ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΜΗΧΑΝΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ

ΤΟΜΕΑΣ ΜΗΧΑΝΟΛΟΓΙΚΩΝ ΚΑΤΑΣΚΕΥΩΝ ΚΑΙ ΑΥΤΟΜΑΤΟΥ ΕΛΕΓΧΟΥ

Διπλωματική εργασία με θέμα:

Υλοποίηση και Πειραματική Εξέταση Μεθοδολογιών Ελέγχου Δύναμης/Ροπής Ρομποτικού Βραχίονα

Αιμίλιος Κομπότης

Επιβλέπων Καθηγητής: Κωνσταντίνος Κυριακόπουλος

Αθήνα, Φεβρουάριος 2008

Ευχαριστίες

Η παρούσα διατριβή πραγματοποιήθηκε στα πλαίσια εκπόνησης της Διπλωματικής Εργασίας του γράφοντα στο Εργαστήριο Αυτομάτου Ελέγχου και Ρυθμίσεως Μηχανών της Σχολής Μηχανολόγων Μηχανικών του Εθνικού Μετσοβίου Πολυτεχνείου. Επιβλέπων Καθηγητής ήταν ο κ. Κωνσταντίνος Κυριακόπουλος, προς στον οποίο θα ήθελα να εκφράσω τις ευχαριστίες μου, τόσο για την εμπιστοσύνη που μου έδειξε αναθέτοντας μου την διεκπεραίωση της παρούσας εργασίας, όσο και για την άριστη συνεργασία μας για την επιτυχημένη ολοκλήρωσή της.

Το ευχάριστο κλίμα που δημιουργούσαν τα υπόλοιπα μέλη του εργαστηρίου, καθώς και η υποστήριξη στις δυσκολίες που προέκυψαν, ήταν πολύ σημαντικά για την ομαλή εξέλιξη της εργασίας αυτής. Ιδιαίτερη μνεία κρίνω απαραίτητο να κάνω στην Υποψήφια Διδάκτορα και άμεση συνεργάτιδά μου, κ. Ξανθή Παπαγεωργίου, για την πολύτιμη συμπαράστασή της καθόλη τη διάρκεια της παραμονής μου στο εργαστήριο, τόσο με ψυχολογικό όσο και με συμβουλευτικό χαρακτήρα, όπως και στον επίσης Υποψήφιο Διδάκτορα κ. Απόλλωνα Οικονομόπουλο, για την κρίσιμη βοήθειά του, μοιραζόμενος τις γνώσεις του σε κάποια προγραμματιστικά και υπολογιστικά προβλήματα που προέκυψαν.

Τέλος, θα ήθελα να ευχαριστήσω την οικογένειά μου, τους φίλους και τα κοντινά μου πρόσωπα που, με την αμέριστη υποστήριξη και εμπιστοσύνη που μου έδειξαν, συνέβαλαν αποφασιστικά στην επίτευξη αυτού του στόχου.

Πρόλογος

Τις τελευταίες τρεις δεκαετίας παρατηρείται ραγδαία ανάπτυξη στον τομέα του ελέγχου ρομπότ με χρήση μεθοδολογιών ελέγχου δύναμης και ροπής. Η ανάπτυξη αυτή συνοδεύτηκε, αλλά και προωθήθηκε, από την ολοένα και βελτιωμένη εξέλιξη στην τεχνολογία των αισθητήρων, σημαντικό εργαλείο για την άμεση παρακολούθηση του περιβάλλοντος και των μεταβαλλόμενων συνθηκών του.

Η αλληλεπίδραση με το περιβάλλον, το οποίο σε πολλές περιπτώσεις είναι εξ ολοκλήρου ή εν μέρει άγνωστο, οδήγησε στη μελέτη των μεθοδολογιών αυτών. Έτσι, η χρήση και λειτουργία των ρομπότ θα μπορούσε να ξεφύγει από την κλασική εφαρμογή βιομηχανικού ρομπότ σε ένα άριστα προβλέψιμο και δεδομένο περιβάλλον. Οι στρατηγικές ελέγχου κίνησης σε ένα τέτοιο περιβάλλον μπορεί να έχει αποτελεσματικότητα σε τέτοιους χώρους και σε συγκεκριμένες διεργασίες, όμως και η παραμικρή δύναμη επαφής θα μπορούσε να οδηγήσει σε αβεβαιότητα και σε ασταθή λειτουργία οποιονδήποτε βραχίονα. Εκτός από τη βασική προϋπόθεση της εξασφάλισης της ορθής λειτουργίας ενός ρομπότ εξαλείφοντας την πιθανότητα κάποιου ατυχήματος ή βλάβης, μια στρατηγική ελέγχου δύναμης μπορεί να επιφέρει και καλύτερα αποτελέσματα εκμεταλλευόμενη τις δυνάμεις αυτές.

Σε ένα μη μοντελοποιημένο περιβάλλον ο έλεγχος μέσω δύναμης καλείται να αντιμετωπίσει τις δυσκολίες του ελέγχου κίνησης λαμβάνοντας υπόψη την ανάδραση των δυνάμεων επαφής που προκύπτουν μεταξύ του περιβάλλοντος και του ρομποτικού βραχίονα. Στη συνέχεια, με βάση αυτές τις δυνάμεις, έχει τη δυνατότητα να αναπροσαρμόσει την τροχιά του για την πραγματοποίηση της εργασίας που κλήθηκε να επιτελέσει. Η μέτρηση των δυνάμεων και ροπών επαφής πραγματοποιείται σε πραγματικό χρόνο με χρήση αισθητήρων δύναμης/ροπής, είτε τοποθετημένων στον ακροδέκτη του ρομποτικού βραχίονα ή κατάλληλα προσαρμοσμένων στο σώμα του. Η εφαρμογή ενός νόμου ελέγχου δύναμης μπορεί να πραγματοποιείται παράλληλα με κάποιον έλεγχο κίνησης ώστε να μπορούμε να εκμεταλλευτούμε τα πλεονεκτήματα και των δύο μεθόδων.

Τέλος, με την ανάπτυξη της νευρορομποτικής (neurobotics) και της τηλεδιαχείρισης (telemanipulation), η εμπειρία που αποκομίζεται από την εφαρμογή μεθοδολογιών ελέγχου δύναμης είναι μεγάλης σημασίας, καθώς στις περισσότερες περιπτώσεις ο χρήστης καλείται να επιτελέσει διεργασίες στις οποίες είναι απαραίτητη η χρήση και ανάδραση της δύναμης που εφαρμόζεται, υποβοηθούμενος από ένα ρομποτικό σύστημα.

Περιεχόμενα:

Ευχαριστίες *1*
Πρόλογος *4*
Περιεχόμενα: *5*
ΚΕΦΑΛΑΙΟ 1ο : Εισαγωγικά *8*
1.1 Διατύπωση του Προβλήματος **8**
1.2 Σημασία του Προβλήματος **8**
1.3 Προηγούμενη δουλειά και αντίστοιχες Μελέτες **8**
1.4 Δομή Εργασίας **9**
ΚΕΦΑΛΑΙΟ 2ο : Ανάλυση Μεθοδολογιών Ελέγχου Δύναμης και Ροπής *11*
Εισαγωγή: **11**
2.1 Αλληλεπίδραση ρομποτικού βραχίονα με το περιβάλλον **12**
2.2 Έλεγχος Συμμόρφωσης (compliance control) **14**
2.3 Έλεγχος Εμπέδησης (impedance control) **17**
2.4 Έλεγχος Δύναμης **20**
Εισαγωγή: 20
2.41 Έλεγχος Δύναμης με εσωτερικό βρόχο Θέσης (inner position loop) 20
2.42 Έλεγχος Δύναμης με εσωτερικό βρόχο Ταχύτητας (inner velocity loop) 22
2.43 Παράλληλος έλεγχος Δύναμης/Θέσης 23
2.5 Φυσικοί και Τεχνητοί περιορισμοί **25**
ΚΕΦΑΛΑΙΟ 3ο : Περιγραφή Εγκατάστασης *27*
Εισαγωγή: **27**
3.1 Ρομποτικός βραχίονας PA-10 της Mitsubishi **28**
Εισαγωγη: 28
3.11 Ο ελεγκτής του PA-10 29
3.12 Το πρωτόκολλο επικοινωνίας του PA-10 30
3.2 JR3 αισθητήρας δύναμης/ροπής **32**
Εισαγωγη: 32
3.21 Αισθητήρας JR3 με ενσωματωμένα ηλεκτρονικά (onboard electronics) 33
3.22 Επεξεργασία Ψηφιακού Σήματος- ΕΨΣ (Digital Signal Processing - DSP) 34

3.23 Δέκτης διαύλου PCI 35
3.24 Σύνοψη της αρχιτεκτονικής δομής δέκτη JR3 με ΕΨΣ 36

3.3 Σύνδεση αισθητήρα JR3 με τον ακροδέκτη του βραχίονα PA-10 40

Εισαγωγή: 40
3.31 Σχεδιασμός φλάντζας σύνδεσης και επιλογή υλικού 41
3.32 Θεωρητικός έλεγχος αντοχής με χρήση πεπερασμένων στοιχείων 44
3.33 Κατασκευή και επεξεργασία αλουμινένιας φλάντζας 47
3.34 Ολοκλήρωση σύνδεσης πειραματικού εξοπλισμού 50

3.4 : Βαθμονόμηση (calibration) του αισθητήρα JR3 δύναμης/ροπής 51

Εισαγωγή: 51
3.41 Συλλογή τιμών δύναμης και ροπής μέσω του αισθητήρα με χρήση πρότυπων βαριδίων 52
3.42 Διερεύνηση θορύβου κατά τη συλλογή τιμών του αισθητήρα 56
3.43 Μετρήσεις τιμών δυνάμεων και ροπών και απόκλιση από τις πραγματικές 62
3.44 Μεθοδολογίες βαθμονόμησης και κριτήρια αξιολόγησης 65
3.441 Εφαρμογή πολυωνυμικής συνάρτησης προσέγγισης και μειονεκτήματα 67
3.442 Εφαρμογή τμηματικών συνεχών πολυωνύμων και μειονεκτήματα 70
3.443 Θεωρία παρεμβολής μέσω κυβικών splines 73
3.444 Εφαρμογή παρεμβολής με χρήση κυβικών splines και διορθώσεις 77
3.45 Αξιολόγηση βαθμονόμησης με χρήση κυβικών splines και έλεγχος ακριβείας 81
3.46 Έλεγχος και διόρθωση μηδενικής αντιστάθμισης (zero offset) 82

ΚΕΦΑΛΑΙΟ 4ο : Ολοκλήρωση και Πειράματα 87

Εισαγωγή: 87

4.1 Περιγραφή Πειράματος 88

4.2 Προετοιμασία πειραματικής διάταξης 89

4.21 Θεωρητική επισκόπηση 89
4.22 Αναγωγή δυνάμεων στο τελικό σημείο δράσης 91
4.23 Λογισμικό εφαρμογής ελέγχου δύναμης 93
4.24 Χρονομέτρηση κύκλου λειτουργίας ελέγχου δύναμης 94

4.3 Εκτέλεση Πειράματος 97

4.4 Αποτελέσματα Πειράματος 99

ΚΕΦΑΛΑΙΟ 5ο : Αξιολόγηση - Μελλοντικές Κατευθύνσεις 106

5.1 Αξιολόγηση Πειραματικών αποτελεσμάτων 106

5.2 Κατευθύνσεις και Προτάσεις για περαιτέρω ανάπτυξη 107

ΠΑΡΑΡΤΗΜΑ *108*

Α. Βοηθητικό λογισμικό, Πίνακες και Διαγράμματα **109**

Α.1 Ρουτίνες παρουσίασης τιμών φορτίσεων του αισθητήρα 109

Α.2 Συγκεντρωτικοί Πίνακες μετρήσεων τιμών δυνάμεων και ροπών πριν τη βαθμονόμηση του αισθητήρα 111

2.1 Δυνάμεις Fx, Fy, Fz 111

2.2 Ροπές Mx, My, Mz 115

Α.3 Ενδεικτική εφαρμογή τμηματικών συνεχών πολυωνύμων σε συμπληρωματικά πεδία ορισμού για τις φορτίσεις Fx και Fz 118

3.1 Τμηματικά συνεχή πολυώνυμα για την Fx (N) 118

3.2 Τμηματικά συνεχή πολυώνυμα για την Fy (N) 119

Α.4 Ρουτίνες υπολογισμού συναρτήσεων κυβικών splines 120

4.1 Spline.c - ρουτίνα εύρεσης συντελεστών M_i 120

4.2 Splint.c - ρουτίνα υπολογισμού y για κάθε τιμή του x 122

Α.5 Διαγράμματα προσομοίωσης σφάλματος με χρήση συναρτήσεων κυβικών splines 123

5.1 Fx, Fy, Fz Δυνάμεις 123

5.2 Mx, My, Mz Ροπές 124

Β. Περιγραφή και Ανάλυση λογισμικού **126**

Β.1 Δομή λογισμικού 126

Β.2 Ανάλυση κώδικα 127

2.1 JR3_force.c 127

2.2 ForceControl.cpp (για 7 βαθμούς ελευθερίας) 137

2.3 Fcontrol2dof.cpp (για 2 βαθμούς ελευθερίας) 156

2.4 PA-10.c 178

Γ. Εγχειρίδιο **180**

Γ.1 Εγχειρίδιο λογισμικού 180

Γ.2 Μετάφραση και χρήση λογισμικού ελέγχου σε Linux 188

Γ.3 Σύνδεση εξαρτημάτων και πραγματοποίηση πειραμάτων 190

ΕΥΡΕΤΗΡΙΟ *195*

Πίνακες: 195

Διαγράμματα: 195

Εικόνες: 196

Σχήματα: 197

ΒΙΒΛΙΟΓΡΑΦΙΑ *199*

ΚΕΦΑΛΑΙΟ 1ο : Εισαγωγικά

1.1 Διατύπωση του Προβλήματος

Η παρούσα εργασία αφορά στην ανάπτυξη και εφαρμογή μεθοδολογιών ελέγχου δύναμης και ροπής στο ρομποτικό βραχίονα 7 βαθμών ελευθερίας PA-10 της Mitsubishi Heavy Industries με χρήση του αισθητήρα δύναμης/ροπής 6 βαθμών ελευθερίας της JR3. Συγκεκριμένα, ζητείται η κατασκευή της απαραίτητης φλάντζας για την προσαρμογή του αισθητήρα στο ρομποτικό βραχίονα, η ανάπτυξη λογισμικού για την επικοινωνία και λήψη τιμών δύναμης και ροπής από αυτόν, η βαθμονόμησή του και η σωστή μηδενική αντιστάθμιση σε όλους τους άξονες μέτρησης και η ανάπτυξη του θεωρητικού μοντέλου και του αντίστοιχου λογισμικού ελέγχου για την πραγματοποίηση ελέγχου μέσω δύναμης.

Απώτερος σκοπός είναι η πραγματοποίηση πειραμάτων ελέγχου δύναμης που θα μπορούν να χρησιμοποιηθούν ως βάση για την περαιτέρω ανάπτυξη και εξέλιξη των υπαρχουσών μεθοδολογιών.

1.2 Σημασία του Προβλήματος

Η επιτυχής πραγματοποίηση πειραμάτων με χρήση μεθοδολογιών ελέγχου δύναμης/ροπής μάς δίνει τη δυνατότητα αξιολόγησης και επαλήθευσης των μεθοδολογιών αυτών σε σχέση με τα προβλήματα που καλούνται να επιλύσουν. Επίσης, σημαντικό είναι το γεγονός πως ανοίγονται νέοι δρόμοι για την πραγματοποίηση πιο σύνθετων εφαρμογών ελέγχου που απαιτούν ανάδραση δύναμης. Έτσι το ρομπότ μας από χειριστήριος βραχίονας μπορεί να γίνει πλέον επιδέξιος βραχίονας σε συνδυασμό και με την ευελιξία που του προσφέρουν οι 7 βαθμοί ελευθερίας.

1.3 Προηγούμενη δουλειά και αντίστοιχες Μελέτες

Η πειραματική εξέταση μεθοδολογιών ελέγχου δύναμης/ροπής που αναλύεται στην παρούσα εργασία είναι κυρίως εμπνευσμένη από τη δημοσίευση *"Towards Recognition of Control Variables for an Exoskeleton"* , (βιβλιογραφική αναφορά [7]).

Στη δημοσίευση αυτή προσεγγίζεται το ζήτημα της σύνδεσης μιας εξωσκελετικής διάταξης με το άνω άκρο ενός ανθρώπου για την υποστήριξή του και τη βελτίωση της λειτουργίας του. Η χρήση ενός τέτοιου εξωσκελετού συνδεδεμένου με ένα ανθρώπινο χέρι απαιτεί το σχεδιασμό ενός ελέγχου, σύμφωνα με τον οποίο η πρόθεση για κίνηση από τον άνθρωπο ανιχνεύεται από το κατάλληλο μηχανοτρονικό σύστημα και οι κινητήρες του

εξωσκελετικού ρομποτικού βραχίονα εφαρμόζουν την κατάλληλη ροπή, ώστε ο εξωσκελετός να υποβοηθά ή να ενισχύει την κίνηση του χεριού. Στη συνέχεια, και έπειτα από τη μοντελοποίηση του συστήματος, γίνεται προσομοίωση εφαρμογής διαφόρων μεθοδολογιών ελέγχου δύναμης/ροπής για τη συνεργασία του ζεύγους ανθρώπινο χέρι-τεχνητό χέρι. Στην παρούσα εργασία γίνεται προσπάθεια πειραματικής επαλήθευσης και βελτίωσης των μεθοδολογιών αυτών με τη χρήση ενός ρομποτικού βραχίονα με έναν αισθητήρα δύναμης/ροπής συνδεδεμένο στον ακροδέκτη του και πραγματοποίηση ελέγχου σε πραγματικό χρόνο.

Επίσης, επιτυχημένα πειράματα έμμεσου ελέγχου δύναμης με χρήση παρόμοιου αισθητήρα με αυτόν της δικής μας εφαρμογής παρουσιάζονται στα "*Force/Torque Sensing Applied to Industrial Robotic Deburring*" και "*Force control experiments for industrial applications: a test case using an industrial deburring example* ", (βιβλιογραφική αναφορά [8] και [9] αντίστοιχα). Τέλος, ενδιαφέρουσα μελέτη παράλληλου ελέγχου δύναμης και θέσης με χρήση του ίδιου ρομποτικού βραχίονα με αυτόν της εφαρμογής μας (PA-10 7 βαθμών ελευθερίας) παρουσιάζεται στο "*Unified Force and Motion Control Using an Open System Real-Time Architecture on a 7 DOF PA-10 Robot* " (βιβλιογραφική αναφορά [10]).

1.4 Δομή Εργασίας

- **Κεφάλαιο 1**

  Πραγματοποιείται μια σύντομη εισαγωγική ανάπτυξη του προβλήματος που καλείται να επιλύσει η παρούσα εργασία

- **Κεφάλαιο 2**

  Αναπτύσσονται διεξοδικά σε θεωρητικό και μαθηματικό επίπεδο οι κυριότερες μεθοδολογίες ελέγχου δύναμης και ροπής τις οποίες καλούμαστε να εξετάσουμε πειραματικά.

- **Κεφάλαιο 3**

  Περιγράφονται σε συντομία ο ρομποτικός βραχίονας και ο αισθητήρας δύναμης/ροπής που διαθέτουμε και αναλύονται λεπτομερώς η κατασκευή της φλάντζας για τη σύνδεσή τους, όπως και όλα τα βήματα και οι μεθοδολογίες που ακολουθήσαμε για τη βαθμονόμηση του αισθητήρα και τη σωστή μηδενική αντιστάθμισή του.

- **Κεφάλαιο 4**

  Αρχικά καταγράφεται η απαραίτητη προετοιμασία που πραγματοποιήσαμε για τη διεξαγωγή πειραμάτων ελέγχου δύναμης/ροπής και στη συνέχεια παρουσιάζεται πλήρως η πειραματική διαδικασία και τα αποτελέσματα που προέκυψαν από αυτή.

- **Κεφάλαιο 5**

  Έπειτα από αξιολόγηση των πειραματικών αποτελεσμάτων παρουσιάζονται προτάσεις για μελλοντικά πειράματα και κατευθύνσεις της έρευνας.

- **Παράρτημα**

  - **Α.** Συγκεντρώνονται οι βοηθητικοί πίνακες και διαγράμματα, καθώς και το λογισμικό που αναπτύχθηκε για την επίλυση επιμέρους προβλημάτων

  - **Β.** Περιγράφεται η δομή του λογισμικού που χρησιμοποιήθηκε για τις πειραματικές εφαρμογές και γίνεται εκτενής ανάλυση όλων των εντολών των κυρίως προγραμμάτων ελέγχου και του προγράμματος επικοινωνίας και επεξεργασίας τιμών δύναμης/ροπής μέσω του αισθητήρα

  - **Γ.** Εγχειρίδιο λογισμικού με όλα τα βήματα που χρειάζονται για τη χρήση του κεντρικού προγράμματος ελέγχου και των βοηθητικών υποπρογραμμάτων, για τη μετάφραση και λειτουργία αυτών σε λειτουργικό σύστημα Linux και για την ορθή εγκατάσταση και σύνδεση του αισθητήρα με το ρομποτικό βραχίονα.

ΚΕΦΑΛΑΙΟ 2ο : Ανάλυση Μεθοδολογιών Ελέγχου Δύναμης και Ροπής

Εισαγωγή:

Μία από τις θεμελιώδεις προϋποθέσεις για την επιτυχία μιας εφαρμογής ενός ρομποτικού βραχίονα είναι η δυνατότητά του να μπορεί να χειρίζεται επιτυχώς την αλληλεπίδρασή του με το περιβάλλον. Η ποσότητα που περιγράφει αρκετά αποτελεσματικά την κατάσταση της αλληλεπίδρασης του ρομποτικού βραχίονα με το περιβάλλον είναι η δύναμη επαφής (contact force), που εφαρμόζεται στον ακροδέκτη του (end effector).
Υψηλές τιμές της δύναμης επαφής είναι γενικότερα ανεπιθύμητες, καθώς μπορούν να προκαλέσουν παραμορφώσεις, τόσο στο χειριστήριο ρομποτικό βραχίονα όσο και στο αντικείμενο που θέλουμε να χειριστούμε.

Σκόπιμη είναι η ανάλυση των αρχών της μηχανικής συμμόρφωσης (compliance) και εμπέδησης (impedance), με κύριο σκοπό την ενσωμάτωση των μετρήσεων της δύναμης επαφής στη στρατηγική ελέγχου. Στη συνέχεια παρουσιάζονται μεθοδολογίες ελέγχου δύναμης/ροπής που προκύπτουν από αντίστοιχες μεθοδολογίες ελέγχου κίνησης κατάλληλα διαμορφωμένες για την ολοκλήρωση ενός βρόχου ανάδρασης καθορισμένου από τις εξωτερικές δυνάμεις (closure of an outer force regulation feedback loop).

Τέλος, γίνεται μια σύντομη ανάλυση των φυσικών περιορισμών, που διαμορφώνονται από τη γεωμετρία της εφαρμογής, και των τεχνητών περιορισμών, που καθορίζονται από τη στρατηγική ελέγχου, με σκοπό τον πλήρη σχεδιασμό του ελέγχου σε εφαρμογή που παρουσιάζονται φυσικές επαφές και αλληλεπιδράσεις.

2.1 Αλληλεπίδραση ρομποτικού βραχίονα με το περιβάλλον

Η εκτέλεση μιας εφαρμογής χειρισμού (manipulation task) συχνά απαιτεί την αλληλεπίδραση μεταξύ του χειριστήριου οργάνου, δηλαδή του ρομποτικού μας βραχίονα, με το περιβάλλον. Μια ολοκληρωμένη ταξινόμηση των πιθανών εφαρμογών χειρισμού θα ήταν πρακτικώς δύσκολη και χρονοβόρα, χάρη στη μεγάλη ποικιλία που τη διακρίνει και μια τέτοια ταξινόμηση δεν θα μπορούσε να μας βοηθήσει περισσότερο στην εύρεση μιας γενικής στρατηγικής για τον έλεγχο της αλληλεπίδρασης. Τυπικές εφαρμογές τέτοιου τύπου είναι η συναρμολόγηση μηχανικών εξαρτημάτων, η ανίχνευση του περιγράμματος αντικειμένων και η χρησιμοποίηση εργαλείων για εφαρμογή σε μηχανουργικές κατεργασίες.

Κατά την αλληλεπίδραση με αυτό, το περιβάλλον θέτει περιορισμούς στις γεωμετρικές τροχιές που ακολουθούνται από τον ακροδέκτη του βραχίονα. Αυτή η κατάσταση γενικά αναφέρεται ως περιορισμένη κίνηση (constrained motion).

Η χρήση μιας καθαρά στρατηγικής ελέγχου κίνησης για τον έλεγχο αλληλεπίδρασης απαιτεί σχεδιασμό υψηλής ακρίβειας όσον αφορά στο σχεδιασμό της τροχιάς του ακροδέκτη του βραχίονα. Επίσης, το σύστημα ελέγχου πρέπει να εξασφαλίζει πως η θέση του ακροδέκτη αποκλίνει όσο το δυνατόν λιγότερο από την επιθυμητή, σε όλο το μήκος της δεδομένης τροχιάς.

Συνεπώς, η επιτυχία μιας εφαρμογής αλληλεπίδρασης με το περιβάλλον με χρήση αλγορίθμων ελέγχου κίνησης εξαρτάται εξ ολοκλήρου από την ακρίβεια του σχεδιασμού και την επίδοση του ελέγχου. Για αυτό το στόχο είναι κρίσιμη η κατοχή ενός πολύ λεπτομερούς μοντέλου, τόσο για το χειριστήριο βραχίονα (κινηματικά και δυναμικά μοντέλα), όσο και για το ίδιο το περιβάλλον (μηχανικά χαρακτηριστικά και γεωμετρία). Το μοντέλο του χειριστήριου βραχίονα μπορούμε να το γνωρίζουμε με μεγάλη ακρίβεια, αλλά είναι δύσκολο, τις περισσότερες φορές, να αποκτήσουμε μια ολοκληρωμένη και λεπτομερή περιγραφή του περιβάλλοντος. Είναι αναπόφευκτο να εμφανίζονται σφάλματα στο σχεδιασμό μιας τροχιάς που προσδιορίζεται για τον ακροδέκτη του βραχίονα, που δεν είναι πλέον κατάλληλη για την ορθή εκπλήρωση και ολοκλήρωση της αρχικής εφαρμογής.

Ένας άλλος παράγοντας που επηρεάζει την αποτελεσματικότητα της προσέγγισης ενός προβλήματος ελέγχου περιορισμένης κίνησης με χρήση μεθόδων ελέγχου κίνησης είναι η ακρίβεια θέσης του ακροδέκτη του βραχίονα σε σχέση με το περιβάλλον. Στην πράξη υπάρχει ένα κάτω όριο για την ακρίβεια αναπαραγωγής της προδιαγραφόμενης τροχιάς κατά την εκτέλεση μιας εφαρμογής, εξαιτίας τόσο του σφάλματος θέσης του χειριστήριου βραχίονα, όσο και της ακριβούς θέσης του περιβάλλοντος.

Η ύπαρξη σφαλμάτων μοντελοποίησης και πεπερασμένης ακρίβειας θέσης επηρεάζει τον πλήρη προσδιορισμό της απόλυτης θέσης χειριστήριου βραχίονα και περιβάλλοντος. Μια άμεση συνέπεια αυτού του γεγονότος είναι η αδυναμία να γνωρίζουμε επακριβώς και τη

σχετική τους θέση. Για την κατανόηση της σημασίας των επιπτώσεων αυτής της διαπίστωσης είναι αρκετό να παρατηρήσουμε πως για την εκτέλεση μιας απλής εφαρμογής συναρμολόγησης με προσέγγιση θέσης, η σχετική θέση μεταξύ των εξαρτημάτων πρέπει να είναι συνεχώς γνωστή με ακρίβεια μιας τάξης μεγέθους μεγαλύτερης από τις κατασκευαστικές ανοχές του κάθε εξαρτήματος ξεχωριστά. Από τη στιγμή που η απόλυτη θέση ενός εξαρτήματος είναι γνωστή επακριβώς, ο χειριστήριος βραχίονας πρέπει να καθοδηγεί το άλλο εξάρτημα με την ίδια ακρίβεια.

Η ενυπάρχουσα δυσκολία για τον ακριβή σχεδιασμό και έλεγχο κίνησης ενός βραχίονα αναγκαστικά προβάλλει το ζήτημα της ανάλυσης των επιπτώσεων της απόκλισης της τροχιάς κάτω από ιδεατές συνθήκες αλληλεπίδρασης με το περιβάλλον.

Όταν ο βραχίονας κατευθύνεται μέσω αλγορίθμων ελέγχου θέσης, κάθε απόκλιση από την ακριβή τροχιά σε σχέση με την αναφορική προκαλεί αντίδραση του συστήματος ελέγχου. Αυτή η διαδικασία τείνει να ελαχιστοποιεί αυτή την απόκλιση ανεξάρτητα από την αιτία που την προκάλεσε. Για αυτό το λόγο, εάν η απόκλιση της σχεδιασμένης τροχιάς είναι αποτέλεσμα της αλληλεπίδρασης του βραχίονα με το περιβάλλον, προκύπτουν δυνάμεις αντίδρασης και ο έλεγχος θέσης προσπαθεί να μειώσει την απόκλιση, όπως θα έκανε για οποιαδήποτε διαταραχή που θα ήταν αντιτιθέμενη στην κίνηση του ακροδέκτη. Σε αυτή την περίπτωση όμως το αποτέλεσμα του ελέγχου μπορεί να είναι η αύξηση της δύναμης επαφής, κάτι το οποίο δεν συνοδεύεται με τη μείωση της απόκλισης. Αυτή η κατάσταση μπορεί να οδηγήσει στην αύξηση της δύναμης επαφής μέχρι να ληφθεί υπόψη το φυσικό όριο που καθορίζεται από τον κορεσμό (saturation) του μεταλλάκτη ή να έχουμε μηχανική καταστροφή ενός από τα εξαρτήματα που συμμετέχουν στην αλληλεπίδραση.

Όσο μεγαλύτερη είναι η δυσκαμψία του περιβάλλοντος και η ακρίβεια του ελέγχου θέσης, τόσο ευκολότερα μπορεί να προκύψει μια ασταθής συνθήκη επαφής, όπως αυτή που μόλις περιγράφτηκε. Στην ουσία, μεγάλες εξαναγκασμένες δυνάμεις αντίδρασης είναι το αποτέλεσμα παραμόρφωσης ενός δύσκαμπτου περιβάλλοντος υπό την εφαρμογή ισχυρού ελέγχου θέσης.

Η μέτρηση της κατάστασης της αλληλεπίδρασης προφανώς παρέχεται από τα χαρακτηριστικά της δύναμης επαφής μεταξύ του βραχίονα και του περιβάλλοντος. Για τον κατάλληλο χειρισμό της αλληλεπίδρασης είναι απαραίτητο να χρησιμοποιήσουμε στρατηγικές ελέγχου που μας επιτρέπουν να λάβουμε υπόψη τους περιορισμούς που επιβάλλονται από τις δυνάμεις αλληλεπίδρασης, είτε μέσω ενός έμμεσου τρόπου, με την κατάλληλη χρήση νόμων ελέγχου θέσης, ή με άμεσο τρόπο, μέσω νόμων ελέγχου δύναμης.

2.2 Έλεγχος Συμμόρφωσης (compliance control)

Για μία λεπτομερή ανάλυση της αλληλεπίδρασης μεταξύ ενός ρομποτικού βραχίονα και του περιβάλλοντος αξίζει να μελετήσουμε τη συμπεριφορά ενός συστήματος που καθοδηγείται από μια μεθοδολογία ελέγχου θέσης, όταν εμφανίζονται δυνάμεις επαφής.
Από τη στιγμή που αυτές οι μεθοδολογίες περιγράφονται στο λειτουργικό χώρο, είναι βολικό να αναφερόμαστε σε αυτές σαν μεθοδολογίες ελέγχου λειτουργικού χώρου (operational space control schemes).

Θεωρούμε το δυναμικό μοντέλο:

$$B(q)\ddot{q} + C(q,\dot{q})\dot{q} + F\dot{q} + g(q) = u - J^T(q)h \qquad (2.1)$$

όπου το h είναι το διάνυσμα των δυνάμεων επαφής που ασκούνται στον ακροδέκτη του ρομποτικού βραχίονα από το περιβάλλον.

Είναι σαφές πως στην περίπτωση που έχουμε $h \neq 0$, η μεθοδολογία ελέγχου που βασίζεται στη σχέση νόμου ελέγχου $u = g(q) + J_A^T(q)K_P\tilde{x} - J_A^T K_D J_A(q)\dot{q}$ (Έλεγχος PD με αντιστάθμιση βαρύτητας) δεν εξασφαλίζει ότι ο ακροδέκτης του βραχίονα θα βρεθεί στην επιθυμητή θέση x_d. Στην πραγματικότητα, εφόσον όπως γνωρίζουμε $\tilde{x} = x_d - x$, στη θέση ισορροπίας έχουμε:

$$J_A^T(q)K_p\tilde{x} = J^T(q)h \qquad (2.2)$$

και με την υπόθεση πως έχουμε Ιακωβιανό μητρώο πλήρους βαθμού (full-rank Jacobian) έχουμε:

$$\tilde{x} = K_P^{-1}T_A^T(x)h = K_P^{-1}h_A \qquad (2.3)$$

όπου το h_A είναι το διάνυσμα των ισοδύναμων γενικευμένων δυνάμεων. Η σχέση (2.3) δείχνει πως ο ρομποτικός βραχίονας στη φάση ισορροπίας, υπό την επίδραση νόμου ελέγχου θέσης, συμπεριφέρεται ως ένα γενικευμένο ελατήριο στο λειτουργικό χώρο (operational space) με συμμόρφωση K_P^{-1} σε σχέση με τη δύναμη h_A. Υποθέτοντας ότι το μητρώο K_P είναι διαγώνιο, παρατηρείται πως η γραμμική συμμόρφωση (εξαιτίας των συνιστωσών των δυνάμεων) είναι ανεξάρτητη της θέσης. Αντιθέτως, η συμμόρφωση στρέψης (εξαιτίας των συνιστωσών των ροπών) εξαρτάται από τη συγκεκριμένη κατάσταση και διαμόρφωση του ρομποτικού βραχίονα μέσω του μητρώου T_A.

Σε μια άλλη περίπτωση όπου $h \in N(J^T)$ έχουμε $\ddot{\tilde{x}} = 0$ με $h \neq 0$, όταν για παράδειγμα οι δυνάμεις επαφής εξισορροπούνται πλήρως από τη μηχανολογική δομή του ρομποτικού βραχίονα.

Για μία βαθύτερη κατανόηση της αλληλεπίδρασης μεταξύ ενός ρομποτικού βραχίονα και του περιβάλλοντος είναι απαραίτητο να έχουμε μια αναλυτική περιγραφή των δυνάμεων επαφής. Μια πραγματική επαφή είναι ένα φυσικά κατανεμημένο φαινόμενο, στο οποίο εμπλέκονται τόσο τα τοπικά χαρακτηριστικά του ρομποτικού βραχίονα, όσο και του περιβάλλοντος. Επιπροσθέτως, φαινόμενα τριβής που τυπικά υπάρχουν μεταξύ των συνεργαζόμενων μερών ενισχύουν την πολυπλοκότητα της ίδιας της φύσης της επαφής.

Από τη σκοπιά της μοντελοποίησης είναι απαραίτητη μια λεπτομερής περιγραφή της επαφής. Για την ανάδειξη των θεμελιωδών πλευρών του ελέγχου αλληλεπίδρασης είναι βολικό να καταλήξουμε σε ένα απλό, αλλά σημαντικό μοντέλο επαφών. Για αυτό το σκοπό, θεωρούμε ένα απεπλεγμένο ελαστικά συμμορφωμένο περιβάλλον (elastically compliant environment) που περιγράφεται από το μοντέλο:

$$h = \begin{bmatrix} f \\ \mu \end{bmatrix} = \begin{bmatrix} \mathrm{K}_f & O \\ O & K_\mu \end{bmatrix} \begin{bmatrix} dp \\ \omega dt \end{bmatrix} = K \begin{bmatrix} dp \\ \omega dt \end{bmatrix} \qquad (2.4)$$

όπου dp είναι το διάνυσμα μετασχηματισμού στους άξονες αναφοράς και ωdt είναι το διάνυσμα της στοιχειώδους περιστροφής περί αυτών των αξόνων. Για αυτό το διάνυσμα $[dp^T \quad \omega^{\mathrm{T}} dt]^T$ περιγράφει μια γενικευμένη μετατόπιση σε σχέση με τη θέση ηρεμίας του περιβάλλοντος. Το μητρώο δυσκαμψίας K είναι τυπικά θετικά ημι-ορισμένο. Στην πραγματικότητα το περιβάλλον δεν ασκεί δυνάμεις αντίδρασης προς τις κατευθύνσεις που επιτρέπεται μη περιορισμένη (unconstrained) κίνηση.

Η σχέση (2.4) μπορεί να γραφτεί σε σχέση με μεταβλητές του λειτουργικού χώρου ως εξής:

$$h = KT_A(x)dx \qquad (2.5)$$

όπου το dx εκφράζει τη γενικευμένη μετατόπιση του λειτουργικού χώρου σε σχέση με την απαραμόρφωτη θέση ηρεμίας του περιβάλλοντος x_e

$$dx = x - x_e \qquad (2.6)$$

και με $h_A = T_A^T(x)h$ έχουμε :

$$h_A = T_A^T(x)KT_A(x)dx = K_A(x)(x - x_e) \qquad (2.7)$$

που επιτρέπει να συσχετίσουμε τις ισοδύναμες δυνάμεις στο ρομποτικό βραχίονα με την παραμόρφωση του περιβάλλοντος μέσω του μητρώου K_A, δηλαδή το μητρώο δυσκαμψίας

του περιβάλλοντος. Το μητρώο K_A^{-1}, εάν αυτό μπορεί να οριστεί, είναι το μητρώο συμμόρφωσης του περιβάλλοντος. Αντιπροσωπεύει μία παθητική συμμόρφωση (passive compliance), αφότου περιγράφει μια ενυπάρχουσα ιδιότητα του περιβάλλοντος στο λειτουργικό χώρο, διαλεγμένη για να εκφράσει τη θέση και τον προσανατολισμό του ακροδέκτη του ρομποτικού βραχίονα. Υπενθυμίζοντας ότι το μητρώο K_A είναι μόνο θετικά ημι-ορισμένο, η αρχή της συμμόρφωσης δεν μπορεί να εφαρμοστεί γενικώς σε όλο το λειτουργικό χώρο, παρά μόνο στις κατευθύνσεις στις οποίες η κίνηση του ακροδέκτη περιορίζεται από το περιβάλλον.

Από την άλλη πλευρά να σημειώσουμε πως το μητρώο K_P^{-1} στη σχέση (2.3) αντιπροσωπεύει μία ενεργητική συμμόρφωση (active compliance), καθώς εφαρμόζεται στο βραχίονα μέσω ενός κατάλληλου ελέγχου θέσης. Με το μοντέλο του περιβάλλοντος στη σχέση (2.7) η σχέση (2.3) γίνεται:

$$\tilde{x} = K_P^{-1} K_A(x)(x - x_e) \qquad (2.8)$$

στην ισορροπία η θέση του ακροδέκτη δίνεται από τη σχέση:

$$x_\infty = (I + K_P^{-1} K_A(x))^{-1} (x_d + K_P^{-1} K_A(x) x_e) \qquad (2.9)$$

όπου η δύναμη επαφής φαίνεται να είναι:

$$h_{A\infty} = (I + K_A(x) K_P^{-1})^{-1} K_A(x)(x_d - x_e) \qquad (2.10)$$

Αναλύοντας τη σχέση (2.9) βλέπουμε πως η θέση ισορροπίας εξαρτάται από τη θέση ηρεμίας του περιβάλλοντος, όπως και από την επιθυμητή θέση που επιβάλλεται από το σύστημα ελέγχου στο ρομποτικό βραχίονα. Η αλληλεπίδραση των δύο συστημάτων, του περιβάλλοντος και του βραχίονα, επηρεάζεται από το κοινό βάρος των αντίστοιχων χαρακτηριστικών της συμμόρφωσης. Είναι έτσι δυνατόν να αυξηθεί η ενεργητική συμμόρφωση, ώστε ο βραχίονας να κυριαρχεί επί του περιβάλλοντος, και αντιστρόφως. Αυτή η κυριαρχία μπορεί να καθοριστεί ξεχωριστά σε σχέση με τις κατευθύνσεις του λειτουργικού χώρου. Για μια δεδομένη δυσκαμψία του περιβάλλοντος, σε σχέση και με την καθορισμένη εφαρμογή αλληλεπίδρασης, υπάρχει δυνατότητα επιλογής μεγάλων τιμών των στοιχείων του K_P για τις κατευθύνσεις στις οποίες το περιβάλλον πρέπει να συμμορφωθεί και αντίστοιχα μικρές των στοιχείων του K_P για τις κατευθύνσεις στις οποίες ο βραχίονας πρέπει να συμμορφωθεί.

Η σχέση (2.10) δίνει την τιμή των δυνάμεων επαφής στη θέση ισορροπίας, αποκαλύπτοντας έτσι πως ίσως είναι απαραίτητο να συντονίζεται ανάλογα η συμμόρφωση του περιβάλλοντος με αυτή του βραχίονα σε συγκεκριμένες κατευθύνσεις του λειτουργικού χώρου. Στην πραγματικότητα, κατά μήκος μιας κατεύθυνσης με υψηλή δυσκαμψία του

περιβάλλοντος είναι καλύτερα να γίνεται συμμόρφωση του βραχίονα, ώστε να ελαττώνεται σταδιακά η ένταση της αλληλεπίδρασης μέσω της κατάλληλης επιλογής της επιθυμητής θέσης. Σε αυτή την περίπτωση, η θέση ισορροπίας x_∞ του ακροδέκτη πρακτικά συμπίπτει με την απαραμόρφωτη θέση x_e του περιβάλλοντος, και ο βραχίονας δημιουργεί μια δύναμη αλληλεπίδρασης εξαρτημένη από τα αντίστοιχα στοιχεία του K_P που καθορίζεται από την επιλογή των στοιχείων $x_d - x_e$ κατά μήκος της σχετικής κατεύθυνσης.

Στην άλλη περίπτωση της υψηλής συμμόρφωσης του περιβάλλοντος, όταν ο βραχίονας έχει γίνει δύσκαμπτος, η θέση ισορροπίας x_∞ του ακροδέκτη είναι πολύ κοντά στην επιθυμητή θέση x_d, και είναι το περιβάλλον που δημιουργεί ελαστικές δυνάμεις κατά μήκος των περιορισμένων κατευθύνσεων που μας ενδιαφέρουν.

2.3 Έλεγχος Εμπέδησης (impedance control)

Είναι τώρα απαραίτητο να αναλύσουμε την αλληλεπίδραση του ρομποτικού βραχίονα με το περιβάλλον κάτω από την επίδραση ενός ελέγχου αντίστροφης δυναμικής στο λειτουργικό χώρο. Με αναφορά στο μοντέλο (2.1) θεωρούμε το νόμο ελέγχου $u = B(q)y + n(q,\dot{q})$

Με την παρουσία δυνάμεων στον ακροδέκτη του ρομποτικού βραχίονα, ο ελεγχόμενος βραχίονας περιγράφεται από τη σχέση:

$$\ddot{q} = y - B^{-1}(q)J^T(q)h \qquad (2.14)$$

που αποκαλύπτει την ύπαρξη έκφρασης μη-γραμμικής σύζευξης εξαιτίας των δυνάμεων επαφής. Επιλέγουμε το y ως:

$$y = J_A^{-1}(q)M_d^{-1}(M_d\ddot{x}_d + K_D\tilde{\dot{x}} + K_P\tilde{x} - M_d\dot{J}_A(q,\dot{q})\dot{q}) \qquad (2.15)$$

όπου το M_d είναι ένα θετικά ορισμένο διαγώνιο μητρώο. Αντικαθιστώντας τη σχέση (2.15) στη σχέση (2.14) και λαμβάνοντας υπόψη τη διαφορική κινηματική δευτέρου βαθμού προκύπτει:

$$M_d\ddot{\tilde{x}} + K_D\tilde{\dot{x}} + K_P\tilde{x} = M_dB_A^{-1}(q)h_A \qquad (2.16)$$

όπου $B_A(q) = J_A^{-T}(q)B(q)J_A^{-1}(q)$ είναι το μητρώο αδράνειας του ρομποτικού βραχίονα στο χώρο λειτουργίας. Το μητρώο αυτό είναι εξαρτημένο από τη δεδομένη κατάσταση και είναι θετικά ορισμένο εάν το μητρώο J_A είναι πλήρους βαθμού.

Η σχέση (2.16) αποδεικνύει τη σχέση μεταξύ του διανύσματος των επακόλουθων δυνάμεων $M_d B_A^{-1} h_A$ και του διανύσματος των μετατοπίσεων $\tilde{x}$ στο λειτουργικό χώρο μέσω μιας γενικευμένης μηχανολογικής εμπέδησης (mechanical impedance). Αυτή η εμπέδηση μπορεί να αποδοθεί σε ένα μηχανολογικό σύστημα χαρακτηριζόμενο από ένα μητρώο μάζας M_d, ένα μητρώο απόσβεσης K_D και ένα μητρώο δυσκαμψίας K_P που επιτρέπει να καθοριστεί η δυναμική συμπεριφορά κατά μήκος των κατευθύνσεων του λειτουργικού χώρου.

Η παρουσία του B_A^{-1} κάνει το σύστημα συζευγμένο. Εάν είναι επιθυμητό να διατηρηθεί η γραμμικότητα και η αποσύζευξη κατά τη διάρκεια της αλληλεπίδρασης με το περιβάλλον, είναι απαραίτητο να μετρηθεί η γενικευμένη δύναμη επαφής. Αυτό μπορεί να επιτευχθεί χρησιμοποιώντας τους κατάλληλους αισθητήρες δύναμης, οι οποίοι συνήθως προσαρμόζονται στον καρπό του ρομποτικού βραχίονα. Επιλέγοντας:

$$u = B(q)y + n(q,\dot{q}) + J^T(q)h \qquad (2.17)$$

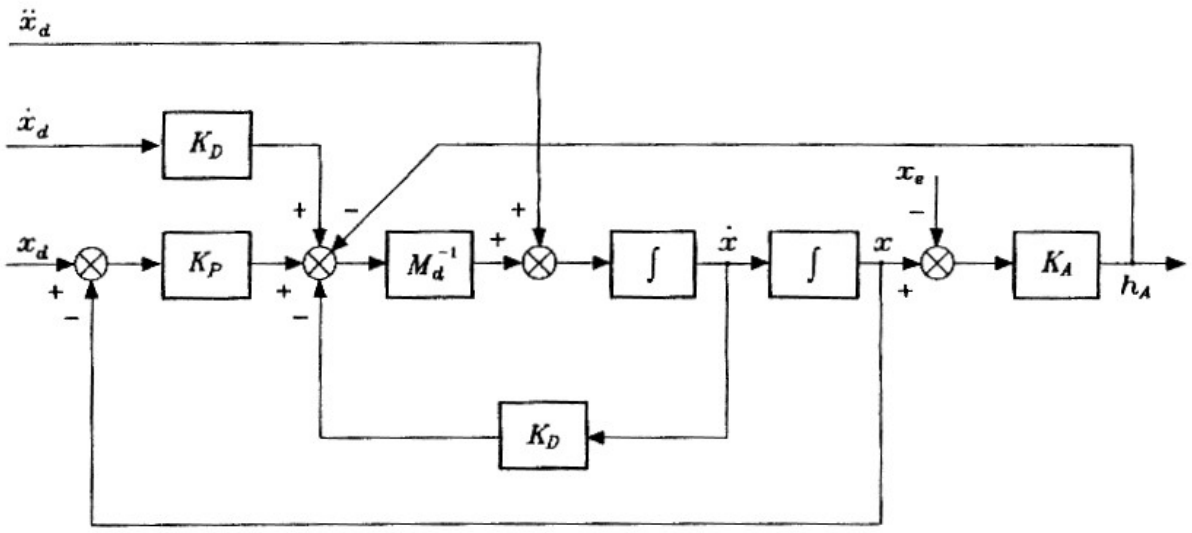
με

$$y = J_A^{-1}(q)M_d^{-1}(M_d\ddot{x}_d + K_D\dot{\tilde{x}} + K_P\tilde{x} - M_d\dot{J}_A(q,\dot{q})\dot{q} - h_A) \qquad (2.18)$$

με την υπόθεση μετρήσεων δυνάμεων χωρίς σφάλματα προκύπτει:

$$M_d\ddot{\tilde{x}} + K_D\dot{\tilde{x}} + K_P\tilde{x} = M_d B_A^{-1}(q)h_A \qquad (2.19)$$

Αξίζει να σημειωθεί πως η προσθήκη του όρου $J^T h$ στη σχέση (2.17) αντισταθμίζει επακριβώς τις δυνάμεις επαφής και στη συνέχεια καθιστά το βραχίονα απείρως δύσκαμπτο σε σχέση με τις εξωτερικές φορτίσεις. Για να αποδοθεί συμμορφωτική συμπεριφορά στο βραχίονα, ο όρος $-J_A^{-1}M_d^{-1}h_A$ έχει εισαχθεί στη σχέση (2.18), το οποίο μας επιτρέπει να χαρακτηρίσουμε το βραχίονα ως μία γραμμική εμπέδηση (linear impedance) σε σχέση με τις ισοδύναμες δυνάμεις h_A, όπως παρουσιάζεται στη σχέση (2.19). Το σχεδιάγραμμα ελέγχου που προκύπτει από ένα βραχίονα σε επαφή με ελαστικό περιβάλλον υπό έλεγχο εμπέδησης παρουσιάζεται στο Σχήμα 1.



Σχήμα 1. Ισοδύναμο σχεδιάγραμμα ελέγχου βραχίονα σε επαφή με ελαστικό περιβάλλον υπό έλεγχο εμπέδησης.

Η συμπεριφορά του συστήματος (2.19) στη θέση ισορροπίας είναι ανάλογη με την περιγραφή της (2.2). Παρόλα αυτά, συγκρίνοντας με έναν έλεγχο συμμόρφωσης προσδιορισμένο από το μητρώο K_P, η σχέση (2.19) επιτρέπει τον πλήρη προσδιορισμό των χαρακτηριστικών της δυναμικής του συστήματος μέσω μιας ενεργής εμπέδησης (active impedance) προσδιορισμένης από τα μητρώα M_d, K_D και K_P. Αυτά τα μητρώα συνήθως λαμβάνονται ως διαγώνια. Επίσης, δεν είναι δύσκολο να παρατηρήσουμε πως η εμπέδηση εξαρτάται από τη διάταξη του συστήματος (configuration) όσον αφορά στις συνιστώσες των δυνάμεων, ενώ εξαρτάται από την τρέχουσα διάταξη του ρομποτικού βραχίονα όσον αφορά στις συνιστώσες των ροπών μέσω του μητρώου T_A.

Επιπλέον, παρόμοια με την ενεργητική και την παθητική συμμόρφωση, το μοντέλο της παθητικής εμπέδησης μπορεί να χρησιμοποιηθεί εάν η δύναμη αλληλεπίδρασης h_A παράγεται κατά την επαφή με ένα περιβάλλον κατάλληλης μάζας, απόσβεσης και δυσκαμψίας. Σε αυτή την περίπτωση, το σύστημα του βραχίονα με το περιβάλλον μπορεί να χαρακτηριστεί ως ένα μηχανολογικό σύστημα αποτελούμενο από τις δύο εμπεδήσεις παράλληλα, και τότε η δυναμική συμπεριφορά εξαρτάται και καθορίζεται από τη σχετική βαρύτητα αυτών των δύο.

2.4 Έλεγχος Δύναμης

Εισαγωγή:

Στις μεθοδολογίες που αναλύθηκαν η δύναμη αλληλεπίδρασης μπορούσε να ελεγχθεί έμμεσα με μεταβολή της τιμής αναφοράς x_d του συστήματος ελέγχου κίνησης του βραχίονα. Έτσι, η αλληλεπίδραση του βραχίονα και του περιβάλλοντος ελεγχόταν εξίσου έμμεσα μέσω της συμμόρφωσης (compliance) του περιβάλλοντος και είτε της συμμόρφωσης (compliance) ή της εμπέδησης (impedance) του βραχίονα.

Εάν είναι επιθυμητός ο ακριβής έλεγχος της δύναμης επαφής, είναι απαραίτητο να καταστρώσουμε μεθοδολογίες ελέγχου που επιτρέπουν τον άμεσο καθορισμό της επιθυμητής δύναμης ελέγχου. Η ανάπτυξη ενός συστήματος ελέγχου δύναμης (force control), κατ' αναλογία με ένα σύστημα ελέγχου κίνησης, θα απαιτούσε να χρησιμοποιηθεί ένας σταθεροποιητικός έλεγχος PD στο σφάλμα της δύναμης εκτός από την τυπική αντιστάθμιση της μη-γραμμικότητας (nonlinear). Οι μετρήσεις δυνάμεων συνήθως αλλοιώνονται από το θόρυβο και στην πράξη δεν μπορούμε εύκολα να κάνουμε παραγώγιση. Η σταθεροποιητική διαδικασία μπορεί να προκύψει από την κατάλληλη ολοκλήρωση των δεδομένων της ταχύτητας. Σαν αποτέλεσμα, ένα τυπικό σύστημα ελέγχου δύναμης χαρακτηρίζεται από έναν νόμο ελέγχου που δε βασίζεται μόνο στις μετρήσεις των τιμών των δυνάμεων, αλλά και σε μετρήσεις των τιμών της ταχύτητας και τελικά και σε μετρήσεις των τιμών της θέσης.

Η πραγματοποίηση μιας μεθοδολογίας ελέγχου δύναμης είναι εφικτή με την ολοκλήρωση ενός βρόχου ανάδρασης ρύθμισης εξωτερικής δύναμης (outer force regulation feedback loop) παράγοντας τα δεδομένα εισόδου σε μια τυπική μεθοδολογία ελέγχου κίνησης. Επομένως, οι μεθοδολογίες ελέγχου δύναμης που αναλύονται στη συνέχεια είναι βασισμένες στη χρήση ελέγχου θέσης αντίστροφης δυναμικής (inverse dynamic position control). Ωστόσο, να σημειώσουμε πως η χρήση νόμων ελέγχου δύναμης έχει νόημα μόνο στις κατευθύνσεις του χώρου λειτουργίας που ενδέχεται να παρουσιαστούν δυνάμεις αλληλεπίδρασης μεταξύ του βραχίονα και του περιβάλλοντος.

2.41 Έλεγχος Δύναμης με εσωτερικό βρόχο Θέσης (inner position loop)

Αναφορικά με το νόμο αντίστροφης δυναμικής (Σχέση 2.17) έχουμε το νόμο ελέγχου:

$$y = J_A^{-1}(q)M_d^{-1}(-K_D\dot{x} + K_P(x_F - x) - M_d\dot{J}_A(q,\dot{q})\dot{q}) \qquad (2.20)$$

όπου το x_F είναι η κατάλληλη αναφορά που σχετίζεται με το σφάλμα δύναμης.

Να σημειώσουμε ότι ο νόμος ελέγχου (2.20) δεν προβλέπει αντισταθμιστικές διεργασίες που σχετίζονται με το $\dot{x}_F$ και το $\ddot{x}_F$. Αντικαθιστώντας τη (2.20) στη (2.17) οδηγούμαστε έπειτα από αλγεβρικές διεργασίες στο ακόλουθο σύστημα:

$$M_d \ddot{x} + K_D \dot{x} + K_P x = K_P x_F \qquad (2.21)$$

που δείχνει πως οι σχέσεις (2.17) και (2.20) πραγματοποιούν έλεγχο θέσης περνώντας από το x στο x_F μέσω δυναμικής που καθορίζεται από τα μητρώα M_d, K_D και K_P.

Ας χρησιμοποιήσουμε το h_{Ad} για να προσδιορίσουμε την επιθυμητή σταθερή δύναμη. Η σχέση μεταξύ του x_F και του σφάλματος της δύναμης μπορεί να παραστεί συμβολικά ως:

$$x_F = C_F (h_{Ad} - h_A) \qquad (2.22)$$

όπου το C_F είναι ένα διαγώνιο μητρώο του οποίου τα στοιχεία δίνουν τις εντολές ελέγχου που πρέπει να εφαρμοστούν κατά μήκος των κατευθύνσεων του χώρου λειτουργίας που μας ενδιαφέρουν. Οι σχέσεις (2.21) και (2.22) αποκαλύπτουν πως ο νόμος ελέγχου δύναμης είναι ανεπτυγμένος πάνω στη βάση προϋπαρχουσών βρόχων ελέγχου θέσης.

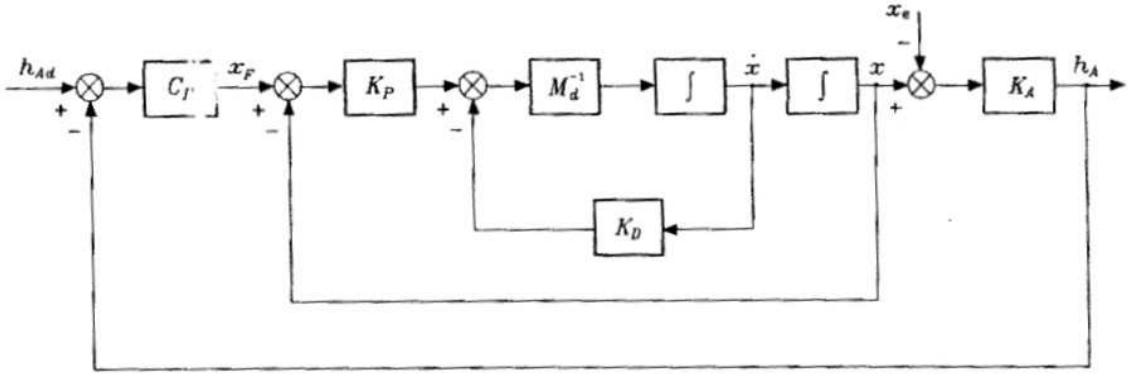
Στην υπόθεση του ελαστικά συμμορφώσιμου περιβάλλοντος που περιγράφεται στη σχέση (2.7) η σχέση (2.21) με τη (2.22) γίνεται:

$$M_d \ddot{x} + K_D \dot{x} + K_P (I + C_F K_A) x = K_P C_F (K_A x_e + h_{Ad}) \qquad (2.23)$$

Για να ληφθεί η απόφαση για το είδος του ελέγχου που θα καθοριστεί για το C_F είναι σκόπιμο να συσχετιστούν οι σχέσεις (2.7), (2.21) και (2.22) με το σχεδιάγραμμα ελέγχου στο Σχήμα 2 που προκύπτει λογικά από το σχεδιάγραμμα ελέγχου στο Σχήμα 1.

Υποδηλώνεται πως εάν το C_F εμπεριείχε μια καθαρά αναλογική (proportional) δράση ελέγχου, τότε το h_A δε θα έφτανε ποτέ το h_{Ad} και το x_e θα επηρέαζε τη δύναμη αλληλεπίδρασης ακόμα και σε σταθερή κατάσταση.

Ακόμη, εάν το C_F εμπεριείχε μία δράση ελέγχου ολοκλήρωσης (integral) στις συνιστώσες της γενικευμένης δύναμης, θα ήταν δυνατό να επιτύχουμε $h_A = h_{Ad}$ στη σταθερή κατάσταση και την ίδια στιγμή να αποτρέψουμε την επιρροή του x_e πάνω στο h_A.



Σχήμα 2. Σχεδιάγραμμα Ελέγχου δύναμης με χρήση εσωτερικού βρόχου θέσης.

Για αυτό το λόγο, μια βολική επιλογή για το C_F είναι μια αναλογική-ολοκληρωτική (proportional-integral / PI) δράση:

$$C_F = K_F + K_I \int_0^t (\cdot) d\varsigma \qquad (2.24)$$

Το δυναμικό σύστημα που είναι αποτέλεσμα των σχέσεων (2.23) και (2.24) είναι 3ου βαθμού και είναι απαραίτητο να ορίσουμε επαρκώς τα μητρώα K_D, K_P, K_F και K_I σε σχέση με τα χαρακτηριστικά του περιβάλλοντος. Λόγω της τυπικά υψηλής τιμής της δυσκαμψίας του περιβάλλοντος, η επίδραση της αναλογικής (proportional) και της ολοκληρωτικής (integral) δράσης πρέπει να συμπεριληφθούν. Η επιλογή των K_F και K_I επηρεάζει τα περιθώρια ευστάθειας και το εύρος ζώνης (bandwidth) του συστήματος υπό έλεγχο δύναμης. Με την υπόθεση πως επιτυγχάνεται ευσταθής ισορροπία έχουμε $h_{A,\infty} = h_{Ad}$ και τότε:

$$K_A x_\infty = K_A x_e + h_{Ad} \qquad (2.25)$$

2.42 Έλεγχος Δύναμης με εσωτερικό βρόχο Ταχύτητας (inner velocity loop)

Από το σχεδιάγραμμα στο Σχήμα 2 μπορεί να παρατηρηθεί πως εάν ανοίξει ο βρόχος ανάδρασης θέσης, το x_F εκφράζει μια αναφορά σε ταχύτητα και τότε υπάρχει σχέση ολοκλήρωσης μεταξύ x_F και x . Έτσι οδηγούμαστε στο να αναγνωρίσουμε ότι, σε αυτή την περίπτωση, η δύναμη αλληλεπίδρασης με το περιβάλλον συμπίπτει με την επιθυμητή τιμή στη σταθερή κατάσταση, ακόμα και με έναν αναλογικό ελεγκτή δύναμης (proportional force controller) C_F. Για την ακρίβεια επιλέγοντας:

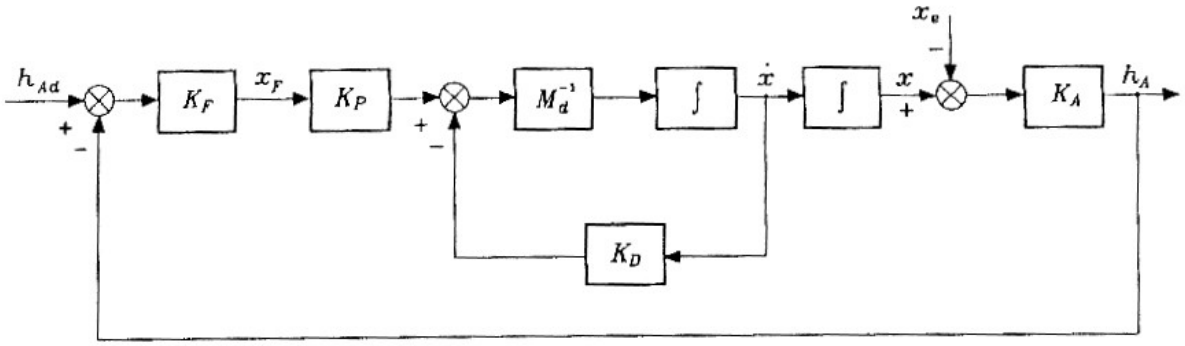
$$y = J_A^{-1}(q)M_d^{-1}(-K_D\dot{x} + K_P x_F - M_d \dot{J}_A(q,\dot{q})\dot{q}) \qquad (2.26)$$

με μια καθαρή δομή αναλογικού (proportional) ελέγχου $(C_F = K_F)$ στο σφάλμα δύναμης προκύπτει:

$$x_F = K_F(h_{Ad} - h_A) \qquad (2.27)$$

και το δυναμικό σύστημα περιγράφεται από τη σχέση:

$$M_d\ddot{x} + K_D\dot{x} + K_P K_F K_A x = K_P K_F (K_A x_e + h_{Ad}) \qquad (2.28)$$



Σχήμα 3. Σχεδιάγραμμα Ελέγχου δύναμης με χρήση εσωτερικού βρόχου ταχύτητας.

Η σχέση μεταξύ της θέσης και της δύναμης επαφής στην ισορροπία δίνεται από τη σχέση (2.25). Το σχετικό σχεδιάγραμμα ελέγχου παρουσιάζεται στο Σχήμα 3. Τέλος, να τονίσουμε ότι ο σχεδιασμός ελέγχου έχει απλοποιηθεί, καθώς το σύστημα που προκύπτει είναι τώρα 2ου βαθμού. (Στην πραγματικότητα τα μητρώα K_P και K_F δεν είναι ανεξάρτητα και μπορούμε να αναφερόμαστε σε ένα μόνο μητρώο $K'_F = K_P K_F$.)

2.43 Παράλληλος έλεγχος Δύναμης/Θέσης

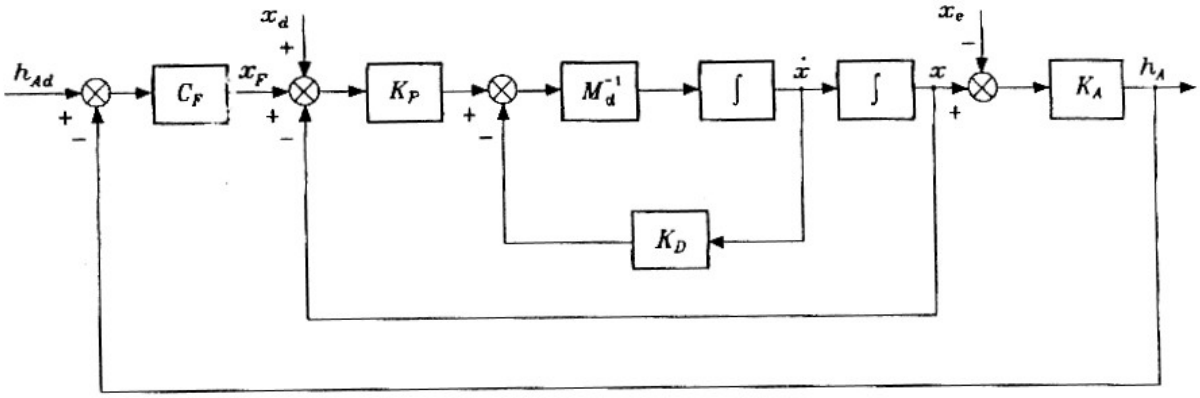
Οι μεθοδολογίες ελέγχου δύναμης που παρουσιάστηκαν απαιτούν τη συμφωνία των πληροφοριών της δύναμης με τα γεωμετρικά χαρακτηριστικά του περιβάλλοντος. Στην πραγματικότητα, εάν το h_{Ad} έχει συνιστώσες εκτός του $R(K_A)$ οι σχέσεις (2.23) και (2.28) δείχνουν πως κατά μήκος των αντίστοιχων κατευθύνσεων του λειτουργικού χώρου οι συνιστώσες του h_{Ad} μεταφράζονται ως συνιστώσες ταχύτητας που προκαλούν κίνηση της θέσης του ακροδέκτη. Εάν υπάρχει σωστός σχεδιασμός του h_{Ad} κατά μήκος των

κατευθύνσεων εκτός του $R(K_A)$, η κίνηση που προκύπτει, καθορισμένη από τον αντίστοιχο έλεγχο κίνησης, τείνει να φέρει τη θέση του ακροδέκτη στο μηδέν, στην περίπτωση της σχέσης (2.23), και την ταχύτητα του ακροδέκτη στο μηδέν, στην περίπτωση της σχέσης (2.28). Για αυτό το λόγο, οι παραπάνω μεθοδολογίες ελέγχου δεν επιτρέπουν τον έλεγχο κίνησης ούτε στις αποδεκτές κατευθύνσεις της εφαρμογής.

Εάν θέλουμε να προσδιορίσουμε την επιθυμητή θέση x_d του ακροδέκτη, όπως στις μεθοδολογίες καθαρού ελέγχου κίνησης, το σχεδιάγραμμα στο Σχήμα 2 μπορεί να προσαρμοστεί προσθέτοντας το στοιχείο x_d στην είσοδο που οι θέσεις αθροίζονται. Αυτό αντιστοιχεί στην επιλογή του νόμου ελέγχου:

$$y = J_A^{-1}(q)M_d^{-1}(-K_D\dot{x} + K_P(\tilde{x} + x_F) - M_d\dot{J}_A(q,\dot{q})\dot{q}) \qquad (2.29)$$

όπου $\tilde{x} = x_d - x$. Το σχεδιάγραμμα που προκύπτει παρουσιάζεται στο Σχήμα 4 και ονομάζεται παράλληλος έλεγχος δύναμης/θέσης (parallel force/position control), με την ύπαρξη μιας δράσης ελέγχου θέσης $K_P\tilde{x}$ παράλληλα με μια δράση ελέγχου δύναμης $K_P C_F(h_{Ad} - h_A)$.



Σχήμα 4. Σχεδιάγραμμα παράλληλου ελέγχου με δύναμη/θέση.

Επαληθεύουμε εύκολα ότι σε αυτή την περίπτωση η θέση ισορροπίας ικανοποιεί τη σχέση:

$$x_\infty = x_d + C_F(K_A(x_e - x_\infty) + h_{Ad}) \qquad (2.30)$$

Συνεπώς, κατά μήκος των κατευθύνσεων εκτός του $R(K_A)$ που η κίνηση δεν είναι περιορισμένη, η συνιστώσα θέσης x_d επιτυγχάνεται μέσω του x. Αντιστρόφως, κατά μήκος των κατευθύνσεων εντός του $R(K_A)$, που η κίνηση περιορίζεται, το x_d αντιμετωπίζεται ως μια επιπλέον διαταραχή. Τέλος, μέσω του C_F εξασφαλίζουμε την προσέγγιση του h_{Ad} στη σταθερή κατάσταση εις βάρος του σφάλματος θέσης x, ανάλογα με τη συμμόρφωση του περιβάλλοντος.

2.5 Φυσικοί και Τεχνητοί περιορισμοί

Μεθοδολογίες ελέγχου αλληλεπίδρασης μπορούν να χρησιμοποιηθούν για την εκτέλεση κινήσεων με περιορισμούς μόνο όταν οι θέσεις αναφοράς είναι συμβατές με τη γεωμετρία του περιβάλλοντος.

Μια πραγματική εφαρμογή χειρισμού ρομποτικού βραχίονα χαρακτηρίζεται από σύνθετες καταστάσεις επαφών, στις οποίες κάποιες κατευθύνσεις υπόκεινται σε περιορισμό της θέσης του ακροδέκτη του βραχίονα και κάποιες άλλες σε περιορισμούς λόγω δυνάμεων αλληλεπίδρασης. Κατά τη διάρκεια της εκτέλεσης της εφαρμογής η φύση των περιορισμών μπορεί να μεταβάλλεται ουσιαστικά.

Η απαίτηση χειρισμού σύνθετων περιπτώσεων επαφών απαιτεί τη δυνατότητα καθορισμού και εκτέλεσης ελέγχου και της θέσης του ακροδέκτη του ρομποτικού βραχίονα, αλλά και της δύναμης επαφής. Ένα κλασικό παράδειγμα είναι σε μια εφαρμογή φινιρίσματος επιφάνειας, όπου η κίνηση του εργαλείου καθορίζεται να είναι εφαπτομενική προς την επιφάνεια του τεμαχίου, ενώ ταυτόχρονα απαιτείται να ασκείται καθορισμένη σταθερή δύναμη κάθετα προς αυτή.

Μια θεμελιώδης άποψη που πρέπει να συνυπολογίσουμε είναι ότι δεν είναι δυνατό να επιβάλλουμε αυθαίρετα τιμές θέσης και δύναμης σε κάθε κατεύθυνση. Ως αποτέλεσμα, πρέπει να εξασφαλιστεί πως οι τροχιές αναφοράς του συστήματος ελέγχου είναι συμβατές με τους περιορισμούς που προκύπτουν λόγω του περιβάλλοντος κατά τη διάρκεια της εκτέλεσης μιας εφαρμογής, έτσι ώστε να επιτυγχάνεται ένας σωστός καθορισμός του προβλήματος ελέγχου.

Η κινηματικοστατική (kineto-static) ανάλυση της κατάστασης αλληλεπίδρασης του βραχίονα και του περιβάλλοντος οδηγεί στα ακόλουθα συμπεράσματα:

- Κατά μήκος κάθε βαθμού ελευθερίας του χώρου εργασίας το περιβάλλον επιβάλλει περιορισμούς είτε δύναμης είτε θέσης στον ακροδέκτη του ρομποτικού βραχίονα. Αυτοί οι περιορισμοί ορίζονται ως φυσικοί περιορισμοί (natural constraints), καθώς καθορίζονται άμεσα από τη γεωμετρία του χώρου που λαμβάνει χώρα η εφαρμογή.
- Κατά μήκος κάθε βαθμού ελευθερίας του χώρου εργασίας ο ρομποτικός βραχίονας μπορεί να ελέγξει μόνο τις μεταβλητές που δεν υπόκεινται στους φυσικούς περιορισμούς. Οι τιμές αναφοράς για αυτές τις μεταβλητές ορίζονται ως τεχνητοί περιορισμοί (artificial constraints), καθώς επιβάλλονται λαμβάνοντας υπόψη τη στρατηγική της εκτέλεσης της εκάστοτε εφαρμογής.

Να σημειώσουμε πως τα δύο αυτά είδη περιορισμών είναι συμπληρωματικά, καθώς λαμβάνουν υπόψη διαφορετικές μεταβλητές σε κάθε βαθμό ελευθερίας. Επίσης επιτρέπουν

τον ολοκληρωμένο καθορισμό της εφαρμογής εμπεριέχοντας όλες τις μεταβλητές. Οι κινηματικοστατικές (kineto-static) μεταβλητές σχετίζονται με αυτούς τους περιορισμούς.

Πρέπει να τονιστεί πως τα ανωτέρω παραδείγματα, που παρουσιάζουν εκτέλεση στρατηγικών ελέγχου κίνησης και δύναμης, αναμφίβολα ερμηνεύουν την παρουσία φυσικών και τεχνητών περιορισμών. Παρατηρώντας την απλή γεωμετρία της δεδομένης εφαρμογής, οι βαθμοί ελευθερίας περιγράφονται επαρκώς σε σχέση με τη βάση αναφοράς, αναλαμβάνοντας να εκφράσουν ποσότητες του λειτουργικού χώρου.

Σε μια πιο γενικευμένη περίπτωση μπορούμε να παρουσιάσουμε ένα πλαίσιο περιορισμών (constraint frame) O_c - x_c y_c z_c, όχι απαραίτητα ευθυγραμμισμένο με τη βάση αναφοράς, με σκοπό την απλοποίηση της περιγραφής της εφαρμογής και τον καθορισμό των φυσικών περιορισμών και στη συνέχεια των αντίστοιχων τεχνητών περιορισμών.

Η εφαρμογή του πλαισίου περιορισμών απλοποιεί αισθητά το σχεδιασμό των εφαρμογών, αλλά η στρατηγική ελέγχου πρέπει προφανώς να λάβει υπόψη το μητρώο περιστροφής (που είναι πρακτικά μεταβαλλόμενο κατά τη διάρκεια της εκτέλεσης της εφαρμογής), ώστε να μπορούμε να μετασχηματίσουμε ποσότητες εκφρασμένες στη βάση αναφοράς σε αντίστοιχες εκφρασμένες στο πλαίσιο περιορισμών.

ΚΕΦΑΛΑΙΟ 3ο : Περιγραφή Εγκατάστασης

Εισαγωγή:

Για την πραγματοποίηση πειραμάτων ελέγχου δύναμης/ροπής και την εξέταση των μεθοδολογιών που αναπτύχθηκαν στο προηγούμενο κεφάλαιο, είναι απαραίτητο να έχουμε στη διάθεσή μας τον απαραίτητο εξοπλισμό. Σκοπός αυτού του κεφαλαίου είναι η περιγραφή της εγκατάστασης που χρησιμοποιήθηκε για τις εφαρμογές ελέγχου και η πλήρης ανάλυση όλων των εξαρτημάτων και της λειτουργίας τους. Επίσης γίνεται αναφορά σε τροποποιήσεις του υπάρχοντος εξοπλισμού και η κατασκευή επιπλέον εξαρτημάτων που κρίθηκαν απαραίτητα.

Αρχικά περιγράφεται η βάση μιας εφαρμογής ελέγχου, ένας ρομποτικός βραχίονας και συγκεκριμένα ο ρομποτικός βραχίονας PA-10 επτά βαθμών ελευθερίας που είναι διαθέσιμος για τις εφαρμογές μας. Στη συνέχεια γίνεται περιγραφή της συσκευής που σκόπευε να μας παρέχει ανάδραση στο σύστημα ελέγχου, ένας αισθητήρας, και συγκεκριμένα ένας αισθητήρας δύναμης/ροπής μιας και η μελέτη μας θα επικεντρωθεί σε νόμους ελέγχου δύναμης/ροπής.

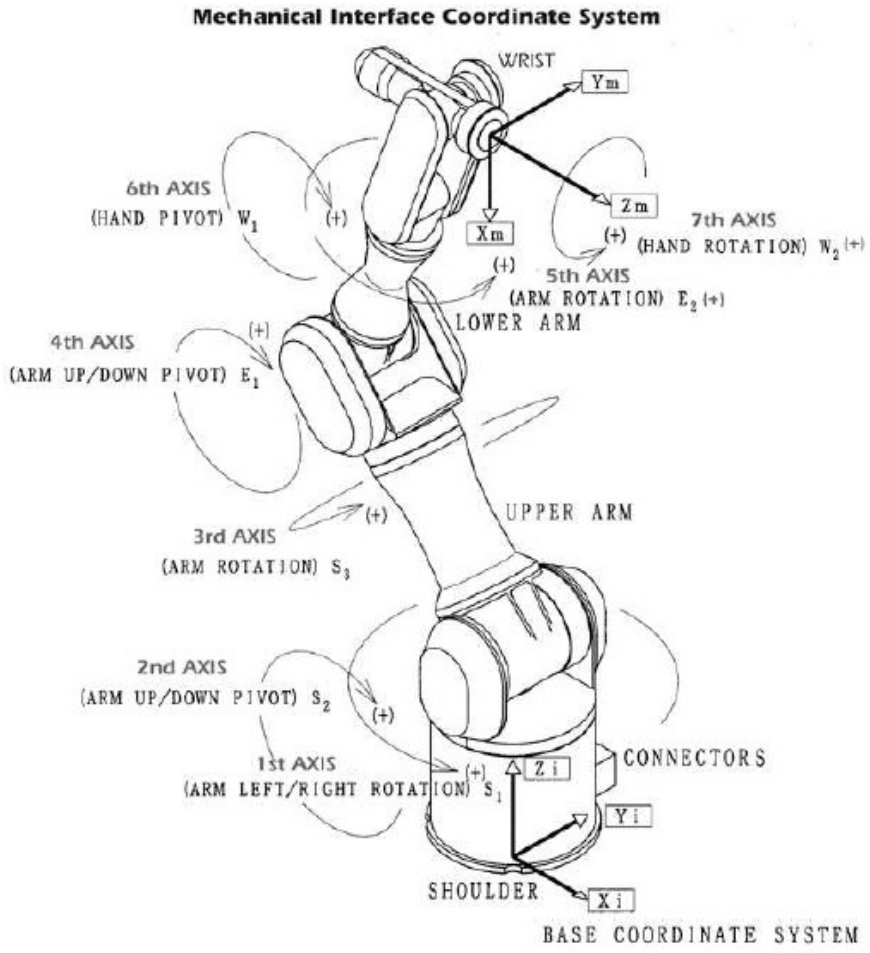
Η κατασκευή ενός μέσου σύνδεσης των δύο κύριων εξαρτημάτων, μια αλουμινένια φλάντζα εξετάζεται και αναλύεται στη συνέχεια, καθώς και κάποιες τροποποιήσεις και επεμβάσεις που κρίθηκαν απαραίτητες για τη σωστή λειτουργία της.

Τέλος, γίνεται ανάλυση βήμα προς βήμα μιας επίπονης, αλλά εξίσου απαραίτητης διαδικασίας βαθμονόμησης του αισθητήρα και στους 6 βαθμούς ελευθερίας του, καθώς και παράθεσης όλων των μεθοδολογιών που εξετάστηκαν μέχρι την τελική βαθμονόμηση της συσκευής μας. Παράλληλα έγινε μελέτη για την αντιστάθμιση διαταραχών στις μετρήσεις του αισθητήρα λόγω του βάρους του (ή και του βάρους κάποιου επιπλέον εξαρτήματος προσδεδεμένου σε αυτόν) σε άμεση επικοινωνία με τον ελεγκτή του ρομπότ, ώστε να αντισταθμίζεται το διάνυσμα του επιπλέον βάρους, ανάλογα με τον προσανατολισμό του τελικού σημείου δράσης του ρομποτικού βραχίονα.

