Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Μηχανολόγων Μηχανικών
Τομέας Μηχανολογικών Κατασκευών & Αυτομάτου Ελέγχου
Εργαστήριο Αυτομάτου Ελέγχου

ΕΞΟΜΑΛΥΝΣΗ ΒΑΔΙΣΜΑΤΟΣ ΡΟΜΠΟΤΙΚΟΥ ΣΚΥΛΟΥ ΓΙΑ ΤΗΝ ΒΕΛΤΙΩΣΗ ΤΗΣ ΤΕΧΝΗΤΗΣ ΟΡΑΣΗΣ

Γεώργιος Θ. Τσαγκογέωργας

Διπλωματική Εργασία υπό την επίβλεψη του Καθηγητή
Κωνσταντίνου Ι. Κυριακόπουλου

Αθήνα 2008

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Μηχανολόγων Μηχανικών
Τομέας Μηχανολογικών Κατασκευών & Αυτομάτου Ελέγχου
Εργαστήριο Αυτομάτου Ελέγχου

ΕΞΟΜΑΛΥΝΣΗ ΒΑΔΙΣΜΑΤΟΣ ΡΟΜΠΟΤΙΚΟΥ ΣΚΥΛΟΥ ΓΙΑ ΤΗΝ ΒΕΛΤΙΩΣΗ ΤΗΣ ΤΕΧΝΗΤΗΣ ΟΡΑΣΗΣ

Γεώργιος Θ. Τσαγκογέωργας

Διπλωματική Εργασία υπό την επίβλεψη του Καθηγητή
Κωνσταντίνου Ι. Κυριακόπουλου

Αθήνα 2008

Στους γονείς μου
Θοδωρή, Βασιλική

Στα αδέλφια μου
Γρηγόρη, Γιάννα, Ελένη

Πρόλογος

Η παρούσα διπλωματική εργασία πραγματεύεται την εξομάλυνση βαδίσματος ρομποτικού σκύλου για βελτίωση της τεχνητής όρασης. Έλαβε χώρα στο Εργαστήριο Αυτομάτου Ελέγχου και Ρυθμίσεων Μηχανών, της Σχολής Μηχανολόγων Μηχανικών ΕΜΠ, με επιβλέποντα τον Καθηγητή κ. Κωνσταντίνο Κυριακόπουλο.

Θα ήθελα να ευχαριστήσω τον κ. Κυριακόπουλο για την εμπιστοσύνη που έδειξε στο πρόσωπο μου, καθώς και για τις πολύτιμες συμβουλές του κατά τη διάρκεια της εργασίας. Θα ήθελα επίσης να ευχαριστήσω τους Υποψήφιους Διδάκτορες Νικόλαο Τσόκα, Γεώργιο Καρρά και Δήμητρα Πανάγου, για την πολύτιμη βοήθεια και τη συμπαράσταση που μου προσέφεραν, καθώς και για τις ευφυείς ιδέες τους. Τέλος, θα ήθελα να ευχαριστήσω όλα τα μέλη του Εργαστηρίου Αυτομάτου Ελέγχου για το ευχάριστο κλίμα συνεργασίας που δημιουργούσαν, καθώς και για την βοήθεια που πρόθυμα μου παρείχαν.

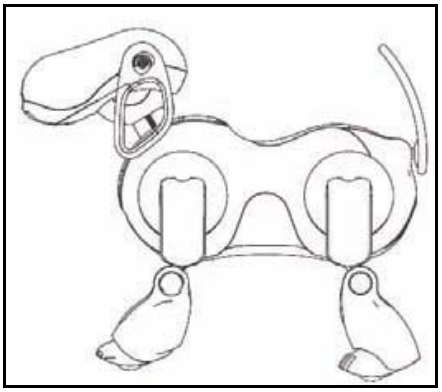
Περιεχόμενα

Κεφάλαιο 1 - Εισαγωγή και Ορισμός Προβλήματος

1.1 Γενικά

Τα ρομπότ χρησιμοποιούνται για συγκεκριμένες επαναληπτικές κινήσεις και σε διαφορετικούς χώρους π.χ συναρμολόγηση και κατασκευή διατάξεων, εξερεύνηση στη θάλασσα και στο διάστημα. Τα ρομπότ υπερτερούν σε σχέση με τον άνθρωπο σε ταχύτητα, ακρίβεια και επαναληψιμότητα. Τα μειονεκτήματα τους είναι η έλλειψη ικανότητας να σκεφτούν και να δρουν ανεξάρτητα, να συνεργαστούν, να μάθουν από τα λάθη τους και να προσαρμόζονται στις αλλαγές του περιβάλλοντος. Πολλά βαδίζοντα ρομπότ είναι μόνο ικανά να περπατούν σε επίπεδο έδαφος με γνωστή και προκαθορισμένη επιφάνεια. Οποιαδήποτε αλλαγή του εδάφους συχνά θα κάνει το ρομπότ ανίκανο να περπατήσει, εκτός και αν προσαρμόσει το περπάτημα του. Τα βαδίζοντα ρομπότ πρέπει να είναι ικανά να προσαρμόσουν την κατεύθυνση και το μοτίβο του περπατήματος, λόγω των εξωτερικών αλλαγών, όπως ανομοιομορφίες του εδάφους και κινούμενα αντικείμενα.

Για την ανάπτυξη νέων αλγορίθμων βαδίσματος χρησιμοποιείται ένα βαδίζων τετράποδο ρομπότ από την Sony Corporation [Sony Corporation 2005b], το οποίο ονομάζεται ERS-7 AIBO. AIBO σημαίνει ‘φίλος’ στα Ιαπωνικά και είναι ένα εμπορικό προϊόν, φτιαγμένο με στόχο την ψυχαγωγία των ανθρώπων. Το AIBO (σχ.1.1) είναι ένα παιχνίδι με πολλούς αισθητήρες και σερβοκινητήρες, χρησιμοποιείται σε πολλές διαφορετικές ερευνητικές περιοχές, όπως τεχνητή όραση, ρομποτική, τεχνητή νοημοσύνη και σε *embedded systems*. Το AIBO είναι μια ιδανική πλατφόρμα για να δοκιμαστούν νέοι αλγόριθμοι και ιδέες, επειδή η Sony διέθεσε το βασικό λογισμικό ‘OPEN-R’ [Sony Corporation 2005a] για τον προγραμματισμό του ρομπότ. Επιπλέον, έχουν αναπτυχθεί υψηλότερου επιπέδου πλατφόρμες προγραμματισμού, που βασίζονται στο ‘OPEN-R’, έχοντας ως αποτέλεσμα ο προγραμματισμός να είναι ευκολότερος και ταυτόχρονα να μπορούν να υλοποιηθούν πιο σύνθετες εργασίες. Δύο βασικές πλατφόρμες που υπάρχουν και συνεχίζεται η ανάπτυξη τους έως και σήμερα είναι το URBI (Universal Real-time Behavior Interface) [http://www.urbiforge.com], που χρησιμοποιήθηκε στην παρούσα εργασία και το Tekkotsu [http://www.tekkotsu.org].



Σχήμα 1.1: Πλάγια όψη AIBO

1.2 Καθορισμός του προβλήματος

Η δυσκολία με τα βαδίζοντα ρομπότ είναι η δημιουργία και η βελτιστοποίηση του βαδίσματος τους. Οι πολλοί βαθμοί ελευθερίας δημιουργούν πρόβλημα στην υλοποίηση ενός βαδίσματος σταθερού και γρήγορου. Η υλοποίηση του περπατήματος μπορεί να γίνει με διάφορους τρόπους. Μπορεί να ρυθμιστεί με το χέρι, δηλαδή να γίνουν πολλές δοκιμές για τις διάφορες παραμέτρους του βαδίσματος, το οποίο απαιτεί πολύ χρόνο για να πραγματοποιηθεί και έχει ως αποτέλεσμα ένα σταθερό αλλά αργό και μη προσαρμόσιμο βάδισμα. Άλλη επιλογή είναι να δημιουργηθούν μοτίβα βαδίσματος με αλγόριθμους, οι οποίοι μπορούν να προσαρμόζουν το περπάτημα στις αλλαγές του περιβάλλοντος και να το βελτιστοποιούν ώστε να είναι σταθερό και γρήγορο.

Επιπλέον, ένα βασικό πρόβλημα που δημιουργείται σε βαδίζοντα ρομπότ με ενσωματωμένη κάμερα είναι η κακής σταθεροποίησης εικόνες από την κάμερα λόγω της κίνησης και κατά συνέπεια η δυσκολία στην περαιτέρω επεξεργασίας τους. Αυτό συμβαίνει διότι κατά την διάρκεια του βαδίσματος το ρομπότ κινείται και καθ' ύψος επηρεάζοντας την στόχευση της κάμερα. Αυτό μπορούμε να το αντιληφθούμε καλύτερα αν σκεφτούμε το περπάτημα ενός ανθρώπου που κατά την διάρκεια της κίνησης μετακινείται ολόκληρο το σώμα του καθ' ύψος κάνοντας μια ταλάντωση. Άλλα ο άνθρωπος έχει την δυνατότητα να κινεί τα μάτια ώστε να κρατά σταθερό στόχο στο οπτικό του πεδίο. Στο ρομπότ -ERS-7- το πρόβλημα της όρασης είναι εντονότερο σε σχέση με άλλα βαδίζοντα ρομπότ, διότι είναι μη συμμετρικό και η κάμερα βρίσκεται τοποθετημένη στο κεφάλι του, που σημαίνει ότι επιδρά εκτός από την κίνηση του βαδίσματος και η αδράνεια του κεφαλιού.

Στην παρούσα εργασία γίνεται η προσπάθεια σταθεροποίησης της κάμερας σε καθορισμένο ύψος και προσανατολισμό σε σχέση με το έδαφος ώστε να ελαχιστοποιηθεί το πρόβλημα που περιγράφεται παραπάνω. Αναπτύχθηκε αλγόριθμος που χρησιμοποιεί μεθόδους βελτιστοποίησης για την δημιουργία του βαδίσματος και την ταυτόχρονη κίνηση του κεφαλιού για την επίτευξη καλύτερου αποτελέσματος.

1.3 Δομή Εργασίας

Στο 2^{ο} κεφάλαιο της εργασίας περιγράφεται το ρομπότ και το λογισμικό που χρησιμοποιεί (λειτουργικό σύστημα, πλατφόρμα προγραμματισμού). Ως πλατφόρμα προγραμματισμού περιγράφεται εκτός από το βασικό λογισμικό της κατασκευάστριας εταιρίας και μια πλατφόρμα υψηλότερου προγραμματιστικά επιπέδου, που έχει αναπτυχθεί για το συγκεκριμένο ρομπότ, την οποία χρησιμοποιήσαμε.

Στο 3^{ο} κεφάλαιο παρουσιάζεται η κινηματική ανάλυση του ρομπότ. Επιπλέον, γίνεται μια γενική περιγραφή για τους τύπους των βαδισμάτων και την ισορροπία των βαδιζόντων ρομπότ και παρουσιάζεται η μεθοδολογία που ακολουθήθηκε για την δημιουργία του βαδίσματος και την ισορροπία του ρομπότ. Τέλος, υπολογίζεται το ύψος της κάμερας από το έδαφος, που θα μας χρειαστεί για την σύνταξη της αντικειμενικής συνάρτησης στο πρόβλημα της βελτιστοποίησης.

Στο 4^{ο} κεφάλαιο γίνεται ο ορισμός του προβλήματος της βελτιστοποίησης και ο καθορισμός των φάσεων του βαδίσματος για τις οποίες πρέπει να επιλυθεί. Παρουσιάζονται αναλυτικά η αντικειμενική συνάρτηση που επιθυμούμε να βελτιστοποιήσουμε και οι περιορισμοί που ισχύουν σε κάθε φάση του βαδίσματος. Τέλος, αναφέρεται ο τρόπος επίλυσης σε περιβάλλον Matlab και ο τρόπος επικοινωνίας με το ρομποτ από το περιβάλλον αυτό.

Στο 5^{ο} κεφάλαιο παρουσιάζονται τα αποτελέσματα και σχολιάζεται η εφαρμογή της μεθοδολογίας που αναπτύχτηκε. Επιπλέον, δίνονται μελλοντικές κατευθύνσεις για το συγκεκριμένο πρόβλημα.

Στο παράρτημα 1 δίνονται τεχνικές λεπτομέρειες για την λειτουργία του ρομπότ και τον τρόπο με τον οποίο μπορούμε να το ελέγχουμε. Επίσης, δίνονται στοιχεία που είναι χρήσιμο να γνωρίζουμε κατά τον προγραμματισμό του.

Στο παράρτημα 2 παρουσιάζεται ο κώδικας που αναπτύχθηκε για περιβάλλον Matlab.

Κεφάλαιο 2 - Περιγραφή Ρομποτική Διάταξης

Στο κεφάλαιο αυτό περιγράφονται αναλυτικά τα συστήματα υλισμικού (*hardware)* και λογισμικού (*software*) που συνθέτουν τη ρομποτική διάταξη.

2.1 Περιγραφή υλισμικού (*hardware*) της ρομποτικής διάταξης

Το ρομπότ SONY AIBO ERS7 (σχ.2.1) είναι ένα αυτόνομο ρομπότ το οποίο λειτουργεί με έναν επεξεργαστή 64 bits RISC με συχνότητα 576MHz. Το ρομπότ χρησιμοποιεί δύο τύπους μνήμης, εσωτερική και αφαιρούμενη. Η εσωτερική είναι 64 MB και η αφαιρούμενη είναι ένα 32 MB Memory Stick. Το ρομπότ προγραμματίζετε μέσω του Memory Stick.

2.1.1 Αισθητήρες

Το ρομπότ διαθέτει διάφορους αισθητήρες, που ανανεώνονται κάθε 32 ms. Οι αισθητήρες μπορούν να κατηγοριοποιηθούν σε τρεις διαφορετικές αισθήσεις: Αίσθηση, Όραση και Ακοή.

Αίσθηση: Το AIBO έχει δύο τύπους αισθητήρων αφής, ηλεκτροστατικούς και πίεσης. Οι ηλεκτροστατικοί αισθητήρες είναι τοποθετημένοι στο κεφάλι και στην πλάτη του σκύλου. Οι αισθητήρες πίεσης είναι τοποθετημένοι κάτω από τις πατούσες και το πηγούνι. Επιπλέον, το AIBO έχει αισθητήρες επιτάχυνσης, που είναι ευαίσθητοι στην κίνηση στις 3 κατευθύνσεις, π.χ αριστερά – δεξιά, μπρος – πίσω, πάνω – κάτω. Επίσης, το ρομπότ διαθέτει έναν αισθητήρα θερμοκρασίας και έναν αισθητήρα κραδασμών.

Όραση: Το AIBO έχει έναν CMOS αισθητήρα εικόνας, που χρησιμοποιείται για να λαμβάνει εικόνες από το περιβάλλον. Ο αισθητήρας εικόνας έχει 350.000 pixels και έχει την δυνατότητα να λαμβάνει εικόνες με 30 frames το δευτερόλεπτο. Η κάμερα μπορεί και δουλεύει με τρεις αναλύσεις 208x160, 104x80 και 52x40. Επίσης, το AIBO έχει αισθητήρες μέτρησης απόστασης, που είναι τοποθετημένοι στην μύτη και στο στήθος. Οι αισθητήρες μέτρησης της απόστασης στην μύτη είναι ο ένας μεγάλου εύρους και ο άλλος μικρού. Μαζί και οι δύο λειτουργούν για αποστάσεις από 10 cm έως 90 cm.

Ακοή: Το AIBO μπορεί να ηχογραφεί ήχους με δύο στερεοφωνικά μικρόφωνα που είναι τοποθετημένα στην κάθε πλευρά του κεφαλιού.

2.1.2 Επενεργητές

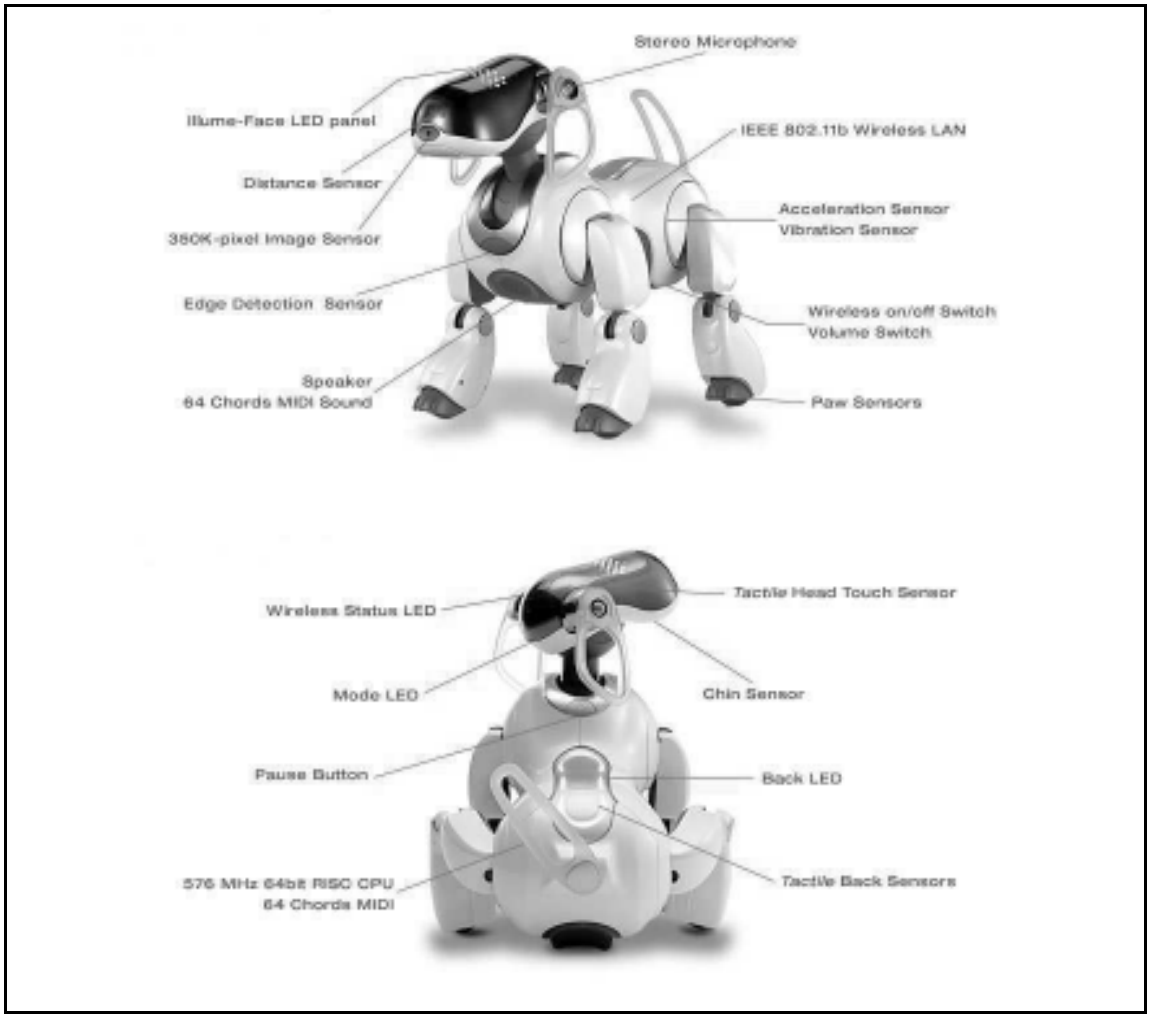
Το AIBO έχει διάφορους επενεργητές, μου μπορούν να κατηγοριοποιηθούν σε δύο τύπους: κίνηση και επικοινωνία.

