10,0,-32.0,1000,0},   //end of step 2

  {-10,0,-31.0,1000,0},
  {-3.3,6.7,-31.0,1000,0},
  {-3.3,6.7,-34.0,1000,0},
  {-3.3,6.7,-34.0,1000,1},
  {-3.3,6.7,-31.0,1000,1},
  {-10,0,-31.0,1000,1},
  {-10,0,-32.0,1000,1},
```

```
{-10,0,-32.0,1000,0},  //end of step 3

{-10,0,-31.0,1000,0},
{3.3,-6.7,-31.0,1000,0},
{3.3,-6.7,-34.0,1000,0},
{3.3,-6.7,-34.0,1000,1},
{3.3,-6.7,-31.0,1000,1},
{-10,0,-31.0,1000,1},
{-10,0,-32.0,1000,1},
{-10,0,-32.0,1000,0},   //end of step 4

{-10,0,-31.0,1000,0},
{3.3,0,-31.0,1000,0},
{3.3,0,-34.0,1000,0},
{3.3,0,-34.0,1000,1},
{3.3,0,-31.0,1000,1},
{-10,0,-31.0,1000,1},
{-10,0,-32.0,1000,1},
{-10,0,-32.0,1000,0},   //end of step 5

{-10,0,-31.0,1000,0},
{3.3,6.7,-31.0,1000,0},
{3.3,6.7,-34.0,1000,0},
{3.3,6.7,-34.0,1000,1},
{3.3,6.7,-31.0,1000,1},
{-10,0,-31.0,1000,1},
{-10,0,-32.0,1000,1},
{-10,0,-32.0,1000,0},   //end of step 6

{-10,0,-31.0,1000,0},
{10.0,-6.7,-31.0,1000,0},
{10.0,-6.7,-34.0,1000,0},
{10.0,-6.7,-34.0,1000,1},
{10.0,-6.7,-31.0,1000,1},
{-10,0,-31.0,1000,1},
{-10,0,-32.0,1000,1},
{-10,0,-32.0,1000,0},  //end of step 7

{-10,0,-31.0,1000,0},
{10.0,0,-31.0,1000,0},
{10.0,0,-34.0,1000,0},
{10.0,0,-34.0,1000,1},
{10.0,0,-31.0,1000,1},
{-10,0,-31.0,1000,1},
{-10,0,-32.0,1000,1},
{-10,0,-32.0,1000,0},   //end of step 8

{-10,0,-31.0,1000,0},
{10.0,6.7,-31.0,1000,0},
{10.0,6.7,-34.0,1000,0},
{10.0,6.7,-34.0,1000,1},
{10.0,6.7,-31.0,1000,1},
{-10,0,-31.0,1000,1},
{-10,0,-32.0,1000,1},
{-10,0,-32.0,1000,0},  //end of step 9

{-10,0,-31,1000,0},
```

```
 {0,0,-31,1000,0}
};

  digitalWrite(led, HIGH); //ενεργοποίηση του LED μόλις ξεκινάει η κίνηση

 //all servos horizontal
 servo_1.writeMicroseconds(servo1[offset_1]);
 servo_2.writeMicroseconds(servo2[offset_2]);
 servo_3.writeMicroseconds(servo3[offset_3]);

 delay(2000);

 //main function
 for (int i=0; i <= 73; i++){

  delta_calcInverse (positionMatrix[i][0],positionMatrix[i][1],positionMatrix[i][2],t1,t2,t3);
   angle_1 = round (t1);
  Serial.println(angle_1);
    servo_1.writeMicroseconds(servo1[offset_1-angle_1]);

    angle_2 = round (t2);
  Serial.println(angle_2);
    servo_2.writeMicroseconds(servo2[offset_2-angle_2]);

    angle_3 = round (t3);
  Serial.println(angle_3);
 Serial.println(    );
   servo_3.writeMicroseconds(servo3[offset_3-angle_3]);

  if (positionMatrix[i][4] == 0){
   digitalWrite(em, LOW);
  }
  else {
   digitalWrite(em, HIGH);
  }

  delay (positionMatrix[i][3]);
 }

 digitalWrite(led, LOW); //απενεργοποίηση του LED μόλις σταματάει η λειτουργία

    //all servos horizontal
 servo_1.writeMicroseconds(servo1[offset_1]);
 servo_2.writeMicroseconds(servo2[offset_2]);
 servo_3.writeMicroseconds(servo3[offset_3]);

 delay(10000);
}
```

6.3. Μελλοντικές εφαρμογές

Ο βασικός σκοπός αυτής της διπλωματικής εργασίας ήταν η κατασκευή του ρομπότ και όχι η ανάπτυξη κάποιας ιδιαίτερα απαιτητικής εφαρμογής. Γι’ αυτό και η εφαρμογή που αναπτύξαμε έχει ως σκοπό την ανάδειξη της ομαλής και σωστής λειτουργίας του ρομπότ. Ωστόσο, το ίδιο το ρομπότ εκ κατασκευής (μηχανολογικά και ηλεκτρονικά), αλλά και το software δίνουν τη δυνατότητα για χρήση του ρομπότ σε περισσότερο πολύπλοκες εφαρμογές. Ενδιαφέρον παρουσιάζουν οι περιπτώσεις όπου το ρομπότ θα μπορεί να αναγνωρίζει αντικείμενα μέσω μιας κάμερας και έτσι, με το κατάλληλο λογισμικό, να μπορεί να «παίζει» παιχνίδια, όπως σκάκι ή ντάμα, είτε με τον ηλεκτρομαγνήτη, είτε με αντικατάστασή του με μια αρπάγη.

Βιβλιογραφία

[1] Εισαγωγή στη Ρομποτική - John J Craig

[2] Ρομποτική - Σπύρος Γ. Τζαφέστας

[3] Προσομοίωση κίνησης delta robot στο Matlab, για χρήση σε διαδραστικά παιχνίδια - Διπλωματική εργασία του διπλωματούχου ΗΜΜΥ Αγγελίνου Ευάγγελου

[4] Descriptive Geometric Kinematic Analysis of Clavel's "Delta" Robot - P.J. Zsombor-Murray

Αναφορές στο διαδίκτυο

[1] http://en.wikipedia.org/wiki/Delta_robot

[2] http://forums.trossenrobotics.com/tutorials/introduction-129/delta-robot-kinematics-3276/

[3] http://www.jfedor.org/lego/delta.nxc

[4] http://www.parallemic.org/Reviews/Review002.html

[5] http://phys.org/news/2011-11-delta-robot-swiss-made-fastest-world.html

[6] https://sites.google.com/site/volksrobot/

[7] http://mattgreensmith.wordpress.com/2011/11/26/making-an-arduino-controlled-delta-robot/

[8] http://roboticarts.wordpress.com/tag/diy-delta-robot-robotics-arduino/

[9] http://cats-fs.rpi.edu/~wenj/ECSE641S07/delta_calibration.pdf

[10] http://docs.nonpolynomial.com/libnifalcon/pdf/DeltaRobot-InverseDirectAndIntermediateJacobians.pdf

[11] http://www.phidgets.com/products.php?product_id=3208_0

[12] http://www.youtube.com/watch?v=-XSXfqd1N58