3.1 Ρομποτικός βραχίονας PA-10 της Mitsubishi

Εισαγωγή:

Ο βραχίονας PA-10 της Mitsubishi Heavy Industries (Σχήμα 5) είναι ένας ρομποτικός βραχίονας με 7 βαθμούς ελευθερίας. Οι βαθμοί αυτοί είναι τοποθετημένοι σε αντιστοιχία με τους βαθμούς ελευθερίας του ανθρώπινου βραχίονα, με αποτέλεσμα ο ρομποτικός βραχίονας να είναι ιδιαίτερα ευέλικτος και, έχοντας έναν πλεονάζοντα βαθμό ελευθερίας, να μπορεί να πραγματοποιήσει πολύπλοκες κινήσεις στο χώρο. Αναλυτικά, η διάταξη του βραχίονα όπου διακρίνονται όλοι οι βαθμοί ελευθερίας του παρουσιάζεται στο Σχήμα 5.



Σχήμα 5. Σχηματική αναπαράσταση του PA-10.

Οι σύνδεσμοι του βραχίονα οδηγούνται από κινητήρες ΣΡ χωρίς ψήκτρες (DC brushless) μέσω αρμονικών μειωτήρων με σχέση μετάδοσης 50. Η ροπή (πριν τη μείωση στροφών) και η μέγιστη ταχύτητα περιστροφής διαφοροποιείται από κινητήρα σε κινητήρα, όπως φαίνεται στον Πίνακα 1 , όπου S = Shoulder (ώμος), E = Elbow (αγκώνας), W = Wrist (καρπός)

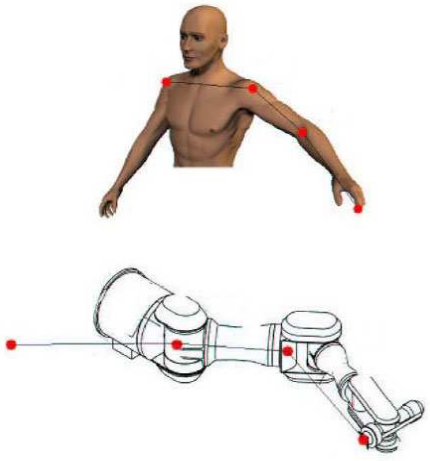
β.ε.	ονομαστική ροπή *(πριν τη μείωση)*	μέγιστη ταχύτητα *(μετά τη μείωση)*	Μηχανικό όριο
1 (S1)	4.64 $N \cdot m$	1 rad/s	±177.0°
2 (S2)	4.64 $N \cdot m$	1 rad/s	± 94.0°
3 (S3)	2.00 $N \cdot m$	2 rad/s	±174.0°
4 (E1)	2.00 $N \cdot m$	2 rad/s	±137.0°
5 (E2)	0.29 $N \cdot m$	2 rad/s	±255.0°
6 (W1)	0.29 $N \cdot m$	2π rad/s	±165.0°
7 (W2)	0.29 $N \cdot m$	2π rad/s	±255.0°

Πίνακας 1. Τεχνικά χαρακτηριστικά βραχίονα PA-10

3.11 Ο ελεγκτής του PA-10

Ο βραχίονας συνοδεύεται από έναν ελεγκτή, ο οποίος αναλαμβάνει να οδηγήσει απευθείας τους σερβοκινητήρες και να παρέχει ένα ενδιάμεσο διαδικτυακής επικοινωνίας (network interface) τεχνολογίας ARCnet, για επικοινωνία με τον H/Y. Μέσω του ενδιάμεσου αυτού ο ελεγκτής λαμβάνει τις εντολές κίνησης και επιστρέφει την τρέχουσα κατάσταση του βραχίονα.

Ο ελεγκτής διαθέτει 8 (οκτώ) μικροεπεξεργαστές, από έναν για τον έλεγχο κάθε άρθρωσης (σύνολο 7), και έναν για την κεντρική επίβλεψη της λειτουργίας του βραχίονα και τον έλεγχο των επικοινωνιών. Κάθε ένας από τους μικροεπεξεργαστές ελέγχου είναι επιφορτισμένος με τον έλεγχο και την επίβλεψη της αντίστοιχης άρθρωσης, δηλαδή "τρέχει" το νόμο ελέγχου της άρθρωσης, ενώ παράλληλα συλλέγει δεδομένα από τους αισθητήρες της (ροπή, γωνιακή ταχύτητα και γωνιακή θέση) και τα προωθεί προς τον H/Y. Ταυτόχρονα επιτηρεί την απρόσκοπτη λειτουργία της άρθρωσης που ελέγχει και λαμβάνει τα κατάλληλα μέτρα (διακοπή λειτουργίας και αποστολή σημάτων συναγερμού) όταν παρατηρηθεί κάποια ανωμαλία, όπως υπέρβαση του ορίου θέσης, μεγάλη μεταβολή της ταχύτητας. υπερθέρμανση ή υπερβολική κατανάλωση ισχύος.



Σχήμα 6. Αντιστοιχία των βαθμών ελευθερίας του ρομποτικού βραχίονα σε σχέση με αυτούς του ανθρώπου.

Ο μικροεπεξεργαστής συντονισμού είναι επιφορτισμένος με τον έλεγχο της διαδικτυακής επικοινωνίας ARCnet, καθώς και με την επίβλεψη της γενικής λειτουργίας του βραχίονα, στα πλαίσια της οποίας ελέγχει παραμέτρους από το εξωτερικό περιβάλλον, αλλά και την ίδια τη λειτουργία του ελεγκτή (π.χ. πάτημα κουμπιού στάσης κινδύνου, σφάλμα δικτυακού ενδιαμέσου, σφάλμα τροφοδοσίας κ.λπ.).

Ο Η/Υ μπορεί να αποστέλλει στον ελεγκτή, για κάθε άρθρωση, την επιθυμητή ταχύτητα ή την επιθυμητή ροπή. Ο ελεγκτής εσωτερικά τρέχει δύο πολύ γρήγορους νόμους ελέγχου, έναν PI για τον έλεγχο ρεύματος, με διάρκεια του βρόχου ελέγχου 100 μs, και έναν PID για τον έλεγχο ταχύτητας, με διάρκεια του βρόχου ελέγχου 400 μs. Κάθε άρθρωση μπορεί να ελέγχεται με έναν από τους δύο τρόπους, καθιστώντας το βραχίονα ιδιαίτερα ευέλικτο και δίνοντας πρακτικά απεριόριστες δυνατότητες προγραμματισμού ενός «soft» controller.

Συνολικά πρόκειται για μία στιβαρότατη κατασκευή φτιαγμένη με γνώμονα την όσο το δυνατόν απρόσκοπτη και ανεκτική σε σφάλματα (fail-safe) λειτουργία. Το γεγονός ότι χρησιμοποιεί ένα ενδιάμεσο δικτυακής επικοινωνίας τύπου ARCnet, σε συνδυασμό με το ότι παρέχεται η πλήρης προδιαγραφή του πρωτοκόλλου επικοινωνίας, επιτρέπει την πλήρη αξιοποίηση των δυνατοτήτων του βραχίονα.

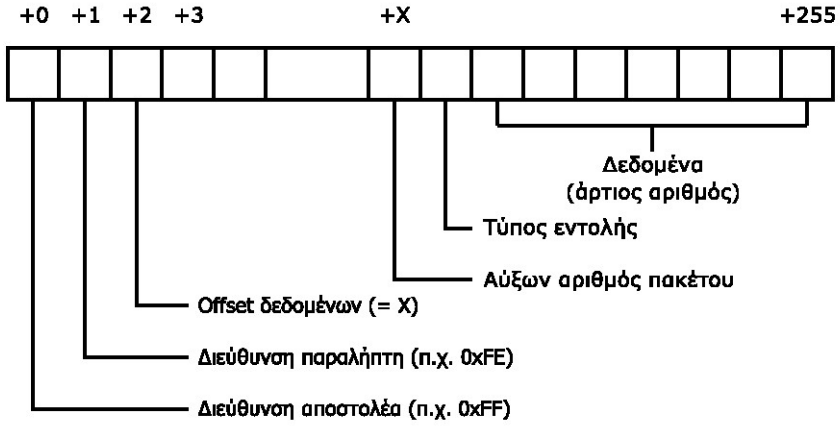
3.12 Το πρωτόκολλο επικοινωνίας του PA-10

Η επικοινωνία του ελεγκτή του PA-10 με τον Η/Υ γίνεται με ένα μοντέλο κύκλου εντολής - απάντησης. Σε κάθε κύκλο επικοινωνίας ο Η/Υ στέλνει ένα πακέτο δεδομένων με εντολές προς τον ελεγκτή (servo driver), ο οποίος με τη σειρά του απαντά στέλνοντας πληροφορίες για την τρέχουσα κατάσταση του βραχίονα (θέση / ροπή / ταχύτητα για κάθε άρθρωση) ή και του ίδιου του ελεγκτή. Υπάρχουν τρία είδη εντολών που μπορεί να λάβει ο ελεγκτής:

- *Εντολές κίνησης*, οι οποίες ελέγχουν την κίνηση του βραχίονα. Αυτή είναι η κανονική επικοινωνία.
- *Εντολές ρύθμισης*, οι οποίες αλλάζουν τις τιμές παραμέτρων του βραχίονα ή του ίδιου του ελεγκτή, οι οποίες βρίσκονται αποθηκευμένες στην EEPROM του ελεγκτή.
- *Ειδικές εντολές*, οι οποίες χρησιμοποιούνται σε εξαιρετικές περιπτώσεις είτε για να «λυθεί» το φρένο μίας άρθρωσης, είτε για να επανέλθει ο ελεγκτής σε κατάσταση λειτουργίας μετά τον εντοπισμό ανωμαλίας.

Η επικοινωνία γίνεται μέσω του δικτύου ARCnet, χρησιμοποιώντας τον «κοντό» τύπο πακέτου (256 bytes). Σύμφωνα με την κατασκευάστρια εταιρία, Mitsubishi Heavy Industries, οι διευθύνσεις του Η/Υ και του ελεγκτή πρέπει να είναι OxFF και OxFE

αντίστοιχα και ο αύξων αριθμός του πακέτου να διατηρείται ο ίδιος σε κάθε ζεύγος εντολής - απάντησης.



Σχήμα 7. Η μορφή του πακέτου δεδομένων στην επικοινωνία του PA-10.

Ο τύπος εντολής γράφεται στο πακέτο ως ένας κεφαλαίος χαρακτήρας ASCII. Οι διαθέσιμοι τύποι εντολής είναι οι εξής:

1. "S" (communication start): Η εντολή αυτή σηματοδοτεί την έναρξη της κανονικής επικοινωνίας για τον έλεγχο της κίνησης του βραχίονα. Η εντολή αυτή δεν περιέχει δεδομένα και ακολουθείται πάντα από μία ή περισσότερες εντολές ‘c’.

2. "C" (control data transmission): Η εντολή αυτή χρησιμοποιείται στον κυρίως βρόχο επικοινωνίας για τον έλεγχο της κίνησης του βραχίονα. Περιέχει τα δεδομένα για την κίνηση κάθε μίας από τις 7 (επτά) αρθρώσεις του βραχίονα.
3. "E" (control termination): Η εντολή αυτή, που δεν περιέχει δεδομένα, σηματοδοτεί τη λήξη της επικοινωνίας.
4. "T" (brake release): Η εντολή αυτή χορηγείται σε εξαιρετικές περιπτώσεις αντί της ‘S’, προκειμένου να ενεργοποιηθεί ή να απενεργοποιηθεί "χειροκίνητα" η πέδη μίας ή περισσοτέρων αρθρώσεων. Δεν περιέχει δεδομένα.
5. "M" (mechanism zero-point measure): Η εντολή αυτή χρησιμοποιείται για την αντιστοίχιση της τρέχουσας θέσης μίας οποιασδήποτε άρθρωσης στο σημείο μηδενός της. Συνοδεύεται από τον αύξοντα αριθμό της άρθρωσης.
6. "P" (EEPROM table contents change) και "R" (EEPROM table contents transmission request): Οι εντολές αυτές χρησιμοποιούνται για την ανάγνωση και εγγραφή των παραμέτρων που βρίσκονται αποθηκευμένες στην EEPROM του ελεγκτή. Συνοδεύονται από τα δεδομένα προσπέλασης της EEPROM.
7. "A" (alarm clear): Η εντολή αυτή επαναφέρει τον μικροεπεξεργαστή μίας οποιασδήποτε άρθρωσης σε λειτουργική κατάσταση μετά από την κατάσταση συναγερμού. Συνοδεύεται από τον αύξοντα αριθμό της άρθρωσης.

3.2 JR3 αισθητήρας δύναμης/ροπής

Εισαγωγή:

Οι αισθητήρες JR3 έχουν τη δυνατότητα να υπολογίζουν τη δύναμη σε τρεις ορθογώνιους άξονες, τη ροπή γύρω από κάθε άξονα, και να ορίζουν πλήρως τη φόρτιση στην εκάστοτε θέση του αισθητήρα. Χρησιμοποιούν ένα μοναδικό σχεδιασμό, όπου η βάση του αισθητήρα έχει ενσωματωμένες βίδες που επιτρέπουν την άμεση σύνδεση με τη φλάντζα του εργαλείου του χρήστη. Το πάνω μέρος του μεταλλάκτη είναι πανομοιότυπο με την αντίστοιχη φλάντζα του εργαλείου που απαιτείται. Αυτό το χαρακτηριστικό εξαλείφει την ανάγκη για επιπλέον διατάξεις προσαρμογής, εξοικονομώντας, έτσι, παράλληλα βάρος και μειώνοντας το μέγεθος της εγκατάστασης.

Τέλος, οι αισθητήρες JR3 είναι εξαιρετικά δύσκαμπτοι, με επακόλουθο να έχουν ελάχιστη επίδραση στην ακρίβεια της δυναμικής του συστήματος και του καθορισμού της θέσης του. Επίσης, με την υψηλή δυσκαμψία προκύπτει υψηλότερη συχνότητα συντονισμού απ' ό,τι επιτυγχάνεται με τους περισσότερους συμβατικούς αισθητήρες.

3.21 Αισθητήρας JR3 με ενσωματωμένα ηλεκτρονικά (onboard electronics)

Ο αισθητήρας δύναμης/ροπής JR3 που χρησιμοποιήθηκε παρουσιάζεται παρακάτω μαζί με τα βασικά του χαρακτηριστικά, όπως αυτά δίνονται από τον κατασκευαστή.

Εικόνα 1. Ο αισθητήρας δύναμης/ροπής JR3 μοντέλο 67M25A

Αισθητήρες με ενσωματωμένα ηλεκτρονικά		
Μοντέλο	Διαστάσεις	Μέγιστες φορτίσεις (lbs.) για x,y,z άξονα
67M25A	67 mm διαμ. 25 mm πάχος	15, 25, 50

Πίνακας 2. Βασικά χαρακτηριστικά του αισθητήρα.

Οι αισθητήρες JR3 με ενσωματωμένα ηλεκτρονικά (onboard electronics) χρησιμοποιούν την πιο πρόσφατη τεχνολογία για να παρέχουν δεδομένα δύναμης και ροπής σε πολύ υψηλό εύρος ζώνης (bandwidth) και με πολύ χαμηλά επίπεδα θορύβου. Τα σήματα από τα πιεζοηλεκτρικά ελάσματα (foil strain gauge) ενισχύονται και μετατρέπονται σε ψηφιακές απεικονίσεις της δύναμης και της ροπής "φορτώνοντας" δεδομένα από το ηλεκτρονικό

σύστημα που εμπεριέχεται στον αισθητήρα. Τα δεδομένα μεταδίδονται προς τον ηλεκτρονικό δέκτη σε σύγχρονη σειριακή μορφή. Όλα τα κυκλώματα αναλογικού σήματος χαμηλού επιπέδου, όπως και του αναλογοψηφιακού μετατροπέα (AD), βρίσκονται στο εσωτερικό σώμα του αισθητήρα προστατευμένα από πιθανή ηλεκτρομαγνητική επίδραση με το μεταλλικό σώμα του αισθητήρα.

Δεδομένα και από τους 6 (έξι) βαθμούς ελευθερίας του αισθητήρα επιστρέφονται με συχνότητα 8 kHz. Η ροή των ψηφιακών δεδομένων μπορεί να ελεγχθεί από πολλούς διαφορετικούς δέκτες ταυτόχρονα. Η χρήση πολλαπλών δεκτών επιτρέπει το σύστημα ελέγχου και παρακολούθησης ή συλλογής δεδομένων να εκτελείται από τελείως ξεχωριστά συστήματα.

Η ροή των δεδομένων εμπεριέχει πληροφορίες για χαρακτηριστικά του αισθητήρα και την αρχική βαθμονόμηση, με έλεγχο ανάδρασης την ηλεκτρική τάση του. Τα αποθηκευμένα δεδομένα βαθμονόμησής του επιτρέπουν την αλλαγή του αισθητήρα, χωρίς να είναι απαραίτητη η επαναρύθμιση του κυκλώματος του δέκτη. Η ανάδραση της ηλεκτρικής τάσης του μας επιτρέπει να χρησιμοποιούμε μεγάλα μήκη από μικρά σε διαστάσεις σύρματα μέσα στο καλώδιό του. Η τροφοδοσία και τα σήματα δεδομένων μπορούν να διέρχονται μέσω δακτυλιδιών επαφής ψήκτρας (slip rings), εάν αυτό είναι απαραίτητο.

3.22 Επεξεργασία Ψηφιακού Σήματος- ΕΨΣ (Digital Signal Processing - DSP)

Οι δέκτες JR3 βασισμένοι σε ΕΨΣ (DSP), που σχεδιάστηκαν για συνεργασία με τους JR3 αισθητήρες δύναμης/ροπής που μεταδίδουν σειριακά δεδομένα, χρησιμοποιούν επίσης την τελευταία τεχνολογία, για να παρέχουν δεδομένα δύναμης και ροπής σε 6 (έξι) βαθμούς ελευθερίας και σε πολύ υψηλό εύρος ζώνης (bandwidth). Η χρήση ενός ισχυρού ολοκληρωμένου κυκλώματος ΕΨΣ (DSP) επιτρέπει στο δέκτη JR3 να παρέχει απεμπλεγμένα και ψηφιακά φιλτραρισμένα δεδομένα σε συχνότητα 8 kHz ανά κανάλι. Αυτή η συχνότητα μετάδοσης δεδομένων είναι κατά μία τάξη μεγέθους μεγαλύτερη σε σχέση με αντίστοιχες που ήταν διαθέσιμες στο παρελθόν.

Η πλακέτα του δέκτη είναι πλέον διαθέσιμη και σε PCI σε αντίθεση με τις προηγούμενες εκδόσεις VME και ISA. Οι πλακέτες μοιράζονται κοινή αρχιτεκτονική, που συνίσταται σε έναν Επεξεργαστή Ψηφιακού Σήματος-ΕΨΣ (DSP) και σε ένα διπλής εισόδου διεύθυνσης (dual-ported address) μοιρασμένο χώρο 16k των 2 byte ανά λέξη, στον οποίο μπορούν να διαβάσουν και να γράψουν τόσο ο πάροχος (host) όσο και ο ΕΨΣ. Η χρήση μνήμης RAM διπλής εισόδου επιτρέπει στον πάροχο (host) να διαβάζει δεδομένα από τον ΕΨΣ με πολύ λιγότερο επιπλέον χρόνο (overhead) . Επίσης επιτρέπει στον πάροχο (host) να επαναδιαμορφώνει τον ΕΨΣ (DSP) άμεσα, μέσω εγγραφής εντολών διαμόρφωσης στη μνήμη RAM.

Η κάρτα του δέκτη περιέχει κύκλωμα για να δέχεται τη ροή των ψηφιακών δεδομένων από τον αισθητήρα, όπως και κύκλωμα για τον έλεγχο και τη ρύθμιση της παροχής ρεύματος στον αισθητήρα. Με τη ρύθμιση παροχής ρεύματος αυτόματα εξ αποστάσεως και με τα διαφορικά ψηφιακά σήματα δεδομένων που μεταδίδονται, οι προδιαγραφές των καλωδίων του αισθητήρα είναι αρκετά ελαστικές. Έτσι, δύσκαμπτα και ακριβά καλώδια δεν είναι πλέον απαραίτητα.

Ο Επεξεργαστής Ψηφιακού Σήματος-ΕΨΣ (DSP) χρησιμοποιείται για την επεξεργασία των δεδομένων που λαμβάνονται από τον αισθητήρα. Εκτελεί διάφορες λειτουργίες, όπως αφαίρεση αντιστάθμισης (offset removal), απέμπλεξη δεδομένων (data decoupling), ανίχνευση φαινομένου κορεσμού (saturation detection), ψηφιακό χαμηλόσυχνο φιλτράρισμα (digital low pass filtering) και υπολογισμός του μεγέθους του διανύσματος της δύναμης. Τα επεξεργασμένα δεδομένα και οι διάφοροι καταχωρητές κατάστασης (status registers) εντοπίζονται σε συγκεκριμένες περιοχές της μνήμης RAM διπλής εισόδου.

3.23 Δέκτης διαύλου PCI

Η πλακέτα δέκτη διαύλου PCI, η οποία συνδέεται με τις περισσότερες θύρες PCI κλασικών τύπων προσωπικού υπολογιστή, είναι μια μικρή (περίπου στο μέγεθος της σύνδεσης) κάρτα των 5V και 33MHz. Καταλαμβάνει ένα τμήμα μεγέθους 512kbyte στο χώρο διευθύνσεων των PCI. Η μνήμη εύρους 16 bit του Επεξεργαστή Ψηφιακού Σήματος-ΕΨΣ (DSP) χαρτογραφείται σε ένα δίαυλο PCI εύρους 32 bit. Η προσπέλαση του δέκτη PCI γίνεται απλούστατα με τη μορφή μνήμης στον δίαυλο PCI. Δεν χρειάζεται ειδική διαμόρφωση από το χρήστη, καθώς ο δίαυλος PCI είναι ένας δίαυλος που αναγνωρίζεται αυτόματα από το σύστημα (plug and play)

Εικόνα 2. Η κάρτα PCI, δέκτης των σημάτων του αισθητήρα.

3.24 Σύνοψη της αρχιτεκτονικής δομής δέκτη JR3 με ΕΨΣ

Όπως αναφέραμε, ο χώρος της διπλής εισόδου διευθύνσεων αποτελείται από 16kb των 2 byte ανά λέξη. Τα πρώτα 8kb αυτού του χώρου είναι μνήμη RAM, ενώ τα υπόλοιπα 8kb καταχωρητές κατάστασης και άλλα χαρακτηριστικά. Το μεγαλύτερο κομμάτι της δραστηριότητας του χρήστη πραγματοποιείται στις πρώτες 256 λέξεις του κοινού χώρου μνήμης. Λεπτομέρειες όσον αφορά στην ανάγνωση και εγγραφή από και προς τον κοινό χώρο μνήμης διαφοροποιούνται από δέκτη σε δέκτη.

Τα ανεπεξέργαστα δεδομένα από τον αισθητήρα διέρχονται από ένα μητρώο απεμπλοκής και αφαιρούνται οι αντισταθμίσεις. Αυτή η διαδικασία αφαιρεί τόσο διαρροές (cross-coupling) όσο και το βάρος αντιστάθμισης. Ένα παράπλευρο αποτέλεσμα είναι ότι καθίσταται δύσκολο να διαπιστώσουμε εάν ο αναλογοψηφιακός μετατροπέας (ADC) του μεταλλάκτη δύναμης/ροπής έχει κορεστεί. Εάν ο ADC έχει κορεστεί, ο βρόχος ανάδρασης που χρησιμοποιεί τα δεδομένα της δύναμης καθίσταται ασταθής. Για την ειδοποίηση του χρήστη γι' αυτή την κατάσταση, ο JR3 Επεξεργαστής Ψηφιακού Σήματος-ΕΨΣ (DSP) ελέγχει τα ανεπεξέργαστα δεδομένα και υποδεικνύει πότε τα δεδομένα του αισθητήρα πλησιάζουν τον κορεσμό και πότε είναι κορεσμένα.

Τα απεμπλεγμένα δεδομένα διέρχονται μέσω διαδοχικών χαμηλόσυχνων φίλτρων (low-pass filters). Κάθε ένα από τα διαδοχικά φίλτρα υπολογίζεται στο 1/4 και έχει συχνότητα αποκοπής (cutoff frequency) στο 1/4 σε σχέση με το αμέσως προηγούμενο φίλτρο. Η συχνότητα αποκοπής ενός φίλτρου είναι το 1/16 από τη συχνότητα δειγματοληψίας (sample rate) του ίδιου του φίλτρου. Για έναν τυπικό αισθητήρα με συχνότητα δειγματοληψίας τα 8kHz η συχνότητα αποκοπής για το πρώτο φίλτρο θα είναι 500Hz. Αντίστοιχα τα επόμενα φίλτρα θα έκοβαν τις συχνότητες στα 125 Hz, 31.25 Hz, 7.81 Hz κ.λπ. Η καθυστέρηση μέσω του φίλτρου είναι περίπου ίση με:

$$\text{Καθυστέρηση} \cong \frac{1}{\text{Συχνότητα Αποκοπής}}$$

Συνεπώς η καθυστέρηση μέσω ενός φίλτρου των 500Hz θα είναι:

$$\frac{1}{500Hz} \cong 2ms$$

Επειδή η καθυστέρηση διαφοροποιείται ανάλογα με τη συχνότητα των δεδομένων, αυτή η τιμή, αν και είναι μία πολύ καλή προσέγγιση, δεν είναι ακριβής. Για καλύτερη περιγραφή

των χαρακτηριστικών των φίλτρων προτείνεται μελέτη των γραφημάτων που ακολουθούν και περιγράφουν την απόκριση των φίλτρων.

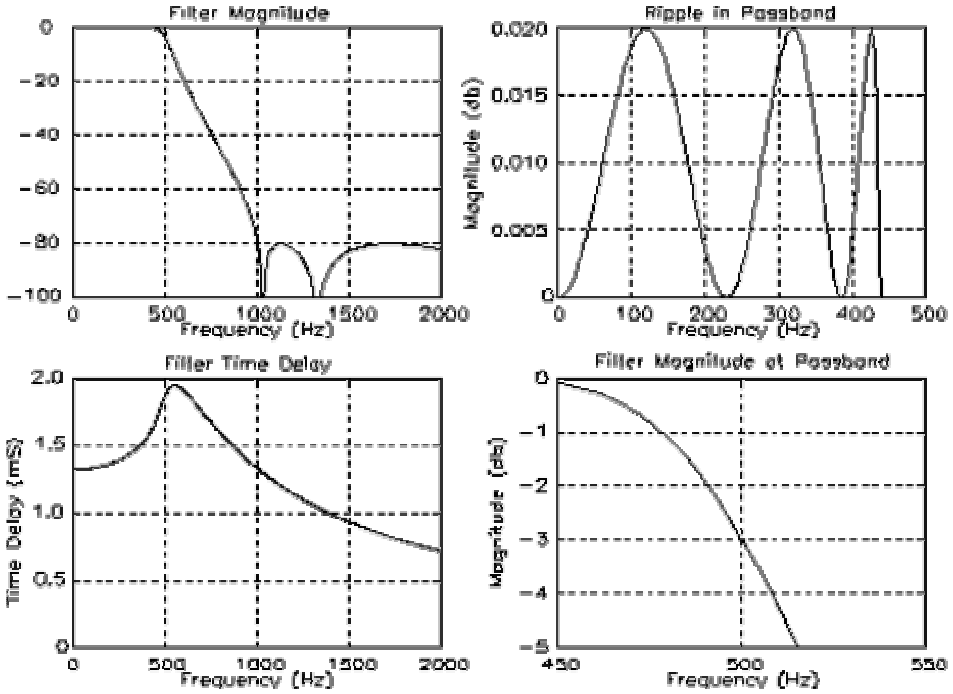
Το μέγεθος των διανυσμάτων δύναμης και ροπής υπολογίζεται από κάθε πακέτο δεδομένων. Επομένως, υπάρχουν διανύσματα που υπολογίζονται από απεμπλεγμένα αφιλτράριστα δεδομένα, όπως και από κάθε πακέτο φιλτραρισμένων δεδομένων. Τα αφιλτράριστα διανύσματα υπολογίζονται στο 1/2 του εύρους ζώνης (bandwidth) των αφιλτράριστων δεδομένων. Τα διανύσματα που υπολογίζονται από τα φιλτραρισμένα δεδομένα υπολογίζονται στο τετραπλάσιο της συχνότητας αποκοπής του φίλτρου ή στο 1/4, όταν υπολογίζεται και το φίλτρο.

Κάθε πακέτο δεδομένων (8 τιμές) μπορεί να ελεγχθεί για κορυφές και κοιλάδες. Τα δεδομένα ελέγχονται σε πλήρες εύρος ζώνης (full bandwidth) και, εάν ανιχνευτεί νέο ελάχιστο ή μέγιστο τότε αυτό αποθηκεύεται. Τα μέγιστα και ελάχιστα μπορούν να αναγνωστούν ή και να επαναφερθούν στις αρχικές συνθήκες υπό την καθοδήγηση του χρήστη. Κάθε πακέτο δεδομένων (8 τιμές) μπορεί να χρησιμοποιηθεί για υπολογισμούς. Με ένα καθορισμένο από το χρήστη χρονικό διάστημα είναι εφικτό να υπολογίζεται και να αναφέρεται και η πρώτη παράγωγος του πακέτου δεδομένων.

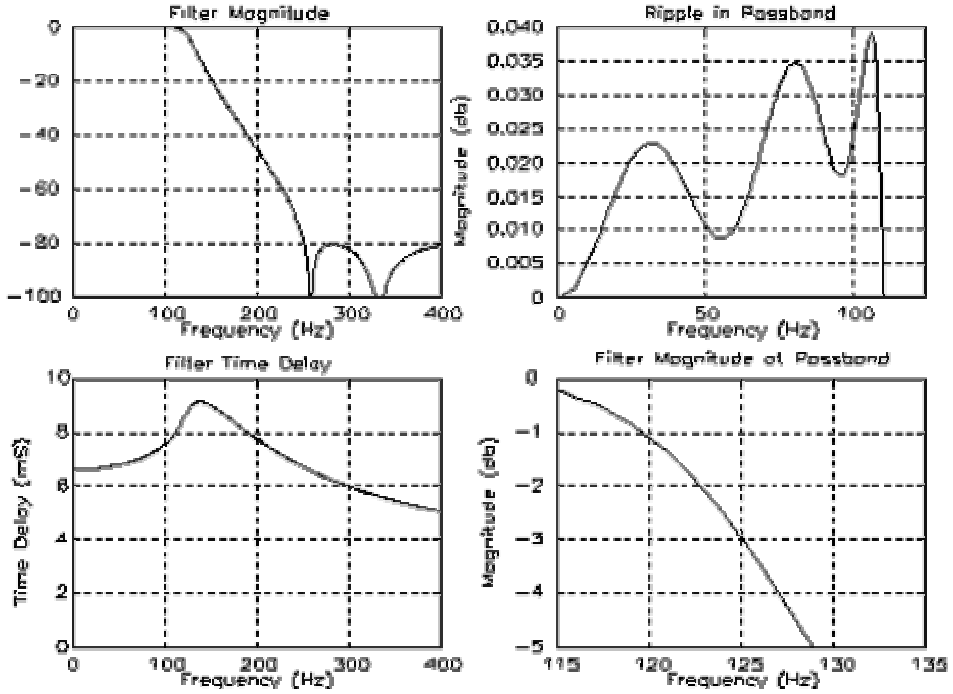
Ο χρήστης μπορεί να εφαρμόσει μετασχηματισμούς του συστήματος συντεταγμένων του αισθητήρα και να θέσει οποιαδήποτε αρχή του συστήματος συντεταγμένων και οποιονδήποτε προσανατολισμό. Αυτό επιτρέπει, για παράδειγμα, στο χρήστη να ευθυγραμμίσει τους άξονες του αισθητήρα δύναμης/ροπής με αυτούς του εργαλείου που χρησιμοποιεί, γεγονός που μπορεί να απλοποιήσει αισθητά τους μαθηματικούς υπολογισμούς που χρειάζονται για την υλοποίηση ενός ελέγχου δύναμης/ροπής.

Ο JR3 Επεξεργαστής Ψηφιακού Σήματος-ΕΨΣ (DSP) μπορεί να ρυθμιστεί από το χρήστη, ώστε να ελέγχει το όριο πάνω από το οποίο μία τιμή θεωρείται αποδεκτή (threshold). Έτσι ο JR3 ΕΨΣ αντιδρά μόνο όταν κάποιο δεδομένο ξεπερνά την προκαθορισμένη αυτή τιμή. Αυτό μας δίνει τη δυνατότητα να αποφεύγουμε την επεξεργασία μικρών τιμών φορτίσεων λόγω σφάλματος, με ελάχιστη καθυστέρηση (overhead).

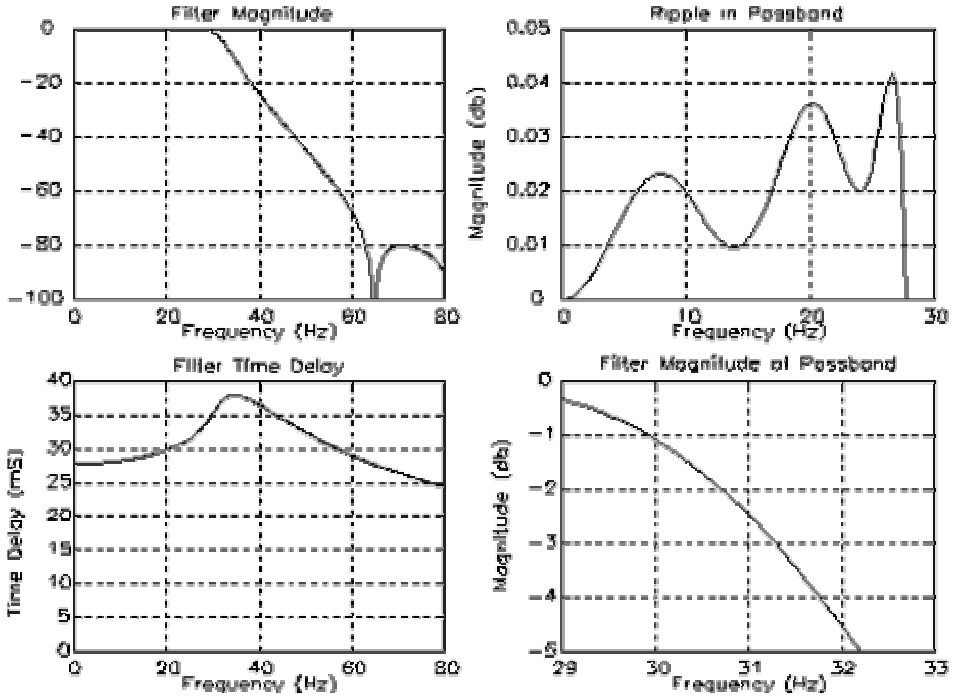
Τα διαγράμματα που ακολουθούν παρουσιάζουν τα χαρακτηριστικά των ψηφιακών φίλτρων. Παρουσιάζεται η απόκριση των $3^{ων}$ (τριών) πρώτων φίλτρων. Η έξοδος των φίλτρων μεταβάλλεται ελάχιστα, όσο η συχνότητα του φίλτρου πέφτει εξαιτίας της διαδοχικότητας των φίλτρων. Τα δεδομένα για τα φίλτρα μετά το τρίτο είναι ουσιαστικά τα ίδια με το φίλτρο No.3 και συνεπώς δεν παρουσιάζονται. Η μόνη διαφορά για τα φίλτρα No.4 - No.6 είναι πως ο άξονας των συχνοτήτων θα έπρεπε να διαιρεθεί με το 4 (τέσσερα) για κάθε διαδοχικό φίλτρο από το γράφημα για το φίλτρο No.3



Διάγραμμα 1. Χαρακτηριστικές φίλτρου για το φίλτρο No.1 με δεδομένα αισθητήρα στα 8Khz.



Διάγραμμα 2. Χαρακτηριστικές φίλτρου για το φίλτρο No.2 με δεδομένα αισθητήρα στα 8Khz.



Διάγραμμα 3. Χαρακτηριστικές φίλτρου για το φίλτρο No.3 με δεδομένα αισθητήρα στα 8Khz.

3.3 Σύνδεση αισθητήρα JR3 με τον ακροδέκτη του βραχίονα PA-10

Εισαγωγή:

Για να μπορέσουμε να εκμεταλλευτούμε τις δυνατότητες του αισθητήρα μας σε έναν πειραματικό έλεγχο δύναμης/ροπής ρομποτικού βραχίονα, πρέπει να εξασφαλίσουμε την επιτυχημένη και σταθερή σύνδεσή του με αυτόν με όσο το δυνατόν πιο απλή, ελαφριά (για να μην επηρεάζει πολύ τη δυναμική του συστήματός μας), αλλά και αρκούντως στιβαρή κατασκευή. Στην προκειμένη περίπτωση ήταν απαραίτητη η κατασκευή μιας φλάντζας πάνω στην οποία θα μπορούσαμε να βιδώσουμε στο κέντρο της τον JR3 αισθητήρα μας και μέσω του εξωτερικού δακτυλίου της θα συνδέαμε πάλι με κοχλίες το σύστημά μας με την αντίστοιχη φλάντζα του ακροδέκτη του ρομποτικού βραχίονα.

Η ελαχιστοποίηση του βάρους μίας κατασκευής κρίνεται και από τον τελικό σχεδιασμό της, αλλά φυσικά και από την επιλογή του υλικού της. Όλες όμως αυτές οι παράμετροι κρίνουν σημαντικά την τελική αντοχή και ευκαμψία τού προς κατασκευή τεμαχίου. Για την αποδοχή του τελικού σχεδίου έγινε θεωρητικός έλεγχος αντοχής μέσω ενός από τα πιο σύγχρονα προγράμματα ανάλυσης κατασκευών με πεπερασμένα στοιχεία.

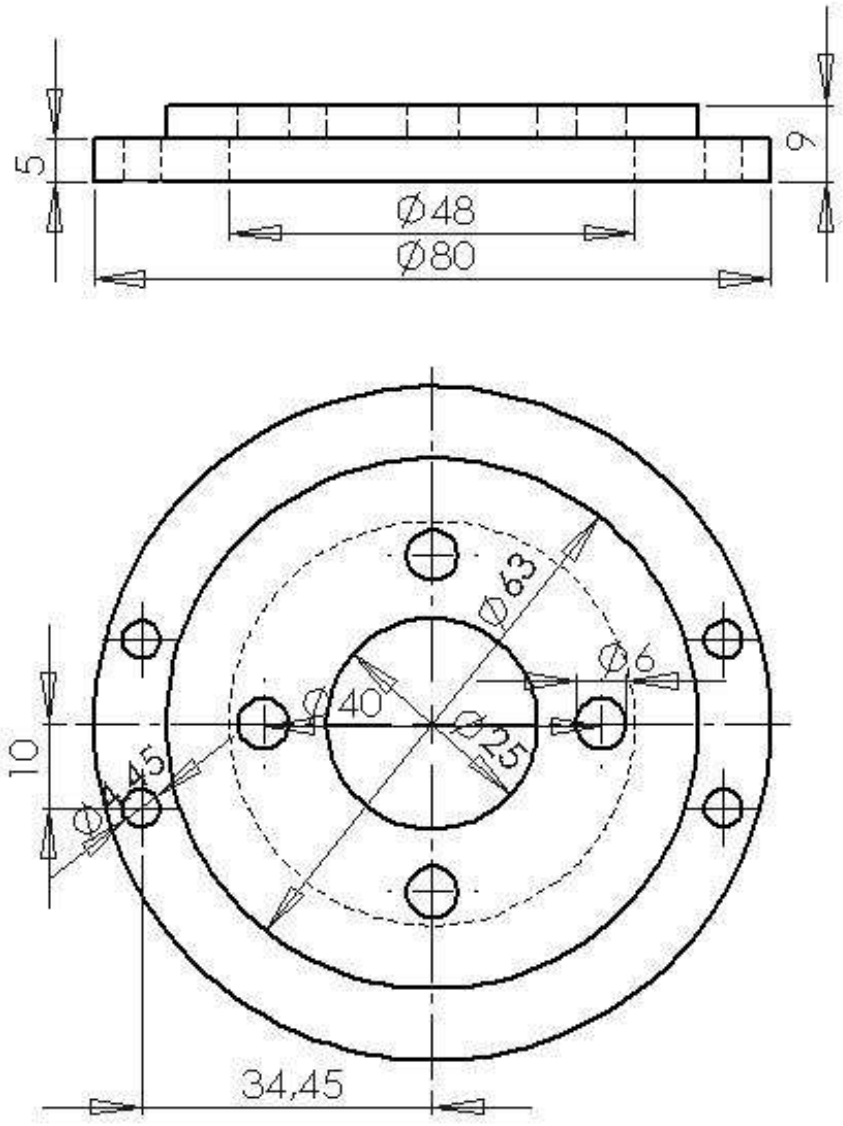
Τέλος, αναλύεται η κατασκευή του τελικού τεμαχίου και οι διορθωτικές κινήσεις και κατεργασίες που προέκυψαν, τόσο λόγω κάποιων αρχικών κατασκευαστικών σφαλμάτων της νέας φλάντζας, όσο και κάποιων ατελειών της ενδιάμεσης φλάντζας του ακροδέκτη του ρομποτικού βραχίονα.

Τα απαραίτητα εξαρτήματα και τα βήματα που πρέπει να ακολουθηθούν για την τελική σύνδεση του αισθητήρα με τον ρομποτικό βραχίονα παρουσιάζονται αναλυτικά στο αντίστοιχο κεφάλαιο του εγχειριδίου στο Παράρτημα Γ.3.

3.31 Σχεδιασμός φλάντζας σύνδεσης και επιλογή υλικού

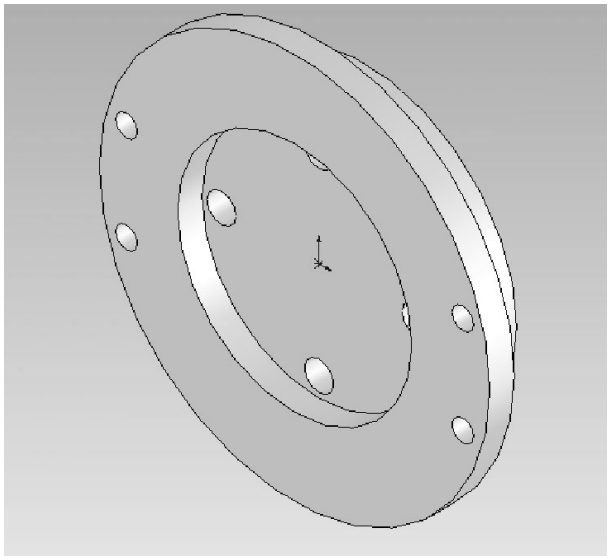
Ο σχεδιασμός της φλάντζας σύνδεσης καθορίζεται σχεδόν πλήρως από τις επιφάνειες των εξαρτημάτων που καλείται να συνδέσει. Από τη μία πλευρά της απαιτείται η συνεργασία με τον ακροδέκτη του ρομποτικού βραχίονα και από την άλλη με τον αισθητήρα δύναμης/ροπής.

Ο σχεδιασμός έγινε με βάση τα παραπάνω κριτήρια και με γνώμονα την ελαχιστοποίηση του χρησιμοποιούμενου υλικού, χωρίς όμως να παραμελούμε τη στιβαρότητα της τελικής κατασκευής. Το αναλυτικό μηχανολογικό σχέδιο της κατασκευής με πλήρεις διαστάσεις παρουσιάζεται στο Σχήμα 8.

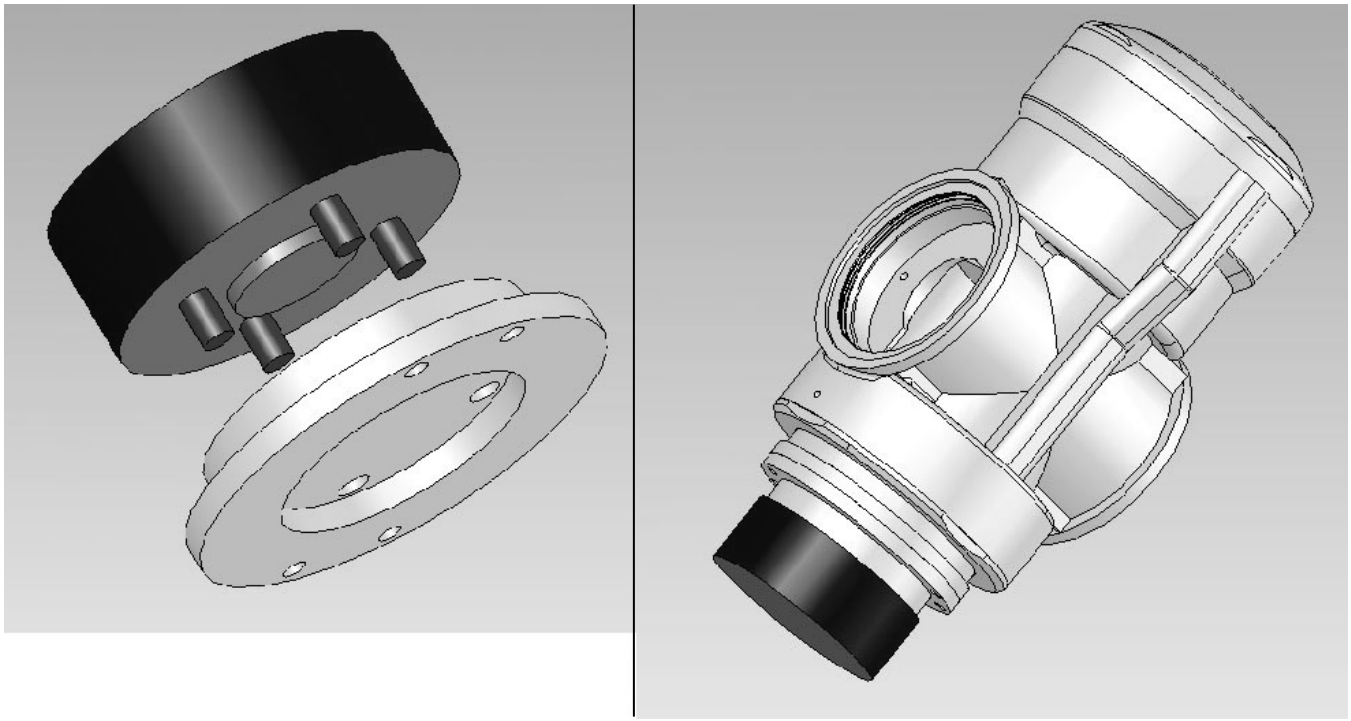


Σχήμα 8. Μηχανολογικό κατασκευαστικό σχέδιο της φλάντζας σύνδεσης.

Ο σχεδιασμός έγινε αρχικά τρισδιάστατα με τη βοήθεια του σχεδιαστικού λογισμικού πακέτου SolidWorks πριν το σχέδιο εξαχθεί σε δισδιάστατο μηχανολογικό σχέδιο με ονομαστικές διαστάσεις. Στα σχήματα που ακολουθούν περιγράφεται η τρισδιάστατη απεικόνιση της φλάντζας σύνδεσης με ισομετρική άποψη, καθώς και η συνεργασία της φλάντζας με τον αισθητήρα δύναμης/ροπής και τον ακροδέκτη του ρομποτικού βραχίονα. Τα σχέδια έγιναν εισαγωγή στην υπολογιστική σχεδιαστική εφαρμογή, για να ελεγχθεί η αρχική σχεδίαση της φλάντζας μέσω μιας εικονικής συναρμολόγησης.

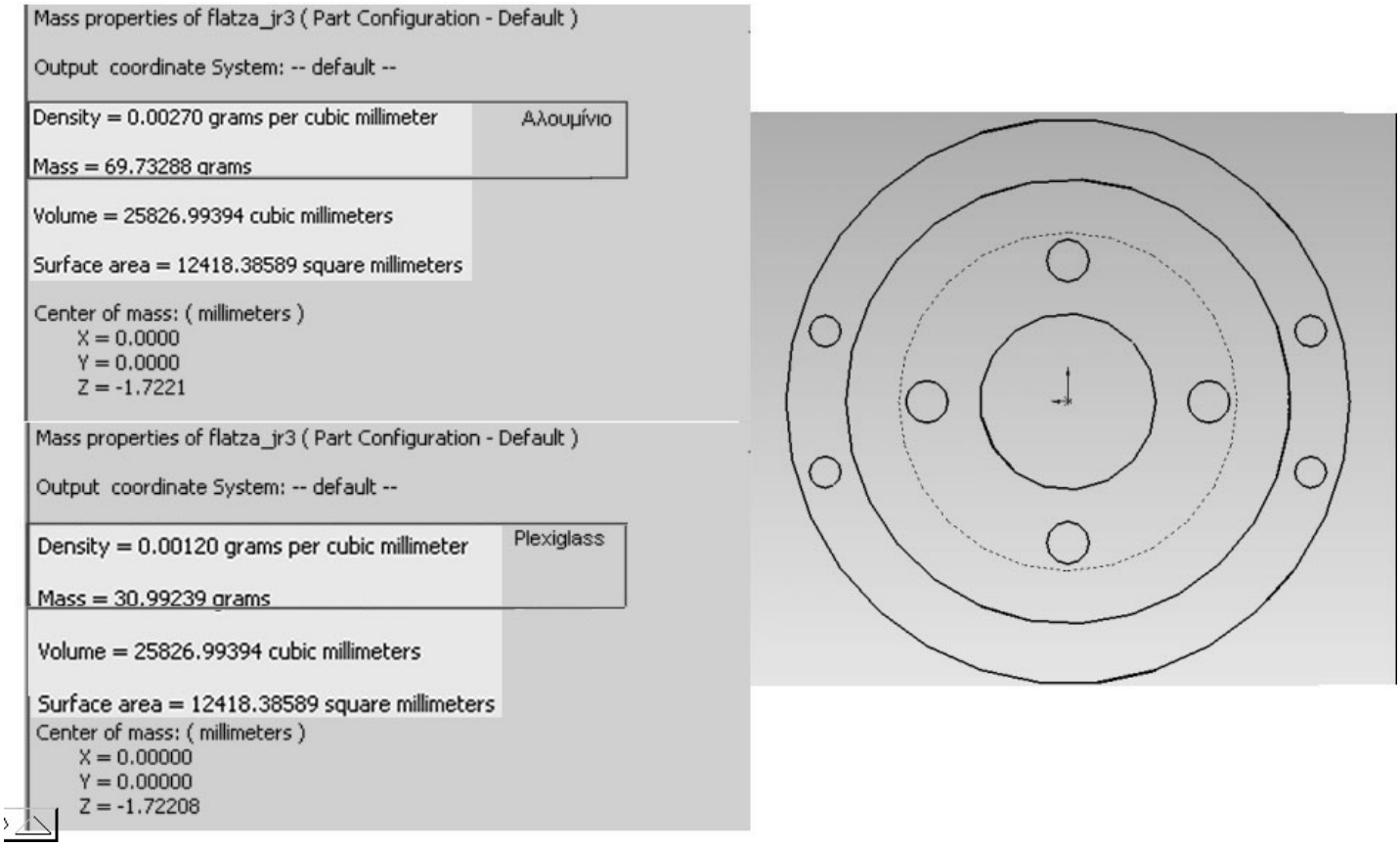


Σχήμα 9. Ισομετρική τρισδιάστατη απεικόνιση της φλάντζας σύνδεσης.



Σχήμα 10. Τρισδιάστατη απεικόνιση εικονικής συναρμολόγησης αισθητήρα και ακροδέκτη μέσω της φλάντζας σύνδεσης .

Επίσης, με τη χρήση των κατάλληλων εργαλείων και επιλογών είχαμε τη δυνατότητα να εκτιμήσουμε τον όγκο του τελικού κομματιού και κατά συνέπεια το συνολικό του βάρος εισάγοντας την πυκνότητα του υλικού που επιθυμούμε. Έγινε υπολογισμός για Αλουμίνιο και Πλεξιγκλάς (Plexiglas) με το δεύτερο να υπολογίζεται 2.3 φορές ελαφρύτερο.



Σχήμα 11. Χαρακτηριστικά της φλάντζας σύνδεσης σε σχέση και με το υλικό κατασκευής.

Παρά το μικρότερο βάρος του υλικού Πλεξιγκλάς, μια αλουμινένια κατασκευή κρίνεται αρκούντως ελαφριά και επιλέγεται και λόγω των υπόλοιπων μηχανικών ιδιοτήτων του για τη συγκεκριμένη εφαρμογή, στην οποία θα παρεμβάλλεται μεταξύ δύο εξίσου μεταλλικών επιφανειών. Οι υπόλοιπες μηχανικές ιδιότητες του κράματος αλουμινίου που επιλέχθηκε αναλύονται στον Πίνακα 3:

Value	Property
69000 N/mm^2	Elastic Modulus
0.33	Poissons Ratio
27000 N/mm^2	Shear Modulus
2.4e-005	Thermal Expansion Coefficient
0.0027 g/mm^3	Density
200 W/m K	Thermal Conductivity
900 J/kg K	Specific Heat
68.9356 N/mm^2	Tensile Strength
27.5742 N/mm^2	Yield Strength

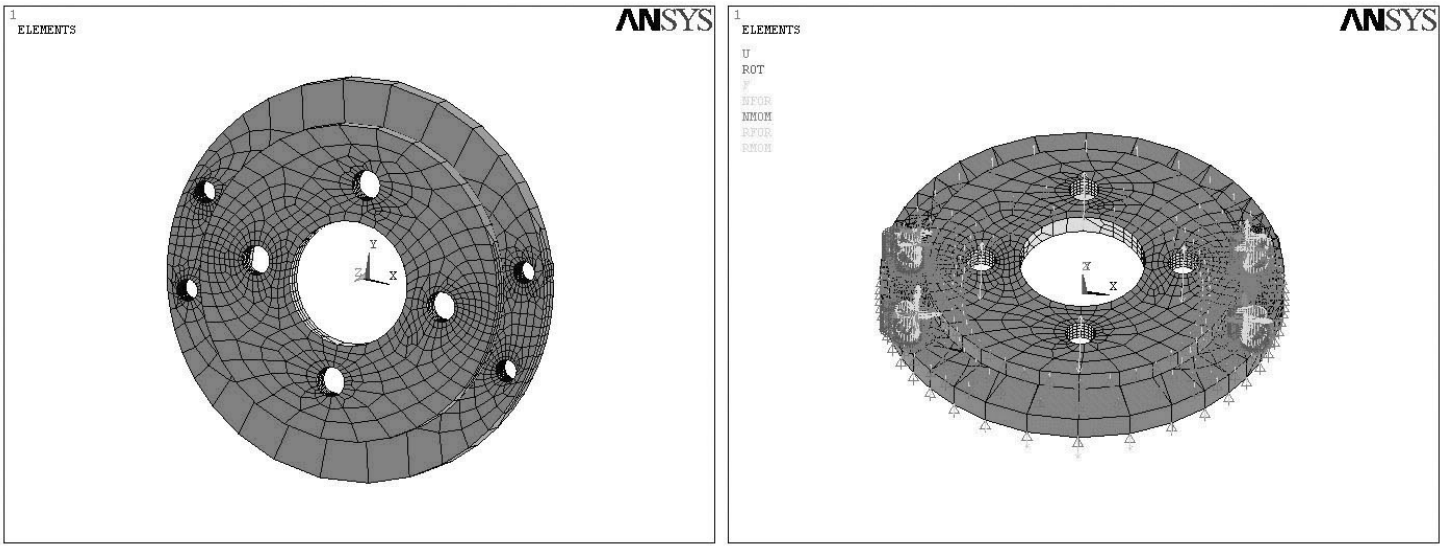
Πίνακας 3. Μηχανικές ιδιότητες απλού κράματος αλουμινίου.

3.32 Θεωρητικός έλεγχος αντοχής με χρήση πεπερασμένων στοιχείων

Με βάση τις μηχανικές ιδιότητες του κράματος αλουμινίου που επιλέχθηκε για την κατασκευή της φλάντζας σύνδεσης και τα πλήρη κατασκευαστικά του χαρακτηριστικά, κρίθηκε απαραίτητος ένας τυπικός έλεγχος αντοχής σε θεωρητικό επίπεδο, συνυπολογίζοντας φορτίσεις μεγαλύτερες από τις μέγιστες αναμενόμενες σε μια τυπική εφαρμογή, με σκοπό την επίτευξη της απαραίτητης ασφάλειας.

Η διαδικασία αυτή μπορούσε να επιταχυνθεί και να βελτιωθεί χρησιμοποιώντας ένα από τα πιο σύγχρονα πακέτα λογισμικού CAD - CAE[1] με χρήση πεπερασμένων στοιχείων, το ANSYS. Πρώτο στάδιο ήταν η αποθήκευση του μηχανολογικού μας σχεδίου μέσω του προγράμματος SolidWorks σε μορφή αρχείου που θα μπορούσε να χρησιμοποιηθεί από το ANSYS χωρίς κάποια απώλεια δεδομένων. Η μορφή που επιλέχτηκε ήταν του τύπου ".IGS". Βέβαια το ANSYS μας παρείχε εξίσου τη δυνατότητα της εκ νέου σχεδίασης του υπό έλεγχο αντοχής τεμαχίου, αλλά προτιμήσαμε το ήδη ελεγμένο σχέδιο για την αποφυγή πιθανών παραλείψεων.

Πρώτο μας μέλημα η κατασκευή του πλέγματος των πεπερασμένων στοιχείων στην εικονική μας φλάντζα με χρήση τετράπλευρων στοιχείων αυτόματου μεγέθους, ανάλογα με την ακρίβεια που απαιτείται σε κάθε περιοχή του τεμαχίου. Το αποτέλεσμα ήταν η ύπαρξη πυκνότερου πλέγματος στις περιοχές που θα εφαρμόσουν οι κοχλιώσεις και αναμένονται μεγαλύτερες και πολυπλοκότερες φορτίσεις και αραιότερου πλέγματος στις πιο απομακρυσμένες από τις οπές αυτές περιοχές. Για την πληρέστερη ανάλυση του φαινομένου και την αποφυγή λανθασμένων υπολογισμών παραμετροποιήθηκαν και οι στηρίξεις της φλάντζας, τόσο λόγω των κοχλιών, όσο και των επιφανειών επαφής (βλ. Σχήμα 12).

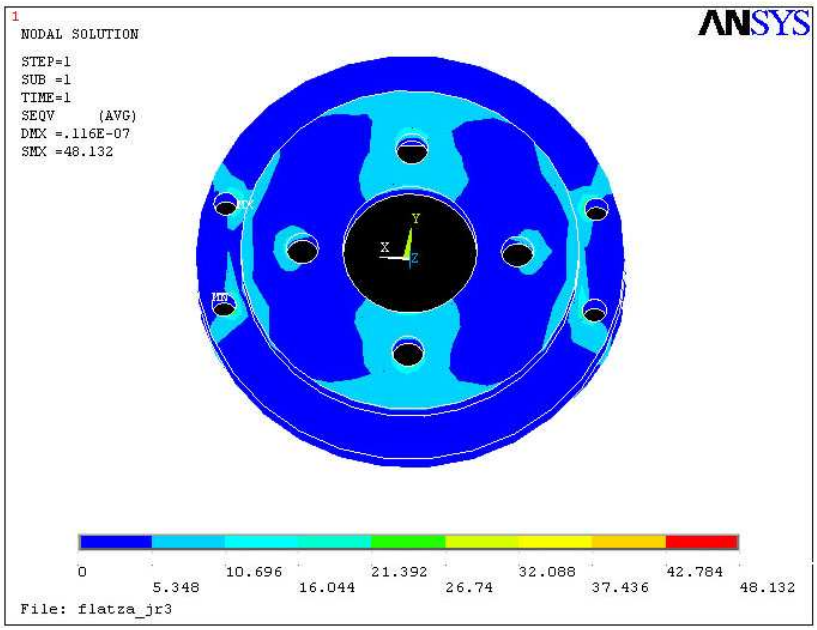


Σχήμα 12. Πλέγμα πεπερασμένων στοιχείων και στηρίξεις του εικονικού ελέγχου αντοχής της φλάντζας.

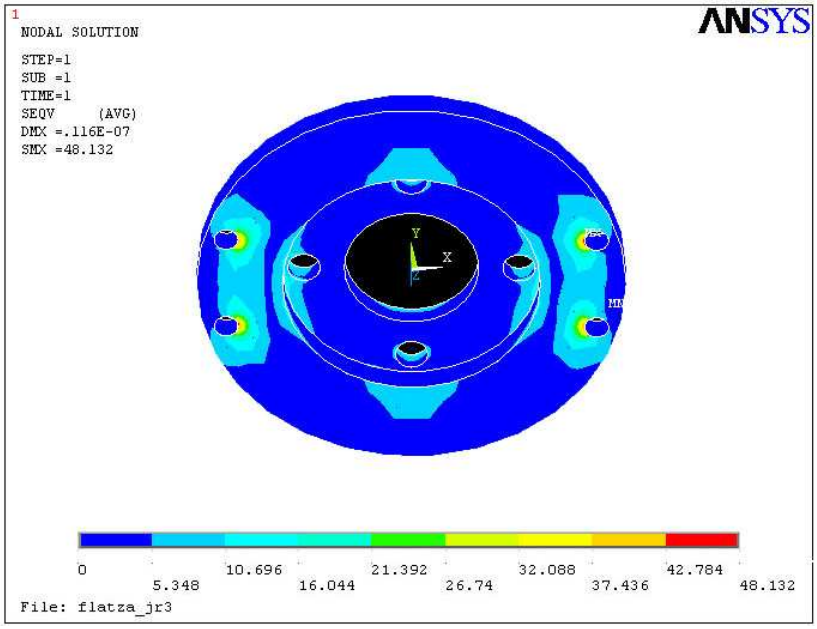
[1] *CAD : Computer Aided Design, CAE : Computer Aided Engineering,* δηλαδή σχεδιασμός και εφαρμογές μηχανικής με τη βοήθεια υπολογιστή.

Τέλος εισήχθησαν στο σύστημα εικονικές δυνάμεις της τάξεως των 100 N περίπου, που όπως αναφέρθηκε είναι μεγαλύτερες από τις αναμενόμενες που θα εφαρμοστούν κατά τη διάρκεια των πειραματικών εφαρμογών.

Η επίλυση του μοντέλου και η γραφική απεικόνιση των τάσεων από τις δύο πλευρές της φλάντζας, οι οποίες συνδέονται με το ρομποτικό βραχίονα και τον αισθητήρα δύναμης/ροπής αντίστοιχα, παρουσιάζονται στα Σχήματα 13 και 14.

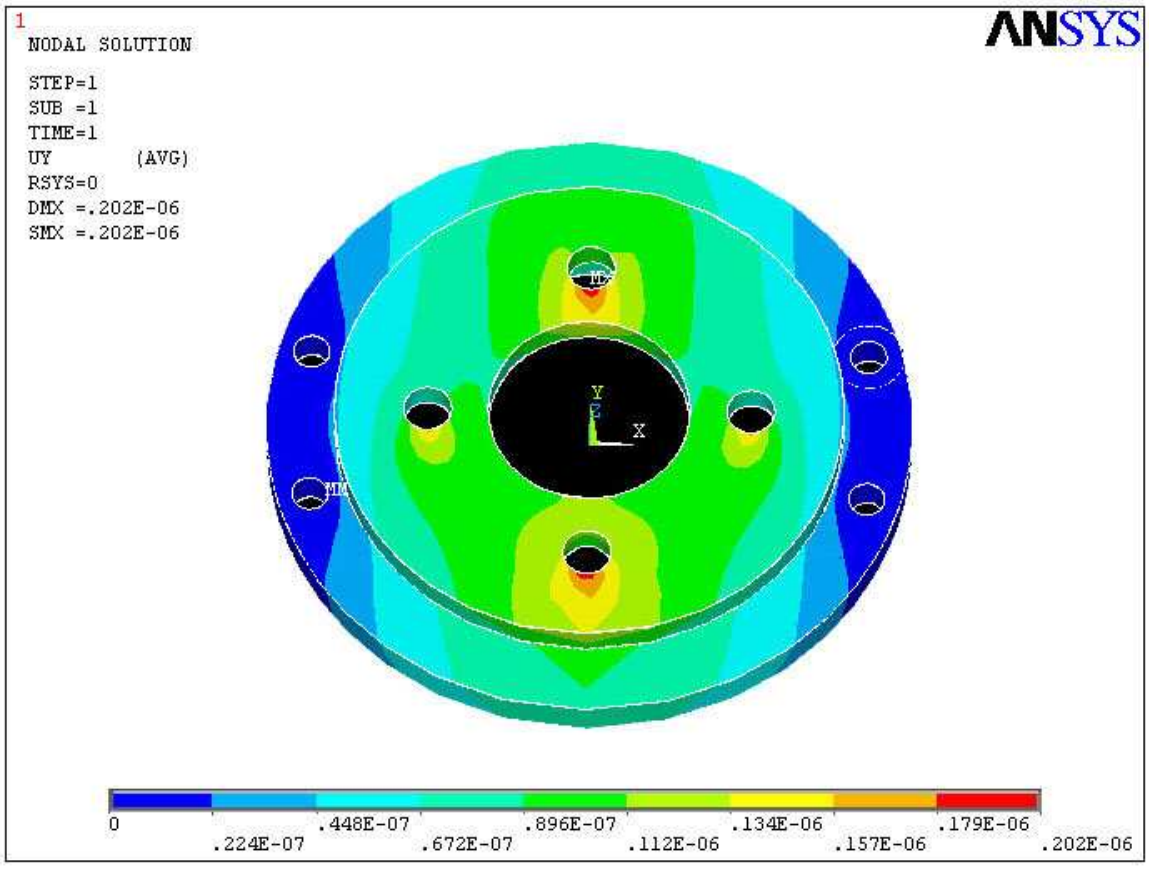


Σχήμα 13. Απεικόνιση τάσεων της φλάντζας στην πλευρά σύνδεσης με τον αισθητήρα με εφαρμογή κάθετης φόρτισης ~100 N.



Σχήμα 14. Απεικόνιση τάσεων της φλάντζας στην πλευρά σύνδεσης με τον ακροδέκτη του ρομποτικού βραχίονα με εφαρμογή κάθετης φόρτισης ~100 N.

Η κλιμάκωση των χρωμάτων από το μπλε στο ερυθρό αντιπροσωπεύουν την κλιμάκωση της εφαρμοσμένης τάσης στη φλάντζα. Η μέγιστη τιμή που εμφανίζεται στην περιοχή που έχουμε συνεργασία της κοχλίωσης του αισθητήρα με τη φλάντζα μας οδηγεί στο συμπέρασμα πως, παρά το μέγεθος της φόρτισης, όλες οι τάσεις στη χρησιμοποιούμενη φλάντζα βρίσκονται εντός ορίων της ελαστικής περιοχής παραμόρφωσης. Επίσης, οι παραμορφώσεις της φλάντζας μελετήθηκαν από τα αντίστοιχα διαγράμματα, όπως παρουσιάζεται για παράδειγμα στο Σχήμα 15 η κατανομή των παραμορφώσεων για εφαρμογή δύναμης περίπου 100 N στον άξονα y (παράλληλα με τη μετωπική επιφάνεια της φλάντζας). Ακόμα και οι μέγιστες τιμές είναι αρκετά μικρές, ώστε να θεωρείται η φλάντζα μας τυπικά απαραμόρφωτη στο εύρος των εφαρμογών που πρόκειται να χρησιμοποιηθεί.



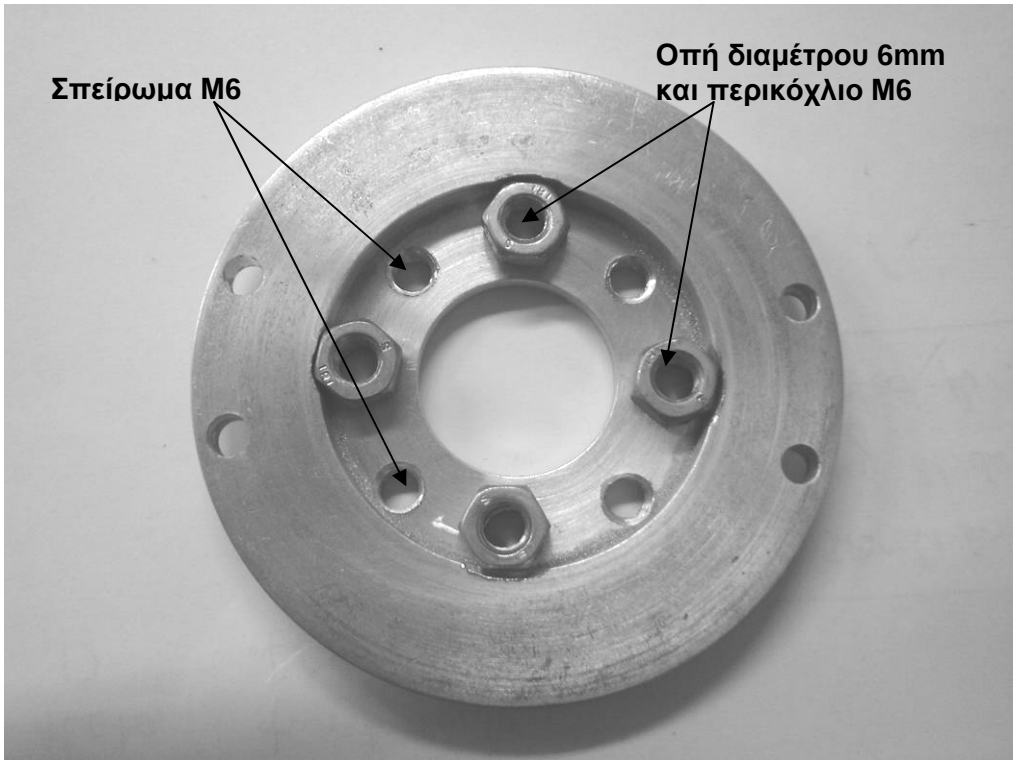
Σχήμα 15. Απεικόνιση παραμορφώσεων της φλάντζας με εφαρμογή παράλληλης δύναμης προς τη μετωπική επιφάνεια (άξονας y) ~ 100 N.

3.33 Κατασκευή και επεξεργασία αλουμινένιας φλάντζας

Τα σχέδια της φλάντζας δόθηκαν σε μηχανουργείο για την κατασκευή δύο ίδιων κομματιών, ένα για τη σύνδεση του αισθητήρα με τον ακροδέκτη του ρομποτικού βραχίονα και ένα για πιθανή πρόσδεσή του στην εξωτερική πλευρά του αισθητήρα. Όπως έχουμε ήδη αναφέρει, στις δύο πλευρές του αισθητήρα υπάρχει μια αντιστοιχία διαστάσεων που μας επιτρέπει να προσδέσουμε το ίδιο εξάρτημα και στις δύο πλευρές.

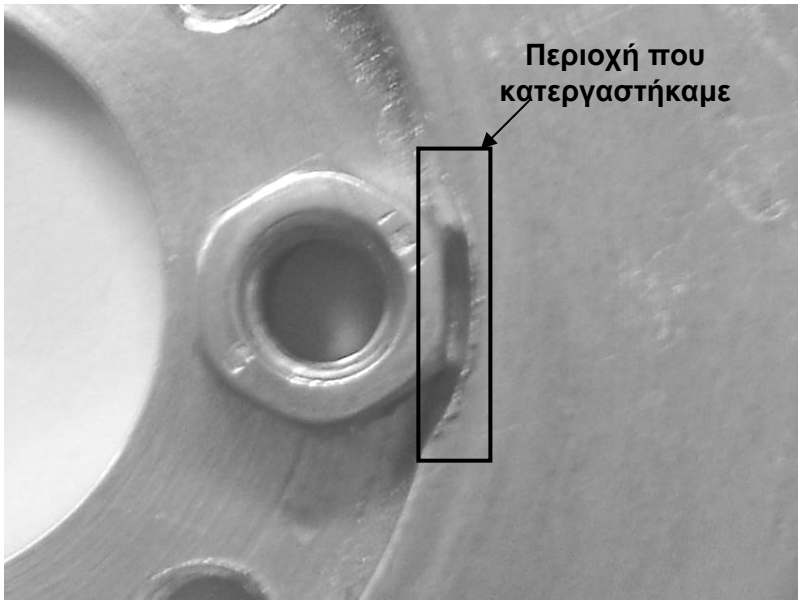
Η φλάντζα που κατασκευάστηκε διέθετε σπείρωμα M6 πάνω στο οποίο μπορούσαμε να βιδώσουμε τον αισθητήρα μέσω των ενσωματωμένων κοχλιών που διέθετε. Κατασκευαστική ατέλεια όμως δεν επέτρεπε τη σωστή συναρμογή μεταξύ της φλάντζας και του αισθητήρα, καθώς δεν συνέπιπταν πλήρως οι άξονες των δύο τεμαχίων.

Η λύση που δόθηκε ήταν η διάνοιξη νέων οπών διαμέτρου 6 mm συμμετρικά ως προς τις αρχικές οπές και η χρήση περικοχλίων πάνω στα οποία βιδώσαμε τους ενσωματωμένους κοχλίες του αισθητήρα. Η κάτω πλευρά της φλάντζας όπου φαίνεται καθαρά η τελική της μορφή παρουσιάζεται στην Εικόνα 3.



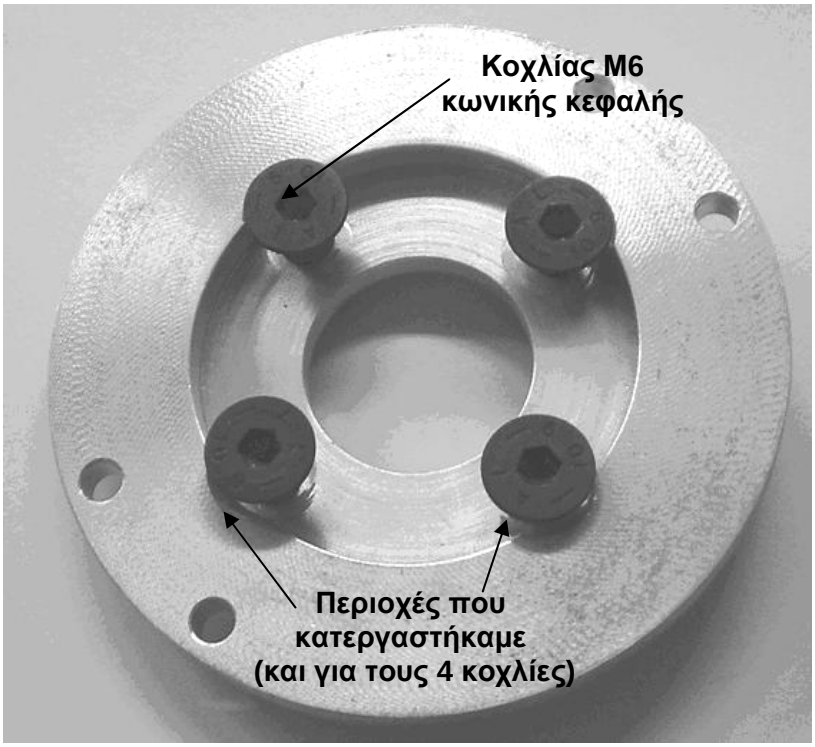
Εικόνα 3. Απεικόνιση της κάτω πλευράς της φλάντζας και παρουσίαση των οπών για τη σύνδεση του αισθητήρα.

Για να μπορούν να τοποθετηθούν σωστά τα περικόχλια έγινε κατεργασία αφαίρεσης υλικού στον εσωτερικό δακτύλιο της φλάντζας στις περιοχές δίπλα στις οπές, όπως παρουσιάζεται στην Εικόνα 4.



Εικόνα 4. Λεπτομέρεια κατεργασίας φλάντζας σύνδεσης για την προσαρμογή του περικοχλίου.

Ανάλογη κατεργασία έγινε και στη βοηθητική φλάντζα για να μπορούν να περνούν οι κοχλίες M6 κωνικής κεφαλής που χρησιμοποιούνται για τη σύνδεση της βοηθητικής φλάντζας με την εξωτερική επιφάνεια του αισθητήρα.

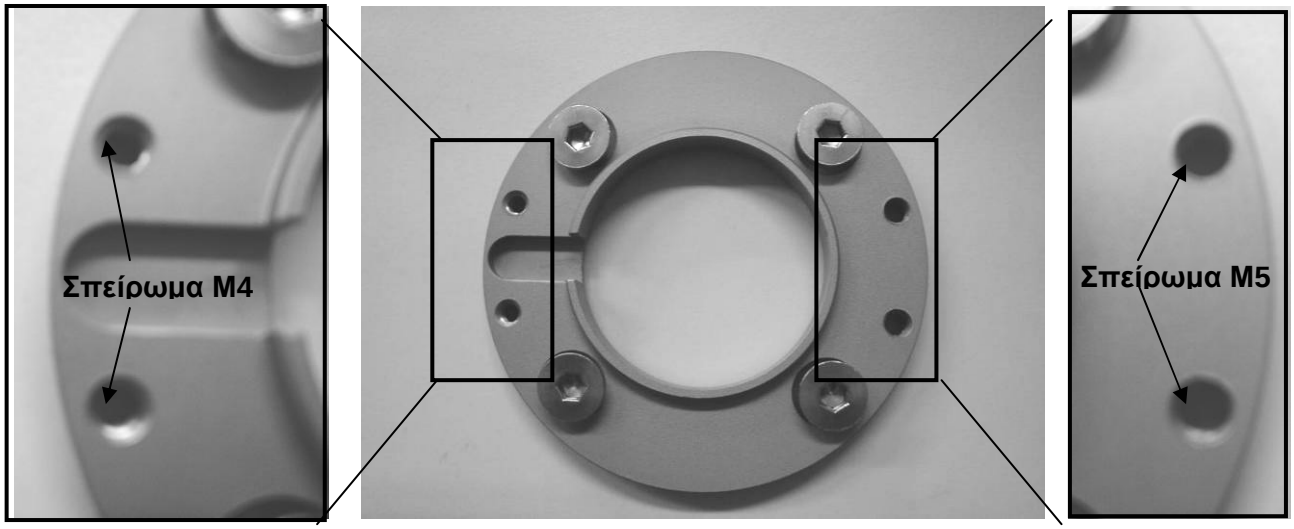


Εικόνα 5. Βοηθητική φλάντζα με τους κατάλληλους κοχλίες και την κατεργασία για την προσαρμογή τους.

Ένα άλλο πρόβλημα που αντιμετωπίστηκε ήταν η ύπαρξη δύο διαφορετικών σπειρωμάτων στη φλάντζα του ρομπότ πάνω στην οποία θέλαμε να συνδέσουμε τα

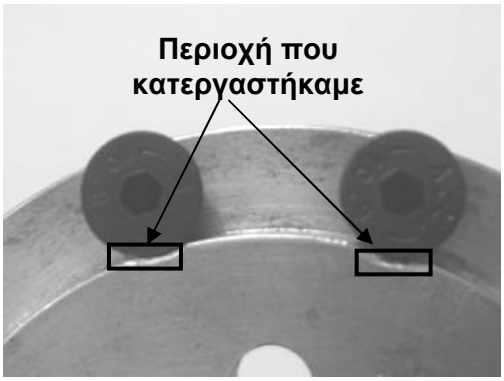
εξαρτήματά μας. Από τη μία πλευρά είχαμε μετρικό σπείρωμα M4 και από την άλλη άγνωστο αμερικανικής τυποποίησης σπείρωμα λίγο μεγαλύτερης διάστασης για το οποίο αδυνατούσαμε να βρούμε τους κατάλληλους κοχλίες. Για αυτό ανοίξαμε νέο σπείρωμα (αναγκαστικά μεγαλύτερης διάστασης) M5.

Στην Εικόνα 6 παρουσιάζεται η φλάντζα του ρομπότ όπου σημειώνονται οι διαστάσεις των τελικών σπειρωμάτων.



Εικόνα 6. Φλάντζα του ακροδέκτη του ρομποτικού βραχίονα και λεπτομερής απεικόνιση των σπειρωμάτων που διαθέτει.