Κίνηση: Το AIBO έχει συνολικά είκοσι βαθμούς ελευθερίας, έχει τέσσερα πόδια με τρεις βαθμούς ελευθερίας σε κάθε πόδι. Στο κεφάλι του έχει τρεις

συνδέσμους, δύο βαθμούς ελευθερίας στην ουρά και έναν στο στόμα. Επιπλέον, έχει δύο βαθμούς ελευθερίας στα αυτιά.

Επικοινωνία: Το AIBO έχει τρεις τύπους επικοινωνίας: οπτική, ηχητική και ασύρματη.

- Η οπτική επικοινωνία γίνεται με LEDs. Υπάρχουν LEDs στο κεφάλι και στην πλάτη του σκύλου.
- Η ηχητική επικοινωνία γίνεται με το μεγάφωνο, που βρίσκεται στο στήθος του. Το ηχείο μπορεί να παίζει πολυφωνικούς ήχους.
- Η ασύρματη επικοινωνία γίνεται μέσω της IEEE 802.11b ασύρματης κάρτας, που διαθέτει.



Σχήμα 2.1: SONY AIBO ERS-7

2.2 Περιγραφή λογισμικού (*software*) της ρομποτικής διάταξης

Σε αυτή την παράγραφο περιγράφονται το λειτουργικό σύστημα του ρομπότ και οι πλατφόρμες που χρησιμοποιούνται για τον προγραμματισμό του. Οι πλατφόρμες προγραμματισμού που περιγράφονται είναι το OPEN-R SDK από την

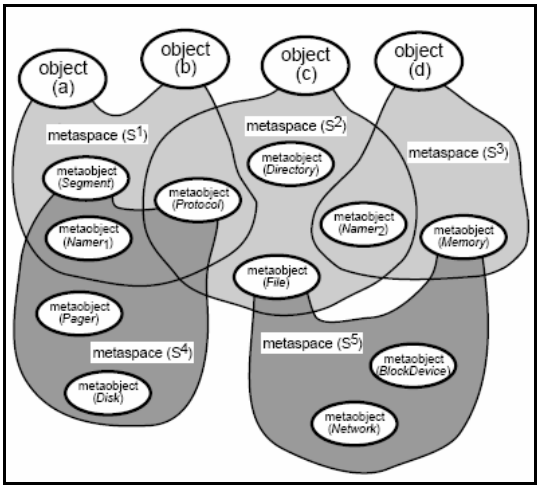
SONY και το URBI, που χρησιμοποιεί υψηλότερου επιπέδου προγραμματισμό, το οποίο χρησιμοποιήθηκε για τον προγραμματισμό του ρομπότ.

2.2.1 Το λειτουργικό σύστημα APERIOS

Το Aperios [1] είναι ένα κατανεμημένο, πραγματικού χρόνου λειτουργικό σύστημα που χρησιμοποιεί *objects* βασισμένο στην *meta-level* αρχιτεκτονική[2]. Αυτό είναι γνωστό με την ονομασία Apertos και υπάρχει σε πολλά προϊόντα της Sony σήμερα, συμπεριλαμβανομένου και του AIBO. Είναι πολύ μικρό σε σχέση με άλλα λειτουργικά συστήματα με μέγεθος 100 KB.

Το μόνο συστατικό του Aperios είναι το *object*. Το *object* είναι ελεύθερο να κινηθεί σε ένα κατανεμημένο περιβάλλον και μπορεί να αναπτυχθεί και να προσαρμοστεί στο περιβάλλον του. Επομένως, η σημασιολογία και η κυριότητα για το *object* αλλάζει κατά την διάρκεια της εκτέλεσης η όταν αναπτύσσεται ή κινείται. Με αυτό τον τρόπο υπάρχει ένας *object/meta-object* διαχωρισμός. Ένα *object* είναι μόνο ένας φορέας πληροφορίας, αν και ένα *meta-object* καθορίζει τη σημασιολογία της συμπεριφοράς του.

Κάθε *object* έχει το δικό του σύνολο των *meta-objects*, τα οποία δίνουν στο *object* αφηρημένες οδηγίες ή *meta-operations* που καθορίζουν την σημασιολογία του *object.* Καθώς ένα *meta-object* είναι ταυτόχρονα *object*, υπάρχουν *meta-objects* για αυτά τα *meta-objects.* Αυτό σημαίνει ότι υπάρχει μια *meta-hierarchy*, εντός της οποίας ένα *object* και το σύνολο των *meta-objects* του που καθορίζονται, όπως φαίνεται στο σχήμα 2.1.



Σχήμα 2.1: Διαχωρισμός *object/metaobject* και *metahierachy*

Το *object* χρησιμοποιεί την *meta-hierarchy* των *meta-objects* για να καθορίσει την συμπεριφορά του. Το σύνολο των *meta-objects* που χρησιμοποιεί ένα *object* ονομάζεται *meta-space.* Αν ένα *object* δεν μπορεί να συνεχίσει την εκτέλεση του, καθώς αναπτύσσεται ή αλλάζει, τότε μπορεί να μετακινηθεί, π.χ αλλάζοντας το σύνολο των *meta-objects* του, σε ένα νέο *meta-space* για να συνεχίσει την εκτέλεση του. Για παράδειγμα, αν αυτό θέλει να χρησιμοποιήσει TCP/IP επικοινωνία και το *meta-space* του δεν την υποστηρίζει, το *object* μετακινείται σε άλλο *meta-space* που υποστηρίζει αυτό τον τύπο της επικοινωνίας.

Κατά την διάρκεια της εκτέλεσης, μια Aperios εφαρμογή υποστηρίζεται από έναν από τους πολλούς *meta-spaces*. Κάθε *meta-space* περιέχει το ελάχιστο ένα *meta-object,* που ονομάζεται *reflector.* Ένας *reflector* δρα σαν μια πύλη. Αυτός εμποδίζει την είσοδο των εισερχόμενων αιτημάτων στον *meta-space* και τα προωθεί στο κατάλληλο *meta-object*. Για παράδειγμα, ένα αίτημα προγραμματισμού θα σταλεί από τον *reflector* στο *meta-object* που είναι υπεύθυνο για τον προγραμματισμό.

2.2.2 Πλατφόρμα προγραμματισμού SONY OPEN-R

Το OPEN-R [3] είναι ένα λογισμικό που αποτελείται από υπομονάδες (*modules)*, χρησιμοποιεί *objects* και είναι βασισμένο στην *layer* αρχιτεκτονική. Το λειτουργικό σύστημα Aperios το χρησιμοποιεί σαν ένα αφηρημένο *layer*. Το OPEN-R αποτελείται από δύο *layer*s: ένα *layer* του συστήματος και ένα *layer* των εφαρμογών.

Το *layer* του συστήματος περιέχει όλες τις απαραίτητες υπηρεσίες για να έχει πρόσβασή στις συσκευές του ρομπότ. Υπάρχουν τρία σημαντικά *objects* που παρέχονται από το *layer* του συστήματος και όλα παρέχουν διαφορετικές υπηρεσίες στο *layer* των εφαρμογών.

- OVirtualRobotComm – Χειρίζεται τους αισθητήρες, τους επενεργητές (συνδέσμους και LEDs) και την κάμερα.
- OVirtualRobotAudioComm – Χειρίζεται την ηχητική επικοινωνία, μεγάφωνα και μικρόφωνα.
- ANT object – Χειρίζεται την TCP/IP επικοινωνία.

Το *layer* των εφαρμογών είναι όπου υπάρχει ένα πρόγραμμα γραμμένο από το χρήστη. Αυτό χρησιμοποιεί το *layer* των εφαρμογών για να έχει πρόσβαση στις συσκευές του ρομπότ.

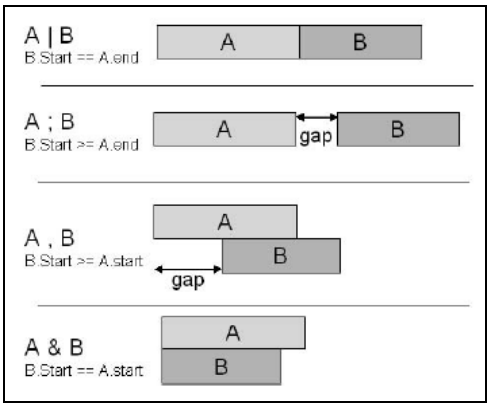
Οι υπηρεσίες του *layer* του συστήματος, που παίρνουν δεδομένα από το *layer* των εφαρμογών, στέλνουν ένα ενδεικτικό *event* στο *object* που είναι έτοιμο να λάβει περισσότερα δεδομένα, προηγουμένως αφού έχουν χειριστεί τα τελευταία δεδομένα που στάλθηκαν. Οι υπηρεσίες που στέλνουν δεδομένα στο *layer* των εφαρμογών στέλνουν ένα ενδεικτικό *event* στο *object* ότι έχουν σταλεί τα απαραίτητα δεδομένα. Το *layer* του συστήματος περιέχει OPEN-R *modules*, που ονομάζονται *objects.* Αυτά τα *objects* είναι διαφορετικά από τα C++ *objects,* αλλά υπάρχουν ομοιότητες. Έχουν την ίδια βασική λειτουργία από τα C++ *objects* και επιπλέον έχουν μερικές επεκτάσεις, όπως πολλαπλά σημεία εισόδου. Τα *modules* επικοινωνούν μέσω μηνυμάτων. Αυτά τα μηνύματα περνούν μέσω επικοινωνίας των *objects*, που χρησιμοποιούν προκαθορισμένα κανάλια επικοινωνίας. Τα κανάλια ορίζονται σε ένα αρχείο ρυθμίσεων. Το *module* που στέλνει το μήνυμα ονομάζεται *subject* και το *module* που λαμβάνει το μήνυμα ονομάζεται *observer.* Το μήνυμα μπορεί να μεταφέρει οποιοδήποτε C++ τύπο δεδομένων και ένας *identifier* αναγνωρίζει ποια μέθοδος πρέπει να εκτελεστεί για το μήνυμα που μόλις έφθασε. Ένα *module* μπορεί να έχει οποιοδήποτε αριθμό από *subjects* και *observers* ενσωματωμένους αλλά αυτό μπορεί να διαχειρίζεται ένα μόνο μήνυμα κάθε φορά. Αυτό συμβαίνει επειδή κάθε *module* στο OPER-R είναι *single-threaded.* Με αυτό τον τρόπο κάθε *module* πρέπει να έχει την δική του ουρά μηνυμάτων.

Κάθε OPEN-R πρόγραμμα περιέχει ένα η περισσότερα *modules* που τρέχουν ταυτόχρονα. Το γεγονός ότι το σύστημα είναι οργανωμένο σε υπομονάδες (*modules)* το κάνει να είναι εύκολο να αντικατασταθεί ένα *module* χωρίς να χρειάζεται να μεταγλωττιστεί ξανά ολόκληρο το πρόγραμμα. Επίσης, το OPEN-R συμπεριλαμβάνει λογισμικό για επεξεργασία από απόσταση πράγμα που σημαίνει ότι μπορεί να εκτελούνται μέρη του προγράμματος σε διαφορετική πλατφόρμα, π.χ σε ένα desktop υπολογιστή.

2.2.3 Πλατφόρμα προγραμματισμού URBI (Universal Real-time Behavior Interface)

Η πλατφόρμα URBI [4] βασίζεται στην *client/server* αρχιτεκτονική, βασισμένη σε μια γλώσσα προγραμματισμού που ονομάζεται URBI. Είναι δυνατόν να γίνει σύνδεση στο ρομπότ με έναν απλό *telnet client* και εισαγωγή URBI κώδικα για τον έλεγχο του ρομπότ και του λογισμικού που βρίσκεται στο ρομπότ. Η γλώσσα προγραμματισμού URBI έχει χαρακτηριστικά για να χειρίζεται παράλληλο και βασισμένο σε γεγονότα προγραμματισμό, απευθείας ενσωματωμένα στην σημασιολογία της γλώσσας. Μια αρχιτεκτονική στοιχείων είναι επίσης ενσωματωμένη στην γλώσσα μέσω μιας οικείας *object oriented* αρχιτεκτονικής. Η URBI είναι αρκετά όμοια με τις άλλες *scripting* γλώσσες, όπως είναι η Ruby, με σύνταξη που βασίζεται στην C/C++ για να είναι όσο το δυνατόν οικεία στους νέους χρήστες.

Ο παράλληλος προγραμματισμός δεν επιτυγχάνετε μέσω *threads* ή *callbacks* σε αυτή την γλώσσα, αλλά η URBI εισάγει τέσσερις τελεστές εντολών: ΄΄;΄΄ ΄΄,΄΄ ΄΄|΄΄ ΄΄&΄΄. Καθώς, το ερωτηματικό εισάγει την συνήθη σειριακή σημασιολογία, οι άλλοι τελεστές προσθέτουν περισσότερες δυνατότητες, συμπεριλαμβανομένου του παράλληλου προγραμματισμού, για το πώς θα εκτελεστούν δύο εντολές. Στο σχήμα 2.2 φαίνεται η χρήση των τελεστών. Φαίνεται ότι οι τελεστές ΄΄|΄΄ και ΄΄&΄΄ είναι ακριβής, καθώς επιβάλουν μια άμεση σειριακή ή παράλληλη εκτέλεση εντολών. Αυτό μπορεί να χρησιμοποιηθεί για να περιγράψει τον συγχρονισμό των διαδικασιών. Οι τελεστές ΄΄,΄΄ και ΄΄;΄΄ δεν είναι ακριβής καθώς η σημασιολογία τους επιτρέπει την ύπαρξη ενός κενού ανάμεσα τους. Με αυτό τον τρόπο είναι πολύ εύκολο να προγραμματιστεί σειριακή/παράλληλη εκτέλεση εντολών.



Σχήμα 2.2: Τα τέσσερα είδη των τελεστών στην γλώσσα URBI

Ένα άλλο σημαντικό χαρακτηριστικό της γλώσσας URBI είναι ο προγραμματισμός βασισμένος σε γεγονότα και πως τα χειρίζεται η γλώσσα. Βλέποντας τον κώδικα “at (condition) action;”, το γεγονός “action” θα εκτελείτε μια φορά, κάθε φορά που η συνθήκη “condition” θα είναι αληθής. Άλλη εντολή χειρισμού γεγονότων είναι η “whenever”, η οποία θα εκτελεί το γεγονός όταν η συνθήκη είναι αληθής και για όλη την διάρκεια που η συνθήκη είναι αληθής.

Η γλώσσα URBI είναι μια αντικειμενοστραφής γλώσσα, επομένως μπορούν να χρησιμοποιηθούν και να καθοριστούν *objects* όπως συνηθίζεται. Το πιο σημαντικό είναι ότι μπορούν να συνδεθούν *C++ objects* στην γλώσσα πολύ εύκολα. Τα *C++ objects* είναι ορατά και μπορούν να χρησιμοποιηθούν όπως κάθε άλλο *object* στην URBI και ονομάζονται *UObjects*. Τα *UObjects* μπορούν να συνδεθούν, αλλά μπορούν επίσης να αποσυνδεθούν και να εκτελεστούν εξ αποστάσεως σαν αυτόνομα εκτελέσιμα προγράμματα, παίρνοντας την διεύθυνση του URBI *Engine server* ως παράμετρο. Επιπλέον, η *client/server* αρχιτεκτονική επιτρέπει την αλληλεπίδραση της URBI, μέσω μιας σύνδεσης *socket*, με οποιαδήποτε γλώσσα προγραμματισμού. Το *Liburbi* είναι ένα σύνολο *GPL* βιβλιοθηκών για διάφορες γλώσσες προγραμματισμού όπως C++, C, Java, Matlab, Ruby, Python και άλλες, που επιτρέπει την δημιουργία συνδέσεων με το URBI. Αυτό είναι πολύ χρήσιμο διότι μπορεί να χρησιμοποιηθεί οποιαδήποτε γλώσσα προγραμματισμού για να αποστέλλονται και να λαμβάνονται μηνύματα από το ρομπότ με την χρήση του *Liburbi.*

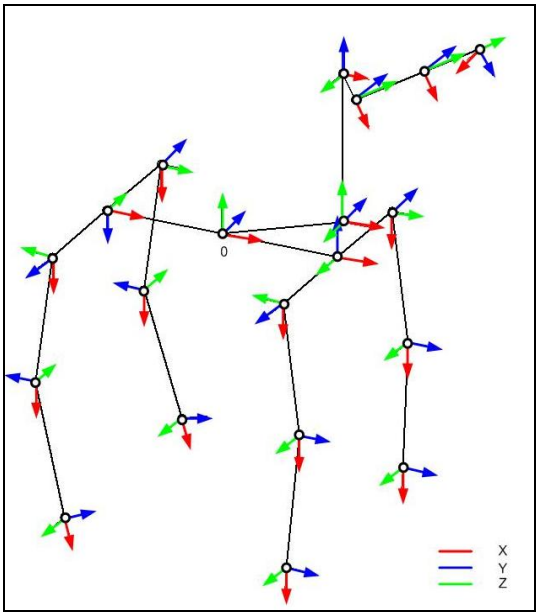
Κεφάλαιο 3 – Ανάπτυξη Μαθηματικού Μοντέλου

Σε αυτό το κεφάλαιο παρουσιάζεται αναλυτικά ο μαθηματικός ορισμός του προβλήματος. Αρχικά, παρουσιάζεται η κινηματική ανάλυση του ρομπότ. Έπειτα ακολουθούν οι μεθοδολογίες για την δημιουργία βαδίσματος και για την ισορροπία του ρομπότ.

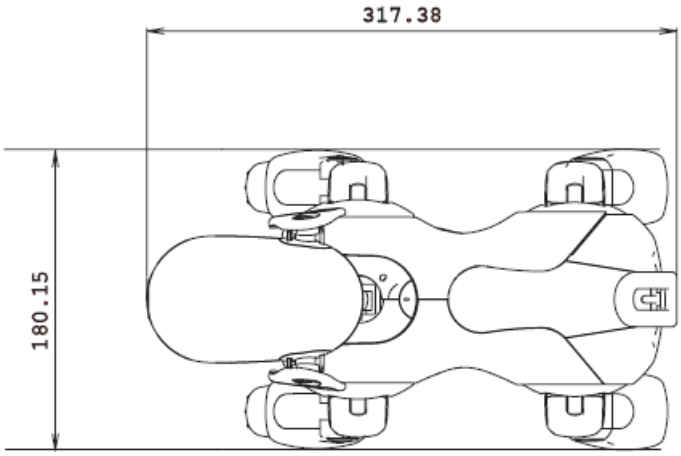
3.1 Κινηματική ανάλυση ρομποτικού σκύλου

Όπως αναφέρεται στο κεφ.2.1, ο ρομποτικός σκύλος έχει 20 βαθμούς ελευθερίας εκ των οποίων θα χρησιμοποιήσουμε στην κινηματική ανάλυση μόνο αυτούς που παρέχουν κίνηση στο σώμα και στο κεφάλι. Δηλαδή, θα χρησιμοποιήσουμε τις τρεις περιστροφικές αρθρώσεις κάθε ποδιού και τις 3 περιστροφικές αρθρώσεις του κεφαλιού.

