ΧΩΡΙΚΟΣ ΒΡΑΧΙΟΝΑΣ 3ΒΕ

**Πίνακας περιεχομένων**

Contents

[Η κατασκευή 2](#_Toc525241064)

[Source code 3](#_Toc525241065)

[Hardware 3](#_Toc525241066)

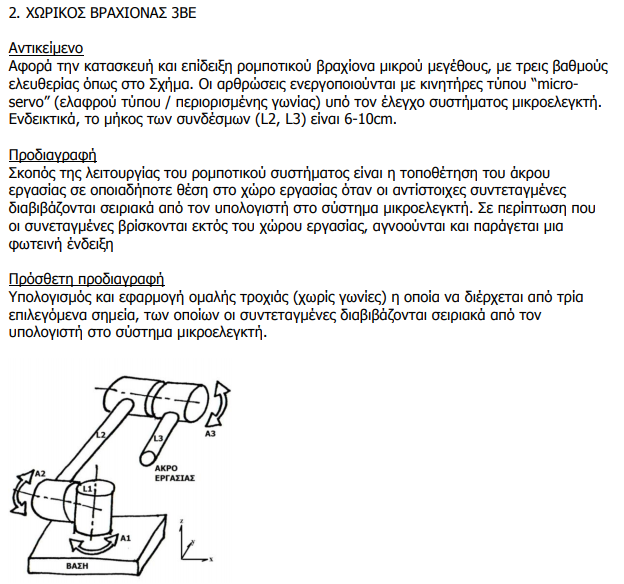
[Τα μαθηματικά πίσω από τη κατασκευή 4](#_Toc525241067)

[Project Fork 4](#_Toc525241068)

[Χρήση/ Manual 6](#_Toc525241069)

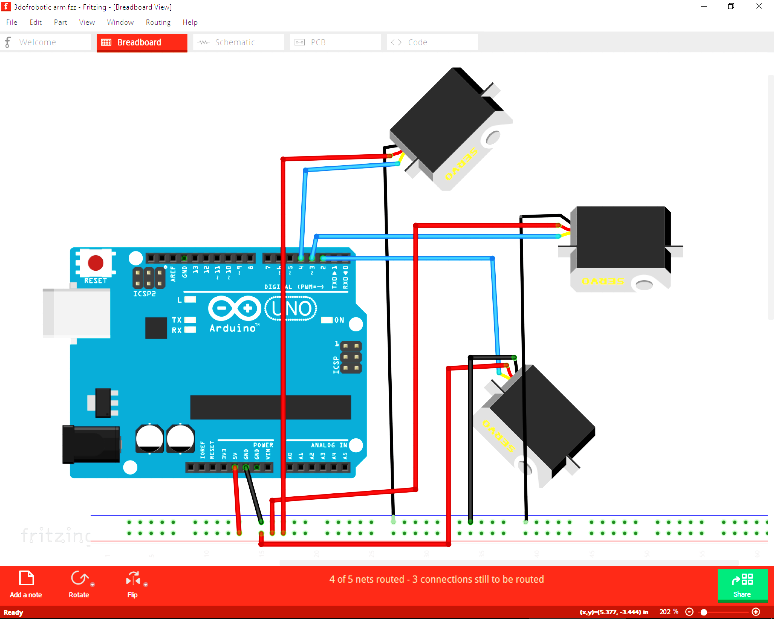
Christianidis Vasileios

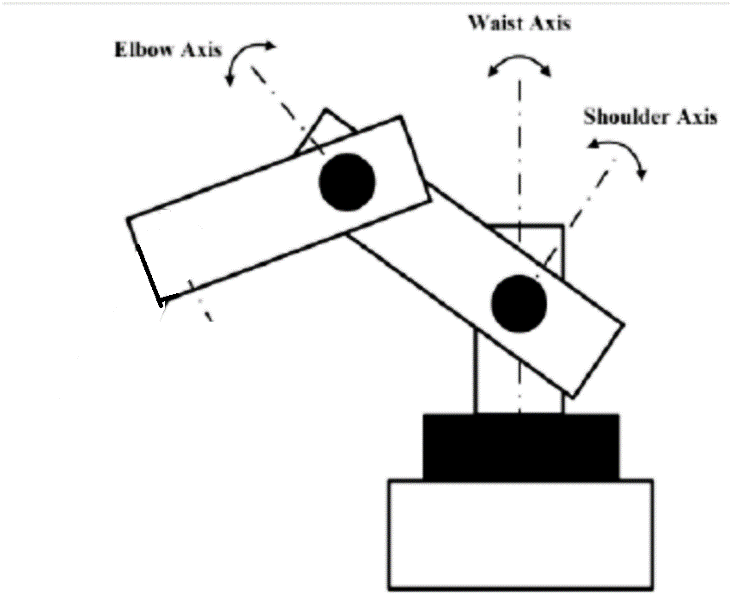
[basilisvirus@hotmail.com](mailto:basilisvirus@hotmail.com)