Για να μπορεί να χωρέσει απρόσκοπτα η κεφαλή του κοχλία M5 από τις αντίστοιχες οπές της φλάντζας του ρομπότ χρειάστηκε να κατεργαστούμε τοπικά την περιοχή γύρω από τις βίδες και να αφαιρέσουμε υλικό, όπως φαίνεται στην **Εικόνα**.

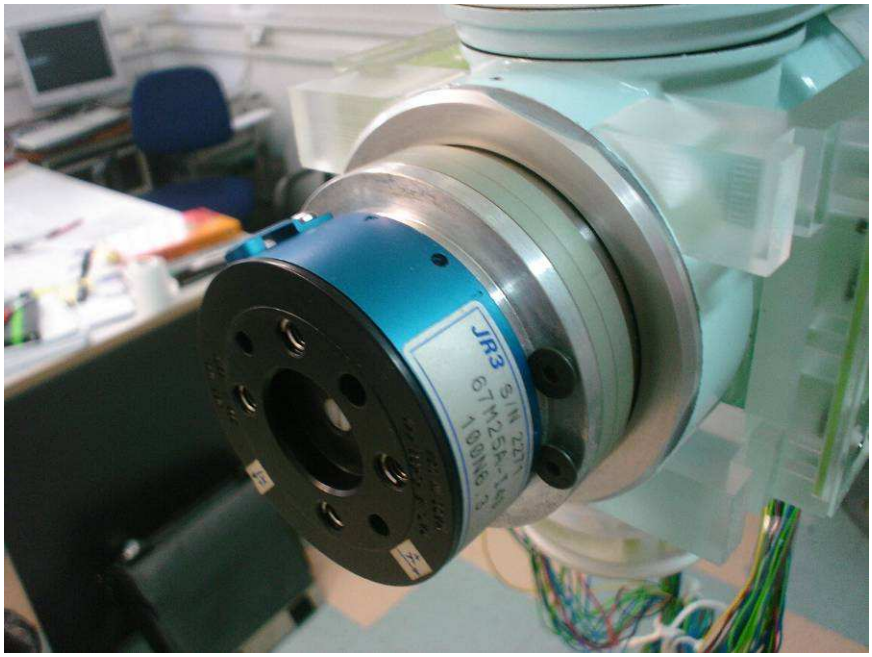


Εικόνα 7. Λεπτομέρεια κατεργασίας της πάνω πλευράς της φλάντζας σύνδεσης για την προσαρμογή των κοχλίων M5.

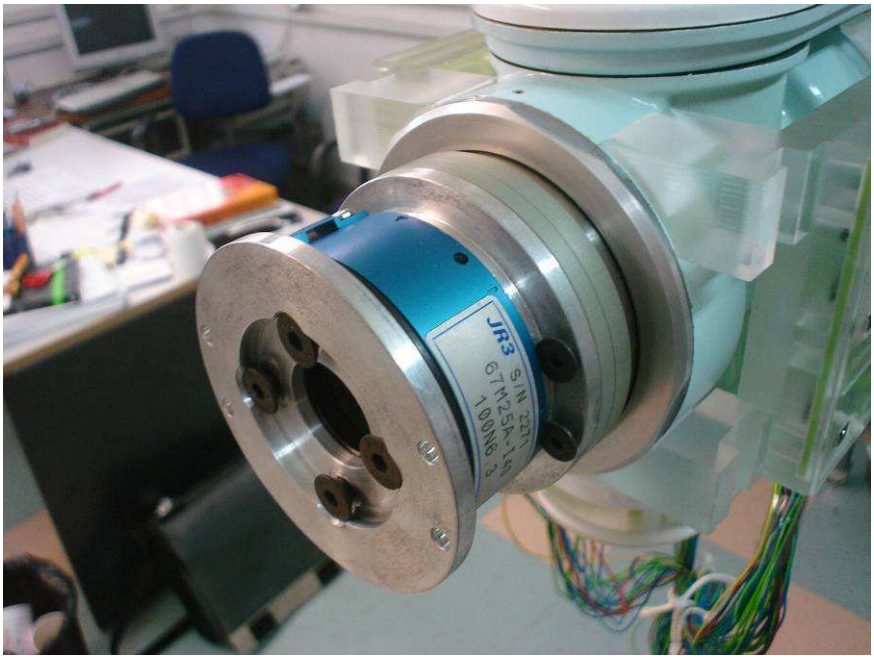
Αντίστοιχα στην ενδιάμεση φλάντζα μεγαλώσαμε λίγο τις δύο οπές από την πλευρά που πρέπει να περάσουνε οι βίδες M5, ώστε να μην υπάρχει δυσκολία κατά τη σύνδεση των εξαρτημάτων.

3.34 Ολοκλήρωση σύνδεσης πειραματικού εξοπλισμού

Ακολουθώντας τη διαδικασία σύνδεσης του αισθητήρα στον ακροδέκτη του ρομποτικού βραχίονα που περιγράφεται στο Εγχειρίδιο Γ.3 καταλήγουμε στην εξής συνδεσμολογία.



Εικόνα 8. Προοπτική απεικόνιση τελικής σύνδεσης αισθητήρα με τον ακροδέκτη του ρομποτικού βραχίονα.



Εικόνα 9. Προοπτική απεικόνιση τελικής σύνδεσης αισθητήρα με τον ακροδέκτη του ρομποτικού βραχίονα και επιπλέον σύνδεση της βοηθητικής φλάντζας.

3.4 : Βαθμονόμηση (calibration) του αισθητήρα JR3 δύναμης/ροπής

Εισαγωγή:

Κατά τη χρήση ενός μετρητικού οργάνου, όπως ο αισθητήρας που θα χρησιμοποιήσουμε, είναι απαραίτητο οι ενδείξεις που παίρνουμε να ανταποκρίνονται στα πραγματικά μεγέθη που μετράμε. Όσο κι αν είναι ελεγχόμενες οι συνθήκες αλληλεπίδρασης του οργάνου μέτρησης με το περιβάλλον, τίποτα δεν μπορεί να μας εξασφαλίσει μεγαλύτερη ακρίβεια και περισσότερη ασφάλεια κατά τη διάρκεια εκτέλεσης, τόσο ενός απλού, όσο και ενός σύνθετου πειράματος, από τη σωστή και όσο το δυνατόν ακριβέστερη βαθμονόμησή του.

Σε πρώτο επίπεδο είναι σημαντικό να εντοπιστεί και να επαληθευτεί η ακρίβεια του αισθητήρα σε συνδυασμό με τους ψηφιακούς περιορισμούς του αναλογοψηφιακού μετατροπέα που χρησιμοποιούμε για να λαμβάνουμε τις μετρούμενες τιμές σε έναν υπολογιστή, καθώς και το μη συστηματικό σφάλμα μέτρησης (θόρυβος) που παρουσιάζεται κατά τις μετρήσεις. Παράλληλα πρέπει να κατασκευαστεί μια ελεγχόμενη διάταξη για τη συλλογή μετρήσεων που θα χρησιμοποιηθούν για την εύρεση καμπύλων που θα προσεγγίζουν τη λειτουργία του αισθητήρα και στους έξι βαθμούς ελευθερίας του.

Στη συνέχεια, με γνώμονα την ακρίβεια που υπολογίστηκε και παίρνοντας τις κατάλληλες μετρήσεις σε όλους τους άξονες δυνάμεων και ροπών του αισθητήρα, συσχετίζοντας τις τιμές αυτές με τις θεωρητικές, μπορούμε να αναπτύξουμε μεθόδους εξάλειψης του συστηματικού σφάλματος σε πραγματικό χρόνο. Οι μέθοδοι αυτοί για να μπορούν να εφαρμοστούν πρέπει τα αποτελέσματά τους να πληρούν κάποια βασικά κριτήρια, μέσω των οποίων θα γίνει η αξιολόγησή τους και η επιλογή της καλύτερης εξ' αυτών.

Τέλος, είναι απαραίτητο να επαληθευθεί η αποτελεσματικότητα της μεθόδου βαθμονόμησης που επιλέξαμε, όχι μόνο σε θεωρητική βάση, αλλά παίρνοντας τυχαίες μετρήσεις χρησιμοποιώντας τη ρουτίνα βαθμονόμησης και συσχετίζοντάς τες και αυτές με τις αναμενόμενες πραγματικές τιμές.

3.41 Συλλογή τιμών δύναμης και ροπής μέσω του αισθητήρα με χρήση πρότυπων βαριδίων

Για την ορθή λειτουργία του αισθητήρα στις εφαρμογές που θα τον χρησιμοποιήσουμε είναι απαραίτητο να ακολουθηθεί μια σαφής διαδικασία βαθμονόμησης, με δικαιολόγηση όλων των βημάτων που ακολουθήθηκαν. Το πρώτο στάδιο είναι η σωστή και ακριβής συλλογή συγκεκριμένων τιμών με την κατάλληλη διάταξη και με ασφάλεια ως προς την ακρίβεια των μετρήσεων αυτών.

Για τη μελέτη της συμπεριφοράς του αισθητήρα σε φορτίσεις σε όλο το εύρος της μελλοντικής λειτουργίας του και στους έξι βαθμούς ελευθερίας του, καθώς και για την επιτυχή βαθμονόμησή του, ήταν απαραίτητο να μελετήσουμε τις ενδείξεις που παίρνουμε ασκώντας γνωστές δυνάμεις και ροπές σε όλους τους άξονες λειτουργίας του. Η περαιτέρω σύγκριση των ενδείξεων αυτών με τις αναμενόμενες και η μαθηματική επεξεργασία αυτών, θα καθιστούσε δυνατή την κατασκευή ενός μοντέλου, το οποίο κάθε φορά θα διόρθωνε την μετρούμενη ένδειξη προς την πραγματική.

Ο τρόπος που επιλέξαμε να εφαρμόσουμε εκ των προτέρων γνωστές φορτίσεις ήταν η κατάλληλη τοποθέτηση πρότυπων βαρών στον αισθητήρα παράλληλα κάθε φορά με τον προς μελέτη άξονα, όσον αφορά στη μέτρηση των δυνάμεων, και κάθετα σε δεδομένη απόσταση ως προς τον άξονα περιστροφής για τη μέτρηση των εκάστοτε ροπών.

Εικόνα 10. Τα πρότυπα βαρίδια που χρησιμοποιήθηκαν για τη βαθμονόμηση.

Τα πρότυπα βαρίδια που χρησιμοποιήσαμε ήταν των 50gr, 100gr, 200gr, 500gr, 1kg και 2kg με το συνδυασμό των οποίων μπορούσαμε να επιτύχουμε όλα τα ενδιάμεσα βάρη που χρειαζόμασταν. Όπως θα δούμε και παρακάτω, για την επαλήθευση της ορθής βαθμονόμησης χρησιμοποιήσαμε και τα βαρίδια κάτω των 50gr για να ελέγξουμε τον αισθητήρα και τη συμπεριφορά του μοντέλου βαθμονόμησης σε πιο "τυχαίες" τιμές των δυνάμεων, μακριά από τα αρχικώς μετρούμενα σημεία.

Οι μετρήσεις δεν θα έπρεπε να είναι ούτε πολύ λίγες, με ενδεχόμενη μια λανθασμένη καμπύλη παρεμβολής πάνω στα μετρούμενα σημεία, αλλά ούτε και πάρα πολλές, κάτι που θα δυσχέραινε χωρίς κάποιο ιδιαίτερο κέρδος την μαθηματική επεξεργασία τους, κυρίως την επεξεργασία που πρέπει να γίνεται σε πραγματικό χρόνο (real-time), κατά τη λειτουργία του αισθητήρα σε κανονικές συνθήκες.

Για την μέτρηση των δυνάμεων επιλέξαμε βήμα 50gr (περίπου 0,5N) και 100gr (περίπου 1N) σε περιοχές που οι ενδείξεις ήταν πιο ομαλές. Σε κάποιες περιπτώσεις πήραμε μετρήσεις με βήμα μικρότερο των 50gr για να αντιμετωπίσουμε περιοχές μετρήσεων που το σφάλμα του αισθητήρα παρουσίαζε μεγάλες μεταβολές σε μικρό εύρος λειτουργίας και καθιστούσε δύσκολο το σωστό υπολογισμό του στα ενδιάμεσα σημεία.

Για την μέτρηση των ροπών επιλέξαμε βήμα 100gr σε απόσταση 3.5cm από των άξονα εφαρμογής για τη μέτρηση της Mz (περίπου 0,34 $N \cdot m/10$) και σε απόσταση 4cm για τη μέτρηση των Mx και Mz (περίπου 0,4 $N \cdot m/10$). Οι αποστάσεις αυτές ήταν στα όρια της κατασκευαστικής μας διάταξης, για να μπορούμε να έχουμε όσο το δυνατόν μεγαλύτερο εύρος τιμών μπορούμε.

Συγκεντρωτικά παραθέτουμε το εύρος τιμών που τελικά συλλέξαμε για όλες τις δυνάμεις και ροπές (Fx, Fy, Fz & Mx, My, Mz). Σε αυτό το πεδίο τιμών πραγματοποιήθηκε και η βαθμονόμηση και σκοπός μας είναι εντός αυτού του πεδίου να λειτουργεί ο αισθητήρας κατά τη χρήση του σε μελλοντικές εφαρμογές, για να έχουμε όσο το δυνατόν ακριβέστερες τιμές.

Δυνάμεις

Fx (N)		**Fy (N)**		**Fz (N)**	
min	max	min	max	min	max
-16.7	18.6	-16.7	18.6	-21.6	19.6

Ροπές

Mx(N*m/10)		**My(N*m/10)**		**Mz(N*m/10)**	
min	max	min	max	min	max
-7.8	7.8	-7.8	7.8	-6.9	6.9

Πίνακας 4. Εύρος μετρούμενων τιμών δυνάμεων και ροπών

Τα πρότυπα βαρίδια, με τη δεδομένη μεγάλη ακρίβεια ως προς το βάρος τους, μας δίνουν τη δυνατότητα να έχουμε μεγάλη ακρίβεια και στην εκτίμηση των δυνάμεων και ροπών που ασκούμε μέσω αυτών στον αισθητήρα μας. Απαραίτητη προϋπόθεση είναι να

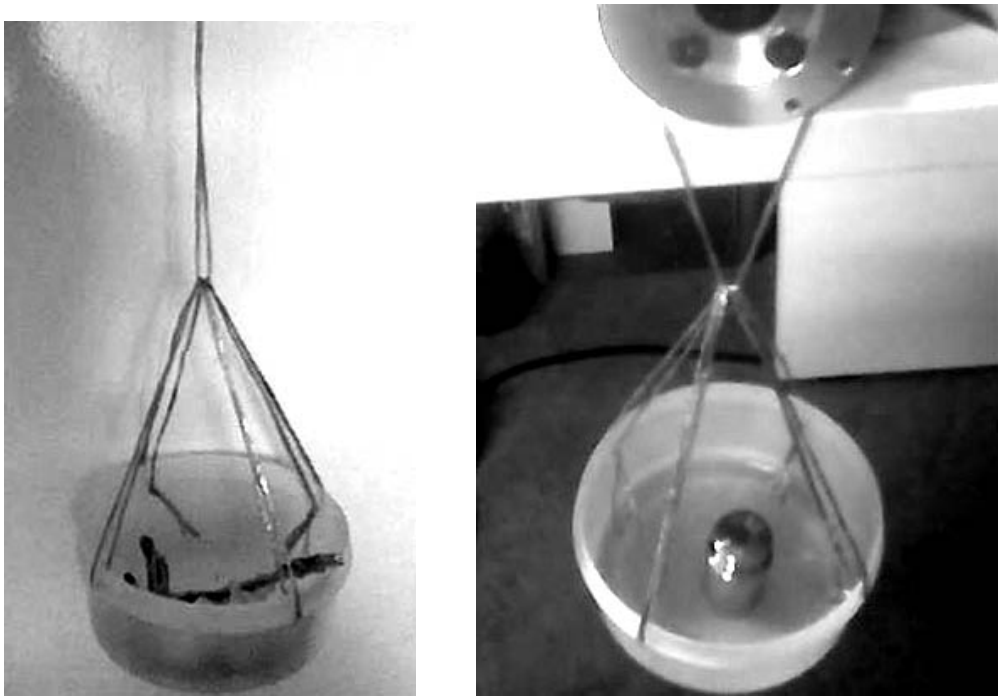
έχουμε ικανοποιητική ακρίβεια και στα άλλα μεγέθη που εμφανίζονται στις συναρτήσεις υπολογισμού δύναμης και ροπής. Συγκεκριμένα, τα μέτρα των μεγεθών που χρειαζόμαστε υπολογίζονται ως εξής:

Δύναμη $|F| = m \cdot g$ Ροπή $|M| = |F| \cdot r = m \cdot g \cdot r$

Με δεδομένη λοιπόν την ακρίβεια της μάζας m στον υπολογισμό των δυνάμεων και ροπών χρησιμοποιούμε την επιτάχυνση της βαρύτητας της περιοχής με δύο δεκαδικά ψηφία, όπου $g = 9.81 m/s^2$, και μετράμε το r με ακρίβεια της τάξεως χιλιοστού (mm) με χρήση παχυμέτρου. Μεγαλύτερη ακρίβεια δεν θα είχε ουσιαστικό ενδιαφέρον, καθώς θα ήταν κατά πολύ μεγαλύτερη από την ίδια την ακρίβεια μετρήσεως τιμών μέσω του αισθητήρα. Σκοπός μας ήταν δηλαδή να εξαλείψουμε αρχικώς οποιοδήποτε σφάλμα μέτρησης θα επηρεαζόταν από την ακρίβεια των μεγεθών που θα χρησιμοποιούσαμε.

Όμως, σημαντικότερος παράγοντας σφάλματος δεν αποδείχθηκε ο ανωτέρω προβληματισμός. Οι μετρήσεις έπρεπε να γίνονται για ένα μέγεθος και έναν άξονα εφαρμογής κάθε φορά και χωρίς τα βαρίδια να επηρεάζουν καθόλου τους άλλους άξονες, χωρίς δηλαδή το μέγεθος της εκτιμώμενης μέσω των τύπων δύναμης να αναλύεται σε περισσότερες συνιστώσες, με αποτέλεσμα να έχουμε σύνθετες ενδείξεις από τον αισθητήρα.

Σημαντικές αβλεψίες σε αυτό το ζήτημα θα μπορούσαν να προκαλέσουν πολλά προβλήματα στη βαθμονόμηση που έπεται. Επίσης, επιρροές δυνάμεων από το περιβάλλον, ταλαντώσεις των βαριδίων ή κακό μηδενισμό των τιμών (zero offset) των ενδείξεων έπρεπε να αποφευχθούν για τον ίδιο λόγο.



Εικόνα 11. Ιδιοκατασκευή για την τοποθέτηση των πρότυπων βαριδίων.

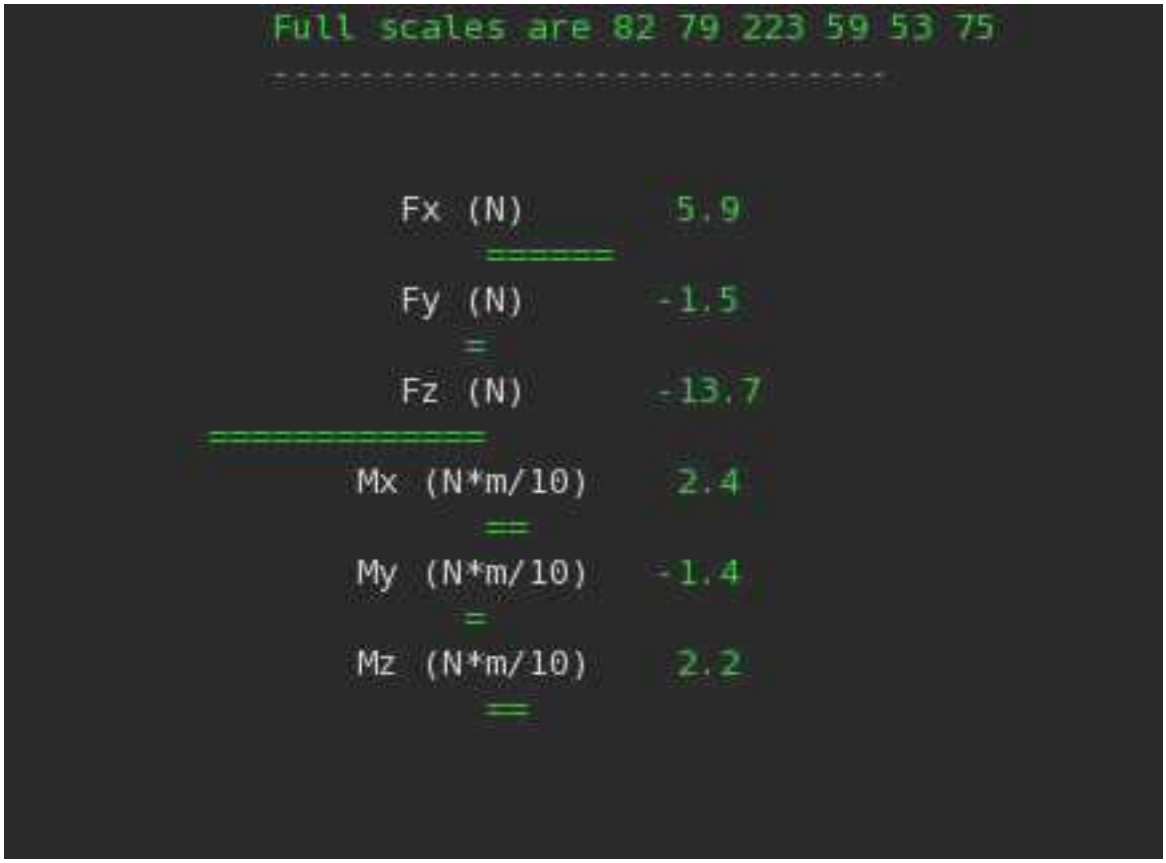
Για την εφαρμογή των βαριδίων επάνω στον αισθητήρα έγινε η σκέψη για την κατασκευή μιας πολύ απλής και ελαφριάς, αλλά ανθεκτικής κατασκευής σε μορφή "ζυγαριάς". Χρησιμοποιήθηκε ένα πλαστικό και ανάλαφρο δοχείο, μέσα στο οποίο τοποθετούνταν τα βαρίδια, το οποίο με χρήση σπάγκων από τέσσερα σημεία του προσδενόταν κατάλληλα στην κατασκευή μας.

Ο σπάγκος έχει συμπεριφορά εύκαμπτου φορέα που, όπως γνωρίζουμε από τη στατική μηχανική, μπορεί να δεχθεί μόνο αξονική εφελκυστική δύναμη και δεν μεταδίδει ροπές. Το αποτέλεσμα ήταν πως, οπουδήποτε κι αν εφαρμόζαμε την ιδιοκατασκευή με ενσωματωμένο κάποιο γνωστό βάρος μας, γνωρίζαμε ότι αυτή ασκούσε αποκλειστικά δύναμη με την κατεύθυνση της βαρύτητας.

Για την ολοκληροποίηση εν τέλει της συλλογής των μετρήσεων με όσο το δυνατόν σωστότερες και ακριβέστερες τιμές, εκτός από το κατασκευαστικό κομμάτι που προηγήθηκε, ήταν σημαντικό οι μετρήσεις να επαναληφθούν και να επαληθευθούν πολλές φορές και σε άλλες χρονικές περιόδους με πιθανότατα άλλες συνθήκες. Έτσι, παρά το χρονοβόρο της διαδικασίας αυτής καταλήξαμε σε πίνακες τιμών η ορθότητα των οποίων μας εγγυάται μια ορθή βαθμονόμηση.

Η απεικόνιση των τιμών στην οθόνη του υπολογιστή πραγματοποιήθηκε με τη βοήθεια προγράμματος σχεδιασμένου στη γλώσσα προγραμματισμού C (πάνω στην οποία θα σχεδιασθεί και το σύνολο των ρουτινών και εφαρμογών της παρούσας εργασίας). Με βάση τη βιβλιοθήκη συναρτήσεων ncurses χρησιμοποιήθηκαν οι υπορουτίνες drawline και drawtext για την απεικόνιση γραμμών ανάλογα με το μέγεθος της κάθε φόρτισης, με αλλαγή χρώματος πάνω από κάποιο όριο και των μονάδων των φορτίσεων αντίστοιχα. Στον κεντρικό κώδικα του προγράμματος (main) γίνεται ο σχεδιασμός του "παραθύρου" που θα εμφανίζονται οι τιμές με κλήση των δύο προαναφερθήσων υπορουτινών και στη συνέχεια γίνεται ο υπολογισμός και η παρουσίαση των τιμών στην οθόνη. Οι τιμές εισάγονται στο πρόγραμμά μας με τη βοήθεια του jr3pci-ioctl.h αρχείου κεφαλής (header file), που επικοινωνεί απευθείας με την κάρτα του αισθητήρα και παίρνει άμεσα τις τιμές. Για την αποφυγή αδικαιολόγητου τερματισμού του προγράμματος γίνεται έλεγχος της σωστής ενεργοποίησης και λειτουργίας του αισθητήρα, καθώς και της δυνατότητας προβολής χρωμάτων στον υπολογιστή που "τρέχει" η εφαρμογή.

Η παράθεση του κώδικα που αναλύθηκε ανωτέρω γίνεται στο Παράρτημα Α.1. Ακολουθεί μια εικόνα που παρουσιάζει το πρόγραμμα αυτό σε λειτουργία με τυχαίες φορτίσεις του αισθητήρα και ακρίβεια ενός δεκαδικού ψηφίου για κάθε φόρτιση.

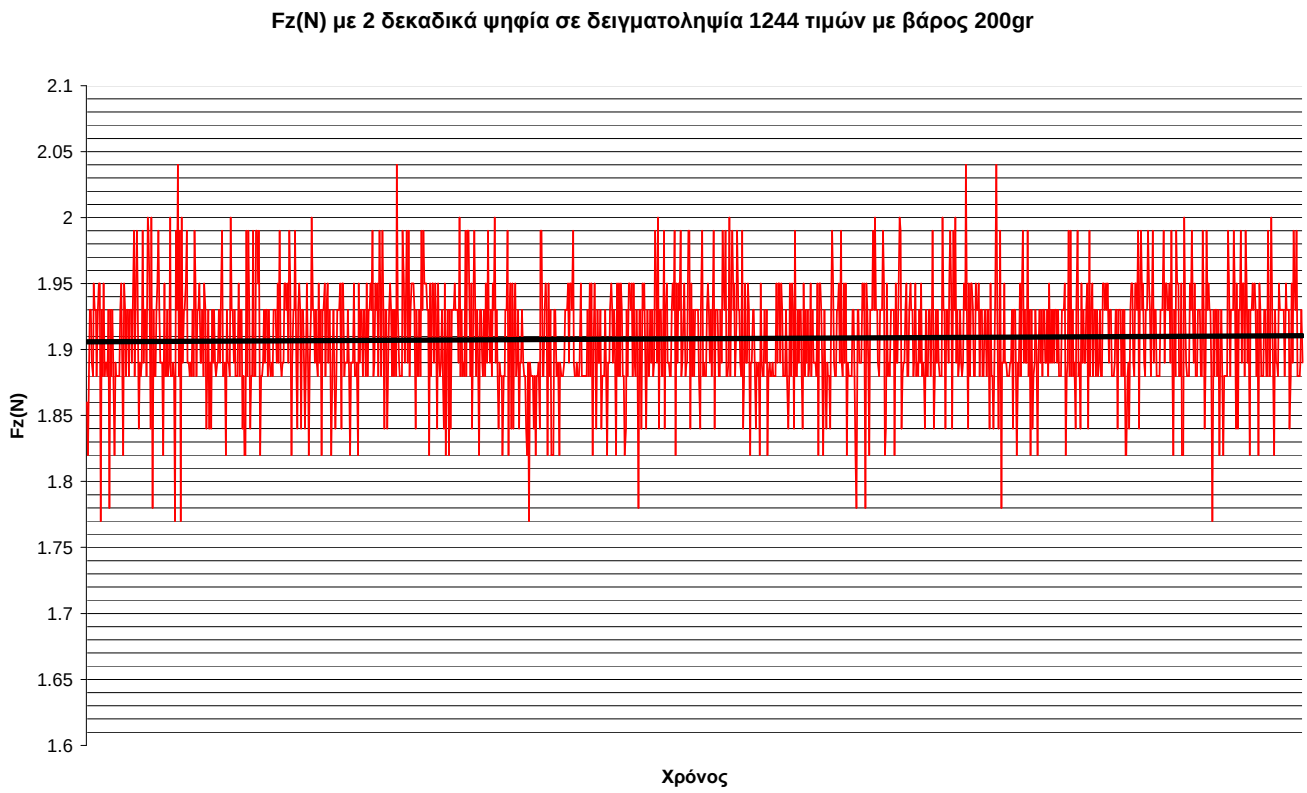


Εικόνα 12. Οθόνη ανάγνωσης ανεπεξέργαστων τιμών φορτίσεων.

3.42 Διερεύνηση θορύβου κατά τη συλλογή τιμών του αισθητήρα

Από τις πρώτες κιόλας δοκιμαστικές μετρήσεις με τον αισθητήρα μας είχαμε παρατηρήσει πως η ένδειξη που λαμβάναμε, ανεξάρτητα από την ορθότητά της σε σχέση με τη δύναμη ή ροπή που εφαρμόζαμε, ακόμα και σε περιπτώσεις σταθερής φόρτισης είχε κάποια μεταβλητότητα σε σχέση με το χρόνο. Όσο πιο μεγάλη ακρίβεια επιθυμούσαμε παρουσιάζοντας την ένδειξη με περισσότερα δεκαδικά ψηφία, τόσο πιο αισθητό ήταν το φαινόμενο.

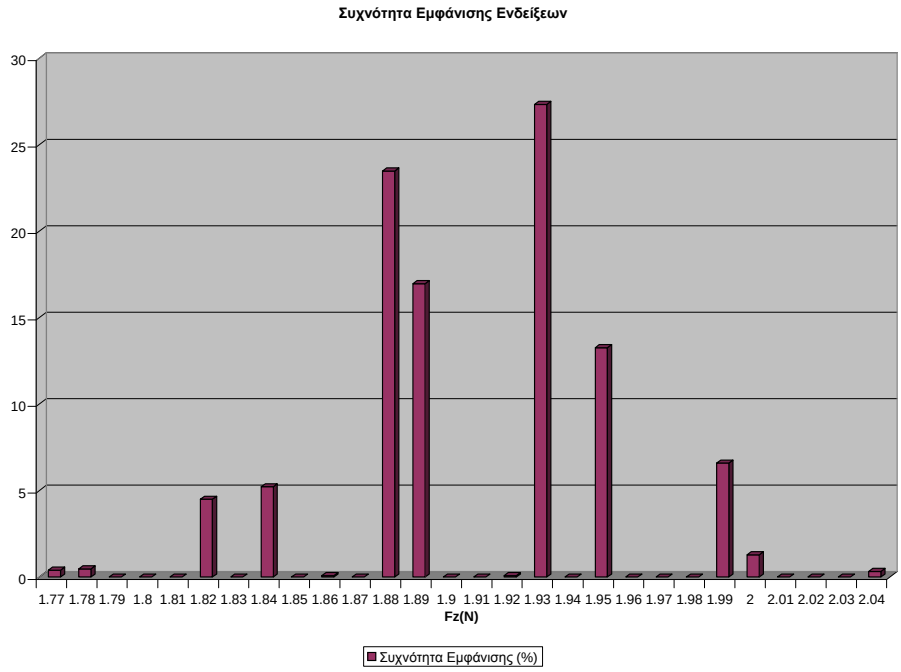
Στα επίπεδα ακριβείας, που είχαμε εκ των προτέρων εκτιμήσει ότι θα κινηθούμε σε αυτή την εργασία, το φαινόμενο αυτό δεν ήταν πολύ ανησυχητικό, αλλά ήταν άξιο μελέτης, ώστε να μπορούμε να έχουμε πλήρη αντίληψη για τις τιμές των δυνάμεων και των ροπών που παίρνουμε μέσω του αισθητήρα, που ενδεχομένως θα χρησιμοποιηθούν σε σημαντικές εφαρμογές στις οποίες δεν υπάρχουν περιθώρια σφαλμάτων.



Διάγραμμα 4. Διακύμανση σήματος αισθητήρα για σταθερή φόρτιση Fz με ακρίβεια δύο δεκαδικών ψηφίων.

Με μια πρώτη ματιά ήταν ορατό πως η ένδειξη κάθε φορά ταλαντωνόταν χωρίς κάποια συγκεκριμένη συχνότητα ή τρόπο γύρω από μια τιμή που θεωρήθηκε χαρακτηριστική των μετρήσεών μας. Προσμετρώντας έναν αριθμό δειγμάτων μέτρησης για συγκεκριμένη σταθερή φόρτιση, προέκυπτε με ευκολία και ακρίβεια μια μέση τιμή. Η τιμή αυτή συνέπιπτε σε όλες τις περιπτώσεις με την χαρακτηριστική τιμή της εκάστοτε μέτρησης και θεωρήθηκε ότι αυτή η τιμή προσδιορίζει το μετρούμενο μέγεθος και πρέπει να χρησιμοποιηθεί για τη βαθμονόμηση. Άλλωστε η τιμή αυτή ταυτίζεται με την τιμή που εμφανίζεται με τη μεγαλύτερη συχνότητα.

Δοκιμαστικά έγινε ανάλυση σε δείγμα ενδείξεων με δύο δεκαδικά ψηφία ακριβείας σε μέτρηση της δύναμης Fz. Μια γραφική του αναπαράσταση σαν αυτή που προηγήθηκε μας βοηθάει να το αντιληφθούμε σαν το σήμα του αισθητήρα για ένα συγκεκριμένο χρονικό διάστημα υπό σταθερή φόρτιση, όπου διακρίνεται και φανερά η χαρακτηριστική μέση τιμή του.

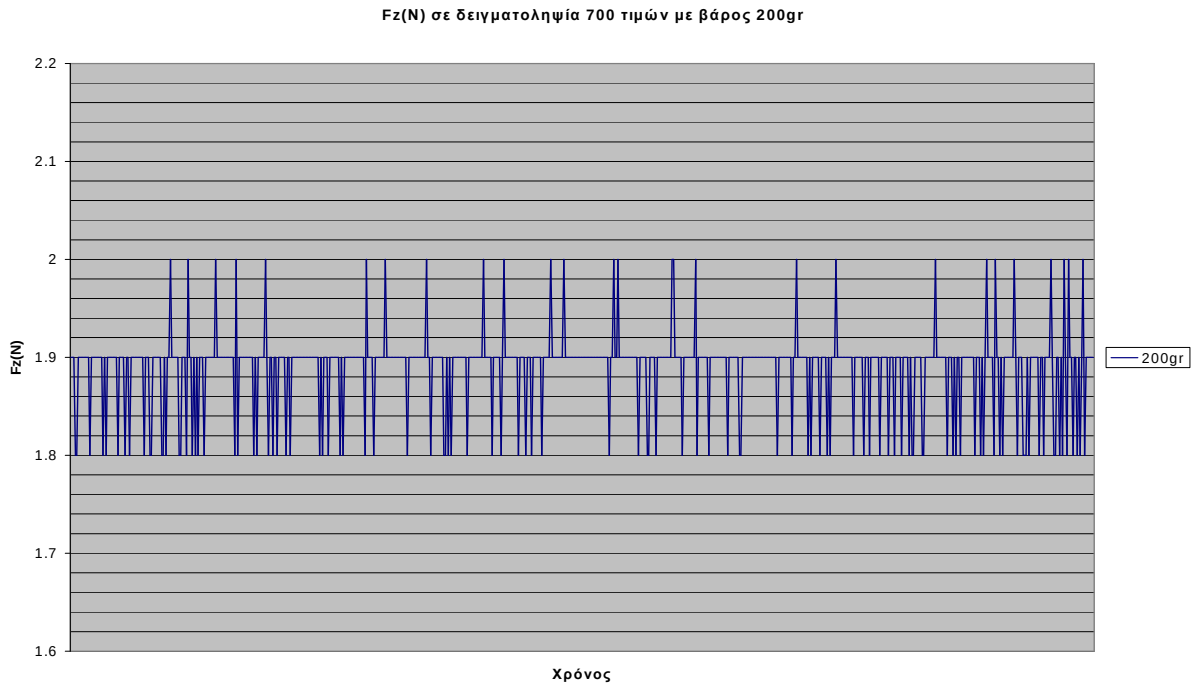


Διάγραμμα 5. Συχνότητα εμφάνισης (%) ενδείξεων αισθητήρα με σταθερή φόρτιση δύναμης Fz ~1.9N.

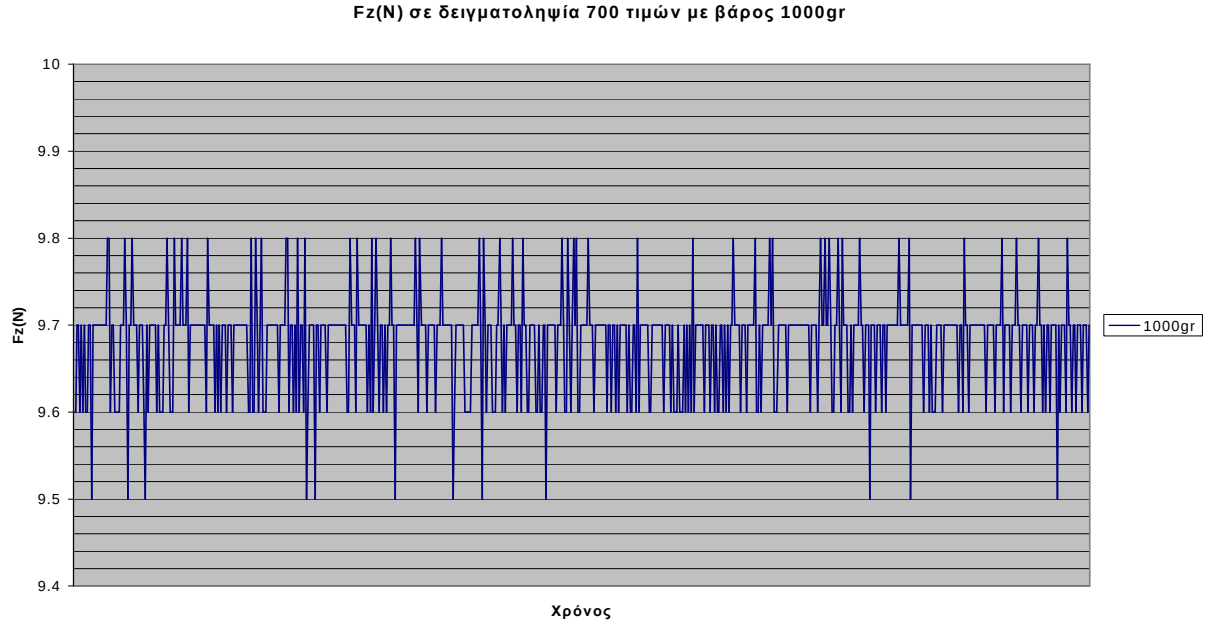
Είναι φανερό ότι η "ζώνη" του σήματος εκτείνεται σε ένα εύρος που θα καθιστούσε άσκοπη την προσπάθεια μέτρησης της δύναμης με ακρίβεια δύο δεκαδικών ψηφίων, όχι τόσο λόγω του συγκεκριμένου φαινομένου, αλλά κυρίως λόγω της διακριτικότητας του οργάνου, όπως αυτή υπολογίστηκε στο προηγούμενο κεφάλαιο. Επιλέχθηκε όμως αυτό το βάθος στις μετρήσεις, για να μελετήσουμε το φαινόμενο του θορύβου στο όριό του. Είναι φανερό πως σε αυτή την ακρίβεια των δύο δεκαδικών ψηφίων είναι κάποιες τιμές που ο αισθητήρας δεν μπορεί καν να πάρει, το οποίο γίνεται καλύτερα αντιληπτό έχοντας κατασκευάσει ένα ραβδόγραμμα συχνότητας εμφάνισης των διαφορετικών τιμών. Παρατηρούμε επίσης ότι οι ενδείξεις ακολουθούν περίπου κανονική κατανομή γύρω από τη μέση τιμή που υπολογίστηκε.

Προχωρώντας σε δειγματοληψία μετρήσεων σταθερής φόρτισης με ακρίβεια ενός δεκαδικού ψηφίου, κρίθηκε απαραίτητο η μελέτη του φαινομένου του θορύβου να γίνει σε ένα εύρος φορτίσεων διαφορετικής τάξης μεγέθους. Με αυτόν τον τρόπο είχαμε τη δυνατότητα να διαπιστώσουμε κατά πόσο αλλάζει ή και επιδεινώνεται το φαινόμενο αυτό σε μεγαλύτερες φορτίσεις και κατά πόσο πρέπει να συνυπολογίσουμε αυτόν τον παράγοντα.

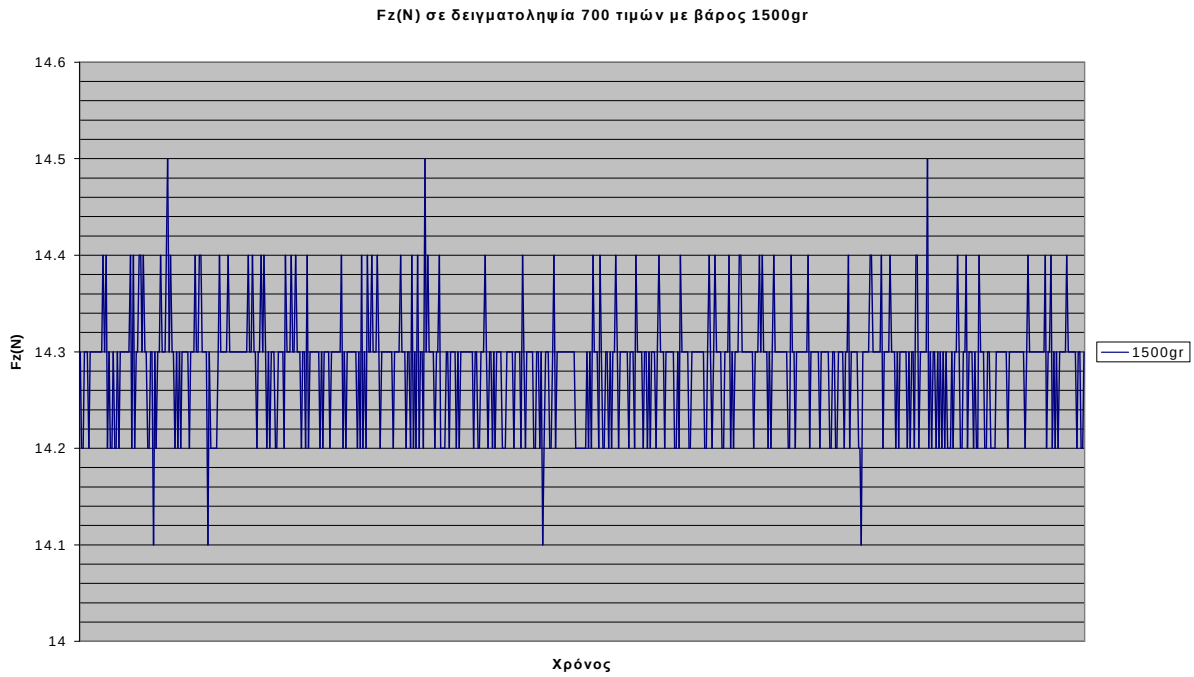
Για να διαπιστώσουμε τα ανωτέρω πήραμε δείγμα επτακοσίων τιμών από εφαρμογή σταθερής δύναμης μέσω βάρους 200gr, 1000gr και 1500gr με ακρίβεια τιμών δύναμης της τάξεως του 0,1N.



Διάγραμμα 6. Διακύμανση σήματος δύναμης Fz με ακρίβεια ενός δεκαδικού ψηφίου μέσω εφαρμογής βάρους 200gr.



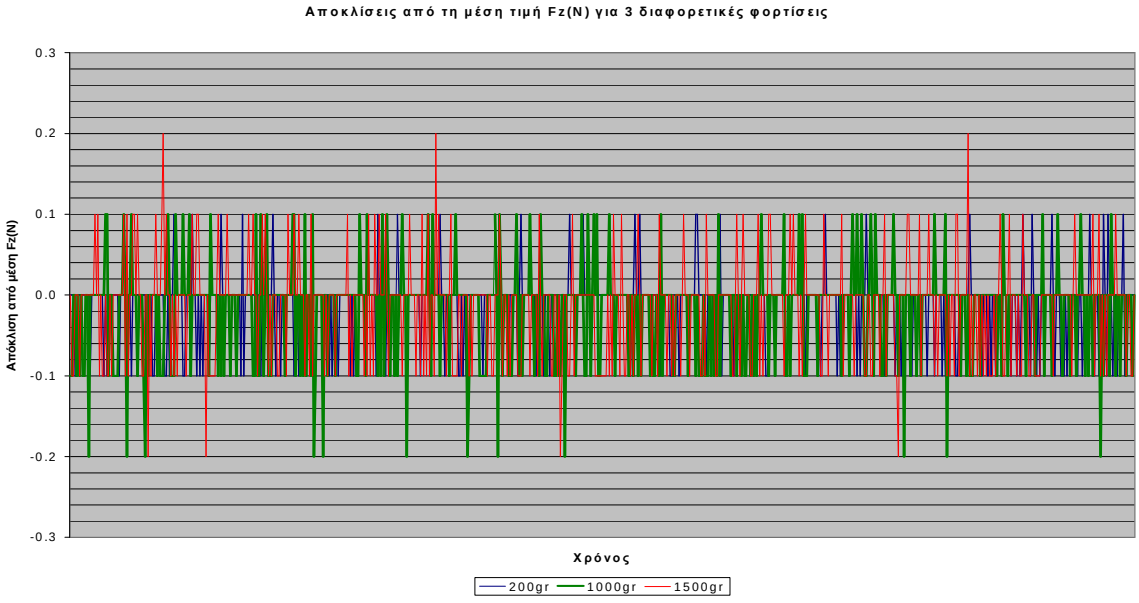
Διάγραμμα 7. Διακύμανση σήματος δύναμης Fz με ακρίβεια ενός δεκαδικού ψηφίου μέσω εφαρμογής βάρους 1000gr.



Διάγραμμα 8. Διακύμανση σήματος δύναμης Fz με ακρίβεια ενός δεκαδικού ψηφίου μέσω εφαρμογής βάρους 1500gr.

Παρατηρούμε πως ο θόρυβος που παραμένει τελικά στη μέτρηση των τιμών του αισθητήρα, έπειτα και από όλα τα φίλτρα που τον συνοδεύουν (στη συγκεκριμένη περίπτωση της εφαρμογής δύναμης κατά τον άξονα Z), είναι πρακτικά ανεξάρτητος του μεγέθους της δύναμης που εφαρμόζουμε αναφορικά με το εύρος που προβλέπεται να λειτουργήσει ο αισθητήρας μας (στο πεδίο (-20 , 20) N).

Αυτό γίνεται σαφέστερο με την υπέρθεση των τριών διαγραμμάτων σε ένα κοινό διάγραμμα με κοινό άξονα την απόκλιση των μετρούμενων τιμών από τη μέση τιμή.



Διάγραμμα 9. Υπέρθεση αποκλίσεων σταθερών διαφορετικών φορτίσεων Fz.

Μελετώντας το παραπάνω Διάγραμμα 9 παρατηρούμε πως και στις τρεις διαφορετικές *σταθερές* (η κάθε μία σε συνάρτηση με το χρόνο) φορτίσεις εμφανίζονται με μικρή περιοδικότητα αποκλίσεις της τάξεως του |0,1| N. Σε φορτίσεις πιο κοντά στο μέγιστο της προβλεπόμενης φόρτισης έχουμε σποραδικές εμφανίσεις αποκλίσεων της τάξεως των |0,2| N, η συχνότητα των οποίων είναι τόσο μικρή που μας επιτρέπει ως ένα βαθμό να τις αμελήσουμε.

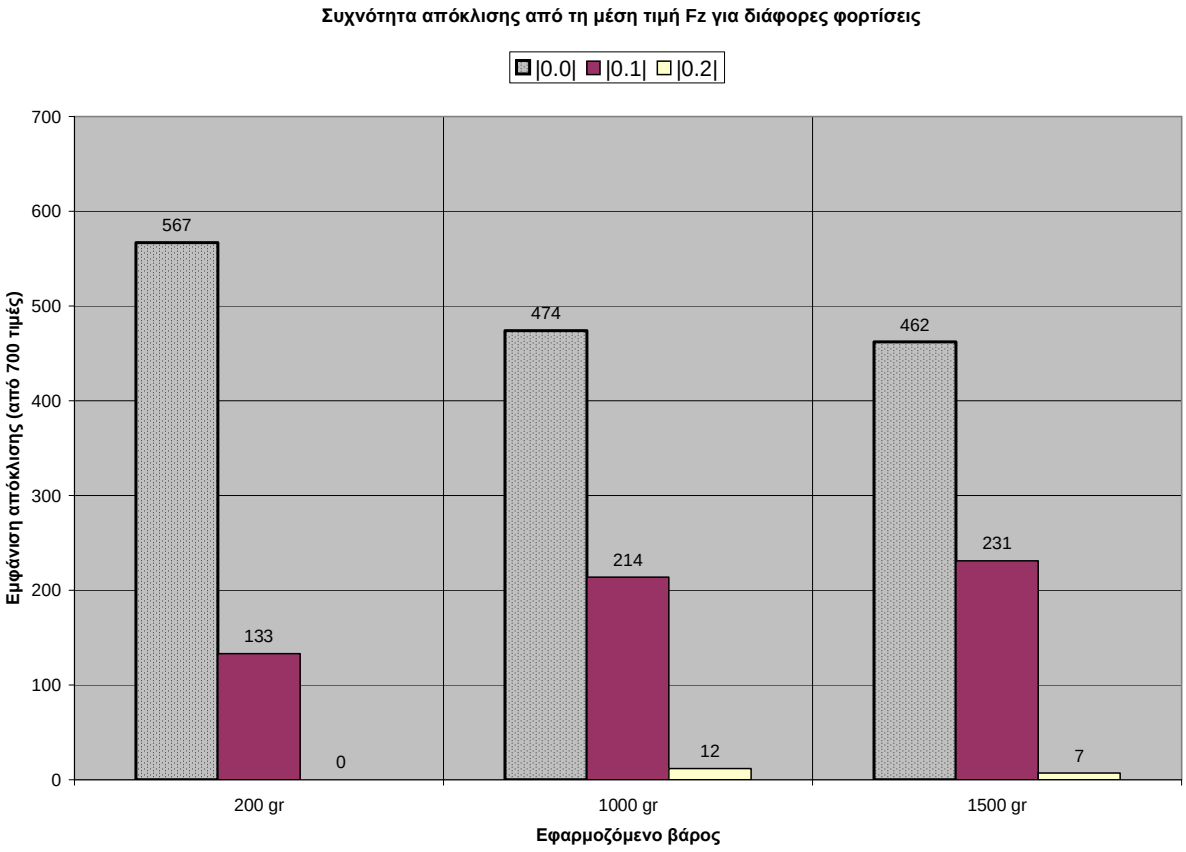
Ομαδοποιώντας τις παραπάνω τιμές σε πίνακες μπορούμε να μελετήσουμε καλύτερα το φαινόμενο:

Εφαρμογή Βάρους:	*200 gr*
Μέσος όρος:	**1.9 N**
Απόκλιση (ABS)	Εμφάνιση
|0.0|	567
|0.1|	133
|0.2|	0
Σύνολο δείγματος	700

Εφαρμογή Βάρους:	*1000 gr*
Μέσος όρος:	**9.7 N**
Απόκλιση (ABS)	Εμφάνιση
|0.0|	474
|0.1|	214
|0.2|	12
Σύνολο δείγματος	700

Εφαρμογή Βάρους:	*1500 gr*
Μέσος όρος:	**14.3 N**
Απόκλιση (ABS)	Εμφάνιση
|0.0|	462
|0.1|	231
|0.2|	7
Σύνολο δείγματος	700

Πίνακας 5. Εμφάνιση αποκλίσεων από τη μέση τιμή δύναμης Fz για δείγμα 700 μετρήσεων με την ίδια φόρτιση



Διάγραμμα 10. Ραβδόγραμμα αποκλίσεων από τη μέση τιμή για διαφορετικές μεγέθους φορτίσεις.

Έτσι μπορούμε να μετασχηματίσουμε το σύνολο των μετρήσεων σε ένα πιο ευανάγνωστο ραβδόγραμμα, όπου φαίνεται καθαρά πως τα επίπεδα θορύβου, και κατά συνέπεια το όριο της ακρίβειάς μας στις μετρήσεις, είναι κατά βάση $\pm 0{,}1N$.

Σε πανομοιότυπα επίπεδα κινείται, όπως προαναφέραμε, και ο θόρυβος σε φορτίσεις κατά τους άλλους άξονες (Fy, Fz). Όσον αφορά στις ροπές, το όριο της ακρίβειας εκτιμήθηκε αντίστοιχα σε $\pm 0{,}1N \cdot m/10$.

3.43 Μετρήσεις τιμών δυνάμεων και ροπών και απόκλιση από τις πραγματικές

Με τη μεθοδολογία και τις παραμέτρους που αναλύθηκαν ανωτέρω έγιναν μετρήσεις στους έξι βαθμούς ελευθερίας του αισθητήρα και στους τρεις άξονες, δηλαδή τόσο για θετικές όσο και για αρνητικές δυνάμεις και ροπές. Οι μετρήσεις συγκρίθηκαν άμεσα με τις πραγματικές τιμές που θα έπρεπε να εμφανιστούν εάν ο αισθητήρας ήταν σωστά βαθμονομημένος. Υπολογίστηκε το απόλυτο και το σχετικό σφάλμα και συγκεντρώθηκαν αναλυτικά όλα τα δεδομένα σε Πίνακες που τους παραθέτουμε πλήρως στο Παράρτημα Α.2 για μελλοντική μελέτη και χρήση. Οι τιμές σφάλματος που ξεπερνούν το μέγιστο ανεκτό σφάλμα ($\pm 0{,}1N$ για τις δυνάμεις και $\pm 0{,}1N \cdot m/10$ για τις ροπές) παρουσιάζονται με έντονους χαρακτήρες για ευκολότερη αντίληψη του σφάλματος που είναι πάνω από αυτά τα όρια.

Το απόλυτο σφάλμα στον υπολογισμό των δυνάμεων κυμαινόταν από μηδενική τιμή (0,0N) μέχρι και 0,9N στην χειρότερη περίπτωση, όμως στις περισσότερες περιπτώσεις που υπήρχε σφάλμα αυτό δεν ξεπερνούσε τα 0,4N. Σφάλμα της τάξεως των $\pm 0{,}1N$, αν και προσμετρήθηκε στη χάραξη των καμπύλων παρεμβολής και προσέγγισης, δεν θεωρείται συστηματικό σφάλμα, καθώς όπως αναλύσαμε στα προηγούμενα υποκεφάλαια, ταυτίζεται με το εύρος του "θορύβου" του αισθητήρα.

Αντίστοιχα, το απόλυτο σφάλμα στον υπολογισμό των ροπών κυμαινόταν από μηδενική τιμή ($0{,}0\,N \cdot m/10$) μέχρι και $1{,}3\,N \cdot m/10$ στην χειρότερη περίπτωση, όμως στις περισσότερες περιπτώσεις που υπήρχε σφάλμα αυτό δεν ξεπερνούσε τα $0{,}5\,N \cdot m/10$. Όπως και στη περίπτωση των δυνάμεων σφάλμα της τάξεως των $\pm 0{,}1N \cdot m/10$ αν και προσμετρήθηκε στη χάραξη των καμπύλων παρεμβολής και προσέγγισης, δεν θεωρείται συστηματικό σφάλμα, καθώς και αυτό ταυτίζεται με το εύρος του "θορύβου" του αισθητήρα.

Για να κατανοήσουμε καλύτερα τη συμπεριφορά του αισθητήρα, τοποθετήσαμε τις τιμές και τα σφάλματα σε διαγράμματα, έτσι ώστε να μπορούμε ευκολότερα να δοκιμάσουμε και να επιλέξουμε μεθόδους βαθμονόμησης.

Βασικός μας σκοπός ήταν να εξαλείψουμε το σφάλμα είτε υπολογίζοντας και προσθέτοντας σε κάθε μας μέτρηση ένα συντελεστή διόρθωσης δF (ή δM) είτε μετασχηματίζοντας άμεσα την ένδειξη που θα μας δίνει ο αισθητήρας στην αντίστοιχη σωστή

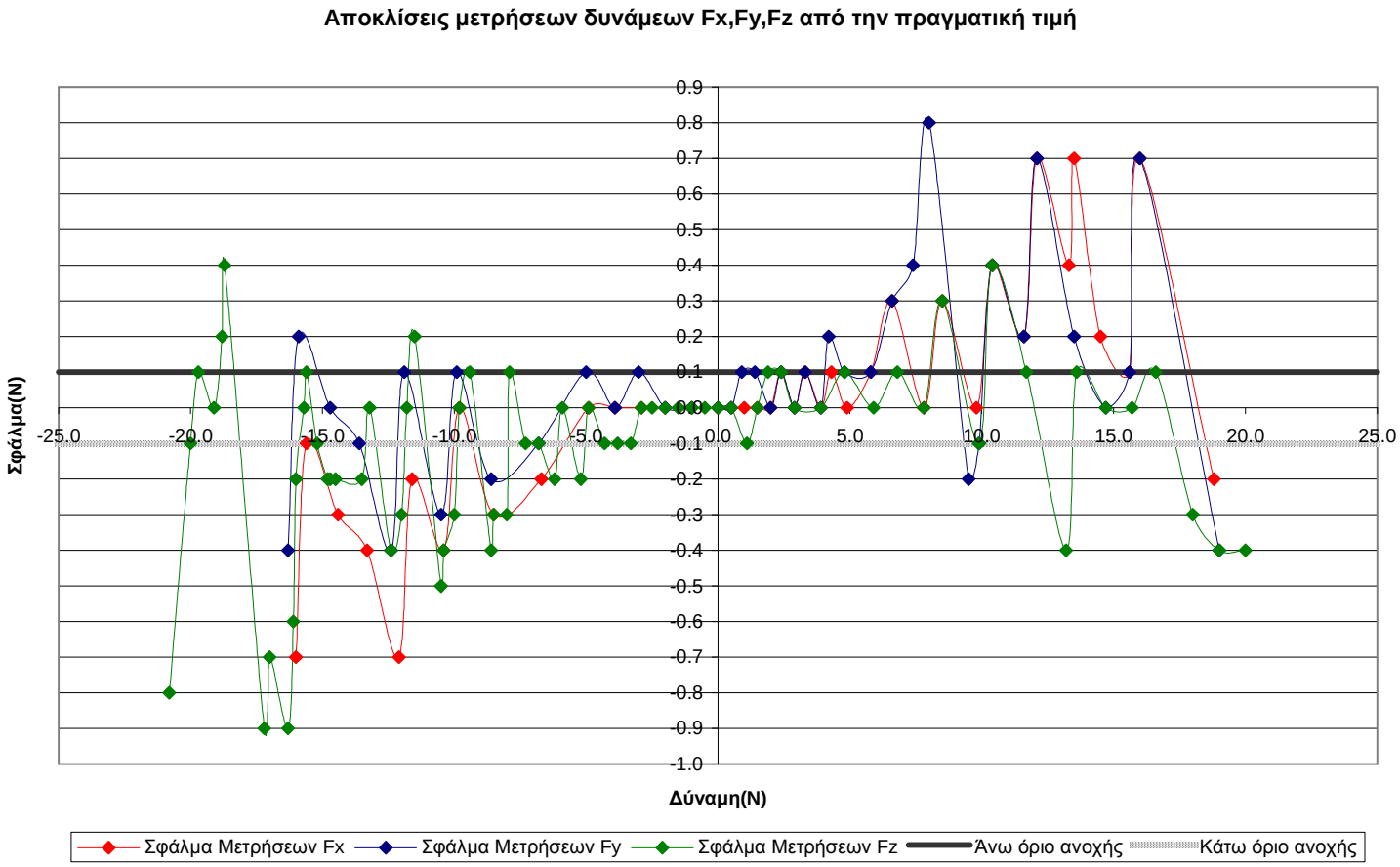
τιμή, μέσω κάποιας ή κάποιων συναρτήσεων. Γι' αυτό και στα διαγράμματα έχουμε στον άξονα x το μοναδικό μετρούμενο μέγεθος, τη δύναμη ή τη ροπή της ένδειξης, και στον άξονα y το σφάλμα σε κάθε περίπτωση εκφρασμένο ως το συντελεστή διόρθωσης δF (ή δM) που πρέπει να προσθέσουμε ώστε:

$$F_{\text{πραγματική}} = F_{\text{ένδειξης}} + \delta F$$

$$M_{\text{πραγματική}} = M_{\text{ένδειξης}} + \delta M$$

Έτσι, εκτός από τα μεγέθη που μας χρειάζονται, τα διαγράμματα αυτά προσφέρουν καλύτερη εποπτεία της παρέκκλισης των ενδείξεων από τις πραγματικές τιμές και η ταυτόχρονη τοποθέτηση των δυνάμεων όλων των αξόνων σε κοινό διάγραμμα (αντίστοιχα και τον ροπών) προσφέρουν αμεσότητα στη σύγκριση όμοιων μεγεθών.

Στα διαγράμματα σχεδιάζονται τα άνω και κάτω όρια ανοχής στο σφάλμα των μετρήσεών μας, καθορίζοντας έτσι τη ζώνη πάνω στην οποία το εμφανιζόμενο σφάλμα πιθανότατα έχει προκύψει από το θόρυβο, όπως αυτός αναλύθηκε στο προηγούμενο κεφάλαιο.

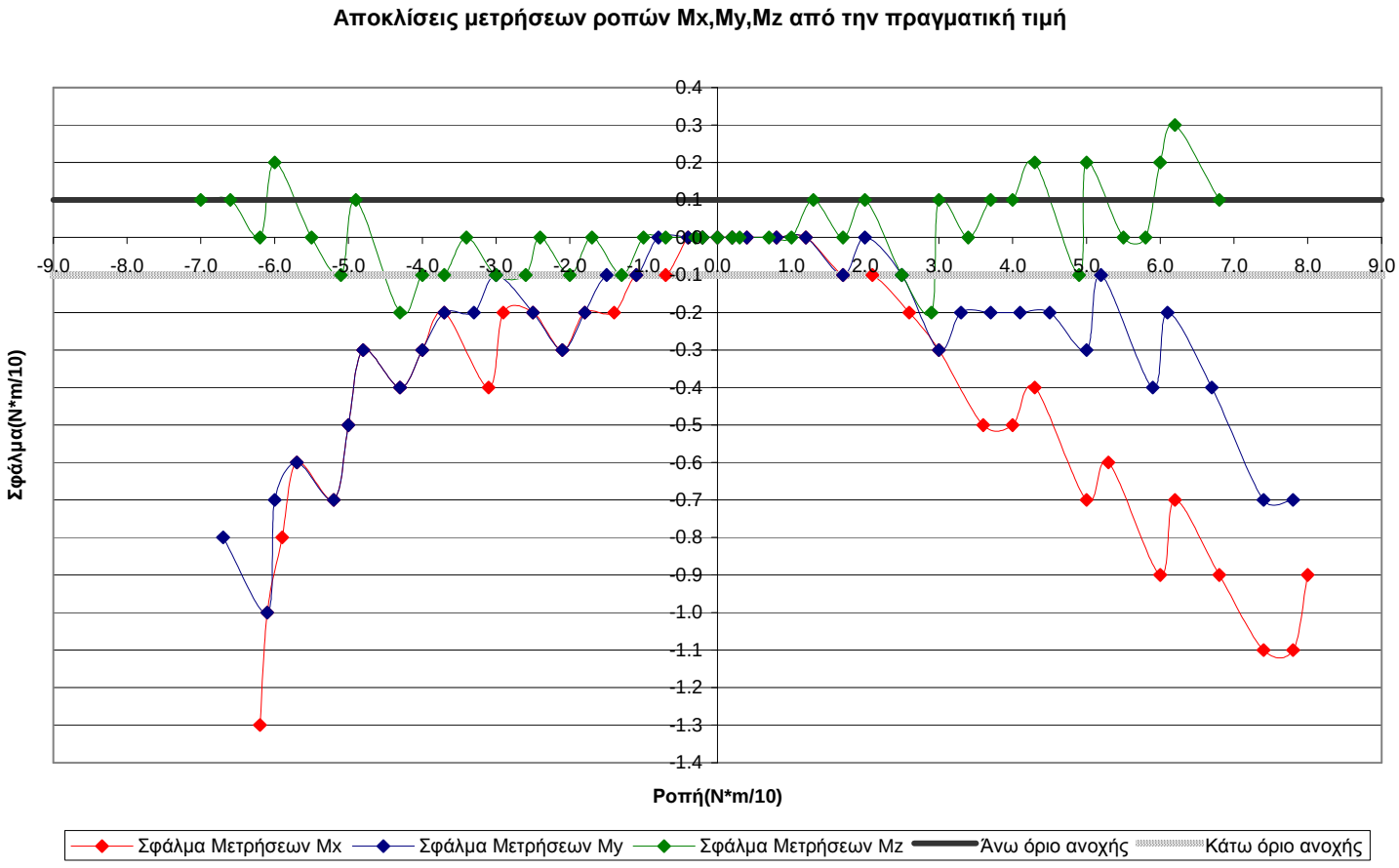


Διάγραμμα 11. Αποκλίσεις μετρήσεων δυνάμεων από την πραγματική τιμή.

Παρατηρούμε πως σημαντικά και υπολογίσιμα σφάλματα στα διαγράμματα των δυνάμεων παρουσιάζονται σε μετρήσεις μεγαλύτερες των 5.0 N σε απόλυτη τιμή που κατά κανόνα αυξάνονται όσο μεγαλώνει η δύναμη που θέλουμε να μετρήσουμε. Σε γενικές γραμμές έχουμε παρόμοια συμπεριφορά εξέλιξης του σφάλματος και στους τρεις άξονες μέτρησης, κυρίως μεταξύ των δυνάμεων στους άξονες x και y που η λειτουργία του αισθητήρα για τη μέτρησή τους είναι αντίστοιχη (βλέπε κεφάλαιο *3.41: Συλλογή τιμών δύναμης και ροπής μέσω του αισθητήρα με χρήση πρότυπων βαριδίων).*

Αξιοσημείωτο είναι πως σχεδόν σε όλες τις περιπτώσεις η ένδειξη του αισθητήρα είναι σε απόλυτη τιμή μικρότερη της πραγματικής. Έτσι στα θετικά των δυνάμεων ο συντελεστής διόρθωσης δF που πρέπει να προστεθεί για την εύρεση της πραγματικής τιμής είναι κατά κανόνα θετικό μέγεθος, ενώ στα αρνητικά των δυνάμεων είναι αρνητικό μέγεθος.

Η διαφαινόμενη ταλάντωση του σφάλματος των δυνάμεων γύρω από τη μηδενική τιμή σε κάποιες περιοχές μετρήσεων προοικονομεί δυσκολίες κατά την βελτιστοποίηση της βαθμονόμησης, για αυτό έγινε προσπάθεια πυκνότερης συλλογής τιμών σε αυτές.



Διάγραμμα 12. Αποκλίσεις μετρήσεων ροπών από την πραγματική τιμή.

Αναλύοντας συνοπτικά και το διάγραμμα που αφορά τις ροπές σημειώνουμε αντίστοιχα ορθή συμπεριφορά του αισθητήρα και χωρίς βαθμονόμηση σε τιμές κάτω των $0.2\,N \cdot m/10$ σε απόλυτη τιμή. Παρομοίως με τις δυνάμεις έχουμε μεταξύ των ροπών στους άξονες x και y αντίστοιχη συμπεριφορά του σφάλματος για ένα μεγάλο εύρος τιμών, καθώς η λειτουργία του αισθητήρα για τη μέτρησή τους είναι αντίστοιχη (βλέπε κεφάλαιο *3.41: Συλλογή τιμών δύναμης και ροπής μέσω του αισθητήρα με χρήση πρότυπων βαριδίων).*

Επίσης σε όλο το εύρος των ροπών των αξόνων x και y οι μετρήσεις δείχνουν πως ο συντελεστής διόρθωσης δM είναι πάντα αρνητικός, δηλαδή σε θετικές τιμές ροπών Mx και My οι ενδείξεις μας είναι πάντα μεγαλύτερες των πραγματικών και σε αρνητικές τιμές αυτών οι ενδείξεις μας είναι πάντα μικρότερες των πραγματικών.

Άμεσα παρατηρήσιμο είναι πως συγκριτικά με όλα τα υπόλοιπα μετρούμενα μεγέθη η ροπή κατά των άξονα z, η Mz, παρουσιάζει ομαλότητα σε όλο το εύρος της με σχετικά πολύ μικρό σφάλμα.

3.44 Μεθοδολογίες βαθμονόμησης και κριτήρια αξιολόγησης

Πριν ακόμα από την πλήρη συγκέντρωση δοκιμαστικών τιμών με τη χρήση των πρότυπων βαριδίων, διενεργήσαμε μαθηματικούς υπολογισμούς στα υπάρχοντα δεδομένα την κάθε στιγμή, για να δοκιμάσουμε μεθόδους βαθμονόμησης.

Ξεκινώντας από μαθηματικά απλούστερες μεθόδους και καταλήγοντας σε πιο σύνθετες, αλλά και πιο ακριβείς, καθορίσαμε εν τέλει τα βασικά κριτήρια που μας οδήγησαν στην επιλογή μιας μεθόδου.

Καταρχάς είναι απαραίτητο να σημειώσουμε ότι, σε εφαρμογές που απαιτούνται υπολογισμοί σε πραγματικό χρόνο, οποιαδήποτε μαθηματική πράξη και ρουτίνα μεταφράζεται σε καθυστέρηση, η οποία μπορεί να προκαλέσει δυσλειτουργίες στη σωστή συμπεριφορά ενός ρομποτικού συστήματος. Όσον αφορά στο υπολογιστικό κομμάτι της βαθμονόμησης, σε όλες τις μεθόδους που δοκιμάσαμε, φροντίσαμε ο μεγαλύτερος όγκος των υπολογισμών να γίνει εκτός χρόνου λειτουργίας(off-line), ώστε αυτή η παράμετρος να μη μας επηρεάζει άμεσα.

Έτσι, σκοπός μας ήταν να δημιουργήσουμε μητρώα και συναρτήσεις με βάση το σύνολο των τιμών που συλλέξαμε από τον αισθητήρα για κάθε φόρτιση, να τα εισάγουμε κατάλληλα στο πρόγραμμα με το οποίο χειριζόμαστε τη λειτουργία του αισθητήρα και να παράγουμε τις διορθωμένες τιμές με σκοπό αυτές να προσεγγίζουν τις πραγματικές, με τη μεγαλύτερη δυνατή ακρίβεια.

Μελετήσαμε αρχικά γραφήματα που με βάση τις συναρτήσεις και τα μητρώα βαθμονόμησης παριστούσαν τις διορθωτικές αλλαγές που έπρεπε να γίνουν στην ένδειξή

μας για κάθε πιθανή τιμή δύναμης ή ροπής που θα μπορούσαμε να πάρουμε από τον αισθητήρα. Έτσι, μπορούσε να επιβεβαιωθεί η ορθότητα των μαθηματικών υπολογισμών, αλλά και να γίνει ορατή κάποια μη αναμενόμενη συμπεριφορά του μοντέλου μακριά από τα γνωστά σημεία που εκφράζουν τις τιμές που είχαμε ήδη συλλέξει, ώστε να διορθωθεί αντιστοίχως.

Τα *βασικά κριτήρια* για την αξιολόγηση και την εύρεση της πιο ενδεδειγμένης μεθόδου καθορίστηκαν τα εξής:

- *Ακρίβεια αποτελεσμάτων*: Όπως αναλύθηκε στα προηγούμενα κεφάλαια και υπολογίστηκε η ακρίβεια με την οποία παίρνουμε τιμές από τον αισθητήρα, έτσι και έπειτα στη βαθμονόμηση έπρεπε να διατηρήσουμε τα ίδια επίπεδα ακριβείας. Για αυτό οι ενδείξεις των βαθμονομημένων δυνάμεων δεν έπρεπε να απέχουν περισσότερο από $\pm 0{,}1N$ από τις πραγματικές και αντίστοιχα οι ενδείξεις των ροπών περισσότερο από $\pm 0{,}1N \cdot m/10$ από τις πραγματικές.
- *Συνέχεια των τιμών της φόρτισης:* Αυτονόητο ήταν πως για κάθε δυνατή τιμή φόρτισης μέσα στο πεδίο των εφαρμογών μας έπρεπε να υπάρχει μια αντίστοιχη τιμή που θα έχει προκύψει από τα μαθηματικά μας μοντέλα και θα προσεγγίζει κατά πολύ την πραγματική. Σε μεθόδους, όμως, που γίνεται χρήση πολλαπλών συναρτήσεων σε υποσύνολα τιμών του πεδίου ορισμού και υπάρχει ασυνέχεια στα σημεία σύνδεσής τους, συνεπάγεται γύρω από μία τιμή να υπάρχουν σημαντικές αποκλίσεις και διακυμάνσεις σε σχέση με την πραγματική μεταβολή του φαινομένου και έτσι να υπάρχει ταλάντωση των ενδείξεών μας γύρω από αυτή την τιμή με τις συνακόλουθες δυσλειτουργίες που αυτό μπορεί να προκαλέσει σε εφαρμογές ελέγχου σε μια ρομποτική διάταξη.
- *Συνέχεια της 1ης και της 2ης παραγώγου των τιμών της φόρτισης:* Ένα κριτήριο που είναι απολύτως επιθυμητό να πληρείται, ώστε να έχουμε όσο το δυνατόν ομαλότερο προφίλ των βαθμονομημένων τιμών μας για περισσότερη ασφάλεια κατά τη χρήση αυτών των τιμών στις ρουτίνες ελέγχου σε μια ρομποτική εφαρμογή.
- *Προσαρμοστικότητα της μεθόδου σε νέα δεδομένα:* Όλες οι μέθοδοι απαιτούν μαθηματική επεξεργασία πινάκων τιμών, διαδικασία που μπορεί να είναι χρονοβόρα και πολύπλοκη αν εξαρτώνται κάποιοι υπολογισμοί από τη μορφή και τον αριθμό των ίδιων μας των δεδομένων. Η δυνατότητα, όμως, να μπορούν να κατασκευαστούν μοντέλα και προγραμματιστικές ρουτίνες, που εύχρηστα και ταχύτατα μπορούν στο μέλλον να επαναχρησιμοποιηθούν, χωρίς την εκ νέου κατασκευή ενός μαθηματικού μοντέλου, κρίθηκε σημαντική. Με μια ευέλικτη μέθοδο μπορούμε ακόμα πιο εύκολα να επαληθεύσουμε τις μετρήσεις μας και να πραγματοποιήσουμε και άλλες σε περιοχές τιμών που δεν παρουσιάζουν ομαλότητα

ως προς τα αποτελέσματά τους, όπως και τελικά έγινε και θα δούμε στη συνέχεια του κεφαλαίου.

- *Ανεκτό υπολογιστικό κόστος:* Η μέτρηση δυνάμεων και ροπών μέσω του αισθητήρα μας, απαιτεί ταυτόχρονα την τάχιστη βαθμονόμησή τους μέσω του λογισμικού που θα δημιουργήσουμε, ώστε να μην υπάρχει χρονική καθυστέρηση τέτοια που θα καθιστούσε προβληματική μια εφαρμογή ελέγχου σε πραγματικό χρόνο (real time). Οποιαδήποτε μέθοδος, που θα απαιτούσε κρίσιμο υπολογιστικό χρόνο για τον υπολογισμό αυτό κατά τη διάρκεια των μετρήσεων, θα μπορούσε να επηρεάσει αρνητικά τη σωστή λειτουργία του ρομποτικού βραχίονα και την αλληλεπίδρασή του με το περιβάλλον. Έτσι, ανεξάρτητα από τον όγκο επεξεργασίας δεδομένων πριν τη λειτουργία του αισθητήρα, πρέπει να ελαχιστοποιήσουμε τους υπολογισμούς κατά τη φάση της λειτουργίας του.
- *Εφαρμογή κοινής μεθόδου για όλους τους βαθμούς ελευθερίας:* Η ύπαρξη κοινής μεθόδου για τη βαθμονόμηση και των έξι διαφορετικών μετρούμενων φορτίσεων θα έχει ως αποτέλεσμα έναν εύχρηστο, συμπυκνωμένο και εύκολα τροποποιήσιμο κώδικα προγραμματισμού με όλα τα θετικά της ευελιξίας για τη μελλοντική του χρήση.

3.441 Εφαρμογή πολυωνυμικής συνάρτησης προσέγγισης και μειονεκτήματα

Η χρήση πολυωνύμων διαφόρων βαθμών, ως εργαλείο παρεμβολής και προσέγγισης, παρουσιάζει ένα πλήθος πλεονεκτημάτων αλλά και ένα κύριο μειονέκτημα. Τα βασικότερα πλεονεκτήματα είναι:

• Τα πολυώνυμα έχουν απλές μαθηματικές εκφράσεις.

• Ο υπολογισμός των αγνώστων συντελεστών των πολυωνύμων ακολουθεί εύκολες και προφανείς διαδικασίες,

• Η χρήση ενός πολυωνύμου, του οποίου έχουν ήδη βρεθεί οι τιμές των συντελεστών του, ώστε να υπολογιστεί η τιμή απόκρισης y που αντιστοιχεί σε μια οποιαδήποτε τιμή εισόδου x, έχει πρακτικά μηδενικό υπολογιστικό κόστος,

• Τα πολυώνυμα είναι συνεχώς διαφορίσιμες συναρτήσεις (πρακτικά, όσες φορές χρειαστεί).

• Τα πολυώνυμα έχουν "ελεγχόμενο" σφάλμα αριθμητικής παρεμβολής.

Από την άλλη πλευρά, το κύριο μειονέκτημά τους είναι ότι ανάλογα με τα κομβικά σημεία (x_i, y_i) , i = 0, . . . ,N με τα οποία δημιουργείται το πολυώνυμο και το βαθμό του πολυωνύμου, εύκολα μπορούν να εμφανιστούν ταλαντωτικές συμπεριφορές, που τις περισσότερες φορές δεν συμβαδίζουν με τη "φυσική" του αντίστοιχου προβλήματος στο οποίο αναφέρονται.

Η προσέγγιση μιας καμπύλης, η οποία περιγράφεται διακριτά με N+1 δεδομένα σημεία, μπορεί να πραγματοποιηθεί αναζητώντας το πολυώνυμο εκείνο (βαθμού που συνήθως έχει αποφασισθεί εκ των προτέρων) το οποίο την προσεγγίζει με το βέλτιστο τρόπο (best fit). Διατηρώντας χαμηλό το βαθμό του πολυωνύμου (και σαφώς μικρότερο του N) η καμπύλη που παριστά το πολυώνυμο δεν θα διέρχεται από μερικά ή όλα τα δεδομένα σημεία.