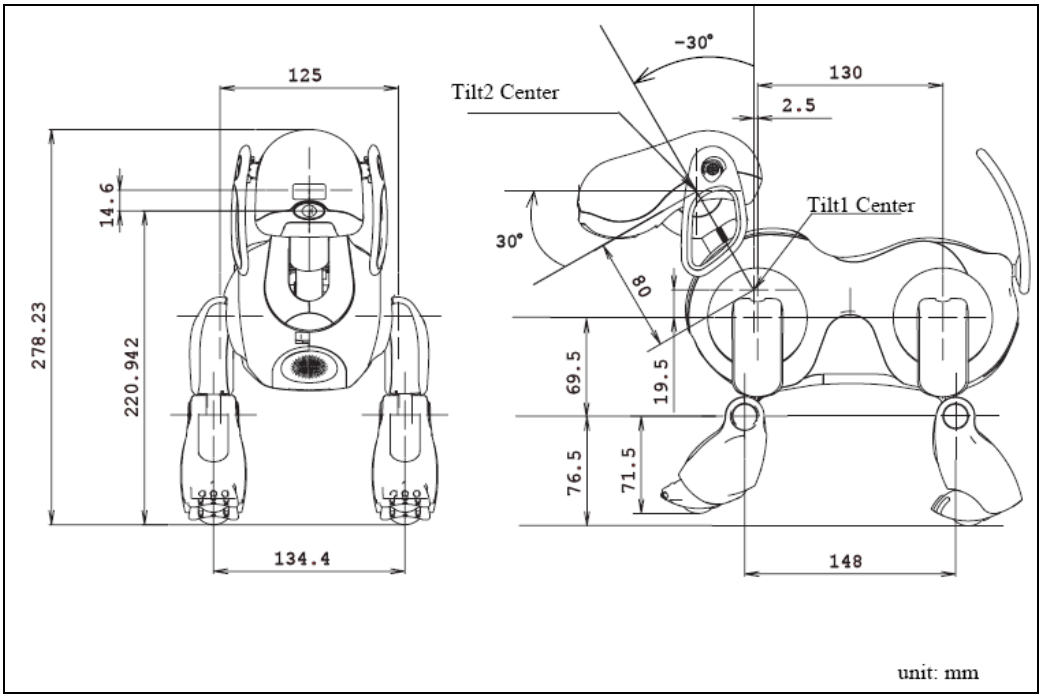
Η κινηματική ανάλυση του ρομπότ θα γίνει με την μέθοδο Denavit – Hartenberg [5], [6] για κάθε πόδι και για το κεφάλι ξεχωριστά με αρχικό πλαίσιο αναφοράς 0, όπως φαίνεται στο σχήμα 3.1. Το σημείο 0 επιλέχθηκε στο μέσο του σώματος του ρομπότ στις τρεις διαστάσεις του χώρου. Επιπλέον, στο σχήμα 3.1 φαίνονται τα συστήματα συντεταγμένων σε κάθε άρθρωση σύμφωνα με την μέθοδο D-H. Στα σχήματα 3.2α,β,γ φαίνονται οι διαστάσεις του ρομπότ, που θα χρησιμοποιήσουμε για τον υπολογισμό του κινηματικού μοντέλου.



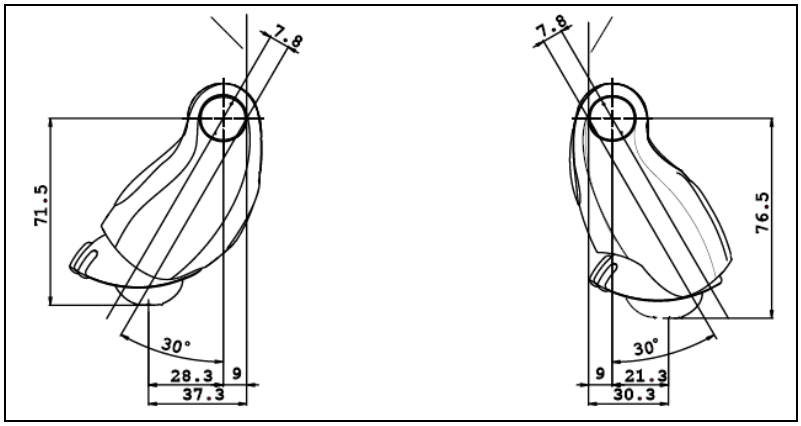
Σχήμα 3.1: Συστήματα συντεταγμένων κατά D-H



Σχήμα 3.2α: Διαστάσεις ρομπότ



Σχήμα 3.2β: Διαστάσεις ρομπότ

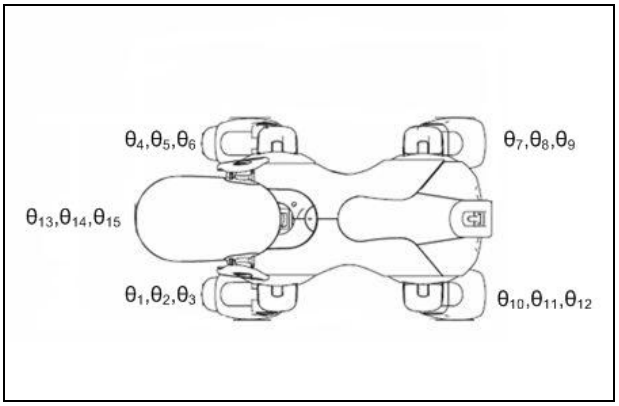


Σχήμα 3.2γ: Διαστάσεις ρομπότ

Όπως είναι γνωστό, η μέθοδος D-H παράγει μια συστηματική παραγωγή πινάκων A που συνδέουν τη θέση και τον προσανατολισμό ενός συνδέσμου ως προς τον προηγούμενο. Παρακάτω, φαίνεται ο πίνακας που συνδέει δυο διαδοχικούς συνδέσμους.

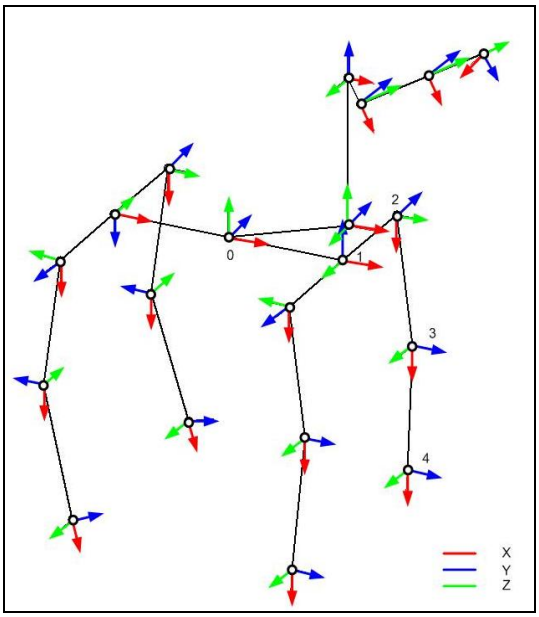
$$A_i^{i-1}(q_i) = \begin{bmatrix} c_{\theta_i} & -s_{\theta_i}c_{\alpha_i} & s_{\theta_i}s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i}c_{\alpha_i} & -c_{\theta_i}s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (3.1)$$

Ακολουθούν οι πίνακες με τις παραμέτρους D&H και οι πίνακες A που συνδέουν τους συνδέσμους, για κάθε πόδι ξεχωριστά. Επιπλέον, μετά τους πίνακες A υπολογίζονται οι συντεταγμένες του τελικού σημείου, δηλαδή το σημείο του ποδιού που πατάει στο έδαφος, ως προς το σημείο 0 σχημα 3.1. Παρακάτω φαίνονται μόνο οι σχέσεις που είναι αναγκαίες για την επίλυση του προβλήματος. Στον υπολογισμό, που ακολουθεί, χρησιμοποιούνται οι μεταβλητές θ_1 έως θ_{15} και έχουν οριστεί ως εξής: θ_1 έως θ_{12} για τις αρθρώσεις των ποδιών ξεκινώντας από το αριστερό μπροστά πόδι και συνεχίζοντας κυκλικά με την φορά του ρολογιού, σχήμα 3.3, και οι $\theta_{13}, \theta_{14}, \theta_{15}$ για τις αρθρώσεις του κεφαλιού. Στο παράρτημα 1 παρουσιάζεται οι αντιστοιχία των μεταβλητών θ_1 έως θ_{15} με τις μεταβλητές στην πλατφόρμα URBI. Οι μονάδες των μεγεθών είναι χιλιοστά για τις αποστάσεις και rad για τις γωνίες. Ακόμη, στους υπολογισμούς που ακολουθούν έχει γίνει η σύμβαση ότι $c_1 = \cos(\theta_1), s_1 = \sin(\theta_1)$ κ.ο.κ.



Σχήμα 3.3: Καθορισμός γωνιών στους συνδέσμους

3.1.1 Κινηματική Ανάλυση Αριστερού Μπροστά Ποδιού (1)



Σχήμα 3.3: Κινηματική αλυσίδα αριστερού μπροστά ποδιού

Πίνακας Παραμέτρων D&H Αριστερού Μπροστά Ποδιού (1)

i	θ_i	d_i	α	a
1	0	0	1.5708	65
2	θ_1*	-62.5	-1.5708	0
3	θ_2	9	1.5708	69.5
4	θ_3	-4.7	0	70.1646 *2

*2 Το $a_4 = 70.1646$ λόγω του μετατοπισμένου άξονα και της γωνίας των 30^o σχ. 3.2γ.

$$A_1^0 = \begin{bmatrix} 1 & 0 & 0 & 65 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad A_2^1 = \begin{bmatrix} c_1 & 0 & -s_1 & 0 \\ s_1 & 0 & c_1 & 0 \\ 0 & -1 & 0 & -62.5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

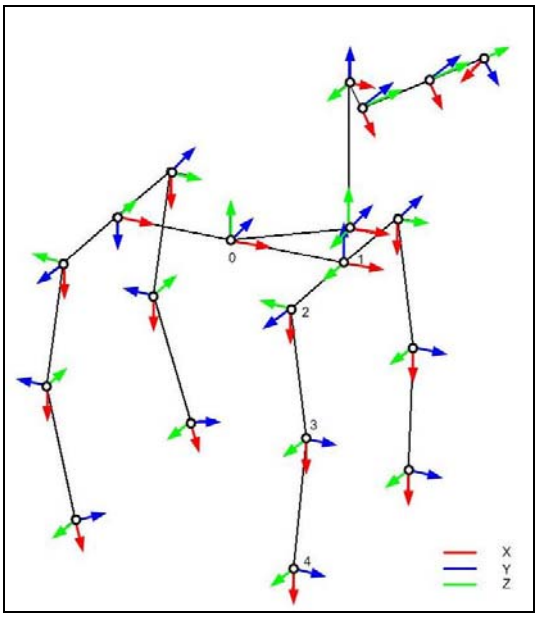
$$A_3^2 = \begin{bmatrix} c_2 & 0 & s_2 & 69.5c_2 \\ s_2 & 0 & -c_2 & 69.5s_2 \\ 0 & 1 & 0 & 9 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad A_4^3 = \begin{bmatrix} c_3 & -s_3 & 0 & 70.1646c_3 \\ s_3 & c_3 & 0 & 70.1646s_3 \\ 0 & 0 & 1 & -4.7 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$x_1 = 70.1646 * c_1c_2c_3 - 70.1646 * s_1s_3 - 4.7 * c_1s_2 + 69.5 * c_1c_2 + 65 - 9 * s_1 \qquad (3.2)$$

$$y_1 = -70.1646 * s_2c_3 - 62.5 - 4.7 * c_2 - 69.5 * s_2 \qquad (3.3)$$

$$z_1 = 70.1646 * s_1c_2c_3 + 70.1646 * c_1s_3 - 4.7 * s_1s_2 + 69.5 * s_1c_2 + 9 * c_1 \qquad (3.4)$$

3.1.2 Κινηματική Ανάλυση Δεξιού Μπροστά Ποδιού (2)



Σχήμα 3.4: Κινηματική αλυσίδα δεξιού μπροστά ποδιού

Πίνακας Παραμέτρων D&H Δεξιού Μπροστά Ποδιού (2)

i	θ_i	d_i	α	a
1	0	0	1.5708	65
2	θ_4*	62.5	1.5708	0
3	θ_5	-9	-1.5708	69.5
4	θ_6	4.7	0	70.1646

$$A_1^0 = \begin{bmatrix} 1 & 0 & 0 & 65 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad A_2^1 = \begin{bmatrix} c_4 & 0 & s_4 & 0 \\ s_4 & 0 & -c_4 & 0 \\ 0 & 1 & 0 & 62.5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

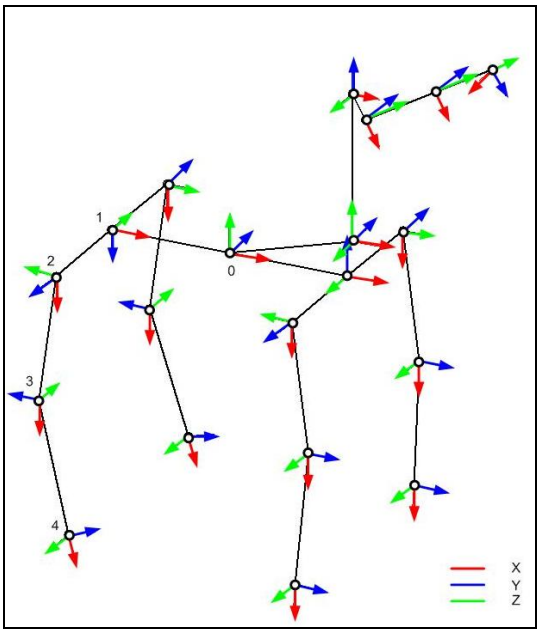
$$A_3^2 = \begin{bmatrix} c_5 & 0 & -s_5 & 69.5c_5 \\ s_5 & 0 & c_5 & 69.5s_5 \\ 0 & -1 & 0 & -9 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad A_4^3 = \begin{bmatrix} c_6 & -s_6 & 0 & 70.1646c_6 \\ s_6 & c_6 & 0 & 70.1646s_6 \\ 0 & 0 & 1 & 4.7 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$x_2 = 70.1646 * c_4 c_5 c_6 - 70.1646 * s_4 s_6 - 4.7 * c_4 s_5 + 69.5 * c_4 c_5 + 65 - 9 * s_4 \qquad (3.5)$$

$$y_2 = 70.1646 * s_5 c_6 + 62.5 + 4.7 * c_5 + 69.5 * s_5 \qquad (3.6)$$

$$z_2 = 70.1646 * s_4 c_5 c_6 + 70.1646 * c_4 s_6 - 4.7 * s_4 s_5 + 69.5 * s_4 c_5 + 9 * c_4 \qquad (3.7)$$

3.1.3 Κινηματική Ανάλυση Δεξιού Πίσω Ποδιού (3)



Σχήμα 3.5: Κινηματική αλυσίδα δεξιού πίσω ποδιού

Πίνακας Παραμέτρων D&H Δεξιού Πίσω Ποδιού (3)

i	θ_i	d_i	α	a
1	0	0	-1.5708	-65
2	θ_7*	-62.5	-1.5708	0
3	θ_8	9	1.5708	69.5
4	θ_9	-4.7	3.1416	70.1646

$$A_1^0 = \begin{bmatrix} 1 & 0 & 0 & -65 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad A_2^1 = \begin{bmatrix} c_7 & 0 & -s_7 & 0 \\ s_7 & 0 & c_7 & 0 \\ 0 & -1 & 0 & -62.5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

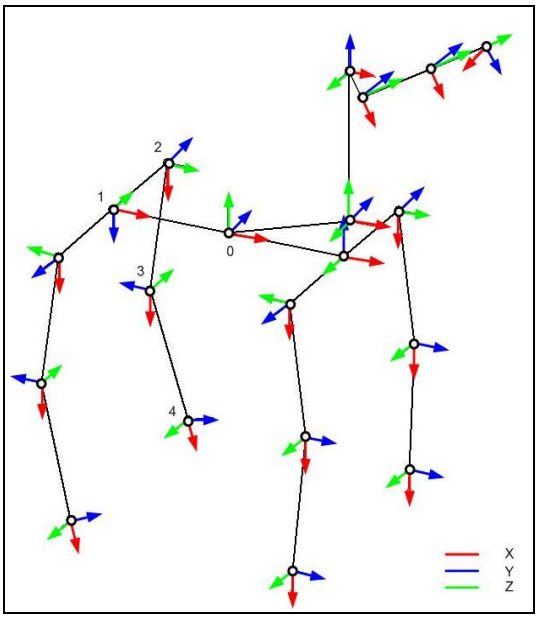
$$A_3^2 = \begin{bmatrix} c_8 & 0 & s_8 & 69.5c_8 \\ s_8 & 0 & -c_8 & 69.5s_8 \\ 0 & 1 & 0 & 9 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad A_4^3 = \begin{bmatrix} c_9 & s_9 & 0 & 70.1646c_9 \\ s_9 & -c_9 & 0 & 70.1646s_9 \\ 0 & 0 & -1 & -4.7 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$x_3 = 70.1646 * c_7c_8c_9 - 70.1646 * s_7s_9 - 4.7 * c_7s_8 + 69.5 * c_7c_8 - 65 - 9 * s_7 \qquad (3.8)$$

$$y_3 = -70.1646 * s_8c_9 - 62.5 - 4.7 * c_8 - 69.5 * s_8 \qquad (3.9)$$

$$z_3 = -70.1646 * s_7c_8c_9 - 70.1646 * c_7s_9 + 4.7 * s_7s_8 - 69.5 * s_7c_8 - 9 * c_7 \qquad (3.10)$$

3.1.4 Κινηματική Ανάλυση Αριστερού Πίσω Ποδιού (4)



Σχήμα 3.6: Κινηματική αλυσίδα δεξιού πίσω ποδιού

Πίνακας Παραμέτρων D&H Αριστερού Πίσω Ποδιού (4)

i	θ_i	d_i	α	a
1	0	0	-1.5708	-65
2	θ_{10}*	62.5	1.5708	0
3	θ_{11}	-9	-1.5708	69.5
4	θ_{12}	4.7	3.1416	70.1646

$$A_1^0 = \begin{bmatrix} 1 & 0 & 0 & -65 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad A_2^1 = \begin{bmatrix} c_{10} & 0 & s_{10} & 0 \\ s_{10} & 0 & -c_{10} & 0 \\ 0 & 1 & 0 & 62.5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_3^2 = \begin{bmatrix} c_{11} & 0 & -s_{11} & 69.5c_{11} \\ s_{11} & 0 & c_{11} & 69.5s_{11} \\ 0 & -1 & 0 & -9 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad A_4^3 = \begin{bmatrix} c_{12} & s_{12} & 0 & 70.1646c_{12} \\ s_{12} & -c_{12} & 0 & 70.1646s_{12} \\ 0 & 0 & -1 & 4.7 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$x_4 = 70.1646 * c_{10}c_{11}c_{12} - 70.1646 * s_{10}s_{12} - 4.7 * c_{10}s_{11} + 69.5 * c_{10}c_{11} - 65 - 9 * s_{10} \quad (3.11)$$

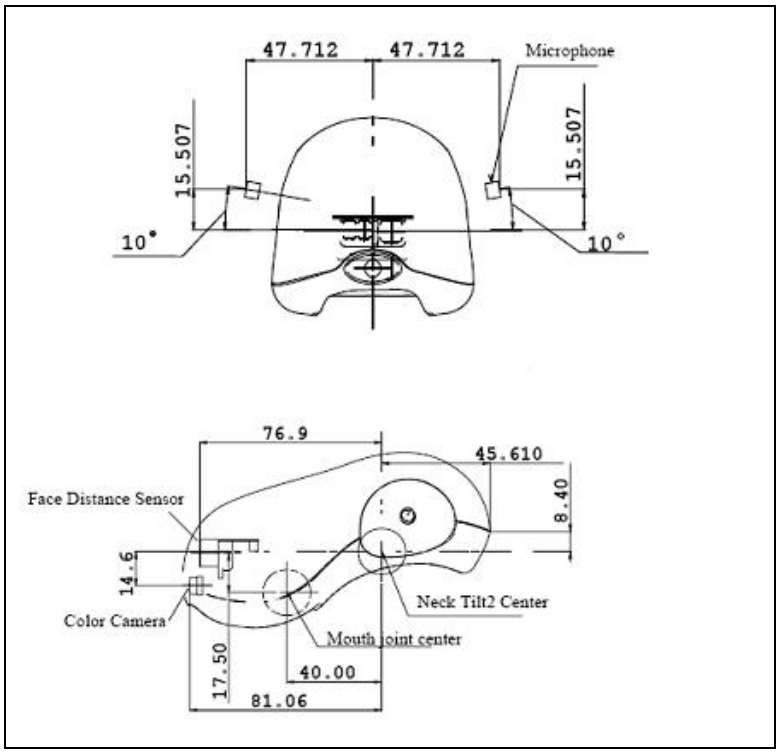
$$y_4 = 70.1646 * s_{11}c_{12} + 62.5 + 4.7 * c_{11} + 69.5 * s_{11} \quad (3.12)$$

$$z_4 = -70.1646 * s_{10}c_{11}c_{12} - 70.1646 * c_{10}s_{12} + 4.7 * s_{10}s_{11} - 69.5 * s_{10}c_{11} - 9 * c_{10} \quad (3.13)$$

*Μετά τον υπολογισμό του μοντέλου για να χρησιμοποιήσουμε το λογισμικό του ρομπότ χρειάζεται να κάνουμε τις εξής αντικαταστάσεις: π.χ για το μπροστά αριστερό πόδι $\theta_1 = q_1 - 1.5708, \theta_2 = q_2, \theta_3 = q_3$. Αυτό που πρέπει να τονίσουμε είναι η αντικατάσταση $\theta_1 = q_1 - 1.5708$, αυτό συμβαίνει διότι όταν στο λογισμικό έχουμε $q_1 = 0$ στο κινηματικό μας μοντέλο έχουμε $\theta_1 = -1.5708 rad$. Το ίδιο ισχύει και για τις μεταβλητές $\theta_4, \theta_7, \theta_{10}$, όπου $\theta_4 = q_4 - 1.5708$, $\theta_7 = q_7 + 1.5708$, $\theta_{10} = q_{10} + 1.5708$.