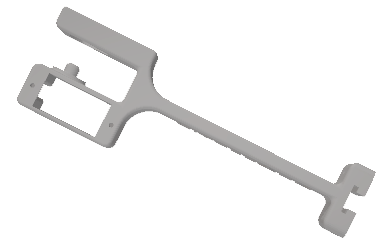
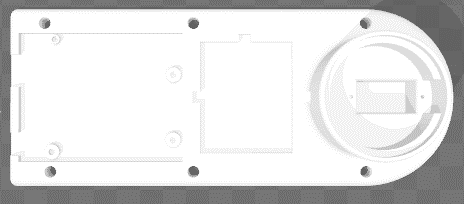
Η κατασκευή

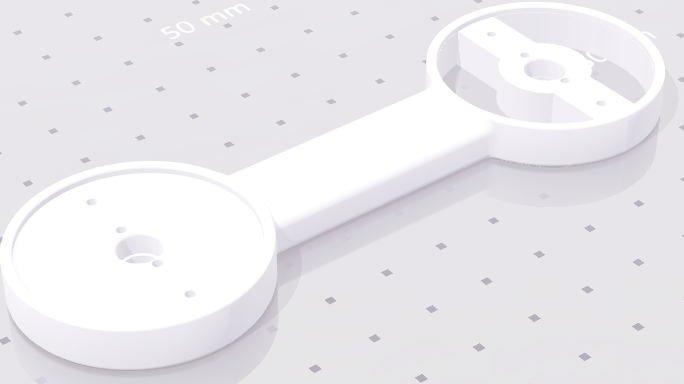
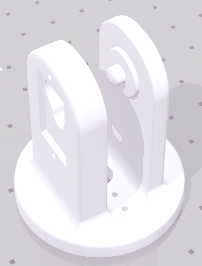
Source code

Ο πηγαίως κώδικας: <https://drive.google.com/open?id=1Mwbe595RzY48CXIqrH_VfTmWk9Hwk52V>

Hardware







Τα αρχεία για τα 3d parts μπορούν να βρεθούν εδώ: <https://drive.google.com/open?id=1dwAuN0cc92pcW_Hri_FHhAWubaCeS61t>

Τα οποία δε τα κατασκεύασα, αλλα τα βρήκα εδω: <http://3dprintedrobots.blogspot.com/p/project-links.html?m=0>

και ειναι compatible με servo μεγέθους: 9G SG90 και MG996R (micro servos)

καποια από τα 3d printed parts είναι 10 φορές μικρότερα από το κανονικό, οπότε θα χρειαστεί να γίνει αυξηση κατα 1000% αυτών των μελών.

Το αρχείο του fritzing το ανέβασα εδώ: <https://drive.google.com/open?id=1pe4eR5G9d9OuKV1X6T0yLQ4_ZlyApwpt>

Τα μαθηματικά πίσω από τη κατασκευή

Βρήκα ένα pdf σε αυτη τη σελίδα: <http://www.academia.edu/9165706/Forward_and_inverse_Kinematics_complete_solutions_3DOF_good_reference_for_CrustCrawler_Smart_Arm_Users_>

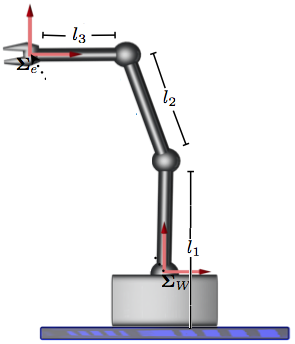
Το οποίο λύνει την αντιστροφη κινηματική για εναν 3DOF βραχίονα (3ΒΕ), στη σελίδα 36 (το s3 παιρνει και αρνητική τιμή για τη 2η λύση αν υπαρχει).