Αυτό που προηγουμένως αναφέρθηκε ως βέλτιστος τρόπος μπορεί ασφαλώς να ερμηνευθεί και να εκφρασθεί μαθηματικά με πολλούς τρόπους. Η συνήθης ερμηνεία του είναι αυτή που επιβάλλει την ελαχιστοποίηση της απόκλισης (deviation) μεταξύ της καμπύλης προσέγγισης και των N + 1 δεδομένων σημείων. Αλλά και ο όρος απόκλιση δέχεται με τη σειρά του πολλαπλές ερμηνείες. Εδώ, αν g(x) είναι το πολυώνυμο προσέγγισης, θα ορίσουμε ως απόκλιση για την τιμή εισόδου x_i , i = 0, . . . , N τη διαφορά

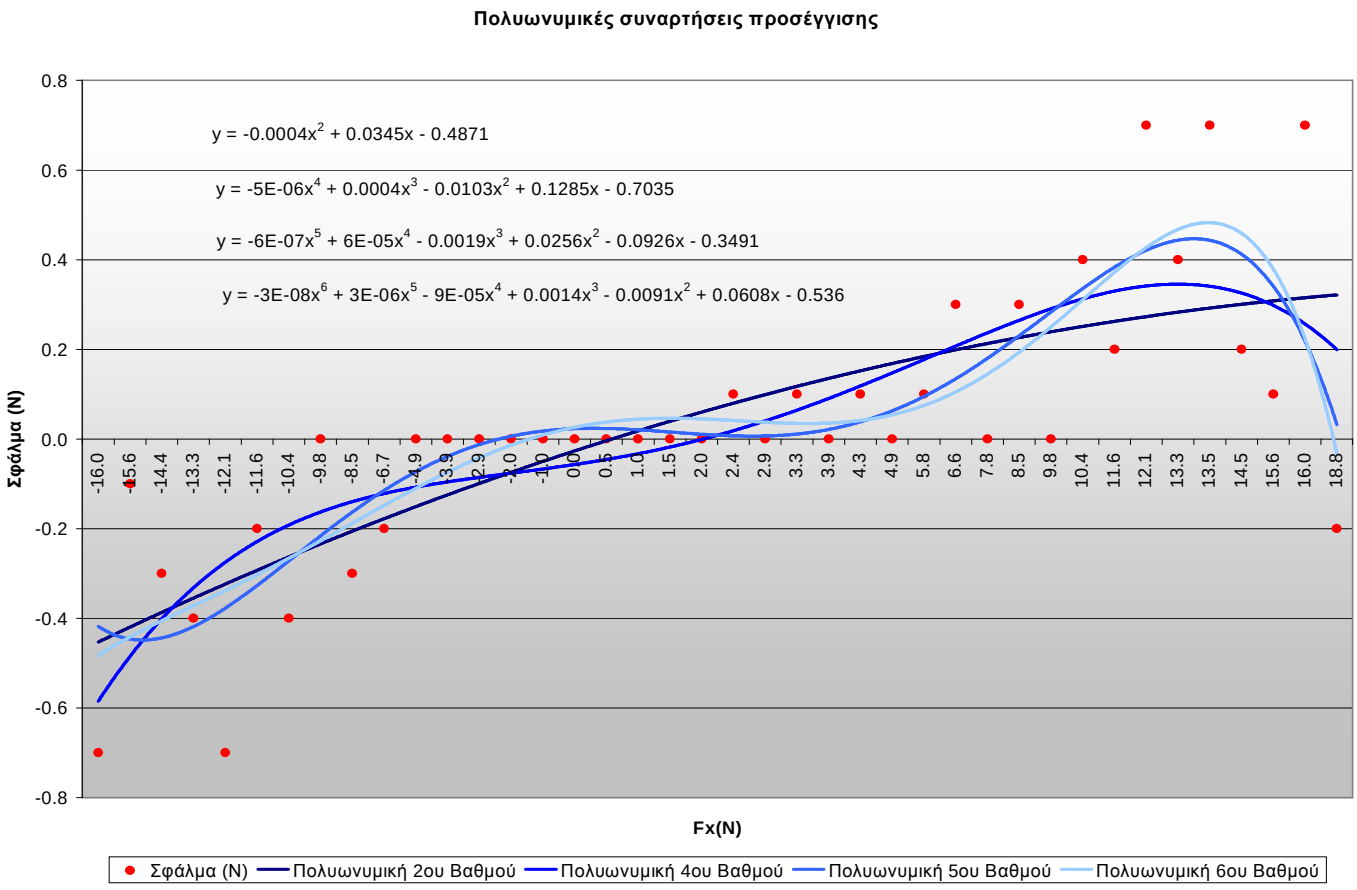
$$e_i = g(x_i) - y_i$$

ενώ η ζητούμενη ελαχιστοποίηση της απόκλισης διατυπώνεται μαθηματικά ως ελαχιστοποίηση του αθροίσματος των τετραγώνων των N + 1 κομβικών αποκλίσεων. Άρα η απαίτηση μας είναι:

$$\sum_{i=0}^{N} (e_i)^2 = \min imum$$

Η τελευταία έκφραση δεν είναι και η μοναδική δυνατότητα. Εναλλακτικά θα μπορούσε να εφαρμοσθεί το λεγόμενο κριτήριο minimax, το οποίο καθορίζει ως βέλτιστη καμπύλη (συνήθως ευθεία) προσέγγισης εκείνη από την οποία ελαχιστοποιείται η μέγιστη απόλυτη τιμή της απόκλισης. Το κριτήριο minimax συνήθως παράγει κακής ποιότητας προσεγγίσεις, αν υπάρχει ένα σημείο από τα N + 1 δεδομένα που είναι αρκετά απομακρυσμένο από τα υπόλοιπα και για αυτό δεν θα το χρησιμοποιήσουμε.

Για να υλοποιήσουμε τα ανωτέρω χαράσσουμε πολυωνυμικές συναρτήσεις προσέγγισης στα διαγράμματα ένδειξης-σφάλματος, όπως αυτά παρουσιάζονται στο Κεφάλαιο 3.43. Για την οικονομία της ανάλυσης παραθέτουμε και μελετάμε έναν βαθμό ελευθερίας του αισθητήρα, τη μέτρηση της δύναμης Fx, που η συμπεριφορά του είναι χαρακτηριστική και των υπολοίπων βαθμών ελευθερίας και για αυτό τα αποτελέσματα είναι πανομοιότυπα. Δοκιμάστηκαν πολυώνυμα $2^{ου}$, $4^{ου}$, $5^{ου}$ και $6^{ου}$ βαθμού με τη μέθοδο ελαχίστων τετραγώνων, οι συναρτήσεις των οποίων υπολογίστηκαν με χρήση του λογισμικού πακέτου Ms Excel και παρουσιάζονται πάνω στο Διάγραμμα 13.



Διάγραμμα 13. Εφαρμογή πολυωνυμικών συναρτήσεων για την προσέγγιση του σφάλματος του σήματος.

Είναι προφανές πως με την αύξηση του βαθμού του πολυωνύμου έχουμε βελτίωση της προσέγγισης των μετρήσεών μας σε όλο το εύρος τους. Ιδιαίτερα στις μεγάλες τιμές της Fx δύναμης μόνο τα πολυώνυμα από 5^{ου} βαθμού και πάνω μπορούν να προσομοιώσουν την «τάση» των αποκλίσεων των μετρούμενων τιμών.

Παρατηρούμε ότι η ταλαντωτική συμπεριφορά του σφάλματος σε κάποιες πολύ κοντινές περιοχές τιμών δεν μπορεί να προσεγγιστεί επαρκώς ούτε από πολυώνυμα κατά πολύ μεγαλύτερου βαθμού. Έτσι, η χρήση μιας πολυωνυμικής συνάρτησης, έστω και καθολικής για όλο το εύρος μετρήσεων μιας εκ των φορτίσεων του αισθητήρα μας, δε μας εξασφαλίζει την ακριβή προσομοίωση της απόκλισης dF (ή dM). Κατά συνέπεια δεν μας είναι εύχρηστη μια τέτοια τεχνική για την πλήρη εξασφάλιση της προκαθορισμένης ακρίβειας που επιθυμούμε να επιτύχουμε (Υπενθυμίζουμε $\pm 0{,}1N$ για τις δυνάμεις και $\pm 0{,}1N \cdot m/10$ για τις ροπές).

Όμως η μέθοδος της πολυωνυμικής προσέγγισης πληροί επαρκώς τα κριτήρια περί συνέχειας τιμών και παραγώγων τής χρησιμοποιούμενης για βαθμονόμηση συνάρτησης (πρακτικά απείρως διαφορίσιμη, όπως προαναφέραμε), τον εύκολο επαναπροσδιορισμό των τιμών και συντελεστών του σε τυχόν αλλαγές των μετρήσεων και στο σχεδόν μηδενικό

υπολογιστικό χρόνο εύρεσης και υπολογισμού μιας ενδιάμεσης τιμής, χάρη στην απλότητα των εκφράσεών τους.

Το κριτήριο της ακρίβειας των αποτελεσμάτων παρόλα αυτά, με τη μη πληρότητά του, καθορίζει την απόρριψη αυτής της μεθόδου. Παρότι η μέθοδος αυτή δοκιμάστηκε μόνο σε έναν βαθμό ελευθερίας του αισθητήρα, η απαίτηση για εφαρμογή μιας κοινής μεθόδου για όλους τους υπόλοιπους βαθμούς ελευθερίας, καθώς και η σχετική ομοιότητα που παρουσιάζουν αυτοί στη συμπεριφορά τους, κρίνουν άσκοπη τη δοκιμή και εφαρμογή της στις μετρήσεις των υπόλοιπων φορτίσεων.

Ως εκ τούτου, η δοκιμή και εφαρμογή μεθόδων με χρήση περισσοτέρων της μίας συνάρτησης παρεμβολής ή προσέγγισης κρίνεται απαραίτητη, και σε τέτοιες μεθόδους θα επικεντρωθούμε και στη συνέχεια. Η προσπάθεια να διατηρηθεί η χρήση πολυωνύμων, και αν είναι δυνατόν όχι ιδιαίτερα πολύ υψηλού βαθμού, με όλα τα πλεονεκτήματα που αυτό μας προσφέρει, θα συνεχιστεί με μεγάλη προσοχή για την αποφυγή ταλαντωτικών συμπεριφορών στα ενδιάμεσα σημεία.

3.442 Εφαρμογή τμηματικών συνεχών πολυωνύμων και μειονεκτήματα

Στην προηγούμενη μέθοδο που αναλύσαμε η πολυωνυμική προσέγγιση στηρίχθηκε στη δημιουργία ενιαίου πολυωνύμου που ικανοποιούσε τη δεδομένη θέση των N+1 σημείων στο επίπεδο (x , y). Εναλλακτικά είναι δυνατόν η προσέγγιση (ή η παρεμβολή σε ανάλογες περιπτώσεις) να στηριχθεί και πάλι σε πολυώνυμα, χωρίς όμως το πολυώνυμο να προκύπτει από το συνυπολογισμό και των N + 1 δεδομένων σημείων μας, αλλά μόνο από ένα μικρό υποσύνολό τους.

Στην περίπτωση αυτή η αριθμητική προσέγγιση με τμηματικά συνεχή πολυώνυμα, και γραφικά η τελική καμπύλη, αποτελείται από τη σύνδεση διαδοχικών καμπυλών. Καθεμιά από αυτές τις καμπύλες αντιπροσωπεύει χαμηλού βαθμού, άρα εύχρηστα και οικονομικά σε υπολογιστικό κόστος, πολυώνυμα με συμπληρωματικά πεδία ορισμού. Το πεδίο ορισμού κάθε πολυωνύμου είναι ένα ή περισσότερα διαδοχικά υποδιαστήματα από την αλληλουχία των N υποδιαστημάτων που ορίζουν τα N + 1 δεδομένα σημεία.

Η κάθε συνάρτηση δεν διέρχεται υποχρεωτικά από τα γνωστά σημεία, αλλά η μορφή της αναπαριστά την "τάση" της πραγματικής συνάρτησης του αισθητήρα με τη μικρότερη δυνατή απόκλιση από τα δεδομένα σημεία. Πρόκειται για τυπική διαχείριση σημείων που προέκυψαν από πειραματικές μετρήσεις με σφάλματα και στην περίπτωση αυτή αναφερόμαστε σε συναρτήσεις προσέγγισης (approximating functions) ή βέλτιστης προσαρμογής (best fit).

Επιλέξαμε το πεδίο ορισμού κάθε πολυωνυμικής συνάρτησης, καθώς και τον βαθμό της, στο δεδομένο πρόβλημά μας με βάση τη συμπεριφορά των τιμών μας. Καθορίστηκαν,

δηλαδή, πεδία ορισμού με μικρότερο εύρος και έγινε χρήση πολυωνύμων μεγαλύτερου βαθμού σε περιοχές απότομης μεταβολής των αποκλίσεων των ενδείξεων από τις πραγματικές τιμές, και αντίθετα συναρτήσεις μικρότερου πολυωνυμικού βαθμού εφαρμοσμένες σε πιο ευρέα πεδία ορισμού χρησιμοποιήθηκαν στις περιοχές τιμών με ομαλότητα στις μεταβολές τους.

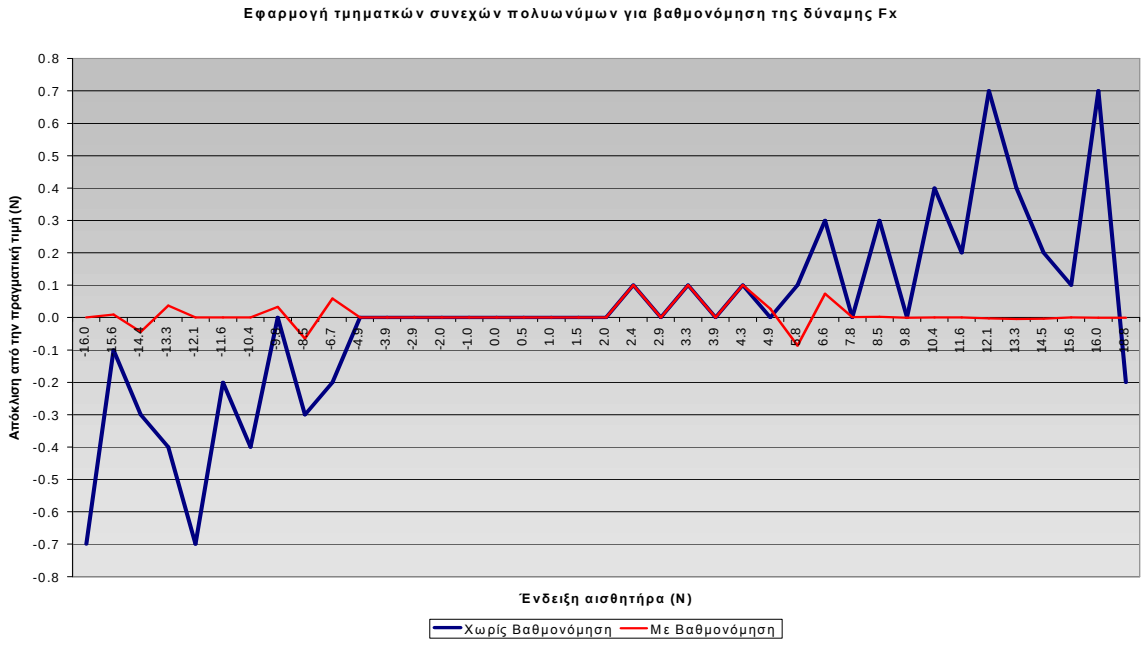
Η κατάστρωση αυτών των εξισώσεων έγινε επίσης με σκοπό να καλύπτεται πλήρως το κριτήριο περί ακριβείας των αποτελεσμάτων ($\pm 0{,}1N$ για τις δυνάμεις και $\pm 0{,}1N \cdot m/10$ για τις ροπές). Έτσι, προέκυψε πως η προεργασία για την εφαρμογή αυτής της μεθόδου βαθμονόμησης απαιτεί επεξεργασία πολλών παραμέτρων, με αποτέλεσμα μια αλλαγή τους να απαιτεί επαναπροσδιορισμό από την αρχή όλων των εξισώσεων και των πεδίων ορισμού που μας χρειάζονται. Αυτό καθιστά δυσκίνητη αυτή τη μέθοδο όσον αφορά σε μια γενίκευσή της σε περισσότερες από μία εφαρμογές.

Δοκιμαστικά εφαρμόσαμε τη μέθοδο των τμηματικών συνεχών πολυωνύμων για βαθμονόμηση σε δύο βαθμούς ελευθερίας του αισθητήρα μας, στις δυνάμεις Fx και Fy. Οι αναλυτικές συναρτήσεις για τη διόρθωση των ενδείξεων του αισθητήρα, μέσω των συναρτήσεων υπολογισμού της απόκλισης dF για αυτές τις δύο φορτίσεις, καθώς και τα πεδία ορισμού των συναρτήσεων αυτών, παρουσιάζονται ενδεικτικά στο Παράρτημα Α.3

Οι συναρτήσεις αυτές είναι προσεγγιστικά πολυώνυμα το πολύ μέχρι της τάξεως $2^{ου}$ βαθμού, δηλαδή έχουν ευχρηστία και απλότητα. Δόθηκε μεγάλη προσοχή, ώστε να έχουμε συνέχεια των τιμών στα όρια των επιμέρους πεδίων ορισμών, για να έχουμε και συνέχεια στο συνολικό πεδίο ορισμού. Το αντίθετο θα προκαλούσε μεγάλα προβλήματα σε μελλοντικές εφαρμογές.

Στα ακόλουθα Διαγράμματα 14 και 15 παρουσιάζεται το σφάλμα του αισθητήρα πριν και μετά την εφαρμογή αυτής της μεθόδου στους δύο βαθμούς ελευθερίας που τη δοκιμάσαμε.

Παρατηρούμε πως και στις δύο περιπτώσεις, που ούτως ή άλλως έχουν παρόμοια συμπεριφορά, μετά την εφαρμογή της μεθόδου των τμηματικών συνεχών πολυωνύμων, οι αποκλίσεις των ενδείξεων της φόρτισης μέσω του αισθητήρα, σε σχέση με τις αναμενόμενες πραγματικές τιμές, είναι σχεδόν μηδενικές.



Διάγραμμα 14. Εφαρμογή τμηματικών συνεχών πολυωνύμων για τη βαθμονόμηση της δύναμης Fx.

Διάγραμμα 15. Εφαρμογή τμηματικών συνεχών πολυωνύμων για τη βαθμονόμηση της δύναμης Fy.

Όμως η συνέχεια των συναρτήσεων που απαιτήσαμε με τη χρήση πολυωνύμων και θέτοντάς την ως παράμετρο στα σημεία σύνδεσης δεν μας εξασφαλίζει και τη συνέχεια της κλίσης της συνάρτησής μας. Μπορεί με μια πρώτη εκτίμηση κάτι τέτοιο να μην φαίνεται τόσο υπολογίσιμο, όμως, όπως προαναφέραμε, ενδέχεται να θέσει σε κίνδυνο την ομαλή λειτουργία της διάταξής μας, όταν οι τιμές των φορτίσεών μας βρίσκονται κοντά σε αυτές τις περιοχές.

Έτσι, παρά τα πλεονεκτήματα που αναδεικνύονταν μέσω αυτής της μεθόδου, κρίθηκε απαραίτητο η δοκιμή και εφαρμογή μιας αντίστοιχης, με συγκριτικά μεγαλύτερο υπολογιστικό κόστος μεν, αλλά με εξάλειψη των επικίνδυνων μειονεκτημάτων της παρούσας. Αποφασίσαμε, τέλος, να χρησιμοποιήσουμε μέθοδο παρεμβολής και όχι προσέγγισης, όπως μέχρι τώρα, δηλαδή συνάρτησης που να διέρχεται και από τα N + 1 σημεία της γραφικής απεικόνισης του σφάλματος του αισθητήρα.

3.443 Θεωρία παρεμβολής μέσω κυβικών splines

Οι κυβικές splines χρησιμοποιούνται σήμερα ευρύτατα ως μέθοδος τμηματικά συνεχούς παρεμβολής. Τα δεδομένα κομβικά σημεία (x_i, y_i) αντιμετωπίζονται κατά ζεύγη διαδοχικών σημείων που μεταξύ τους εφαρμόζεται κυβική παρεμβολή.

Μεταξύ διαδοχικών κομβικών σημείων (x_i, y_i) και (x_{i+1}, y_{i+1}) τα πολυώνυμα παρεμβολής για τις συντεταγμένες x και y δίνονται παραμετρικά από τις εκφράσεις:

$$g_x(u) = a_0 + a_1 u + a_2 u^2 + a_3 u^3$$

$$g_y(u) = b_0 + b_1 u + b_2 u^2 + b_3 u^3 \quad (1)$$

με τους οκτώ συντελεστές $\alpha_o, \ldots, b_3$ να έχουν διαφορετικές τιμές στα επιμέρους διαστήματα. Τονίζεται ότι, παρότι θα μπορούσαμε να χρησιμοποιήσουμε διπλούς δείκτες όπως $\alpha_{0,j}$ αντί α_0, το αποφεύγουμε για λόγους απλότητας. Στη συνέχεια θα ασχοληθούμε με την παρουσίαση του υπολογισμού των συντελεστών α_j , j = 0, 3 για το διάστημα (x_i, y_i)*)* ως (x_{i+1}, y_{i+1}) αφού ο υπολογισμός των b_j είναι ακριβώς όμοιος.

Για τον υπολογισμό τους επιβάλλουμε αρχικά τις δύο προφανείς απαιτήσεις:

$$g_x(0) = a_0 = x_i$$

$$g_x(1) = a_0 + a_1 + a_2 + a_3 = x_{i+1} \qquad (2)$$

Ας συμβολίσουμε με M_i , i = 0, . . . , N τις τιμές της δεύτερης παραγώγου όπου

$$M_i = (\ddot{g}_x(u))_i = \left(\frac{d^2 x}{du^2}\right)_i$$

στα N + 1 δεδομένα κομβικά σημεία.

Με βάση τη μαθηματική έκφραση των συναρτήσεων (1) η δεύτερη παράγωγός της είναι:

$$\ddot{g}_x(u) = 2a_2 + 6a_3 u$$

ενώ για τα κομβικά σημεία που βρίσκονται στα άκρα κάθε διαστήματος μπορούμε να γράψουμε ότι:

$$\ddot{g}_x(0) = M_i = 2a_2$$

$$\ddot{g}_x(1) = M_{i+1} = 2a_2 + 6a_3$$

οπότε $\alpha_2 = M_i/2$ και $6\alpha_3 = M_{i+1} - M_i$ από τις οποίες προκύπτει εύκολα η (γραμμική ως προς u) έκφραση για τη δεύτερη παράγωγο, που είναι η:

$$\ddot{g}_x(u) = M_i + (M_{i+1} - M_i)u \qquad (3)$$

Ολοκληρώνοντας στη συνέχεια δύο φορές την εξίσωση (3) προκύπτει τελικά ότι:

$$g_x(u) = \frac{1}{2}M_i u^2 + \frac{1}{6}(M_{i+1} - M_i)u^3 + A + Bu \qquad (4)$$

όπου εμφανίζονται δύο σταθερές ολοκλήρωσης *A* και B. Ο υπολογισμός των A και B γίνεται απαιτώντας να ικανοποιηθούν οι σχέσεις (2) από τις οποίες προκύπτει:

$$A = x_i$$

$$B = x_{i+1} - x_i - \frac{1}{6}M_{i+1} - \frac{1}{3}M_i$$

και διατυπώνεται έτσι η τελική μορφή της (4) ως

$$g_x(u) = x_i + \left[(x_{i+1} - x_i) - \frac{1}{6}M_{i+1} - \frac{1}{3}M_i \right] \cdot u \; + \frac{1}{2}M_i u^2 + \frac{1}{6}(M_{i+1} - M_i) \cdot u^3 \qquad (5)$$

Η σχέση αυτή αντικαθιστά τη γενική γραφή (1) με συντελεστές οι οποίοι πλέον διαθέτουν συγκεκριμένες εκφράσεις.

Με βάση τη σχέση (5) η υλοποίηση του σχήματος θα ήταν εφικτή, αρκεί να ήταν διαθέσιμες οι τιμές των παραγώγων M_i , i = 0, . . . , N. Είναι εύλογο να συμπεράνει κανείς ότι η διαθεσιμότητα τέτοιας πληροφορίας δεν είναι εύκολη και κάθε μέθοδος που την απαιτεί γίνεται εκ προοιμίου ένα πρακτικά μη χρήσιμο εργαλείο. Η ιδέα για να ξεπεραστεί η ανάγκη για την επιπλέον πληροφορία, που εκ πρώτης όψεως απαιτεί η μέθοδος, είναι αυτή να παραχθεί έμμεσα από λογικές συνθήκες που θα επιβληθούν στα δεδομένα κομβικά σημεία.

Μια λογική (και που όπως στη συνέχεια θα αποδειχτεί πολύ βολική) συνθήκη είναι να απαιτήσουμε τα τμηματικά συνεχή πολυώνυμα να έχουν συνεχείς πρώτες παραγώγους στα εσωτερικά κομβικά σημεία. Με τον όρο εσωτερικά εννοούμε όλα τα σημεία πλην των ακραίων (x_o, y_o) και (x_{N+1}, y_{N+1}). Επιλέγουμε αρχικά το διάστημα $(x_i, y_i) \rightarrow (x_{i+1}, y_{i+1})$ στο οποίο η πρώτη παράγωγος του g_x σύμφωνα με τη σχέση (5) είναι:

$$\dot{g}_x(u) = x_{i+1} - x_i - \frac{1}{6}M_{i+1} - \frac{1}{3}M_i + M_i u \ + \frac{1}{2}(M_{i+1} - M_i)\cdot u^2 \qquad (6)$$

Ομοίως, για το προηγούμενο διάστημα $(x_{i-1}, y_{i-1}) \rightarrow (x_i, y_i)$ ισχύει ότι:

$$\dot{g}_x(u) = x_i - x_{i-1} - \frac{1}{6}M_i - \frac{1}{3}M_{i-1} + M_{i-1} u \ + \frac{1}{2}(M_i - M_{i-1})\cdot u^2 \qquad (7)$$

Η τιμή της (6) για u = 0 και της (7) για u = 1 αντιστοιχούν στο ίδιο σημείο, τον κόμβο i, και ως εκ τούτου πρέπει να ταυτίζονται. Εξισώνοντας προκύπτει ότι:

$$M_{i-1} + 4M_i + M_{i+1} = 6(x_{i-1} - 2x_i + x_{i+1}) \qquad (8)$$

Εξισώσεις της μορφής της (8) διατυπώνονται για όλους τους κόμβους i = 1, . . . , N - 1, εκτός δηλαδή των δύο ακραίων, όπως και προαναφέρθηκε. Έτσι απομένουν άλλες δύο εξισώσεις ως προς τα M_i, ώστε να διατυπωθεί ένα γραμμικό σύστημα με N+1 εξισώσεις για N+1 αγνώστους. Οι υπόλοιπες δύο εξισώσεις αναγκαστικά προκύπτουν επιβάλλοντας δύο οριακές συνθήκες στον πρώτο και στον τελευταίο κόμβο. Υπάρχουν διάφορες δυνατές περιπτώσεις και αναλύονται ως εξής:

- Να γνωρίζουμε ή έστω να υποθέτουμε ότι οι δεύτερες παράγωγοι στα ακραία κομβικά σημεία είναι μηδενικές. Συνεπώς οι δύο νέες εξισώσεις που συμπληρώνουν το σύστημα είναι οι:

  $$M_o = 0 \quad , \quad M_N = 0 \qquad (9)$$

  και τότε αναφερόμαστε σε φυσικές *splines.*

- Να είναι γνωστές οι κλίσεις, δηλαδή οι τιμές των πρώτων παραγώγων $\dot{g}_x$ και $\dot{g}_y$ στα κομβικά σημεία (x_o, y_o) και (x_{N+1}, y_{N+1}). Η γνωστή τιμή της $\dot{g}_x$ στο (x_o, y_o), έστω d_0, συσχετίζεται με τις γειτονικές τιμές των x_i και M_i μέσω της σχέσης:

  $$2M_0 + M_i = 6(x_i - x_0 - d_0) \qquad (10)$$

  η οποία αποδεικνύεται με βάση την έκφραση (6). Με όμοιο τρόπο στο διάστημα $(x_{N-1}, y_{N-1}) \rightarrow (x_N, y_N)$ και για u = 1 αποδεικνύεται ότι η αντίστοιχη σχέση (10) είναι η:

  $$-M_{N-1} + 2M_N = 6(-x_N + x_{N-1} + d_N) \qquad (11)$$

  όπου με d_N συμβολίστηκε η γνωστή τιμή της $\dot{g}_x$ στο (x_N, y_N).

- Να γίνεται η υπόθεση ότι υπάρχει σταθερή κλίση δεύτερης παραγώγου στα άκρα, δηλαδή:

  $$M_0 = M_1 \text{ και } M_N = M_{N-1}$$

Ανεξάρτητα από την επιλογή των οριακών συνθηκών έχουμε σαν αποτέλεσμα τη διατύπωση ενός συστήματος N+1 εξισώσεων με N+1 αγνώστους, δηλαδή τα M_i , i = 0, . . . , N. Με βάση τα προηγούμενα το σύστημα αυτό παίρνει την εξής μορφή:

$$\begin{bmatrix} \gamma_{0,0} & \gamma_{0,1} & & & & & \\ 1 & 4 & 1 & & & & \\ & 1 & 4 & 1 & & & \\ & & & \vdots & & & \\ & & & 1 & 4 & 1 & \\ & & & & \gamma_{N,N-1} & \gamma_{N,N} \end{bmatrix} \begin{bmatrix} M_0 \\ M_1 \\ M_2 \\ \vdots \\ M_{N-1} \\ M_N \end{bmatrix} = 6 \begin{bmatrix} h_0 \\ x_0 - 2x_1 + x_2 \\ x_1 - 2x_2 + x_3 \\ \vdots \\ x_{N-2} - 2x_{N-1} + x_N \\ h_N \end{bmatrix} \quad (12)$$

όπου με $\gamma_{i,j}$ και h_i συμβολίστηκαν όροι που προκύπτουν από την επιλογή των δύο οριακών συνθηκών. Η επίλυση του συστήματος (10) απαιτεί ένα "υποπρόγραμμα" που επιλύει γραμμικά συστήματα με το μητρώο των συντελεστών να έχει τριδιαγώνια μορφή. Έχοντας υπολογίσει τους συντελεστές M_i , i = 0, . . . , N, δημιουργείται η συνάρτηση παρεμβολής με χρήση της εξίσωσης (5) τμηματικά, σε κάθε δηλαδή διάστημα μεταξύ δύο διαδοχικών σημείων από τα δεδομένα κομβικά σημεία. Η ίδια διαδικασία επαναλαμβάνεται και για τη συνάρτηση $g_y(u)$.

Είναι σαφές ότι οι κυβικές splines είναι μια μέθοδος που διορθώνει αρκετά από τα προβλήματα της χρήσης τμηματικά συνεχών πολυωνύμων με χρήση άλλων μεθόδων. Όμως, παρόλα αυτά, παρουσιάζει πολλές φορές προβλήματα ταλαντώσεων σε κάθε σημείο στο οποίο η δεύτερη παράγωγος δεν είναι συνεχής. Να σημειώσουμε επίσης πως ένα βασικό χαρακτηριστικό των τμηματικά συνεχών κυβικών splines είναι ότι κάθε μεταβολή στις οριακές συνθήκες (αλλαγή τιμής των $\gamma_{i,j}$) ή η μετακίνηση της θέσης ενός σημείου (x_i, y_i) απαιτεί εκ νέου επίλυση του συστήματος (12) και επανακαθορισμό των τιμών των M_i.

Παρόλα αυτά, δεδομένου της εφαρμογής μας και της σταθερότητας των αρχικών μας μετρήσεων, η εκ νέου επίλυση του συστήματος δεν προβλέπεται συχνή. Ακόμα και σε αυτή την περίπτωση φροντίσαμε να υλοποιήσουμε μια εύχρηστη προγραμματιστικά εφαρμογή για την ταχύτατη επίλυση και εύρεση των απαραίτητων συντελεστών. Επίσης, με στενή παρακολούθηση των καμπύλων παρεμβολής στα δεδομένα μας και διορθωτική επέμβαση σε περιοχές με αυξημένη ταλαντωτική συμπεριφορά, είχαμε τη δυνατότητα να αντιμετωπίσουμε το μειονέκτημα αυτής της μεθόδου.

Η εφαρμογή μας ανάγεται στο πρόβλημα εύρεσης μια τιμής του y (στην περίπτωσή μας της διορθωμένης τιμής) για μια εμβόλιμη τιμή του x (τυχαία φόρτιση). Έτσι θα μπορούμε να εκτιμούμε και να υπολογίζουμε την απόκλιση

μέτρησης σε όλες τις δυνατές τιμές και σε κάθε έναν από τους 6 βαθμούς ελευθερίας του αισθητήρα, βασισμένοι στις αρχικές μας μετρήσεις. Για την υλοποίηση αυτού του στόχου και έχοντας επιλύσει το σύστημα (12) και υπολογίσει τους συντελεστές M_i, βρίσκουμε το διάστημα $(x_i, y_i) \rightarrow (x_{i+1}, y_{i+1})$ με $x_i \leq x < x_{i+1}$, έπειτα επιλύουμε την εξίσωση (5) για τη δεδομένη τιμή x, ώστε να υπολογισθεί το αντίστοιχο u, και τέλος με την τιμή u που προέκυψε χρησιμοποιούμε την αντίστοιχη εξίσωση για την εύρεση του y.

3.444 Εφαρμογή παρεμβολής με χρήση κυβικών splines και διορθώσεις

Για την ελαχιστοποίηση των πράξεων θα κάνουμε παρεμβολή σε διάγραμμα που θα απεικονίζει την $F_{πραγματική}$ σε σχέση με την $F_{ένδειξης}$, ενσωματώνοντας έτσι τις αποκλίσεις που υπολογίστηκαν εμμέσως στις συναρτήσεις μας και παίρνοντας σαν αποτέλεσμα κατευθείαν την τιμή που χρειαζόμαστε. Παρόλα αυτά για την ανάλυση της αποτελεσματικότητας και αυτής της μεθόδου, αλλά και του ελέγχου ακριβείας της θα χρησιμοποιήσουμε και πάλι διαγράμματα (ένδειξης) / (απόκλισης από την πραγματική τιμή), που προσφέρουν καλύτερη εποπτεία και πιο ευδιάκριτη αντίληψη της προσομοίωσης και αντιστάθμισης του σφάλματος.

Για να ξεκινήσουμε είναι απαραίτητο να προγραμματίσουμε μία ρουτίνα που να επιλύει το σύστημα (12) του Κεφαλαίου 3.443, ώστε να υπολογίσουμε τους συντελεστές M_i , i = 0, . . . , N, τις τιμές της $2^{ης}$ παραγώγου, δηλαδή, στα κομβικά σημεία. Ο κώδικας αυτός είναι σε γλώσσα προγραμματισμού C και μπορεί να χρησιμοποιηθεί και για τους 6 βαθμούς ελευθερίας του αισθητήρα, αλλάζοντας τις συντεταγμένες των σημείων των διανυσμάτων x [i] και y [i] και των αρχικών συνθηκών της τιμής της 1ης παραγώγου της συνάρτησης παρεμβολής στο σημείο 1 και στο σημείο N, δηλαδή των yp1 και ypn.

Αναλύοντας την προγραμματιστική ρουτίνα *spline.c* (βλέπε Παράρτημα Α.4.1) που τελικά χρησιμοποιήσαμε παρατηρούμε τα εξής:

Δίνουμε σαν είσοδο της ρουτίνας τα διανύσματα x [1.....N] και y [1.....N] που εκφράζουν τα σημεία (x_i , y_i) του επιπέδου μέσω των οποίων θέλουμε να σχηματιστεί η καμπύλη παρεμβολής. Επίσης δίνουμε, όπως αναφέραμε, τις τιμές y' [1] και y'[N], αλλαγή των οποίων απαιτεί επανάληψη από την αρχή όλων των υπολογισμών. Εάν η τιμή της y' [1] ή και της y'[N] είναι ίση ή μεγαλύτερη από την τιμή 1×10^{30} , που εκφράζει μια σχεδόν απειρίζουσα πρώτη παράγωγο, η ρουτίνα σηματοδοτείται να θέσει τις αντίστοιχες οριακές συνθήκες σε φυσικές Splines, δηλαδή με μηδενική δεύτερη παράγωγο σε αυτό το όριο. Εάν η τιμή δεν ξεπερνά το 1×10^{30} , τότε οι οριακές συνθήκες καθορίζονται με βάση μιας συγκεκριμένης τιμής της πρώτης παραγώγου.

Στη συνέχεια μέσω ενός βρόχου (loop) ανάλυσης των επιμέρους συντελεστών του τριδιαγωνιαίου αλγόριθμου χρησιμοποιούμε τα y2 και u διανύσματα για να αποθηκεύσουμε

προσωρινά τους συντελεστές που απεμπλέχτηκαν. Έπειτα με τη χρήση ενός ακόμα βρόχου του τριδιαγωνιαίου αλγορίθμου προχωρούμε σε μια ανάποδη διαδικασία αντικατάστασης των συντελεστών που ψάχνουμε.

Ο κώδικας έτσι μας επιστρέφει ένα διάνυσμα y'' [1.....N] , όπου στην ρουτίνα εμφανίζεται σαν y2(i), το οποίο ουσιαστικά εκφράζει τις τιμές των M_i του συστήματος (12) του Κεφαλαίου 3.443, δηλαδή τις δεύτερες παραγώγους της συνάρτησης παρεμβολής πάνω στα σημεία (x_i , y_i) από τα οποία περνάει.

Απαραίτητη είναι η ενσωμάτωση της βιβλιοθήκης nrutil μέσω των `nrutil.h` και `nrutil.c` για τη δυνατότητα χρήσης διανυσμάτων (όπως στη συγκεκριμένη περίπτωση του u) .

Φροντίζουμε να αποθηκεύσουμε το διάνυσμα αυτό σε ένα αρχείο για μελλοντική χρήση και επεξεργασία. Το αρχείο αυτό είναι της μορφής " *'xx'*y2.dat " , όπου στο πρώτο γράμμα γράφουμε το είδος της φόρτισης (F ή M) και στο δεύτερο τον άξονα φόρτισης (x ή y ή z). Σημειώνουμε ότι η διαδικασία που προηγήθηκε εφαρμόζεται για μία και μόνο φορά για κάθε βαθμό ελευθερίας του αισθητήρα και δεν επιβαρύνει το υπολογιστικό κόστος της συνολικής εφαρμογής κατά τη διάρκεια της λειτουργίας της σε πραγματικό χρόνο

Στη συνέχεια πρέπει να προγραμματίσουμε μια ρουτίνα που για κάθε εμβόλιμη τιμή του x θα μας προσδιορίζει την τιμή του y με βάση τις συναρτήσεις παρεμβολής. Ουσιαστικά, στη δική μας εφαρμογή, με είσοδο της ένδειξης της τιμής του αισθητήρα για τη δεδομένη φόρτιση θα παίρνουμε σαν αποτέλεσμα την πραγματική τιμή της φόρτισης, δηλαδή θα έχουμε μια διαδικασία βαθμονόμησης σε πραγματικό χρόνο.

Για να μπορούμε να ελέγξουμε το σύνολο της διαδικασίας, καθώς και την αποτελεσματικότητά της, σχεδιάσαμε μια ρουτίνα στην οποία μέσω ενός βρόχου μπορούσαμε να υπολογίσουμε και να αποθηκεύσουμε σε ένα αρχείο το σύνολο των "βαθμονομημένων" τιμών y για κάθε δυνατή τιμή της ένδειξης x. Αυτό υλοποιήθηκε μέσω επαναληπτικής διαδικασίας εύρεσης του y από το κάτω έως το άνω όριο των τιμών της κάθε φόρτισης με βήμα 0.1, που αντιστοιχεί στην προκαθορισμένη ακρίβεια που θέσαμε ($0{,}1N$ για τις δυνάμεις και $0{,}1N \cdot m/10$ για τις ροπές)

Υλοποιήσαμε λοιπόν την προγραμματιστική ρουτίνα *splint.c* (βλέπε Παράρτημα Α.4.2) με τα εξής χαρακτηριστικά:

Δίνουμε σαν είσοδο της ρουτίνας τα διανύσματα xa [1.....N] και ya [1.....N] που εκφράζουν τα αντίστοιχα x [1.....N] και y [1.....N] της ρουτίνας spline.c και προσδιορίζουν τα γνωστά σημεία (x_i , y_i) από τα οποία διέρχεται η καμπύλη παρεμβολής. Επίσης δίνουμε το διάνυσμα y2a [1.....N] που αντιστοιχεί στις δεύτερες παραγώγους της συνάρτησης παρεμβολής στα σημεία (x_i , y_i) και ουσιαστικά είναι το διάνυσμα που υπολογίσαμε και αποθηκεύσαμε μέσω της ρουτίνας spline.c.

Στη συνέχεια δημιουργούμε ένα αρχείο στο οποίο θα αποθηκεύσουμε όλα τα σημεία (x_i , y_i) που προσδιορίζουν τις διορθωμένες τιμές της κάθε φόρτισης σε σχέση με τις ενδείξεις του αισθητήρα. Αυτό θα μας βοηθήσει στο να μοντελοποιήσουμε την προσομοίωση του σφάλματος και να μελετήσουμε τη συμπεριφορά των συναρτήσεων κυβικών splines πριν τις χρησιμοποιήσουμε στην εφαρμογή μας.

Ακολούθως, έπειτα και από έναν έλεγχο ότι έχουν μπει σωστά οι τιμές εισόδου, μέσω ενός βρόχου διχοτόμησης (αφού θα αντιμετωπίζουμε "τυχαία" τιμή φόρτισης κάθε φορά στην πραγματική λειτουργία) προσδιορίζεται μεταξύ ποιων γνωστών τιμών βρίσκεται η τυχαία τιμή που θέλουμε να διορθώσουμε προς την πραγματική. Βάσει αυτών των δεδομένων υπολογίζουμε με την πολυωνυμική συνάρτηση κυβικών splines τη νέα διορθωμένη τιμή της φόρτισης.

Έπειτα από την εφαρμογή του κώδικα που προγραμματίσαμε, ελέγχουμε τη διαδικασία, κυρίως μεταξύ των τιμών που έχουμε ήδη συλλέξει, αφού χρησιμοποιούμε συναρτήσεις παρεμβολής και είναι δεδομένα ότι "περνάνε" από τα γνωστά σημεία, αλλά δεν γνωρίζουμε τη συμπεριφορά των συναρτήσεων μακριά από αυτά.

Χρησιμοποιούμε το αρχείο με τις ενδείξεις του κάθε βαθμού ελευθερίας σε σχέση με τις αντίστοιχες βαθμονομημένες τιμές και το μετατρέπουμε σε σημεία σε διάγραμμα που παριστά τις ενδείξεις αυτές του αισθητήρα σε σχέση με τη μεταβολή δF (ή δM) που έχει υπολογιστεί ότι πρέπει να γίνει για τη σωστή βαθμονόμηση. Δηλαδή από σχέση $F_{\text{ένδειξης}}$/ $F_{\text{πραγματική}}$ και τους τύπους:

$$F_{\text{πραγματική}} = F_{\text{ένδειξης}} + \delta F$$

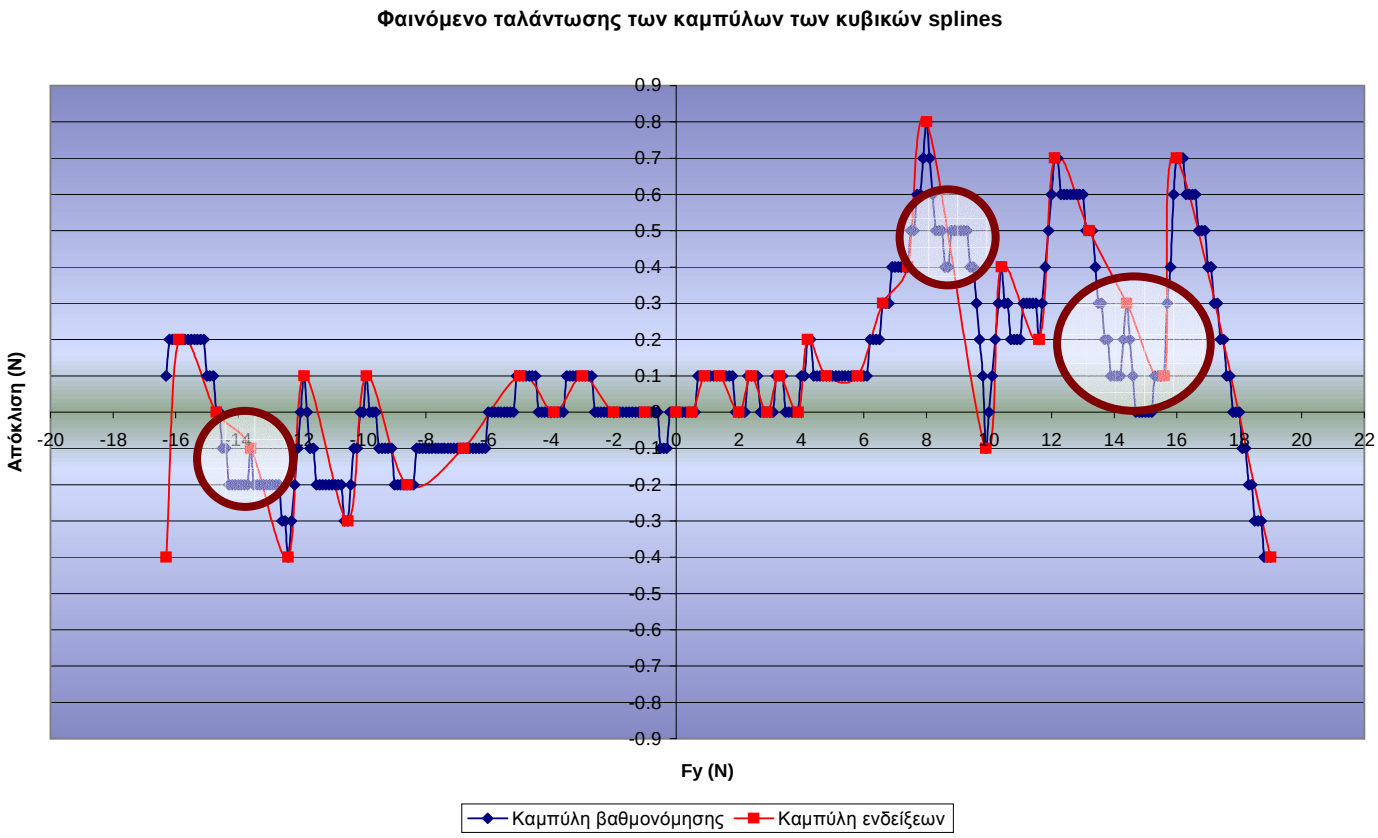
$$M_{\text{πραγματική}} = M_{\text{ένδειξης}} + \delta M$$

του Κεφαλαίου 3.3 κάνουμε διάγραμμα $\delta F/F_{\text{ένδειξης}}$ και $\delta M/M_{\text{ένδειξης}}$, όπου

$\delta F= F_{\text{βαθμονομημένη}} - F_{\text{ένδειξης}}$ θεωρώντας ότι $F_{\text{βαθμονομημένη}} = F_{\text{πραγματική}}$ και

$\delta M= M_{\text{βαθμονομημένη}} - M_{\text{ένδειξης}}$ θεωρώντας ότι $M_{\text{βαθμονομημένη}} = M_{\text{πραγματική}}$

Υπερθέτουμε και τις τιμές που συλλέξαμε στην αρχή με τις αντίστοιχες αποκλίσεις δF (ή δM) από τις πραγματικές και αναμένουμε οι δύο καμπύλες να ταυτιστούν, καθώς η μία είναι προϊόν της άλλης με χρήση κυβικής παρεμβολής με splines. Παρά την αναμενόμενη ταύτιση των καμπυλών παρατηρήθηκε ταλάντωση της καμπύλης προσέγγισης σε κάποιες περιοχές μετρήσεων, άλλες φορές σε λιγότερο βαθμό και άλλες πολύ πιο έντονα, κάτι που θεωρούσαμε πιθανό πρόβλημα από την αρχή.



Διάγραμμα 16. Παρατήρηση φαινομένου ταλάντωσης κατά την αρχική εφαρμογή της παρεμβολής μέσω κυβικών splines.

Υπερθέτουμε και τις τιμές που συλλέξαμε στην αρχή με τις αντίστοιχες αποκλίσεις δF (ή δM) από τις πραγματικές και αναμένουμε οι δύο καμπύλες να ταυτιστούν, καθώς η μία είναι προϊόν της άλλης με χρήση κυβικής παρεμβολής με splines. Παρά την αναμενόμενη ταύτιση των καμπυλών παρατηρήθηκε ταλάντωση της καμπύλης προσέγγισης σε κάποιες περιοχές μετρήσεων, άλλες φορές σε λιγότερο βαθμό και άλλες πολύ πιο έντονα, κάτι που θεωρούσαμε πιθανό πρόβλημα από την αρχή.

Για την αντιμετώπιση του προβλήματος της ταλάντωσης χρησιμοποιήθηκαν δύο τρόποι. Σε περιοχές τιμών όπου τα σημεία απείχαν σχετικά και εμφανιζόταν ταλάντωση μεταξύ τους συλλέξαμε τιμές και σε ενδιάμεσα σημεία για να ομαλοποιήσουμε τις καμπύλες, και επαναλάβαμε τη διαδικασία της βαθμονόμησης. Στις υπόλοιπες περιπτώσεις παρατηρήσαμε ότι εμφανιζόταν ταλάντωση, επειδή είχαμε απότομες μεταβολές του σφάλματος σε πολύ μικρές περιοχές τιμών. Έγινε προσπάθεια εξομάλυνσης των καμπυλών σε αυτές τις περιπτώσεις ελαττώνοντας την τιμή της δεύτερης παραγώγου της συνάρτησης σε γειτονικά σημεία και επαναπροσδιορισμός νέων καμπυλών βαθμονόμησης.

Ως αποτέλεσμα προέκυψαν καμπύλες βαθμονόμησης με ομαλή συμπεριφορά μεταξύ των γνωστών σημείων των φορτίσεων και στους έξι βαθμούς ελευθερίας του αισθητήρα, οι οποίες μετασχηματισμένες σε καμπύλες προσομοίωσης σφάλματος (αφού αυτή η μέθοδος

μας βοηθά, όπως είδαμε, στο να μελετήσουμε σωστά τις συναρτήσεις βαθμονόμησης) ταυτίζονται σχεδόν απόλυτα με τις καμπύλες του πραγματικού σφάλματος

Οι παραστάσεις αυτές παρουσιάζονται στο Παράρτημα Α.5 για περαιτέρω μελέτη και αξιολόγηση.

3.45 Αξιολόγηση βαθμονόμησης με χρήση κυβικών splines και έλεγχος ακριβείας

Έπειτα από την πλήρη ανάλυση της μεθόδου παρεμβολής με χρήση κυβικών splines, καθώς και την αντιμετώπιση των προβλημάτων που παρουσιάστηκαν κατά την εφαρμογή της, είμαστε σε θέση να αξιολογήσουμε την ικανότητα αυτής της μεθόδου να καλύψει τις ανάγκες μας στη βαθμονόμηση του αισθητήρα δύναμης ροπής, σε συσχετισμό και με τα κριτήρια που ορίσαμε στο Κεφάλαιο 3.44.

Οι πολυωνυμικές συναρτήσεις 3ου βαθμού, που χρησιμοποιήθηκαν με σημείο ένωσης τα γνωστά σημεία φορτίσεων, εξασφαλίζουν προφανώς συνέχεια των τιμών σε όλο το πεδίο ορισμού. Η ιδιότητα της μεθόδου των κυβικών splines να διατηρεί τη συνέχεια και της 1ης παραγώγου στα σημεία ένωσης των πολυωνυμικών συναρτήσεων μας καλύπτει πλήρως το κριτήριο συνέχειας της παραγώγου των συναρτήσεων βαθμονόμησης σε όλο το πεδίο ορισμού, καθώς σε όλα τα υπόλοιπα σημεία έχουμε πολυωνυμικές συναρτήσεις που ως γνωστόν μπορούμε να παραγωγίσουμε όσες φορές θέλουμε.

Παρότι η μέθοδος που χρησιμοποιήσαμε απαιτεί εκ νέου υπολογισμό των Mi δεύτερων παραγώγων στα κομβικά σημεία, καθώς και των συναρτήσεων 3ου βαθμού που χρησιμοποιούμε, η χρήση των υπορουτινών που προγραμματίσαμε (βλέπε Παράρτημα Α.4) για τον υπολογισμό των δεδομένων που χρειαζόμαστε καθιστούν αυτή τη διαδικασία γρήγορη και ευέλικτη.

Επίσης να σημειώσουμε πως, επειδή οι περισσότεροι υπολογισμοί γίνονται εκτός χρόνου λειτουργίας, το υπολογιστικό κόστος της μεθόδου σε λειτουργία σε πραγματικό χρόνο είναι αμελητέο, καθώς οι υπολογισμοί που απαιτούνται είναι η εύρεση του πεδίου μεταξύ δύο γνωστών σημείων που βρίσκεται η τιμή που θέλουμε να διορθώσουμε και ο υπολογισμός της διορθωμένης τιμής με ένα πολυώνυμο χαμηλού βαθμού. Η μέθοδος αυτή είναι κοινή για όλους τους βαθμούς ελευθερίας του αισθητήρα, κάτι που κάνει τη διαδικασία πιο συνοπτική και ευέλικτη, αφού χρησιμοποιούμε τις ίδιες ρουτίνες που καλούμε για τον υπολογισμό της κάθε φόρτισης, μεταβάλλοντας απλά κάθε φορά τις αρχικές συνθήκες.

Η ακρίβεια των αποτελεσμάτων της μεθόδου που επιτυγχάνεται σε θεωρητικό επίπεδο είναι απόλυτη, τόσο σε εφαρμογή της μεθόδου στις μετρούμενες δυνάμεις, όσο και στις ροπές, αφού η μέθοδος που χρησιμοποιήσαμε χειρίζεται συναρτήσεις παρεμβολής. Όσον αφορά στην ακρίβεια στις ενδιάμεσες τιμές από τις αρχικά μετρούμενες, αλλά και για να εξασφαλίσουμε πως όλη η διαδικασία έγινε χωρίς λάθη, αποφασίσαμε να εξετάσουμε εκ

νέου την όλη μέθοδο. Με χρήση τυχαίων βαρών (με τη χρήση των πρότυπων βαριδίων, βλέπε Κεφάλαιο 3.41), δηλαδή με εφαρμογή τυχαίας φόρτισης σε κάθε βαθμό ελευθερίας του αισθητήρα, σημειώσαμε τις νέες ενδείξεις που προέκυπταν ήδη "φιλτραρισμένες" από τις ρουτίνες βαθμονόμησης που σχεδιάσαμε. Με αυτές τις τιμές και με συσχέτισή τους με τις πραγματικές τιμές των φορτίσεων (γνωστές από τα βαρίδια) μπορούμε να ελέγξουμε πλέον αν η εφαρμογή της μεθόδου βαθμονόμησης συρρικνώνει το σφάλμα σε επίπεδα κάτω των $\pm 0{,}1N$ για τις δυνάμεις και $\pm 0{,}1N \cdot m/10$ για τις ροπές, που είναι η και αρχική μας απαίτηση.

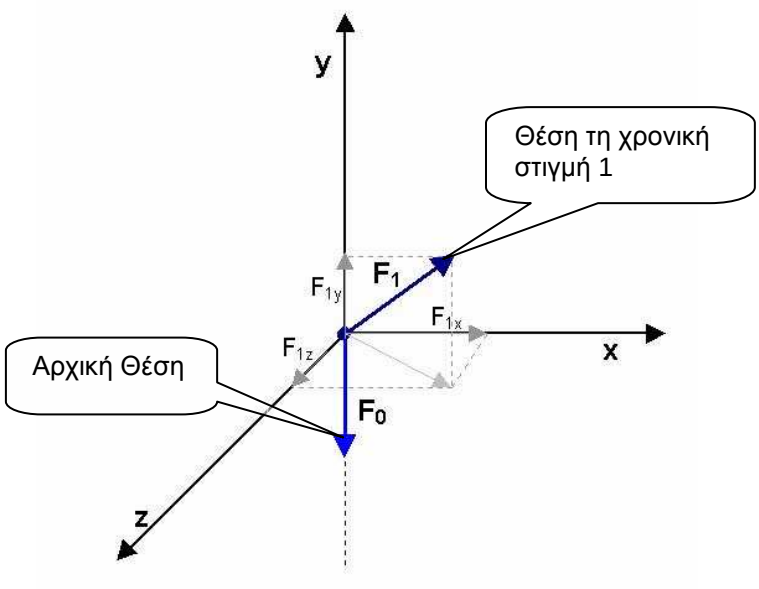
3.46 Έλεγχος και διόρθωση μηδενικής αντιστάθμισης (zero offset)

Όταν μια μετρητική διάταξη μας δίνει ακόμα και μια βαθμονομημένη τιμή δεν μπορούμε να εξασφαλίσουμε την ακρίβειά της, εάν πρωτίστως δεν έχουμε πραγματοποιήσει σωστή μηδενική αντιστάθμιση. Η μηδενική αντιστάθμιση είναι η διαδικασία με την οποία εξασφαλίζουμε τη σωστή (μηδενική) ένδειξη σε ένα μετρητικό όργανο, όταν είναι μηδενικό σε μέγεθος το μετρούμενο φαινόμενο. Είναι ουσιαστικά ο καθορισμός μιας ορθής βάσης για τη μετέπειτα σωστή βαθμονόμηση του οργάνου.

Κατά την εκκίνηση του αισθητήρα δύναμης/ροπής που χρησιμοποιούμε, πραγματοποιείται κάθε φορά μηδενική αντιστάθμιση για τη δεδομένη κατάστασή του την ακριβή στιγμή εκκίνησης. Με λίγα λόγια, εάν έχουμε προσθέσει ένα εξάρτημα με δεδομένο βάρος στην άκρη του αισθητήρα και τον εκκινήσουμε, οι τιμές που θα παίρνουμε θα είναι εξίσου μηδενικές, όπως και αν τον εκκινούσαμε χωρίς το πρόσθετο αυτό βάρος. Για την ίδια αιτία αναιρείται το διάνυσμα βάρους του ίδιου του αισθητήρα κάθε φορά σε σχέση με την κατεύθυνσή του κάθε φορά, ανάλογα με τον προσανατολισμό του αισθητήρα τη στιγμή εκκίνησης των μετρήσεων.

Μπορεί σε πρώτο επίπεδο η αυτόματη αυτή διαδικασία του αισθητήρα να παρουσιάζεται αρκετά βοηθητική, αλλά ουσιαστικά μας προκαλεί δύο προβλήματα, η σημασία των οποίων εξαρτάται από την εκάστοτε εφαρμογή, και γι' αυτό πρέπει αυτά να επιλυθούν για την χρηστικότητα της μεθοδολογίας που αναπτύσσουμε στην παρούσα εργασία, στο σύνολο των πιθανών εφαρμογών. Τα προβλήματα που παρουσιάζονται είναι τα εξής:

- *Απρόβλεπτη επίδραση βάρους αισθητήρα και πιθανού επιπλέον εξαρτήματος στις τιμές των φορτίσεων κατά την περιστροφή του συστήματος στο χώρο*:
- *Μη ταύτιση των καμπύλων βαθμονόμησης με τις αναμενόμενες τιμές, όταν ο αισθητήρας έχει διαφορετική μηδενική αντιστάθμιση:*



Σχήμα 16. Περιστροφή διανύσματος βάρους στο χώρο.

Ας θεωρήσουμε, λοιπόν, ότι το σύστημα συντεταγμένων έχει τη μορφή του Σχήματος 16 και το βάρος της διάταξης είναι δύναμη παράλληλη με τον y άξονα με αρνητική φορά (F_0). Κατά την εκκίνηση του αισθητήρα όλες οι τιμές προσαρμόζονται στο 0 (μηδέν), επομένως στον άξονα y έχουμε αγνοήσει μια υπάρχουσα δύναμη $F_{0y}= -F_0$. Αντίστοιχα, η δύναμη που αγνοούμε κατά την εκκίνηση μπορεί να αναλύεται σε περισσότερους άξονες, αλλά χάριν ευκολίας επιδιώκουμε την απλούστευση του προβλήματος.

Σε μια τυχαία κίνηση και αλλαγή του προσανατολισμού του αισθητήρα στο χώρο, το βάρος πλέον της διάταξης έχει διαφορετικές συνιστώσες σε σχέση με την αρχική κατάσταση. Θεωρώντας ότι έχουμε κατά αναλογία περιστροφή του διανύσματος του βάρους σε σχέση με σταθερό σύστημα συντεταγμένων (ενώ στην ουσία πραγματοποιείται το αντίθετο), έχουμε το αντίστοιχο διάνυσμα F_1 που αναλύεται σε F_{1x}, F_{1y}, F_{1z} και ουσιαστικά εμφανίζει μη μηδενικές τιμές δυνάμεων στο σύστημά μας. Στον άξονα y, παράλληλα του οποίου βρισκόταν το αρχικό διάνυσμα βάρους, στην τιμή που εμφανίζεται, πρέπει να συνυπολογίσουμε και τον παράγοντα $F_{0y} =-F_0$ που αγνόησε το σύστημά μας κατά την αρχική μηδενική αντιστάθμιση.

Για να επιτύχουμε, λοιπόν, σωστή μηδενική αντιστάθμιση, πρέπει κάθε δεδομένη στιγμή για κάθε έναν από τους τρεις άξονες να μπορούμε να γνωρίζουμε και να αφαιρούμε τη συνιστώσα του βάρους και να προσθέτουμε την αντίστοιχη συνιστώσα που αγνοήθηκε κατά την εκκίνηση, δηλαδή:

$$\left\{ \begin{array}{l} Fx_{offset} = Fx_{\acute{\varepsilon}\nu\delta\varepsilon\iota\xi\eta\varsigma} + F_{x0} - F_{x1} \\ Fy_{offset} = Fy_{\acute{\varepsilon}\nu\delta\varepsilon\iota\xi\eta\varsigma} + F_{y0} - F_{y1} \\ Fz_{offset} = Fz_{\acute{\varepsilon}\nu\delta\varepsilon\iota\xi\eta\varsigma} + F_{z0} - F_{z1} \end{array} \right\}$$

Το βάρος της διάταξης κατά την εκκίνηση του αισθητήρα πρέπει να είναι υπολογισμένο από πριν και να αναλύεται στο βάρος του ίδιου του αισθητήρα, που είναι σταθερό, και στο βάρος του εξαρτήματος που έχει προσδέσει ο χρήστης σε αυτόν.

Ο υπολογισμός, όμως, των συνιστωσών του βάρους σε δεδομένη χρονική στιγμή και περιστροφή απαιτεί, εκτός από το μέτρο του διανύσματος που παραμένει σταθερό, και τον προσανατολισμό του διανύσματος σε σχέση με το καρτεσιανό μας σύστημα συντεταγμένων. Γνωρίζοντας τον προσανατολισμό του αισθητήρα μέσω των γωνιών Euler (ZYX), ο υπολογισμός των οποίων γίνεται εύκολα με βοήθεια της ορθής κινηματικής του ρομποτικού βραχίονα, με δεδομένες γωνίες αρθρώσεων, μπορούμε να υπολογίσουμε πώς αναλύεται το βάρος στους τρεις άξονες.

Έστω οι γωνίες α, β και γ που εκφράζουν την περιστροφή ως προς τον x, τον y και τον z άξονα αντίστοιχα. Το μητρώο 3x3 που εκφράζει την περιστροφή κατά Euler του αρχικού διανύσματος με βάση αυτές τις γωνίες πολλαπλασιάζεται με τις αρχικές συντεταγμένες του F_0 διανύσματος του βάρους του αισθητήρα (και του πιθανού επιπλέον εξαρτήματος) και παράγει τις συντεταγμένες F_{1x}, F_{1y}, F_{1z} του διανύσματος του βάρους στην αντίστοιχη θέση 1. Οι συντεταγμένες αυτές εκφράζουν τους όρους που πρέπει τη δεδομένη στιγμή να αφαιρεθούν από την ένδειξη του αισθητήρα για τους αντίστοιχους άξονες, σύμφωνα με το ανωτέρω σύστημα εξισώσεων. Η μαθηματική απεικόνιση της παραπάνω διεργασίας για τον υπολογισμό των F_{1x}, F_{1y}, F_{1z} είναι η εξής:

$$\begin{bmatrix} \cos\beta\cdot\cos\gamma & -\cos\beta\cdot\sin\gamma\cdot\cos\alpha+\sin\beta\cdot\sin\alpha & \cos\beta\cdot\sin\gamma\cdot\sin\alpha+\sin\beta\cdot\cos\alpha \\ \sin\gamma & \cos\gamma\cdot\cos\alpha & -\cos\gamma\cdot\sin\alpha \\ -\sin\beta\cdot\cos\gamma & \sin\beta\cdot\sin\gamma\cdot\cos\alpha+\cos\beta\cdot\sin\alpha & -\sin\beta\cdot\sin\gamma\cdot\sin\alpha+\cos\beta\cdot\cos\alpha \end{bmatrix}\cdot\begin{bmatrix} F_{0x} \\ F_{0y} \\ F_{0z} \end{bmatrix}=\begin{bmatrix} F_{1x} \\ F_{2x} \\ F_{3x} \end{bmatrix}$$

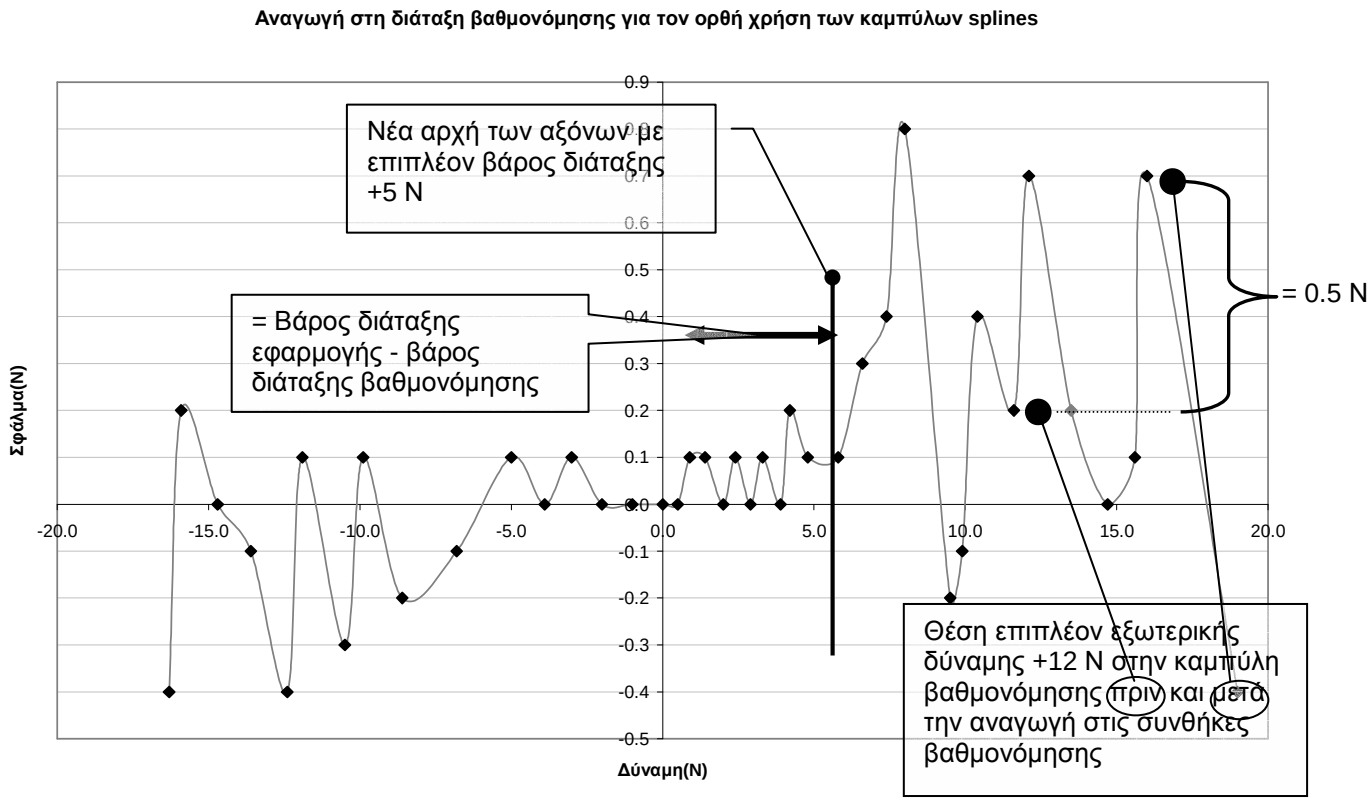
όπου επαναλαμβάνουμε πως για τη ευκολότερη αντιμετώπιση του ζητήματος θέτουμε αρχική θέση τέτοια, ώστε δύο εκ των συνιστωσών F_{0x}, F_{0y}, F_{0z} να είναι μηδενικές, ώστε η τρίτη να ισούται με το εκ των προτέρων υπολογισμένο F_0. (π.χ $F_{0x} = F_{0z} = 0$, ώστε $F_{0y}= F_0$ όπως εφαρμόσαμε εμείς στην πράξη)

Για την ορθή χρήση των καμπύλων βαθμονόμησης σε κάθε πιθανή διάταξη χρήσης του αισθητήρα πρέπει να γίνεται αναγωγή των μετρήσεων στη διάταξη βαθμονόμησης, να υπολογιστούν οι ενδείξεις με βάση το μοντέλο βαθμονόμησης που αναλύθηκε στα προηγούμενα κεφάλαια, και ύστερα να μετασχηματιστούν πλέον με βάση τη νέα διάταξη στην οποία παίρνουμε μετρήσεις.

Το βάρος του ίδιου του αισθητήρα υπολογίστηκε με ακρίβεια μιας τάξης μεγέθους μεγαλύτερη από αυτή του αισθητήρα, στα 0.51 N. Όπως περιγράφεται και στο αντίστοιχο κεφάλαιο ανάλυσης του κώδικα της βαθμονόμησης, η τιμή αυτή αποθηκεύεται σε μια σταθερά και προστίθεται με την αντίστοιχη σταθερά του επιπλέον βάρους που ασκείται μέσω

κάποιου μόνιμα προσαρμοσμένου εξαρτήματος. Στην περίπτωση της διάταξης βαθμονόμησης είχε προστεθεί μια επιπλέον φλάντζα και τέσσερις βίδες, το συνολικό δε βάρος όλης της διάταξης ήταν 1.38 N. Αυτή η τιμή ορίζεται ως η αντιστάθμιση βαθμονόμησης.

Έτσι, πριν την εφαρμογή των καμπύλων βαθμονόμησης, έχοντας υπολογίσει το συνολικό βάρος της μόνιμης διάταξης της εφαρμογής μας (βάρος αισθητήρα (0.51N) + βάρος επιπλέον εξαρτημάτων) προσθέτουμε προσωρινά στην ένδειξη της δύναμης τη διαφορά του βάρους της διάταξης της εφαρμογής με την τιμή της αντιστάθμισης βαθμονόμησης. Αυτή η διαφορά εκφράζει το επιπλέον (ή λιγότερο) βάρος της διάταξης που χρησιμοποιούμε στην εφαρμογή μας σε σχέση με τη διάταξη βαθμονόμησης και, λόγω της αρχικής μηδενικής αντιστάθμισης της συσκευής, έχει αγνοηθεί από τους υπολογισμούς μας. Στο τέλος της βαθμονόμησης αυτό το ποσό το αφαιρούμε ξανά, για να επανέλθουμε στην τρέχουσα κατάσταση της εφαρμογής. Για την πληρέστερη κατανόηση της διαδικασίας και το λόγο της πραγματοποίησής της παραθέτουμε το παρακάτω Διάγραμμα 17.



Διάγραμμα 17. Αναγωγή εξωτερικής δύναμης στη διάταξη βαθμονόμησης για τον ορθή χρήση των καμπύλων splines

Όπως παρατηρούμε, εάν είχαμε αγνοήσει πλήρως το επιπλέον βάρος της διάταξης (έστω +5 N) σε σχέση με τις συνθήκες βαθμονόμησης, μία εξωτερική δύναμη παράλληλα με τον άξονα που μελετάμε με ένδειξη της τάξεως των +12 N θα διορθωνόταν κατά 0.2 N με βάση το υπάρχον μοντέλο βαθμονόμησης. Όμως η πραγματική θέση της δεδομένης

κατάστασης στην καμπύλη βαθμονόμησης είναι η θέση 5+12=17 N, συπολογίζοντας και την επιπλέον επίδραση του βάρους της διάταξης. Σε αυτή τη θέση, η διόρθωση της τιμής είναι της τάξεως των 0.7 N. Άμεσα παρατηρούμε πως η παράλειψη αυτής της διαδικασίας στο συγκεκριμένο παράδειγμα μας προκαλεί ένα σημαντικό σφάλμα στην ένδειξή μας που φτάνει τα 0.5 N.

ΚΕΦΑΛΑΙΟ 4ο : Ολοκλήρωση και Πειράματα

Εισαγωγή:

Βασικός σκοπός της παρούσας διατριβής είναι η πειραματική εξέταση μεθοδολογιών ελέγχου δύναμης/ροπής με τη χρήση του διαθέσιμου ρομποτικού βραχίονα PA-10 επτά (7) βαθμών ελευθερίας. Η αναλυτική περιγραφή των βημάτων προσαρμογής, χρήσης και λειτουργίας του διαθέσιμου εξοπλισμού, καθώς και η εκτενής θεωρητική ανάλυση των προτεινόμενων μεθοδολογιών ελέγχου για τη δεδομένη εφαρμογή που προηγήθηκε, είχε σαν σκοπό τη βαθύτερη κατανόηση των παραμέτρων της διαδικασίας που ακολουθήθηκε και των δυσκολιών που μπορούν να προκύψουν σε αντίστοιχες περιστάσεις.

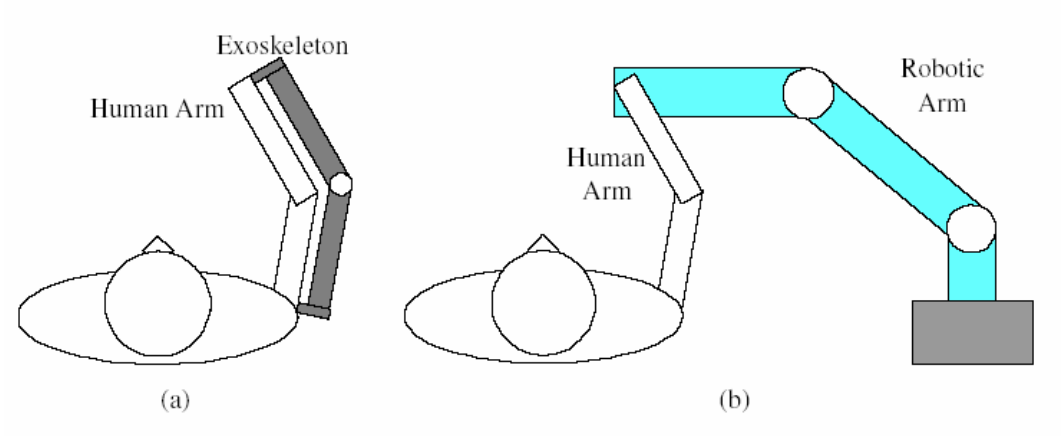
Στο ίδιο πνεύμα, το κεφάλαιο που ακολουθεί έχει σαν κύριο στόχο, όχι μόνο να περιγράψει τις συνθήκες και τα αποτελέσματα πειραμάτων ελέγχου δύναμης, αλλά και να αναλύσει πλήρως τη διαδικασία για την προετοιμασία και τη σωστή διεξαγωγή τους, έτσι ώστε να μπορούν να επαναληφθούν στο μέλλον τα ίδια ή και πολυπλοκότερα πειράματα.

4.1 Περιγραφή Πειράματος

Σκοπός του πειράματος που ακολουθεί, είναι η εφαρμογή οδήγησης του ακροδέκτη ενός ρομποτικού βραχίονα στον οποίο είναι προσαρμοσμένο ένα ανθρώπινο χέρι, με σκοπό να ακολουθεί την πρόθεση κίνησης του ανθρώπου αυτού. Το σενάριο αυτό είναι εμπνευσμένο από την Νευρορομποτική επιστήμη και έχει σαν σκοπό την ανάπτυξη συστημάτων για την μηχανική αποκατάσταση της λειτουργικότητας μελών τραυματισμένων ατόμων και ατόμων με ειδικές ανάγκες, όπως και για την ενίσχυση της δύναμης ενός ατόμου, όχι απαραίτητα με κάποια κινητική δυσλειτουργία, για την πραγματοποίηση μιας δύσκολης εργασίας.

Μια τυπική εφαρμογή είναι η εξωτερική σύνδεση μιας διάταξης με το άνω άκρο ενός ανθρώπου για την υποστήριξή του και τη βελτίωση της λειτουργίας του. Τέτοια τυπική διάταξη μπορεί να είναι ένας "έξυπνος" εξωσκελετός. Η χρήση ενός τέτοιου εξωσκελετού συνδεδεμένου με ένα ανθρώπινο χέρι απαιτεί το σχεδιασμό ενός ελέγχου, σύμφωνα με τον οποίο η πρόθεση για κίνηση από τον άνθρωπο ανιχνεύεται από το κατάλληλο μηχανοτρονικό σύστημα και οι κινητήρες του εξωσκελετικού ρομποτικού βραχίονα εφαρμόζουν την κατάλληλη ροπή, ώστε ο εξωσκελετός να υποβοηθά ή να ενισχύει την κίνηση του χεριού.

Ένα θεμελιώδες ζήτημα στην επιστήμη της Νευρορομποτικής, που σχετίζεται με αυτό το πρόβλημα, είναι ο προσδιορισμός των μεταβλητών ελέγχου που χρησιμοποιεί το κεντρικό νευρικό σύστημα για τον έλεγχο της ενεργοποίησης του μυϊκού συστήματος. Στη συγκεκριμένη εφαρμογή γίνεται προσπάθεια να ελέγξουμε το σύστημα ανθρώπινου και τεχνητού χεριού με πληροφορίες που λαμβάνουμε από έναν αισθητήρα δύναμης/ροπής προσαρμοσμένο ενδιάμεσά τους και βασίζεται σε αναλύσεις για το πώς ένας άνθρωπος αντιλαμβάνεται και αναπαράγει δυνάμεις.



Σχήμα 17. Σχηματική αναπαράσταση διάταξης εξωσκελετού (a) Σε μια πραγματική εφαρμογή (b) Στο πειραματικό ανάλογο.

Η διάταξη που εξετάζουμε φαίνεται στο Σχήμα 17 (a) , όπου για λόγους απλοποίησης μπορούμε να μελετήσουμε τη διάταξη στο Σχήμα 17 (b) που έχει ανάλογη συμπεριφορά. Μεταβάλλοντας έναν συντελεστή ενίσχυσης α μπορούμε να προσδιορίσουμε τη δύναμη που θα ασκηθεί από το τεχνητό μέλος στο περιβάλλον, με την ίδια φορά που ασκείται η δύναμη από το χρήστη, μέσω εφαρμογής των κατάλληλων ροπών στους κινητήρες του. Μηδενικός συντελεστής έχει ως αποτέλεσμα την εφαρμογή ίδιας δύναμης από το τεχνητό μέλος, σύμφωνα με την τιμή που λαμβάνει από τον αισθητήρα.

4.2 Προετοιμασία πειραματικής διάταξης

4.21 Θεωρητική επισκόπηση

Παρακάτω αναλύεται εν συντομία το θεωρητικό μοντέλο που χρησιμοποιήθηκε στη συγκεκριμένη εφαρμογή με βάση τις αναλύσεις μεθοδολογιών ελέγχου που παρατίθονται στο Κεφάλαιο 2. Η δομή της ανάλυσης βασίζεται στην ακολουθία των βημάτων που πραγματοποιούνται στο λογισμικό ελέγχου.

Το σύστημά μας μπορεί να περιγραφεί από το εξής δυναμικό μοντέλο δευτέρας τάξεως:

$$A(q)\cdot\ddot{q}+C(q,\dot{q})\cdot\dot{q}+G(q)+Fr(\dot{q})=\tau-J^{T}\cdot F \quad (4.1)$$

Χρησιμοποιώντας νόμο ελέγχου αντίστροφης δυναμικής και μετρήσεις των δυνάμεων μέσω του αισθητήρα, για τον υπολογισμό των ροπών που θέλουμε να στείλουμε στις αρθρώσεις του ρομποτικού βραχίονα καταλήγουμε στην εξής σχέση:

$$\tau=A(q)\cdot y+C(q,\dot{q})\cdot\dot{q}+G(q)+Fr(\dot{q})+J^{T}\cdot F \quad (4.2)$$

όπου

- A το μητρώο της αδράνειας
- C το μητρώο των φυγόκεντρων ροπών και ροπών Coriolis
- G το διάνυσμα των βαρυτικών ροπών
- Fr το διάνυσμα των τριβών στις αρθρώσεις (ενσωματώνει και τη δυσκαμψία)
- F το διάνυσμα των εξωτερικών δυνάμεων και ροπών

Αντικαθιστώντας τη σχέση (4.2) στη σχέση (4.1) προκύπτει:

$$\ddot{q} = y \qquad (4.3)$$

που περιγράφει τη γραμμικοποιημένη εξίσωση του συστήματος.