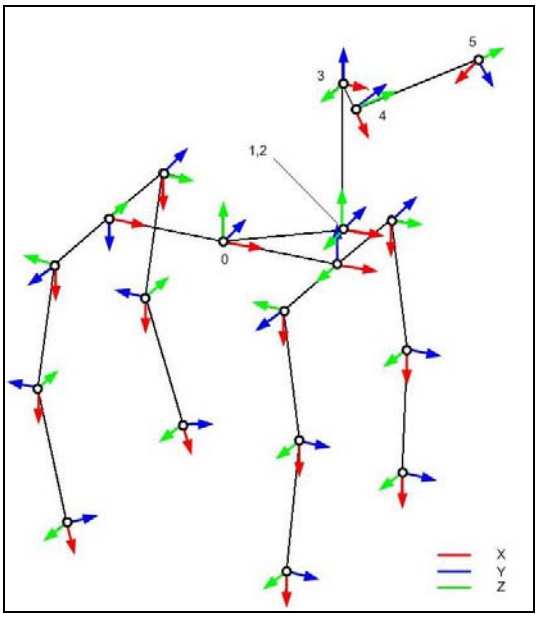
3.1.5 Κινηματική Ανάλυση Κεφαλιού

Σε αυτή την παράγραφο, υπολογίζεται το κινηματικό μοντέλο του κεφαλιού. Αρχικά, υπολογίζουμε την κινηματική από το σημείο 0 του ρομπότ έως την κάμερα. Έπειτα, υπολογίζεται η κινηματική από το σημείο 0 έως το κέντρο βάρους του κεφαλιού. Το κέντρο βάρους έχει ορισθεί προσεγγιστικά. Στο σχ. 3.7 φαίνονται οι διαστάσεις, που θα χρησιμοποιηθούν, για τον υπολογισμού του κινηματικού μοντέλου και σε αυτή την περίπτωση οι αποστάσεις είναι σε χιλιοστά και οι γωνίες σε rad.



Σχήμα 3.7: Διαστάσεις κεφαλιού

3.1.5.1 Κινηματική Ανάλυση Κεφαλιού - Κάμερας



3.8: Κινηματική αλυσίδα κάμερας

Πίνακας Παραμέτρων D&H Κάμερας

i	θ_i	d_i	α	a
1	0	19.5	1.5708	67.5
2	θ_{13}	0	-1.5708	0
3	θ_{14}	80	1.5708	0
4	θ_{15} *	0	-1.5708	14.6
5	-1.5708	81.06	0	0

$$* \theta_{15} = q_{15} - 1.5708$$

$$A_1^0 = \begin{bmatrix} 1 & 0 & 0 & 67.5 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 19.5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad A_2^1 = \begin{bmatrix} c_{13} & 0 & -s_{13} & 0 \\ s_{13} & 0 & c_{13} & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_3^2 = \begin{bmatrix} c_{14} & 0 & s_{14} & 0 \\ s_{14} & 0 & -c_{14} & 0 \\ 0 & 1 & 0 & 80 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad A_4^3 = \begin{bmatrix} c_{15} & 0 & -s_{15} & 14.6c_{15} \\ s_{15} & 0 & c_{15} & 14.6s_{15} \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_5^4 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 81.06 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

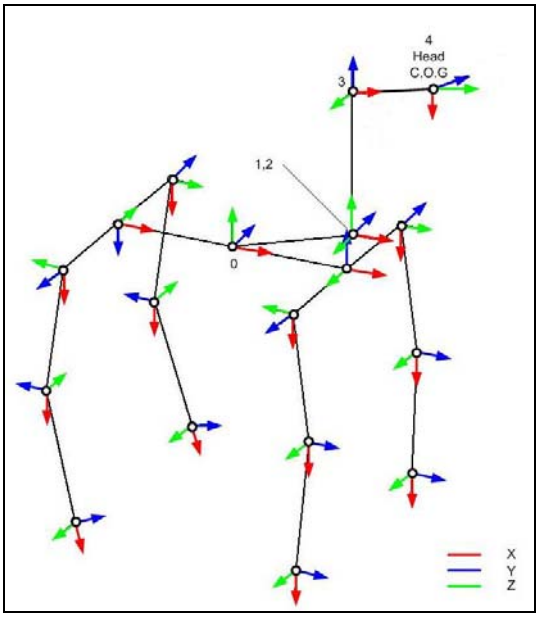
$$x_{CAM} = 67.5 - 81.06 * c_{13}c_{14}s_{15} - 81.06 * s_{13}c_{15} + 14.6 * c_{13}c_{14}c_{15} - 14.6 * s_{13}\, s_{15} - 80 * s_{13} \quad (3.14)$$

$$y_{CAM} = -81.06 * s_{14}s_{15} + 14.6 * s_{14}c_{15} \quad (3.15)$$

$$z_{CAM} = 19.5 - 81.06 * s_{13}c_{14}s_{15} + 81.06 * c_{13}c_{15} + 14.6 * s_{13}c_{14}c_{15} + 14.6 * c_{13}s_{15} + 80 * c_{13} \quad (3.16)$$

$$R_{CAM=}\begin{bmatrix} c_{13} * s_{14} & -c_{13} * c_{14} * c_{15} + s_{13} * s_{15} & -c_{13} * c_{14} * s_{15} - s_{13} * c_{15} \\ -c_{14} & -s_{14} * c_{15} & -s_{14} * s_{15} \\ s_{13} * s_{14} & -s_{13} * c_{14} * c_{15} - c_{13} * s_{15} & -s_{13} * c_{14} * s_{15} + c_{13} * c_{15} \end{bmatrix} \quad (3.17)$$

3.1.5.2 Κινηματική Ανάλυση Κεφαλιού – Κέντρο Βάρους



3.9: Κινηματική αλυσίδα κεφαλιού – κέντρο βάρους

Πίνακας Παραμέτρων D&H Κέντρου Βάρους

i	θ_i	d_i	α	a
1	0	19.5	1.5708	67.5
2	θ_{13}	0	-1.5708	0
3	θ_{14}	80	1.5708	0
4	θ_{15}*	0	0	20.79

* $\theta_{15} = q_{15} - 1.5708$

$$A_1^0 = \begin{bmatrix} 1 & 0 & 0 & 67.5 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 19.5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad A_2^1 = \begin{bmatrix} c_{13} & 0 & -s_{13} & 0 \\ s_{13} & 0 & c_{13} & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_3^2 = \begin{bmatrix} c_{14} & 0 & s_{14} & 0 \\ s_{14} & 0 & -c_{14} & 0 \\ 0 & 1 & 0 & 80 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad A_4^3 = \begin{bmatrix} c_{15} & -s_{15} & 0 & 20.79c_{15} \\ s_{15} & c_{15} & 0 & 20.79s_{15} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$x_{H_{COG}} = 20.79 * c_{13}c_{14}c_{15} - 20.79 * s_{13}s_{15} + 67.5 - 80 * s_{13} \quad (3.18)$$

$$y_{H_{COG}} = 20.79 * s_{14}c_{15} \quad (3.19)$$

$$z_{H_{COG}} = 20.79 * s_{13}c_{14}c_{15} + 20.79 * c_{13}s_{15} + 19.5 + 80 * s_{13} \quad (3.20)$$

3.2 Περιγραφή βαδίσματος τετράποδων ρομπότ

3.2.1 Τύποι βαδίσματος

Οι αρχικές έρευνες για την κίνηση βαδιζόντων μηχανών βασίζονταν στην παρατήρηση, κατανόηση και μοντελοποίηση βαδισμάτων από την φύση. Πολλά από αυτά τα βαδίσματα πραγματοποιούνται με τέτοια αλληλουχία κίνησης των ποδιών, ώστε να είναι περιοδικά. Άλλο χαρακτηριστικό αυτών των βαδισμάτων είναι ότι το σώμα κάνει συνεχώς σταθερή κίνηση και όλα τα πόδια κινούνται ταυτόχρονα. Έτσι, αυτά μπορούμε να τα ονομάσουμε συνεχή βαδίσματα (continuous gaits). Τα συνεχή βαδίσματα έχουν μελετηθεί εκτενώς. Η μοντελοποίηση, η οποία περιγράφει την αλληλουχία των ποδιών και ο διαχωρισμός του μοντέλου σε ευθείας και κυκλικής τροχιάς βάδισμα, έγινε μεταξύ του 1968 και του 1989 [McGhee και Frank, 1968; Zhang και Song, 1989], αλλά μόνο για συνθήκες επίπεδης επιφάνειας. Τα θεωρητικά αποτελέσματα αυτών των βαδισμάτων ήταν σημαντικά, αλλά δεν ήταν εφικτά για εφαρμογές σε πραγματικά βαδίζοντα ρομπότ σε μη επίπεδη επιφάνεια. Αργότερα, έγιναν κάποιες προσπάθειες να προσαρμοστούν αυτά τα βαδίσματα για μη επίπεδη επιφάνεια με συγκεκριμένες συνθήκες [Kumar και Waldron, 1989; Jimenez και Gonzalez de Santos, 1997].

Όταν οι ανωμαλίες της επιφάνειας είναι τόσο μεγάλες ώστε είναι δύσκολο να επιτευχθεί ένα συνεχές βάδισμα, τα θηλαστικά και τα έντομα μπορούν και αλλάζουν το βάδισμα τους σε πιο ασφαλές. Αυτό το βάδισμα χαρακτηρίζεται από την διαδοχική κίνηση των ποδιών και του σώματος. Το σώμα κινείται μπροστά/πίσω με όλα τα πόδια τοποθετημένα με ασφάλεια στο έδαφος. Μετά το πέρας της κίνησης του σώματος ένα πόδι κινείται, τα υπόλοιπα τρία και το σώμα είναι ακινητοποιημένα. Αυτά τα βαδίσματα, ονομάζονται μη συνεχή βαδίσματα (discontinuous gaits) επειδή υπάρχει ενδιάμεση κίνηση μόνο του σώματος και είναι πολύ αποδοτικά σε πραγματικές εφαρμογές. Η διαδοχική κίνηση σώματος-ποδιών, καθιστά τα μη

συνεχή βαδίσματα κατάλληλα για κίνηση σε μη επίπεδες επιφάνειες, καθώς κάθε πόδι χαμηλώνει, μέχρι να έρθει σε επαφή με το έδαφος, ενώ τα υπόλοιπα και το σώμα δεν κινούνται.

Για επιφάνειες με απρόσιτες περιοχές, όπως μεγάλες τρύπες ή επικίνδυνες περιοχές (π.χ ναρκοπέδια), το βάδισμα μπορεί να σπάει την περιοδικότητα του και να μετατρέπεται σε άλλο είδος βαδίσματος, που να επιλέγει που θα πατήσουν τα πόδια και την αλληλουχία της κίνησης σύμφωνα με τις συνθήκες της επιφάνειας. Αυτό το βάδισμα, ονομάζεται μη περιοδικό βάδισμα ή ελεύθερο βάδισμα (free gait).

Μπορούμε να κατατάξουμε τα βαδίσματα σε δύο μεγάλες κατηγορίες βασιζόμενοι στην περιοδικότητα, ως περιοδικά και μη περιοδικά. Έτσι ένα βάδισμα είναι περιοδικό εάν οι διάφορες φάσεις του βαδίσματος (άνοδος ποδιού, τοποθέτηση ποδιού και κίνηση σώματος) συμβαίνουν στις ίδιες χρονικές στιγμές του κύκλου του βαδίσματος. Από την άλλη πλευρά, εξετάζοντας το βάδισμα από την κίνηση του σώματος υπάρχουν βαδίσματα που το σώμα κινείται καθ' όλη την διάρκεια της κίνησης -συνεχή- και βαδίσματα που το σώμα κινείται διακεκομμένα -μη συνεχή-.

3.2.1.1 Μη περιοδικά – ελεύθερα βαδίσματα

Τα περιοδικά βαδίσματα, που αναφερθήκαν παραπάνω, παρουσιάζουν αρκετά πλεονεκτήματα, έτσι ώστε να είναι γενικευμένη η χρήση τους. Όμως, παρουσιάζουν και αξιοσημείωτα μειονεκτήματα από την αυστηρότητα του ορισμού τους, που εμποδίζει την προσαρμοστικότητα τους.

Βασικό πρόβλημα των αλγορίθμων βαδίσματος είναι η πλοήγηση σε μια δοσμένη διαδρομή. Τα περιοδικά βαδίσματα μπορούν να ακολουθούν ευθείες ή κυκλικές τροχιές, αλλά ο συνδυασμός αυτών των τροχιών για να ακολουθήσουν μια συγκεκριμένη πορεία είναι προβληματικός. Αυτό συμβαίνει επειδή τα περιοδικά βαδίσματα χρειάζονται συγκεκριμένες αρχικές θέσεις για τα πόδια (στο πλαίσιο αναφοράς του σώματος) για να ολοκληρώσουν την κίνηση και αυτές οι θέσεις εξαρτώνται από την τροχιά. Αν και αρκετές λύσεις έχουν προταθεί για την σύνδεση διαφορετικών περιοδικών βαδισμάτων, ένα πλήρες προσαρμόσιμο βάδισμα πρέπει να μπορεί να βρει ικανοποιητική στήριξη και την κατάλληλη αλληλουχία κίνησης των ποδιών για να ακολουθήσουν κάθε τροχιά ξεκινώντας από κάθε αρχική θέση.

Ένας άλλος περιορισμός των περιοδικών βαδισμάτων είναι η αναποτελεσματικότητα τους όταν η περιοχή βαδίσματος περιέχει απαγορευμένες περιοχές, π.χ περιοχές ακατάλληλες για στήριξη του ρομπότ. Απαγορευμένη περιοχή ορίζεται, για παράδειγμα, μια τρύπα ή μια κάθετη ακμή σε απότομη επιφάνεια. Πάλι, ένα πλήρες προσαρμόσιμο βάδισμα πρέπει να έχει την δυνατότητα να αλλάζει τα σημεία στήριξης και ακόμη την αλληλουχία των ποδιών για να κινείται σε μια απότομη επιφάνεια, ώστε να αποφεύγει την τοποθέτηση των ποδιών σε τέτοια σημεία.

Αυτά τα προβλήματα υποκίνησαν την μελέτη των ελεύθερων βαδισμάτων στα πρώτα χρόνια των βαδιζόντων ρομπότ [Kugushev and Jaroshevskij,1975]. Σε ένα ελεύθερο βάδισμα, η αλληλουχία των ποδιών, η στήριξη και η κίνηση του σώματος σχεδιάζεται με έναν μη καθορισμένο, ευέλικτο τρόπο ως συνάρτηση της τροχιάς, των ιδιομορφιών του εδάφους και την κατάσταση του ρομπότ. Γι' αυτό το λόγο τα

ελεύθερα βαδίσματα είναι πιο ισχυρά έναντι των άλλων βαδισμάτων όταν χρειαζόμαστε ένα υψηλό λειτουργικό επίπεδο κίνησης.

Ένας μεγάλος αριθμός ελεύθερων βαδισμάτων για τετράποδα και εξάποδα ρομπότ έχουν αναπτυχθεί έως σήμερα. Τα ελεύθερα βαδίσματα έχουν αποδειχτεί αποτελεσματικά στον έλεγχο εξαπόδων [Wettergreen and Thorpe, 1992; Salmi and Halme, 1996], λόγω του μεγάλου αριθμού πιθανών επιλογών (στήριξης, κίνησης σώματος) που ικανοποιούν τους αναπόφευκτους κινηματικούς και ισορροπίας περιορισμούς. Στην περίπτωση των τετραπόδων ρομπότ, οι αλγόριθμοι ελεύθερων βαδισμάτων είναι πιο εύκολο να καταλήξουν σε αδιέξοδο, δηλαδή σε κατάσταση όπου οι βασικοί περιορισμοί δεν μπορούν να ικανοποιηθούν λόγω των συνδέσμων. Ο αριθμός των επιτυχών ελεύθερων βαδισμάτων είναι αισθητά μικρότερος για τα τετράποδα σε σχέση με τα εξάποδα και βάση των αποτελεσμάτων χρειάζεται επιπλέον δουλεία για να επιτευχθεί μια αποδοτική κίνηση τετραπόδων σε απότομες επιφάνειες με απαγορευμένες περιοχές σε πραγματικές συνθήκες.

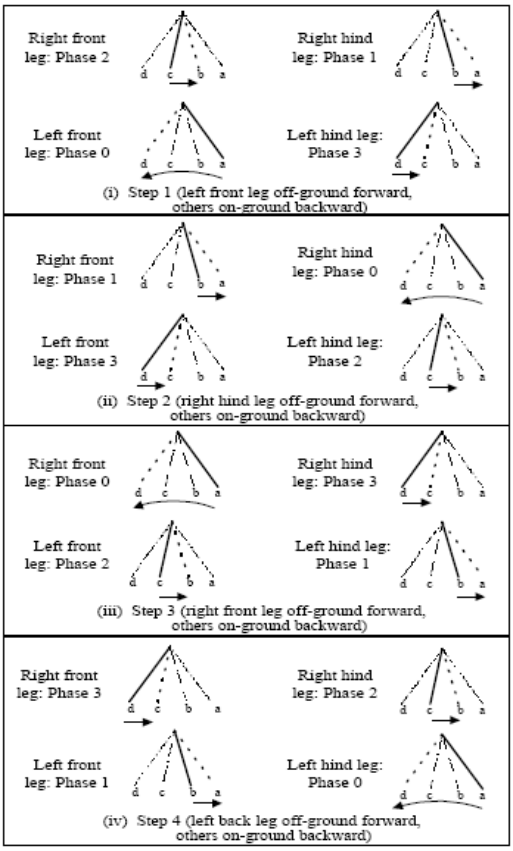
3.2.2 Δημιουργία βαδίσματος για το AIBO ERS-7

Στην περίπτωση μας αναπτύχθηκε ένα βάδισμα για κίνηση σε ευθεία τροχιά. Δεδομένου ότι το πρόβλημα προς επίλυση είναι η σταθεροποίηση της κάμερας σε καθορισμένο ύψος για την βελτιστοποίηση της όρασης, το ζητούμενο από το βάδισμα είναι η ομαλότητα χωρίς μεγάλες μετατοπίσεις, καθ' ύψος, του κέντρου βάρους του ρομπότ και κατ' επέκταση του κεφαλιού, που βρίσκεται η κάμερα. Στόχος είναι η επίτευξη του παραπάνω με ένα συνεχές βάδισμα, που να προσομοιώνει την κίνηση ενός πραγματικού σκύλου [8].