Project Fork

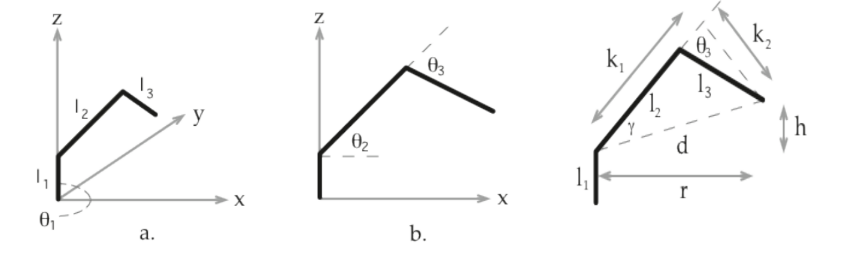
Σε περίπτωση που προσπαθησετε να αναπαργετε τη κατασκευή, θα πρεπει να αλλαξετε τον κωδικα ανάλογα με τον τρόπο που τοποθετήσατε τα servo σας, θα αλλάζουν οι οριακές τιμές των γωνιών και αλλα, οπως το μεγεθος των βραχιονων.

Λεπτομερή οι αλλαγές:

**---**Στη 16η γραμμή του κώδικα, τα l1, l2, l3 αντιστοιχούν στα παρακάτω μήκη:

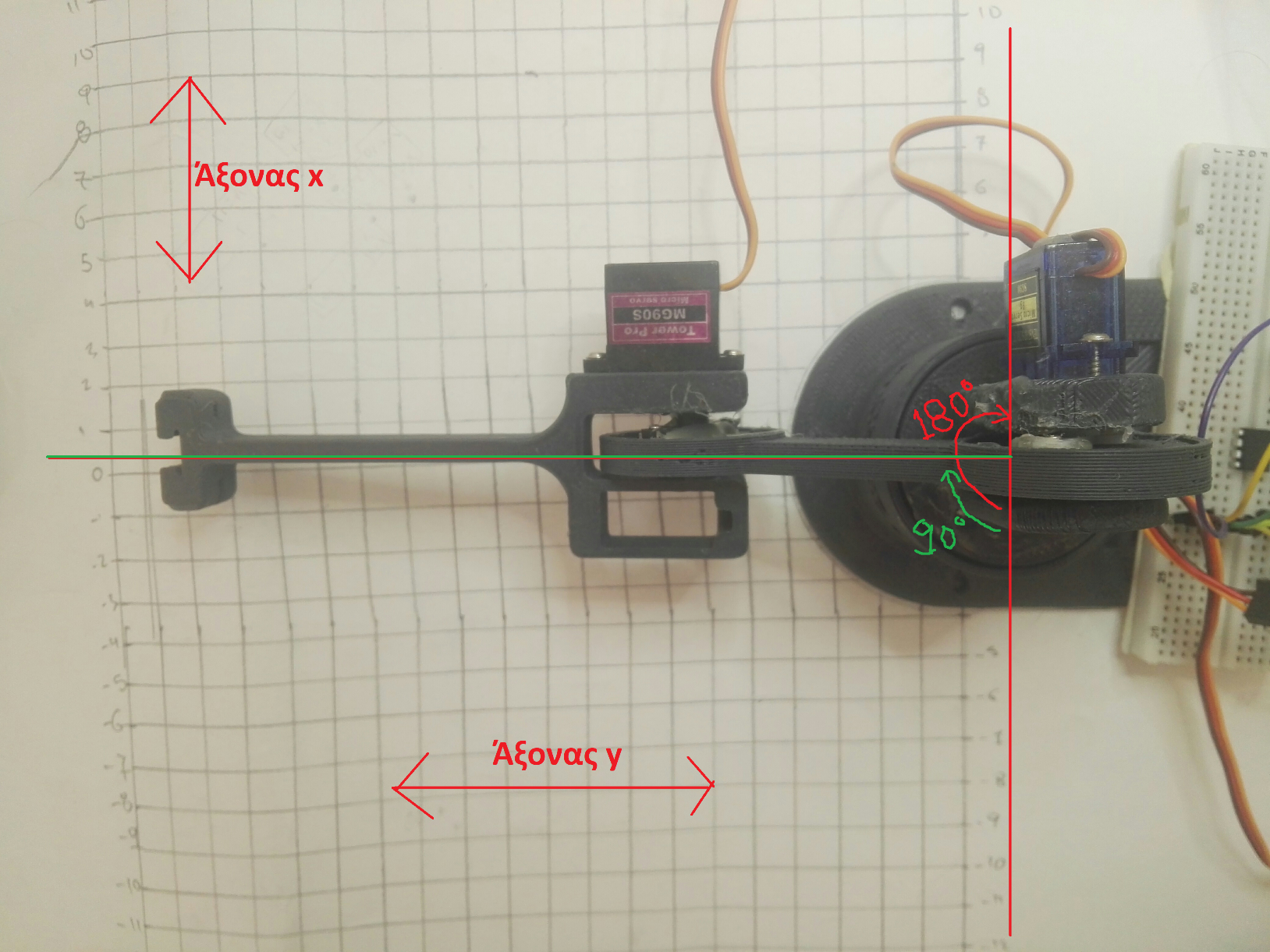


**---**Οι αρχικές γωνίες που αποκτάνε τα servo λόγω της βιβλιοθήκης servo.h είναι οι 90o (γωνίες servo). Αυτές τις γωνίες παίρνουν ολα τα servo τη στιγμή που χρησιμοποιείται η μέθοδος servo.attach(pin value) στην void setup() {}, σε οποιοδηποτε pin και αν γίνει το attach. Όμως, στη πραγματικότητα, το πρόγραμμά μας, επειδή χρησιμοποιεί ως γωνίες τις προκαθορισμένες γωνίες που έχουμε θέσει και χρησιμοποιούμε από τις μαθηματικές μας εξισώσεις, (βλεπε εικόνα παρακάτω),



Υπάρχει πιθανότητα κάποιες ή όλες οι θεωρητικές μας γωνίες να μην ανταποκρίνονται στις φυσικές μας γωνίες. Ως φυσικές γωνίες εννοούμε αυτές που πραγματικά θέλουμε και βλέπουμε στον χτισμένο βραχίονά μας και ανταποκρίνονται στη παραπάνω εικόνα. Ενώ ως θεωρητικές γωνίες, εννοούμε κάποιες άλλες γωνίες, οι