Κατασκευάζουμε το νόμο ελέγχου για το σύστημα που περιγράψαμε παραπάνω:

$$y = J^{-1}M_d^{-1}(-K_d\dot{x}_d - K_p x_d - M_d \cdot \dot{J} \cdot \dot{q}_d + u) \qquad (4.4)$$

όπου

$$u = (1+a)F \qquad (4.5)$$

με F το διάνυσμα των δυνάμεων/ροπών που λαμβάνουμε από τον αισθητήρα και α το συντελεστή ενίσχυσης της εξωτερικής δύναμης.

Μέσω της ορθής κινηματικής του βραχίονα έχουμε:

$$\dot{x} = J_A(q) \cdot \dot{q} \qquad (4.6)$$

$$\ddot{x} = J_A(q) \cdot \ddot{q} + \dot{J}_A(q) \cdot \dot{q} \qquad (4.7)$$

Αντικαθιστώντας τις σχέσεις (4.4), (4.7) στην εξίσωση (4.3) προκύπτει η ακόλουθη σχέση:

$$M_d\ddot{x}_d + K_d\dot{x} + K_p x = u \qquad (4.8)$$

που είναι η εξίσωση του συστήματος στο λειτουργικό χώρο.

Εκφράζοντας τη σχέση (4.1) ως προς 2 βαθμούς ελευθερίας έχουμε να κάνουμε τις εξής παρατηρήσεις:

Μπορούμε χωρίς σημαντική επίδραση στο αποτέλεσμα να αγνοήσουμε το μητρώο της αδράνειας A και το μητρώο των φυγόκεντρων ροπών C. Εφόσον το πείραμα γίνει σε επίπεδο κάθετο προς αυτό της βαρύτητας αγνοούμε πλήρως και το μητρώο των βαρυτικών ροπών G.

Έτσι η εξίσωση (4.1) καταλήγει ως:

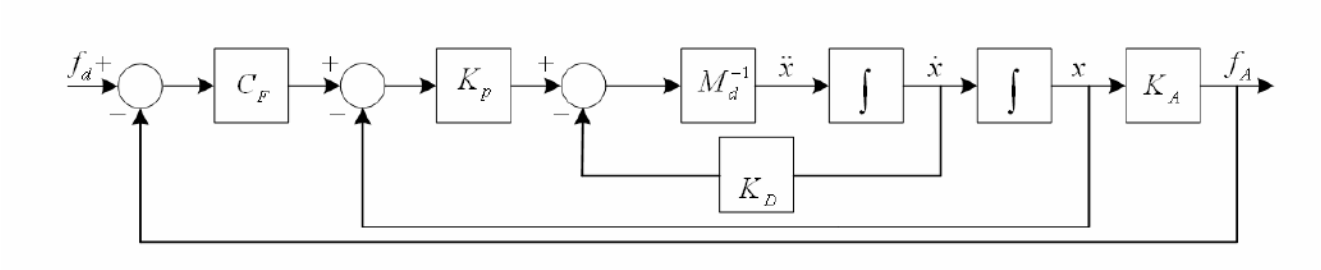
$$I \cdot \ddot{q} + Fr(\dot{q}) = \tau - J^T \cdot F \qquad (4.9)$$

και χρησιμοποιώντας νόμο ελέγχου αντίστροφης δυναμικής και μετρήσεις των δυνάμεων μέσω του αισθητήρα, για τον υπολογισμό των ροπών που θέλουμε να στείλουμε στις αρθρώσεις του ρομποτικού βραχίονα καταλήγουμε στην εξής σχέση:

$$\tau = I \cdot y + Fr(\dot{q}) + J^T \cdot F \qquad (4.10)$$

όπου I ο μοναδιαίος πίνακας και y ο νόμος ελέγχου (4.4) που θα χρησιμοποιήσουμε αντίστοιχα και στο σύστημα με δύο βαθμούς ελευθερίας.

Το σχεδιάγραμμα ελέγχου που εκφράζει συνοπτικά το παραπάνω μαθηματικό μοντέλο είναι το εξής:



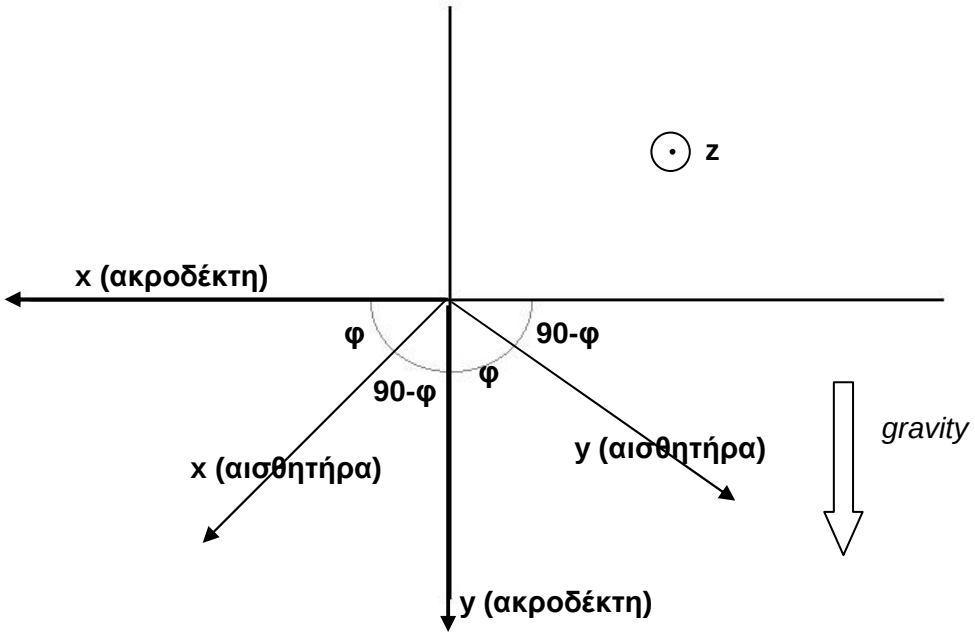
Σχήμα 18. Σχεδιάγραμμα Ελέγχου της πειραματικής εφαρμογής: Έλεγχος δύναμης με εσωτερικό βρόχο Θέσης.

4.22 Αναγωγή δυνάμεων στο τελικό σημείο δράσης

Κατασκευαστικά η φλάντζα σύνδεσης του αισθητήρα με τον ακροδέκτη του ρομποτικού βραχίονα, όπως αναλύθηκε στο προηγούμενο κεφάλαιο, προσφέρει μια ισχυρή σύνδεση των εξαρτημάτων αυτών με κοχλιώσεις και περικόχλια, που προσφέρουν μια ασφαλή λειτουργία σε συνθήκες πειραμάτων.

Όμως, κάποιες κατασκευαστικές διαμορφώσεις και βελτιώσεις είχαν ως αποτέλεσμα την απόκλιση του συστήματος συντεταγμένων του ακροδέκτη του ρομποτικού βραχίονα, σε σχέση με τους άξονες μέτρησης δυνάμεων και ροπών του αισθητήρα. Για να μπορεί να αντιληφθεί το δυναμικό μας μοντέλο, και εν τέλει το σύστημα ελέγχου, τις ακριβείς συνιστώσες των δυνάμεων που εφαρμόζονται από το περιβάλλον μέσω του αισθητήρα, είναι απαραίτητο να υπολογιστεί αυτή η γεωμετρική απόκλιση, ώστε με τον κατάλληλο μετασχηματισμό να ανάγουμε τις δυνάμεις που μετρούμε από τον αισθητήρα στο τελικό σημείο δράσης.

Τα πλαίσια συντεταγμένων του αισθητήρα και του ακροδέκτη του ρομποτικού βραχίονα συμπίπτουν ως προς τον άξονα z και αυτό που έπρεπε να προσδιοριστεί ήταν η γωνιακή απόκλιση περιστροφής ως προς z (γωνία φ) των αξόνων x, y, όπως περιγράφεται παραστατικά στο παρακάτω σχήμα:



Σχήμα 19. Υπολογισμός γωνίας απόκλισης πλαισίου ακροδέκτη βραχίονα και αισθητήρα.

Για να υπολογιστεί αυτή η γωνία με ακρίβεια, χρησιμοποιήσαμε τον υπάρχοντα εξοπλισμό με τον εξής τρόπο:

Τοποθετήσαμε το ρομποτικό βραχίονα στη μηδενική θέση, με μηδενικές γωνίες δηλαδή σε κάθε άρθρωση, ώστε να επιτύχουμε το καρτεσιανό σύστημα συντεταγμένων του ακροδέκτη, που φαίνεται στο Σχήμα 17, με τον άξονα y ταυτόσημο με τον άξονα της βαρυτικής δύναμης και τον άξονα x κάθετο προς αυτή.

Βάζοντας διαδοχικά βάρη στον αισθητήρα μπορέσαμε να δούμε πώς αναλύεται η μετρούμενη δύναμη ως προς τους x και y του αισθητήρα. Γεωμετρικά η συνιστώσα στο x άξονα εκφράζει ως προς τη γωνία που αναζητούμε το $Fx = Fcos (90-φ) = Fsin (φ)$, όπου F το μέτρο της βαρυτικής δύναμης που εφαρμόζεται πάνω στον άξονα y του συστήματος συντεταγμένων του ακροδέκτη του ρομποτικού βραχίονα.

Αντίστοιχα η συνιστώσα στον άξονα y εκφράζει ως προς τη γωνία που αναζητούμε το $Fy = Fcos (φ)$. Για να απαλείψουμε το μέτρο της δύναμης F, διαφορετικής σε κάθε μέτρηση, παίρνουμε το λόγο $Fx / Fy = tan (φ)$. Έτσι, μπορούμε να υπολογίσουμε τη $φ = atan (Fx / Fy)$ σε κάθε μας μέτρηση, που, με δεδομένη την ακρίβεια των μετρήσεων, μπορεί να μας δώσει ικανοποιητική ακρίβεια στον υπολογισμό αυτής της γωνίας. Να σημειώσουμε ότι είναι προτιμότερη η εφαρμογή σχετικά μεγάλων βαρών για τον υπολογισμό των συνιστωσών Fx και Fy, ώστε να επηρεάζεται το αποτέλεσμα, όσο το δυνατόν λιγότερο από την ακρίβεια της μέτρησης των δυνάμεων.

Έπειτα από μια σειρά μετρήσεων προκύπτουν τα εξής αποτελέσματα, που προσδιορίζουν τη γωνία απόκλισης φ του συστήματος συντεταγμένων του ακροδέκτη του ρομποτικού βραχίονα ως προς το σύστημα συντεταγμένων των μετρήσεων του αισθητήρα.

Fx (N)	Fy(N)	Fx/Fy	
7.2	6.9	1.04	
9.8	9.3	1.05	Μέση τιμή
12.3	11.6	1.06	1.05
13.7	13.1	1.05	Φ=atan(1.05)
14.8	14	1.06	**46.45^0**

Πίνακας 6. Πειραματικός υπολογισμός γωνίας απόκλισης του πλαισίου συντεταγμένων αισθητήρα σε σχέση με τον ακροδέχτη του βραχίονα.

Έτσι, στον κώδικα ελέγχου πρέπει να προστεθεί ένα μπλοκ εντολών που θα ανάγει τις δυνάμεις που προέρχονται από τον αισθητήρα στο σύστημα συντεταγμένων του τελικού σημείου δράσης. Υπενθυμίζεται πως οι νέες τιμές προκύπτουν έπειτα από περιστροφή γύρω από τον z άξονα κατά γωνία φ.

$$\begin{bmatrix} Fx' \\ Fy' \\ Fz' \end{bmatrix} = \begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} Fx \\ Fy \\ Fz \end{bmatrix}$$

4.23 Λογισμικό εφαρμογής ελέγχου δύναμης

Για να πραγματοποιηθούν τα πειράματα ελέγχου δύναμης/ροπής και να επιτύχουμε τη σωστή ανάδραση στο σύστημα από τις εξωτερικές δυνάμεις και ροπές, που λαμβάνουμε μέσω του αισθητήρα, έπρεπε να αναπτυχθεί το ανάλογο λογισμικό.

Η ρουτίνα που αναπτύχθηκε για την επεξεργασία, βαθμονόμηση και μηδενική αντιστάθμιση των τιμών δύναμης/ροπής του αισθητήρα είναι η JR3_force.c, που περιγράφεται αναλυτικά στο Παράρτημα Β 2.1.

Η εφαρμογή του νόμου ελέγχου προγραμματίστηκε και για όλον το βραχίονα (επτά βαθμούς ελευθερίας) στο ForceControl.cpp, αλλά και για δύο βαθμούς ελευθερίας (τις αρθρώσεις 2 και 4 του βραχίονα) με τους υπόλοιπους πέντε βαθμούς κλειδωμένους στο Fcontrol2dof.cpp, που αναλύονται διεξοδικά στο Παράρτημα Β 2.2 και 2.3 αντίστοιχα.

Για την επιτυχή λειτουργία του κεντρικού κώδικα και στις δύο περιπτώσεις ήταν απαραίτητη η σύνδεσή του με υπορουτίνες, που υπολογίζουν το δυναμικό μοντέλο του συστήματος, την ορθή κινηματική, την αντίστροφη κινηματική και την Ιακωβιανή του βραχίονα, βοηθητικές ρουτίνες υπολογισμού πράξεων μεταξύ μητρώων, βιβλιοθήκες του συστήματος για την επικοινωνία με τον ελεγκτή του ρομπότ, όπως και φυσικά τη ρουτίνα υπολογισμού των τιμών δύναμης και ροπής από τον αισθητήρα.

Η συγκεντρωτική δομή του κεντρικού κώδικα και τον βοηθητικών υπορουτινών περιγράφεται στο αντίστοιχο κεφάλαιο του Παραρτήματος Β.1. Η αναλυτική χρήση και λειτουργία των επιμέρους υπορουτινών περιγράφεται στο Παράρτημα Γ.1.

4.24 Χρονομέτρηση κύκλου λειτουργίας ελέγχου δύναμης

Με σκοπό το σωστό συγχρονισμό της ρουτίνας δυναμικού ελέγχου, που δημιουργήθηκε με τη διαδικασία αποστολής και λήψης πληροφοριών στον ελεγκτή του ρομπότ και που πραγματοποιείται μέσω των συναρτήσεων υψηλού επιπέδου της βιβλιοθήκης libpa10, είναι απαραίτητο να γνωρίζουμε με ακρίβεια το χρόνο που απαιτείται για να υπολογιστούν όλα τα απαραίτητα δεδομένα σε ένα κύκλο λειτουργίας.

Για την επίτευξη αυτού του σκοπού έγινε χρήση της εντολής `gettimeofday`. Ένα απλό πρόγραμμα με το οποίο μπορεί να γίνει επιτυχώς μία τέτοια χρονομέτρηση διαρθρώνεται παρακάτω:

```
#include
.
.
int main ()
{
  struct timeval start, end;
  .
  .
  .
  gettimeofday(&start, NULL);
    printf("Start time: %ld.%06ld\n", start.tv_sec,
  start.tv_usec);

    {

      " Διαδικασία προς χρονομέτρηση "

    }

  gettimeofday(&end, NULL);
  printf("End time: %ld.%06ld\n", end.tv_sec, end.tv_usec);

  end.tv_sec -= start.tv_sec;
  if (end.tv_usec < start.tv_usec) {
     end.tv_sec--;
     end.tv_usec+= 1000000;
     }
  end.tv_usec -= start.tv_usec;
  printf("time: %ld.%06ld\n", end.tv_sec, end.tv_usec);

}
```

Η λεπτομερής ανάλυση χρήσης και λειτουργίας κάθε εντολής δεν είναι τόσο απαραίτητη, όσο η δομή του συνολικού προγράμματος. Με τις κουκκίδες υπονοούνται άλλες εντολές του κυρίως προγράμματος που δεν υπόκεινται σε χρονομέτρηση. Με έντονα γράμματα εκφράζεται το μπλοκ των εντολών, ο χρόνος υπολογισμού των οποίων θέλουμε να χρονομετρηθεί.

Για μεγαλύτερη ακρίβεια ως προς το αποτέλεσμα χρονομέτρησης, κυρίως για διεργασίες με πολύ μικρό κύκλο λειτουργίας, είναι προτιμότερο να επαναλαμβάνεται περισσότερες φορές της μίας ο κύκλος λειτουργίας, έτσι ώστε ο χρονομετρητής να μας επιστρέφει χρόνο πολλαπλάσιο του κύκλου λειτουργίας. Ο χρόνος αυτός, διαιρεμένος με τον αριθμό επαναλήψεων, έχει ως αποτέλεσμα το "μέσο" χρόνο διάρκειας ενός κύκλου με μεγαλύτερη ακρίβεια, σε σχέση με τη χρονομέτρηση ενός μόνο κύκλου. Ουσιαστικά στον παραπάνω κώδικα πρέπει να αντικαταστήσουμε το προς χρονομέτρηση μπλοκ εντολών με το εξής:

```
for ( int repeat = 0; repeat < 10000; repeat++ )  {
    {
       " Διαδικασία προς χρονομέτρηση "
    }
}
```

Στη συγκεκριμένη περίπτωση απαιτούμε χίλιες επαναλήψεις της διαδικασίας, άρα θα διαιρέσουμε το συνολικό χρόνο που μας επιστρέφει ο χρονομετρητής με το χίλια, για τον εντοπισμό της διάρκειας υπολογισμού ενός κύκλου της διαδικασίας.

Για τον υπολογισμό του χρόνου υπολογισμού ενός κύκλου διεργασίας του πλήρους δυναμικού μοντέλου και ελέγχου και των επτά βαθμών ελευθερίας, που ελέγχει ο κώδικας ForceControl.cpp, έγιναν πολλές διαφορετικές μετρήσεις για να εξασφαλιστεί η ακρίβεια του αποτελέσματος. Οι μετρήσεις έγιναν έχοντας προσαρμόσει τον κώδικα σε κατάσταση προσομοίωσης και ουσιαστικά αγνοούνται μόνο οι συναρτήσεις υψηλού επιπέδου που επικοινωνούν με τη libpa10.

Ο χρόνος κυμαινόταν εντός κάποιων στενών ορίων για δύο διαφορετικές καταστάσεις που έγιναν μετρήσεις.

α) Ακίνητος βραχίονας: Στο σύνολο των υπολογισμών ουσιαστικά είχαμε πολλές συνιστώσες μηδενικές σε πράξεις μεταξύ πινάκων και ο βρόχος της αντίστροφης κινηματικής δεν υπολογιζόταν. Ουσιαστικά εκφράζει τον ελάχιστο χρόνο στον οποίο μπορεί να πραγματοποιηθεί ένας κύκλος υπολογισμών σε κανονική λειτουργία, αλλά σε κύκλο που δεν έχουμε κίνηση του βραχίονα

β) Βραχίονας σε πλήρη κίνηση: Όλοι οι υπολογισμοί είναι πλήρεις για όλες τις αρθρώσεις και υπάρχει είσοδος στο σύστημα και τιμών δυνάμεων και ροπών. Ο βρόχος της αντίστροφης κινηματικής υπολογίζεται με τη μέγιστη δυνατή ακρίβεια και όλες οι πράξεις

ψευδοολοκλήρωσης γίνονται αναφερόμενες σε χρόνο dt ίσο με τον κύκλο λειτουργίας. Ουσιαστικά εκφράζει το μέγιστο χρόνο στον οποίο μπορεί να πραγματοποιηθεί ένας κύκλος υπολογισμών σε κανονική λειτουργία.

Έγιναν μετρήσεις για 1.000, 10.000, 20.000, 30.000 και 50.000 επαναλήψεις για τις δύο καταστάσεις που περιγράφτηκαν με τα εξής αποτελέσματα:

Κατάσταση	**Ακίνητος βραχίονας** (minimum time)	**Βραχίονας σε πλήρη κίνηση** (maximum time)
Χρόνος (ms)	1,33	3.67

Πίνακας 7. Αποτελέσματα χρονομέτρησης κύκλου λειτουργίας λογισμικού ελέγχου ForceControl.cpp.

Οι μετρήσεις έγιναν σε τυπικό λειτουργικό περιβάλλον Linux, αντίστοιχο του λειτουργικού συστήματος, πάνω στο οποίο θα πραγματοποιηθεί η εφαρμογή του νόμου ελέγχου σε πραγματικές συνθήκες.

Σκοπός μας, όμως, είναι σε καμία περίπτωση να μην ξεπεράσει ο χρόνος υπολογισμού ενός κύκλου λειτουργίας του νόμου ελέγχου την περίοδο επικοινωνίας, που καθορίζεται μέσω των συναρτήσεων υψηλού επιπέδου της βιβλιοθήκης libpa10 από το σύστημά μας. Σε αυτή την περίπτωση θα χαθεί ο συγχρονισμός και η επικοινωνία με τον ελεγκτή του ρομπότ, έστω και στιγμιαία με απρόβλεπτες συνέπειες για την εφαρμογή μας.

Έτσι επιλέγουμε κύκλο επικοινωνίας στα **5ms**, χρόνο διπλάσιο της επικοινωνίας με τον ελεγκτή του ρομπότ (είναι στα 2.5ms). Αυτό σημαίνει ότι η αντίστοιχη συνάρτηση υψηλού επιπέδου καθορίζεται ως:

```
pa10_init(5000000);
```

καθώς ο χρόνος επικοινωνίας καθορίζεται σε nanosecond (ns) = 10^{-9} second = 10^{-6} milisecond. Άρα 5 ms = 5.000.000 ns.

Αντίστοιχα, έγινε χρονομέτρηση και του χρόνου υπολογισμού ενός κύκλου διεργασίας του πλήρους δυναμικού μοντέλου και ελέγχου των δύο βαθμών ελευθερίας, που ελέγχει ο κώδικας FControl2dof.cpp.

Τα αποτελέσματα ήταν τουλάχιστον μιας τάξης μεγέθους καλύτερα από τον αντίστοιχο κώδικα ελέγχου και των επτά βαθμών ελευθερίας. Έτσι είχαμε τη δυνατότητα να χρησιμοποιήσουμε εντολές επικοινωνίας χαμηλού επιπέδου και να επιτύχουμε κύκλο επικοινωνίας στα **2.5ms**, που είναι και το μέγιστο που μπορούμε. Οι αντίστοιχες εντολές που χρησιμοποιήθηκαν αναλύονται στο Παράρτημα Β 2.3 στην ανάλυση του κώδικα FControl2dof.cpp.

4.3 Εκτέλεση Πειράματος

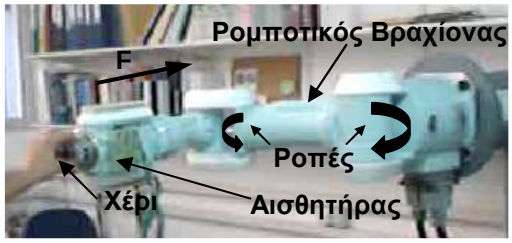
Για λόγους ασφάλειας του ρομποτικού βραχίονα και βέλτιστη λειτουργία του δυναμικού μοντέλου, ούτως ώστε να μπορούμε να αξιολογήσουμε επαρκώς τα πειραματικά μας αποτελέσματα, έγινε η πειραματική εφαρμογή στο επίπεδο κάθετο στη βαρυτική δύναμη χρησιμοποιώντας τις αρθρώσεις 2 και 4 του ρομποτικού βραχίονα PA-10 και βάζοντας φρένα στις υπόλοιπες. Έτσι χρησιμοποιήθηκε και δοκιμάστηκε το λογισμικό ελέγχου Fcontrol2dof.cpp, που έχει αναπτυχθεί για αυτή την εφαρμογή. Η παράθεση του κώδικα ForceControl.cpp, που γίνεται στο Παράρτημα, γίνεται με κάθε επιφύλαξη, καθώς δεν έχει εξεταστεί σε πραγματική λειτουργία από το γράφοντα και είναι ενδεικτική για μελλοντική χρήση και μελέτη.

Ένα επιπλέον πλεονέκτημα της πειραματικής εφαρμογής στους 2 βαθμούς ελευθερίας είναι πως λειτουργεί στη μεγαλύτερη δυνατή συχνότητα (μικρότερη δυνατή περίοδο) επικοινωνίας, τα 2.5ms ανά κύκλο. Είναι σημαντικό να σημειώσουμε πως, σε εφαρμογές ελέγχου δύναμης, η όσο το δυνατόν αμεσότερη και ταχύτερη απόκριση του συστήματος είναι απολύτως απαραίτητη για την ασφάλεια του ρομποτικού βραχίονα και την ορθή λειτουργία του. Σαν παράδειγμα μπορούμε να περιγράψουμε την επαφή του ρομποτικού βραχίονα με ένα συμπαγές εμπόδιο σε περιβάλλον που δεν έχει καθόλου υποχωρητικότητα. Οι δυνάμεις που αναπτύσσονται σε αυτή την περίπτωση αυξάνονται ραγδαία και ένας "αργός" έλεγχος δύναμης δεν θα προλάβει να ενημερώσει το σύστημα ελέγχου ορθά για την αντιμετώπιση αυτής της δυσμενής κατάστασης.

Η πειραματική διαδικασία ρυθμίστηκε να διαρκέσει 8000 κύκλους επικοινωνίας με τον ελεγκτή του ρομποτικού βραχίονα, και επειδή, όπως είπαμε, η περίοδος κάθε κύκλου επικοινωνίας ρυθμίστηκε στα 2.5ms, η συνολική διάρκεια του πειράματος ήταν 20 δευτερόλεπτα. Σε κάθε κύκλο λειτουργίας, με βάση την τρέχουσα γωνιακή θέση και γωνιακή ταχύτητα των αρθρώσεων και τις τιμές των δυνάμεων που λαμβάναμε από τον αισθητήρα με εφαρμογή του νόμου ελέγχου δύναμης, που περιγράφτηκε αναλυτικά στα προηγούμενα κεφάλαια, στέλναμε ροπές στους κινητήρες του ρομποτικού βραχίονα.

Στο σχήμα που ακολουθεί παρουσιάζεται μια τυπική θέση του συστήματος ανθρώπινου χεριού, ρομποτικού βραχίονα και αισθητήρα δύναμης κατά τη διάρκεια του πειράματος, όπου εφαρμογή δύναμης F μέσω του ανθρώπου προς την κατεύθυνση που παρουσιάζεται έχει ως αποτέλεσμα, μέσω του νόμου ελέγχου δύναμης, την εφαρμογή κατάλληλης ροπής στις δύο αρθρώσεις για την αναπαραγωγή αυτής της δύναμης από το τελικό σημείο δράσης του ρομποτικού βραχίονα. Στη συγκεκριμένη περίπτωση, που δεν έχουμε κάποιο φορτίο να μετακινήσουμε, έχουμε θέσει το συντελεστή ενίσχυσης α ίσο με το μηδέν. Μια ενισχυμένη δύναμη μέσω του ρομποτικού βραχίονα, από τη στιγμή που αυτή δεν εφαρμόζεται στο

περιβάλλον, θα προκαλούσε ραγδαία επιτάχυνση του προς τη φορά κίνησης, αφού με την αντιστάθμιση και των τριβών δεν υπάρχει κάποια αντίσταση.

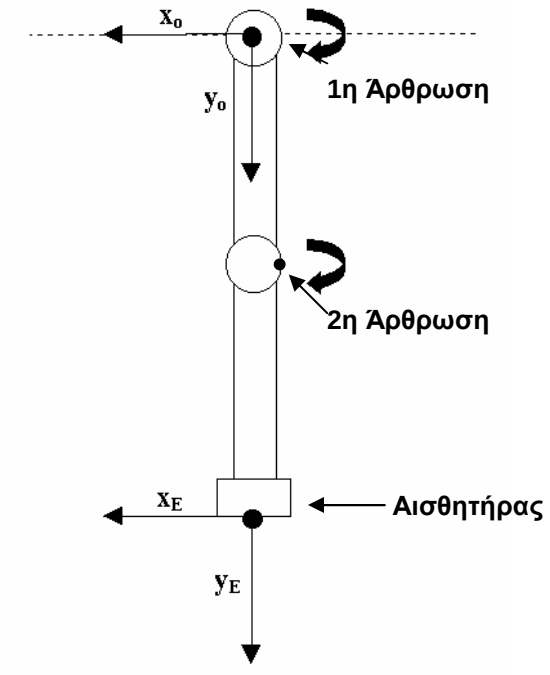


Εικόνα 13. Εκτέλεση πειράματος. Διάταξη σε τυχαία κατάσταση

Για κάθε κύκλο λειτουργίας γινόταν συλλογή και αποθήκευση όλων των απαραίτητων μεταβλητών ελέγχου και κατάστασης του συστήματος για να αναλυθούν πλήρως με το πέρας των πειραμάτων.

Τα πειραματικά δεδομένα που αναλύονται στο επόμενο κεφάλαιο είναι αποτέλεσμα ενός πειράματος, σε ένα κομμάτι της διάρκειας του οποίου δοκιμάστηκε πολλές φορές η γρήγορη εναλλαγή της φοράς άσκησης της δύναμης και ως προς τον x, αλλά και ως προς τον y άξονα για τη μελέτη της απόκρισης του συστήματος. Έτσι, θα μπορούσε να ελεγχθεί κατά πόσο αντιδρά στη εφαρμογή της επιθυμητής δύναμης ο ρομποτικός βραχίονας και διαισθητικά από το χρήστη, αλλά και μαθηματικά μελετώντας τα αντίστοιχα διαγράμματα.

Τέλος, για τη σωστή κατανόηση των πειραμάτων που ακολουθούν παρουσιάζουμε το σύστημα συντεταγμένων της βάσης και του τελικού σημείου δράσης πάνω στα οποία υπολογίστηκαν οι παράμετροι της διάταξης.



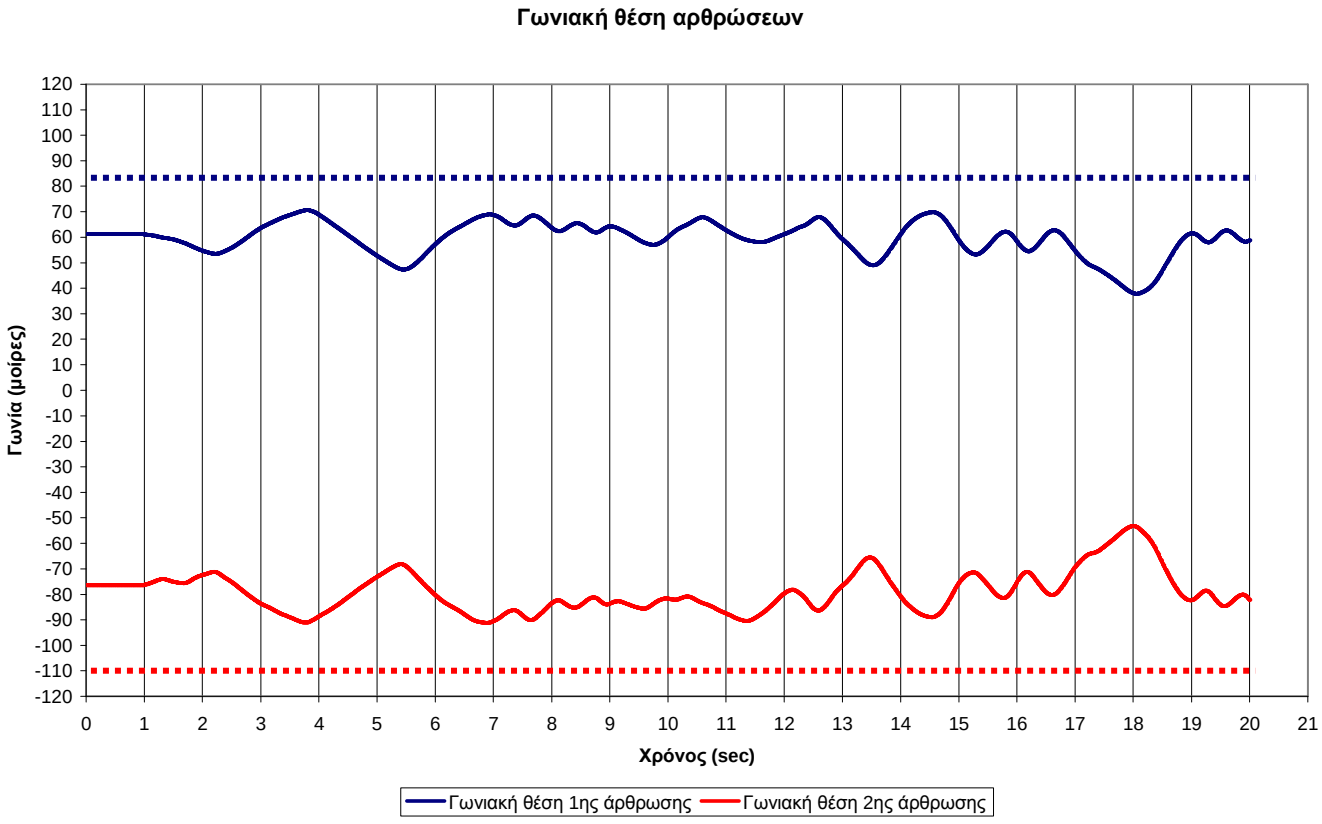
Σχήμα 20. Σχηματική αναπαράσταση κάτοψης πειραματικής διάταξης και συστήματα συντεταγμένων.

4.4 Αποτελέσματα Πειράματος

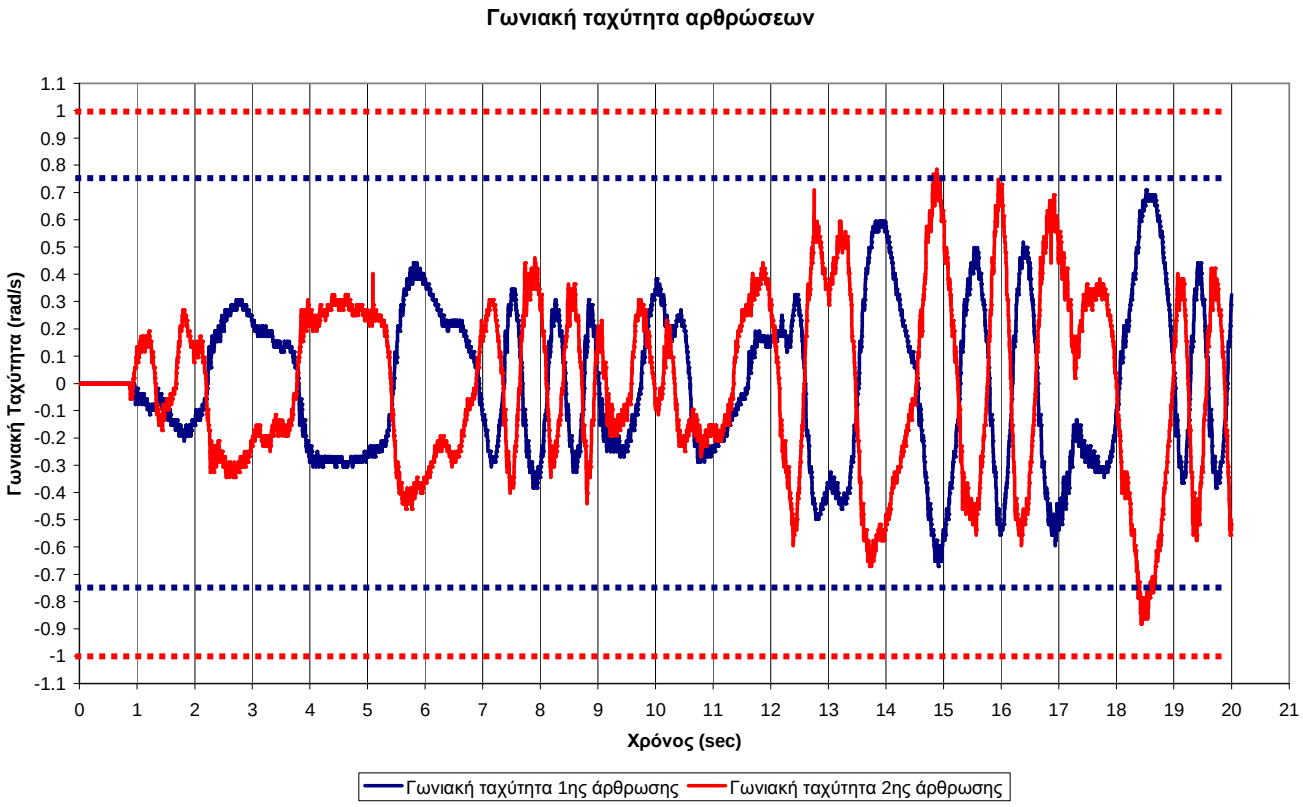
Κατά την πειραματική εφαρμογή ο ρομποτικός βραχίονας αντιδρούσε άμεσα ακόμα και με ελάχιστη εφαρμογή δύναμης προς κάποια κατεύθυνση, όπως ήταν και το επιθυμητό. Απότομες εναλλαγές εφαρμογής δύναμης από το χρήστη απέδειξαν πως και ο νόμος ελέγχου δύναμης αποκρινόταν άμεσα και έστελνε τις κατάλληλες ροπές στους κινητήρες.

Η μέγιστη δύναμη που εφαρμόστηκε καθόλη τη διάρκεια του πειράματος δεν ξεπέρασε τα 15 N, τιμή που εμφανίστηκε μόνο κατά τις απότομες εναλλαγές κατεύθυνσης της δύναμης και αντιστοιχούσε και στην επιβράδυνση του τελικού σημείου δράσης μέχρι το μηδέν, αλλά και την επιτάχυνσή του προς την αντίθετη από την προηγούμενη φορά. Τα χαμηλά επίπεδα δυνάμεων ήταν αναμενόμενα και επιθυμητά, καθώς η κίνηση του βραχίονα προς την ίδια φορά κίνησης του χεριού δεν επέτρεπε, αλλά και δεν απαιτούσε την εφαρμογή μεγαλύτερων δυνάμεων από τη στιγμή που δεν υπήρχε κάποιο εμπόδιο.

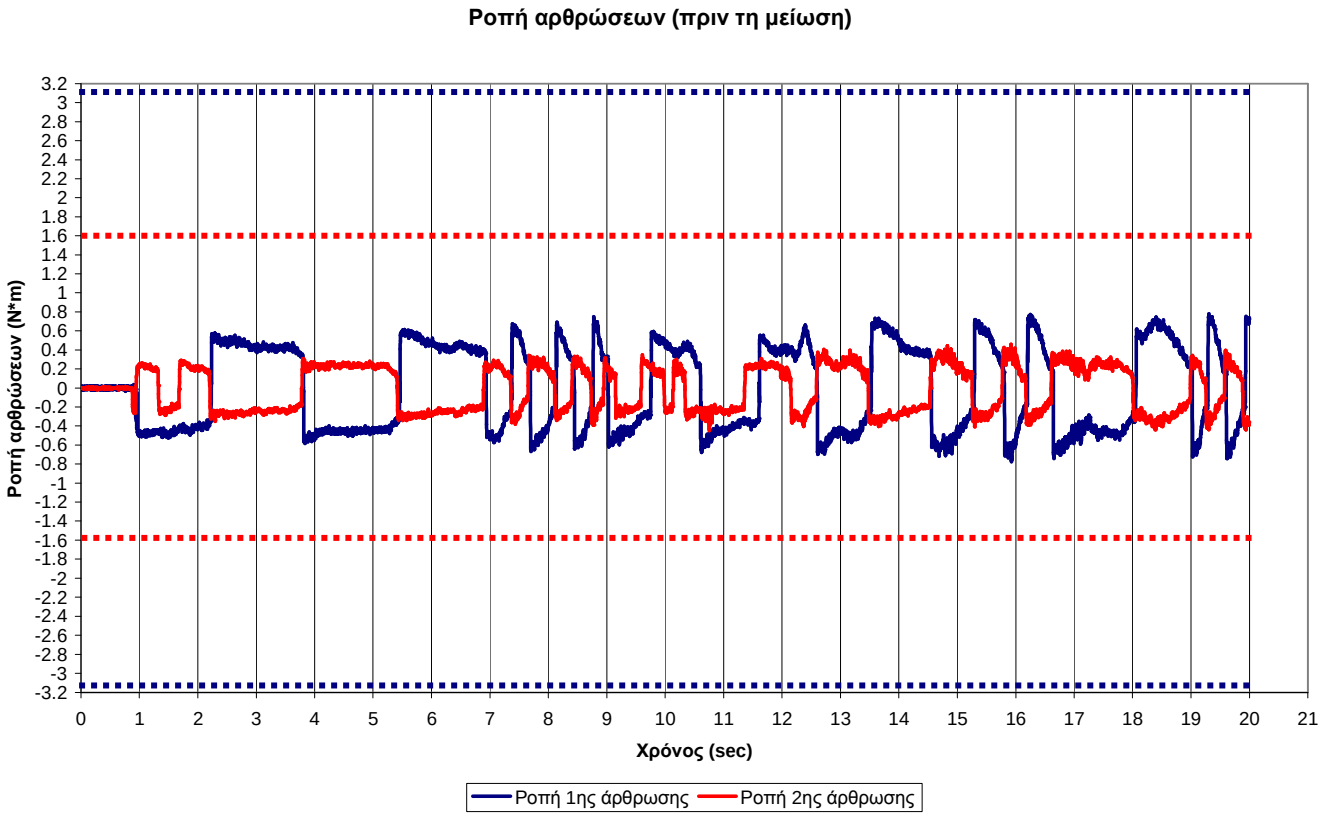
Ακολουθεί η παράθεση των σημαντικότερων διαγραμμάτων μεταβλητών κατάστασης κατά την εκτέλεση του πειράματος και στη συνέχεια αναλύουμε τα αποτελέσματα που προκύπτουν μελετώντας καθένα από αυτά.



Διάγραμμα 18. Γωνιακή θέση αρθρώσεων και μηχανικά όρια.

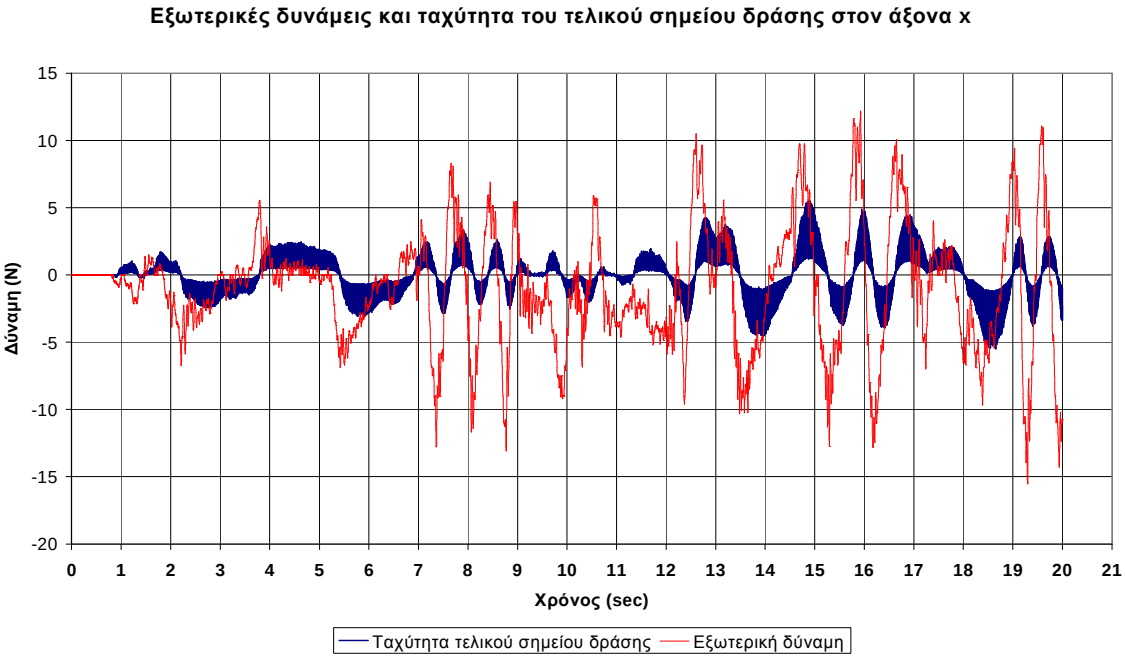


Διάγραμμα 19. Γωνιακή ταχύτητα αρθρώσεων και μηχανικά όρια.

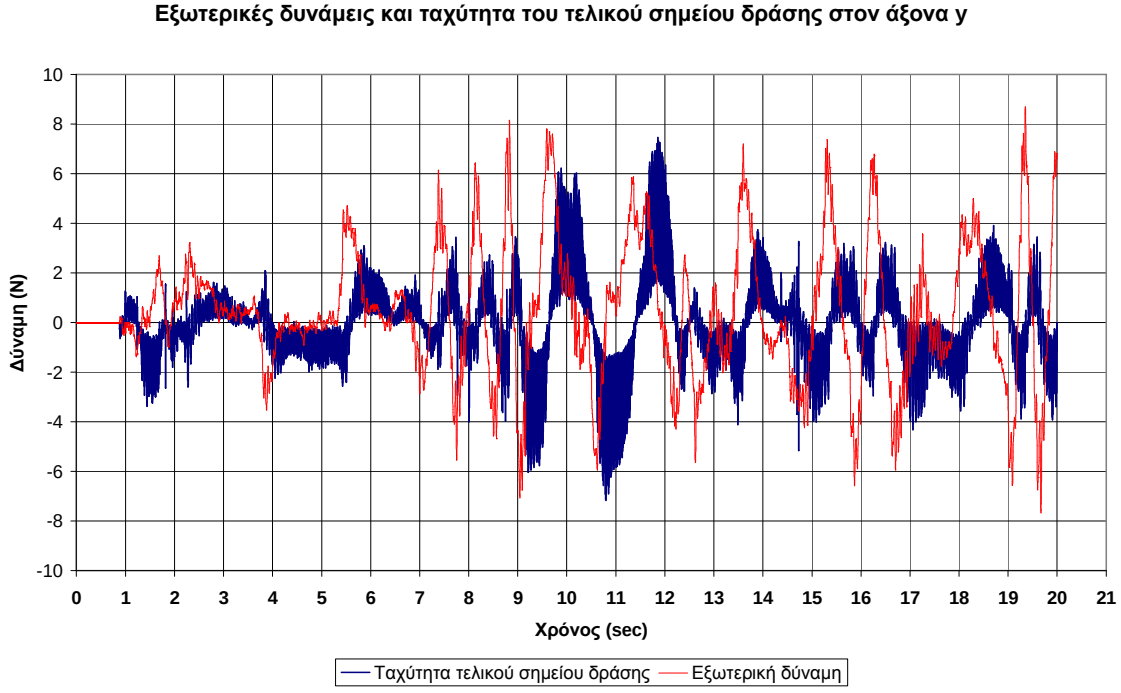


Διάγραμμα 20. Ροπή αρθρώσεων πριν τη μείωση και μηχανικά όρια.

Σε κάθε κύκλο λειτουργίας γίνεται έλεγχος σε τρία μεγέθη, τη γωνιακή θέση, τη γωνιακή ταχύτητα και τη ροπή που εφαρμόζεται σε κάθε άρθρωση. Σε περίπτωση που αυτά τα μεγέθη βρίσκονται εντός των προκαθορισμένων ορίων, συνεχίζεται άσκοπα η διεξαγωγή του πειράματος. Στα τρία παραπάνω διαγράμματα βλέπουμε αυτές τις μεταβλητές σε συνάρτηση με το χρόνο για κάθε άρθρωση ξεχωριστά και με διακεκομμένη γραμμή φαίνονται τα αντίστοιχα όρια που έχουμε θέσει.



Διάγραμμα 21. Εξωτερική δύναμη και ταχύτητα του τελικού σημείου δράσης στον άξονα x.



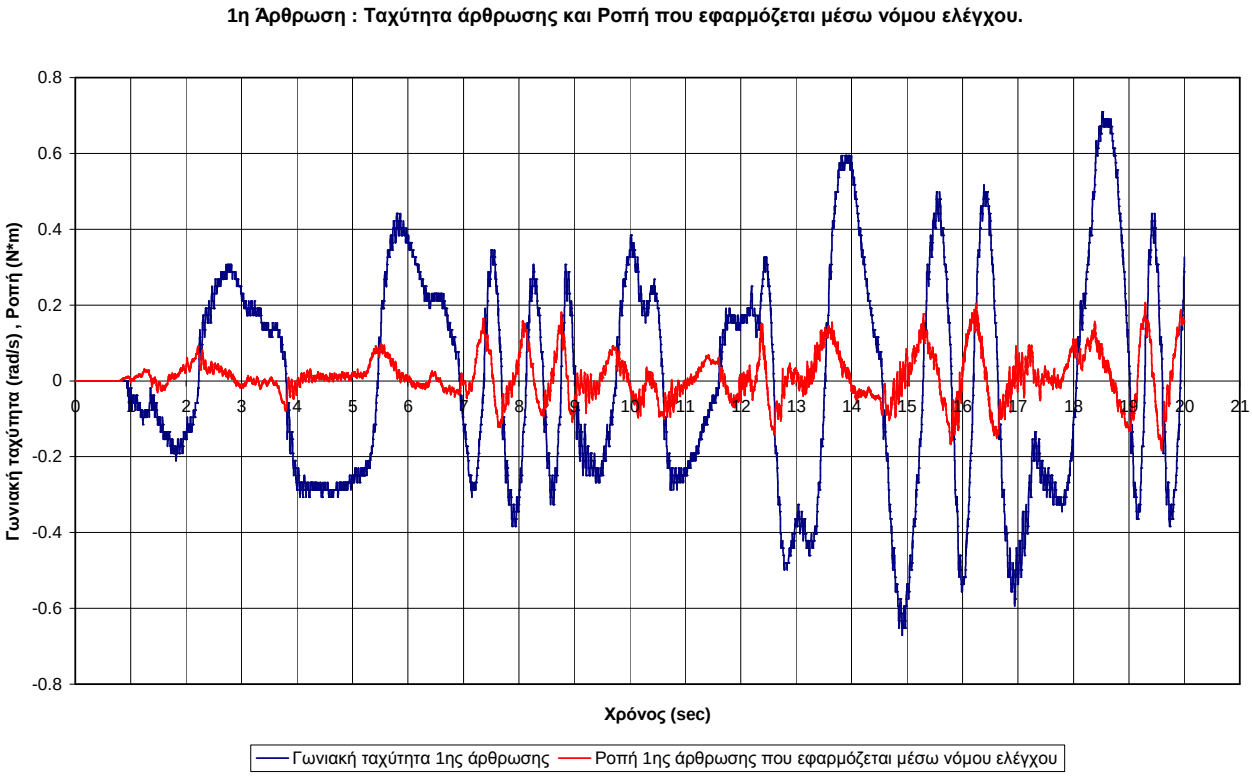
Διάγραμμα 22. Εξωτερική δύναμη και ταχύτητα του τελικού σημείου δράσης στον άξονα y.

Στα παραπάνω διαγράμματα έχει γίνει διαφορετική προβολή της δύναμης που ασκείται από το χρήστη και της αντίστοιχης ταχύτητας του τελικού σημείου δράσης, ανάλογα με τον άξονα στον οποίο αναφέρονται, καθώς τα μεγέθη αυτά είναι εξαρτημένα μεταξύ τους ανά άξονα και οι δύο άξονες είναι ανεξάρτητοι. Μελετώντας τα με προσοχή βγάζουμε τα εξής συμπεράσματα:

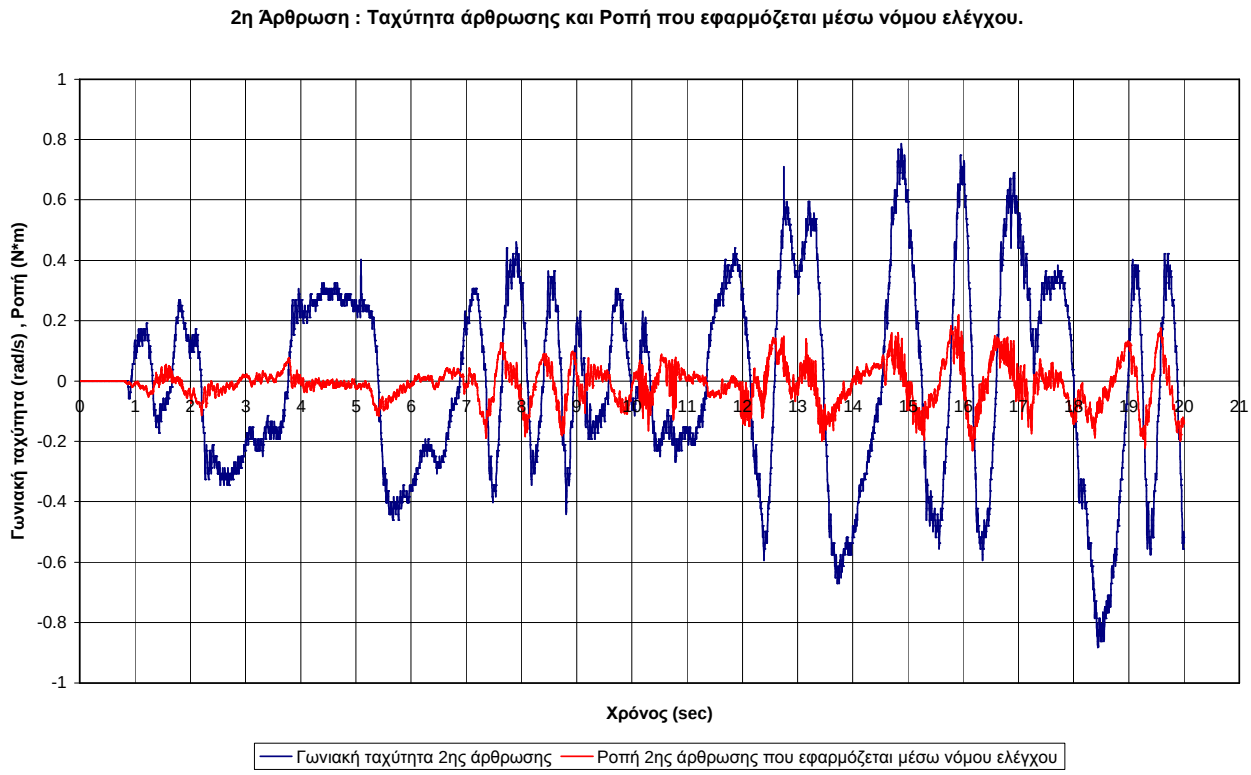
Όπως ήταν αναμενόμενο, η σταθερή εφαρμογή μηδενικής δύναμης έχει ως αποτέλεσμα τη διατήρηση σταθερής ταχύτητας για το χρονικό διάστημα αυτό.

Η εφαρμογή μιας δύναμης προς μία κατεύθυνση έχει ως αποτέλεσμα την βαθμιαία επιτάχυνση του τελικού σημείου δράσης προς αυτή την κατεύθυνση με την ίδια φορά της δύναμης.

Τα τοπικά μέγιστα ή ελάχιστα (οι κορυφές στο διάγραμμα) της ταχύτητας του τελικού σημείου δράσης παρατηρούνται ακριβώς τη χρονική στιγμή που αλλάζει η φορά της εφαρμοζόμενης δύναμης. Αυτό αποδεικνύει την ταχύτατη απόκριση του συστήματος στις δυνάμεις αυτές.



Διάγραμμα 23. Ροπή μέσω νόμου ελέγχου και ταχύτητα 1ης άρθρωσης.

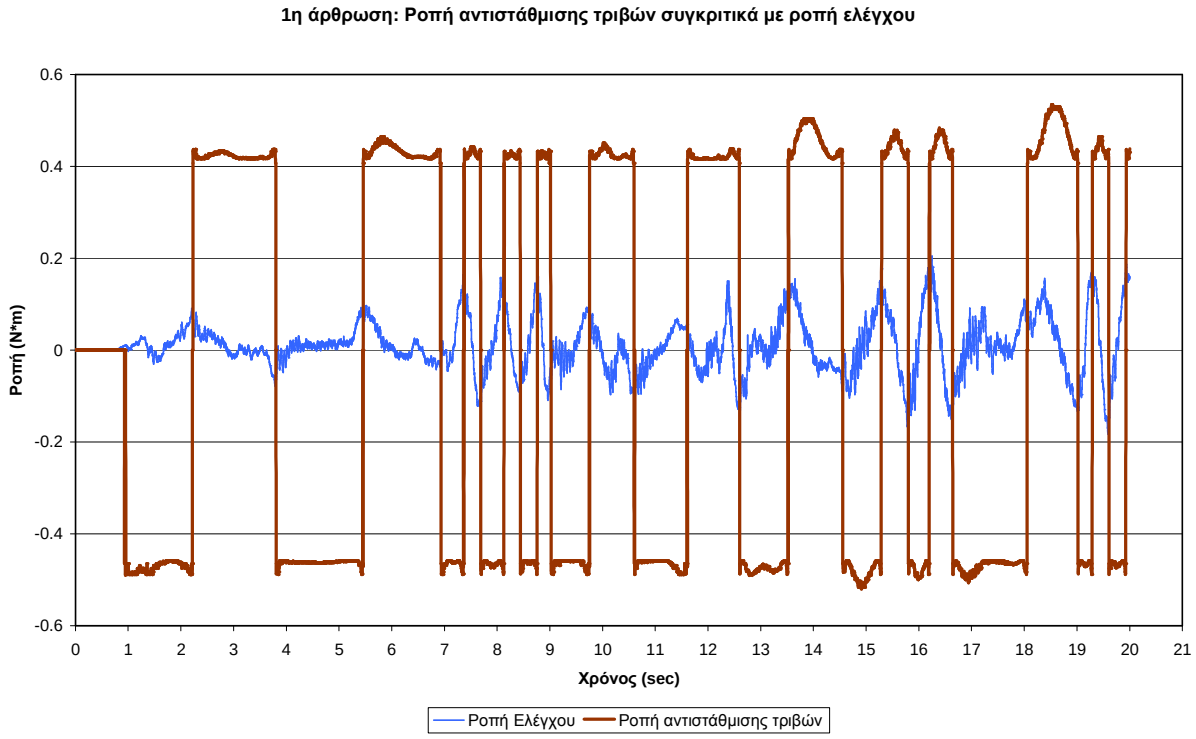


Διάγραμμα 24. Ροπή μέσω νόμου ελέγχου και ταχύτητα 2ης άρθρωσης.

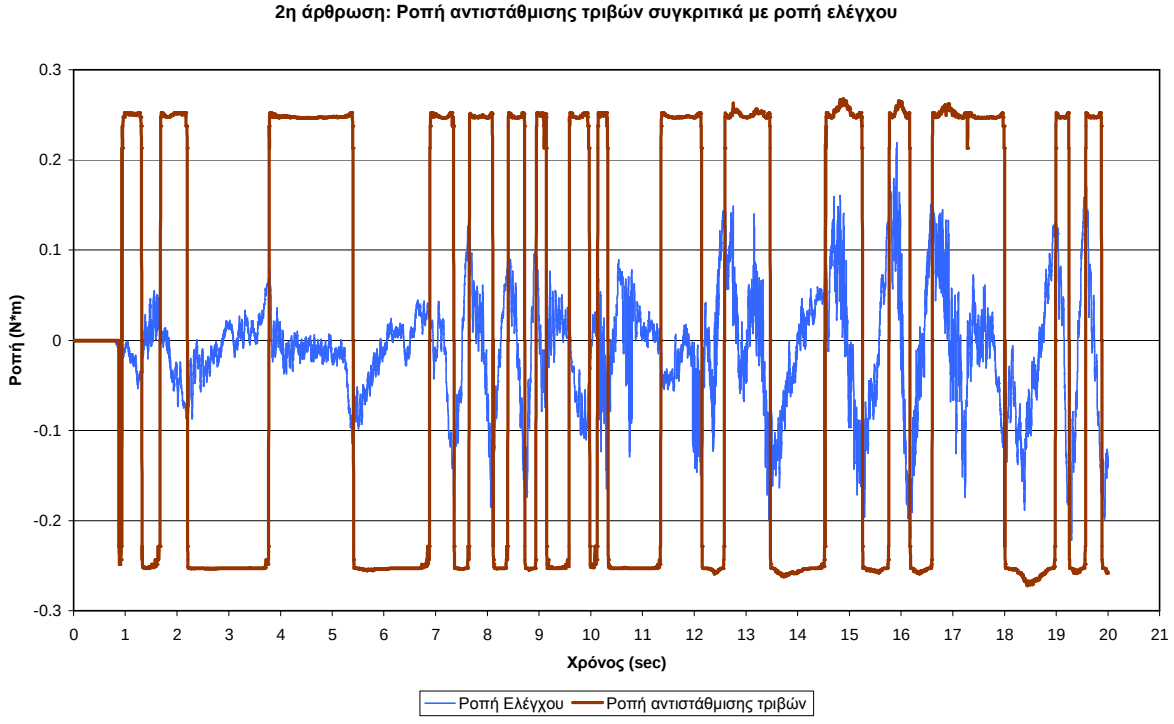
Σε αντίστοιχα διαγράμματα στο χώρο των αρθρώσεων συγκρίνουμε ανά άρθρωση τη γωνιακή ταχύτητα που μετράει ο ελεγκτής του ρομποτικού βραχίονα ανά χρονική στιγμή με την τιμή της ροπής που στέλνουμε μέσω του νόμου ελέγχου στην άρθρωση αυτή. Θεωρητικά η τιμή της ροπής αυτής είναι η υπεύθυνη για την κίνηση του ρομποτικού βραχίονα, καθώς το υπόλοιπο κομμάτι της ροπής που στέλνουμε στους κινητήρες των αρθρώσεων αφορά την αντιστάθμιση της τριβής (μέσω του δυναμικού μοντέλου) των αρθρώσεων και της εφαρμοζόμενης εξωτερικά δύναμης.

Έτσι αναμένουμε να παρατηρήσουμε πλήρη εξάρτηση της μεταβολής της γωνιακής ταχύτητας από τη ροπή μέσω του νόμου ελέγχου για να επαληθεύσουμε την ορθή εφαρμογή του.

Βλέπουμε πως μηδενική ροπή λόγω του νόμου ελέγχου έχει ως αποτέλεσμα να διατηρείται σταθερή η γωνιακή ταχύτητα της αντίστοιχης άρθρωσης. Οποιαδήποτε μεταβολή προς κάποια φορά έχει ως αποτέλεσμα να επιταχύνεται η άρθρωση αυτή προς αυτή τη φορά. Επίσης στις χρονικές στιγμές που έχουμε τοπικά μέγιστα στη γωνιακή ταχύτητα κάθε άρθρωσης είναι όταν αλλάζει η φορά της ροπής που υπολογίζεται μέσω του νόμου ελέγχου αναγκάζοντας την άρθρωση να επιβραδυνθεί. Έτσι, όπως αναμέναμε, η κίνηση των αρθρώσεων καθορίζεται μέσω της ροπής, που υπολογίζεται μέσω του νόμου ελέγχου με πολύ ταχεία απόκριση.



Διάγραμμα 25. Ροπή μέσω νόμου ελέγχου και ροπή αντιστάθμισης τριβής για την 1η άρθρωση.



Διάγραμμα 26. Ροπή μέσω νόμου ελέγχου και ροπή αντιστάθμισης τριβής για την 2η άρθρωση.

Μια ενδεικτική παράθεση σε κοινό διάγραμμα της ροπής που απαιτείται για την αντιστάθμιση της τριβής μιας άρθρωσης και της ροπής που εφαρμόζουμε μέσω του νόμου ελέγχου στην άρθρωση αυτή, μας οδηγεί στα εξής συμπεράσματα:

Εκτός του γεγονότος πως φαίνεται καθαρά πλέον η εναλλαγή της φοράς περιστροφής της άρθρωσης από τις ροπές που στέλνουμε για να υπερνικήσουμε τις τριβές φαίνεται και η σχέση της εναλλαγής αυτής με τις κορυφές της ροπής που εφαρμόζουμε μέσω του νόμου ελέγχου.

Πρέπει όμως να παρατηρήσουμε πως οι ροπές για την υπερνίκηση των τριβών στην 1η άρθρωση είναι μιας τάξης μεγέθους μεγαλύτερες σε σχέση με τις ροπές λόγω του νόμου ελέγχου και αυτό σημαίνει πως σε περίπτωση κάποιας απόκλισης του δυναμικού μοντέλου τριβής του ρομποτικού βραχίονα σε σχέση με το πραγματικό υπάρχει σημαντική επίδραση στην όλη διάταξη και στην ομαλή και επιτυχημένη διεξαγωγή αντίστοιχου πειράματος.

ΚΕΦΑΛΑΙΟ 5ο : Αξιολόγηση - Μελλοντικές Κατευθύνσεις

5.1 Αξιολόγηση Πειραματικών αποτελεσμάτων

Τόσο η συμπεριφορά του ρομποτικού βραχίονα κατά την εκτέλεση των πειραμάτων, όσο και τα αποτελέσματα, που καταγράφηκαν με τη μορφή παραμέτρων και μεταβλητών του συστήματος σε συνάρτηση με το διακριτό χρόνο του κύκλου υπολογισμών και μελετήθηκαν σε διαγράμματα, έδειξαν πως η πειραματική εφαρμογή πλησίαζε πάρα πολύ τα αναμενόμενα από τις προσομοιώσεις και τις θεωρητικές προσεγγίσεις του θέματος.

Εφαρμογή έστω και μικρής δύναμης από τον άνθρωπο μπορούσε να ανιχνευτεί μέσω του αισθητήρα και να ακολουθηθεί αυτή η πρόθεση κίνησης από το ρομποτικό βραχίονα, ακόμα και σε γρήγορες εναλλαγές της φοράς εφαρμογής της δύναμης.

Ένα μειονέκτημα που παρατηρήθηκε κατά τη διάρκεια των πειραμάτων ήταν ή δυσκολία να φέρουμε σε ακινησία το ρομποτικό βραχίονα, εξισορροπώντας πλήρως τις δυνάμεις επαφής. Το συνηθέστερο αποτέλεσμα ήταν μια μικρού πλάτους ταλάντωση γύρω από το σημείο ισορροπίας με περιοδική εναλλαγή της φοράς περιστροφής της 1ης άρθρωσης της διάταξής μας (2η σε σειρά άρθρωση του ρομποτικού βραχίονα στο σύνολο). Έπειτα από πολλές επαναλήψεις του πειράματος και προσεκτική μελέτη των αποτελεσμάτων, συμπεράναμε πως μια μικρή απόκλιση στον υπολογισμό της ροπής για την αντιστάθμιση της τριβής σε αυτή την άρθρωση μπορούσε να προκαλέσει αυτό το φαινόμενο.

Λόγω της κατασκευής και του όγκου αυτής της άρθρωσης, η ροπή για την αντιστάθμιση των τριβών σε αυτή ήταν κατά κανόνα μια τάξη μεγέθους μεγαλύτερη από τη ροπή που εφαρμόζαμε μέσω του νόμου ελέγχου, και ένα μικρό σχετικό σφάλμα θα μπορούσε να επηρεάσει αρκετά την τελική συμπεριφορά του ρομποτικού βραχίονα σε σχέση με την αναμενόμενη.

Επίσης δοκιμάστηκαν περιπτώσεις με συντελεστή ενίσχυσης μεγαλύτερο του μηδενός, επειδή όμως δεν υπήρχε κάποιο φορτίο να αντισταθεί σε αυτή τη δύναμη που ασκούσε ο βραχίονας, το αποτέλεσμα ήταν να έχουμε μεγαλύτερες επιταχύνσεις σε σχέση με εφαρμογή ίσης δύναμης με μηδενικό συντελεστή ενίσχυσης. Η μη ύπαρξη αντιστεκόμενου φορτίου μπορούσε να έχει ως αποτέλεσμα και με μικρή σχετικά δύναμη το τελικό σημείο δράσης να λάμβανε τέτοια επιτάχυνση, που σε λίγο χρόνο να υπερβαίναμε τη μέγιστη ταχύτητα κάποιας άρθρωσης, κάτι που για λόγους ασφαλείας είχαμε προγραμματίσει να έχει ως αποτέλεσμα τον πλήρη τερματισμό του προγράμματος και της επικοινωνίας μας με τον ελεγκτή του ρομποτικού βραχίονα.

5.2 Κατευθύνσεις και Προτάσεις για περαιτέρω ανάπτυξη

Σε πρώτο επίπεδο πρέπει να προτείνουμε τις βελτιώσεις και τροποποιήσεις της δεδομένης πειραματικής διάταξης για καλύτερα και ακριβέστερα αποτελέσματα:

- Εκ νέου υπολογισμός και επαλήθευση του μοντέλου αντιστάθμισης της τριβής για τις δύο αρθρώσεις που χρησιμοποιούμε στην πειραματική εφαρμογή (2η και 4η του ρομποτικού βραχίονα), για να εξαλείψουμε τα σφάλματα που προκαλούνται από αυτόν τον παράγοντα.
- Υπολογισμός των υπόλοιπων παραμέτρων του δυναμικού μοντέλου (μητρώο αδράνειας, μητρώο φυγόκεντρων δυνάμεων και δυνάμεων Coriolis) για τον ανηγμένο σε δύο βαθμούς ελευθερίας ρομποτικό βραχίονα, παρά τη χαμηλή επίδραση αυτών των παραγόντων σε ένα πείραμα χαμηλών σχετικά ταχυτήτων.
- Προσαρμογή ενός φορτίου στον ακροδέκτη του ρομποτικού βραχίονα και συμπεριφορά του νόμου ελέγχου με συντελεστή ενίσχυσης μεγαλύτερο του μηδενός, με σκοπό με μικρή εφαρμογή δύναμης να μπορούμε να μετακινήσουμε και να χειριστούμε ένα κατά πολύ μεγαλύτερο φορτίο.
- Χρήση εναλλακτικών μεθοδολογιών ελέγχου (π.χ έλεγχος δύναμης με εσωτερικό βρόχο ταχύτητας, παράλληλος έλεγχος) και σύγκριση των αποτελεσμάτων.

Στη συνέχεια γίνονται προτάσεις, που, με βάση τα δεδομένα πειραματικά αποτελέσματα και την εμπειρία και τεχνογνωσία που αποκτήθηκε, μπορούν να είναι άμεσα εφικτές και πραγματοποιήσιμες.

- Μελέτη μεθοδολογιών ελέγχου για περισσότερες των δύο αρθρώσεων και σε τρισδιάστατα προβλήματα με επιθυμητό προσανατολισμό
- Επαλήθευση και χρήση του πλήρους δυναμικού μοντέλου του ρομποτικού βραχίονα και εφαρμογές νόμου ελέγχου δύναμης σε πραγματικό χρόνο εκμεταλλευόμενοι και τους επτά βαθμούς ελευθερίας του και την ανάλογη επιδεξιότητα που αυτοί μας προσδίδουν (διαθέσιμο ενδεικτικό λογισμικό στο Παράρτημα Β 2.2)
- Σύνθετες εφαρμογές με χρήση του ρομποτικού βραχίονα σαν εξωσκελετό, προσδίδοντας ανεπτυγμένες δυνατότητες σε έναν ανθρώπινο βραχίονα μέσω της σύζευξης του τεχνητού και του φυσικού μέλους.

ΠΑΡΑΡΤΗΜΑ

Α. Βοηθητικό λογισμικό, Πίνακες και Διαγράμματα

Α.1 Ρουτίνες παρουσίασης τιμών φορτίσεων του αισθητήρα

```
void drawline(int col, int value) {
     int start;
     if (value>=0) {
          start=40;
     } else {
          start=40+value;
     }
     if (fabs(value)<10) color_set(2,NULL);
     else if (fabs(value)>10) color_set(3,NULL);
     mvhline(col,start,'=',abs(value));
}
```

```
void drawtext(int sensors) {
     color_set(1,NULL);
     if (sensors>0) {
          mvaddstr(4,36,"Fx (N)");
          mvaddstr(6,36,"Fy (N)");
          mvaddstr(8,36,"Fz (N)");
          mvaddstr(10,34,"Mx (N*m*10)");
          mvaddstr(12,34,"My (N*m*10)");
          mvaddstr(14,34,"Mz (N*m*10)");
     }
}
```

```
int main(void)
{
    short f, b;
    six_axis_array fm,ac;
    force_array fs, as;
    float temp,val[6];
    int ret,i,fd,j,k,col,sensors=0;

    initscr();
    cbreak();
    noecho();

    if ((fd=open("/dev/jr3",O_RDWR)) < 0) {      //Checking device

      perror("Can't open device. No way to read force!");
    }
    ioctl(fd,IOCTL0_JR3_ZEROOFFS);
    ret=ioctl(fd,IOCTL0_JR3_GET_FULL_SCALES,&fs);

    if (has_colors()) {                          //Checking Colours
          start_color();
          init_pair(1, COLOR_WHITE, COLOR_BLACK);
          init_pair(2, COLOR_GREEN, COLOR_BLACK);
          init_pair(3, COLOR_RED, COLOR_BLACK);
          curs_set(0);
          while (1) {
          if (sensors>0)
```

```
                ioctl(fd,IOCTL0_JR3_FILTER0,&fm);
          if (sensors>0) {
                erase();
                drawtext(sensors);
                drawline(5,fm.f[0]*fs.f[0]/16384);
                drawline(7,fm.f[1]*fs.f[1]/16384);
                drawline(9,fm.f[2]*fs.f[2]/16384);
                drawline(11,fm.m[0]*fs.m[0]/16384);
                drawline(13,fm.m[1]*fs.m[1]/16384);
                drawline(15,fm.m[2]*fs.m[2]/16384);
          mvprintw(0,30,"Full scales are %d %d %d %d %d
%d\n",fs.f[0],fs.f[1],fs.f[2],fs.m[0],fs.m[1],fs.m[2]);
          mvprintw(1,30,"---------------------------");
          j=4;

          for (i=0;i<3;i++) {         //Calculating Forces
                val[i]=0;
                temp=fm.f[i]*fs.f[i];
                val[i]=temp/16384;
                if (val[i]>=0)
                     col=49;
                else
                     col=48;
                mvprintw(j,col,"%3.1f",val[i]);
                                        //Screen Printing of Forces(1 decimal)
                j=j+2;
          }

          for (i=0;i<3;i++) {               //Calculating Torques
                val[i+3]=0;
                temp=fm.m[i]*fs.m[i];
                val[i+3]=temp/16384;
                if (val[i+3]>=0)
                     col=49;
                else
                     col=48;
                mvprintw(j,col,"%3.1f",val[i+3]);
                                        //Screen Printing of Forces  (1 decimal)
                j=j+2;
                }
          }
          refresh();
     }
    } else {
     printw("This program requires a color terminal");
     getch();
    }
    endwin();

    exit(1);
}
```

Α.2 Συγκεντρωτικοί Πίνακες μετρήσεων τιμών δυνάμεων και ροπών πριν τη βαθμονόμηση του αισθητήρα

2.1 Δυνάμεις Fx, Fy, Fz

Μετρήσεις Fx				***Σφάλμα***	
βάρος (gr)	**Fx (N)**	*Πραγματική δύναμη(N)*		eFx (N)	%
-1700	-16.0	*-16.677*	-16.7	**-0.7**	4.20%
-1600	-15.6	*-15.696*	-15.7	-0.1	0.64%
-1500	-14.4	*-14.715*	-14.7	**-0.3**	2.04%
-1400	-13.3	*-13.734*	-13.7	**-0.4**	2.91%
-1300	-12.1	*-12.753*	-12.8	**-0.7**	5.49%
-1200	-11.6	*-11.772*	-11.8	**-0.2**	1.70%
-1100	-10.4	*-10.791*	-10.8	**-0.4**	3.71%
-1000	-9.8	*-9.81*	-9.8	0.0	0.00%
-900	-8.5	*-8.829*	-8.8	**-0.3**	3.40%
-700	-6.7	*-6.867*	-6.9	**-0.2**	2.91%
-500	-4.9	*-4.905*	-4.9	0.0	0.00%
-400	-3.9	*-3.924*	-3.9	0.0	0.00%
-300	-2.9	*-2.943*	-2.9	0.0	0.00%
-200	-2.0	*-1.962*	-2.0	0.0	0.00%
-100	-1.0	*-0.981*	-1.0	0.0	0.00%
0	0.0	*0*	0.0	0.0	0.00%
50	0.5	*0.4905*	0.5	0.0	0.00%
100	1.0	*0.981*	1.0	0.0	0.00%
150	1.5	*1.4715*	1.5	0.0	0.00%
200	2.0	*1.962*	2.0	0.0	0.00%
250	2.4	*2.4525*	2.5	0.1	4.08%
300	2.9	*2.943*	2.9	0.0	0.00%
350	3.3	*3.4335*	3.4	0.1	2.91%
400	3.9	*3.924*	3.9	0.0	0.00%
450	4.3	*4.4145*	4.4	0.1	2.27%
500	4.9	*4.905*	4.9	0.0	0.00%
600	5.8	*5.886*	5.9	0.1	1.70%
700	6.6	*6.867*	6.9	**0.3**	4.37%
800	7.8	*7.848*	7.8	0.0	0.00%
900	8.5	*8.829*	8.8	**0.3**	3.40%
1000	9.8	*9.81*	9.8	0.0	0.00%
1100	10.4	*10.791*	10.8	**0.4**	3.71%
1200	11.6	*11.772*	11.8	**0.2**	1.70%
1300	12.1	*12.753*	12.8	**0.7**	5.49%
1400	13.3	*13.734*	13.7	**0.4**	2.91%
1450	13.5	*14.2245*	14.2	**0.7**	4.92%
1500	14.5	*14.715*	14.7	**0.2**	1.36%
1600	15.6	*15.696*	15.7	0.1	0.64%
1700	16.0	*16.677*	16.7	**0.7**	4.20%
1900	18.8	*18.639*	18.6	**-0.2**	-1.07%

Πίνακας 8. Μετρήσεις δύναμης Fx και αποκλίσεις από την πραγματική τιμή.

Μετρήσεις Fy				***Σφάλμα***	
βάρος (gr)	**Fy (N)**	*Πραγματική δύναμη(N)*		eFy (N)	%
-1700	-16.3	*-16.677*	-16.7	**-0.4**	2.40%
-1600	-15.9	*-15.696*	-15.7	**0.2**	-1.27%
-1500	-14.7	*-14.715*	-14.7	0.0	0.00%
-1400	-13.6	*-13.734*	-13.7	-0.1	0.73%
-1300	-12.4	*-12.753*	-12.8	**-0.4**	3.14%
-1200	-11.9	*-11.772*	-11.8	0.1	-0.85%
-1100	-10.5	*-10.791*	-10.8	**-0.3**	2.78%
-1000	-9.9	*-9.81*	-9.8	0.1	-1.02%
-900	-8.6	*-8.829*	-8.8	**-0.2**	2.27%
-700	-6.8	*-6.867*	-6.9	-0.1	1.46%
-500	-5.0	*-4.905*	-4.9	0.1	-2.04%
-400	-3.9	*-3.924*	-3.9	0.0	0.00%
-300	-3.0	*-2.943*	-2.9	0.1	-3.40%
-200	-2.0	*-1.962*	-2.0	0.0	0.00%
-100	-1.0	*-0.981*	-1.0	0.0	0.00%
0	0.0	*0*	0.0	0.0	0.00%
50	0.5	*0.4905*	0.5	0.0	0.00%
100	0.9	*0.981*	1.0	0.1	10.19%
150	1.4	*1.4715*	1.5	0.1	6.80%
200	2.0	*1.962*	2.0	0.0	0.00%
250	2.4	*2.4525*	2.5	0.1	4.08%
300	2.9	*2.943*	2.9	0.0	0.00%
350	3.3	*3.4335*	3.4	0.1	2.91%
400	3.9	*3.924*	3.9	0.0	0.00%
450	4.2	*4.4145*	4.4	**0.2**	4.53%
500	4.8	*4.905*	4.9	0.1	2.04%
600	5.8	*5.886*	5.9	0.1	1.70%
700	6.6	*6.867*	6.9	**0.3**	4.37%
800	7.4	*7.848*	7.8	**0.4**	5.10%
900	8.0	*8.829*	8.8	**0.8**	9.06%
950	9.5	*9.3195*	9.3	**-0.2**	-2.15%
1000	9.9	*9.81*	9.8	-0.1	-1.02%
1100	10.4	*10.791*	10.8	**0.4**	3.71%
1200	11.6	*11.772*	11.8	**0.2**	1.70%
1300	12.1	*12.753*	12.8	**0.7**	5.49%
1400	13.5	*13.734*	13.7	**0.2**	1.46%
1450	14.4	*14.2245*	14.2	**-0.2**	-1.41%
1500	14.7	*14.715*	14.7	0.0	0.00%
1600	15.6	*15.696*	15.7	0.1	0.64%
1700	16.0	*16.677*	16.7	**0.7**	4.20%
1900	19.0	*18.639*	18.6	**-0.4**	-2.15%

Πίνακας 9. Μετρήσεις δύναμης Fy και αποκλίσεις από την πραγματική τιμή.