Το βάδισμα που αναπτύχθηκε είναι προσαρμόσιμο ώστε να ελαχιστοποιεί την κατακόρυφη μετατόπιση του σώματος του ρομπότ. Δεν είναι προσαρμόσιμο για κίνηση σε μη επίπεδη επιφάνεια, αν και μπορεί να γίνει κατάλληλο και για τέτοιου είδους κίνηση με τις κατάλληλες μετατροπές, αλλά είναι εκτός της παρούσας μελέτης.

Το βάδισμα που χρησιμοποιήθηκε χωρίζεται σε 4 φάσεις. Αποτελείται από την φάση 0 έως την φάση 3, όπως φαίνεται στο σχήμα 3.10. Στην φάση 0, ένα πόδι σηκώνεται από το έδαφος και μετακινείται εμπρός. Για τις επόμενες τρεις φάσεις αυτό το πόδι θα βρίσκετε στο έδαφος και σπρώχνει προς τα πίσω. Όλα τα πόδια συνεργάζονται για να δημιουργήσουν την δύναμη ώστε να κινηθεί το ρομπότ μπροστά. Όταν ένα πόδι βρίσκεται στον αέρα και κινείται εμπρός τα υπόλοιπα τρία είναι στο έδαφος και σπρώχνουν πίσω. Η διαφορά μεταξύ των φάσεων 1,2,3 είναι ότι το κάθε πόδι βρίσκεται σε διαφορετική θέση σχετικά με το σώμα. Ένας κύκλος των φάσεων του ποδιού γίνεται με την εξής σειρά 0-3-2-1-0. Θεωρώντας ότι η συνολική κίνηση κάθε ποδιού (σχετικά με το σώμα) ορίζεται ως το 100%, τότε των τεσσάρων ποδιών οι θέσεις d, c, b, a ορίζονται ως το 100%, 67%, 33%, 0% της κίνησης τους αντίστοιχα. Οι λεπτομέρειες της κίνησης των τεσσάρων φάσεων ακολουθούν παρακάτω:

- Φάση 0: Κίνηση από την θέση a στην θέση d (προς τα μπροστά κίνηση).
- Φάση 3: Κίνηση από τη θέση d στην θέση c (προς τα πίσω κίνηση).
- Φάση 2: Κίνηση από τη θέση c στην θέση b (προς τα πίσω κίνηση).
- Φάση 1: Κίνηση από την θέση b στην θέση a (προς τα πίσω κίνηση).



Σχήμα 3.10: Φάσεις Βαδίσματος

Σύμφωνα με το κινηματικό μοντέλο τα πόδια έχουν ως σημείο αναφοράς το κέντρο βάρους του σώματος. Οπότε, για κάθε φάση του βαδίσματος το τελικό σημείο κάθε ποδιού θα έχει τρεις συντεταγμένες (x, y, z) ως προς το κέντρο βάρους του σώματος σχήμα 3.1. Ο άξονας x, όπως φαίνεται στο σχήμα 3.1 είναι ο άξονας κατά τον οποίο έχουμε την κίνηση. Ο καθορισμός των συντεταγμένων x για κάθε πόδι κατά την διάρκεια του περπατήματος προκύπτει από την παραπάνω μεθοδολογία. Οι συντεταγμένες y, z προκύπτουν έπειτα από την επίλυση προβλήματος βελτιστοποίησης κεφ. 3.5.

Ένα σημαντικό σημείο για την επιτυχία ενός βαδίσματος είναι η ισορροπία του ρομπότ κατά την διάρκεια της κίνησης. Στην φάση κατά την οποία ένα πόδι βρίσκεται στον αέρα και τα υπόλοιπα τρία σπρώχνουν το σώμα προς τα εμπρός το ρομπότ κινδυνεύει να ανατραπεί προς την πλευρά του ποδιού που είναι στον αέρα.

3.3 Ισορροπία στα βαδίζοντα ρομπότ

Η έρευνα για την ισορροπία στα βαδίζοντα ρομπότ ξεκίνησε στα μέσα του 1960, όταν οι McGhee και Frank (1968) καθόρισαν την στατική ισορροπία ενός ιδανικού βαδίζοντος ρομπότ. Ακολουθώντας τον ορισμό τους, ένα ιδανικό βαδίζον ρομπότ είναι στατικά ευσταθές εάν η οριζόντια προβολή του κέντρου βάρους του βρίσκεται εντός του πολυγώνου στήριξης. Ως πολύγωνο στήριξης ορίζεται το πολύγωνο που δημιουργείται από τα πόδια που βρίσκονται τοποθετημένα στο έδαφος. Ιδανικό ρομπότ θεωρείται αυτό που έχει πόδια με μηδενική μάζα και δεν επιδρούν σε αυτό δυναμικά φαινόμενα.

Η ιδέα της στατικής ισορροπίας εμπνεύστηκε από τα έντομα. Τα έντομα χρησιμοποιούν τα πόδια τους, που έχουν σχεδόν μηδενική μάζα, για να στηρίξουν το σώμα τους, κατά την διάρκεια του περπατήματος και να δώσουν ταυτόχρονα ώθηση σε αυτό. Έτσι, αντί να κινούν το σώμα τους προσπαθώντας να διατηρήσουν την ισορροπία τους, έχουν τέτοια αλληλουχία βημάτων ώστε να εξασφαλίζεται στατική ισορροπία. Η πρώτη δημιουργία βαδιζόντων μηχανών χρησιμοποιεί την παραπάνω αρχή κίνησης [Kumar και Waldron, 1989]. Τα πρώτα βαδίζοντα ρομπότ ήταν τεράστιοι μηχανισμοί με πολύ μεγάλης μάζας συνδέσμους, δύσκολο να ελεγχθούν [Song και Waldron, 1989]. Η χρησιμοποίηση στατικού ευσταθές περπατήματος μπορούσε να διευκολύνει τον έλεγχο τους. Όμως, κατά την διάρκεια της κίνησης, λόγω της μεγάλης μάζας των συνδέσμων και του σώματος, αναπτύσσονταν αδρανειακά φαινόμενα και άλλα δυναμικά στοιχεία (τριβή, ελαστικότητα), ώστε η κίνηση του ρομπότ να μπορεί να επιτευχθεί μόνο με μικρή ταχύτητα. Μ ’αυτόν τον τρόπο ο έλεγχος του στατικού ευσταθές περπατήματος επιδρά αρνητικά στην ταχύτητα κίνησης.

Ο μόνος τρόπος για να αυξηθεί η ταχύτητα κίνησης του ρομπότ είναι να ληφθούν υπόψη τα δυναμικά φαινόμενα για τον έλεγχο της ισορροπίας του. Η πολυπλοκότητα για να ληφθεί υπόψη ολόκληρου του ρομπότ η δυναμική, οδήγησε του ερευνητές να ψάχνουν λύσεις, σχεδιάζοντας πολύ απλοποιημένες μηχανολογικές κατασκευές, που είχαν μόνο λίγους βαθμούς ελευθερίας [Raibert, 1986; Wong και Orin, 1993; Buehler, 1998], έτσι ώστε να χρησιμοποιούνται τα κριτήρια ισορροπίας που χρησιμοποιούνται για τα δίποδα (ανθρωποειδή ρομπότ) προσθέτοντας ένα ζεύγος ποδιών. Η κίνηση αυτών είναι περιορισμένη σε επίπεδη επιφάνεια, επειδή τα κριτήρια ισορροπίας που χρησιμοποιούν – βασίζονται στο Zero Moment Point [Vukobratic και Juricic, 1969] – είναι αξιόπιστα μόνο για αυτό τον τύπο επιφάνειας [Goswami, 1999; Kimura, 1990; Yoneda, 1996]. Επίσης, η μηχανολογική απλοποίηση αυτών των κατασκευών κάνουν το ρομπότ άχρηστο για πραγματικές εφαρμογές. Μικρή πρόοδος έχει επιτευχθεί, ώστε να συμπεριλαμβάνονται τα δυναμικά φαινόμενα, που μειώνουν την απόδοση των στατικά ισορροπούντων μηχανών [Gonzalez de Santos, 1998; Kang, 1997; Lin και Song, 1993; Papadopoulos και Rey, 1996; Yoneda και Hirose, 1997; Garcia και Gonzalez de Santos, 2005].

Τα δυναμικά κριτήρια ισορροπίας, που αναπτύχθηκαν για το περπάτημα τετραπόδων, φαίνεται να δίνουν διαφορετικούς τύπους και ονόματα σε μια απλή ιδέα: Το πρόσημο της ροπής, γύρω από τις άκρες του πολυγώνου στήριξης, που δημιουργείται από τα δυναμικά φαινόμενα στο κέντρο μάζας του ρομπότ. Η καταλληλότητα του κάθε κριτηρίου για κάθε ξεχωριστή εφαρμογή (π.χ χειρισμός δυνάμεων και ροπών, μη επίπεδη επιφάνεια κ.λ.π) δεν είναι καθόλου ξεκαθαρισμένο.

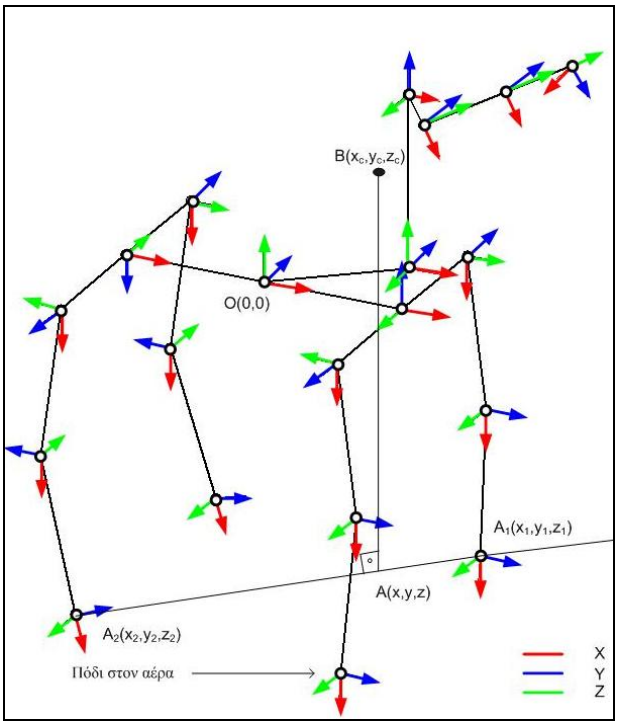
Η χρήση ενός κριτηρίου ισορροπίας κατάλληλο για μια εφαρμογή δεν σημαίνει ότι είναι κατάλληλο για κάποια άλλη εφαρμογή και θα οδηγήσει σε αποτυχία.

3.3.1 Ισορροπία AIBO ERS-7

Στην περίπτωση που εξετάζουμε χρησιμοποιούμε ένα απλοποιημένο δυναμικό κριτήριο ισορροπίας, διότι δεν γνωρίζουμε τα κατασκευαστικά στοιχεία του ρομπότ, μάζες και αδράνειες των επιμέρους εξαρτημάτων. Η κατασκευάστρια εταιρία δεν τα έχει δημοσιοποιήσει, έχοντας ως αποτέλεσμα να δυσκολεύει η υλοποίηση ενός ικανοποιητικού βαδίσματος. Όμως, η έλλειψη γνώσης αυτών των στοιχείων έχει οδηγήσει στην εκπόνηση εργασιών για την εκτίμηση αυτών [9]. Τα στοιχεία αυτά θα χρησιμοποιηθούν για την ισορροπία του ρομπότ.

Όπως αναφέρθηκε παραπάνω, το κρίσιμο σημείο που μπορεί να χάσει την ισορροπία του το ρομπότ είναι όταν ένα πόδι βρίσκεται στον αέρα. Σε αυτή την φάση υπάρχει ο κίνδυνος να ανατραπεί το ρομπότ προς την πλευρά του ποδιού που βρίσκεται στον αέρα. Για να εξασφαλίσουμε την ισορροπία υπολογίζουμε την φορά της ροπής που προκαλείται από το κέντρο μάζας του ρομπότ στον άξονα που περνά από τα διαγώνια πόδια σχήμα 3.11. Για να υπάρχει ισορροπία θα πρέπει η φορά της ροπής να είναι αντίθετη από την πλευρά του ποδιού που είναι στο αέρα.

Θεωρούμε ότι η ροπή που υπολογίζουμε προκαλείται από την μάζα του κεφαλιού (0.180 kg) και την μάζα του σώματος (0.9132 kg), θεωρώντας τις μάζες των υπόλοιπων μερών αμελητέες και αγνοούμαι τα αδρανειακά φαινόμενα λόγω της μικρής ταχύτητας κίνησης.



Σχήμα 3.11: Υπολογισμός ροπής στον άξονα των διαγώνιων ποδιών που βρίσκονται στο έδαφος

Σύμφωνα με το σχήμα 3.11, γεωμετρικά έχουμε:

$$x = x_1 + \lambda(x_2 - x_1),\ y = y_1 + \lambda(y_2 - y_1),\ z = z_1 + \lambda(z_2 - z_1) \qquad (3.21)$$

και $\mathbf{AB} = (x - x_c, y - y_c, z - z_c)$, (3.22)

όπου x_c, y_c, z_c είναι οι συνιστώσες του διανύσματος του κέντρου μάζας από το σημείο O(0,0) του ρομπότ.

Δηλαδή,

$$\mathbf{R}(x_c, y_c, z_c) = \frac{m_{body} * \mathbf{r}_{body} + m_{head} * \mathbf{r}_{head}}{m_{body} + m_{head}}, \qquad (3.23)$$

όπου $\mathbf{r}_{body} = (0,0,0)$ και $\mathbf{r}_{head}(x_{H_{COG}}, y_{H_{COG}}, z_{H_{COG}})$. Τα $x_{H_{COG}}, y_{H_{COG}}, z_{H_{COG}}$ τα παίρνουμε από το κινηματικό μοντέλο σχέσεις 3.18, 3.19, 3.20.

Το διάνυσμα που περνά από τον άξονα A_2A_1 είναι:

$$\mathbf{A_2A_1} = (x_2 - x_1, y_2 - y_1, z_2 - z_1) \qquad (3.24)$$

Τα διανύσματα $\mathbf{AB}$ και $\mathbf{A_2A_1}$ είναι κάθετα, οπότε $\mathbf{AB} \bullet \mathbf{A_2A_1} = \mathbf{0}$, άρα:

$$(x - x_c)(x_2 - x_1) + (y - y_c)(y_2 - y_1) + (z - z_c)(z_2 - z_1) = 0 \qquad (3.25)$$

Αντικαθιστούμε τις σχέσεις 3.21 στην 3.25 και λύνοντας ως προς λ έχουμε:

$$\lambda = -\frac{(x_1 - x_c)(x_2 - x_1) + (y_1 - y_c)(y_2 - y_1) + (z_1 - z_c)(z_2 - z_1)}{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \qquad (3.26)$$

Κάνοντας αντικατάσταση της 3.26 στις 3.21 παίρνουμε τις συντεταγμένες του σημείο A.

Υπολογίζουμε την δύναμη που ασκείται στο κέντρο μάζας:

$$\mathbf{F} = m * \mathbf{g} \qquad (3.27)$$

όπου $m = m_{body} + m_{head} = 1.0932kg$

Η ροπή του σημείου Β ως προς τον άξονα A_2A_1 υπολογίζεται ως:

$$\tau = \mathbf{AB} \times \mathbf{F} = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ x - x_c & y - y_c & z - z_c \\ F_x & F_y & F_z \end{vmatrix} \tag{3.28}$$

Άρα:

$$\begin{aligned} \tau_x &= \left(y_c - y\right) F_z + \left(-z_c + z\right) F_y \\ \tau_y &= \left(z_c - z\right) F_x + \left(-x_c + x\right) F_z \end{aligned} \tag{3.29}$$

Στην περίπτωση μας, ενδιαφερόμαστε για την ροπή στους άξονες x,y γιατί σε αυτούς είναι κρίσιμη η φορά της για να μην έχουμε ανατροπή.

3.4 Υπολογισμός του ύψους της κάμερας από το έδαφος

Αρχικά θα υπολογίσουμε το ύψος του σημείου 0 από το έδαφος και έπειτα σε αυτό θα προσθέσουμε το ύψος (z) της κάμερας από το σημείο 0. Το ύψος του σημείου 0 από το έδαφος είναι συνάρτηση των ποδιών που ακουμπούν στο έδαφος. Αρχικά, θα πρέπει να υπολογίσουμε την απόσταση του σημείου C (σχ.3.12) από το έδαφος, βάση των A, B, έπειτα την απόσταση του D, βάση των F,E και τέλος την ζητούμενη του O, βάση των C,D.

Σύμφωνα με το σχήμα 3.12 λόγω της ομοιότητας των τριγώνων έχουμε:

$$\frac{AC}{AB} = \frac{CB}{AB} = \frac{1}{2} = \frac{CC'}{BB'} \tag{3.30}$$

Όμως,

$$z_C = z_A + CC' = z_A + \frac{BB'}{2} = z_A + \frac{z_B - z_A}{2} = \frac{z_A + z_B}{2} \tag{3.31}$$

Ομοίως,

$$z_D = \frac{z_E + z_F}{2} \tag{3.32}$$

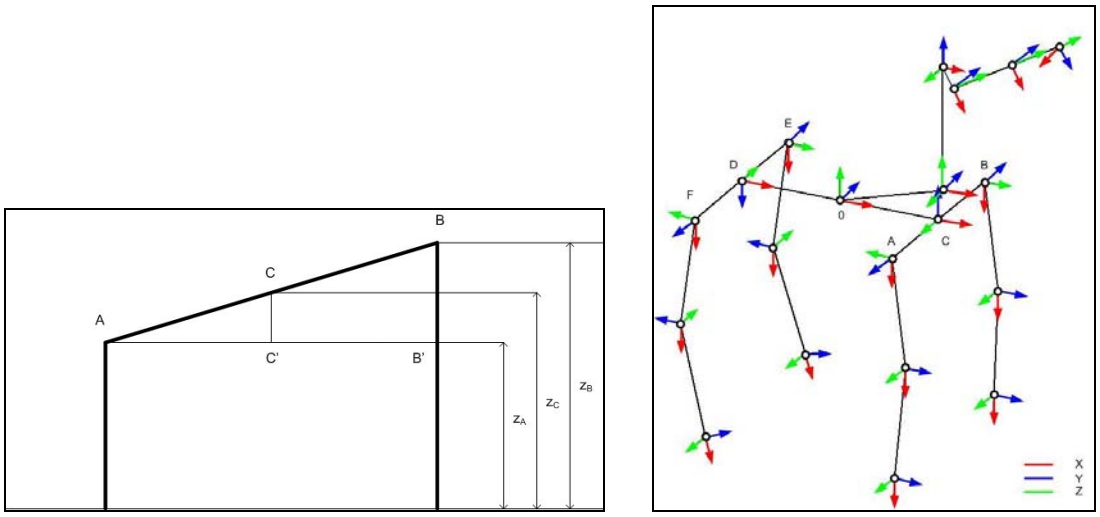
Άρα,

$$z_0 = \frac{z_C + z_D}{2} = \frac{\frac{z_A + z_B}{2} + \frac{z_E + z_F}{2}}{2} = \frac{z_A + z_B + z_E + z_F}{4} \tag{3.33}$$

Αυτός ο τύπος ισχύει στην περίπτωση που και τα τέσσερα πόδια ακουμπούν στο έδαφος. Στην περίπτωση που ένα πόδι βρίσκεται στον αέρα π.χ το μπροστά δεξιά (z_A) τότε:

$$z_0 = \frac{z_B + \frac{z_E + z_F}{2}}{2} = \frac{2z_B + z_E + z_F}{4} \qquad (3.34)$$

Ομοίως υπολογίζεται η απόσταση z_0 εάν βρίσκεται κάποιο από τα υπόλοιπα τρία στον αέρα.