οποίες χρησιμοποιεί το πρόγραμμά μας, οι οποίες συσχετίζονται με τις φυσικές μας γωνίες με μια εξίσωση, την οποία θα πρέπει και να βρούμε. Για αυτό θα χρησιμοποιήσουμε manual calibration.

Η τεχνική εδώ της χειροκίνητης βαθμονόμησης θα είναι πολύ πρόχειρη. Θα πρεπει να φτιαξετε ενα νεο προγραμμα στο Arduino ide, χωρις να αλλαξετε τη κατασκευη φυσικα, και χρησιμοποιώντας την έτοιμη βιβλιοθήκη Servo.h, να δίνετε δικές σας γωνίες σε μια μια ξεχωρηστά άρθρωση/servo και να δείτε αν αυτές οι γωνίες που δίνετε αντικατοπτρίζει/ανταποκρίνεται/είναι ίσες με τις φυσικές γωνίες, όχι μόνο στη τιμή αλλά και τη κατεύθυνση των γωνιών. τα θ2 και θ3 έχουν θετική κατεύθυνση προς τα επάνω (αριστερόστροφα), ενώ το θ1 μπορεί να έχει ειτε δεξιόστροφη είτε αριστερόστροφη κίνηση. Το θ1 δε θα χρειαστεί καθόλου ρύθμιση, εαν τοποθετήσουμε τη βάση του βραχίονα έτσι ώστε όταν το servo να δείχνει 90o οι βραχίονες να είναι στην ευθεία με τον y άξονα όπως παρακάτω:



Αν είμαστε ευχαριστημένοι με τη βάση, φτιάχνουμε την πρώτη μας εξίσωση, στη περίπτωσή μου:

Για τη Θ1:

TH1(Deg)= Deg

Για τη Θ2:

TH2(Deg)= (-Deg)+140

Για τη Θ3:

TH3(Deg)= Deg + 125

Στις γραμμές κώδικα 25, 29 και 33 θα βάλετε τις δικές σας συναρτήσεις.