Μετρήσεις Fz				***Σφάλμα***	
βάρος (gr)	**Fz (N)**	*Πραγματική δύναμη(N)*		eFz (N)	%
-2000	-20.0	*-19.62*	-19.6	**0.4**	-2.04%
-1900	-19.0	*-18.639*	-18.6	**0.4**	-2.15%
-1800	-18.0	*-17.658*	-17.7	**0.3**	-1.70%
-1700	-16.6	*-16.677*	-16.7	-0.1	0.60%
-1600	-15.7	*-15.696*	-15.7	0.0	0.00%
-1500	-14.7	*-14.715*	-14.7	0.0	0.00%
-1400	-13.6	*-13.734*	-13.7	-0.1	0.73%
-1300	-13.2	*-12.753*	-12.8	**0.4**	-3.14%
-1200	-11.7	*-11.772*	-11.8	-0.1	0.85%
-1100	-10.4	*-10.791*	-10.8	**-0.4**	3.71%
-1000	-9.9	*-9.81*	-9.8	0.1	-1.02%
-900	-8.5	*-8.829*	-8.8	**-0.3**	3.40%
-800	-7.8	*-7.848*	-7.8	0.0	0.00%
-700	-6.8	*-6.867*	-6.9	-0.1	1.46%
-600	-5.9	*-5.886*	-5.9	0.0	0.00%
-500	-4.8	*-4.905*	-4.9	-0.1	2.04%
-400	-3.9	*-3.924*	-3.9	0.0	0.00%
-300	-2.9	*-2.943*	-2.9	0.0	0.00%
-250	-2.4	*-2.4525*	-2.5	-0.1	4.08%
-200	-1.9	*-1.962*	-2.0	-0.1	5.10%
-150	-1.5	*-1.4715*	-1.5	0.0	0.00%
-100	-1.1	*-0.981*	-1.0	0.1	-10.19%
-50	-0.5	*-0.4905*	-0.5	0.0	0.00%
0	0.0	*0*	0.0	0.0	0.00%
50	0.5	*0.4905*	0.5	0.0	0.00%
100	1.0	*0.981*	1.0	0.0	0.00%
150	1.5	*1.4715*	1.5	0.0	0.00%
200	2.0	*1.962*	2.0	0.0	0.00%
250	2.5	*2.4525*	2.5	0.0	0.00%
300	2.9	*2.943*	2.9	0.0	0.00%
350	3.3	*3.4335*	3.4	0.1	2.91%
400	3.8	*3.924*	3.9	0.1	2.55%
450	4.3	*4.4145*	4.4	0.1	2.27%
500	4.9	*4.905*	4.9	0.0	0.00%
550	5.2	*5.3955*	5.4	**0.2**	3.71%
600	5.9	*5.886*	5.9	0.0	0.00%
650	6.2	*6.3765*	6.4	**0.2**	3.14%
700	6.8	*6.867*	6.9	0.1	1.46%
750	7.3	*7.3575*	7.4	0.1	1.36%
800	7.9	*7.848*	7.8	-0.1	-1.27%
850	8.0	*8.3385*	8.3	**0.3**	3.60%
900	8.5	*8.829*	8.8	**0.3**	3.40%
920	8.6	*9.0252*	9.0	**0.4**	4.43%
950	9.4	*9.3195*	9.3	-0.1	-1.07%
1000	9.8	*9.81*	9.8	0.0	0.00%
1050	10.0	*10.3005*	10.3	**0.3**	2.91%
1100	10.4	*10.791*	10.8	**0.4**	3.71%
1120	10.5	*10.9872*	11.0	**0.5**	4.55%

1150	11.5	*11.2815*	11.3	**-0.2**	-1.77%
1200	11.8	*11.772*	11.8	0.0	0.00%
1250	12.0	*12.2625*	12.3	**0.3**	2.45%
1300	12.4	*12.753*	12.8	**0.4**	3.14%
1350	13.2	*13.2435*	13.2	0.0	0.00%
1400	13.5	*13.734*	13.7	**0.2**	1.46%
1500	14.5	*14.715*	14.7	**0.2**	1.36%
1520	14.7	*14.9112*	14.9	**0.2**	1.34%
1540	14.8	*15.1074*	15.1	**0.3**	1.99%
1560	15.2	*15.3036*	15.3	0.1	0.65%
1580	15.6	*15.4998*	15.5	-0.1	-0.65%
1600	15.7	*15.696*	15.7	0.0	0.00%
1650	16.0	*16.1865*	16.2	**0.2**	1.24%
1700	16.1	*16.677*	16.7	**0.6**	3.60%
1750	16.3	*17.1675*	17.2	**0.9**	5.24%
1800	17.0	*17.658*	17.7	**0.7**	3.96%
1850	17.2	*18.1485*	18.1	**0.9**	4.96%
1870	18.7	*18.3447*	18.3	**-0.4**	-2.18%
1900	18.8	*18.639*	18.6	**-0.2**	-1.07%
1950	19.1	*19.1295*	19.1	0.0	0.00%
2000	19.7	*19.62*	19.6	-0.1	-0.51%
2050	20.0	*20.1105*	20.1	0.1	0.50%
2200	20.8	*21.582*	21.6	**0.8**	3.71%

Πίνακας 10. Μετρήσεις δύναμης Fz και αποκλίσεις από την πραγματική τιμή.

2.2 Ροπές Mx, My, Mz

Μετρήσεις Mx				***Σφάλμα***	
βάρος (gr)	**Mx(N*m/10)**	*Πραγματική ροπή(N*m/10)*		eMx (N*m/10)	%
-2000	-6.8	*-7.848*	-7.8	**-1.0**	12.74%
-1900	-6.2	*-7.4556*	-7.5	**-1.3**	17.44%
-1800	-6.1	*-7.0632*	-7.1	**-1.0**	14.16%
-1700	-5.9	*-6.6708*	-6.7	**-0.8**	11.99%
-1600	-5.7	*-6.2784*	-6.3	**-0.6**	9.56%
-1500	-5.2	*-5.886*	-5.9	**-0.7**	11.89%
-1400	-5.0	*-5.4936*	-5.5	**-0.5**	9.10%
-1300	-4.8	*-5.1012*	-5.1	**-0.3**	5.88%
-1200	-4.3	*-4.7088*	-4.7	**-0.4**	8.49%
-1100	-4.0	*-4.3164*	-4.3	**-0.3**	6.95%
-1000	-3.7	*-3.924*	-3.9	**-0.2**	5.10%
-900	-3.1	*-3.5316*	-3.5	**-0.4**	11.33%
-800	-2.9	*-3.1392*	-3.1	**-0.2**	6.37%
-700	-2.5	*-2.7468*	-2.7	**-0.2**	7.28%
-600	-2.1	*-2.3544*	-2.4	**-0.3**	12.74%
-500	-1.8	*-1.962*	-2.0	**-0.2**	10.19%
-400	-1.4	*-1.5696*	-1.6	**-0.2**	12.74%
-300	-1.1	*-1.1772*	-1.2	-0.1	8.49%
-200	-0.7	*-0.7848*	-0.8	-0.1	12.74%
-100	-0.4	*-0.3924*	-0.4	0.0	0.00%
-50	-0.2	*-0.1962*	-0.2	0.0	0.00%
0	0.0	*0*	0.0	0.0	0.00%
50	0.2	*0.1962*	0.2	0.0	0.00%
100	0.4	*0.3924*	0.4	0.0	0.00%
200	0.8	*0.7848*	0.8	0.0	0.00%
300	1.2	*1.1772*	1.2	0.0	0.00%
400	1.7	*1.5696*	1.6	-0.1	-6.37%
500	2.1	*1.962*	2.0	-0.1	-5.10%
600	2.6	*2.3544*	2.4	**-0.2**	-8.49%
700	3.0	*2.7468*	2.7	**-0.3**	-10.92%
800	3.6	*3.1392*	3.1	**-0.5**	-15.93%
900	4.0	*3.5316*	3.5	**-0.5**	-14.16%
1000	4.3	*3.924*	3.9	**-0.4**	-10.19%
1100	5.0	*4.3164*	4.3	**-0.7**	-16.22%
1200	5.3	*4.7088*	4.7	**-0.6**	-12.74%
1300	6.0	*5.1012*	5.1	**-0.9**	-17.64%
1400	6.2	*5.4936*	5.5	**-0.7**	-12.74%
1500	6.8	*5.886*	5.9	**-0.9**	-15.29%
1600	7.4	*6.2784*	6.3	**-1.1**	-17.52%
1700	7.8	*6.6708*	6.7	**-1.1**	-16.49%
1800	8.0	*7.0632*	7.1	**-0.9**	-12.74%
1900	8.2	*7.4556*	7.5	**-0.7**	-9.39%
2000	8.6	*7.848*	7.8	**-0.8**	-10.19%

Πίνακας 11. Μετρήσεις ροπής Mx και αποκλίσεις από την πραγματική τιμή.

Μετρήσεις My				***Σφάλμα***	
βάρος (gr)	**My(N*m/10)**	*Πραγματική ροπή(N*m/10)*		eMy (N*m/10)	%
-2000	-6.8	*-7.848*	-7.8	**-1.0**	12.74%
-1900	-6.7	*-7.4556*	-7.5	**-0.8**	10.73%
-1800	-6.1	*-7.0632*	-7.1	**-1.0**	14.16%
-1700	-6.0	*-6.6708*	-6.7	**-0.7**	10.49%
-1600	-5.7	*-6.2784*	-6.3	**-0.6**	9.56%
-1500	-5.2	*-5.886*	-5.9	**-0.7**	11.89%
-1400	-5.0	*-5.4936*	-5.5	**-0.5**	9.10%
-1300	-4.8	*-5.1012*	-5.1	**-0.3**	5.88%
-1200	-4.3	*-4.7088*	-4.7	**-0.4**	8.49%
-1100	-4.0	*-4.3164*	-4.3	**-0.3**	6.95%
-1000	-3.7	*-3.924*	-3.9	**-0.2**	5.10%
-900	-3.3	*-3.5316*	-3.5	**-0.2**	5.66%
-800	-3.0	*-3.1392*	-3.1	-0.1	3.19%
-700	-2.5	*-2.7468*	-2.7	**-0.2**	7.28%
-600	-2.1	*-2.3544*	-2.4	**-0.3**	12.74%
-500	-1.8	*-1.962*	-2.0	**-0.2**	10.19%
-400	-1.5	*-1.5696*	-1.6	-0.1	6.37%
-300	-1.1	*-1.1772*	-1.2	-0.1	8.49%
-200	-0.8	*-0.7848*	-0.8	0.0	0.00%
-100	-0.4	*-0.3924*	-0.4	0.0	0.00%
-50	-0.2	*-0.1962*	-0.2	0.0	0.00%
0	0.0	*0*	0.0	0.0	0.00%
50	0.2	*0.1962*	0.2	0.0	0.00%
100	0.4	*0.3924*	0.4	0.0	0.00%
200	0.8	*0.7848*	0.8	0.0	0.00%
300	1.2	*1.1772*	1.2	0.0	0.00%
400	1.7	*1.5696*	1.6	-0.1	-6.37%
500	2.0	*1.962*	2.0	0.0	0.00%
600	2.5	*2.3544*	2.4	-0.1	-4.25%
700	3.0	*2.7468*	2.7	**-0.3**	-10.92%
800	3.3	*3.1392*	3.1	**-0.2**	-6.37%
900	3.7	*3.5316*	3.5	**-0.2**	-5.66%
1000	4.1	*3.924*	3.9	**-0.2**	-5.10%
1100	4.5	*4.3164*	4.3	**-0.2**	-4.63%
1200	5.0	*4.7088*	4.7	**-0.3**	-6.37%
1300	5.2	*5.1012*	5.1	-0.1	-1.96%
1400	5.9	*5.4936*	5.5	**-0.4**	-7.28%
1500	6.1	*5.886*	5.9	**-0.2**	-3.40%
1600	6.7	*6.2784*	6.3	**-0.4**	-6.37%
1700	7.4	*6.6708*	6.7	**-0.7**	-10.49%
1800	7.8	*7.0632*	7.1	**-0.7**	-9.91%
1900	8.0	*7.4556*	7.5	**-0.5**	-6.71%
2000	8.2	*7.848*	7.8	**-0.4**	-5.10%

Πίνακας 12. Μετρήσεις ροπής My και αποκλίσεις από την πραγματική τιμή.

Μετρήσεις Mz				***Σφάλμα***	
βάρος (gr)	**Mz(N*m/10)**	*Πραγματική ροπή(N*m/10)*		eMz (N*m/10)	*%*
-2000	-7.0	*-6.867*	-6.9	0.1	-1.46%
-1900	-6.6	*-6.52365*	-6.5	0.1	-1.53%
-1800	-6.2	*-6.1803*	-6.2	0.0	0.00%
-1700	-6.0	*-5.83695*	-5.8	**0.2**	-3.43%
-1600	-5.5	*-5.4936*	-5.5	0.0	0.00%
-1500	-5.1	*-5.15025*	-5.2	-0.1	1.94%
-1400	-4.9	*-4.8069*	-4.8	0.1	-2.08%
-1300	-4.3	*-4.46355*	-4.5	**-0.2**	4.48%
-1200	-4.0	*-4.1202*	-4.1	-0.1	2.43%
-1100	-3.7	*-3.77685*	-3.8	-0.1	2.65%
-1000	-3.4	*-3.4335*	-3.4	0.0	0.00%
-900	-3.0	*-3.09015*	-3.1	-0.1	3.24%
-800	-2.6	*-2.7468*	-2.7	-0.1	3.64%
-700	-2.4	*-2.40345*	-2.4	0.0	0.00%
-600	-2.0	*-2.0601*	-2.1	-0.1	4.85%
-500	-1.7	*-1.71675*	-1.7	0.0	0.00%
-400	-1.3	*-1.3734*	-1.4	-0.1	7.28%
-300	-1.0	*-1.03005*	-1.0	0.0	0.00%
-200	-0.7	*-0.6867*	-0.7	0.0	0.00%
-100	-0.3	*-0.34335*	-0.3	0.0	0.00%
-50	-0.2	*-0.171675*	-0.2	0.0	0.00%
0	0.0	*0*	0.0	0.0	0.00%
50	0.2	*0.171675*	0.2	0.0	0.00%
100	0.3	*0.34335*	0.3	0.0	0.00%
200	0.7	*0.6867*	0.7	0.0	0.00%
300	1.0	*1.03005*	1.0	0.0	0.00%
400	1.3	*1.3734*	1.4	0.1	7.28%
500	1.7	*1.71675*	1.7	0.0	0.00%
600	2.0	*2.0601*	2.1	0.1	4.85%
700	2.5	*2.40345*	2.4	-0.1	-4.16%
800	2.9	*2.7468*	2.7	**-0.2**	-7.28%
900	3.0	*3.09015*	3.1	0.1	3.24%
1000	3.4	*3.4335*	3.4	0.0	0.00%
1100	3.7	*3.77685*	3.8	0.1	2.65%
1200	4.0	*4.1202*	4.1	0.1	2.43%
1300	4.3	*4.46355*	4.5	**0.2**	4.48%
1400	4.9	*4.8069*	4.8	-0.1	-2.08%
1500	5.0	*5.15025*	5.2	**0.2**	3.88%
1600	5.5	*5.4936*	5.5	0.0	0.00%
1700	5.8	*5.83695*	5.8	0.0	0.00%
1800	6.0	*6.1803*	6.2	**0.2**	3.24%
1900	6.2	*6.52365*	6.5	**0.3**	4.60%
2000	6.8	*6.867*	6.9	0.1	1.46%

Πίνακας 13. Μετρήσεις ροπής Mz και αποκλίσεις από την πραγματική τιμή.

Α.3 Ενδεικτική εφαρμογή τμηματικών συνεχών πολυωνύμων σε συμπληρωματικά πεδία ορισμού για τις φορτίσεις Fx και Fz

3.1 Τμηματικά συνεχή πολυώνυμα για την Fx (N)

με $\boldsymbol{Fx_{new} = Fx + \delta Fx}$

(όπου *δFx* ο συντελεστής διόρθωσης εξαρτημένος από την Fx)

πεδίο ορισμού	**συνάρτηση**
[-16,0 , -15,6)	$Fx_{new} = Fx + (1.5Fx + 23.3)$
[-15,6 , -12,1)	$Fx_{new} = Fx + (-0.0181(Fx)^2 - 0.666Fx - 6.0936)$
[-12,1 , -11,6)	$Fx_{new} = Fx + (Fx + 11.4)$
[-11,6 , -9,8)	$Fx_{new} = Fx + (0.463(Fx)^2 + 10.019Fx + 53.719)$
[-9,8 , -4,9)	$Fx_{new} = Fx + (0.0461(Fx)^2 + 0.6878Fx + 2.2804)$
[-4,9 , 4,9)	$Fx_{new} = Fx$
[4,9 , 7,8)	$Fx_{new} = Fx + (-0.1106(Fx)^2 + 1.4207Fx - 4.3327)$
[7,8 , 9,8)	$Fx_{new} = Fx + (-0.3297(Fx)^2 + 5.8022Fx - 25.2)$
[9,8, 10,4)	$Fx_{new} = Fx + (0.6667Fx - 6.5333)$
[10,4 , 12,1)	$Fx_{new} = Fx + (0.6863(Fx)^2 - 15.265Fx + 84.925)$
[12,1 , 15,6)	$Fx_{new} = Fx + (0.0339(Fx)^2 - 1.1097Fx + 9.167)$
[15,6 , 16,0)	$Fx_{new} = Fx + (1.5Fx - 23.3)$
[16,0 , 18,0]	$Fx_{new} = Fx + (-0.3214Fx + 5.8429)$

Πίνακας 14. Βαθμονόμηση Fx με χρήση τμηματικών συνεχών πολυωνύμων.

3.2 Τμηματικά συνεχή πολυώνυμα για την Fy (N)

με $Fy_{new} = Fy + \delta Fy$

(όπου *δFy* ο συντελεστής διόρθωσης εξαρτημένος από την Fy)

πεδίο ορισμού	**συνάρτηση**
[-16,3 , -15,9)	$Fy_{new} = Fy + (1.5Fy + 24.05)$
[-15,9 , -12,4)	$Fy_{new} = Fy + (-0.0181(Fy)^2 - 0.6769Fy - 5.995)$
[-12,4 , -11,9)	$Fy_{new} = Fy + (Fy + 12)$
[-11,9 , -9,9)	$Fy_{new} = Fy + (0.4762(Fy)^2 + 10.381Fy + 56.2)$
[-9,9 , -5,0)	$Fy_{new} = Fy + (0.0461(Fy)^2 + 0.6971Fy + 2.4497)$
[-5,0 , 3,9)	$Fy_{new} = Fy$
[3,9 , 5,8)	$Fy_{new} = Fy + (-0.089(Fy)^2 + 0.8821Fy - 2.0326)$
[5,8 , 8,0)	$Fy_{new} = Fy + (0.113(Fy)^2 - 1.2673Fy + 3.6702)$
[8,0, 9,9)	$Fy_{new} = Fy + (-0.4737Fy + 4.5895)$
[9,9 , 11,6)	$Fy_{new} = Fy + (-0.6863(Fy)^2 + 14.931Fy - 80.659)$
[11,6 , 12,1)	$Fy_{new} = Fy + (Fy - 11.4)$
[12,1 , 15,6)	$Fy_{new} = Fy + (-0.1709Fy + 2.7622)$
[15,6 , 16,0]	$Fy_{new} = Fy + (1.5Fy - 23.3)$
[16,0 , 19,0]	$Fy_{new} = Fy + (-0.3667Fy + 6.5667)$

Πίνακας 15. Βαθμονόμηση Fy με χρήση τμηματικών συνεχών πολυωνύμων.

Α.4 Ρουτίνες υπολογισμού συναρτήσεων κυβικών splines

4.1 **Spline.c** - ρουτίνα εύρεσης συντελεστών M_i

spline.c

```
#include <stdio.h>
#include <stdlib.h>
#include "nrutil.h"
#include "nrutil.c"


int main(void)

{
int i,k,l,n= -N- ;
float p,qn,sig,un,*u,yp1=0,ypn=0,y2[-N-];
float x[-N-]={ x[1],x[2].....x[N] };
float y[-N-]={ y[1],y[2].....y[N] };
 FILE* fp;


///////////////Splines Calculation//////////////////////

u=vector(1,n-1);

if (yp1>0.99e30)
    y2[1]=u[1]=0.0;
else {
     y2[1]=-0.5;
     u[1]=(3.0/(x[2]-x[1]))*((y[2]-y[1])/(x[2]-x[1])-yp1);
}

for (i=2;i<=n-1;i++){
     sig=(x[i]-x[i-1])/(x[i+1]-x[i-1]);
     p=sig*y2[i-1]+2.0;
     y2[i]=(sig-1.0)/p;
     u[i]=(y[i+1]-y[i])/(x[i+1]-x[i])-(y[i]-y[i-1])/(x[i]-x[i-1]);
     u[i]=(6.0*u[i]/(x[i+1]-x[i-1])-sig*u[i-1])/p;
}

if (ypn>0.99e30)
    qn=un=0.0;

else{
     qn=0.5;
     un=(3.0/(x[n]-x[n-1]))*(ypn-(y[n]-y[n-1])/(x[n]-x[n-1]));
}

y2[n]=(un-qn*u[n-1])/(qn*y2[n-1]+1.0);
for (k=n-1;k>=1;k--)
    y2[k]=y2[k]*y2[k+1]+u[k];
free_vector(u,1,n-1);
```

```
/////////////////Data Saving////////////////////

if ((fp = fopen("__y2.dat","w"))==NULL){
     printf("Cannot open file\n");
}


for (l=1;l<n+1;l++) {
     fprintf(fp,"%3.3f,\n",y2[l]);
}

fclose(fp);

}
```

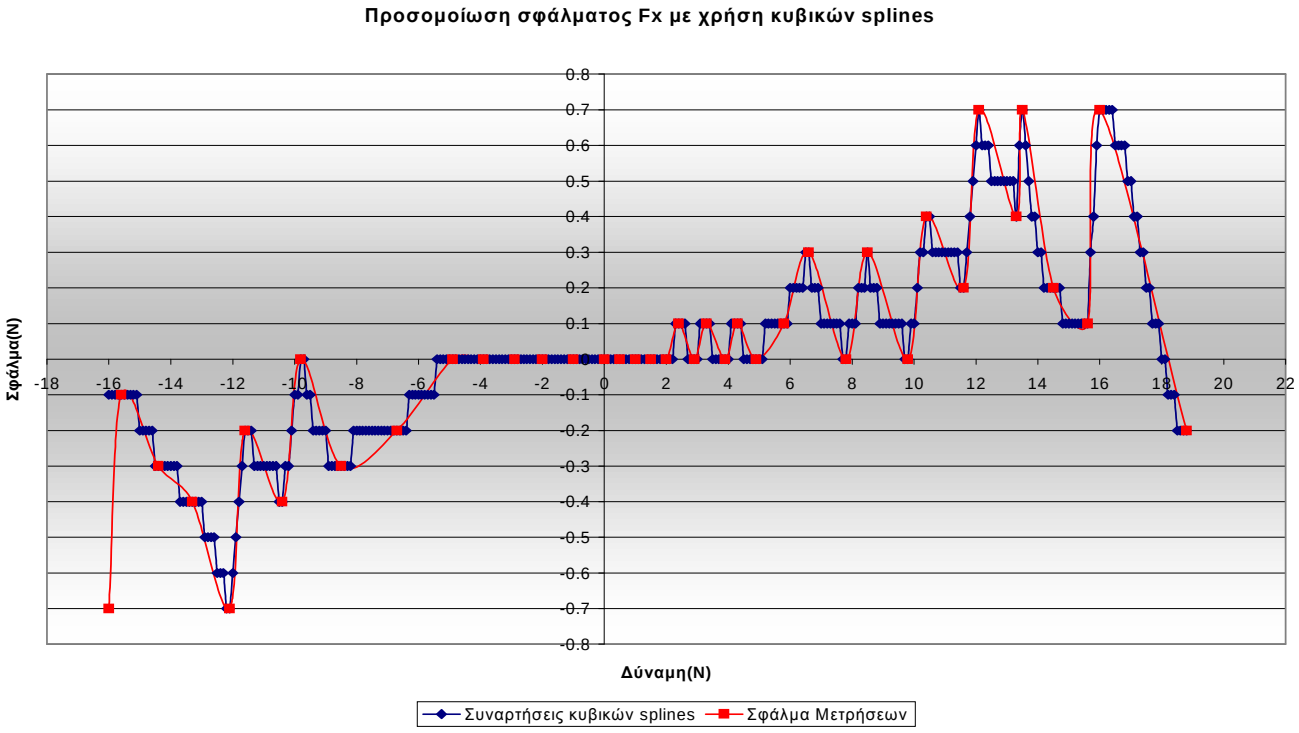
4.2 **Splint.c** - ρουτίνα υπολογισμού y για κάθε τιμή του x

splint.c

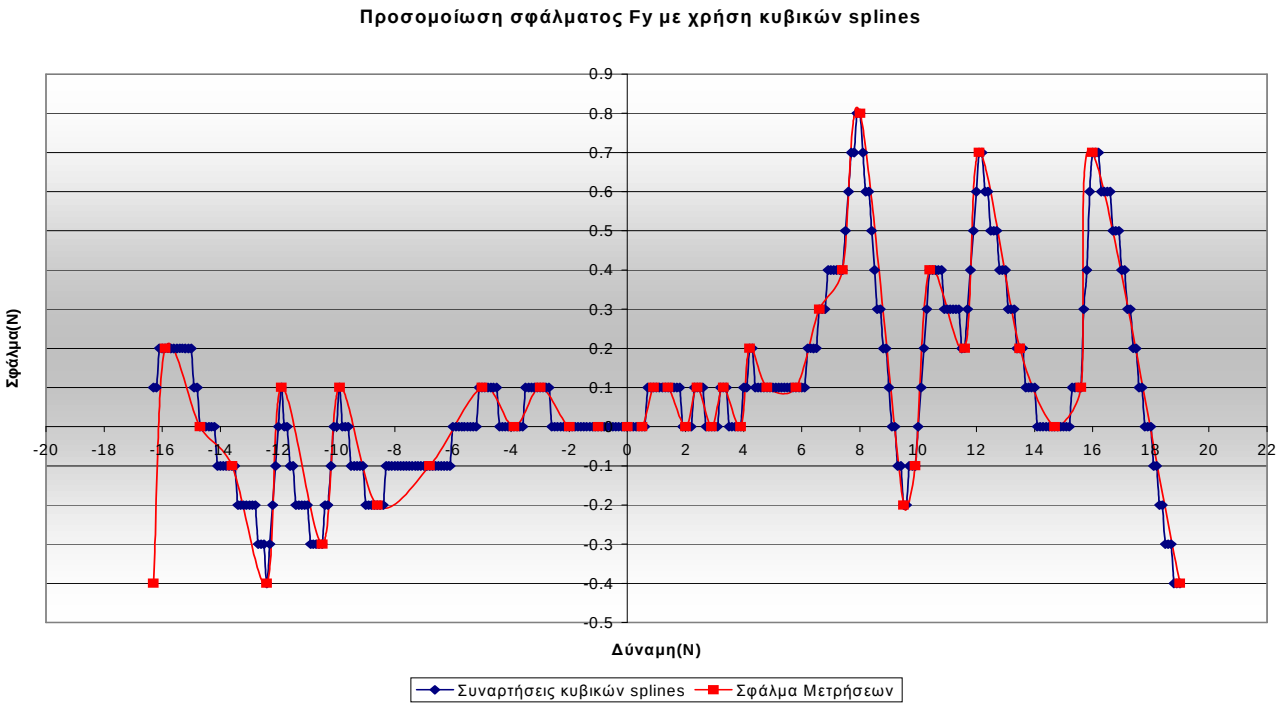
```
#include <stdio.h>
#include <stdlib.h>
#include "nrutil.h"
#include "nrutil.c"

int main(void)

{
int n= -N- ,klo,khi,k;

float xa[-N-]={ x[1],x[2].....x[N] };
float ya[-N-]={ y[1],y[2].....y[N] };
float y2a[-N-]={ y2[1],y2[2].....y2[N] };
float x;
float y;

float h,b,a;
FILE* fp;

if ((fp = fopen("__SplintCheck.dat","w"))==NULL){
     printf("Cannot open file\n");
}

{
void nrerror(char error_text[]);

for (x=-16;x<18.9;x=x+0.1) {

    klo=1;
    khi=n;

    while(khi-klo>1){
    k=(khi+klo)>>1;
    if(xa[k]>x)khi=k;
    else klo=k;
}

h=xa[khi]-xa[klo];
if (h==0) nrerror ("Bad input to routine");
a=(xa[khi]-x)/h;
b=(x-xa[klo])/h;
y=a*ya[klo]+b*ya[khi]+((a*a*a-a)*y2a[klo]+(b*b*b-b)*y2a[khi])*
(h*h)/6.0;

fprintf(fp,"%3.1f  ",x);
fprintf(fp,"%3.1f\n",y);
}

}
}
```

Α.5 Διαγράμματα προσομοίωσης σφάλματος με χρήση συναρτήσεων κυβικών splines

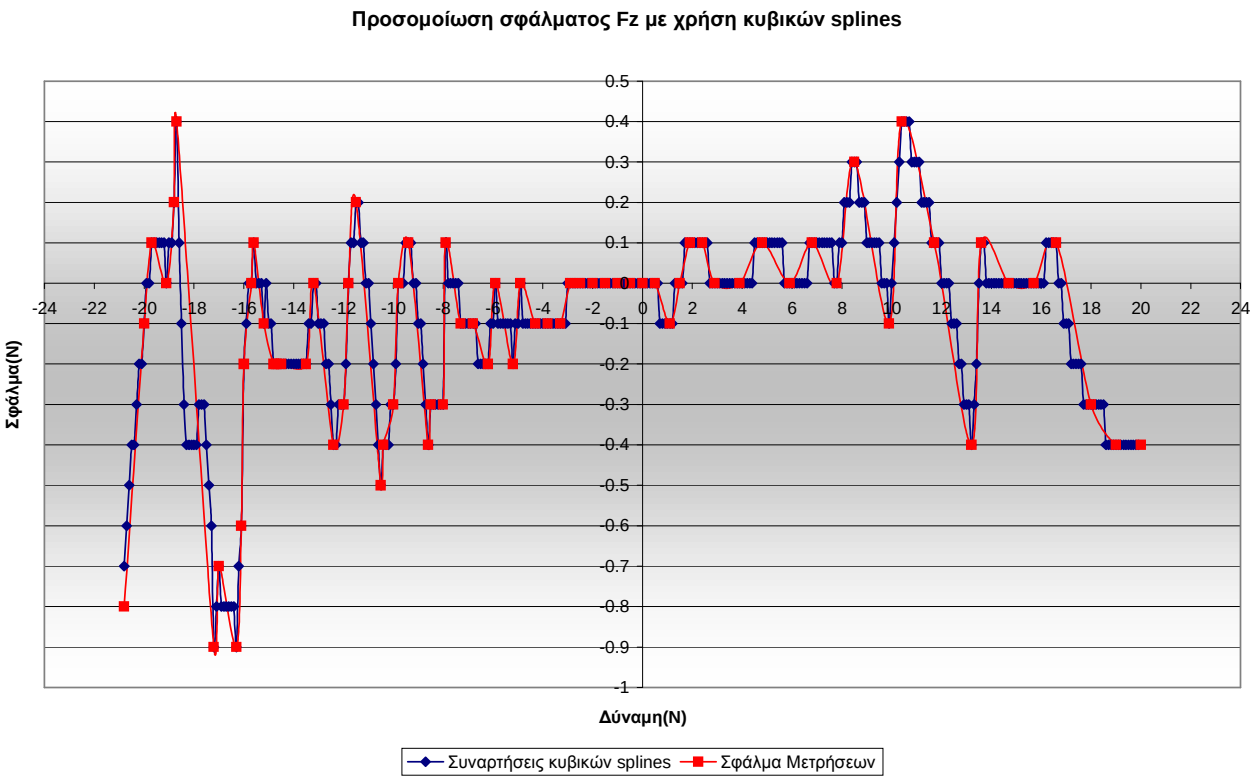
5.1 Fx, Fy, Fz Δυνάμεις



Διάγραμμα 27. Προσομοίωση σφάλματος Fx με χρήση κυβικών splines.

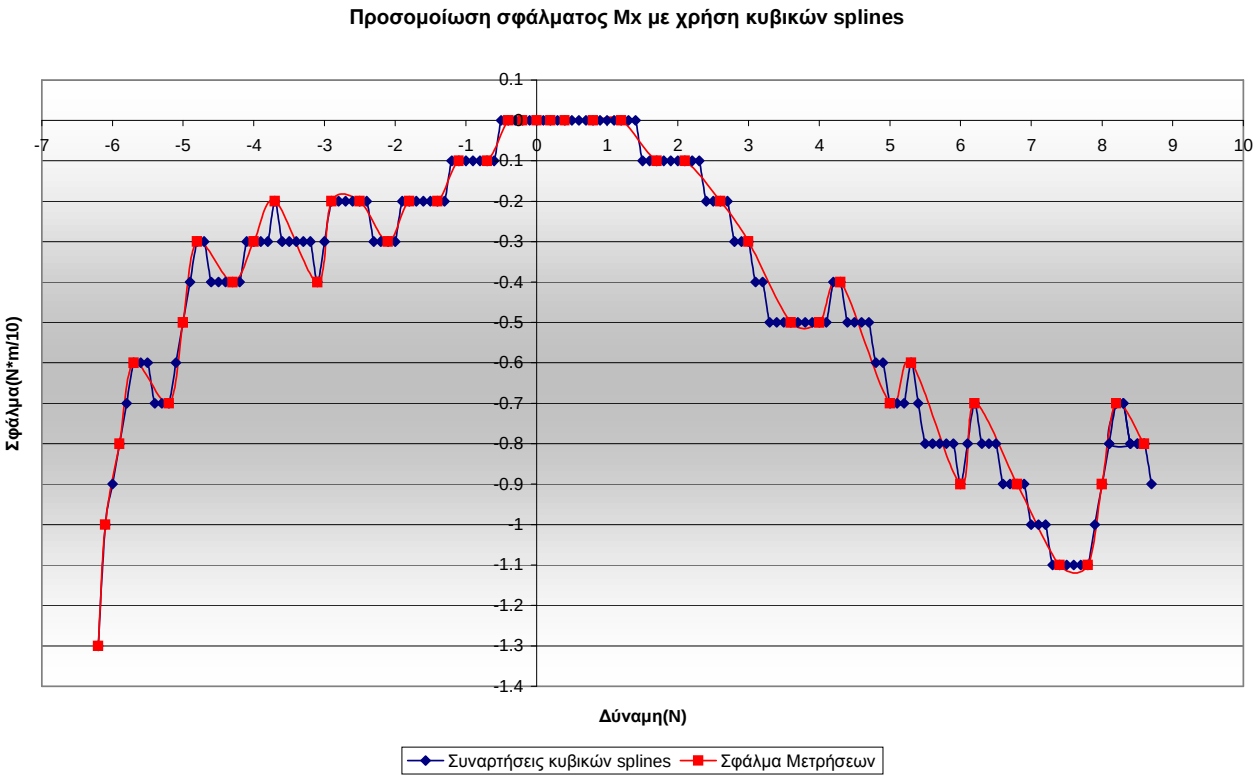


Διάγραμμα 28. Προσομοίωση σφάλματος Fy με χρήση κυβικών splines.

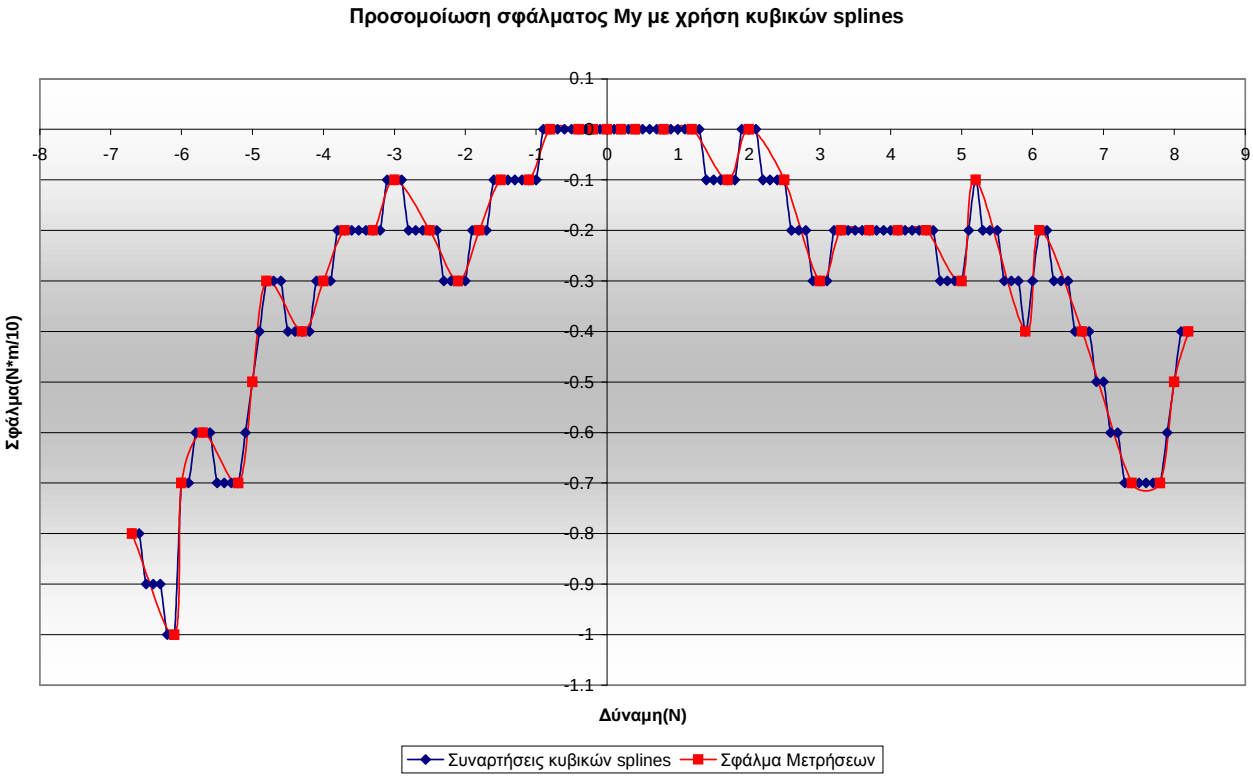


Διάγραμμα 29. Προσομοίωση σφάλματος Fz με χρήση κυβικών splines.

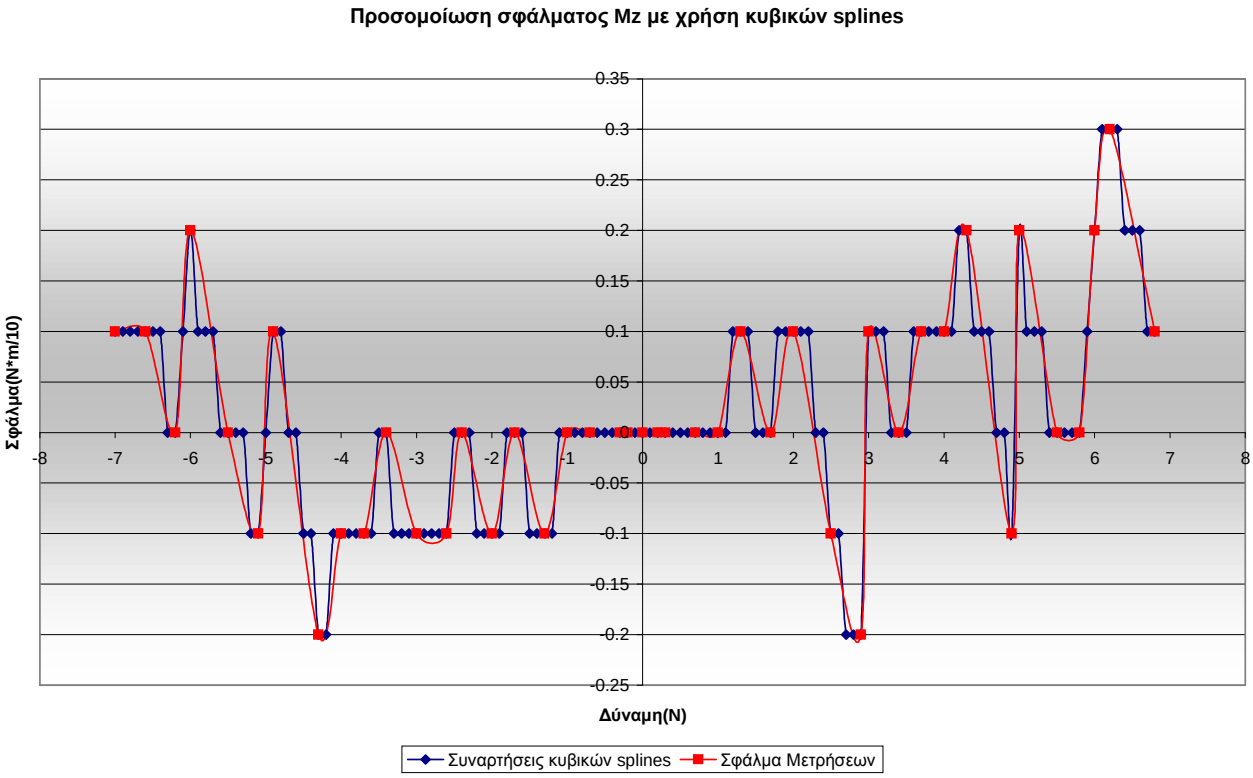
5.2 Mx, My, Mz Ροπές



Διάγραμμα 30. Προσομοίωση σφάλματος Mx με χρήση κυβικών splines.



Διάγραμμα 31. Προσομοίωση σφάλματος My με χρήση κυβικών splines.

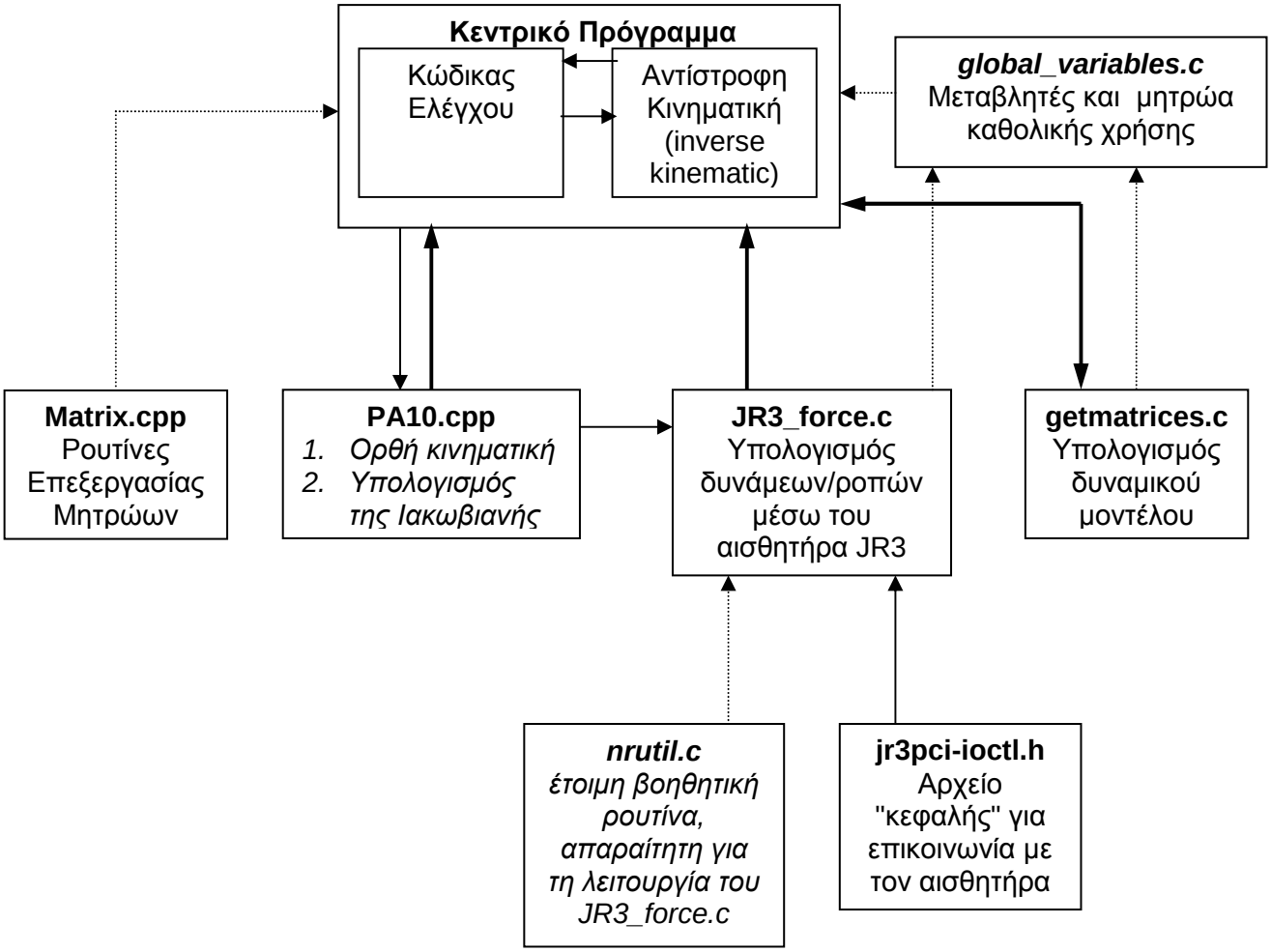


Διάγραμμα 32. Προσομοίωση σφάλματος Mz με χρήση κυβικών splines.

Β. Περιγραφή και Ανάλυση λογισμικού

Β.1 Δομή λογισμικού

Παρουσιάζεται το πλήρες διάγραμμα συσχετίσεων του κεντρικού κώδικα ForceControl.cpp με τις επιμέρους υπορουτίνες, απαραίτητες για την επίτευξη ελέγχου δύναμης/ροπής, που βρίσκονται στον υποφάκελο του ελέγχου.



όπου με → περιγράφεται η κύρια κατεύθυνση εισροής πληροφοριών μεταξύ των διασυνδεόμενων υπορουτινών. Η διακεκομμένη γραμμή αναφέρεται σε βοηθητικές και έμμεσες λειτουργίες του συστήματος.

Οι διαφορές στη δομή του κεντρικού κώδικα ελέγχου Fcontrol2dof.cpp είναι ελάχιστες και αφορούν την υπορουτίνα PA10.cpp. Στην περίπτωση αυτή το αρχείο είναι σε γλώσσα προγραμματισμού C (PA10.c) και υπολογίζει και την αντίστροφη κινηματική του συστήματος, κάτι που δεν γίνεται πλέον στον κεντρικό κώδικα.

Β.2 Ανάλυση κώδικα

2.1 JR3_force.c

```
#include <math.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/ioctl.h>
#include "jr3pci-ioctl.h"
#include "nrutil.h"
#include "nrutil.c"
#include "JR3_force.h"
```

Σύνδεση με τις απαραίτητες βιβλιοθήκες και υποπρογράμματα για την ορθή λειτουργία του κώδικα. Τα " " αναφέρονται σε αρχεία που πρέπει να βρίσκονται στον ίδιο φάκελο με το τρέχον πρόγραμμα. Μεταξύ αυτών το αρχείο jr3pci-ioctl.h που καθορίζει την επικοινωνία με τον αισθητήρα μέσω της κάρτας του και τη λήψη των τιμών των φορτίσεων.

```
float y;
six_axis_array fm,ac;
force_array fs, as;
int ret,fd,ka,sensors=0;
    short f, b;
    float temp,val[6]={0,0,0,0,0,0};
    int i,j,col,calibr[6]={0,0,0,0,0,0};

    int n,l;
```

Ορισμός και αρχικοποίηση μεταβλητών για τον αισθητήρα, τον υπολογισμό των splines και των βασικών μεταβλητών του συστήματος val (όπου προσωρινά αποθηκεύονται οι τελικές τιμές των δυνάμεων και των ροπών) και calibr (που καθορίζει αν το αντίστοιχο στοιχείο του μητρώου των φορτίσεων είναι εκτός ορίων βαθμονόμησης (τιμή 1) ή εντός αυτών (τιμή 0)).

```
    float r[3][3];                ///////Euler transformation Matrix
    float init[3];       //////Force Vector with (0,0,0) orientation
    float offset[3];     /////Force Vector with starting orientation
```

Ορισμός μητρώων και διανυσμάτων απαραίτητων για τη διαδικασία της μηδενικής αντιστάθμισης.

```
    float direction[3]={0,-1,0};     ///////Gravity direction (= -y)
```

Με αυτό το διάνυσμα εισάγουμε στο σύστημα την ανάλυση του μοναδιαίου διανύσματος της βαρύτητας στους τρεις άξονες μέτρησης δυνάμεων του αισθητήρα (x, y, z), όταν το σύστημά μας βρίσκεται στη μηδενική θέση λειτουργίας (π.χ. σε ένα ρομποτικό σύστημα, στη θέση όπου όλα τα q=0). Για έλεγχο υπολογίζουμε το μέτρο αυτού του διανύσματος, που πρέπει να είναι πάντα ίσο με τη μονάδα. Για παράδειγμα, όταν στη μηδενική θέση ταυτίζονται το διάνυσμα της βαρύτητας με τον άξονα y του αισθητήρα, η τιμή που πρέπει να δώσουμε στο διάνυσμα direction είναι η {0,1,0}.

```
float df[6];                                ///////Correction Force
float jr3_own_gravity_force=0.51,calibr_off=1.38;
float extra_offset_force=0.87;               ///////User defined
```

Η τιμή του `jr3_own_gravity_force` καθορίζει τη δύναμη που ασκείται από το βάρος της ίδιας της συσκευής του αισθητήρα, όπως αυτή υπολογίστηκε, και μας χρησιμεύει στον υπολογισμό της επίδρασης του ίδιου βάρους της συσκευής, στη μέτρησή μας κατά την περιστροφή της στο χώρο. Αντίστοιχα στη μεταβλητή extra_offset_force εισάγουμε την αντίστοιχη δύναμη του βάρους ενός πιθανού εξαρτήματος που έχουμε συνδέσει με τον αισθητήρα. Η τιμή του `calibr_off` αναφέρεται στο συνολικό βάρος αισθητήρα και εξαρτημάτων κατά τη διαδικασία της βαθμονόμησης και χρησιμεύει για την αναγωγή των μετρούμενων δυνάμεων στις συνθήκες βαθμονόμησης, έτσι ώστε να επιτύχουμε την ορθή χρήση των καμπύλων splines.

```
float thresh=0.2;  //////if ABS|Force|<thresh values are ignored

double sin( double arg );
double cos( double arg );
```

Με τη μεταβλητή `thresh` καθορίζουμε το κατώφλι (threshold), κάτω από το οποίο δεν θα λαμβάνονται υπόψη οι τιμές των φορτίσεων, για την αποφυγή εισροής θορύβου στο σύστημα, όταν αυτό βρίσκεται σε σταθερή κατάσταση.

```
/////// Spline Arrays for Force calculation

    float FXxa[40]={-16.0,-15.6,-14.4,-13.3,-12.1,-11.6,-10.4,-9.8,-
8.5,-6.7,-4.9,-3.9,-2.9,-2.0,-
1.0,0.0,0.5,1.0,1.5,2.0,2.4,2.9,3.3,3.9,4.3,4.9,5.8,6.6,7.8,8.5,9.8,
10.4,11.6,12.1,13.3,13.5,14.5,15.6,16.0,18.8};
    float FXya[40]={-16.7,-15.7,-14.7,-13.7,-12.8,-11.8,-10.8,-9.8,-
8.8,-6.9,-4.9,-3.9,-2.9,-2.0,-
1.0,0.0,0.5,1.0,1.5,2.0,2.5,2.9,3.4,3.9,4.4,4.9,5.9,6.9,7.8,8.8,9.8,
10.8,11.8,12.8,13.7,14.2,14.7,15.7,16.7,18.6};
    float FXy2a[40]={2.209,-0.251,-0.246,0.061,-0.542,0.926,-
1.140,0.761,-0.123,-0.083,0.022,-0.006,0.002,-0.002,0.005,-
0.029,0.110,-0.409,1.528,-2.614,2.788,-2.529,2.406,-
1.986,0.848,0.349,-1.394,1.915,-1.589,1.660,-1.133,1.002,-
1.014,2.331,-2.052,1.258,0.929,-0.564,-0.747,0.667};

    float FYxa[41]={-16.3,-15.9,-14.7,-13.6,-12.4,-11.9,-10.5,-9.9,-
8.6,-6.8,-5.0,-3.9,-3.0,-2.0,-
1.0,0.0,0.5,0.9,1.4,2.0,2.4,2.9,3.3,3.9,4.2,4.8,5.8,6.6,7.4,8.0,9.5,
9.9,10.4,11.6,12.1,13.5,14.2,14.7,15.6,16.0,19.0};
    float FYya[41]={-16.7,-15.7,-14.7,-13.7,-12.8,-11.8,-10.8,-9.8,-
8.8,-6.9,-4.9,-3.9,-2.9,-2.0,-
1.0,0.0,0.5,1.0,1.5,2.0,2.5,2.9,3.4,3.9,4.4,4.9,5.9,6.9,7.8,8.8,9.3,
9.8,10.8,11.8,12.8,13.7,14.2,14.7,15.7,16.7,18.6};
    float FYy2a[41]={-2.000,-0.364,-1.108,1.118,-1.542,0.990,-
0.145,0.747,-0.070,-0.282,0.499,-0.528,0.290,-0.030,-0.168,0.067,-
0.842,-0.823,2.054,-2.785,2.983,-3.191,4.480,-3.832,0.923,0.345,-
0.832,2.047,-3.026,0.319,0.061,-0.075,0.571,-1.020,0.100,0.055,-
0.116,0.849,-0.677,-0.446,0.378};

    float FZxa[72]={-20.8,-20.0,-19.7,-19.1,-18.8,-18.7,-17.2,-
17.0,-16.3,-16.1,-16.0,-15.7,-15.6,-15.2,-14.8,-14.7,-14.5,-13.5,-
13.2,-12.4,-12.0,-11.8,-11.5,-10.5,-10.4,-10.0,-9.8,-9.4,-8.6,-8.5,-
8.0,-7.9,-7.3,-6.8,-6.2,-5.9,-5.2,-4.9,-4.3,-3.8,-3.3,-2.9,-2.5,-
2.0,-1.5,-1.0,-
0.5,0.0,0.5,1.1,1.5,1.9,2.4,2.9,3.9,4.8,5.9,6.8,7.8,8.5,9.9,10.4,11.
7,13.2,13.6,14.7,15.7,16.6,18.0,19.0,20.0};
    float FZya[72]={-21.6,-20.1,-19.6,-19.1,-18.6,-18.3,-18.1,-
17.7,-17.2,-16.7,-16.2,-15.7,-15.5,-15.3,-15.0,-14.9,-14.7,-13.7,-
13.2,-12.8,-12.3,-11.8,-11.3,-11.0,-10.8,-10.3,-9.8,-9.3,-9.0,-8.8,-
8.3,-7.8,-7.4,-6.9,-6.4,-5.9,-5.4,-4.9,-4.4,-3.9,-3.4,-2.9,-2.5,-
2.0,-1.5,-1.0,-
0.5,0.0,0.5,1.0,1.5,2.0,2.5,2.9,3.9,4.9,5.9,6.9,7.8,8.8,9.8,10.8,11.
8,12.8,13.7,14.7,15.7,16.7,17.7,18.6,19.6};
    float FZy2a[72]={-0.117,-0.066,-0.066,-3.508,2.224,10.340,-
9.385,7.866,-7.331,5.583,28.984,-35.071,17.195,-12.345,4.065,-
0.164,0.377,-1.050,2.443,-4.344,2.281,2.254,-0.588,-1.542,1.586,-
1.472,1.533,-0.553,-1.757,4.085,-4.969,4.509,-4.265,4.368,-
3.700,4.927,-5.494,5.422,-4.282,1.802,-0.789,1.354,-1.358,0.326,-
0.087,0.023,0.006,0.000,1.282,-0.159,-0.647,-0.545,0.428,0.187,-
0.526,0.658,-0.936,1.697,-2.375,1.216,-1.366,-0.367,0.009,-
0.457,1.021,0.262,-0.654,0.608,0.073,-0.098,0.196};

    float MXxa[43]={-6.8,-6.2,-6.1,-5.9,-5.7,-5.2,-5.0,-4.8,-4.3,-
4.0,-3.7,-3.1,-2.9,-2.5,-2.1,-1.8,-1.4,-1.1,-0.7,-0.4,-
0.2,0.0,0.2,0.4,0.8,1.2,1.7,2.1,2.6,3.0,3.6,4.0,4.3,5.0,5.3,6.0,6.2,
6.8,7.4,7.8,8.0,8.2,8.6};
```

```
    float MXya[43]={-7.8,-7.5,-7.1,-6.7,-6.3,-5.9,-5.5,-5.1,-4.7,-
4.3,-3.9,-3.5,-3.1,-2.7,-2.4,-2.0,-1.6,-1.2,-0.8,-0.4,-
0.2,0.0,0.2,0.4,0.8,1.2,1.6,2.0,2.4,2.7,3.1,3.5,3.9,4.3,4.7,5.1,5.5,
5.9,6.3,6.7,7.1,7.5,7.8};
    float MXy2a[43]={0.000,3.700,-0.502,-1.340,0.552,-0.514,-
0.496,0.993,0.196,-0.777,7.567,-6.200,-0.183,3.181,-2.934,2.883,-
2.875,2.900,-3.033,0.814,-0.225,0.084,-0.112,0.293,-1.060,1.181,-
0.991,0.223,-0.513,0.729,2.124,-4.219,4.614,-5.675,7.707,-
6.639,1.803,-0.571,1.152,0.228,-6.064,-0.671,0.730};

    float MYxa[43]={-6.8,-6.7,-6.1,-6.0,-5.7,-5.2,-5.0,-4.8,-4.3,-
4.0,-3.7,-3.3,-3.0,-2.5,-2.1,-1.8,-1.5,-1.1,-0.8,-0.4,-
0.2,0.0,0.2,0.4,0.8,1.2,1.7,2.0,2.5,3.0,3.3,3.7,4.1,4.5,5.0,5.2,5.9,
6.1,6.7,7.4,7.8,8.0,8.2};
    float MYya[43]={-7.8,-7.5,-7.1,-6.7,-6.3,-5.9,-5.5,-5.1,-4.7,-
4.3,-3.9,-3.5,-3.1,-2.7,-2.4,-2.0,-1.6,-1.2,-0.8,-0.4,-
0.2,0.0,0.2,0.4,0.8,1.2,1.6,2.0,2.4,2.7,3.1,3.5,3.9,4.3,4.7,5.1,5.5,
5.9,6.3,6.7,7.1,7.5,7.8};
    float MYy2a[43]={0.000,1.874,-2.230,0.655,0.842,0.467,-
6.709,0.198,-0.540,-2.038,2.538,-2.461,-0.046,2.535,-0.104,-
2.121,2.502,-2.182,0.761,-0.201,0.044,0.026,-0.148,0.430,-
1.572,2.916,-2.266,-0.899,3.460,-2.290,0.419,0.615,-2.879,2.472,-
1.109,2.044,-1.660,0.411,-0.349,0.878,-4.571,-2.593,2.184};

    float MZxa[43]={-7.0,-6.6,-6.2,-6.0,-5.5,-5.1,-4.9,-4.3,-4.0,-
3.7,-3.4,-3.0,-2.6,-2.4,-2.0,-1.7,-1.3,-1.0,-0.7,-0.3,-
0.2,0.0,0.2,0.3,0.7,1.0,1.3,1.7,2.0,2.5,2.9,3.0,3.4,3.7,4.0,4.3,4.9,
5.0,5.5,5.8,6.0,6.2,6.8};
    float MZya[43]={-6.9,-6.5,-6.2,-5.8,-5.5,-5.2,-4.8,-4.5,-4.1,-
3.8,-3.4,-3.1,-2.7,-2.4,-2.1,-1.7,-1.4,-1.0,-0.7,-0.3,-
0.2,0.0,0.2,0.3,0.7,1.0,1.4,1.7,2.1,2.4,2.7,3.1,3.4,3.8,4.1,4.5,4.8,
5.2,5.5,5.8,6.2,6.5,6.9};
    float MZy2a[43]={0.000,-0.482,-1.389,0.898,0.448,-0.982,0.469,-
1.185,3.605,-3.570,1.040,3.160,-6.037,5.282,-4.931,4.549,-
2.985,0.724,-0.294,0.049,-0.000,-0.048,0.289,-0.710,2.927,-
4.332,4.217,-2.238,-4.169,1.809,-1.410,1.324,-1.963,0.863,-
0.822,1.201,-1.881,0.474,0.609,-4.254,-2.593,-0.000,0.000};
```

Με τις παραπάνω εντολές εισάγονται στο σύστημα τα απαραίτητα διανύσματα για τον υπολογισμό των καμπυλών splines στη διαδικασία της βαθμονόμησης. Έχουμε ένα πακέτο τριών διανυσμάτων ανά βαθμό ελευθερίας του αισθητήρα, δηλαδή ανά μία από τις έξι φορτίσεις που μετρούνται μέσω αυτού (Fx, Fy, Fz, Mx, My, Mz). Τα δύο πρώτα διανύσματα αναφέρονται στα σημεία (συντεταγμένες x και y αντίστοιχα) που έχουν υπολογιστεί στα διαγράμματα των μετρήσεων και αφορούν ουσιαστικά τις τιμές που μετρήθηκαν μέσω των πρότυπων βαρών και τις αντίστοιχες πραγματικές τιμές. Το τρίτο διάνυσμα, που υπολογίστηκε μέσω του βοηθητικού λογισμικού splint.c, αφορά τις τιμές της δεύτερης παραγώγου στα σημεία αυτά, ώστε να εξασφαλίζεται η συνέχεια της 1ης παραγώγου και η ομαλότητα της καμπύλης παρεμβολής.

```
//// Analysis of gravity force
void force_vector_analysis(){

    init[0]=(jr3_own_gravity_force+extra_offset_force)*direction[0];
    init[1]=(jr3_own_gravity_force+extra_offset_force)*direction[1];
    init[2]=(jr3_own_gravity_force+extra_offset_force)*direction[2];
//////Starting direction
    r[0][0]=cos(beta)*cos(gama);
    r[0][1]=-cos(beta)*sin(gama)*cos(alfa)+sin(beta)*sin(alfa);
    r[0][2]=cos(beta)*sin(gama)*sin(alfa);
    r[1][0]=sin(gama);
    r[1][1]=cos(gama)*cos(alfa);
    r[1][2]=-cos(gama)*sin(alfa);
    r[2][0]=-sin(beta)*cos(gama);
    r[2][1]=sin(beta)*sin(gama)*cos(alfa)+cos(beta)*sin(alfa);
    r[2][2]=-sin(beta)*sin(gama)*sin(alfa)+cos(beta)*cos(alfa);

    df[0]=(r[0][0]*init[0])+(r[0][1]*init[1])+(r[0][2]*init[2]);
    df[1]=(r[1][0]*init[0])+(r[1][1]*init[1])+(r[1][2]*init[2]);
    df[2]=(r[2][0]*init[0])+(r[2][1]*init[1])+(r[2][2]*init[2]);

}
```

Υπορουτίνα υπολογισμού του διανύσματος της βαρύτητας στο χώρο, με σκοπό την αντιστάθμισή του σε κάθε μέτρηση, για την αποφυγή της επίδρασής του σε αυτή. Να σημειωθεί πως κατά την εκκίνηση του αισθητήρα όλες οι τιμές μηδενίζονται, με αποτέλεσμα να αγνοείται το βάρος στην αρχική κατεύθυνση. Για αυτό η ρουτίνα αυτή καλείται και κατά την εκκίνηση, για να υπολογιστούν αυτές οι αρχικές συνιστώσες. Ουσιαστικά σε αυτή τη ρουτίνα υπολογίζουμε το διάνυσμα `df` που προκύπτει από τον πολλαπλασιασμό της μήτρας περιστροφής `r` (με βάση τις γωνίες Euler που υπολογίστηκαν από την ορθή κινηματική) με το διάνυσμα της βαρυτικής δύναμης του αισθητήρα και πιθανών εξαρτημάτων (διάνυσμα `init` που προκύπτει από την τιμή της δύναμης επί το μοναδιαίο διάνυσμα της κατεύθυνσης). Είσοδος του συστήματος είναι οι γωνίες alfa, beta, gama.

```
////////////// Starting JR3 device ///////////////

void JR3_Start_Device(){

    if ((fd=open("/dev/jr3",O_RDWR)) < 0) {
      perror("Can't open device. No way to read force!");
    }
    ioctl(fd,IOCTL0_JR3_ZEROOFFS);
    ret=ioctl(fd,IOCTL0_JR3_GET_FULL_SCALES,&fs);
    if (ret==0) sensors++;
    ret=ioctl(fd,IOCTL1_JR3_GET_FULL_SCALES,&as);
    if (ret==0) sensors++;

    force_vector_analysis();
      for (i=0;i<3;i++) offset[i]=df[i];
}
```

Σε αυτή την υπορουτίνα ενεργοποιούμε τη συσκευή και θέτουμε την εμφάνιση μηνύματος λάθους σε περίπτωση αδυναμίας εκκίνησης της λειτουργίας του. Επίσης, εισάγουμε τις αντισταθμίσεις και τις κλίμακες τιμών που μας παρέχει ο αισθητήρας και καλούμε τη ρουτίνα `force_vector_analysis`, για να αποθηκεύσουμε στο διάνυσμα `offset` τις συνιστώσες της βαρυτικής δύναμης που αγνοήθηκε κατά τη μηδενική αντιστάθμιση στην εκκίνηση του αισθητήρα.

```
//////Cubic Splines routine/////////

void splines (float *xa, float *ya, float *y2a, int ni, float x) {

     int klo,khi,k;
     float h,bi,a,res;
    klo=1;
    khi=ni;
    while(khi-klo>1){
    k=(khi+klo)>>1;
    if(xa[k]>x)khi=k;
    else klo=k;
    }
    h=xa[khi]-xa[klo];
    if (h==0) nrerror ("Bad input to routine");
    a=(xa[khi]-x)/h;
    bi=(x-xa[klo])/h;
    y=a*ya[klo]+bi*ya[khi]+((a*a*a-a)*y2a[klo]+(bi*bi*bi-
bi)*y2a[khi])*(h*h)/6.0;
}
```

Είναι η ρουτίνα υπολογισμού των σημείων της καμπύλης παρεμβολής με τη μέθοδο των splines. Εντοπίζεται αρχικά μεταξύ ποιων γνωστών τιμών (συνιστωσών των σημείων στον άξονα x στο διάγραμμα βαθμονόμησης) βρίσκεται η μετρούμενη τιμή και υπολογίζεται η πραγματική τιμή της δύναμης (συνιστώσα y στο διάγραμμα) που αντιστοιχεί σε αυτή τη μέτρηση.

```
/////////Main Program/////////
void JR3_Get_Force()
{

///////Calculating Forces
     if (sensors>0)
                ioctl(fd,IOCTL0_JR3_FILTER0,&fm);
```

Με την εντολή `ioctl(fd,IOCTL0_JR3_FILTER0,&fm)` λαμβάνουμε την τρέχουσα τιμή που αντιστοιχεί στις δυνάμεις και τις ροπές από τον αισθητήρα με τη μορφή ενός ακεραίου.

```
     for (i=0;i<3;i++) {
          val[i]=0;
          temp=fm.f[i]*fs.f[i];
          val[i] = (temp/16384);
     }
```

Για κάθε συνιστώσα της δύναμης αυτός ο ακέραιος αριθμός πολλαπλασιάζεται με την κλίμακα που του αντιστοιχεί και διαιρείται με την τιμή που εκφράζει το μέγιστο αυτού του ακεραίου (τιμή 16384). Με αυτή τη διαδικασία η τιμή που πήραμε από τον αισθητήρα μετασχηματίζεται σε N (Newton).

```
///////Zero-offset Correction


force_vector_analysis();
```

Στη θέση στην οποία μετρούμε τις δυνάμεις πρέπει να υπολογιστεί ο όρος της βαρυτικής δύναμης που επηρεάζει τη μέτρηση και να αντισταθμιστεί. Υπολογίζεται έτσι το διάνυσμα `df`.

```
val[0]=val[0]+offset[0]-df[0];  ///
val[1]=val[1]+offset[1]-df[1];  /// Only external force applied
val[2]=val[2]+offset[2]-df[2];  ///
```

Στην τιμή που πήραμε από τον αισθητήρα πρέπει να προσθέσουμε την τιμή του βάρους που αγνοήθηκε κατά την εκκίνηση του αισθητήρα (`offset`), ώστε να μπορούμε στη συνέχεια με την αφαίρεση του διανύσματος της τρέχουσας βαρυτικής δύναμης `df` να εξαλείφουμε πλήρως την επίδραση του βάρους στη δεδομένη χρονική στιγμή και τη δεδομένη θέση και προσανατολισμό του αισθητήρα.

```
for (i=0;i<3;i++){          ///In order to calibrate in same state
    val[i]=val[i]+(jr3_own_gravity_force+extra_offset_force)-
calibr_off;
}
```

Αναγάγουμε τις τιμές των δυνάμεων στις συνθήκες στις οποίες έγινε η βαθμονόμηση για τη σωστή χρήση της μεθόδου. Σκοπός είναι να μετασχηματιστεί αυτή η τιμή στην αντίστοιχη βαθμονομημένη και έπειτα να επιστρέψουμε στις συνθήκες μέτρησης ξανά.

```
////////Calibration  of   FORCES  using Cubic Splines
Interpolation///////////

     //Fx
if (val[0]>=-16 && val[0]<=18.8) {
     n=40;
    splines(FXxa,FXya,FXy2a,n,val[0]);
    val[0]=y;
    calibr[0]=0;
}
else calibr[0]=1;
       //Fy
if (val[1]>=-16.3 && val[1]<=19) {
     n=41;
    splines(FYxa,FYya,FYy2a,n,val[1]);
    val[1]=y;
    calibr[1]=0;
}
else calibr[1]=1;
     //Fz
if (val[2]<=20 && val[2]>=-20.8) {
     n=72;
    splines(FZxa,FZya,FZy2a,n,val[2]);
    val[2]=y;
    calibr[2]=0;
}
else calibr[2]=1;

//////////////////////////
```

Για κάθε τιμή της δύναμης που έχουμε, εάν αυτή βρίσκεται εντός τον ορίων βαθμονόμησης, καλούμε τη ρουτίνα υπολογισμού των καμπυλών splines και υπολογίζεται η αντίστοιχη πραγματική τιμή. Σε αυτή την περίπτωση το διάνυσμα ελέγχου της βαθμονόμησης `calibr` στον αντίστοιχο άξονα παίρνει την τιμή 0 (μηδέν). Εάν βρισκόμαστε εκτός των ορίων της βαθμονόμησης, δεν γίνεται κάποιος υπολογισμός και το αντίστοιχο σημείο στο διάνυσμα `calibr` παίρνει την τιμή 1 (ένα).

```
for (i=0;i<3;i++){          ///Back to real Values
    val[i]=val[i]-
(jr3_own_gravity_force+extra_offset_force)+calibr_off;
}
```

Αναγάγουμε τις βαθμονομημένες, πλέον, τιμές στις συνθήκες μέτρησης με αντίστροφη διαδικασία.

```
//////Calculation of Torque

        for (i=0;i<3;i++) {
         val[i+3]=0;
         temp=fm.m[i]*fs.m[i];
            val[i+3] = (temp/16384);
     }
```

Αντίστοιχος μετασχηματισμός της ακέραιας τιμής που παίρνουμε από τον αισθητήρα σε μονάδες μέτρησης ροπής Nm/10.

```
/////////Calibration  of   TORQUES  using Cubic Splines
Interpolation/////////
    //Mx
if (val[3]>=-6.2 && val[3]<=8.6) {
      n=43;
    splines(MXxa,MXya,MXy2a,n,val[3]);
    val[3]=y;
    calibr[3]=0;
}
else calibr[3]=1;
    //My
if (val[4]>=-6.7 && val[4]<=8.2) {
      n=43;
    splines(MYxa,MYya,MYy2a,n,val[4]);
    val[4]=y;
    calibr[4]=0;
}
else calibr[4]=1;
    //Mz
if (val[5]>=-7.0 && val[5]<=6.8) {
      n=43;
    splines(MZxa,MZya,MZy2a,n,val[5]);
    val[5]=y;
    calibr[5]=0;
}
else calibr[5]=1;
```

Για κάθε τιμή της ροπής που έχουμε, εάν αυτή βρίσκεται εντός τον ορίων βαθμονόμησης, καλούμε τη ρουτίνα υπολογισμού των καμπυλών splines και υπολογίζεται η αντίστοιχη πραγματική τιμή. Σε αυτή την περίπτωση το διάνυσμα ελέγχου της βαθμονόμησης `calibr` στον αντίστοιχο άξονα παίρνει την τιμή 0 (μηδέν). Εάν βρισκόμαστε εκτός των ορίων της βαθμονόμησης δεν γίνεται κάποιος υπολογισμός και το αντίστοιχο σημείο στο διάνυσμα `calibr` παίρνει την τιμή 1 (ένα).

```
//// Direction Correction (x axis)
val[0]=-val[0];
val[3]=-val[3];
```

Για τη δεδομένη πειραματική διαδικασία και με σκοπό την ύπαρξη αντιστοιχίας του καρτεσιανού συστήματος συντεταγμένων του τελικού σημείου δράσης του ρομπότ και του αισθητήρα, είναι απαραίτητη η αντιστροφή της φοράς του άξονα x, το οποίο αντιστοιχεί σε αλλαγή προσήμου στις δυνάμεις κατά x και στις ροπές ως προς x.

```
/// Threshold
for (i=0;i<6;i++) {
     if (val[i]>=-thresh && val[i]<=thresh) val[i]=0;
}
```

Όταν μια τιμή φόρτισης είναι κατά απόλυτη τιμή μικρότερη της τιμής του κατωφλιού (threshold) που έχει οριστεί στην αρχή του κώδικα, η τιμή αυτή μηδενίζεται και αγνοείται.

```
/// Converting torque to N*m
for (i=3;i<6;i++) val[i]=val[i]/10;
```

Για την έξοδο των τιμών των ροπών σε σύστημα S.I. γίνεται μετατροπή τους σε Nm.

```
/// Returning to Control
for (i=0;i<6;i++) Force[i]=val[i];

}
```

Αποθηκεύουμε το τελικό διάνυσμα των φορτίσεων στην κοινής χρήσης μεταβλητή `Force` που έχει οριστεί μέσω του αρχείου global_variables.c.

2.2 ForceControl.cpp (για 7 βαθμούς ελευθερίας)

```
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <pa10.h>
#include "global_variables.c"
#include "PA10.h"
#include "PA10.cpp"
#include "JR3_force.h"
#include "JR3_force.c"
#include "Matrix.h"
#include "Matrix.cpp"
#include "getmatrices.h"
#include "getmatrices.c"
```

Όπως σε όλα τα προγράμματα σε γλώσσα προγραμματισμού C και C++, οι εντολές `include` στην αρχή του κώδικα καθορίζουν τις βιβλιοθήκες και τα υποπρογράμματα, που είναι απαραίτητα για την πραγματοποίηση κάποιων λειτουργιών. Συγκεκριμένα, για τη δεδομένη εφαρμογή, εκτός των βασικών `stdlib.h` και `stdio.h` είναι απαραίτητη και η σύνδεση με τη βιβλιοθήκη `lm` μέσω του κεφαλικού αρχείου `math.h` για την πραγματοποίηση συγκεκριμένων μαθηματικών πράξεων, όπως τριγωνομετρικοί υπολογισμοί.

Η βιβλιοθήκη `libpa10` ενσωματώνεται μέσω του αρχείου κεφαλής pa10.h και καθορίζει τις συναρτήσεις επικοινωνίας, αποστολής και λήψης δεδομένων από το πρόγραμμά μας στον ελεγκτή του ρομπότ, όπως αναλύεται και στη συνέχεια του κώδικα.

Τέλος, ενσωματώνουμε στο κυρίως πρόγραμμα τις υπορουτίνες `global_variables.c` (για τη χρήση κάποιων συγκεκριμένων κοινών μεταβλητών από όλο το πρόγραμμα), `PA10.cpp` (για τον υπολογισμό της ορθής κινηματικής και της Ιακωβιανής του συστήματος), `JR3_force.c` (για την επικοινωνία και λήψη τιμών από τον αισθητήρα δύναμης/ροπής), `Matrix.c` (για τη δυνατότητα πραγματοποίησης πράξεων πινάκων) και `getmatrices.c` (για τον υπολογισμό και την εισαγωγή στο σύστημα του δυναμικού μοντέλου του ρομποτικού βραχίονα PA-10).

Οι τελευταίες υπορουτίνες δεν είναι ενσωματωμένες στο λειτουργικό σύστημα και πρέπει να βρίσκονται στον ίδιο φάκελο με το κυρίως πρόγραμμα κατά τη μετάφρασή του. Αυτό καθορίζεται από τη χρήση των εισαγωγικών (“ “) στα οποία περικλείεται το όνομα της κάθε υπορουτίνας. Να σημειωθεί ότι για τις περισσότερες από τις ανωτέρω πρέπει να ενσωματώνεται και το αντίστοιχο αρχείο κεφαλής (*.h) (Για παραπάνω λεπτομέρειες για τη χρήση και λειτουργία αυτών των υπορουτινών βλέπε Παράρτημα Γ.1 Εγχειρίδιο Λογισμικού)

```
int main ()
{
     PA10 pa1;
```

Η παραπάνω εντολή ενεργοποιεί την υπορουτίνα υπολογισμού ορθής κινηματικής και Ιακωβιανής του συστήματος

```
     struct pa10_t* pa10;
     pa10 = pa10_init(5000000);
```

Οι δύο εντολές που προηγήθηκαν εκκινούν και καθορίζουν την επικοινωνία του λογισμικού με την ενδιάμεση βιβλιοθήκη libpa10 για το συγχρονισμό αποστολής και λήψης δεδομένων από τον ελεγκτή του ρομπότ, όπου η τιμή 5.000.000 καθορίζει την περίοδο της επικοινωνίας σε ns (nanosecond). Αυτή η τιμή αντιστοιχεί σε 5 ms (millisecond).

```
     double angles1[7];
     double position1[3];
     double orientation1[3];
```

Εδώ ορίζουμε το διάνυσμα εισόδου της ορθής κινηματικής `angles1` (οι γωνίες των αρθρώσεων του βραχίονα) και τα διανύσματα που επιστρέφονται ως θέση (`position1`) και προσανατολισμός (`orientation1`) του τελικού σημείου δράσης.

```
     float q_mech_lim[7]={177*PI/180,94*PI/180,174*PI/180,
                          137*PI/180,255*PI/180,165*PI/180,
                          255*PI/180}; //// rad

     float qdot_mech_lim[7]={1,1,2,2,2,2*PI,2*PI};     //// rad/s

     float t_mech_lim[7]={4.64*50,4.64*50,2.00*50,2.00*50,
                          0.29*50,0.29*50,0.29*50}; /// N*m
```

Στα τρία αυτά διανύσματα εισάγονται τα μηχανικά όρια θέσης ταχύτητας και ροπής της κάθε άρθρωσης με σειρά από την πρώτη ως την τελευταία, σύμφωνα με τον κατασκευαστή του ρομποτικού βραχίονα.

```
     float plim=0.0001,orlim=0.001;
```

Ορίζονται τα μέγιστα σφάλματα θέσης (σε mm) και προσανατολισμού (σε rad) για τον τερματισμό του βρόχου υπολογισμού της αντίστροφης κινηματικής του συστήματος.

```
     float dT=0.005;
     float dT2=0.025;
```

Το dt καθορίζει το πεδίο χρόνου ψευδό-ολοκλήρωσης και ψευδό-παραγώγισης (σε δευτερόλεπτα) στους υπολογισμούς κάποιον μεγεθών και πρέπει να ταυτίζεται με την περίοδο επικοινωνίας του προγράμματος με τον ελεγκτή του ρομπότ, για την εξασφάλιση της ορθότητας των πράξεων.