Σχήμα 3.12: Υπολογισμός ύψους της κάμερας από το έδαφος

Το ζητούμενο ύψος της κάμερας από το έδαφος προκύπτει προσθέτοντας το z_0 με το z_{CAM} σχέση 3.16., άρα

$$h_{CAM} = z_0 + z_{CAM} \qquad (3.35)$$

Όμως τα z_A, z_B, z_E, z_F είναι τα z_2, z_1, z_4, z_3 του κινηματικού μοντέλου σχέσεις 3.7, 3.4, 3.13, 3.10 ,αντίστοιχα.

Κεφάλαιο 4 – Πρόβλημα βελτιστοποίησης και επίλυση

Σε αυτό το κεφάλαιο παρουσιάζεται ο ορισμός του προβλήματος της βελτιστοποίησης για τις διάφορες φάσεις του βαδίσματος και τα προγραμματιστικά εργαλεία που χρησιμοποιήθηκαν για την επίλυση του

4.1 Ορισμός προβλήματος βελτιστοποίησης

Για την δημιουργία του βαδίσματος και της κίνηση του κεφαλιού, ώστε να επιτύχουμε ικανοποιητική σταθεροποίηση της κάμερας, χρησιμοποιούμε μεθόδους βελτιστοποίησης για πολυπαραμετρική μη γραμμική εξίσωση με περιορισμούς.

Γενικά ένα πρόβλημα βελτιστοποίησης ορίζεται ως:

Εύρεση του $\min_x f(x)$: $\left\{ \begin{matrix} c(x) \le 0 \\ c_{eq}(x) = 0 \\ lb \le x \le ub \end{matrix} \right\}$, όπου

x : οι μεταβλητές βελτιστοποίησης

$c(x)$:είναι οι ανισοτικοί περιορισμοί

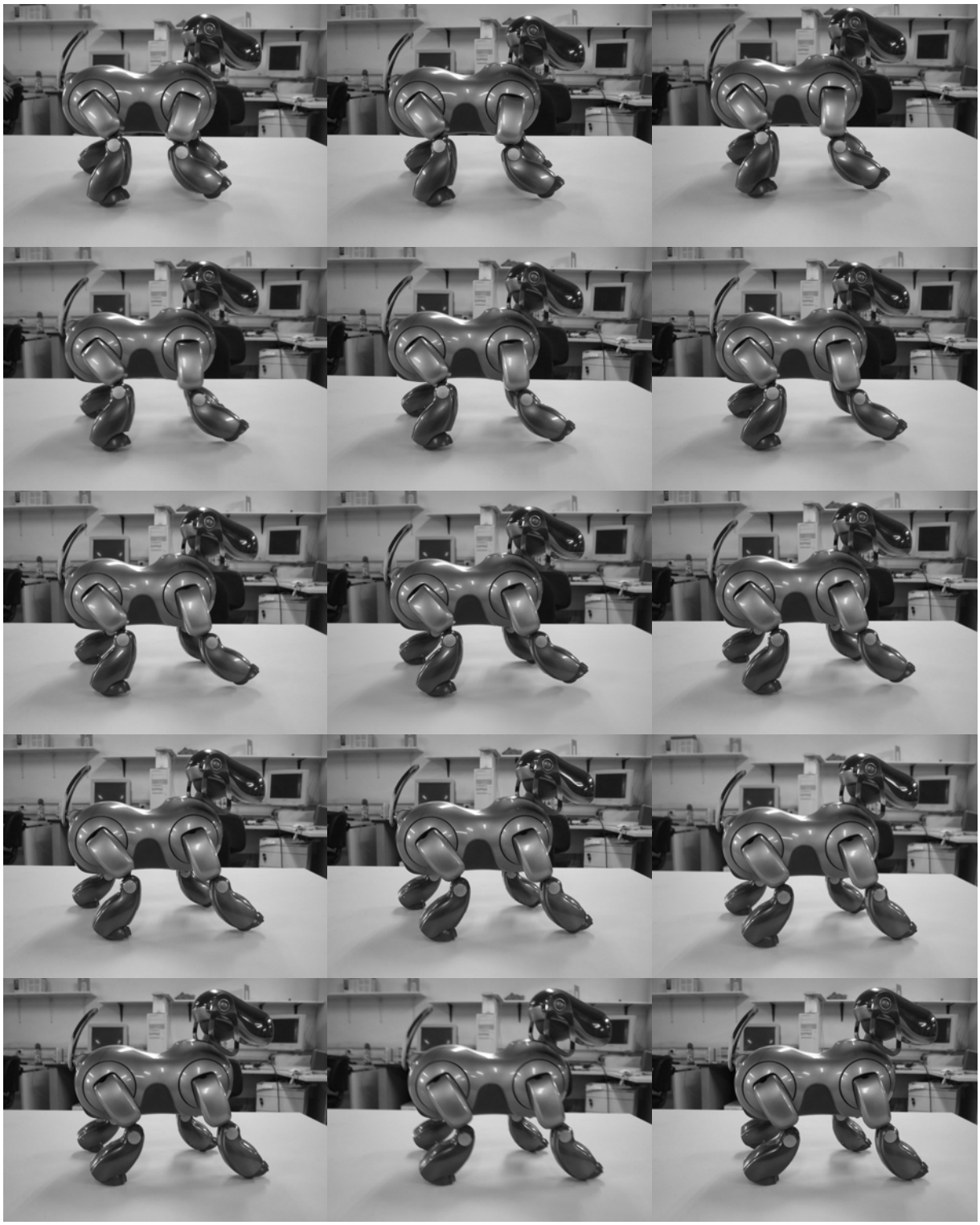
$c_{eq}(x)$: είναι οι ισοτικοί περιορισμοί

lb, up : είναι οι ελάχιστες και μέγιστες τιμές που μπορούν να πάρουν οι μεταβλητές βελτιστοποίησης, αντίστοιχα.

Το προβλήματος της βελτιστοποίησης πρέπει να επιλυθεί αρκετές φορές με διαφορετικούς περιορισμούς για τις διάφορες φάσεις του βαδίσματος. Στο κεφ. 3.2.2 αναφέρεται ότι οι φάσεις του βαδίσματος είναι τέσσερις λόγω του διαφορετικού ποδιού που βρίσκεται στον αέρα. Όμως, κάθε φάση του βαδίσματος την χωρίζουμε σε περισσότερες φάσεις κατά την διάρκεια της επίλυσης του προβλήματος. Αυτό γίνεται για δύο λόγους: Ο πρώτος είναι ότι μια φάση, που βρίσκεται το πόδι στον αέρα, περιλαμβάνει την άνοδο του ποδιού από το έδαφος, την κίνηση του στον αέρα και την κάθοδο του στο έδαφος. Ο δεύτερος λόγος είναι ότι δεν μπορούμε να παρέχουμε στο ρομπότ την θέση των ποδιών μονό σε κάθε φάση από τις τέσσερις χωρίς ενδιάμεσα σημεία. Για αυτούς τους λόγους, κάθε μια από τις τέσσερις φάσεις του βαδίσματος χωρίζεται σε δέκα πέντε φάσεις. Οπότε, συνολικά για ένα πλήρη κύκλο βαδίσματος έχουμε εξήντα φάσεις, δηλαδή το πρόβλημα της βελτιστοποίησης θα επιλυθεί εξήντα φορές. Ο αριθμός των φάσεων καθορίστηκε έπειτα από πειραματική διερεύνηση και κρίθηκε ότι είναι ο ελάχιστος δυνατός που παρέχει ικανοποιητικό αποτέλεσμα.

Στην παράγραφο που ακολουθεί περιγράφεται αναλυτικά ο ορισμός του προβλήματος της βελτιστοποίησης που επιλύεται με τους κατάλληλους περιορισμούς για κάθε φάση του βαδίσματος. Για την καλύτερη κατανόηση των παρακάτω, ως φάσεις Α θα αναφέρονται οι τέσσερις φάσεις που βρίσκεται κάποιο πόδι στον αέρα

και ως φάσεις Β θα αναφέρονται οι φάσεις που χωρίζεται κάθε φάση Α. Στο σχήμα 4.1 φαίνονται οι 15 φάσεις Β για μια φάση Α.



Σχήμα 4.1: Οι 15 φάσεις Β που ορίζουν μια φάση Α

4.1.1 Μεταβλητές βελτιστοποίησης

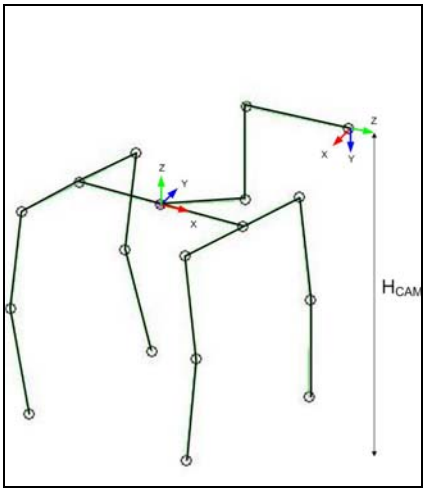
Οι μεταβλητές που επιθυμούμε να βελτιστοποιήσουμε είναι οι 15 γωνίες των συνδέσμων των ποδιών και του κεφαλιού $[q_1, q_2, ..., q_{15}]$, ώστε να ικανοποιούνται η αντικειμενική συνάρτηση και οι περιορισμοί

4.1.2 Αντικειμενική συνάρτηση βελτιστοποίησης

Η αντικειμενική συνάρτηση που θέλουμε να ελαχιστοποιήσουμε είναι:

$$f(q) = (h_{CAM} - h_{target})^2 \qquad (4.1)$$

όπου h_{CAM} είναι το ύψος της κάμερας από το έδαφος σχέση 3.35 και h_{target} είναι το επιθυμητό ύψος σταθεροποίησης της κάμερας (σχήμα 4.2)



Σχήμα 4.2: Ύψος κάμερας από το έδαφος

Στις επόμενες παραγράφους περιγράφονται αναλυτικά οι περιορισμοί της βελτιστοποίησης προκύπτουν από το βάδισμα, την ισορροπία και τα κατασκευαστικά στοιχεία του ρομπότ.

4.1.3 Περιορισμός των μεταβλητών βελτιστοποίησης

Ως περιορισμοί των μεταβλητών βελτιστοποίησης θα τεθούν οι μέγιστες και οι ελάχιστες τιμές που μπορούν να πάρουν οι τιμές των γωνιών των αρθρώσεων. Αυτές οι τιμές δίνονται από τον κατασκευαστή. Οπότε, $\mathbf{lb} \le \mathbf{q} \le \mathbf{ub}$,όπου $\mathbf{q} = [q_1, q_2, ...q_{15}]$ και $\mathbf{lb}, \mathbf{ub}$ δίνονται στο παράρτημα 1.

4.1.4 Ισοτικοί περιορισμοί

4.1.4.1 Περιορισμοί Βαδίσματος κατά τον x άξονα

Ως ισοτικοί περιορισμοί βαδίσματος θα τεθούν οι x συντεταγμένες (σχήμα 3.1) σύμφωνα με το κεφ.3.2.2.

$$
\begin{aligned}
c_{eq,1} &= x_1 - x_{lf} \\
c_{eq,2} &= x_2 - x_{rf} \\
c_{eq,3} &= x_3 - x_{rb} \\
c_{eq,4} &= x_4 - x_{lb}
\end{aligned}
\qquad (4.2)
$$

όπου x_1, x_2, x_3, x_4 δίνονται από τις σχέσεις 3.2, 3.5, 3.8, 3.11, αντίστοιχα και τα $x_{lf}, x_{rf}, x_{rb}, x_{lb}$ καθορίζονται από το κεφάλαιο 3.2.2. Πειραματικά, διαπιστώθηκε ότι ένα καλό εύρος τιμών για τις x συντεταγμένες για τα μπροστά και τα πίσω πόδια είναι:

$$x_{front,\max} = 135, x_{front,\min} = 65$$
$$x_{hind,\max} = -130, x_{hind,\min} = -60$$

Οι τιμές αυτές είναι ορισμένες βάση του συστήματος συνταγμένων Ο στο σχήμα 3.1. Σύμφωνα με το κεφάλαιο 3.2.2 έχουμε:

Πίνακας x συντεταγμένων για τις 4 φάσεις Α του βαδίσματος

Φάσεις	x_{lf}	x_{rf}	x_{rb}	x_{lb}
A-1	65.00	111.67	-106.67	-60.00
A-2	88.33	135.00	-83.33	-130.00
A-3	111.67	65.00	-60.00	-106.67
A-4	135.00	88.33	-130.00	-83.33

Όπως αναφέρεται παραπάνω κάθε φάση Α χωρίζεται σε 15 φάσεις Β. Από τις τιμές του παραπάνω πίνακα με γραμμική παρεμβολή έχουμε τα $x_{lf}, x_{rf}, x_{rb}, x_{lb}$ για τις 15 φάσεις Β κάθε φάσης Α.

4.1.4.2 Περιορισμοί Προσανατολισμού Κάμερας

Για να επιτύχουμε το αποτέλεσμα που επιθυμούμε για την σταθεροποίηση της εικόνας εκτός από το σταθερό ύψος πρέπει να έχουμε και καθορισμένο προσανατολισμό της κάμερας. Επιθυμούμε ο άξονας z της κάμερας να είναι παράλληλος με το έδαφος ή τα συστήματα συντεταγμένων Ο και της κάμερας να είναι όπως στο σχήμα 4.2. Το σύστημα συντεταγμένων στο σημείο 0, και συγκεκριμένα ο άξονας x δεν είναι πάντα παράλληλο με το έδαφός και εξαρτάται από τη διαφορά ύψους των ποδιών. Όμως, η υψομετρική διαφορά των ποδιών είναι μικρή άρα και η γωνία που σχηματίζεται μεταξύ του άξονα x και του εδάφους είναι μικρή και δεδομένου ότι υπάρχει σημαντικός τζόγος στις αρθρώσεις τους κεφαλιού, θεωρούμε ότι το σύστημα συντεταγμένων στο σημείο 0 και το σύστημα συντεταγμένων στο σημείο της κάμερας είναι όπως στο σχήμα 4.2. Άρα προκύπτει ότι για να είναι παράλληλος ο άξονας z με το έδαφος πρέπει να έχουμε από τον πίνακα R_{CAM} (σχ.3.17):

$$R_{CAM} = \begin{bmatrix} c_{13} * s_{14} & -c_{13} * c_{14} * c_{15} + s_{13} * s_{15} & -c_{13} * c_{14} * s_{15} - s_{13} * c_{15} \\ -c_{14} & -s_{14} * c_{15} & -s_{14} * s_{15} \\ s_{13} * s_{14} & -s_{13} * c_{14} * c_{15} - c_{13} * s_{15} & -s_{13} * c_{14} * s_{15} + c_{13} * c_{15} \end{bmatrix} = \begin{bmatrix} 0 & 0 & -1 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad (4.3)$$

4.1.4.3 Περιορισμοί ανύψωσης - τοποθέτησης ποδιού

Για την επίτευξη του βαδίσματος χρειάζεται ενώ όλα τα πόδια είναι τοποθετημένα στο έδαφος, το πόδι που βρίσκεται σε θέση για ανύψωση, να ανυψωθεί από το έδαφος, να κινηθεί χωρίς να έρθει σε επαφή με το έδαφος και να τοποθετηθεί πάλι στο έδαφος σε κατάλληλο ύψος χωρίς να επηρεάσει την ομαλότητα του βαδίσματος. Όπως αναφέρεται παραπάνω το βάδισμα χωρίζεται σε 4 φάσεις, σύμφωνα με το πόδι που βρίσκεται στον αέρα, που τις ονομάσαμε φάσεις Α και κάθε φάση Α, σπάει σε 15 μέρη, που τα ονομάσαμε φάσεις Β. Σε κάθε φάση Α κάθε πόδι πρέπει να ανυψωθεί, να κινηθεί στον αέρα και να ακουμπήσει πάλι στο έδαφος. Για να το πετύχουμε αυτό ορίζουμε ότι στην φάση Β-1 το πόδι θα ανυψωθεί από το έδαφος, στις φάσεις Β-2 έως Β-14 το πόδι θα κινείται χωρίς να ακουμπά το έδαφος και στην φάση Β-15 το πόδι θα τοποθετηθεί στο έδαφος. Τα υπόλοιπα πόδια βρίσκονται στο έδαφος και σπρώχνουν το σώμα εμπρός.

Για να καθορίσουμε την ανύψωση του ποδιού στην φάση Β-1 θέτουμε τον εξής περιορισμό:

$$z_{leg} - c_1 = z_0 \quad (4.4)$$

όπου z_{leg} είναι ανάλογα με το ποίο πόδι βρίσκεται στον αέρα το z_1 ή z_2 ή z_3 ή z_4, σχέσεις 3.4, 3.7, 3.10, 3.13, αντίστοιχα. c_1 είναι ένας σταθερός παράγοντας που καθορίζει το ύψος της ανύψωσης από το έδαφος, πειραματικά καθορίστηκε $c_1 = 5mm$. Και z_0 είναι το ύψος του σώματος του ρομπότ από το έδαφος, όπως καθορίζεται στο κεφ. 3.4 σχέση 3.34.

Για τις φάσεις Β-2 έως και Β-14, επιθυμούμε το πόδι να παραμένει στον αέρα και ο περιορισμός γίνεται:

$$z_{leg} - c_2 = z_0 \quad (4.5)$$

όπου c_2 σταθερός παράγοντάς που πειραματικά καθορίστηκε $c_2 = 11mm$.

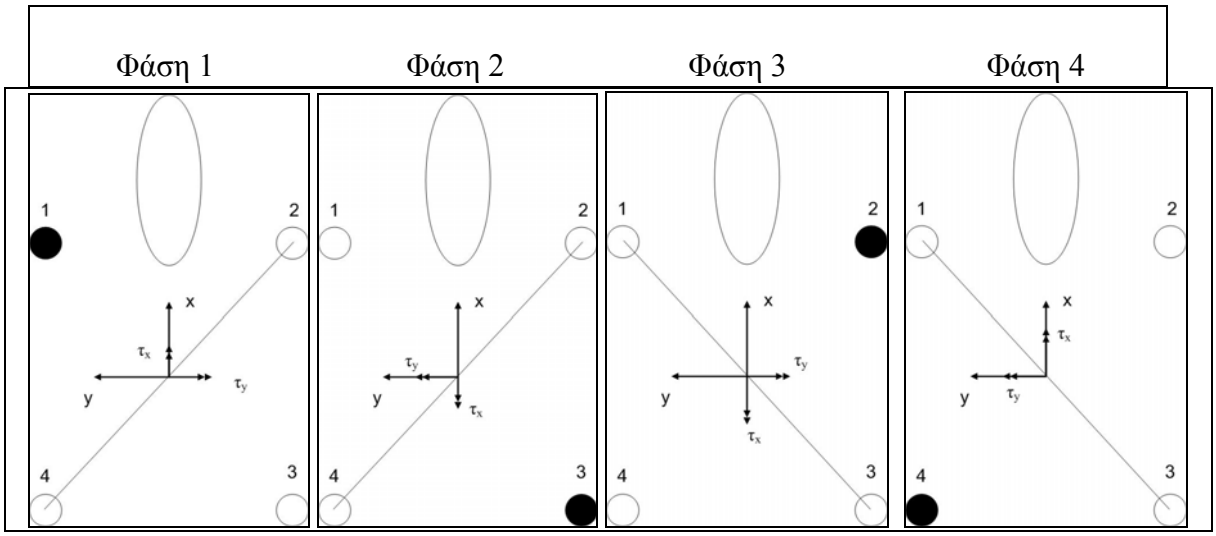
Για την φάση B-15, επιθυμούμε το πόδι να τοποθετηθεί στο έδαφος, άρα είναι προφανές ότι το ύψος του ποδιού που βρίσκονταν στον αέρα πρέπει να εξισωθεί με το ύψος του σώματος του ρομπότ. Άρα:

$$z_{leg} = z_0 \tag{4.6}$$

4.1.4 Ανισοτικοί περιορισμοί

4.1.4.1 Περιορισμοί Ισορροπίας

Ως περιορισμοί ισορροπίας θα τεθούν η ροπή στους άξονες x, y ως προς τον διαγώνιο άξονα των ποδιών που βρίσκονται στο έδαφος, όπως έχει οριστεί στο κεφ. 3.3.1 ώστε να έχει φορά τέτοια που να βοηθάει στην ισορροπία του ρομπότ. Στο σχήμα 4.3 φαίνεται η επιθυμητή φορά της ροπής στους άξονες x, y σύμφωνα με το πόδι που βρίσκεται στον αέρα (έχει σημειωθεί με μαύρο γέμισμα).