**---**Η τελευταία αλλαγή πρέπει να γίνει στους περιορισμούς γωνιών. ο βραχίονάς μας δεν μπορεί να πάρει ολες τις διαθέσιμες γωνίες που του επιτρέπουν οι μαθηματικές εξισώσεις. Εδώ ευτυχώς δεν χρειάζεται να φτιάξουμε μια εξίσωση που να μετατρέπει τα φυσικά όρια στα μαθηματικά όρια, αλλά πάλι με δικό μας calibration όπως έκανα και φένεται στις παρακάτω εικόνες (αν ανοίξετε το link), να εκτιμήσουμε με οσο μεγαλύτερη ακρίβεια μπορούμε το range των γωνιών που θα είναι επιτρεπτό να δωθεί στον βραχίονα.

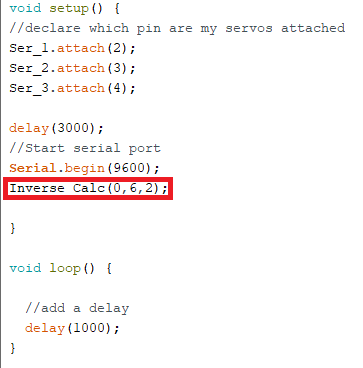
<https://drive.google.com/open?id=1MX1EK6R8p5shqi_fzJayERDFiYCHO3f8>

Συνολικά θα χρειαστούμε 4 όρια γωνιών. Τα όρια που θα βρείτε από τον παραπάνω τρόπο μέσω των εικόνων δεν χρειάζεται να τους αλλάξετε στις θεωρητικές τιμές, διότι όταν ζητηθεί στον βραχίονα να πάει σε ένα σημείο, πρώτα υπολογίζονται μόνο τα μαθηματικά κομμάτια (οι γωνίες που πρέπει να πάρει ο βραχίονας σε φυσικό επίπεδο) και ύστερα γίνεται η σύγκριση με τα όρια γωνιών. Και ύστερα χρησιμεύουν οι εξισώσεις που μετατρέπουν τις φυσικές γωνίες σε θεωρητικές, διοτι ύστερα μπαίνουν τα servo στην όλη κατάσταση. Για τον απλο υπολογισμό του που μπορει και που δε μπορει να πανε τα μελη του βραχίονα δεν χρειάζονται οι εξισώσεις μετατροπής. Έτσι, ότι βρούμε στα όρια μας από τις εικόνες, είναι και αυτό που θα βάλουμε.

Τα όρια γωνιών μπορούμε να τα αλλάξουμε στη γραμμή 38 έως 41 του κώδικα.

Χρήση/ Manual

Για να χρησιμοποιηθεί ο βραχίονας, θα πρέπει να κατεβάσετε το Arduino IDE και να συνδέσετε το hardware σύμφωνα με το schematic πιο πάνω (google drive link). Ύστερα, αφου ακολουθήσετε τις οδηγίες για το “project fork” ανοιξετε το αρχειο απο το φακελο 3Dof\_final στο link, θα παρατηρησετε στις τελευταιες γραμμες κωδικα:

Την Inverse\_Calc(). Για να λειτουργήσει ο βραχίονας, και να παει στο στόχο το άκρο εργασίας του, πρέπει να βάλετε τις συντεταγμένες x,y,z στην inverse calc με αυτη τη σειρα και να πατήσετε το compile & upload sketch στο Arduino.

Επίσης θα παρατηρήσετε οτι με το που φορτωσει ο κώδικας, θα κάνει μια απότομη κίνηση στις 90 μοίρες (προ-προγραμματισμού) και στα 3 σερβο). Και υστερα θα ξεκινήσει να κινείται προς το σημείο που ορίστηκε. Καλό είναι να αποφεύγεται εκείνη η απότομη κίνηση, βάζοντας χειροκίνητα τον βραχίονα στο αρχικό σημείο που θα κινόταν επότομα με το φόρτωμα του προγράμματος.

Μπορείτε να τρέξετε την Inverse\_Calc() παραπάνω από μία φορές σε κάθε upload, με αποτέλεσμα να πηγαίνει με τη σειρά στα σημεία που θέσαμε.

Σημείωση: το Y δε δέχεται αρνητικές τιμές\* και το Z μπορεί να δεχτεί αρνητικές τιμές.