Το dt2 καθορίζει το πεδίο χρόνου ψευδό-ολοκλήρωσης στον εσωτερικό βρόχο υπολογισμού της αντίστροφης κινηματικής του συστήματος. Όσο μικρότερη είναι αυτή η τιμή, τόσο μεγαλύτερη ακρίβεια έχουμε, που την πληρώνουμε όμως σε υπολογιστικό κόστος.

```
float tr_ratio=50.0;
int amp=0,corr=0;
float off_angle=46.455*PI/180;   /// Rz of sensor/end-effector
```

Η τιμή του tr_ratio καθορίζει το λόγο μετάδοσης στους κινητήρες των αρθρώσεων του ρομποτικού βραχίονα και η τιμή του είναι αποτέλεσμα πειραματικών μετρήσεων (παρουσιάζοντας μια απόκλιση από την τιμή 50 που δίνει ο κατασκευαστής).

Αλλάζοντας την τιμή του amp καθορίζουμε την ενίσχυση της εφαρμοσμένης εξωτερικά δύναμης και ροπής μέσω του αισθητήρα, όσον αφορά στην αντίδραση του ρομπότ. Η μεταβλητή corr ελέγχει, στο αντίστοιχο κομμάτι του κώδικα που ακολουθεί, την υπέρβαση των μηχανικών ορίων και εκκινεί τη διαδικασία διόρθωσης της λύσης σε επιτρεπτά όρια. Εδώ αρχικοποιείται σε μηδενική τιμή (όλες οι παράμετροι εντός μηχανικών ορίων)

Τέλος, η τιμή της off_angle είναι αυτή που υπολογίστηκε στο Κεφάλαιο 4.12 και προσδιορίζει τη γωνιακή απόκλιση ως προς z των αξόνων μέτρησης δύναμης του αισθητήρα από το πλαίσιο συντεταγμένων του ακροδέκτη του ρομποτικού βραχίονα.

```
float num;
Matrix Ja(6,7);
Matrix JaT(7,6);
Matrix Ja_r(7,6);
Matrix JaNew(6,7);
Matrix xd(6,1);
Matrix x(6,1);
Matrix x_prev(6,1);
Matrix xdot(6,1);
Matrix x2dot(6,1);
Matrix xddot(6,1);
Matrix x1(6,1);
Matrix xd1dot(6,1);
Matrix e(6,1);
Matrix K(6,6);
Matrix Ke(6,1);
Matrix Sum(6,1);
Matrix J1(7,1);
Matrix JarJa (7,7);
Matrix I(7,7);
Matrix Sum2(7,7);
```

```
    Matrix J2(7,1);
    Matrix qadot(7,1);
    Matrix qdot1(7,1);
    Matrix q1(7,1);
    Matrix q1_prev(7,1);

    Matrix A1(7,7);
    Matrix Cmat1(7,7);
    Matrix G1(7,1);
    Matrix Friction1(7,1);

    Matrix Force_input(6,1);
    Matrix Force1(6,1);
    Matrix Transformation(4,4);
    Matrix u(6,1);
    Matrix y(7,1);
    Matrix t(7,1);

    Matrix Md(6,6);
    Matrix Md_inv(6,6);
    Matrix Ja_rMd_inv(7,6);
    Matrix Kd(6,6);
    Matrix Kdxdot(6,1);
    Matrix Kp(6,6);
    Matrix Kpx(6,1);
    Matrix Sub_y1(6,1);
    Matrix Sub_y2(6,1);
    Matrix Sub_y3(6,1);
    Matrix grad1(6,7);
    Matrix grad2(6,7);
    Matrix grad(6,1);
    Matrix Jad(6,7);
    Matrix Jadqd(6,1);
    Matrix MdJadqd(6,1);

    Matrix Ay(7,1);
    Matrix Cqdot(7,1);
    Matrix SumAyCqdot(7,1);
    Matrix JaF(7,1);
    Matrix SumFrictionJaF(7,1);
    Matrix SumGFrictionJaF(7,1);
```

Στο ανωτέρω μπλοκ εντολών ορίζονται όλα τα απαραίτητα μητρώα που θα χρησιμέψουν στους υπολογισμούς που θα ακολουθήσουν, προσδιορίζοντας τις διαστάσεις τους μέσα στις παρενθέσεις

```
///////// Starting values ///////////
   pa10_getpos(pa10);
   for (int axisno=0; axisno<7; axisno++)  {
     angles1[axisno]=pa10->status.axis[axisno].angle;
     qdot1.Data[axisno][0]=pa10->status.axis[axisno].velocity;
   }
```

Με την εντολή `pa10_getpos(pa10)` καλούμε τον ελεγκτή του ρομποτικού βραχίονα να μας επιστρέψει την κατάσταση των αρθρώσεων τη δεδομένη στιγμή. Έπειτα με ένα βρόχο εξισώνουμε το κάθε στοιχείο του διανύσματος, που έχουμε κατασκευάσει στην αρχή του προγράμματος για τις γωνίες των αρθρώσεων `angles1` και τις γωνιακές τους ταχύτητες `qdot1` (χρειάζεται ο ορισμός `qdot1.Data`, επειδή το διάνυσμα αυτό έχει οριστεί μέσω της υπορουτίνας `Matrix.c`), με τις τιμές που ζητήσαμε από τον ελεγκτή και οι οποίες παρουσιάζονται στη μορφή:

- `pa10->status.axis[axisno].angle` για τις γωνίες και
- `pa10->status.axis[axisno].velocity` για τις ταχύτητες

όπου `axisno` είναι ο δείκτης του δεδομένου βρόχου.

```
//////// Forward Kinematics //////////

   pa1.PA10_For_Kine_Comp(angles1);
   pa1.PA10_GetPose(position1, orientation1);
   for ( int i = 0; i < 3; i++ )
      {
      x.Data[i][0] = position1[i];
      x.Data[i+3][0] = orientation1[i];
        }
```

Με τις δύο πρώτες εντολές καλούμε τον υπολογισμό της ορθής κινηματικής του συστήματος για να γνωρίζουμε τις αρχικές συνθήκες. Στη συνέχεια τοποθετούμε την έξοδο θέσης και προσανατολισμού, που μας επιστρέφει σε ένα ενιαίο διάνυσμα `x`, το οποίο θα χρησιμοποιήσουμε στη συνέχεια.

```
   for ( int r = 0; r < 7; r++ ) q1.Data[r][0] = angles1[r];
   for ( int r = 0; r < 7; r++ ) q1_prev.Data[r][0]=q1.Data[r][0];
   for ( int r = 0; r < 7; r++ ) qdot1.Data[r][0]=0;
   for ( int r = 0; r < 6; r++ ) x_prev.Data[r][0]=x.Data[r][0];
```

Τοποθετούμε στο διάνυσμα `q1` τις τιμές των γωνιών των αρθρώσεων (`angles1`) και, επειδή θεωρούμε ότι οι αρχικές συνθήκες του συστήματος εκφράζουν μόνιμη κατάσταση, εξισώνουμε την τιμή των γωνιών των αρθρώσεων της προηγούμενης χρονικής στιγμής `q1_prev` με την τιμή των γωνιών στις αρχικές συνθήκες `q1`, όπως ομοίως πράττουμε και για τη θέση και τον προσανατολισμό του τελικού σημείου δράσης (`x_prev = x`). Παράλληλα ορίζουμε την αρχική τιμή της γωνιακής ταχύτητας των αρθρώσεων ίση με το μηδέν (`qdot1=0`).

```
// Matrices initialisation

      for ( int r = 0; r < 6; r++)
  {
      K.Data[r][r] = 1;
      I.Data[r][r] = 1;
      Md.Data[r][r] = 1;
      Kd.Data[r][r] = 40;
      Kp.Data[r][r] = 1;
  }
      Md.Data[2][2] = 1;
```

Με τις παραπάνω εντολές κατασκευάζουμε το μοναδιαίο μητρώο `I`, καθορίζουμε τα κέρδη `Md`, `Kd` και `Kp` του νόμου ελέγχου και το κέρδος `K` στο βρόχο επίλυσης της αντίστροφης κινηματικής.

```
// Initial angles (to calculate offset gravity force vector)

  alfa=orientation1[0];
  beta=orientation1[1];
  gama=orientation1[2];
```

Οι γωνίες `alfa`, `beta` και `gama`, που αντιστοιχούν στις γωνίες προσανατολισμού του τελικού σημείου δράσης, παίρνουν τις τιμές της δεδομένης χρονικής στιγμής (όπως αυτές υπολογίστηκαν μέσω της ορθής κινηματικής προηγουμένως), για να εισαχθούν στην υπορουτίνα λήψης και επεξεργασίας τιμών δύναμης και ροπής από τον αισθητήρα, με σκοπό τον υπολογισμό της αρχικής κατεύθυνσης της βαρυτικής δύναμης του αισθητήρα και των υπολοίπων πιθανών επιπλέον εξαρτημάτων, που προσδένονται σε αυτόν, και την τελική αντιστάθμισή της στη συνέχεια.

```
// Starting device and storing offset gravity vector

JR3_Start_Device();
```

Η εντολή αυτή καλείται μία φορά στο πρόγραμμα εκτός του βρόχου ελέγχου. Εκκινεί τον αισθητήρα, αρχικοποιεί τις παραμέτρους του και υπολογίζει την αρχική κατεύθυνσης της βαρυτικής δύναμης του αισθητήρα και των λοιπών επιπλέον εξαρτημάτων.

```
while(1) {
```

Οι εντολές που ακολουθούν μέχρι το τέλος του προγράμματος βρίσκονται μέσα σε έναν επαναληπτικό βρόχο ελέγχου δύναμης/ροπής.

```
    //////// Forward Kinematics /////////

    pa1.PA10_For_Kine_Comp(angles1);
    pa1.PA10_GetPose(position1, orientation1);
    for ( int i = 0; i < 3; i++ )
    {
         x.Data[i][0] = position1[i];
         x.Data[i+3][0] = orientation1[i];
    }
```

Όπως και στον υπολογισμό των αρχικών συνθηκών του συστήματος, έτσι και σε αυτό το μπλοκ εντολών υπολογίζεται η θέση και ο προσανατολισμός του τελικού σημείου δράσης, με βάση τις γωνίες των αρθρώσεων και με χρήση της υποροutίνας της ορθής κινηματικής, για τη δεδομένη χρονική στιγμή. Τα αποτελέσματα εισάγονται στο διάνυσμα `x`.

```
    /// Euler angles for offset correction
    alfa=orientation1[0];
    beta=orientation1[1];
    gama=orientation1[2];
```

Αντίστοιχα με την αρχικοποίηση του αισθητήρα, οι γωνίες `alfa`, `beta` και `gama`, που αντιστοιχούν στις γωνίες προσανατολισμού του τελικού σημείου δράσης, εξισώνονται τη δεδομένη χρονική στιγμή με τον προσανατολισμό που υπολογίστηκε, μέσω της ορθής κινηματικής, για τον υπολογισμό της νέας θέσης του βαρυτικού διανύσματος του αισθητήρα και των λοιπών εξαρτημάτων που φέρει με σκοπό την αντιστάθμισή του.

```
    /// Getting calibrated force/torque values

    JR3_Get_Force();
```

Η εντολή αυτή λαμβάνει τις τιμές της δύναμης και ροπής που εφαρμόζονται τη συγκεκριμένη χρονική στιγμή στον αισθητήρα και τις αποθηκεύει στο κοινής χρήσης διάνυσμα `Force`. Η διαδικασία αυτή περιλαμβάνει το μετασχηματισμό των τιμών αυτών σύμφωνα με τη βαθμονόμηση που έχει γίνει, χρησιμοποιώντας καμπύλες splines και πραγματοποιεί μηδενική αντιστάθμιση της επίδρασης του ίδιου του βάρους του αισθητήρα και των εξαρτημάτων που φέρει. (Για περισσότερες λεπτομέρειες βλέπε Παράρτημα Β.2.1).

```
//// Storing force/torque values in local Matrix ////
    for ( int i = 0; i < 6; i++ )  {
        Force_input.Data[i][0]=Force[i];
    }
```

Αποθηκεύουμε προσωρινά τις τιμές που έχουμε λάβει από τον αισθητήρα στο διάνυσμα, ώστε να τις ανάγουμε στο πλαίσιο του τελικού σημείου δράσης.

```
//// Transformation of force to end effector coordinates
     Transformation.Data[0][0]=cos(off_angle);
     Transformation.Data[0][1]=-sin(off_angle);
     Transformation.Data[1][0]=sin(off_angle);
     Transformation.Data[1][1]=cos(off_angle);

     Transformation.Data[2][2]=1;
```

Με τις παραπάνω εντολές καθορίζεται το μητρώο μετασχηματισμού περιστροφής ως προς z άξονα ως προς τη γωνία απόκλισης (off_angle) των δύο πλαισίων αναφοράς, η τιμή της οποίας δίδεται στην αρχή του προγράμματος.

```
     for( int i = 0; i < 3; i++ )  {
Force1.Data[i][0]=(Transformation.Data[i][0]*Force_input.Data[0][0])
+(Transformation.Data[i][1]*Force_input.Data[1][0])+
(Transformation.Data[i][2]*Force_input.Data[2][0]);

Force1.Data[i+3][0]=(Transformation.Data[i][0]*Force_input.Data[3][0
])+(Transformation.Data[i][1]*Force_input.Data[4][0])+
(Transformation.Data[i][2]*Force_input.Data[5][0]);
     }
```

Στον παραπάνω βρόχο γίνεται αναγωγή των τιμών των δυνάμεων και των ροπών στο πλαίσιο του τελικού σημείου δράσης, έπειτα από πολλαπλασιασμό των τιμών αυτών με το μητρώο μετασχηματισμού.

```
//// Calculating velocities ////

     for ( int i = 0; i < 6; i++ ) xdot.Data[i][0]=(x.Data[i][0]-
                                   x_prev.Data[i][0])/dT;
```

Ψευδό-παραγωγίζοντας ως προς dT υπολογίζουμε τις συνιστώσες της ταχύτητας (γωνιακής και γραμμικής) του τελικού σημείου δράσης χρησιμοποιώντας τις συνιστώσες της θέσης του τη συγκεκριμένη χρονική στιγμή και την αμέσως προηγούμενη.

```
///// Force model input (storing values in global variables)

     for ( int i = 0; i < 7; i++ ) {
         q[i]=q1.Data[i][0];
            qdot[i]=qdot1.Data[i][0];
       }
```

Για να μπορούν να εισαχθούν οι τιμές της γωνίας και της γωνιακής ταχύτητας των αρθρώσεων του ρομποτικού βραχίονα στην υπορουτίνα υπολογισμού του δυναμικού

μοντέλου, τις μεταφέρουμε μία προς μία με τη χρήση του ανωτέρω βρόχου στα κοινής χρήσης διανύσματα q και qdot.

```
/////////////// Force Control ///////////////

    pa1.PA10_Jacobian_Analytic_2(&Ja);
```

Η παραπάνω εντολή υπολογίζει την Ιακωβιανή του συστήματος στη συγκεκριμένη κατάσταση του συστήματος του ρομποτικού βραχίονα και την επιστρέφει στη μορφή πίνακα Ja.

```
    /// u specification ///
    for ( int i = 0; i < 6; i++ )
                        u.Data[i][0]=(1+amp)*Force1.Data[i][0];
```

Έχουμε το νόμο ελέγχου

$$M_d \ddot{x} + K_D \dot{x} + K_P x = u \qquad \text{όπου}$$

$$u = (1 + a) \cdot F$$

το οποίο υπολογίζεται με τον παραπάνω βρόχο και αποθηκεύεται στο διάνυσμα u.

```
    /// xd specification ///
    Md.invert_mat(&Md_inv);
    Kdxdot.mult(&Kd, &xdot);
    Kpx.mult(&Kp, &x);
      Sub_y1.sub(&u, &Kdxdot);
      Sub_y2.sub(&Sub_y1, &Kpx);
    x2dot.mult(&Md_inv, &Sub_y2);
```

Οι παραπάνω εντολές εκφράζουν πράξεις μεταξύ πινάκων (για περισσότερες λεπτομέρειες βλέπε Παράρτημα Γ.1) και εκφράζουν την επίλυση του νόμου ελέγχου που παρουσιάστηκε προηγουμένως ως προς $\ddot{x}$:

$$\ddot{x} = M_d^{-1}(u - K_D \dot{x} - K_P x)$$

για τον υπολογισμό της δεύτερης παραγώγου των συνιστωσών της θέσης και του προσανατολισμού του τελικού σημείου.

Διαδοχικά γίνεται αντιστροφή του μητρώου Md, πολλαπλασιασμός του κέρδους Kd με την τιμή της ταχύτητας xdot του τελικού σημείου δράσης, πολλαπλασιασμός του κέρδους Kp με την τιμή της θέσης x του τελικού σημείου δράσης και αφαίρεση των δύο τελευταίων

αποτελεσμάτων από το διάνυσμα u (u - Kd*xdot – Kp* x). Πολλαπλασιάζοντας το αντεστραμμένο μητρώο με το αποτέλεσμα της προηγούμενης πράξης, μας δίνει την τιμή της δεύτερης παραγώγου των συνιστωσών της θέσης και του προσανατολισμού του τελικού σημείου (επιτάχυνση), η οποία αποθηκεύεται στο διάνυσμα x2dot.

```
   for ( int i = 0; i < 6; i++ )
   xddot.Data[i][0]=xdot.Data[i][0]+(x2dot.Data[i][0]*dT);
   for ( int i = 0; i < 6; i++ )
   xd.Data[i][0]=x.Data[i][0]+(xddot.Data[i][0]*dT);
```

Ψευδό-ολοκληρώνοντας ως προς dT υπολογίζουμε τις συνιστώσες της επιθυμητής ταχύτητας xddot (γωνιακής και γραμμικής) του τελικού σημείου δράσης και στη συνέχεια της επιθυμητής θέσης και προσανατολισμού του (xd). Ουσιαστικά προσθέτουμε στην τιμή της παλιάς ταχύτητας και θέσης την αντίστοιχη στοιχειώδη μεταβολή, λόγω της ύπαρξης της επιθυμητής επιτάχυνσης x2dot.

```
    /////////// Storing previous values ////////////
    for ( int i = 0; i < 7; i++ ) q1_prev.Data[i][0]=q1.Data[i][0];
    for ( int r = 0; r < 6; r++ ) x_prev.Data[r][0]=x.Data[r][0];
```

Πλέον, σαν τιμές της γωνίας των αρθρώσεων και θέσης/προσανατολισμού του τελικού σημείου δράσης της προηγούμενης χρονικής στιγμής ορίζονται για χρήση στον επόμενο κύκλο οι αντίστοιχες τιμές της δεδομένης χρονικής στιγμής, καθώς σκοπός μας είναι να μεταβούμε στην κατάσταση των επιθυμητών τιμών xd και xddot, όπως αυτές υπολογίστηκαν παραπάνω.

```
    ///// Initialising position/orientation Error /////
        for ( int i = 0; i < 6; i++ ) e.Data[i][0] = xd.Data[i][0] -
                                        x.Data[i][0];
```

Στον αλγόριθμο υπολογισμού της αντίστροφης κινηματικής η διαδικασία ανάγεται στην ελαχιστοποίηση της απόκλισης e μεταξύ της επιθυμητής και της τρέχουσας θέσης, όπως αυτή ορίζεται στον παραπάνω βρόχο.

```
////////// Inverse Kinematics algorithm //////////

    while ( e.Data[0][0]>plim || e.Data[1][0]>plim ||
            e.Data[2][0]>plim || e.Data[3][0]>orlim ||
            e.Data[4][0]>orlim || e.Data[5][0]>orlim ||
           e.Data[0][0]<-plim || e.Data[1][0]<-plim ||
            e.Data[2][0]<-plim || e.Data[3][0]<-orlim ||
            e.Data[4][0]<-orlim || e.Data[5][0]<-orlim)
    {
```

Ο εσωτερικός βρόχος της αντίστροφης κινηματικής επαναλαμβάνεται εφόσον έστω και μία από τις τιμές της θέσης και του προσανατολισμού βρίσκονται κατά απόλυτη τιμή πάνω από τα επιτρεπτά όρια `plim` και `orlim` αντίστοιχα, όπως καθορίζεται από την ανωτέρω συνθήκη.

```
/// Forward Kinematics
     pa1.PA10_For_Kine_Comp(angles1);
     pa1.PA10_GetPose(position1, orientation1);
     for ( int i = 0; i < 3; i++ )
     {
          x1.Data[i][0] = position1[i];
          x1.Data[i+3][0] = orientation1[i];
     }
  ///
```

Υπολογίζεται μέσω της ορθής κινηματικής και αποθηκεύεται στο προσωρινό διάνυσμα `x1` (διαφορετικό του `x`) η τρέχουσα θέση και προσανατολισμός της εικονικής μετακίνησης του τελικού σημείου δράσης μέχρι τη σύγκλιση στην επιθυμητή τιμή, έπειτα από στοιχειώδη μεταβολή των γωνιών των αρθρώσεων, που υπολογίζεται στη συνέχεια του αλγορίθμου.

```
pa1.PA10_Jacobian_Analytic_2(&Ja);
Ja.r_pseudo_inverse(&Ja_r);
```

Υπολογίζεται το νέο μητρώο Ιακωβιανής με βάση την ορθή κινηματική που προηγήθηκε, το οποίο ψευδοαντιστέφεται εξάγοντας το μητρώο `Ja_r`.

```
for ( int i = 0; i < 3; i++ ) xd1dot.Data[i][0] = 0;
for ( int i = 0; i < 6; i++ ) e.Data[i][0] = xd.Data[i][0] -
                              x1.Data[i][0];
```

Στους δύο παραπάνω βρόχους γίνεται μηδενισμός των στοιχειωδών γραμμικών ταχυτήτων που υπολογίζονται για τον αλγόριθμο της αντίστροφης κινηματικής και επανυπολογίζεται η απόκλιση e μεταξύ της επιθυμητής και της τρέχουσας θέσης,

```
/*
double ka = 1.0;

qadot.Data[0][0] += ka*(-0.0231)*angles1[0];
qadot.Data[1][0] += ka*(-0.0435)*angles1[1];
qadot.Data[2][0] += ka*(-0.0235)*angles1[2];
qadot.Data[3][0] += -10*ka*(-0.0299)*angles1[3];
qadot.Data[4][0] += ka*(-0.0160)*angles1[4];
qadot.Data[5][0] += ka*(-0.0248)*angles1[5];
qadot.Data[6][0] += 0*(-0.0160)*angles1[6];
  */
```

```
    qadot.Data[0][0] = 0;
    qadot.Data[1][0] = 0;
    qadot.Data[2][0] = 0;
    qadot.Data[3][0] = 0;
    qadot.Data[4][0] = 0;
    qadot.Data[5][0] = 0;
    qadot.Data[6][0] = 0;
```

Το πακέτο εντολών που προηγήθηκε αναφέρεται σε δύο μπλοκ εντολών για τον υπολογισμό των συνιστωσών qadot της επαναληπτικής μεθόδου της αντίστροφης κινηματικής, όπου το πρώτο έχει τιμές διάφορες του μηδενός και είναι ανενεργό με τη μορφή σχολίων (/* εντολές */) και το δεύτερο δίνει μηδενικές τιμές στο διάνυσμα. Ουσιαστικά αυτό το διάνυσμα εκφράζει τη δυνατότητα επιλογής μία από τις άπειρες λύσεις της αντίστροφης κινηματικής, στο επτά (7) βαθμών ελευθερίας ρομποτικό σύστημα που διαθέτουμε, καθορίζοντας την κίνηση των εσωτερικών αρθρώσεων προς μία συγκεκριμένη διάταξη, χωρίς να επηρεάζεται η θέση και ο προσανατολισμός του τελικού σημείου δράσης.

```
    Ke.mult(&K, &e);
    Sum.add(&xd1dot, &Ke);
    J1.mult(&Ja_r, &Sum);
    JarJa.mult(&Ja_r, &Ja);
    Sum2.sub(&I, &JarJa);
    J2.mult(&Sum2, &qadot);
    qdot1.add(&J1, &J2);
```

Οι παραπάνω εντολές υπολογίζουν μέσω πράξεων μεταξύ πινάκων την ακόλουθη παράσταση:

$$\dot{q} = J_A^+(\dot{x}_d + Ke) + (I - J_A^+ J_A)\dot{q}_a$$

όπου J_A^+ η ψευδοαντίστροφος της Ιακωβιανής και $\dot{q}_a$ το διάνυσμα των ταχυτήτων που εκφράζονται στο μηδενικό χώρο των άπειρων λύσεων της αντίστροφης κινηματικής και αφορούν τον έλεγχο της ενδιάμεσης διάταξης του ρομποτικού βραχίονα, ανεξάρτητα από την επίτευξη της επιθυμητής θέσης και προσανατολισμού του τελικού σημείου δράσης.

Σκοπός είναι ο υπολογισμός της στοιχειώδους γωνιακής ταχύτητας των αρθρώσεων που αποθηκεύεται στο διάνυσμα qdot1.

```
    for ( int i = 0; i < 7; i++ )
    q1.Data[i][0] = q1.Data[i][0] + qdot1.Data[i][0]* dT2;

    for ( int i = 0; i < 7; i++ )
         angles1[i] = q1.Data[i][0];
  }

     /////// End of Inverse Kinematics algorithm /////////
```

Στο πρώτο βρόχο ψευδό-ολοκληρώνοντας ως προς dT2 υπολογίζουμε τις συνιστώσες των νέων γωνιών των αρθρώσεων και τις αποθηκεύουμε μέσω του επόμενου βρόχου στο διάνυσμα angles1 για την εκ νέου είσοδο στη ρουτίνα της ορθής κινηματικής. Ο βρόχος σύγκλισης προς την επιθυμητή λύση της αντίστροφης κινηματικής επιστρέφει στον αρχικό έλεγχο, όπου, από τη συνθήκη επανάληψης του αλγορίθμου, καθορίζεται εάν έχουμε πέσει κάτω από τα μέγιστα όρια σφάλματος ή πρέπει να επαναληφθεί η διαδικασία.

```
    for ( int i = 0; i < 7; i++ ) qdot1.Data[i][0]=(q1.Data[i][0]-
                                  q1_prev.Data[i][0])/dT;
```

Από τη στιγμή που μέσω του αλγορίθμου της αντίστροφης κινηματικής έχει υπολογιστεί το νέο διάνυσμα των γωνιών των αρθρώσεων του ρομποτικού βραχίονα, υπολογίζουμε και τη νέα γωνιακή ταχύτητά τους ψευδό-παραγωγίζοντας ως προς dT με χρήση του ανωτέρω βρόχου.

```
/// Checking velocity and position mechanical limits ///

    for ( int i = 0; i < 7; i++ )
      {
     if (qdot1.Data[i][0]>qdot_mech_lim[i])
           {
           qdot1.Data[i][0]=qdot_mech_lim[i];
           q1.Data[i][0]=q1_prev.Data[i][0]+(qdot1.Data[i][0]*dT);
//q1 Correction
           corr=1;
           }
     else if (qdot1.Data[i][0]<-qdot_mech_lim[i])
           {
           qdot1.Data[i][0]=-qdot_mech_lim[i];
           q1.Data[i][0]=q1_prev.Data[i][0]+(qdot1.Data[i][0]*dT);
           corr=1;
           }
     }
```

Με τη χρήση του παραπάνω βρόχου ελέγχουμε για κάθε άρθρωση εάν η επιθυμητή ταχύτητα είναι μεγαλύτερη του άνω ορίου (qdot1>qdot_mech_lim) ή μικρότερη του κάτω ορίου (qdot1<-qdot_mech_lim). Σε αυτή την περίπτωση η ταχύτητα εξισώνεται με το όριο αυτό, επαναπροσδιορίζεται η τιμή της αντίστοιχης νέας γωνίας q1 ψευδό-ολοκληρώνοντας ως προς dT και ενημερώνουμε το σύστημα ότι έχει πραγματοποιηθεί διόρθωση, δίνοντας στη μεταβλητή corr την τιμή 1.

```
      for ( int i = 0; i < 7; i++ )
      {
      if (q1.Data[i][0]>q_mech_lim[i])
           {
           q1.Data[i][0]=q_mech_lim[i];
```

```
          qdot1.Data[i][0]=0;  //qdot1 Correction
          corr=1;
          }
     else if (q1.Data[i][0]<-q_mech_lim[i])
          {
          q1.Data[i][0]=-q_mech_lim[i];
          qdot1.Data[i][0]=0;
          corr=1;
          }
     }
```

Αντίστοιχα, με τη χρήση του παραπάνω βρόχου ελέγχουμε για κάθε άρθρωση εάν η επιθυμητή γωνιακή θέση είναι μεγαλύτερη του άνω μηχανικού ορίου (`q1>q_mech_lim`) ή μικρότερη του κάτω μηχανικού ορίου (`q1<-q_mech_lim`). Σε αυτή την περίπτωση η τιμή της γωνιακής θέσης εξισώνεται με αυτή του μηχανικού ορίου και η γωνιακή ταχύτητα της άρθρωσης που έχει υπολογιστεί παίρνει την τιμή μηδέν. Επιπλέον ενημερώνουμε και πάλι το σύστημα ότι έχει πραγματοποιηθεί διόρθωση, δίνοντας στη μεταβλητή `corr` την τιμή 1.

```
     if (corr==1)
     {
         for ( int r = 0; r < 7; r++ ) angles1[r]=q1.Data[r][0];
          pa1.PA10_For_Kine_Comp(angles1);
          pa1.PA10_GetPose(position1, orientation1);
          for ( int i = 0; i < 3; i++ )
          {
               xd.Data[i][0] = position1[i];
               xd.Data[i+3][0] = orientation1[i];
          }
         for ( int i = 0; i < 6; i++ )
xddot.Data[i][0]=(xd.Data[i][0]-x.Data[i][0])/dT;
     corr=0;
     }
```

Στην περίπτωση και μόνο που έχει πραγματοποιηθεί κάποια διόρθωση των τιμών της γωνιακής θέσης και ταχύτητας των αρθρώσεων (και αυτό επαληθεύεται από το σύστημα εφόσον εντοπίσει στη μεταβλητή `corr` την τιμή 1), πραγματοποιείται το ανωτέρω πακέτο εντολών.

Υπολογίζονται εκ νέου μέσω της ορθής κινηματικής η θέση και ο προσανατολισμός του τελικού σημείου δράσης, χρησιμοποιώντας πλέον τις διορθωμένες τιμές της γωνιακής θέσης, σε σχέση με τα μηχανικά όρια. Τα νέα δεδομένα του τελικού σημείου δράσης έτσι πλέον θα αναφέρονται σε μια δυνατή λύση της διάταξης και με ασφάλεια ως προς τη λειτουργία και την ακεραιότητα του ρομποτικού βραχίονα. Τέλος, ψευδό-παραγωγίζοντας ως προς dT υπολογίζουμε και τη γραμμική και γωνιακή ταχύτητα του τελικού σημείου δράσης και επαναφέρουμε την τιμή της μεταβλητής ελέγχου `corr` την τιμή 0.

```
/// New position
    for ( int i = 0; i < 6; i++ )
    {
    x.Data[i][0]=xd.Data[i][0];
    xdot.Data[i][0]=xddot.Data[i][0];
    }
```

Τοποθετούμε τις τιμές της επιθυμητής θέσης και ταχύτητας του τελικού σημείου δράσης (που υπολογίστηκαν μέσω του νόμου ελέγχου και πιθανώς διορθώθηκαν εντός των μηχανικών ορίων των αρθρώσεων) στα βασικά διανύσματα θέσης και ταχύτητας (`x` και `xdot` αντίστοιχα), τα οποία θα χρησιμοποιήσουμε στη συνέχεια στους υπολογισμούς μας.

```
    for ( int i = 0; i < 7; i++ ) {
    q[i]=q1.Data[i][0];
    qdot[i]=qdot1.Data[i][0];
    }
```

Για τον υπολογισμό των μητρώων του δυναμικού μοντέλου του ρομποτικού βραχίονα πρέπει να εισάγουμε στην αντίστοιχη υπορουτίνα τις τιμές της επιθυμητής γωνιακής θέσης και ταχύτητας των αρθρώσεων. Αυτό πραγματοποιείται περνώντας τις τιμές αυτές (που υπολογίστηκαν παραπάνω μέσω της αντίστροφης κινηματικής και πιθανώς διορθώθηκαν εντός μηχανικών ορίων) στα κοινής χρήσεως αντίστοιχα διανύσματα `q` και `qdot`, τα οποία μπορεί να καλέσει και να χειριστεί και η υπορουτίνα υπολογισμού του δυναμικού μοντέλου `getmatrices.c`.

```
//////// PA-10 matrices Calculation //////////

    getmatrices();
```

Η εντολή αυτή υπολογίζει το δυναμικό μοντέλο και αποθηκεύει τα αντίστοιχα μητρώα στους κοινής χρήσεως πίνακες `A`, `Cmat`, `Friction` και `G` που μπορούν να χρησιμοποιηθούν από το κυρίως πρόγραμμα.

```
    for ( int i = 0; i < 7; i++ ) {
        G1.Data[i][0]=G[i];
        Friction1.Data[i][0]=Friction[i];
        for ( int j = 0; j < 7; j++ ) {
            A1.Data[i][j]=A[i][j];
            Cmat1.Data[i][j]=Cmat[i][j];
            }
    }
```

Τα κοινής χρήσεως μητρώα που υπολογίστηκαν μέσω του δυναμικού μοντέλου εισάγονται μέσω του παραπάνω βρόχου στα αντίστοιχα τοπικά χρήσεως μητρώα που έχουν δημιουργηθεί μέσω της υπορουτίνας Matrix.c, δίνοντάς μας έτσι τη δυνατότητα στη συνέχεια να πραγματοποιούμε πράξεις με ευχέρεια μεταξύ αυτών των μητρώων.

```
    pa1.PA10_For_Kine_Comp(angles1);
    pa1.PA10_GetPose(position1, orientation1);
    pa1.PA10_Jacobian_Analytic_2(&Ja);
```

Υπολογίζουμε εκ νέου την Ιακωβιανή του συστήματος. (Να σημειωθεί ότι για τον ορθό υπολογισμό της Ιακωβιανής πρέπει να προηγείται ο υπολογισμός της ορθής κινηματικής, όπως πραγματοποιείται εδώ).

```
/// y calculation ///
   Ja.r_pseudo_inverse(&Ja_r);
   Ja_rMd_inv.mult(&Ja_r, &Md_inv);
```

Στις παραπάνω εντολές ψευδοαντιστρέφεται η Ιακωβιανή εξάγοντας το μητρώο `Ja_r` και το αποτέλεσμα πολλαπλασιάζεται με το αντίστροφο μητρώο που έχει υπολογιστεί παραπάνω στον κώδικα. Έτσι έχουμε πετύχει την παράσταση:

$$J_A^{-1} \cdot M_d^{-1}$$

```
    /// Jacobian derivative calculation loop
```

Ο αλγόριθμος που ακολουθεί υπολογίζει με έμμεσο τρόπο την παράγωγο της Ιακωβιανής του συστήματος

```
    for ( int i = 0; i < 7; i++ )
            {
          angles1[i]=angles1[i]+dT;
```

Για μία άρθρωση κάθε φορά αυξάνουμε την τιμή της γωνιακής θέσης κατά μία πολύ μικρή τιμή dT.

```
          pa1.PA10_For_Kine_Comp(angles1);
          pa1.PA10_GetPose(position1, orientation1);
          pa1.PA10_Jacobian_Analytic_2(&JaNew);
```

Υπολογίζουμε εκ νέου την Ιακωβιανή του συστήματος για αυτή τη μικρή αλλαγή και την αποθηκεύουμε στην προσωρινή τιμή JaNew. (Να σημειωθεί ότι για τον ορθό υπολογισμό της Ιακωβιανής πρέπει να προηγείται ο υπολογισμός της ορθής κινηματικής, όπως πραγματοποιείται εδώ).

```
            grad1.sub(&JaNew, &Ja);
            grad2.scale(&grad1, (1/dT));
            grad.mult(&grad2, &qdot1);
```

Αφαιρούμε τα δεδομένα της Ιακωβιανής του συστήματός μας από αυτά της προσωρινής Ιακωβιανής, διαιρούμε κάθε στοιχείο με τη τιμή dT και πολλαπλασιάζουμε το μητρώο αυτό με το διάνυσμα των γωνιακών ταχυτήτων των αρθρώσεων. Το αποτέλεσμα είναι ένα διάνυσμα grad.

```
            for ( int j = 0; j < 6; j++ )
                                Jad.Data[j][i]=grad.Data[j][0];
            angles1[i]=q1.Data[i][0];

            }
     ///
```

Το διάνυσμα grad που υπολογίστηκε, αντιστοιχεί στη στήλη της παραγώγου της Ιακωβιανής, που αναφέρεται στην άρθρωση της οποίας τη γωνία μεταβάλλαμε κατά μία πολύ μικρή τιμή dT. Στον παραπάνω βρόχο περνάμε αυτά τα δεδομένα στην αντίστοιχη στήλη του μητρώου Jad, που εκφράζει την παράγωγο της Ιακωβιανής του συστήματος. Επίσης επαναφέρουμε την τιμή της γωνίας που μεταβάλλαμε στην αρχική της, και ο κεντρικός βρόχος υπολογισμού της παραγώγου της Ιακωβιανής επιστρέφει στην αρχή, για να μεταβληθεί κατά μία πολύ μικρή τιμή dT και η τιμή της γωνιακής θέσης της επόμενης άρθρωσης με σκοπό τον υπολογισμό και της επόμενης στήλης του μητρώου Jad.

```
   Jadqd.mult(&Jad, &qdot1);
   MdJadqd.mult(&Md, &Jadqd);
   Sub_y3.sub(&Sub_y2,&MdJadqd);
   y.mult(&Ja_rMd_inv, &Sub_y3);
```

Με τις παραπάνω πράξεις πινάκων υπολογίζεται το μητρώο y σύμφωνα με την εξής παράσταση:

$$y = J^{-1}M_d^{-1}(-K_d\dot{x}_d - K_p x_d - M_d \cdot \dot{J} \cdot \dot{q}_d + u)$$

```
/// torque calculation ///

    JaT.trans(&Ja);

    Ay.mult(&A1, &y);
    Cqdot.mult(&Cmat1, &qdot1);
    SumAyCqdot.add(&Ay, &Cqdot);

    JaF.mult(&JaT, &Force1);
    SumFrictionJaF.add(&Friction1, &JaF);
    SumGFrictionJaF.add(&G1, &SumFrictionJaF);

    t.add(&SumAyCqdot, &SumGFrictionJaF);
```

Τέλος υπολογίζουμε το διάνυσμα των ροπών, που πρέπει να στείλουμε σε κάθε άρθρωση, με τις αντίστοιχες παραπάνω πράξεις που εκφράζουν την παράσταση:

$$\tau = A(q)\cdot y + C(q,\dot{q})\cdot \dot{q} + G(q) + Fr(\dot{q}) + J^T \cdot F$$

```
/// Checking torque mechanical limits ///

   for ( int i = 0; i < 7; i++ )
     {
     if (t.Data[i][0]>t_mech_lim[i])
     {
     t.Data[i][0]=t_mech_lim[i];
     }
     else if (t.Data[i][0]<-t_mech_lim[i])
     {
     t.Data[i][0]=-t_mech_lim[i];
     }
   }
```

Επειδή υπάρχουν περιορισμοί όσον αφορά και στο μέγεθος των ροπών που μπορεί να εφαρμόσουν οι κινητήρες του ρομποτικού βραχίονα, ελέγχουμε αν τυχόν σε κάποια από τις αρθρώσεις έχουμε ξεπεράσει το αντίστοιχο όριο και εφόσον αυτό ισχύει, υποβιβάζουμε τη ροπή αυτή στο επιθυμητό όριο

```
/// Sending Command to PA10 robot

    for (int axisno=0; axisno<7; axisno++)     {
pa10->command.axis[axisno].torque=t.Data[axisno][0]/tr_ratio;

       pa10->command.axis[axisno].ctl=PA10_TORQUE;
       }
```

Για κάθε άρθρωση περνάμε την τιμή της επιθυμητής ροπής στον ενδιάμεσο χώρο αποθήκευσης της libpa10 μέσω της εντολής:

`pa10->command.axis[axisno].torque=t.Data[axisno][0]/tr_ratio;`

που εκφράζει τον προσδιορισμό της ροπής (`torque`). Η μεταβλητή `axisno` εκφράζει το δείκτη που καθορίζει την άρθρωση για την οποία γίνεται ο υπολογισμός. Να σημειωθεί ότι η τιμή που αποστέλλουμε στο ρομποτικό βραχίονα είναι πριν από τη μείωση των στροφών, γι' αυτό η ροπή που υπολογίστηκε για κάθε άρθρωση διαιρείται με το λόγο μετάδοσης `tr_ratio`, που έχει καθοριστεί στην αρχή του προγράμματος.

Η επόμενη εντολή καθορίζει πως ο έλεγχος (`ctl`) του ρομποτικού βραχίονα θα γίνει μέσω αποστολής ροπής στο σύστημά μας για κάθε άρθρωση (`PA10_TORQUE`)

```
        pa10_do(pa10);
```

Η εντολή αυτή αποστέλλει τα δεδομένα, που υπολογίστηκαν στον προηγούμενο βρόχο, στον ελεγκτή του ρομποτικού βραχίονα για την εφαρμογή (στη συγκεκριμένη περίπτωση) των ροπών στις αρθρώσεις. Να σημειωθεί ότι μέσω της εντολής αυτής ο ελεγκτής μάς επιστρέφει πάλι την κατάσταση των αρθρώσεων την τρέχουσα στιγμή, όπως συμβαίνει και με την εντολή `pa10_getpos(pa10)`.

```
/// Preparing data for next loop

   for (int axisno=0; axisno<7; axisno++)  {
     angles1[axisno]=pa10->status.axis[axisno].angle;
     qdot1.Data[axisno][0]=pa10->status.axis[axisno].velocity;
     }

  }

}
```

Όπως και στην αρχή του προγράμματος, με ένα βρόχο εξισώνουμε το κάθε στοιχείο του διανύσματος της γωνιακής θέσης των αρθρώσεων `angles1` και της γωνιακής ταχύτητας `qdot1` (χρειάζεται ο ορισμός `qdot1.Data`, επειδή το διάνυσμα αυτό έχει οριστεί μέσω της υπορουτίνας Matrix.c) με τις τιμές που ζητήσαμε από τον ελεγκτή.

Έτσι, κατά την επανάληψη του κεντρικού βρόχου ελέγχου, θα χρησιμοποιήσουμε πλέον σαν συνθήκες γωνιακής θέσης και ταχύτητας των αρθρώσεων τις πραγματικές τιμές της κατάστασης του ρομποτικού βραχίονα, που πιθανώς μπορεί να αποκλίνουν λίγο από τις επιθυμητές που υπολογίσαμε και προσπαθήσαμε να επιτύχουμε.

2.3 Fcontrol2dof.cpp (για 2 βαθμούς ελευθερίας)

```
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <pa10.h>
#include <posix_time.h>
#include <pthread.h>
#include <stdlib.h>
#include <unistd.h>
#include <termios.h>
#include <fcntl.h>
#include <string.h>
#include <sys/ioctl.h>

#include "global_variables.c"
#include "JR3_force.h"
#include "JR3_force.c"
#include "PA10.h"
#include "PA10.c"
#include "Matrix.h"
#include "Matrix.cpp"
#include "getmatrices.h"
#include "getmatrices.c"
```

Όπως και στην ανάλυση του προηγούμενου κώδικα για τους 7 βαθμούς ελευθερίας, οι εντολές include στην αρχή του κώδικα καθορίζουν τις βιβλιοθήκες και τα υποπρογράμματα που είναι απαραίτητα για την πραγματοποίηση κάποιων λειτουργιών. Συγκεκριμένα, για τη δεδομένη εφαρμογή, εκτός των βασικών stdlib.h και stdio.h είναι απαραίτητη και η σύνδεση με τη βιβλιοθήκη lm μέσω του κεφαλικού αρχείου math.h, για την πραγματοποίηση συγκεκριμένων μαθηματικών πράξεων, όπως τριγωνομετρικοί υπολογισμοί.

Η βιβλιοθήκη libpa10 ενσωματώνεται μέσω του αρχείου κεφαλής pa10.h και καθορίζει τις συναρτήσεις επικοινωνίας, αποστολής και λήψης δεδομένων από το πρόγραμμά μας στον ελεγκτή του ρομπότ, όπως αναλύεται και στη συνέχεια του κώδικα.

Για τον ίδιο λόγο ενσωματώνουμε και κάποιες άλλες βιβλιοθήκες, όπως η posix_time και η pthread, που χρησιμοποιούν οι εντολές χρονισμού της επικοινωνίας με τον ελεγκτή.

Τέλος, ενσωματώνουμε στο κυρίως πρόγραμμα τις υπορουτίνες global_variables.c (για τη χρήση κάποιων συγκεκριμένων κοινών μεταβλητών από όλο το πρόγραμμα), PA10.c (για τον υπολογισμό της ορθής κινηματικής, της αντίστροφης κινηματικής και της Ιακωβιανής του συστήματος), JR3_force.c (για την επικοινωνία και λήψη τιμών από τον αισθητήρα δύναμης/ροπής), Matrix.c (για τη δυνατότητα πραγματοποίησης πράξεων πινάκων) και getmatrices.c (για τον υπολογισμό και την εισαγωγή στο σύστημα του δυναμικού μοντέλου του ρομποτικού βραχίονα PA-10).

Οι τελευταίες υπορουτίνες δεν είναι ενσωματωμένες στο λειτουργικό σύστημα και πρέπει να βρίσκονται στον ίδιο φάκελο με το κυρίως πρόγραμμα κατά τη μετάφρασή του. Αυτό καθορίζεται από τη χρήση των εισαγωγικών (“ “) στα οποία περικλείεται το όνομα της κάθε υπορουτίνας. Να σημειωθεί ότι για τις περισσότερες από τις ανωτέρω πρέπει να ενσωματώνεται και το αντίστοιχο αρχείο κεφαλής (*.h) (Για παραπάνω λεπτομέρειες για τη χρήση και λειτουργία αυτών των υπορουτινών βλέπε Παράρτημα Γ.1 Εγχειρίδιο Λογισμικού)

```
#define Nrows 7
#define Ncols 7


#define PERIOD 2500000          /* usec */
```

Πριν το κυρίως πρόγραμμα γίνονται κάποιες πρώτες ρυθμίσεις για την επικοινωνία με τον ελεγκτή του ρομποτικού βραχίονα. Η σημαντικότερη είναι ο προσδιορισμός της περιόδου επικοινωνίας μέσω της σταθεράς `PERIOD` , η οποία ορίζεται στα 2.500.000 ns, δηλαδή 2.5 ms.

```
struct app_param_t
{
int run;
} param1;
```

Η μεταβλητή `param1.run` που δομείται στις παραπάνω εντολές καθορίζει εάν θα επαναλαμβάνεται ο βρόχος του ελέγχου δύναμης και πότε θα διακόπτεται, ανάλογα με το εάν η τιμή της είναι 1 (στην πρώτη περίπτωση) ή 0 (στη δεύτερη περίπτωση).

```
void
Die (char *mess)
{
     perror (mess);
     param1.run = 0;
     printf ("Attempting to exit gracefully...\n");
     usleep(1000);
}
```

Η παραπάνω υπορουτίνα μηδενίζει τη μεταβλητή και οδηγεί `param1.run` έτσι το πρόγραμμα σε ομαλή έξοδο, όταν προκύψει κάποιο σφάλμα στη ρύθμιση της επικοινωνίας το οποίο την καλεί.

```
int main ()
{
```

Οι εντολές που ακολουθούν περικλείονται στο κυρίως πρόγραμμα του ελέγχου.

```
/* Matrix stuff */
     int i, loopcount, fd, BytesWritten;
     int axisno;
     int signum;
     float karvel[7] = { 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 };
     float qc[7] = { 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0 };
     float limit[7] = { 147.0, 83.0, 90.0, 110.0, 200.0, 95.0, 90.0
};
     float timediff;

     FILE *output;
     sigset_t set;
     timer_t periodic_timer;
     struct sigevent sig;
     struct sched_param mysched;
     struct itimerspec required, old;
     struct timespec first, res, current, last, before, now;
     struct pa10_t *pa10;  /*robot */
     struct termios options;
```

Με τις παραπάνω εντολές γίνεται αρχικοποίηση σημαντικών για το πρόγραμμα μεταβλητών και διανυσμάτων, όπως τα διανύσματα στα οποία θα αποθηκεύσουμε προσωρινά τη γωνιακή θέση qc και ταχύτητα `karvel`, σύμφωνα με τα στοιχεία που μας στέλνει ο ελεγκτής, τα μηχανικά όρια της γωνιακής θέσης των αρθρώσεων `limit` που θα χρησιμοποιήσουμε για τον έλεγχο ασφαλείας. Επίσης προσδιορίζεται η κατασκευή ενός αρχείου `output` και δομούνται μεταβλητές και ρουτίνες για την επικοινωνία με τον ελεγκτή.

```
     float q_mech_lim[2]={83*PI/180,110*PI/180};
                                        //// rad (less than maximum)
     float qdot_mech_lim[2]={0.5,1};
                                        //// rad/s (half of maximum)
     float t_mech_lim[2]={156.0,80.0};
                                        /// N*m (less than maximum)
```

Στα τρία αυτά διανύσματα εισάγονται τα μηχανικά όρια θέσης ταχύτητας και ροπής της κάθε άρθρωσης που θα χρησιμοποιήσουμε στην εφαρμογή, με σειρά από την πρώτη ως την τελευταία, σύμφωνα με τον κατασκευαστή του ρομποτικού βραχίονα. Να σημειωθεί ότι για τη συγκεκριμένη εφαρμογή τα όρια που θέσαμε ήταν ακόμα μικρότερα από τα επιτρεπτά, για να υπάρχει περισσότερη ασφάλεια για την ομαλή λειτουργία του ρομποτικού βραχίονα.

```
     float dT=0.0025;
```

Το `dt` καθορίζει το πεδίο χρόνου ψευδό-ολοκλήρωσης και ψευδό-παραγώγισης (σε δευτερόλεπτα) στους υπολογισμούς κάποιων μεγεθών και πρέπει να ταυτίζεται με την περίοδο επικοινωνίας του προγράμματος με τον ελεγκτή του ρομπότ, για την εξασφάλιση της ορθότητας των πράξεων

```
float tr_ratio=50.0;
int amp=0,corr=0;
float off_angle=46.455*PI/180;  /// Rz of sensor/end-effector
```

Η τιμή του `tr_ratio` καθορίζει το λόγο μετάδοσης στους κινητήρες των αρθρώσεων του ρομποτικού βραχίονα και η τιμή του είναι η τιμή 50 που δίνει ο κατασκευαστής.

Αλλάζοντας την τιμή του `amp` καθορίζουμε την ενίσχυση της εφαρμοσμένης εξωτερικά δύναμης και ροπής μέσω του αισθητήρα, όσον αφορά στην αντίδραση του ρομπότ. Η μεταβλητή `corr` ελέγχει, στο αντίστοιχο κομμάτι του κώδικα που ακολουθεί, την υπέρβαση των μηχανικών ορίων και εκκινεί τη διαδικασία διόρθωσης της λύσης σε επιτρεπτά όρια. Εδώ αρχικοποιείται σε μηδενική τιμή (όλες οι παράμετροι εντός μηχανικών ορίων)

Τέλος, η τιμή της `off_angle` είναι αυτή που υπολογίστηκε στο Κεφάλαιο 4.12 και προσδιορίζει τη γωνιακή απόκλιση ως προς z των αξόνων μέτρησης δύναμης του αισθητήρα από το πλαίσιο συντεταγμένων του ακροδέκτη του ρομποτικού βραχίονα.

```
float num;

int ttt = 0;
```

Η μεταβλητή `ttt` είναι η μεταβλητή ελέγχου που παίρνει την τιμή 1, όταν έχουμε υπερβεί κάποιο από τα μηχανικά όρια και ενεργοποιεί την `param1.run` αμέσως μετά την αποθήκευση των παραμέτρων του συστήματος στο αρχείο εξόδου, ώστε να σταματήσει η επανάληψη του βρόχου ελέγχου δύναμης.

```
Matrix q1(2,1);
Matrix x(2,1);
Matrix Ja(2,2);

Matrix q1_prev(2,1);
Matrix qdot1(2,1);
Matrix x_prev(2,1);
Matrix xdot(2,1);
Matrix x2dot(2,1);
Matrix xd(2,1);
Matrix xddot(2,1);

Matrix Force_input(6,1);
Matrix Force1(2,1);
Matrix Transformation(4,4);

Matrix JaT(2,2);
Matrix Ja_r(2,2);
Matrix JaNew(2,2);

Matrix Friction1(2,1);
```

```
     Matrix u(2,1);
     Matrix y(2,1);
     Matrix t(2,1);

     Matrix I(2,2);

    Matrix Md(2,2);
    Matrix Md_inv(2,2);
    Matrix Ja_rMd_inv(2,2);
    Matrix Kd(2,2);
    Matrix Kdxdot(2,1);
    Matrix Kp(2,2);
    Matrix Kpx(2,1);
    Matrix Sub_y1(2,1);
    Matrix Sub_y2(2,1);
    Matrix Sub_y3(2,1);
    Matrix grad1(2,2);
    Matrix grad2(2,2);
    Matrix grad(2,1);
    Matrix Jad(2,2);
    Matrix Jadqd(2,1);
    Matrix MdJadqd(2,1);

    Matrix JaF(2,1);
    Matrix SumFrictionJaF(2,1);
```

Στο ανωτέρω μπλοκ εντολών ορίζονται όλα τα απαραίτητα μητρώα που θα χρησιμέψουν στους υπολογισμούς που θα ακολουθήσουν, προσδιορίζοντας τις διαστάσεις τους μέσα στις παρενθέσεις

```
     // Matrices initialisation

          for ( int r = 0; r < 2; r++)
     {
          I.Data[r][r] = 1;
          Md.Data[r][r] = 3.5;
          Kd.Data[r][r] = 10.0;
          Kp.Data[r][r] = 0.0;
     }
```

Με τις παραπάνω εντολές κατασκευάζουμε το μοναδιαίο μητρώο I και καθορίζουμε τα κέρδη Md, Kd και Kp του νόμου ελέγχου.

```
     // Initial angles (to calculate offset gravity force vector)

     alfa=0;  // Planar experiment, no gravity force compensation
     beta=0;
     gama=0;
```

Οι γωνίες alfa, beta και gama, που αντιστοιχούν στις γωνίες προσανατολισμού του τελικού σημείου δράσης για να εισαχθούν στην υπορουτίνα λήψης και επεξεργασίας τιμών

δύναμης και ροπής από τον αισθητήρα, με σκοπό τον υπολογισμό της αρχικής κατεύθυνσης της βαρυτικής δύναμης, παίρνουν μηδενικές τιμές, αφού το πείραμα θα γίνει σε ένα επίπεδο κάθετο στο διάνυσμα της βαρύτητας και οι βαρυτικές δυνάμεις δεν επηρεάζουν τη διαδικασία.

```
       //// Transformation of force to end effector coordinates

     Transformation.Data[0][0]=cos(off_angle);
     Transformation.Data[0][1]=-sin(off_angle);
     Transformation.Data[1][0]=sin(off_angle);
     Transformation.Data[1][1]=cos(off_angle);

     Transformation.Data[2][2]=1;
```

Στον παραπάνω πακέτο εντολών γίνεται υπολογισμός του μητρώου περιστροφής με το οποίο θα χρειαστεί να πολλαπλασιάσουμε τις τιμές των δυνάμεων από τον αισθητήρα, για την αναγωγή των τιμών αυτών στο πλαίσιο του τελικού σημείου δράσης.

```
     // Starting device and storing offset gravity vector
        JR3_Start_Device();
```

Η εντολή αυτή καλείται μία φορά στο πρόγραμμα εκτός του βρόχου ελέγχου. Εκκινεί τον αισθητήρα, αρχικοποιεί τις παραμέτρους του και υπολογίζει την αρχική κατεύθυνσης της βαρυτικής δύναμης του αισθητήρα και των λοιπών επιπλέον εξαρτημάτων.

```
     /* Start initialization */
     pa10 = pa10_create();
     param1.run = 1;
        /* Get the maximum possible priority */
     mysched.sched_priority = sched_get_priority_max(SCHED_RR) - 1;
     printf("Setting the scheduler...\n");
        if( sched_setscheduler( 0, SCHED_RR, &mysched ) == -1 )
     {
           Die("ERROR setting the scheduler");
        }

     /* Get the clock's resolution. Should be used for some kind of
check */
        clock_getres(CLOCK_REALTIME_HR, &res);
        printf("Resolution: %ld nsec\n", res.tv_nsec);

     /* File opens and Time settings */
     if ((output = fopen ("output.txt", "w")) == NULL)
     {
           Die ("fopen");
     }

     if (clock_gettime (CLOCK_REALTIME, &current) != 0)
     {
           Die ("clock_gettime");
     }
```

```
     sig.sigev_notify = SIGEV_SIGNAL;
     sig.sigev_signo = SIGRTMIN;

     if (timer_create (CLOCK_REALTIME_HR, &sig, &periodic_timer) !=
0)
     {
          Die ("timer_create");
     }

     if (sigemptyset (&set) != 0)
     {
          Die ("sigemptyset");
     }

     if (sigaddset (&set, SIGRTMIN) != 0)
     {
          Die ("sigaddset");
     }
     pthread_sigmask (SIG_BLOCK, &set, NULL);

     first.tv_sec = 1;
     first.tv_nsec = 0;
     required.it_value = first;
     required.it_interval.tv_nsec = PERIOD;
     required.it_interval.tv_sec = 0;

     if (timer_settime (periodic_timer, 0, &required, &old) != 0)
     {
          Die ("timer_settime");
     }
     else
          printf ("successfully set timer attributes, entering idle
loop\nFirst tick in 1 sec, period %.2f msec\n", PERIOD / 1000000.0);
     last.tv_sec = 0;
     last.tv_nsec = 0;

     sigwait (&set, &signum);
```

Το παραπάνω πακέτο εντολών προετοιμάζει την επικοινωνία του λογισμικού μας με τον ελεγκτή του ρομπότ σε επίπεδο χαμηλού επιπέδου εντολών και σε περίοδο επικοινωνίας, που καθορίστηκε στην αρχή του προγράμματος. Επίσης ελέγχει και κατασκευάζει το αρχείο `output.txt` για την αποθήκευση των αποτελεσμάτων.

```
     printf ("Starting communications...\n");
     pa10_start (pa10);
     printf ("Got response, entering control loop!\n");
```

Η επικοινωνία ξεκινά μέσω της εντολής `pa10_start (pa10)`, σύμφωνα με τις παραμέτρους που ορίστηκαν στο προηγούμενο πακέτο εντολών.

```
    /* Initialize all axes for torque Control */

    for (axisno = 0; axisno <= 6; axisno++)
    {
         pa10->command.axis[axisno].ctl = PA10_TORQUE;
         pa10->command.axis[axisno].velocity = 0.0;
         pa10->command.axis[axisno].torque = 0.0;
    }

    pa10->command.comctl = 0;
```

Η μεταβλητή axisno εκφράζει το δείκτη που καθορίζει την άρθρωση για την οποία εκτελείται μία εντολή. Η εντολή pa10->command.axis[axisno].ctl = PA10_TORQUE καθορίζει πως ο έλεγχος (ctl) του ρομποτικού βραχίονα θα γίνει μέσω αποστολής ροπής στο σύστημά μας για κάθε άρθρωση (PA10_TORQUE). Για ασφάλεια μηδενίζουμε τη ροπή και την ταχύτητα, που αποθηκεύονται στις ενδιάμεσες μεταβλητές μεταξύ του ελεγκτή και του κώδικά μας.

```
    /* Uncomment any axis you would like to lock */

       pa10->command.axis[0].ctl = PA10_BRAKE;
       /*pa10->command.axis[1].ctl = PA10_BRAKE;*/
       pa10->command.axis[2].ctl = PA10_BRAKE;
       /*pa10->command.axis[3].ctl = PA10_BRAKE;*/
       pa10->command.axis[4].ctl = PA10_BRAKE;
       pa10->command.axis[5].ctl = PA10_BRAKE;
       pa10->command.axis[6].ctl = PA10_BRAKE;
```

Καθώς το πείραμα θα γινόταν σε ένα επίπεδο και εμείς θα χειριζόμασταν μόνο δύο από τους βαθμούς ελευθερίας, οι υπόλοιποι 5 θα έπρεπε να κλειδωθούν, και αυτό πραγματοποιήθηκε μέσω των εντολών που προηγούνται προσδιορίζοντας στον έλεγχο της κάθε άρθρωσης τη συνθήκη PA10_BRAKE που τη φρενάρει. Στο συγκεκριμένο πακέτο εντολών βάζουμε σε σχόλια τις αρθρώσεις που δεν θέλουμε να κλειδωθούν (εδώ τη δεύτερη και την τέταρτη άρθρωση του ρομποτικού βραχίονα).

```
    /* Get current position */
    sigwait(&set, &signum);
    pa10_write(pa10);
    clock_gettime(CLOCK_REALTIME_HR, &before);
```

Με την εντολή pa10_write(pa10) πραγματοποιούμε δύο διαδικασίες, καλούμε τον ελεγκτή του ρομποτικού βραχίονα να μας επιστρέψει την κατάσταση των αρθρώσεων τη δεδομένη στιγμή και αποστέλλουμε την επιθυμητή κατάσταση, που έχουμε αποθηκεύσει στις ενδιάμεσες μεταβλητές τύπου pa10->command.axis[axisno]."εντολή". Στη συγκεκριμένη περίπτωση, επειδή έχουμε δηλώσει έλεγχο ροπών και έχουμε μηδενίσει τις

τιμές της ροπής και της ταχύτητας δεν επηρεάζουμε ουσιαστικά την κατάσταση του βραχίονα.

```
/////////////// Starting values For Control /////////////////

          /* Update Current State */

          for (axisno = 0; axisno <= 6; axisno++)
          {
               qc[axisno] = pa10->status.axis[axisno].angle *
                                                    M_PI /180.0;
          }
```

Έπειτα με ένα βρόχο εξισώνουμε το κάθε στοιχείο του διανύσματος, που έχουμε κατασκευάσει στην αρχή του προγράμματος για τις γωνίες των αρθρώσεων qc, με τις τιμές που ζητήσαμε από τον ελεγκτή και οι οποίες παρουσιάζονται στη μορφή: `pa10->status.axis[axisno].angle`

```
       q[0]=qc[1];
       q[1]=qc[3];


    //////// Forward Kinematics /////////

       dir_kinematics();
       x_prev.Data[0][0]=x_[0];
       x_prev.Data[1][0]=x_[1];
```

Εξισώνουμε το καθολικής χρήσεως διάνυσμα q με την τρέχουσα τιμή της γωνιακής θέσης των αρθρώσεων qc (συγκεκριμένα μόνο για τις δύο αρθρώσεις που μας ενδιαφέρουν), γιατί αυτό το διάνυσμα μας χρησιμεύει ως είσοδος για την υπορουτίνα της ορθής κινηματικής. Η εντολή `dir_kinematics` υπολογίζει τη θέση του τελικού σημείου δράσης (συνιστώσες x, y) και αποθηκεύει την έξοδο στο διάνυσμα `x_`. Εξισώνουμε το διάνυσμα `x` με αυτό που μας επέστρεψε η ορθή κινηματική, για να το χρησιμοποιήσουμε στη συνέχεια.

```
    for ( int r = 0; r < 2; r++ ) q1_prev.Data[r][0]=q[r];
```

Επειδή θεωρούμε ότι οι αρχικές συνθήκες του συστήματος εκφράζουν μόνιμη κατάσταση, εξισώνουμε την τιμή των γωνιών των αρθρώσεων της προηγούμενης χρονικής στιγμής `q1_prev` με την τιμή των γωνιών στις αρχικές συνθήκες q.

```
/* Stationary mode */
          for(i=0;i<=50;i++)
     {
          sigwait(&set, &signum);
          for(axisno=0;axisno<=6;axisno++)
          {
          pa10->command.axis[axisno].torque=0.0;
          }
          pa10_write(pa10);
     }

     clock_gettime(CLOCK_REALTIME_HR, &before);

//------------------End of Initilisation----------------------//
```

Με το παραπάνω πακέτο εντολών και πριν ξεκινήσουμε το βρόχο ελέγχου, καθυστερούμε το πρόγραμμα για 50 κύκλους επικοινωνίας (i<=50), έτσι ώστε να εξασφαλίσουμε ότι έχουν προλάβει να ενεργοποιηθούν σωστά όλες οι συνθήκες επικοινωνίας, για να μην χάσουμε στη συνέχεια το χρονισμό μεταξύ του προγράμματός μας και του ελεγκτή του ρομποτικού βραχίονα.

```
     loopcount = 0;

     while (param1.run)
     {
```

Στην πρώτη εντολή αρχικοποιείται ο δείκτης μέτρησης επαναλήψεων του βρόχου που ακολουθεί `loopcount`. Οι παρακάτω εντολές βρίσκονται μέσα σε έναν επαναληπτικό βρόχο ελέγχου δύναμης ροπής. Η συνθήκη πραγματοποίησης του βρόχου είναι να είναι αληθής (τιμή 0) η μεταβλητή `param1.run`. Στη συνέχεια θα μελετήσουμε πότε γίνεται ψευδής αυτή η παράμετρος και οδηγείται το πρόγραμμα σε μια ασφαλή και ομαλή διακοπή.

```
     sigwait (&set, &signum);

     /* Get current time */

     clock_gettime(CLOCK_REALTIME_HR, &now);

     timediff =now.tv_sec - before.tv_sec +
                    (now.tv_nsec -before.tv_nsec) /1000000000.0;
```

Με τις παραπάνω εντολές περιμένουμε το χτύπο του εσωτερικού ρολογιού χρονισμού για την πραγματοποίηση του ελέγχου μία φορά σε κάθε περίοδο επικοινωνίας, το μέγεθος της οποίας καθορίστηκε στην αρχή του προγράμματος.

```
/* Update Current State */

     for (axisno = 0; axisno <= 6; axisno++)
     {
          qc[axisno] = pa10->status.axis[axisno].angle *
                                                    M_PI /180.0;
          karvel[axisno] = pa10->status.axis[axisno].velocity;
     }

        q1.Data[0][0]=qc[1];
        q1.Data[1][0]=qc[3];
        qdot1.Data[0][0]=karvel[1];
        qdot1.Data[1][0]=karvel[3];
```

Έχοντας ήδη καλέσει τον ελεγκτή του ρομποτικού βραχίονα να μας επιστρέψει την κατάσταση των αρθρώσεων τη δεδομένη στιγμή, στο τέλος του βρόχου εξισώνουμε το κάθε στοιχείο του διανύσματος, που έχουμε κατασκευάσει στην αρχή του προγράμματος για τις γωνίες των αρθρώσεων `q1` και τις γωνιακές τους ταχύτητες `qdot1` (χρειάζεται ο ορισμός `q1.Data` και `qdot1.Data`, επειδή το διάνυσμα αυτό έχει οριστεί μέσω της υπορουτίνας `Matrix.c`), με τις τιμές που ζητήσαμε από τον ελεγκτή και οι οποίες παρουσιάζονται στη μορφή:

- `pa10->status.axis[axisno].angle` για τις γωνίες και
- `pa10->status.axis[axisno].velocity` για τις ταχύτητες

και αποθηκεύονται προσωρινά στα διανύσματα `qc` και `karvel` αντίστοιχα.

```
     /* Calculate Full Torque */

     for(axisno=0;axisno<=6;axisno++)
     {
          pa10->command.axis[axisno].torque = 0.0;
     }
```

Για λόγους ασφαλείας μηδενίζουμε τις τιμές των ροπών για κάθε άρθρωση στο ενδιάμεσο διάνυσμα μεταξύ του προγράμματός μας και του ρομποτικού βραχίονα.

```
     //////// Forward Kinematics //////////

        q[0]=q1.Data[0][0];
        q[1]=q1.Data[1][0];
        dir_kinematics();
        x.Data[0][0]=x_[0];
        x.Data[1][0]=x_[1];
```

Όπως και στον υπολογισμό των αρχικών συνθηκών του συστήματος, έτσι και σε αυτό το πακέτο εντολών υπολογίζεται η θέση του τελικού σημείου δράσης με βάση τις γωνίες των

αρθρώσεων και με χρήση της υπορουτίνας της ορθής κινηματικής για τη δεδομένη χρονική στιγμή. Τα αποτελέσματα εισάγονται στο διάνυσμα x.

```
     /// Euler angles for offset correction

     alfa=0;
     beta=0;
     gama=0;
```

Αντίστοιχα με την αρχικοποίηση του αισθητήρα, οι γωνίες alfa, beta και gama, που αντιστοιχούν στις γωνίες προσανατολισμού του τελικού σημείου δράσης, και χρησιμεύουν στην αντιστάθμιση της βαρυτικής δύναμης που ασκεί ο ίδιος ο αισθητήρας, εξισώνονται τη δεδομένη χρονική στιγμή με το μηδέν, αφού το πείραμα θα γίνει σε ένα επίπεδο κάθετο στο διάνυσμα της βαρύτητας και οι βαρυτικές δυνάμεις δεν επηρεάζουν τη διαδικασία.

```
     /// Getting calibrated force/torque values

     JR3_Get_Force();
```

Η εντολή αυτή λαμβάνει τις τιμές της δύναμης και ροπής που εφαρμόζονται τη συγκεκριμένη χρονική στιγμή στον αισθητήρα και τις αποθηκεύει στο κοινής χρήσης διάνυσμα Force. Η διαδικασία αυτή περιλαμβάνει το μετασχηματισμό των τιμών αυτών σύμφωνα με τη βαθμονόμηση, που έχει γίνει χρησιμοποιώντας καμπύλες splines και πραγματοποιεί μηδενική αντιστάθμιση της επίδρασης του ίδιου του βάρους του αισθητήρα και των εξαρτημάτων που φέρει. (Για περισσότερες λεπτομέρειες βλέπε Παράρτημα Β.2.1).

```
     for( int i = 0; i < 3; i++ )  {
     Force_input.Data[i][0]=(Transformation.Data[i][0]*Force[0])+(Tr
ansformation.Data[i][1]*Force[1])+(Transformation.Data[i][2]*Force[2
]);
     Force_input.Data[i+3][0]=(Transformation.Data[i][0]*Force[3])+(
Transformation.Data[i][1]*Force[4])+(Transformation.Data[i][2]*Force
[5]);
     }
```

Στον παραπάνω βρόχο γίνεται αναγωγή των τιμών των δυνάμεων και των ροπών στο πλαίσιο του τελικού σημείου δράσης, έπειτα από πολλαπλασιασμό των τιμών αυτών με το μητρώο μετασχηματισμού.

```
       /// Using planar Force Values

       Force1.Data[0][0]=Force_input.Data[2][0];   /// Fz of sensor
       Force1.Data[1][0]=Force_input.Data[0][0];   /// Fx of sensor
```

Με βάση το σύστημα συντεταγμένων, που έχουμε θέσει στο τελικό σημείο δράσης του πειράματος, αποθηκεύουμε τις τιμές των δυνάμεων που χρειαζόμαστε ανάλογα με τον άξονα στον οποίο εφαρμόζονται. Για παράδειγμα στη συγκεκριμένη εφαρμογή ο άξονας x στο πείραμά μας αντιστοιχεί στον άξονα z του αισθητήρα και ο άξονας y στον άξονα x του αισθητήρα.

```
        //// Calculating velocities ////
        for ( int i = 0; i < 2; i++ ) xdot.Data[i][0]=
                                (x.Data[i][0]-x_prev.Data[i][0])/dT;
```

Ψευδό-παραγωγίζοντας ως προς dT υπολογίζουμε τις συνιστώσες της ταχύτητας (γωνιακής και γραμμικής) του τελικού σημείου δράσης χρησιμοποιώντας τις συνιστώσες της θέσης του τη συγκεκριμένη χρονική στιγμή και την αμέσως προηγούμενη.

```
/////////////// Force Law ///////////////

    /// u specification ///
    for ( int i = 0; i < 2; i++ )
                              u.Data[i][0]=(1+amp)*Force1.Data[i][0];
```

Έχουμε το νόμο ελέγχου

$$M_d \ddot{x} + K_D \dot{x} + K_P x = u \qquad \text{όπου}$$

$$u = (1+a) \cdot F$$

το οποίο υπολογίζεται με τον παραπάνω βρόχο και αποθηκεύεται στο διάνυσμα u.

```
    /// xd specification ///
    Md.invert_mat(&Md_inv);
    Kdxdot.mult(&Kd, &xdot);
    Kpx.mult(&Kp, &x);
      Sub_y1.sub(&u, &Kdxdot);
      Sub_y2.sub(&Sub_y1, &Kpx);

    x2dot.mult(&Md_inv, &Sub_y2);
```

Οι παραπάνω εντολές εκφράζουν πράξεις μεταξύ πινάκων (για περισσότερες λεπτομέρειες βλέπε Παράρτημα Γ.1) και εκφράζουν την επίλυση του νόμου ελέγχου, που παρουσιάστηκε προηγουμένως ως προς $\ddot{x}$:

$$\ddot{x} = M_d^{-1}(u - K_D \dot{x} - K_P x)$$

για τον υπολογισμό της δεύτερης παραγώγου των συνιστωσών της θέσης και του προσανατολισμού του τελικού σημείου.