Σχήμα 4.3: Επιθυμητή φορά ροπής στους άξονες x,y σύμφωνα με το πόδι που βρίσκεται στον αέρα για να ισορροπεί το ρομπότ.

Σύμφωνα με σχήμα 4.3 όταν το μπροστά αριστερό πόδι (1) βρίσκεται στον αέρα οι ανισοτικοί περιορισμοί που προκύπτουν είναι:

$$\begin{aligned} &\tau_{x,1} \geq 0 \Leftrightarrow -\tau_{x,1} \leq 0 \\ &\tau_{y,1} \leq 0 \end{aligned} \tag{4.7}$$

Όλους τους ανισοτικούς περιορισμούς επιθυμούμε να είναι στην μορφή $c(x) \le 0$, διότι είναι η μορφή που θα εισαχθούν στο πρόβλημα της βελτιστοποίησης, γι' αυτό $\tau_{x,1} \ge 0 \Leftrightarrow -\tau_{x,1} \le 0$.

Ομοίως:

Μπροστά δεξί πόδι (2):

$$\begin{aligned} \tau_{x,2} &\le 0 \\ \tau_{y,2} &\le 0 \end{aligned} \tag{4.8}$$

Πίσω δεξί πόδι (3):

$$\begin{aligned} \tau_{x,3} &\le 0 \\ \tau_{y,3} &\ge 0 \Leftrightarrow -\tau_{y,3} \le 0 \end{aligned} \tag{4.9}$$

Πίσω αριστερό πόδι (4):

$$\begin{aligned} \tau_{x,4} &\ge 0 \Leftrightarrow -\tau_{x,4} \le 0 \\ \tau_{y,4} &\ge 0 \Leftrightarrow -\tau_{y,4} \le 0 \end{aligned} \tag{4.10}$$

Παρατηρώντας το σχήμα 4.3, και γνωρίζοντας ότι κάθε μια από αυτές τις φάσεις Α που σημειώνονται χωρίζονται σε επιπλέον φάσεις Β καταλαβαίνουμε ότι για να υπάρχει μια ομαλή μετάβαση από φάση σε φάση π.χ από την φάση 1 στην φάση 2 πρέπει η ροπή τ_x από $\tau_x > 0$ να γίνει $\tau_x < 0$, αντίστοιχα και η ροπή τ_y. Αυτό επιτυγχάνεται στις φάσεις Β, που έχουμε καθορίσει μεταξύ των φάσεων Α 1 και 2, διαμορφώνοντας τους περιορισμούς 4.7 ως εξής:

$$\begin{aligned} -\tau_{x,1} - b_1 &\le 0 \\ \tau_{y,1} - b_2 &\le 0 \end{aligned} \tag{4.11}$$

όπου b_1, b_2 σταθεροί όροι, που παίρνουν τιμές από 0 έως 0.3. Γνωρίζουμε ότι το μέτρο της ροπή σε κάθε άξονα μπορεί να έχει μέγιστη τιμή περίπου 0.3 $N*mm$ κατ' απόλυτη τιμή, γι' αυτό ορίζουμε το μέγιστο των b_1, b_2 σε 0.3. Έχουμε $b_1 = b_2 = 0$ σε κάθε πρώτη φάση Β και $b_1 = b_2 = 0.3$ σε κάθε τελευταία φάση Β. Οι τιμές στις ενδιάμεσες φάσεις υπολογίζονται με γραμμική παρεμβολή.

Ομοίως μετασχηματίζονται οι σχέσεις 4.8, 4.9, 4.10.

4.1.4.2 Περιορισμοί στους άξονες y και z για την επίτευξη ομαλού βαδίσματος.

Η επίλυση του προβλήματος της βελτιστοποίησης πρέπει να εξασφαλίζει την ομαλή μετάβαση από φάση σε φάση του βαδίσματος. Το βάδισμα χωρίζεται σε 60 φάσεις και αντίστοιχα επιλύεται 60 φορές το πρόβλημα της βελτιστοποίησης με διαφορετικούς περιορισμούς. Η επίλυση του προβλήματος της βελτιστοποίησης δεν πρέπει να είναι ανεξάρτητη για κάθε φάση, αλλά πρέπει να εξαρτάται από την αμέσως προηγούμενη επίλυση. Για κάθε φάση επίλυσης έχουμε καθορίσει τις x συντεταγμένες των ποδιών. Τι συμβαίνει όμως με τις y και τις z συντεταγμένες; Για να υπάρχει ομαλότητα στο περπάτημα οι συντεταγμένες των ποδιών στην ν επίλυση του προβλήματος πρέπει να απέχουν από τις συντεταγμένες της ν-1 επίλυσης μικρή απόσταση. Το πρόβλημα της βελτιστοποίησης θα μας δώσει λύση που θα πληρούνται όλοι οι υπόλοιποι περιορισμοί, χωρίς να είναι δυνατή η μετάβαση από φάση σε φάση, έχοντας ως αποτέλεσμα την αποτυχία του βαδίσματος. Γι' αυτό το λόγο πρέπει να οριστούν ανισοτικοί περιορισμοί στις y και z συντεταγμένες των ποδιών δύο διαδοχικών λύσεων για να υπάρχει ομαλή μετάβαση στις φάσεις.

Για τις y συντεταγμένες έχουμε τον περιορισμό:

$$\left(y_{leg}\left(q_{\nu}\right)-y_{leg}\left(q_{\nu-1}\right)\right)^2 \le d_1 \Leftrightarrow \left(y_{leg}\left(q_{\nu}\right)-y_{leg}\left(q_{\nu-1}\right)\right)^2 - d_1 \le 0 \qquad (4.12)$$

όπου $leg = 1,2,3,4$, ν είναι η φάση του βαδίσματος που επιλύεται και d_1 είναι σταθερός όρος που καθορίζει πόσο απέχουν οι y συντεταγμένες δύο διαδοχικών λύσεων, πειραματικά έχει καθοριστεί $d_1 = 3$. Ο παραπάνω περιορισμός πρέπει να εφαρμοστεί και για τα τέσσερα πόδια.

Ομοίως, για τις z συντεταγμένες πρέπει να εφαρμοστεί ο εξής περιορισμός:

$$\left(z_{leg}\left(q_{\nu}\right)-z_{leg}\left(q_{\nu-1}\right)\right)^2 \le d_2 \Leftrightarrow \left(z_{leg}\left(q_{\nu}\right)-z_{leg}\left(q_{\nu-1}\right)\right)^2 - d_2 \le 0 \qquad (4.13)$$

όπου d_2 είναι σταθερός όρος που καθορίζει πόσο απέχουν οι y συντεταγμένες δύο διαδοχικών λύσεων, πειραματικά έχει καθοριστεί $d_1 = 3$.Ο περιορισμός αυτός πρέπει να εφαρμοστεί μόνο για τα τρία πόδια που βρίσκονται στο έδαφος.

4.2 Επίλυση προβλήματος βελτιστοποίησης σε περιβάλλον Matlab

Για την επίλυση του προβλήματος της βελτιστοποίησης χρησιμοποιήθηκε το optimization toolbox του Matlab. Συγκεκριμένα, χρησιμοποιήθηκε η ρουτίνα fmincon, που είναι κατάλληλη για βελτιστοποίηση πολυπαραμετρικής μη γραμμικής εξίσωσης με περιορισμούς, χρησιμοποιώντας τον αλγόριθμο επίλυσης "active-set".

Για την επικοινωνία, την λήψη και αποστολή δεδομένων από και προς το ρομπότ στο περιβάλλον του Matlab, χρησιμοποιήθηκε η βιβλιοθήκη libUrbi [12], της προγραμματιστικής πλατφόρμας URBI [13],[14],[15]. Στο παράρτημα 1 γίνεται μια

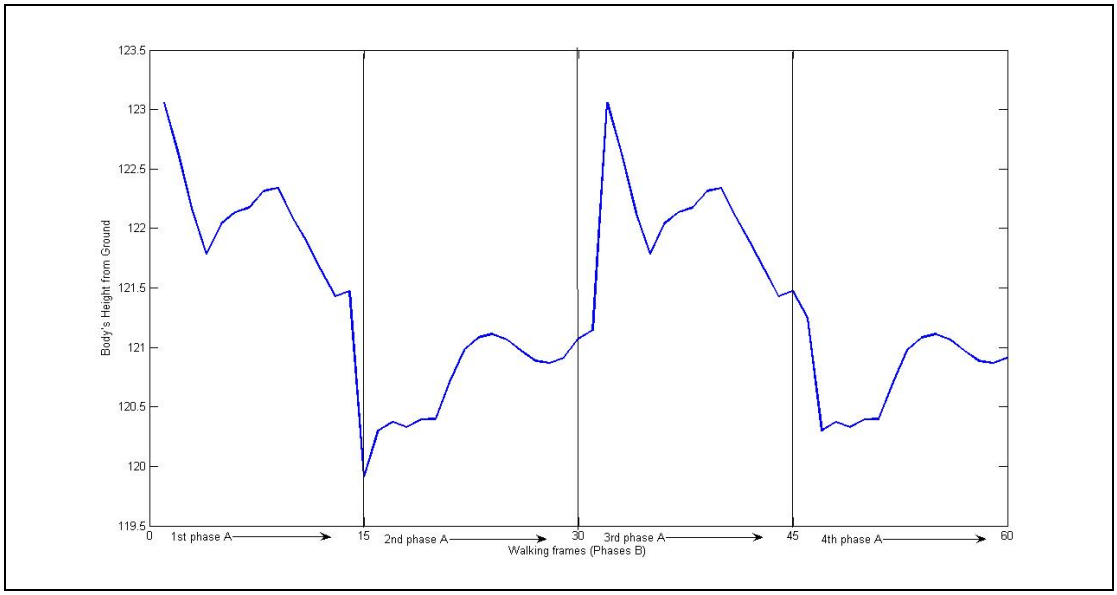
συνοπτική περιγραφή για τον τρόπο εγκατάστασης της βιβλιοθήκης libUrbi και κάποιων βασικών εντολών του URBI.

Όπως αναφέρεται στο κεφ.4.1 το πρόβλημα της βελτιστοποίησης επιλύεται 60 φορές (4 φάσεις Α αποτελούμενες από 15 φάσεις Β) για να οριστεί ένα πλήρης κύκλος βαδίσματος. Ο χρόνος επίλυσης κάθε φάσης Β κυμαίνεται από 100 ms έως 250 ms σε φορητό υπολογιστή με επεξεργαστή AMD Turion 64 X2 Mobile 2.00 GHz. Δηλαδή, ο χρόνος επίλυσης για έναν πλήρη κύκλο βαδίσματος κυμαίνεται από 6 έως 15 sec. Το πακέτο των δεδομένων, αποστέλλεται στο ρομπότ μετά το τέλος της επίλυσης του προβλήματος της βελτιστοποίησης όλων των φάσεων (60) του βαδίσματος.

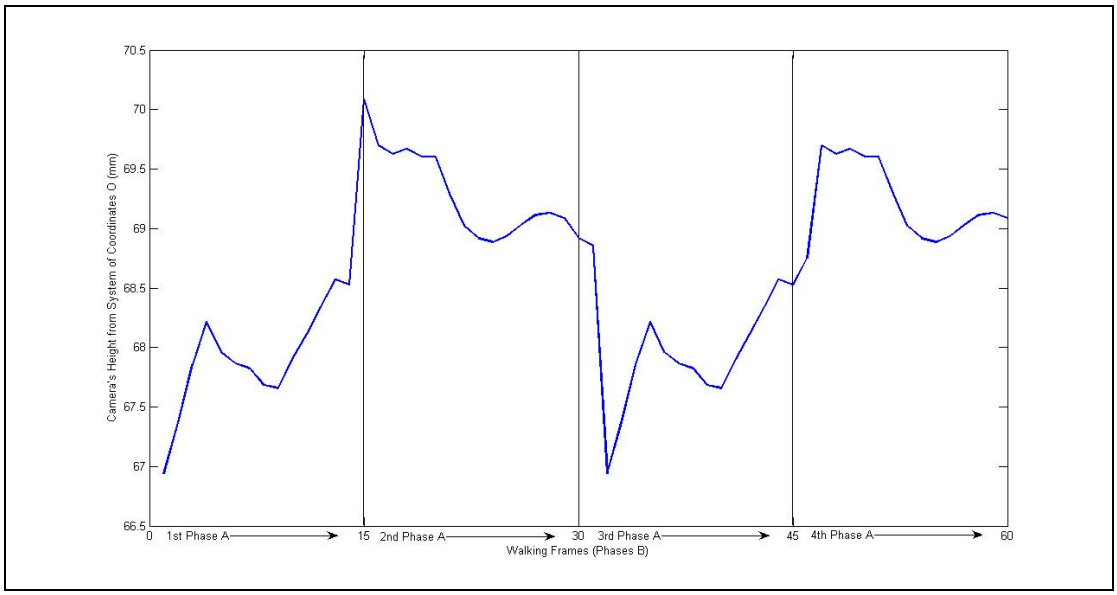
Κεφάλαιο 5 – Αποτελέσματα – Αξιολόγηση – Μελλοντικές Κατευθύνσεις

5.1 Αποτελέσματα

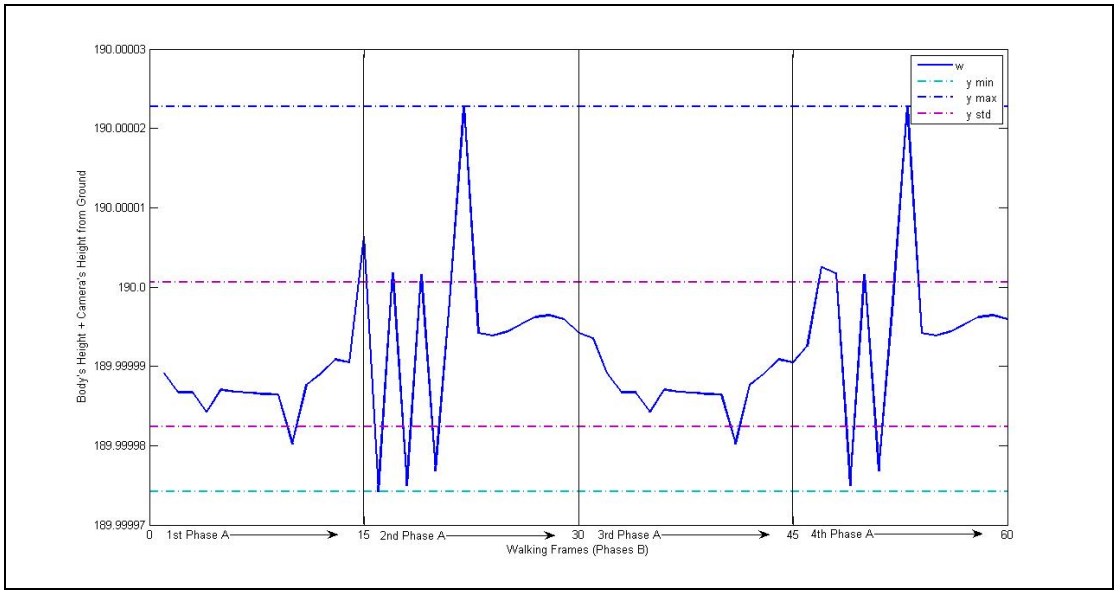
Το πρόβλημα επιλύθηκε για $h_{\_target} = 190mm$ σχέση 4.1. Παρακάτω παρουσιάζονται τρία διαγράμματα που δείχνουν το ύψος του σώματος από το έδαφος, το ύψος της κάμερας από το σύστημα συντεταγμένων Ο και το ύψος της κάμερας από το έδαφος (σχήμα 4.2), συναρτήσει των φάσεων του βαδίσματος.



Σχήμα 4.4: Το ύψος του σώματος από το έδαφος συναρτήσει των φάσεων του βαδίσματος



Σχήμα 4.5: Το ύψος της κάμερας από το σημείο 0 συναρτήσει των φάσεων του βαδίσματος



Σχήμα 4.6: Το ύψος της κάμερας από το έδαφος συναρτήσει των φάσεων του βαδίσματος

5.2 Αξιολόγηση

Σκοπός της παρούσας διπλωματικής ήταν η δημιουργία ενός βαδίσματος που θα βελτιώνει την τεχνητή όραση ενός τετράποδου ρομπότ κατά την διάρκεια της κίνησης του.

Τα αποτελέσματα που παρουσιάζονται παραπάνω κρίνονται ικανοποιητικά δεδομένου του μεγάλου αριθμού των παραμέτρων που χρειάστηκε να βελτιστοποιηθούν (15 γωνίες αρθρώσεων). Η επίλυση του προβλήματος της βελτιστοποίησης έγινε με επιτυχία σε κάθε φάση του βαδίσματος (σχήμα 4.6), ικανοποιώντας όλους τους περιορισμούς που είχαν τεθεί.

Παρατηρώντας το ρομπότ κατά την εκτέλεση του πειράματος με τα παραπάνω δεδομένα διαπιστώνουμε ότι ο μεγάλος τζόγος στις αρθρώσεις του κεφαλιού στερεί το τόσο καλό αποτέλεσμα που φαίνεται στο σχήμα 4.6, όμως υπάρχει σημαντική βελτίωση στην ποιότητα της εικόνας που λαμβάνουμε από το ρομπότ, συγκρίνοντας με το βάδισμα που έχει υλοποιήσει η κατασκευάστρια εταιρία χωρίς να είναι δώσει ιδιαίτερη έμφαση στον συγκεκριμένο τομέα, που με απλή παρατήρηση μπορούμε να το διαπιστώσουμε.

5.3 Μελλοντικές κατευθύνσεις

Το συγκεκριμένο μοντέλο για την δημιουργία του βαδίσματος μπορεί να βελτιωθεί αναπτύσσοντας ένα πληρέστερο κριτήριο ισορροπίας για το βάδισμα, λαμβάνοντας υπόψη και αδρανειακές δυνάμεις, που επιδρούν στο ρομπότ κατά την διάρκεια της κίνησης, Επιπλέον, το κριτήριο ισορροπίας θα είναι πληρέστερο εάν ληφθούν υπόψη όλα τα μέρη του ρομπότ.

Επίσης, βασιζόμενοι στο παρόν μοντέλο μπορεί να αναπτυχθεί κίνηση σε καμπύλη τροχιά και να δοκιμαστούν και άλλοι τύποι βαδίσματος. Τέλος, η κίνηση σε επίπεδο με κλίση είναι θα είναι μια ενδιαφέρουσα και απαιτητική προσπάθεια, που απασχολεί την επιστημονική κοινότητα ιδιαίτερα τα τελευταία χρόνια.