Διαδοχικά γίνεται αντιστροφή του μητρώου Md, πολλαπλασιασμός του κέρδους Kd με την τιμή της ταχύτητας xdot του τελικού σημείου δράσης, πολλαπλασιασμός του κέρδους Kp με την τιμή της θέσης x του τελικού σημείου δράσης και αφαίρεση των δύο τελευταίων αποτελεσμάτων από το διάνυσμα u (u - Kd*xdot – Kp* x). Πολλαπλασιάζοντας το αντεστραμμένο μητρώο με το αποτέλεσμα της προηγούμενης πράξης, μας δίνει την τιμή της δεύτερης παραγώγου των συνιστωσών της θέσης και του προσανατολισμού του τελικού σημείου (επιτάχυνση), η οποία αποθηκεύεται στο διάνυσμα x2dot.

```
//////// Calculating desired position /////////

for ( int i = 0; i < 2; i++ )
        xddot.Data[i][0]=xdot.Data[i][0]+(x2dot.Data[i][0]*dT);
for ( int i = 0; i < 2; i++ )
         xd.Data[i][0]=x.Data[i][0]+(xddot.Data[i][0]*dT);
```

Ψευδό-ολοκληρώνοντας ως προς dT υπολογίζουμε τις συνιστώσες της επιθυμητής ταχύτητας xddot (γωνιακής και γραμμικής) του τελικού σημείου δράσης και στη συνέχεια της επιθυμητής θέσης και προσανατολισμού του (xd). Ουσιαστικά προσθέτουμε στην τιμή της παλιάς ταχύτητας και θέσης την αντίστοιχη στοιχειώδη μεταβολή, λόγω της ύπαρξης της επιθυμητής επιτάχυνσης x2dot.

```
////////// Storing previous values ///////////

for ( int i = 0; i < 2; i++ ) q1_prev.Data[i][0]=q1.Data[i][0];
for ( int r = 0; r < 2; r++ ) x_prev.Data[r][0]=x.Data[r][0];
```

Πλέον σαν τιμές της γωνίας των αρθρώσεων και θέσης/προσανατολισμού του τελικού σημείου δράσης της προηγούμενης χρονικής στιγμής ορίζονται για χρήση στον επόμενο κύκλο οι αντίστοιχες τιμές της δεδομένης χρονικής στιγμής, καθώς σκοπός μας είναι να μεταβούμε στην κατάσταση των επιθυμητών τιμών xd και xddot, όπως αυτές υπολογίστηκαν παραπάνω.

```
///////////// Inverse Kinematics ////////////////

x_[0]=xd.Data[0][0];  //Input Data
x_[1]=xd.Data[1][0];  //Input Data
inv_kinematics();
q1.Data[0][0]=q[0];   //Output Data
q1.Data[1][0]=q[1];   //Output Data
```

Εξισώνουμε το καθολικής χρήσεως διάνυσμα x_ με την επιθυμητή τιμή της θέσης του τελικού σημείου δράσης xd, γιατί αυτό το διάνυσμα μας χρησιμεύει ως είσοδος για την υπορουτίνα της αντίστροφης κινηματικής. Η εντολή inv_kinematics υπολογίζει τη γωνιακή θέση των αρθρώσεων, που αντιστοιχεί σε αυτή τη θέση του τελικού σημείου

δράσης, και αποθηκεύει την έξοδο στο διάνυσμα q. Εξισώνουμε το διάνυσμα q1 με αυτό που μας επέστρεψε η ορθή κινηματική για να το χρησιμοποιήσουμε στη συνέχεια.

```
    //// Calculating desired joint velocity

    for ( int i = 0; i < 2; i++ ) qdot1.Data[i][0]=
                              (q1.Data[i][0]-q1_prev.Data[i][0])/dT;
```

Από τη στιγμή που, μέσω του αλγορίθμου της αντίστροφης κινηματικής, έχει υπολογιστεί το νέο διάνυσμα των (επιθυμητών) γωνιών των αρθρώσεων του ρομποτικού βραχίονα, υπολογίζουμε και τη νέα (επιθυμητή) γωνιακή ταχύτητα τους ψευδό-παραγωγίζοντας ως προς dT με χρήση του ανωτέρω βρόχου.

```
    /// Desired Position trasnformed to Control input Data

    for ( int i = 0; i < 2; i++ )
    {
    x.Data[i][0]=xd.Data[i][0];
    xdot.Data[i][0]=xddot.Data[i][0];
    }
```

Τοποθετούμε τις τιμές της επιθυμητής θέσης και ταχύτητας του τελικού σημείου δράσης (που υπολογίστηκαν μέσω του νόμου) στα βασικά διανύσματα θέσης και ταχύτητας (x και xdot αντίστοιχα) τα οποία θα χρησιμοποιήσουμε στη συνέχεια στους υπολογισμούς μας.

```
///// Force model input - Friction Only (storing values in global
array qdot) //////

     qdot1.Data[0][0]=karvel[1];
     qdot1.Data[1][0]=karvel[3];
        qdot[0]=qdot1.Data[0][0];
        qdot[1]=qdot1.Data[1][0];
```

Για τον υπολογισμό των μητρώων του δυναμικού μοντέλου του ρομποτικού βραχίονα (στη συγκεκριμένη εφαρμογή μόνο του μητρώου της τριβής) πρέπει να εισάγουμε στην αντίστοιχη υπορουτίνα τις τιμές της επιθυμητής γωνιακής ταχύτητας των αρθρώσεων. Αυτό πραγματοποιείται περνώντας τις τιμές αυτές (που υπολογίστηκαν παραπάνω μέσω της αντίστροφης κινηματικής και πιθανώς διορθώθηκαν εντός μηχανικών ορίων) στο κοινής χρήσεως διάνυσμα qdot, το οποίο μπορεί να καλέσει και να χειριστεί και η υπορουτίνα υπολογισμού του μοντέλου τριβής getmatrices.c.

```
//////// PA-10 matrices Calculation //////////

    getmatrices();
    Friction1.Data[0][0]=Friction[0];
    Friction1.Data[1][0]=Friction[1];
```

Η εντολή αυτή υπολογίζει το δυναμικό μοντέλο και αποθηκεύει το αντίστοιχο μητρώο που χρειαζόμαστε στο κοινής χρήσεως διάνυσμα `Friction`, που μπορεί να χρησιμοποιηθεί από το κυρίως πρόγραμμα. (Εδώ αποθηκεύεται άμεσα στο τοπικό διάνυσμα `Friction1`).

```
    ///////Jacobian//////////

    Jacobian();
    for ( int i = 0; i < 2; i++ )  {
        for ( int r = 0; r < 2; r++ ) Ja.Data[i][r]=J[i][r];
    }
```

Υπολογίζουμε την Ιακωβιανή του συστήματος. Σαν είσοδο η Ιακωβιανή χρειάζεται το διάνυσμα των γωνιακών θέσεων (που έχει υπολογιστεί νωρίτερα) και επιστρέφει σαν έξοδο τον πίνακα `J`, του οποίου τις τιμές τοποθετούμε στο αντίστοιχο μητρώο `Ja`, που έχει δημιουργηθεί μέσω της υπορουτίνας Matrix.c, για να μπορούμε να πραγματοποιήσουμε πράξεις στη συνέχεια.

```
    JaT.trans(&Ja);
```

Με την παραπάνω εντολή υπολογίζεται ο ανάστροφος πίνακας της Ιακωβιανής και αποθηκεύεται στο μητρώο `JaT`.

```
// y calculation ///

    Ja.invert_mat(&Ja_r);
    Ja_rMd_inv.mult(&Ja_r, &Md_inv);
```

Στις παραπάνω εντολές ψευδοαντιστρέφεται η Ιακωβιανή, εξάγοντας το μητρώο `Ja_r` και το αποτέλεσμα πολλαπλασιάζεται με το αντίστροφο μητρώο, που έχει υπολογιστεί παραπάνω στον κώδικα. Έτσι έχουμε πετύχει την παράσταση:

$$J_A^{-1} \cdot M_d^{-1}$$

```
     /// Jacobian derivative calculation loop
```

Ο αλγόριθμος που ακολουθεί υπολογίζει με έμμεσο τρόπο την παράγωγο της Ιακωβιανής του συστήματος

```
     for ( int i = 0; i < 2; i++ )
               {
           q[i]=q1.Data[i][0]+0.1;
```

Για μία άρθρωση κάθε φορά αυξάνουμε την τιμή της γωνιακής θέσης κατά μία πολύ μικρή τιμή (0.1).

```
                    ///////Jacobian//////////
                    Jacobian();
                    for ( int m = 0; m < 2; m++ )  {
                        for ( int r = 0; r < 2; r++ )
                                              JaNew.Data[m][r]=J[m][r];
                    }
                    ////////////////////////
```

Υπολογίζουμε εκ νέου την Ιακωβιανή του συστήματος γι' αυτή τη μικρή αλλαγή και την αποθηκεύουμε στην προσωρινή τιμή JaNew.

```
            grad1.sub(&JaNew, &Ja);
            grad2.scale(&grad1, (1/0.1));
            grad.mult(&grad2, &qdot1);
```

Αφαιρούμε τα δεδομένα της Ιακωβιανής του συστήματός μας από αυτά της προσωρινής Ιακωβιανής, διαιρούμε κάθε στοιχείο με τη μικρή τιμή (0.1) που μεταβάλαμε τη γωνία και πολλαπλασιάζουμε το μητρώο αυτό με την το διάνυσμα των γωνιακών ταχυτήτων των αρθρώσεων. Το αποτέλεσμα είναι ένα διάνυσμα grad.

```
                for ( int j = 0; j < 2; j++ )
                                    Jad.Data[j][i]=grad.Data[j][0];
            q[i]=q1.Data[i][0];
            }
     ///
```

Το διάνυσμα grad που υπολογίστηκε αντιστοιχεί στη στήλη της παραγώγου της Ιακωβιανής που αναφέρεται στην άρθρωση, της οποίας τη γωνία μεταβάλλαμε κατά μία πολύ μικρή τιμή (0.1). Στον παραπάνω βρόχο περνάμε αυτά τα δεδομένα στην αντίστοιχη στήλη του μητρώου Jad, που εκφράζει στην παράγωγο της Ιακωβιανής του συστήματος. Επίσης επαναφέρουμε την τιμή της γωνίας που μεταβάλαμε στην αρχική της και ο κεντρικός βρόχος υπολογισμού της παραγώγου της Ιακωβιανής επιστρέφει στην αρχή, για να μεταβληθεί κατά μία πολύ μικρή τιμή (0.1) και η τιμή της γωνιακής θέσης της επόμενης άρθρωσης με σκοπό τον υπολογισμό και της επόμενης στήλης του μητρώου Jad.

```
    Jadqd.mult(&Jad, &qdot1);
    MdJadqd.mult(&Md, &Jadqd);

    Kdxdot.mult(&Kd, &xdot);
    Kpx.mult(&Kp, &x);
      Sub_y1.sub(&u, &Kdxdot);
      Sub_y2.sub(&Sub_y1, &Kpx);
    Sub_y3.sub(&Sub_y2,&MdJadqd);
    y.mult(&Ja_rMd_inv, &Sub_y3);
```

Με τις παραπάνω πράξεις πινάκων υπολογίζεται το μητρώο y σύμφωνα με την εξής παράσταση:

$$y = J^{-1}M_d^{-1}(-K_d\dot{x}_d - K_p x_d - M_d \cdot \dot{J} \cdot \dot{q}_d + u)$$

```
/// torque calculation ///

    JaF.mult(&JaT, &Force1);
    SumFrictionJaF.add(&Friction1, &JaF);
    t.add(&y, &SumFrictionJaF);
```

Τέλος υπολογίζουμε το διάνυσμα των ροπών που πρέπει να στείλουμε σε κάθε άρθρωση με τις αντίστοιχες παραπάνω πράξεις που εκφράζουν την παράσταση:

$$\tau = I \cdot y + Fr(\dot{q}) + J^T \cdot F$$

```
/// Checking torque and velocity mechanical limits ///

    for ( int i = 0; i < 2; i++ )
      {
      if (fabs(t.Data[i][0]) > t_mech_lim[i])
      {
/*          param1.run=0;*/
            ttt = 1;
            printf("Ektos Oriwn Ropwn\n");
            printf("%i %f %f\n", i, t.Data[i][0], t_mech_lim[i]);
      }
```

Με τη χρήση του παραπάνω βρόχου ελέγχουμε για κάθε άρθρωση εάν η επιθυμητή ροπή είναι μεγαλύτερη του άνω ορίου (`t>t_mech_lim`) ή μικρότερη του κάτω ορίου (`t<-t_mech_lim`) χρησιμοποιώντας την εντολή `fabs`, που υπολογίζει την απόλυτη τιμή. Σε περίπτωση που είμαστε εκτός ορίων, θέτουμε τη μεταβλητή ελέγχου `ttt` ίση με 1, για να ενεργοποιήσουμε τον τερματισμό του προγράμματος στη συνέχεια, και τυπώνουμε ένα μήνυμα λάθους, καθώς και την άρθρωση που επιθυμήσαμε να δεχτεί ροπή άνω του ορίου, μαζί με την επιθυμητή ροπή και το προκαθορισμένο όριο, ώστε να μπορούμε να ελέγξουμε το σφάλμα, εάν αυτό εμφανιστεί.

```
      if (fabs(pa10->status.axis[i].velocity) > qdot_mech_lim[i])
      {
            ttt = 1;
            printf("Ektos Oriwn Taxythtas\n");
      }

    }
```

Με τη χρήση του παραπάνω βρόχου ελέγχουμε για κάθε άρθρωση εάν η επιθυμητή ταχύτητα είναι μεγαλύτερη του άνω ορίου (qdot1>qdot_mech_lim) ή μικρότερη του κάτω ορίου (qdot1<-qdot_mech_lim) χρησιμοποιώντας την εντολή fabs, που υπολογίζει την απόλυτη τιμή. Σε περίπτωση που είμαστε εκτός ορίων, θέτουμε τη μεταβλητή ελέγχου ttt ίση με 1, για να ενεργοποιήσουμε τον τερματισμό του προγράμματος στη συνέχεια

```
/// Preparing to send Command to PA10 robot
        pa10->command.axis[1].torque=(t.Data[0][0]/tr_ratio);
        pa10->command.axis[3].torque=(t.Data[1][0]/tr_ratio);
```

Για κάθε άρθρωση περνάμε την τιμή της επιθυμητής ροπής στον ενδιάμεσο χώρο αποθήκευσης της libpa10 μέσω της εντολής:

```
pa10->command.axis[axisno].torque=t.Data[axisno][0]/tr_ratio;
```

που εκφράζει τον προσδιορισμό της ροπής (torque). Η μεταβλητή axisno εκφράζει το δείκτη που καθορίζει την άρθρωση για την οποία γίνεται ο υπολογισμός. Να σημειωθεί ότι η τιμή που αποστέλλουμε στο ρομποτικό βραχίονα είναι πριν από τη μείωση των στροφών, γι' αυτό η ροπή που υπολογίστηκε για κάθε άρθρωση διαιρείται με το λόγο μετάδοσης tr_ratio, που έχει καθοριστεί στην αρχή του προγράμματος.

```
/* Keep Track of Motion | actual position | actual velocity | actual
Torque | y | Computed Friction | JaF | Force Input| x,y position |
*/
     fprintf(output,"%f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f
%f %f\n",qc[1],qc[3], karvel[1],karvel[3],pa10-
>status.axis[1].torque,pa10-
>status.axis[3].torque,t.Data[0][0]/50.0,t.Data[1][0]/50.0,y.Data[0]
[0]/50.0,y.Data[1][0]/50.0,Friction[0]/50.0,Friction[1]/50.0,JaF.Dat
a[0][0]/50,JaF.Data[1][0]/50,Force1.Data[0][0], Force1.Data[1][0],
x.Data[0][0],x.Data[1][0]);
```

Με την παραπάνω εντολή αποθηκεύουμε σε κάθε κύκλο ελέγχου τις μεταβλητές και παραμέτρους του συστήματος, που θέλουμε να μελετήσουμε μετά το πέρας του πειράματος, στο αρχείο output.txt. Συγκεκριμένα αποθηκεύουμε (με τη σειρά που εμφανίζονται στο αρχείο) τη γωνιακή θέση των δύο αρθρώσεων, όπως μας τις επιστρέφει ο ελεγκτής, τη γωνιακή ταχύτητα των δύο αρθρώσεων, όπως μας τις επιστρέφει ο ελεγκτής, τη ροπή στους κινητήρες των δύο αρθρώσεων, όπως μας τις επιστρέφει ο ελεγκτής, τη ροπή που υπολογίσαμε μέσω του νόμου ελέγχου να σταλθεί στους κινητήρες των δύο αρθρώσεων (δια το λόγο μετάδοσης), τη ροπή που υπολογίσαμε και αντιστοιχεί στο νόμο ελέγχου y για τις δύο αρθρώσεις (δια το λόγο μετάδοσης), τη ροπή που υπολογίστηκε μέσω του δυναμικού μοντέλου για την αντιστάθμιση της τριβής στις δύο αρθρώσεις (δια το λόγο μετάδοσης), τη ροπή που υπολογίστηκε μέσω του γινομένου της Ιακωβιανής και των εξωτερικών δυνάμεων για τις δύο αρθρώσεις (δια το λόγο μετάδοσης), τις υπολογισμένες μέσω του αισθητήρα

συνιστώσες των εξωτερικών δυνάμεων κατά x και y άξονα και τις καρτεσιανές συντεταγμένες του τελικού σημείου δράσης σε σχέση με το πλαίσιο της βάσης (κατά x και y άξονα).

```
      if (ttt==1)
           param1.run=0;
```

Εάν η μεταβλητή ελέγχου `ttt` είναι ίση με 1, κάτι το οποίο σημαίνει πως είμαστε εκτός μηχανικών ορίων ταχύτητας ή ροπής, αυτό έχει ως αποτέλεσμα να ορίζεται ως ψευδής (τιμή 0) η συνθήκη `param1.run` πραγματοποίησης του βρόχου και έτσι θα σταματήσει η επανάληψή του και θα οδηγηθούμε σε ομαλό τερματισμό του προγράμματος.

```
//         pa10->command.axis[1].torque=0;
//         pa10->command.axis[3].torque=0;
```

Μέσα σε σχόλια βρίσκονται οι δύο εντολές που μηδενίζουν τις ροπές στις προς έλεγχο αρθρώσεις, πριν αυτές αποσταλούν στον ελεγκτή του ρομπότ. Για την εξακρίβωση της ορθότητας ενός νέου κώδικα ελέγχου του ρομποτικού βραχίονα συνιστάται να βγάζουμε τα σχόλια και να στέλνουμε μηδενικές ροπές στο βραχίονα, ενώ παράλληλα όλες οι μεταβλητές του συστήματος αποθηκεύονται κανονικά στο αρχείο εξόδου. Έτσι μπορούμε να μελετήσουμε τη συμπεριφορά του βραχίονα, χωρίς να τον θέτουμε σε κίνηση, για να ελέγξουμε την ορθότητα των υπολογισμών μας και να πραγματοποιήσουμε οποιοδήποτε πείραμα με ασφάλεια στη συνέχεια.

```
     /* Send the torques to the controller and get current status
back */

     pa10_write(pa10);
```

Η εντολή αυτή αποστέλλει τα δεδομένα που υπολογίστηκαν στον προηγούμενο βρόχο στον ελεγκτή του ρομποτικού βραχίονα για την εφαρμογή (στη συγκεκριμένη περίπτωση) των ροπών στις αρθρώσεις. Να σημειωθεί ότι μέσω της εντολής αυτής ο ελεγκτής μάς επιστρέφει πάλι την κατάσταση των αρθρώσεων την τρέχουσα στιγμή.

```
     /* Check for limit excersion */

     for (axisno = 0; axisno <= 6; axisno++)
          {
          if (fabs(pa10->status.axis[axisno].angle)>limit[axisno])
               {
               param1.run=0;
               printf("Ektos Oriwn Thesis!\n");
               }
          }
```

Με τη χρήση του παραπάνω βρόχου ελέγχουμε για κάθε άρθρωση εάν η γωνιακή θέση, που μας επιστρέφει ο ελεγκτής για κάθε άρθρωση του ρομποτικού βραχίονα, είναι μεγαλύτερη του άνω ορίου ή μικρότερη του κάτω ορίου, χρησιμοποιώντας την εντολή `fabs`, που υπολογίζει την απόλυτη τιμή. Σε περίπτωση που είμαστε εκτός ορίων, ορίζεται ως ψευδής (τιμή 0) η συνθήκη `param1.run` πραγματοποίησης του βρόχου και έτσι θα σταματήσει η επανάληψή του και θα οδηγηθούμε σε ομαλό τερματισμό του προγράμματος. Επίσης τυπώνουμε ένα μήνυμα λάθους, ώστε να μπορούμε να ελέγξουμε το σφάλμα εάν αυτό εμφανιστεί.

```
      if (loopcount >=1*8000) param1.run = 0;
```

Άλλος ένας έλεγχος που ορίζει ως ψευδή (τιμή 0) τη συνθήκη `param1.run` πραγματοποίησης του βρόχου και έτσι σταματάει την επανάληψή του και οδηγεί σε ομαλό τερματισμό του προγράμματος, είναι όταν έχουμε ξεπεράσει έναν καθορισμένο αριθμό επαναλήψεων. Συγκεκριμένα σαν όριο έχουν οριστεί οι 8.000 επαναλήψεις που, με περίοδο κύκλου τα 2.5 ms, αντιστοιχούν σε 20 δευτερόλεπτα λειτουργίας του ελέγχου δύναμης/ροπής.

```
      /* Loop maitenance */
      loopcount++;
      before = now;

  } //// End of Control Loop ////
```

Εδώ είναι το τέλος του βρόχου επανάληψης του ελέγχου δύναμης/ροπής που επιστρέφει στην αρχή, εάν η συνθήκη `param1.run` πραγματοποίησης του είναι αληθής, ή είναι ψευδής και γίνεται εκτέλεση των εντολών που ακολουθούν για τον ομαλό τερματισμό του προγράμματος. Λίγο πριν την επιστροφή στην αρχή του βρόχου ανανεώνουμε το δείκτη μέτρησης επαναλήψεων του βρόχου `loopcount` .

```
//// Stoping Communications and ending /////

printf ("The motion lasted for %d Control Loops\n", loopcount);
printf("Stopping communications\n");
```

Με τις παραπάνω εντολές τυπώνονται στην οθόνη μηνύματα που ενημερώνουν το χρήστη για τον αριθμό των επαναλήψεων που πραγματοποιήθηκαν και πως τερματίζεται η επικοινωνία με τον ελεγκτή του ρομποτικού βραχίονα.

```
if(pa10_stop(pa10)!= 0)
{
     printf("Error stoping communications\n");
}
else
{
     printf("Successfully stopped communications\n\n");
}
```

Εάν η εντολή pa10_stop(pa10) δεν μπορεί να εκτελεστεί για τον τερματισμό της επικοινωνίας, ενημερώνουμε το χρήστη με το αντίστοιχο μήνυμα λάθους, ή εάν ο τερματισμός είναι ομαλός, ενημερώνουμε το χρήστη με ένα μήνυμα επιτυχούς διακοπής της επικοινωνίας.

```


/* Cleanup fd's */

fclose(output);
timer_delete(periodic_timer);
printf("Normal exit\n");
exit(0);

}
```

Με τις τελευταίες εντολές κλείνουμε το αρχείο που δημιουργήσαμε, μηδενίζουμε το χρονομετρητή και τερματίζουμε το πρόγραμμα ενημερώνοντας παράλληλα το χρήστη για τον ομαλό τερματισμό του.

2.4 PA-10.c

```
#include <math.h>
#include <stdlib.h>
#include <stdio.h>
```

Οι εντολές include στην αρχή του κώδικα καθορίζουν τις βιβλιοθήκες και τα υποπρογράμματα που είναι απαραίτητα για την πραγματοποίηση κάποιων λειτουργιών. Συγκεκριμένα, για τη δεδομένη εφαρμογή, εκτός των βασικών stdlib.h και stdio.h είναι απαραίτητη και η σύνδεση με τη βιβλιοθήκη lm μέσω του κεφαλικού αρχείου math.h για την πραγματοποίηση συγκεκριμένων μαθηματικών πράξεων, όπως τριγωνομετρικοί υπολογισμοί.

```
    float L1=0.45,L2=0.55,d2=0.052;
```

Με αυτή την εντολή καθορίζονται οι διαστάσεις του ρομποτικού βραχίονα, με βάση τις οποίες θα υπολογιστούν η ορθή κινηματική, η αντίστροφη κινηματική και η Ιακωβιανή του συστήματος. Έχοντας θέσει το πλαίσιο της βάσης στον άξονα της πρώτης λειτουργικής άρθρωσης (άρθρωση 2 του PA-10), το L1 είναι η απόστασή της από τη δεύτερη λειτουργική άρθρωση (άρθρωση 4 του PA-10), το L2 είναι η απόσταση της δεύτερης λειτουργικής άρθρωσης από το τελικό σημείο δράσης του ρομποτικού βραχίονα και το d2 είναι η επιπλέον απόσταση, που καθορίζεται από τον αισθητήρα και τις φλάντζες σύνδεσης μέχρι το πραγματικό τελικό σημείο δράσης της εφαρμογής. Η τελευταία παράμετρος d2 είναι αυτή που μπορεί να χρειαστεί να μεταβληθεί, ανάλογα με τις τροποποιήσεις της διάταξης από το χρήστη.

```
void dir_kinematics() {

    x_[0]=L1*cos(q[0])+(L2+d2)*cos(q[1]+q[0]);
    x_[1]=L1*sin(q[0])+(L2+d2)*sin(q[1]+q[0]);

}
```

Η παραπάνω υπορουτίνα υπολογίζει τις συντεταγμένες x_ του τελικού σημείου δράσης, σε σχέση με τις γωνίες q των δύο λειτουργικών αρθρώσεων και τα μήκη L1, L2 και d2 που καθορίστηκαν παραπάνω.

```
void inv_kinematics() {

    double q11,q12,qtot,A,B,C;
    A=-2*L1*x_[1];
    B=-2*L1*x_[0];
    C=(L2+d2)*(L2+d2)-L1*L1-x_[0]*x_[0]-x_[1]*x_[1];
```

Στην υπορουτίνα της αντίστροφης κινηματικής, σε σχέση με τις συντεταγμένες του τελικού σημείου δράσης και των διαστάσεων του συστήματος, υπολογίζονται αρχικά οι ενδιάμεσες μεταβλητές `A`, `B` και `C`.

```
    q11=-atan2(B,A)+atan2(C/sqrt(A*A+B*B),sqrt(1-C*C/(A*A+B*B)));
    q12=-atan2(B,A)+atan2(C/sqrt(A*A+B*B),-sqrt(1-C*C/(A*A+B*B)));
```

Στη συνέχεια εντοπίζονται οι δύο διαφορετικές λύσεις για τη γωνιακή θέση της πρώτης άρθρωσης.

```
    if (q[0]/q11>0) q[0]=q11;
       else q[0]=q12;
```

Καθώς ο υπολογισμός της ορθής κινηματικής έχει προηγηθεί διαιρώντας την προηγούμενη τιμή που έχουμε με την πρώτη λύση, και εφόσον έχουμε θετικό αποτέλεσμα (άρα είναι οι δύο αυτές τιμές ομόσημες), επιλέγουμε την πρώτη λύση, αλλιώς τη δεύτερη (που σαν αντίθετη με την πρώτη θα είναι ομόσημη με την παλιά λύση).

```
    qtot=atan2(x_[1]-L1*sin(q[0]),x_[0]-L1*cos(q[0]));
    q[1]=qtot-q[0];

}
```

Τέλος, υπολογίζουμε τη συνολική γωνία του βραχίονα και, αφαιρώντας την τιμή της πρώτης άρθρωσης, υπολογίζουμε και την τιμή της πρώτης.

```
void Jacobian() {

    J[0][0]=-L1*sin(q[0])-(L2+d2)*sin(q[0]+q[1]);
    J[0][1]=-(L2+d2)*sin(q[0]+q[1]);
    J[1][0]=L1*cos(q[0])+(L2+d2)*cos(q[0]+q[1]);
    J[1][1]=(L2+d2)*cos(q[0]+q[1]);

}
```

Η παραπάνω υπορουτίνα υπολογίζει τα στοιχεία της Ιακωβιανής `J` του συστήματος σε σχέση με τις γωνίες `q` των δύο λειτουργικών αρθρώσεων και τα μήκη `L1`, `L2` και `d2` που καθορίστηκαν στην αρχή.

Γ. Εγχειρίδιο

Γ.1 Εγχειρίδιο λογισμικού

Υποπρόγραμμα: **global_variables.c**

Λειτουργία: Χρησιμεύει στον ορισμό των μεταβλητών που έχουν κοινή χρήση από πολλά υποπρογράμματα του κώδικα.

Σύνδεση με το κυρίως πρόγραμμα: Τοποθετούμε το αρχείο *global_variables.c* στον ίδιο φάκελο με το κυρίως πρόγραμμα και προσθέτουμε την αντίστοιχη εντολή include στην αρχή του κυρίως κώδικα, για να το ενσωματώσουμε σε αυτόν

```
#include "global_variables.c"
.
.
```

Σημείωση: Η εντολή αυτή πρέπει να προηγείται όλων των υπολοίπων εντολών include που αναφέρονται στα υποπρογράμματα που χρησιμοποιούν αυτές τις κοινές μεταβλητές, ώστε να μπορούν να τις αναγνωρίζουν.

Υποπρόγραμμα: **PA10.cpp**

Λειτουργία: Υπολογισμός της ορθής κινηματικής του PA-10 ρομπότ 7 βαθμών ελευθερίας καθώς και της Ιακωβιανής του συστήματος.

Σύνδεση με το κυρίως πρόγραμμα: Τοποθετούμε τα αρχεία *PA10.cpp* και *PA10.h* στον ίδιο φάκελο με το κυρίως πρόγραμμα και προσθέτουμε: Εντολές στην αρχή του κώδικα, για να ενσωματώσουμε σε αυτόν τις βοηθητικές υπορουτίνες του ρομπότ (εντολές include), και ορίζουμε τους πίνακες που θα χρειαστούμε σαν είσοδο και έξοδο κατά τη συνεργασία του κεντρικού κώδικα με τον κώδικα PA10.cpp.

```
#include "PA10.h"
#include "PA10.cpp"
.
int main ()
{
     PA10 pa1;
     double angles1[7];
     double position1[3];
     double orientation1[3];
.
}
```

Κλήση υπορουτινών και λήψη αποτελεσμάτων:

1)Ορθή κινηματική: Ουσιαστικά γίνεται χρήση δύο υπορουτινών, της πρώτης με είσοδο τις γωνίες (`angles1`) των αρθρώσεων των 7 βαθμών ελευθερίας του ρομπότ, για τον υπολογισμό της ορθής κινηματικής, και της δεύτερης για την επιστροφή στον κεντρικό κώδικα των τιμών της θέσης και του προσανατολισμού του τελικού σημείου σε δύο ξεχωριστά διανύσματα (`position1,orientation1`), που μπορούμε στη συνέχεια να χρησιμοποιήσουμε.

```
pa1.PA10_For_Kine_Comp(angles1);
pa1.PA10_GetPose(position1, orientation1);
```

2)Υπολογισμός της Ιακωβιανής: Ορίζουμε έναν πίνακα με διαστάσεις 6x7 για την τοποθέτηση των στοιχείων της Ιακωβιανής μέσω της εντολής `Matrix` (για περαιτέρω βλ. την ανάλυση του υποπρογράμματος *Matrix.cpp*), έστω τον πίνακα Ja, και καλούμε την αντίστοιχη υπορουτίνα του PA10.cpp (προτείνεται η δεύτερη από τις δύο επιλύσεις). Για τον υπολογισμό της Ιακωβιανής πρέπει να έχει κληθεί προηγουμένως η υπορουτίνα της ορθής κινηματικής του συστήματος.

```
Matrix Ja(6,7);
.
.
pa1.PA10_Jacobian_Analytic_2(&Ja);
```

Υποπρόγραμμα: **PA10.c**

Λειτουργία: Υπολογισμός της ορθής κινηματικής, την αντίστροφης κινηματικής, καθώς και της Ιακωβιανής του συστήματος *του απλοποιημένου PA-10 ρομπότ*, όπου έχουν μπει φρένα σε όλες τις αρθρώσεις, πλην των αρθρώσεων 2 και 4.

Σύνδεση με το κυρίως πρόγραμμα: Τοποθετούμε τα αρχεία *PA10.c* και το αρχείο *PA10.h* που του αντιστοιχεί (προσοχή είναι διαφορετικό αρχείο σε σχέση με την υπορουτίνα PA10.cpp) στον ίδιο φάκελο με το κυρίως πρόγραμμα και προσθέτουμε: Εντολές στην αρχή του κώδικα για να ενσωματώσουμε σε αυτόν τις βοηθητικές υπορουτίνες του ρομπότ (εντολές include).

```
#include "PA10.h"
#include "PA10.c"
.
.
```

Κλήση υπορουτινών και λήψη αποτελεσμάτων:

1)Ορθή κινηματική: Αποθηκεύουμε στο κοινής χρήσεως διάνυσμα q τις γωνιακές θέσεις των αρθρώσεων (έστω τις τιμές του Matrix q1, όπως στο παράδειγμα που ακολουθεί). Στη συνέχεια ακολουθεί η εντολή dir_kinematics(), που υπολογίζει τη θέση του τελικού σημείου δράσης και την αποθηκεύει στο κοινής χρήσεως διάνυσμα x_, το οποίο μπορούμε στη συνέχεια να περάσουμε σε ένα τοπικό διάνυσμα (στο παράδειγμα το Matrix x).

```
     q[0]=q1.Data[0][0];
     q[1]=q1.Data[1][0];
     dir_kinematics();
     x.Data[0][0]=x_[0];
     x.Data[1][0]=x_[1];
```

2)Αντίστροφη κινηματική: Αποθηκεύουμε στο κοινής χρήσεως διάνυσμα x_ τη θέση του τελικού σημείου δράσης (έστω τις τιμές του Matrix xd, όπως στο παράδειγμα που ακολουθεί). Στη συνέχεια ακολουθεί η εντολή inv_kinematics(), που υπολογίζει τις γωνιακές θέσεις των αρθρώσεων και τις αποθηκεύει στο κοινής χρήσεως διάνυσμα q, το οποίο μπορούμε στη συνέχεια να περάσουμε σε ένα τοπικό διάνυσμα (στο παράδειγμα το Matrix q1).

```
    x_[0]=xd.Data[0][0];  //Input Data
    x_[1]=xd.Data[1][0];  //Input Data
    inv_kinematics();
    q1.Data[0][0]=q[0];   //Output Data
    q1.Data[1][0]=q[1];   //Output Data
```

3)Υπολογισμός της Ιακωβιανής: Ορίζουμε έναν πίνακα με διαστάσεις 2x2 για την τοποθέτηση των στοιχείων της Ιακωβιανής μέσω της εντολής Matrix (για περαιτέρω βλ. την ανάλυση του υποπρογράμματος *Matrix.cpp*), έστω τον πίνακα Ja, και καλούμε την αντίστοιχη υπορουτίνα του PA10.c. Για τον υπολογισμό της Ιακωβιανής πρέπει να έχουμε τοποθετήσει στο κοινής χρήσεως διάνυσμα q τις γωνιακές θέσεις των αρθρώσεων, όπως στην υπορουτίνα της ορθής κινηματικής του συστήματος.

```
Jacobian();
    for ( int i = 0; i < 2; i++ )  {
        for ( int r = 0; r < 2; r++ ) Ja.Data[i][r]=J[i][r];
    }
```

Υποπρόγραμμα: **JR3_force.c**

Λειτουργία: Εκκίνηση λειτουργίας και λήψη και επεξεργασία τιμών δύναμης και ροπής μέσω του αισθητήρα JR3.

Σύνδεση με το κυρίως πρόγραμμα: Τοποθετούμε τα αρχεία *JR3_force.c* και *JR3_force.h* στον ίδιο φάκελο με το κυρίως πρόγραμμα και τα ενσωματώσουμε στον κυρίως κώδικα. (Η ορθή λειτουργία του υποπρογράμματος απαιτεί να τοποθετήσουμε στον ίδιο φάκελο και τα αρχεία nrutil.c, nrutil.h και jr3pci-ioctl.h) Για την αποθήκευση των τιμών δύναμης/ροπής σε ένα διάνυσμα 6x1 (`Force[6]`) και την είσοδο στην υπορουτίνα των τιμών των γωνιών Euler (`alfa, beta, gama` περιστροφή ως προς άξονα x, y, z αντίστοιχα) τοποθετούμε στον ίδιο φάκελο και ενσωματώνουμε στον κυρίως κώδικα και το αρχείο *global_variables.c (βλ. ανάλυση της global_variables.c)* . Τέλος, για την πραγματοποίηση πράξεων με το μητρώο των δυνάμεων ροπών ορίζουμε μέσω της εντολής `Matrix` (για περαιτέρω βλ. την ανάλυση του υποπρογράμματος *Matrix.cpp*) το μητρώο Force1.

```
#include "global_variables.c"
#include "JR3_force.h"
#include "JR3_force.c"
.
.
Matrix Force1(6,1);
```

Κλήση υπορουτινών και λήψη αποτελεσμάτων:

*Βήμα 1. Εκκίνηση λειτουργίας του αισθητήρα: Π*ρέπει να προηγείται ο υπολογισμός της ορθής κινηματικής του συστήματος (για περαιτέρω βλ. την ανάλυση του υποπρογράμματος *PA10.cpp*) και να έχουν εξισωθεί οι γωνίες Euler του συστήματος με αυτές της εισόδου στην υπορουτίνα με τις εξής εντολές:

```
alfa=orientation1[0];
beta=orientation1[1];
gama=orientation1[2];
```

Η εντολή εκκίνησης του αισθητήρα τοποθετείται μια φορά στην αρχή του προγράμματος και εκκινεί τη συσκευή του αισθητήρα αρχικοποιώντας τις τιμές των δυνάμεων και ροπών αντισταθμίζοντας τις αρχικές φορτίσεις

```
JR3_Start_Device();
```

Βήμα 2. Λήψη και χρήση επεξεργασμένων τιμών: Πριν τον υπολογισμό των φορτίσεων πρέπει να προηγείται ο υπολογισμός της ορθής κινηματικής του συστήματος (για περαιτέρω βλ. την ανάλυση του υποπρογράμματος *PA10.cpp*) και να έχουν εξισωθεί οι γωνίες Euler του συστήματος με αυτές της εισόδου στην υπορουτίνα με τις εξής εντολές:

```
alfa=orientation1[0];
beta=orientation1[1];
gama=orientation1[2];
```

Η λήψη και επεξεργασία των τιμών των δυνάμεων και ροπών μέσω του αισθητήρα πραγματοποιείται και αποθηκεύεται στο κοινής χρήσεως διάνυσμα `Force` μέσω της ακόλουθης εντολής:

```
JR3_Get_Force();
```

Σημείωση: Ο έλεγχος, εάν η ληφθείσα τιμή βρίσκεται εντός ορίων της περιοχής τιμών του αισθητήρα που βαθμονομήθηκε, γίνεται μέσω του διανύσματος 6x1 (`calibr[6]`) που είναι ορισμένο μέσω της βοηθητικής υπορουτίνας *global_variables.c* και το οποίο για κάθε βαθμό ελευθερίας του αισθητήρα (με τη σειρά Fx, Fy, Fz, Mx, My, Mz) επιστρέφει την τιμή μηδέν (0), όταν για την αντίστοιχη μέτρηση βρισκόμαστε εντός του πεδίου βαθμονόμησης, και την τιμή ένα (1), όταν βρισκόμαστε εκτός αυτού.

Υποπρόγραμμα: **getmatrices.c**

Λειτουργία: Επεξεργασία και λήψη μητρώων του δυναμικού μοντέλου του Pa-10 ρομπότ.

Σύνδεση με το κυρίως πρόγραμμα: Τοποθετούμε τα αρχεία getmatrices.h και getmatrices.c στον ίδιο φάκελο με το κυρίως πρόγραμμα και τα ενσωματώνουμε στον κυρίως κώδικα. Για την αποθήκευση των απαραίτητων μητρώων για κοινή χρήση, τοποθετούμε στον ίδιο φάκελο και ενσωματώνουμε στον κυρίως κώδικα και το αρχείο *global_variables.c (βλ. ανάλυση του global_variables.c)*. Στο ίδιο αρχείο ορίζονται και τα διανύσματα γωνίας και γωνιακής ταχύτητας q[7], qdot[7], που πρέπει να είναι υπολογισμένα πριν ξεκινήσει η υπορουτίνα,

καθώς είναι η είσοδος του συστήματος. Στη συνέχεια για την πραγματοποίηση πράξεων μεταξύ των μητρώων του δυναμικού μοντέλου ορίζουμε μέσω της εντολής `Matrix` (για περαιτέρω βλ. την ανάλυση του υποπρογράμματος *Matrix.cpp*) μητρώα, που πρέπει να εξισώσουμε με τα αντίστοιχα του δυναμικού μοντέλου. Το μητρώο A αναφέρεται στο μητρώο αδράνειας, το μητρώο Cmat στο μητρώο των φυγόκεντρων ροπών και των ροπών Coriolis, το G στο μητρώο βαρύτητας και το Friction στο μητρώο τριβών που έχει ενσωματωμένο και το μητρώο δυσκαμψίας. Συνολικά, οι αρχικές εντολές που πρέπει να προσθέσουμε στον κεντρικό κώδικα είναι οι εξής:

```
#include "global_variables.c"
#include "getmatrices.h"
#include "getmatrices.c"
.
.
int main ()
{
     Matrix A1(7,7);
     Matrix Cmat1(7,7);
     Matrix G1(7,1);
     Matrix Friction1(7,1);
.
.
```

Κλήση υπορουτίνας και λήψη αποτελεσμάτων:

1)Υπολογισμός μητρώων δυναμικού μοντέλου: Για την εκκίνηση της διαδικασίας (αφού ήδη έχουν υπολογιστεί τα q[7], qdot[7]) χρησιμοποιείται η ακόλουθη εντολή:

```
getmatrices();
```

Και για την αποθήκευση των μητρώων του δυναμικού μοντέλου σε πίνακες του κεντρικού προγράμματος, μέσω των οποίων μπορούμε να κάνουμε πράξεις χάρη στην υπορουτίνα Matrix.c, χρησιμοποιούμε ένα βρόχο παρόμοιο με τον εξής:

```
for ( int i = 0; i < 7; i++ ) {
        G1.Data[i][0]=G[i];
        Friction1.Data[i][0]=Friction[i];
        for ( int j = 0; j < 7; j++ ) {
            A1.Data[i][j]=A[i][j];
            Cmat1.Data[i][j]=Cmat[i][j];
            }
    }
```

Το αντίστοιχο αρχείο **getmatrices.c**, που βρίσκεται στον υποφάκελο Fcontrol2dof και αντιστοιχεί στον υπολογισμό του δυναμικού μοντέλου για το απλοποιημένο σύστημα του PA-

10 (που χρησιμοποιούμε μόνο τους 2 βαθμούς ελευθερίας), υπολογίζει μόνο το κομμάτι που αντιστοιχεί στην τριβή *(αφού ήδη έχουν υπολογιστεί τα* qdot[2] που μας χρειάζονται μόνο), γι' αυτό απαιτούνται μόνο οι εντολές:

```
getmatrices();
    Friction1.Data[0][0]=Friction[0];
    Friction1.Data[1][0]=Friction[1];
```

Με αποτέλεσμα να αποθηκεύουμε στο διάνυσμα `Friction1` (που έχει κατασκευαστεί μέσω του υποπρογράμματος *Matrix.cpp*) την τριβή που πρέπει να αντισταθμιστεί για κάθε μία από τις δύο λειτουργικές μας αρθρώσεις (αρθρώσεις 2 και 4 του ρομποτικού βραχίονα)

Υποπρόγραμμα: **Matrix.cpp**

Λειτουργία: Κατασκευή, πρόσθεση, αφαίρεση, κλιμάκωση, αναστροφή, πολλαπλασιασμός, υπολογισμός ορίζουσας, αντιστροφή και ψευδοαντιστροφή πινάκων.

Σύνδεση με το κυρίως πρόγραμμα: Τοποθετούμε τα αρχεία Matrix.cpp και Matrix.h στον ίδιο φάκελο με το κυρίως πρόγραμμα και τα ενσωματώσουμε στον κυρίως κώδικα:

```
#include "Matrix.h"
#include "Matrix.cpp"
```

Κλήση υπορουτινών και λήψη αποτελεσμάτων:

1)Κατασκευή πινάκων: Έστω πίνακας A διαστάσεων 5x3

```
Matrix A(5,3);
```

2)Πρόσθεση πινάκων: Έστω πίνακες A και B που προστίθενται με αποτέλεσμα τον πίνακα C:

```
C.add(&A, &B);
```

3)Αφαίρεση πινάκων: Έστω πίνακες A και B με τον B να αφαιρείται από τον A με αποτέλεσμα τον πίνακα C:

```
C.sub(&A, &B);
```

4)Κλιμάκωση πίνακα: Έστω πίνακας A, όπου όλα τα στοιχεία του πολλαπλασιάζονται με την τιμή x και αποτέλεσμα είναι ο πίνακας C

```
C.scale(&A, x);
```

5)Αναστροφή πίνακα: Έστω πίνακας A με ανάστροφο του τον A_T

```
A_T.trans(&A);
```

6)Πολλαπλασιασμός πινάκων: Έστω πίνακες A και B πολλαπλασιάζονται με αποτέλεσμα των πίνακα A_B

```
A_B.mult(&A, &B);
```

7)Αντιστροφή πίνακα: Έστω πίνακας A που αντιστρέφεται με αποτέλεσμα τον πίνακα A_Inv

```
A.invert_mat(&A_Inv);
```

8)Ψευδοαντιστροφή πίνακα: Έστω πίνακας A που ψευδοαντιστρέφεται με αποτέλεσμα τον πίνακα A_psInv

```
A.r_pseudo_inverse(&A_psInv);
```

Γ.2 Μετάφραση και χρήση λογισμικού ελέγχου σε Linux

Βήμα 1ο: Εκκινούμε τον υπολογιστή που συνδέεται με τον ελεγκτή του ρομποτικού βραχίονα και στο παράθυρο επιλογής λειτουργικού συστήματος επιλέγουμε το Linux HRT. Παράλληλα ανοίγουμε τον ελεγκτή του ρομπότ μέσω του κεντρικού κουμπιού.

Βήμα 2ο: Έπειτα από την ολοκλήρωση της εκκίνησης του λειτουργικού συστήματος, εισάγουμε το όνομα χρήστη και τον κωδικό του διαχειριστή του συστήματος (έτσι ώστε να έχουμε πρόσβαση και στον κεντρικό φάκελο root του συστήματος).

Βήμα 3ο: Με την εντολή `ls` βλέπουμε τους υποφακέλους που περιέχονται στο φάκελο στον οποίο βρισκόμαστε και με την εντολή `cd` *`όνομα_φακέλου`* εισερχόμαστε σε έναν από αυτούς.

Για τη χρήση και επεξεργασία του λογισμικού που έχει κατασκευαστεί στην παρούσα εργασία, εφόσον βρισκόμαστε στον φάκελο root πληκτρολογούμε διαδοχικά `cd aimilios` και `cd ForceControl` για να χειριστούμε τον έλεγχο δύναμης/ροπής και για τους 7 βαθμούς ελευθερίας (ή αντίστοιχα `cd FControl2dof` για το αντίστοιχο προσαρμοσμένο σε 2 βαθμούς ελευθερίας).

Βήμα 4ο: Πληκτρολογώντας `vi` *`όνομα_αρχείου`* μπορούμε να επεξεργαστούμε οποιοδήποτε κείμενο με χρήση του επεξεργαστή κειμένου του Linux.

π.χ
```
psara ForceControl # : vi ForceControl.cpp
```

Βήμα 5ο: Εάν θέλουμε να μεταφράσουμε ένα αρχείο σε γλώσσα προγραμματισμού C ή C++, χρησιμοποιούμε την εντολή `gcc` ή `g++` αντίστοιχα. Κατά τη μετάφραση αντίστοιχων προγραμμάτων ελέγχου πρέπει να συνδέσουμε με το λογισμικό μας και τις βιβλιοθήκες `libpa10, libthread, libposixtime` και `lm`.
Η πλήρης εντολή είναι η εξής (για C αντίστοιχα με `gcc`):

```
g++  αρχείο_κώδικα.c –ο αρχείο_προγράμματος -libpa10
     -libthread -libposixtime -lm
```

Παράληψη της σύνδεσης με τις παραπάνω βιβλιοθήκες θα προκαλέσει αποτυχία της μετάφρασης.

Βήμα 6ο: Εάν θέλουμε να έχει ο ρομποτικός μας βραχίονας μια συγκεκριμένη διάταξη κατά την εκκίνηση κάποιου πειράματος πληκτρολογούμε την εντολή `position`. Τότε ανοίγει ένα πρόγραμμα παραθυρικής μορφής, στο οποίο μπορούμε να ρυθμίσουμε τη γωνιακή θέση κάθε άρθρωσης και να την πραγματοποιήσουμε εκτελώντας το Execute, που βρίσκεται σαν κουμπί στο περιβάλλον του προγράμματος. Προσοχή πρέπει να δοθεί στην παρατήρηση ότι έπειτα από την εντολή κίνησης πρέπει όλες οι τιμές να έχουν φτάσει στην επιθυμητή τιμή (πράσινο χρώμα) και δεν υπάρχει ακόμα άρθρωση που κινείται (κίτρινο χρώμα). Στην τελευταία περίπτωση επανεκκινούμε τον ελεγκτή και ξανατρέχουμε το πρόγραμμα `position`.

Βήμα 7ο: Εφόσον έχουμε μεταφράσει ήδη το λογισμικό μας και βρισκόμαστε στο φάκελο όπου έχει αποθηκευτεί το τελικό πρόγραμμα, πληκτρολογώντας `./όνομα_προγράμματος` εκτελούμε το πρόγραμμα αυτό.

Μεγάλη προσοχή πρέπει να δοθεί στον πολλαπλό έλεγχο ενός προγράμματος ελέγχου του ρομποτικού βραχίονα, τόσο για λογικά σφάλματα, όσο για θέτηση δικλείδων ασφαλείας στο πιθανό ενδεχόμενο σε κάποια φάση της λειτουργίας ελέγχου του ο ρομποτικός βραχίονας να βρίσκεται κοντά στα μηχανικά του όρια. Η προσέγγιση του βραχίονα εν λειτουργία πρέπει να γίνεται με προσοχή και μόνο αν και εφόσον είναι απαραίτητη η συμμετοχή ανθρώπου σε μία πειραματική διεργασία.

Τέλος, πρέπει πάντα να υπάρχει ένα άτομο που να έχει το κουμπί εκτάκτου ανάγκης ανά χείρας, σε περίπτωση που εκτιμηθεί πιθανός κίνδυνος για τη διάταξη ή τους συμμετέχοντες ανθρώπους.

Σημείωση: Αν θέλουμε να αντιγράψουμε ένα αρχείο από κάποιον άλλο υπολογιστή στο κεντρικό υπολογιστή του ρομποτικού βραχίονα, πραγματοποιούμε την εξής διαδικασία:

Έχοντας ανοίξει και τους δύο υπολογιστές και έχοντας «φορτώσει» λειτουργικό πρόγραμμα Linux και στους δύο υπολογιστές, χρησιμοποιούμε την εντολή `scp`. Πληκτρολογούμε για παράδειγμα από τον υπολογιστή του ρομπότ την εξής εντολή:

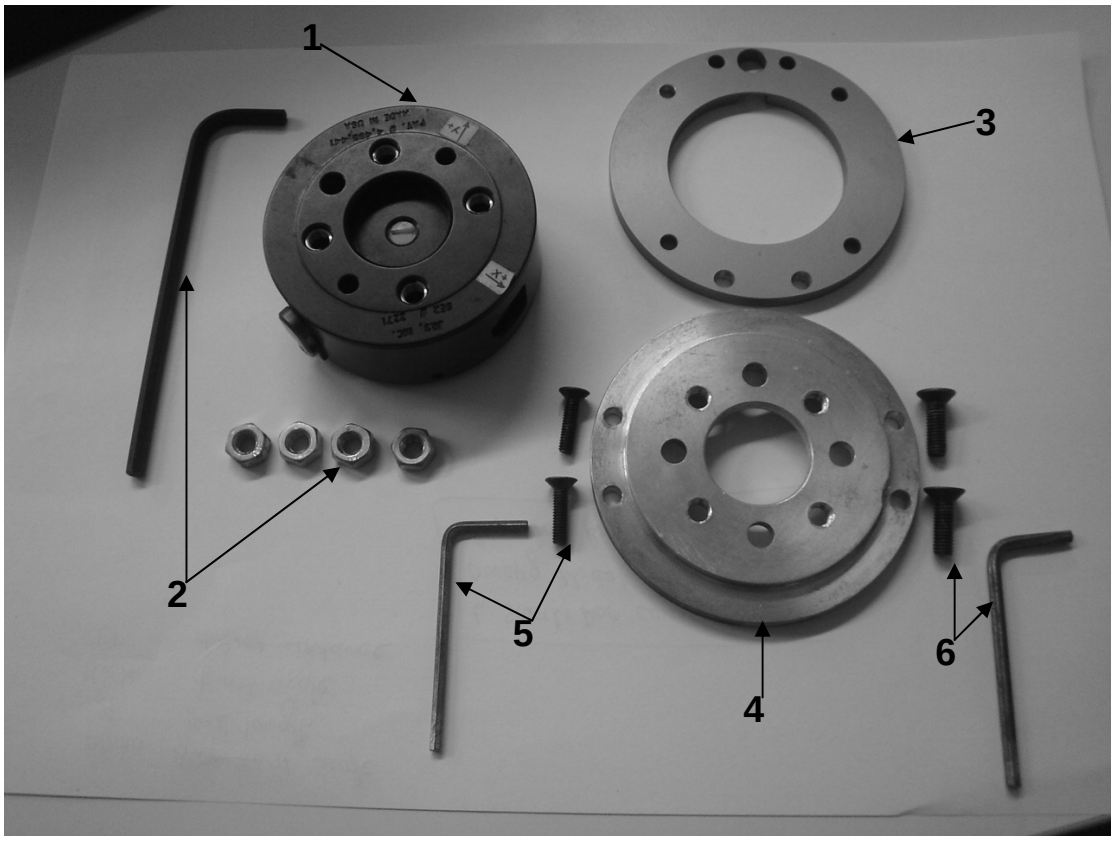
```
scp root@santorini:/root/Home/Fcontrol.c  /root/aimilios
```

και όταν μας ζητηθεί εισάγουμε τον κωδικό πρόσβασης του υπολογιστή στον οποίο θέλουμε να συνδεθούμε (εδώ ο υπολογιστής είναι ο `santorini`) με βάση τον χρήστη που δηλώσαμε στην εντολή (εδώ είναι ο `root`).

Στο συγκεκριμένο παράδειγμα δηλαδή επικοινωνούμε με το χρήστη `root` του υπολογιστή `santorini`, παίρνουμε το αρχείο `Fcontrol.c` που βρίσκεται στο φάκελο `/root/Home/` και το περνάμε στο φάκελο `/root/aimilios` του υπολογιστή που βρισκόμαστε.

Γ.3 Σύνδεση εξαρτημάτων και πραγματοποίηση πειραμάτων

Βήμα 1ο: Συγκέντρωση εξαρτημάτων

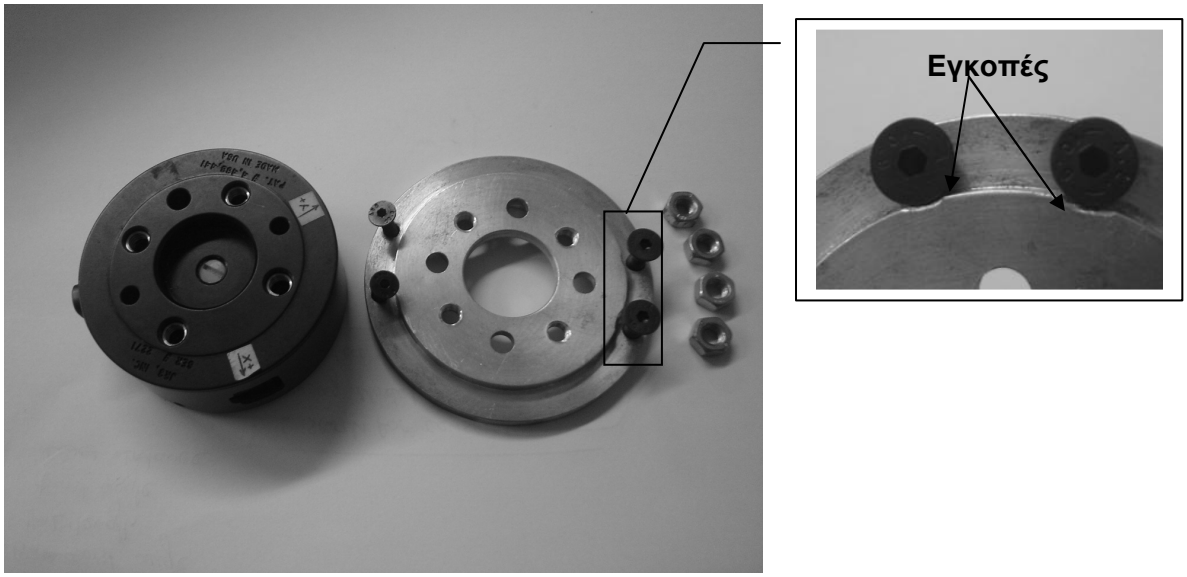


Εικόνα 14. Σύνολο εξαρτημάτων για τη σύνδεση του αισθητήρα στο ρομποτικό βραχίονα.

Για τη σύνδεση του αισθητήρα JR3 με τον ακροδέκτη του ρομποτικού βραχίονα PA-10 χρειαζόμαστε τα εξής εξαρτήματα:

1. Τον αισθητήρα δύναμης/ροπής JR3 μοντέλο 67M25A
2. Τέσσερα (4) περικόχλια M6 και το αντίστοιχο κατσαβίδι άλλεν για κοχλία άλλεν M6
3. Την ενδιάμεση φλάντζα του ρομπότ που λειτουργεί ως αποστάτης
4. Τη φλάντζα σύνδεσης του αισθητήρα JR3
5. Δύο (2) κοχλίες άλλεν M4 με κωνική κεφαλή μήκους 12mm και το αντίστοιχο κατσαβίδι άλλεν
6. Δύο (2) κοχλίες άλλεν M5 με κωνική κεφαλή μήκους 12mm και το αντίστοιχο κατσαβίδι άλλεν

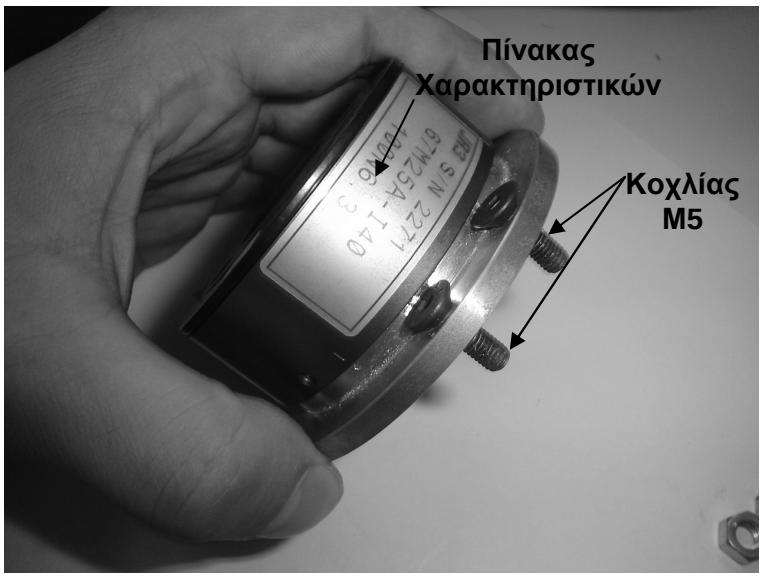
Βήμα 2ο: Τοποθέτηση κοχλιών



Εικόνα 15. Αρχική τοποθέτηση κοχλιών.

Τοποθετούμε πρώτα τους κοχλίες στη φλάντζα σύνδεσης πριν προσαρμόσουμε τον αισθητήρα ανά δύο σε κάθε πλευρά με τους μεγαλύτερους κοχλίες (M5) από την πλευρά που έχει χαραχτεί και η κατάλληλη εγκοπή για την ομαλή εισαγωγή της βίδας λόγω της μεγαλύτερης κεφαλής, όπως φαίνεται καλύτερα στη μεγενθυμένη λεπτομέρεια.

Βήμα 3ο: Τοποθέτηση αισθητήρα

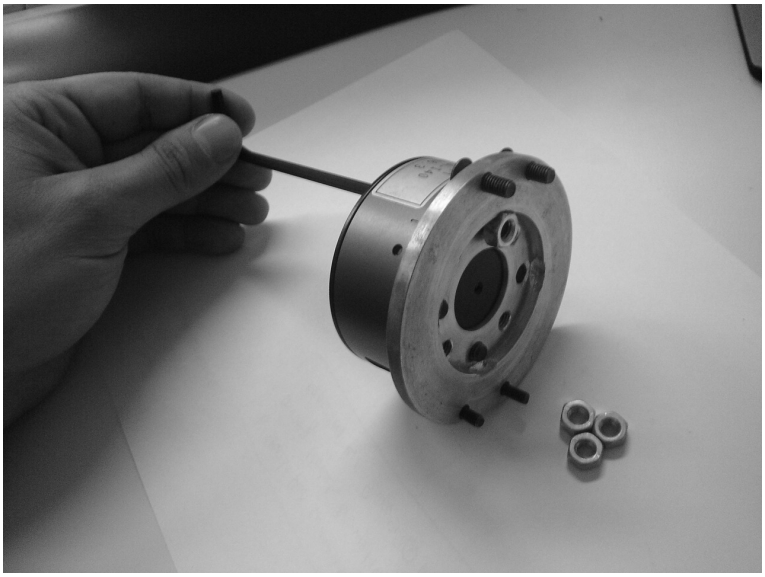


Εικόνα 16. Φορά τοποθέτησης αισθητήρα στη φλάντζα σύνδεσης.

Τοποθετούμε τον αισθητήρα από την επάνω πλευρά της φλάντζας, έτσι ώστε οι ενσωματωμένοι κοχλίες της συσκευής να συμπέσουν με τις κανονικές οπές διαμέτρου 6mm

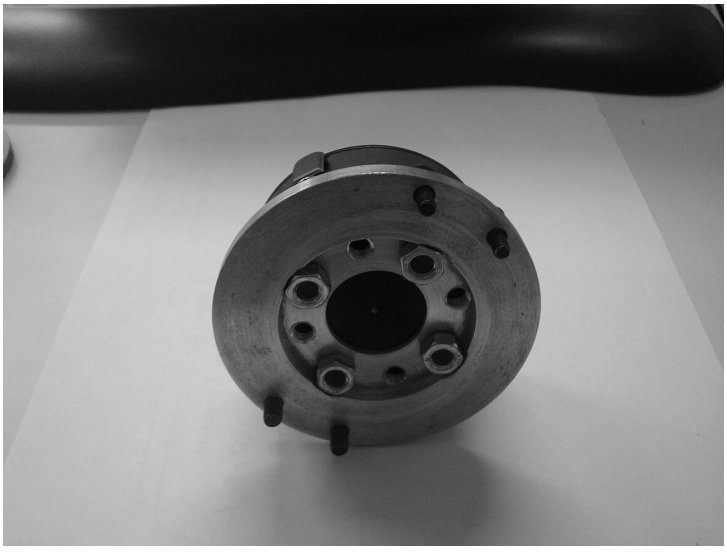
της φλάντζας (και όχι με τις οπές με σπείρωμα, η σύνδεση θα γίνει με περικόχλια). Το κομμάτι του αισθητήρα με τον πίνακα των χαρακτηριστικών πρέπει να συμπίπτει με την πλευρά των κοχλιών Μ5, έτσι ώστε να εξασφαλιστεί και η αντίστοιχη τελική συνδεσμολογία που πραγματοποιήθηκε στην παρούσα εργασία.

Βήμα 4ο: Σύνδεση αισθητήρα και φλάντζας με τη χρήση περικοχλίων



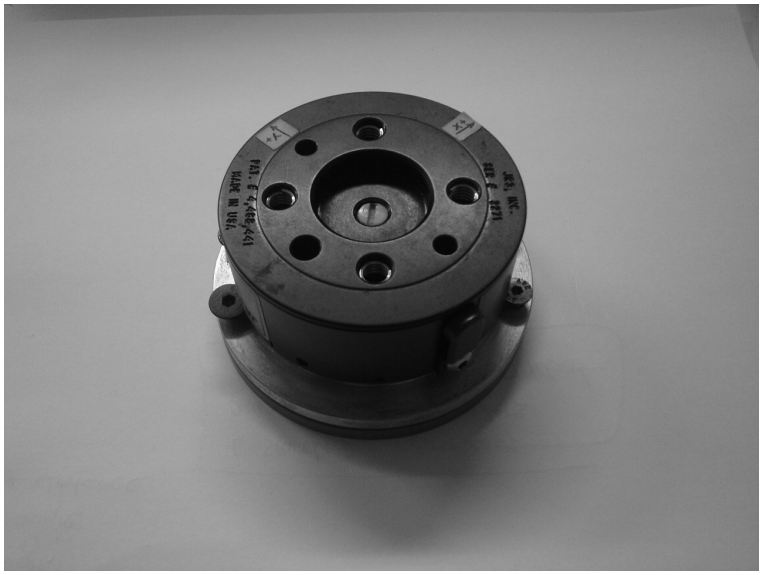
Εικόνα 17. Σύσφιξη αισθητήρα και φλάντζας μέσω περικοχλίων.

Με τη βοήθεια του άλλεν για κοχλία Μ6 μέσω των οπών από την πάνω πλευρά του αισθητήρα, και στηρίζοντας προσεκτικά το περικόχλιο Μ6 στην κάτω πλευρά της φλάντζας βιδώνουμε την εσωτερική βίδα του αισθητήρα στο περικόχλιο. Επαναλαμβάνουμε τη διαδικασία αυτή και για τις υπόλοιπες κοχλιώσεις, με αποτέλεσμα που παρουσιάζεται στην επόμενη εικόνα:



Εικόνα 18. Τελική σύνδεση αισθητήρα και φλάντζας.

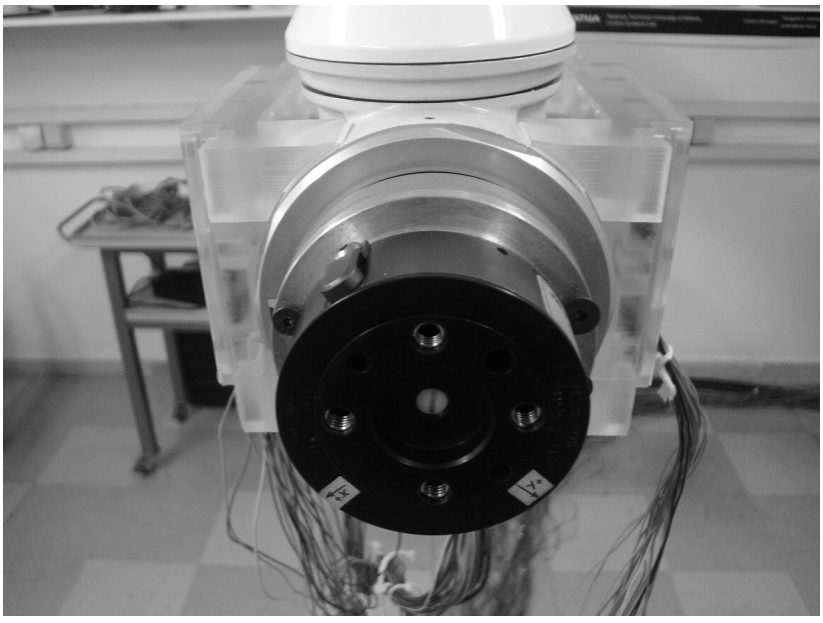
Βήμα 5ο: Προσαρμογή ενδιάμεσης φλάντζας



Εικόνα 19. Τοποθέτηση ενδιάμεσης φλάντζας.

Προσαρμόζουμε την ενδιάμεση φλάντζα κάτω από τη φλάντζα σύνδεσης στους τέσσερις κοχλίες.

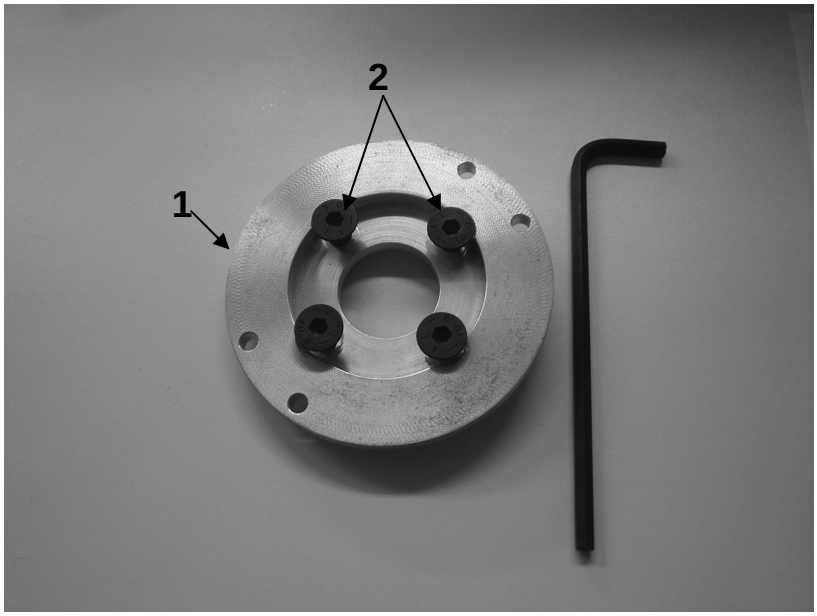
Βήμα 6ο: Τελική σύνδεση στον ακροδέκτη του ρομποτικού βραχίονα



Εικόνα 20. Προσαρμογή του συστήματος φλάντζας-αισθητήρα στο ρομπότ.

Δεν υπάρχει πρόβλημα για την πλευρά που θα βιδώσουμε, καθώς η διαφορετική κοχλίωση που έχουμε στις δύο πλευρές εξασφαλίζουν μοναδικό τρόπο σύνδεσης.

Βήμα 7ο (Προαιρετικό): Σύνδεση επιπλέον φλάντζας στην ακάλυπτη πλευρά του αισθητήρα

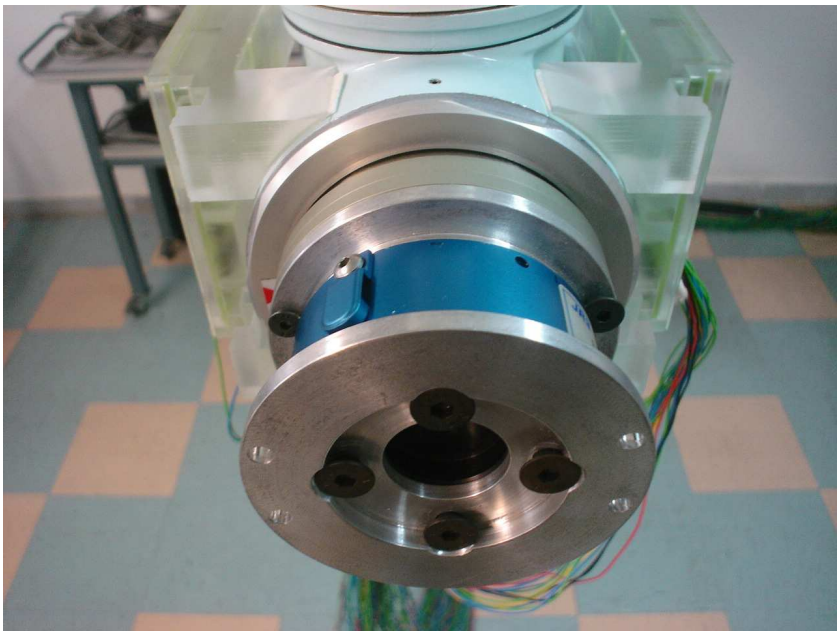


Εικόνα 21. Εξαρτήματα επιπλέον εξωτερικής φλάντζας.

Τα επιπλέον εξαρτήματα που χρειάζονται είναι:

1. Επιπλέον εξωτερική φλάντζα
2. Τέσσερις κοχλίες άλλεν M6 με κωνική κεφαλή και μικρό μήκος

Στην παρακάτω εικόνα παρουσιάζεται η τελική συνδεσμολογία έχοντας προσαρμόσει στο σύστημα και την επιπλέον εξωτερική φλάντζα.



Εικόνα 22. Προσαρμογή της επιπλέον φλάντζας στο υπάρχον σύστημα.

ΕΥΡΕΤΗΡΙΟ

Πίνακες:

Πίνακας 1. Τεχνικά χαρακτηριστικά βραχίονα PA-10 29
Πίνακας 2. Βασικά χαρακτηριστικά του αισθητήρα. 33
Πίνακας 3. Μηχανικές ιδιότητες απλού κράματος αλουμινίου. 43
Πίνακας 4. Εύρος μετρούμενων τιμών δυνάμεων και ροπών 53
Πίνακας 5. Εμφάνιση αποκλίσεων από τη μέση τιμή δύναμης Fz για δείγμα 700 μετρήσεων με την ίδια φόρτιση 61
Πίνακας 6. Πειραματικός υπολογισμός γωνίας απόκλισης του πλαισίου συντεταγμένων αισθητήρα σε σχέση με τον ακροδέχτη του βραχίονα. 93
Πίνακας 7. Αποτελέσματα χρονομέτρησης κύκλου λειτουργίας λογισμικού ελέγχου ForceControl.cpp. 96
Πίνακας 8. Μετρήσεις δύναμης Fx και αποκλίσεις από την πραγματική τιμή. 111
Πίνακας 9. Μετρήσεις δύναμης Fy και αποκλίσεις από την πραγματική τιμή. 112
Πίνακας 10. Μετρήσεις δύναμης Fz και αποκλίσεις από την πραγματική τιμή. 114
Πίνακας 11. Μετρήσεις ροπής Mx και αποκλίσεις από την πραγματική τιμή. 115
Πίνακας 12. Μετρήσεις ροπής My και αποκλίσεις από την πραγματική τιμή. 116
Πίνακας 13. Μετρήσεις ροπής Mz και αποκλίσεις από την πραγματική τιμή. 117
Πίνακας 14. Βαθμονόμηση Fx με χρήση τμηματικών συνεχών πολυωνύμων. 118
Πίνακας 15. Βαθμονόμηση Fy με χρήση τμηματικών συνεχών πολυωνύμων. 119

Διαγράμματα:

Διάγραμμα 1. Χαρακτηριστικές φίλτρου για το φίλτρο No.1 με δεδομένα αισθητήρα στα 8Khz..... 38
Διάγραμμα 2. Χαρακτηριστικές φίλτρου για το φίλτρο No.2 με δεδομένα αισθητήρα στα 8Khz..... 38
Διάγραμμα 3. Χαρακτηριστικές φίλτρου για το φίλτρο No.3 με δεδομένα αισθητήρα στα 8Khz..... 39
Διάγραμμα 4. Διακύμανση σήματος αισθητήρα για σταθερή φόρτιση Fz με ακρίβεια δύο δεκαδικών ψηφίων. 57
Διάγραμμα 5. Συχνότητα εμφάνισης (%) ενδείξεων αισθητήρα με σταθερή φόρτιση δύναμης Fz ~1.9N. 58
Διάγραμμα 6. Διακύμανση σήματος δύναμης Fz με ακρίβεια ενός δεκαδικού ψηφίου μέσω εφαρμογής βάρους 200gr. 59

Διάγραμμα 7. Διακύμανση σήματος δύναμης Fz με ακρίβεια ενός δεκαδικού ψηφίου μέσω εφαρμογής βάρους 1000gr. 59
Διάγραμμα 8. Διακύμανση σήματος δύναμης Fz με ακρίβεια ενός δεκαδικού ψηφίου μέσω εφαρμογής βάρους 1500gr. 60
Διάγραμμα 9. Υπέρθεση αποκλίσεων σταθερών διαφορετικών φορτίσεων Fz. 60
Διάγραμμα 10. Ραβδόγραμμα αποκλίσεων από τη μέση τιμή για διαφορετικές μεγέθους φορτίσεις... 61
Διάγραμμα 11. Αποκλίσεις μετρήσεων δυνάμεων από την πραγματική τιμή. 63
Διάγραμμα 12. Αποκλίσεις μετρήσεων ροπών από την πραγματική τιμή. 64
Διάγραμμα 13. Εφαρμογή πολυωνυμικών συναρτήσεων για την προσέγγιση του σφάλματος του σήματος. 69
Διάγραμμα 14. Εφαρμογή τμηματικών συνεχών πολυωνύμων για τη βαθμονόμηση της δύναμης Fx. 72
Διάγραμμα 15. Εφαρμογή τμηματικών συνεχών πολυωνύμων για τη βαθμονόμηση της δύναμης Fy. 72
Διάγραμμα 16. Παρατήρηση φαινομένου ταλάντωσης κατά την αρχική εφαρμογή της παρεμβολής μέσω κυβικών splines. 80
Διάγραμμα 17. Αναγωγή εξωτερικής δύναμης στη διάταξη βαθμονόμησης για τον ορθή χρήση των καμπύλων splines 85
Διάγραμμα 18. Γωνιακή θέση αρθρώσεων και μηχανικά όρια. 99
Διάγραμμα 19. Γωνιακή ταχύτητα αρθρώσεων και μηχανικά όρια. 100
Διάγραμμα 20. Ροπή αρθρώσεων πριν τη μείωση και μηχανικά όρια. 100
Διάγραμμα 21. Εξωτερική δύναμη και ταχύτητα του τελικού σημείου δράσης στον άξονα x. 101
Διάγραμμα 22. Εξωτερική δύναμη και ταχύτητα του τελικού σημείου δράσης στον άξονα y. 101
Διάγραμμα 23. Ροπή μέσω νόμου ελέγχου και ταχύτητα 1ης άρθρωσης. 102
Διάγραμμα 24. Ροπή μέσω νόμου ελέγχου και ταχύτητα 2ης άρθρωσης. 103
Διάγραμμα 25. Ροπή μέσω νόμου ελέγχου και ροπή αντιστάθμισης τριβής για την 1η άρθρωση. 104
Διάγραμμα 26. Ροπή μέσω νόμου ελέγχου και ροπή αντιστάθμισης τριβής για την 2η άρθρωση. 104
Διάγραμμα 27. Προσομοίωση σφάλματος Fx με χρήση κυβικών splines. 123
Διάγραμμα 28. Προσομοίωση σφάλματος Fy με χρήση κυβικών splines. 123
Διάγραμμα 29. Προσομοίωση σφάλματος Fz με χρήση κυβικών splines. 124
Διάγραμμα 30. Προσομοίωση σφάλματος Mx με χρήση κυβικών splines. 124
Διάγραμμα 31. Προσομοίωση σφάλματος My με χρήση κυβικών splines. 125
Διάγραμμα 32. Προσομοίωση σφάλματος Mz με χρήση κυβικών splines. 125

Εικόνες:

Εικόνα 1. Ο αισθητήρας δύναμης/ροπής JR3 μοντέλο 67M25A 33
Εικόνα 2. Η κάρτα PCI, δέκτης των σημάτων του αισθητήρα. 35

Εικόνα 3. Απεικόνιση της κάτω πλευράς της φλάντζας και παρουσίαση των οπών για τη σύνδεση του αισθητήρα. 47
Εικόνα 4. Λεπτομέρεια κατεργασίας φλάντζας σύνδεσης για την προσαρμογή του περικοχλίου..... 48
Εικόνα 5. Βοηθητική φλάντζα με τους κατάλληλους κοχλίες και την κατεργασία για την προσαρμογή τους. 48
Εικόνα 6. Φλάντζα του ακροδέκτη του ρομποτικού βραχίονα και λεπτομερής απεικόνιση των σπειρωμάτων που διαθέτει..... 49
Εικόνα 7. Λεπτομέρεια κατεργασίας της πάνω πλευράς της φλάντζας σύνδεσης για την προσαρμογή των κοχλίων M5..... 49
Εικόνα 8. Προοπτική απεικόνιση τελικής σύνδεσης αισθητήρα με τον ακροδέκτη του ρομποτικού βραχίονα..... 50
Εικόνα 9. Προοπτική απεικόνιση τελικής σύνδεσης αισθητήρα με τον ακροδέκτη του ρομποτικού βραχίονα και επιπλέον σύνδεση της βοηθητικής φλάντζας..... 50
Εικόνα 10. Τα πρότυπα βαρίδια που χρησιμοποιήθηκαν για τη βαθμονόμηση..... 52
Εικόνα 11. Ιδιοκατασκευή για την τοποθέτηση των πρότυπων βαριδίων. 54
Εικόνα 12. Οθόνη ανάγνωσης ανεπεξέργαστων τιμών φορτίσεων..... 56
Εικόνα 13. Εκτέλεση πειράματος. Διάταξη σε τυχαία κατάσταση 98
Εικόνα 14. Σύνολο εξαρτημάτων για τη σύνδεση του αισθητήρα στο ρομποτικό βραχίονα..... 190
Εικόνα 15. Αρχική τοποθέτηση κοχλιών. 191
Εικόνα 16. Φορά τοποθέτησης αισθητήρα στη φλάντζα σύνδεσης. 191
Εικόνα 17. Σύσφιξη αισθητήρα και φλάντζας μέσω περικοχλίων. 192
Εικόνα 18. Τελική σύνδεση αισθητήρα και φλάντζας. 192
Εικόνα 19. Τοποθέτηση ενδιάμεσης φλάντζας. 193
Εικόνα 20. Προσαρμογή του συστήματος φλάντζας-αισθητήρα στο ρομπότ..... 193
Εικόνα 21. Εξαρτήματα επιπλέον εξωτερικής φλάντζας. 194
Εικόνα 22. Προσαρμογή της επιπλέον φλάντζας στο υπάρχον σύστημα. 194

Σχήματα:

Σχήμα 1. Ισοδύναμο σχεδιάγραμμα ελέγχου βραχίονα σε επαφή με ελαστικό περιβάλλον υπό έλεγχο εμπέδησης. 19
Σχήμα 2. Σχεδιάγραμμα Ελέγχου δύναμης με χρήση εσωτερικού βρόχου θέσης..... 22
Σχήμα 3. Σχεδιάγραμμα Ελέγχου δύναμης με χρήση εσωτερικού βρόχου ταχύτητας..... 23
Σχήμα 4. Σχεδιάγραμμα παράλληλου ελέγχου με δύναμη/θέση. 24
Σχήμα 5. Σχηματική αναπαράσταση του PA-10. 28
Σχήμα 6. Αντιστοιχία των βαθμών ελευθερίας του ρομποτικού βραχίονα σε σχέση με αυτούς του ανθρώπου. 29

Σχήμα 7. Η μορφή του πακέτου δεδομένων στην επικοινωνία του PA-10. 31
Σχήμα 8. Μηχανολογικό κατασκευαστικό σχέδιο της φλάντζας σύνδεσης. 41
Σχήμα 9. Ισομετρική τρισδιάστατη απεικόνιση της φλάντζας σύνδεσης..... 42
Σχήμα 10. Τρισδιάστατη απεικόνιση εικονικής συναρμολόγησης αισθητήρα και ακροδέκτη μέσω της φλάντζας σύνδεσης 42
Σχήμα 11. Χαρακτηριστικά της φλάντζας σύνδεσης σε σχέση και με το υλικό κατασκευής. 43
Σχήμα 12. Πλέγμα πεπερασμένων στοιχείων και στηρίξεις του εικονικού ελέγχου αντοχής της φλάντζας. 44
Σχήμα 13. Απεικόνιση τάσεων της φλάντζας στην πλευρά σύνδεσης με τον αισθητήρα με εφαρμογή κάθετης φόρτισης ~100 N. 45
Σχήμα 14. Απεικόνιση τάσεων της φλάντζας στην πλευρά σύνδεσης με τον ακροδέκτη του ρομποτικού βραχίονα με εφαρμογή κάθετης φόρτισης ~100 N. 45
Σχήμα 15. Απεικόνιση παραμορφώσεων της φλάντζας με εφαρμογή παράλληλης δύναμης προς τη μετωπική επιφάνεια (άξονας y) ~ 100 N. 46
Σχήμα 16. Περιστροφή διανύσματος βάρους στο χώρο. 83
Σχήμα 17. Σχηματική αναπαράσταση διάταξης εξωσκελετού (a) Σε μια πραγματική εφαρμογή (b) Στο πειραματικό ανάλογο. 88
Σχήμα 18. Σχεδιάγραμμα Ελέγχου της πειραματικής εφαρμογής: Έλεγχος δύναμης με εσωτερικό βρόχο Θέσης. 91
Σχήμα 19. Υπολογισμός γωνίας απόκλισης πλαισίου ακροδέκτη βραχίονα και αισθητήρα..... 92
Σχήμα 20. Σχηματική αναπαράσταση κάτοψης πειραματικής διάταξης και συστήματα συντεταγμένων. 98

ΒΙΒΛΙΟΓΡΑΦΙΑ

Βιβλιογραφικές αναφορές:

[1] Lorenzo Sciavicco, Bruno Siciliano. *Modeling and Control of Robot Manipulators*, Naples 1996

[2] Bruno Siciliano, Luigi Villani. *Robot Force Control*, University of Naples 1999

[3] Ευάγγελος Παπαδόπουλος, Κωνσταντίνος Κυριακόπουλος. *Σημειώσεις Ρομποτικής*, Ε.Μ.Π. 2005.

[4] Απόλλων Οικονομόπουλος. *Επικοινωνίες και έλεγχος του βραχίονα Mitsubishi PA-10 σε περιβάλλον GNU/Linux*, Αθήνα 2005.

[5] Νικόλαος Μπόμπος. *Μοντελοποίηση, αναγνώριση δυναμικών παραμέτρων και έλεγχος του ρομποτικού βραχίονα Mitsubishi PA-10*, Αθήνα 2006.

[6] Κ.Χ. Γιαννάκογλου, Ι. Αναγνωστόπουλος, Γ. Μπεργελές. *Αριθμητική Ανάλυση για Μηχανικούς*, Αθήνα 2002.

Άρθρα:

[7] Xanthi Papageorgiou, Joe McIntyre, Kostas J. Kyriakopoulos. *Towards Recognition of Control Variables for an Exoskeleton*, Munich 2006

[8] J. Norberto Pires, John Ramming, Stephen Rauch, Ricardo Araujo. *Force/Torque Sensing Applied to Industrial Robotic Deburring*, University of Coimbra.

[9] Gabriel Afonso, J. Norberto Pires, Nelson Estrela. *Force control experiments for industrial applications: a test case using an industrial deburring example*, Technical School of Viseu, Portugal.

[10] Tao Ming Lim, Qing Hua Xia., Marcelo H Ang Jr., Ser Yong Lim. *Unified Force and Motion Control Using an Open System Real-Time Architecture on a 7 DOF PA-10 Robot*. Singapore Institute of Manufacturing Technology.

Χρήσιμα:

[11] JR3 Inc. *JR3 Force/Torque Sensor Users Manual*, Woodland, California 2001.

[12] MITSUBISHI HEAVY INDUSTRIES, LTD. *General Purpose Robot PA10 Series PA10-7CE Instruction Manual for Installation, Maintenance & Safety.*

[13] *Numerical Recipes in c: The art of scientific computing*, 1988-1992 by Cambridge University.

[14] ANSYS ICEM CFD/AI*Environment 10.0 User Manual, SAS IP, Inc.