Βιβλιογραφία

[1] Y. Yokote, **"The Apertos Reflective Operating System, The Concept and Its Implementation"**. Sony Computer Science Laboratory Inc., 1992.

[2] Jun-ichiro Itoh, Yasuhito Yokote. **"Concurrent Object-Oriented Device Driver Programming in Apertos Operating System".** Sony Computer Science Laboratory Inc. Japan 1994.

[3] Ιστοσελίδα – http://openr.aibo.com

[4] Jean-Christophe Baillie, **"Design Principles for a Universal Robotic Software Platform and Application to URBI"**.

[5] Lorenzo Sciavicco, Bruno Siciliano, **"Modeling and Control of Robot Manipulators"**,1996

[6] Ευάγγελος Παπαδόπουλος, Κωνσταντίνος Κυριακόπουλος, **"Σημειώσεις Ρομποτικής"**, ΕΜΠ, Σχολή Μηχανολόγων Μηχανικών, Τομέας ΜΚ&ΑΕ

[7] Pablo Gonzalez de Santos, Elena Garcia and Joaquin Estremera, **"Quadrupedal Locomotion"**, Springer

[8] S. Peng, C. P. Lam, G.R. Cole, **"A Biologically Inspired Four Legged Walking Robot",** International Conference on Robotics & Automation, 2003

[9] Sebastiaan van Loon, **"Modeling and Gait Control of a Quadruped Robot"**, M.Sc. Thesis, University of Twente, 2005

[10] Stefan Bie, Johan Persson, **"Behavior-based Control of the ERS-7 AIBO Robot"**, Thesis, Department of Computer science, Faculty of Science, Lund University.

[11] Shugen Ma, Takashi Tomiyama, Hideyuki Wada, **"Omni-directional Walking of a Quadruped Robot"**, Proccedings of the 2002 IEEE/RDJ

[12] Matthieu Nottale, **"Introduction to libURBI for Urbi 1.x"**, http://www.gostai.com

[13] Jean-Christophe Baillie, **"URBI Language Specification"**, http://www.gostai.com, 2007

[14] **"URBI Doc For Aibo ERS2xx ERS7 and URBI 1.0 – Devices documentation"**, http://www.gostai.com

[15] Jean-Christophe Baillie, Mathieu Nottale, Benoit Pothier, Nicolas Despres, **"URBI Tutorial for Urbi 1.5"**, http://www.gostai.com

Παράρτημα 1 – Τεχνικές Λεπτομέρειες

Π.1.1 Ρυθμίσεις για σύνδεση ασυρμάτου δικτύου

Παρακάτω, θα ακολουθήσει η περιγραφή των βημάτων που πρέπει να ακολουθήσουμε ώστε να έχουμε επικοινωνία με το ρομπότ και να μπορούμε να το προγραμματίσουμε.

Το Aibo ERS-7 προγραμματίζεται μέσω του ροζ αφαιρούμενου memory stick. Όπως, έχει αναφερθεί παραπάνω για τον προγραμματισμό του χρησιμοποιήσαμε το URBI framework. Αντιγράφουμε, τα περιεχόμενα του φακέλου "**ERS7 URBI1.5 MS**" στο memory stick με την χρήση ενός card reader. Για να έχουμε επικοινωνία μέσω ασυρμάτου δικτύου πρέπει να δημιουργήσουμε ένα αρχείο σε έναν επεξεργαστή κειμένου με το όνομα «WLANCONF.TXT» και να το βάλουμε στο memory stick στην διαδρομή \OPEN-R\SYSTEM\CONF.

WLANCONF.TXT

```
HOSTNAME=AIBO
USE_DHCP=0
ETHER_IP=157.102.51.219
ETHER_NETMASK=255.255.255.0
APMODE=2
ESSID=AIBONET
IP_GATEWAY=157.102.51.200
WEPENABLE=1
WEPKEY=AIBO2
DNS_SERVER_1=157.102.51.42
```

- HOSTNAME: Το όνομα του ρομπότ.
- USE_DHCP: 1 για αυτόματη απόδοση ip στο ρομπότ, 0 για ip καθορισμένη από τον χρήστη. Ενδείκνυται η τιμή 0.
- ETHER_IP: Η ip του ρομπότ.
- APMODE: 1: για σύνδεση μέσω access point, 2: για ad hoc σύνδεση, όπως στην περίπτωση μας που συνδέεται απευθείας το ρομπότ με το laptop.
- ESSID: Το όνομα του δικτύου που θα γίνει η σύνδεση.
- IP_GATEWAY: Η ip του access point ή η ip του laptop σε περίπτωση ad hoc σύνδεσης.
- WEPENABLE: 1: για χρήση κωδικού για να γίνει σύνδεση στο ασύρματο δίκτυο για μεγαλύτερη ασφάλεια, 0: για να γίνεται σύνδεση χωρίς κωδικό.
- WEPKEY: Ο κωδικός για το ασύρματο δίκτυο.

Κάνοντας τα παραπάνω το AIBO θα συνδεθεί αυτόματα στο ασύρματο δίκτυο λίγα δευτερόλεπτα μετά την εκκίνηση του.

Τα αρχεία που αναφέρονται πιο πάνω υπάρχουν στο συνοδευτικό cd.

Π.1.2 Συνοπτική παρουσίαση Matlab libURBI

Για την χρήση του libURBI χρειάζεται να αντιγράψουμε τον φάκελο "Liburbi-matlab" (περιέχεται στο cd) στο current directory του matlab, που εργαζόμαστε και να κάνουμε δεξί κλικ -> Add to path -> Selected Folders and Subfolders. Αφού γίνει αυτή η διαδικασία μπορεί αμέσως να χρησιμοποιηθεί.

Για να δημιουργήσουμε μια σύνδεση μεταξύ του ρομπότ και του υπολογιστή μας, πληκτρολογούμε στο command window του matlab:

```
>>aibo=urbiConnect('157.102.51.219')
```

Η ip είναι αυτή που έχουμε ορίσει στο αρχείο ρύθμισης για το ασύρματο δίκτυο.

Έπειτα πρέπει να στείλουμε την παρακάτω εντολή για να ενεργοποιηθούν οι σερβοκινητήρες στο ρομπότ.

```
>>urbiSend(aibo,'motors on;')
```

Για να δώσουμε μεμονωμένα τιμή σε κάποια άρθρωση του ρομπότ, π.χ την άρθρωσή 1 του αριστερού μπροστά ποδιού στις 30^o , πληκτρολογούμε:

```
>>urbiSend(aibo,'leftLF1.val=30;')
```

Για να πάρουμε την μέτρηση του αισθητήρα στην άρθρωση 1 του μπροστά αριστερού ποδιού:

```
>>urbiGetVariable(aibo,'leftLF1.val;')
```

Τέλος για να παίρνουμε εικόνες από την κάμερα μέσω του matlab:

```
>>for i=1:2000
  a=urbiGetImage(aibo)
  imshow(a)
  end
```

Π.1.3 Μέγιστες και ελάχιστες τιμές των γωνιών των αρθρώσεων

q_i	$\min(q_i)$	$\max(q_i)$
q_1	-2.0944	2.3387
q_2	-0.1571	0.3000
q_3	0.7000	2.0769
q_4	-2.3387	2.0944
q_5	-0.1571	0.3000
q_6	0.7000	2.0769
q_7	-2.0944	2.3387
q_8	-0.1571	0.3000
q_9	0.7000	2.0769
q_{10}	-2.3387	2.0944
q_{11}	-0.1571	0.3000
q_{12}	0.7000	2.0769
q_{13}	-1.3788	0.0349
q_{14}	-1.5882	1.5882
q_{15}	-0.2793	0.7679

Οι τιμές είναι σε rad και οι μεταβλητές έχουν ορισθεί σύμφωνα με το σχήμα 3.3.

Π.1.4 Αντιστοιχία γωνιών με τις μεταβλητές URBI

q_i	Μεταβλητή URBI
q_1	legLF1
q_2	legLF2
q_3	legLF3
q_4	legRF1
q_5	legRF2
q_6	legRF3
q_7	legRH1
q_8	legRH2
q_9	legRH3
q_{10}	legLH1
q_{11}	legLH2
q_{12}	legLH3
q_{13}	neck
q_{14}	headPan
q_{15}	headTilt

Παράρτημα 2 – Κώδικας για την επίλυση του προβλήματος

walk.m

```
fr_min=65; %Front walk limits
fr_max=135;
hin_min=-130; %Hind walk limits
hin_max=-60;
stroke=[fr_max-fr_min;hin_max-hin_min];
step=stroke/3;
legOrder=[1 3 2 4];
%x coors for 4-phase walk
x_In_coors=[fr_min              fr_min+2*step(1)     hin_min+step(2)
hin_min+3*step(2);
            fr_min+step(1)      fr_min+3*step(1)     hin_min+2*step(2)    hin_min;
            fr_min+2*step(1)    fr_min               hin_min+3*step(2)
hin_min+step(2);
            fr_min+3*step(1)    fr_min+step(1)       hin_min
hin_min+2*step(2)];

fr=15; % frames per phase
xlf(1)=x_In_coors(1,1);
xrf(1)=x_In_coors(1,2);
xrb(1)=x_In_coors(1,3);
xlb(1)=x_In_coors(1,4);

%x coors generation
for i=1:4
    leg=legOrder(i);
        if leg==1
        sPF(1)=stroke(1)/fr;
        else
            sPF(1)=step(1)/fr;
        end
        if leg==2
        sPF(2)=stroke(1)/fr;
        else
            sPF(2)=step(1)/fr;
        end
        if leg==3
        sPF(3)=stroke(2)/fr;
        else
            sPF(3)=step(2)/fr;
        end
        if leg==4
        sPF(4)=stroke(2)/fr;
        else
            sPF(4)=step(2)/fr;
        end
    for j=1:fr
        k=(i-1)*fr+j;
        if leg==1
            xlf(k+1)=xlf(k)+sPF(1);
        else
            xlf(k+1)=xlf(k)-sPF(1);
        end
        if leg==2
            xrf(k+1)=xrf(k)+sPF(2);
        else
            xrf(k+1)=xrf(k)-sPF(2);
        end
        if leg==3
            xrb(k+1)=xrb(k)+sPF(3);
        else
            xrb(k+1)=xrb(k)-sPF(3);
        end
        if leg==4
            xlb(k+1)=xlb(k)+sPF(4);
        else
            xlb(k+1)=xlb(k)-sPF(4);
        end
    end


end
```

```
%b1,b2 stability factors
b1=0:.3/(fr-1):.3;
b1=[b1 zeros(1,fr)];
    for j=1:2
        b1=[b1 0:.3/(fr-1):.3];
    end
b2=b1;

x_coor=[xlf' xrf' xrb' xlb'];

% initial q
qP=zeros(1,15);


 for i=1:4 %4-phase

    leg=legOrder(i);
    for j=1:fr
        k=(i-1)*fr+j; %Total frames
        if  k==1 | k==fr+1 | k==2*fr+1 | k==3*fr+1 %First frame of each phase
            [q,fval]=opt(qP,leg,1,x_coor(k,:),b1(k),b2(k));
        elseif k==fr | k==2*fr | k==3*fr | k==4*fr %Last frame of each phase
            [q,fval]=opt(qP,leg,3,x_coor(k,:),b1(k),b2(k));
        else
            [q,fval]=opt(qP,leg,3,x_coor(k,:),b1(k),b2(k));
        end

         qf(k,:)=q;
         qP=q; %store qP from previous solution
    end
 end

 %send data to robot
 for l=1:15
    for i=1:4
        for j=1:fr
            k=(i-1)*fr+j;
            send_to_aibo(qf(k,:),aibo);
        end
    end
end
```

opt.m

```
%q(1)~q(3):left front leg
%q(4)~q(6):right front leg
%q(7)~q(9):right hind leg
%q(10)~q(12):left hind leg
%q(13)~q(15):camera


function [q,fval]=opt(qP,leg,ph,x_coor,b1,b2) %Optimization function

    %qP:solution from previous frame
    %leg: leg on air
    %ph:1->lift leg 2->leg on air 3->leg on ground
    %x_coor: x coordinates for legs

    %tic%start timer
    lb=[-2.3387; -.1571; .3; -2.3387; -.1571; .3; -2.0944; -.1571; .3; -2.3387; -
.1571; .3; -1.3788; -1.5882; -.2793]; %lower bound
    ub=[2.3387; .7; 2.0769; 2.0944; .7; 2.0769; 2.3387; .7; 2.0769; 2.0944; .7;
2.0769; .0349; 1.5882; .7679]; %upper bound

    x0=[-1.1 0 1.5 -1.1 0 1 0 0 1 0 0 1 0 0 0 0 0]; %initial solution
    options = optimset('Algorithm','active-set');%Optimization options
    %matlab function fmincon
    [q,fval] =
fmincon(@(q)obj_func(q,leg,ph),x0,[],[],[],[],lb,ub,@(q)mycon(q,qP,leg,ph,x_coor,b1,b2
),options);
    %toc%stop timer
end
function h_head=obj_func(q,leg,ph)%Objective Function
    h_head=headHeight(q,leg,ph);
    h_target=190;% target height
    h_head=(h_head-h_target)^2;
```

```
end
function [F,Feq]=mycon(q,qP,leg,ph,x_coor,b1,b2)%Constraints
    %F:inequal constraints
    %Feq:equal constraints
    Tor=torque(q,0,0.9132,leg)+torque(q,1,.18,leg); %Torque constraint
    [h_body,h_UpLeg]=bodyHeight(q,leg,ph);
    [yPosP,y1,y2,y3,y4]=ycons(q,qP,leg);%y Constraints
    xPos=xLegPos(q,x_coor);% x Constraints
    [z1,z2,z3,z4]=calculate_z(q);
    [zP1,zP2,zP3,zP4]=calculate_z(qP);
    d2=3; %z constraints
    c1=5; %leg lift
    c2=11;

    F=[ Tor(1)-b1;
        Tor(2)-b2;
        %Camera Orientation Constrains
        -(-cos(q(13))*cos(q(14))*sin(q(15)-1.5708)-sin(q(13))*cos(q(15)-1.5708))-.999;
        -cos(q(13))*cos(q(14))*cos(q(15)-1.5708)+sin(q(13))*sin(q(15)-1.5708)-.001;
        -(-cos(q(13))*cos(q(14))*cos(q(15)-1.5708)+sin(q(13))*sin(q(15)-1.5708))-.001;
        %y constraints
        yPosP(1);
        yPosP(2);
        yPosP(3);
        yPosP(4);
      ];
    if leg==1
        F=[ F;
            (zP2-z2)^2-d2;
            (zP3-z3)^2-d2;
            (zP4-z4)^2-d2
          ];
    elseif leg==3
        F=[ F;
            (zP2-z2)^2-d2;
            (zP1-z1)^2-d2;
            (zP4-z4)^2-d2
           ];
    elseif leg==2
        F=[ F;
            (zP3-z3)^2-d2;
            (zP1-z1)^2-d2;
            (zP4-z4)^2-d2
          ];
    elseif leg==4
        F=[ F;
            (zP3-z3)^2-m;
            (zP1-z1)^2-m;
            (zP2-z2)^2-m
         ];
    end

    if ph==1
        Feq=[   q(14);%q(14)=0
                xPos;
                %leg lift constraint
                h_UpLeg+h_body-c1;
            ];

    elseif ph==2
        Feq=[   q(14);%q(14)=0
                xPos;
                %leg lift constraint
                h_UpLeg+h_body-c2;
            ];
   elseif ph==3
        Feq=[   q(14);%q(14)=0
                xPos;
                %leg lift constraint
                h_UpLeg+h_body;
            ];
    end


end
function h_head=headHeight(q,leg,ph)% Head's height from ground
```

```
    h_body=bodyHeight(q,leg,ph); %Body's height from ground (negative)
    h_body=-h_body;
    %Camera's height h from point 0
    h=39/2-4053/50*sin(q(13))*cos(q(14))*sin(q(15)-
1.5708)+4053/50*cos(q(13))*cos(q(15)-1.5708)+73/5*sin(q(13))*cos(q(14))*cos(q(15)-
1.5708)+73/5*cos(q(13))*sin(q(15)-1.5708)+80*cos(q(13));
    h_head=h_body+h;
end
function [z1,z2,z3,z4]=calculate_z(q)

        z2=70.1646*sin(q(4)-1.5708)*cos(q(5))*cos(q(6))+70.1646*cos(q(4)-
1.5708)*sin(q(6))-4.7*sin(q(4)-1.5708)*sin(q(5))+139/2*sin(q(4)-
1.5708)*cos(q(5))+9*cos(q(4)-1.5708);
        z4=-70.1646*sin(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*cos(q(10)+1.5708)*sin(q(12))+4.7*sin(q(10)+1.5708)*sin(q(11))-
69.5*sin(q(10)+1.5708)*cos(q(11))-9*cos(q(10)+1.5708);
        z1=70.1646*sin(q(1)-1.5708)*cos(q(2))*cos(q(3))+70.1646*cos(q(1)-
1.5708)*sin(q(3))-4.7*sin(q(1)-1.5708)*sin(q(2))+69.5*sin(q(1)-
1.5708)*cos(q(2))+9*cos(q(1)-1.5708);
        z3=-70.1646*sin(q(7)+1.5708)*cos(q(8))*cos(q(9))-
70.1646*cos(q(7)+1.5708)*sin(q(9))+4.7*sin(q(7)+1.5708)*sin(q(8))-
69.5*sin(q(7)+1.5708)*cos(q(8))-9*cos(q(7)+1.5708);

end

function [h_body,h_UpLeg]=bodyHeight(q,leg,ph)
        %h_body: Body's height from ground (negative)
        %h_UpLeg: Leg's height from ground which is in transfer
        %z coordinates

        [z1,z2,z3,z4]=calculate_z(q);
       %         if ph==0 % all legs on ground
%             h_body=mean([mean([z1 z2]) mean([z3 z4])]);
%         elseif ph==1 % one leg in transfer on air
            if leg==1
                h_body=mean([z2 mean([z3 z4])]);
                h_UpLeg=-z1;
            elseif leg==2
                h_body=mean([z1 mean([z3 z4])]);
                h_UpLeg=-z2;
            elseif leg==3
                h_body=mean([z4 mean([z1 z2])]);
                h_UpLeg=-z3;
            elseif leg==4
                h_body=mean([z3 mean([z1 z2])]);
                h_UpLeg=-z4;
            end
%         end
end
function F=torque(q,point,m,leg) %Torque constraints
    g=[0 0 -9.81];%Gravity vector
    if point==0%Body
        xc=0;
        yc=0;
        zc=0;
    elseif point==1 %Head
        xc=2079/100*cos(q(13))*cos(q(14))*cos(q(15)-1.5708)-
2079/100*sin(q(13))*sin(q(15)-1.5708)+135/2-80*sin(q(13));
        yc=2079/100*sin(q(14))*cos(q(15)-1.5708);
        zc=2079/100*sin(q(13))*cos(q(14))*cos(q(15)-
1.5708)+2079/100*cos(q(13))*sin(q(15)-1.5708)+39/2+80*cos(q(13));
        xc=xc*10^-3;
        yc=yc*10^-3;
        zc=zc*10^-3;
    end
    %torqueX
    if leg==1 || leg==3
        F(1)=(yc+.0702*sin(q(5))*cos(q(6))+.0625+.0047*cos(q(5))+.0695*sin(q(5))-
1/1000*(-(70.1646*cos(q(4)-1.5708)*cos(q(5))*cos(q(6))-70.1646*sin(q(4)-
1.5708)*sin(q(6))-47/10*cos(q(4)-1.5708)*sin(q(5))+139/2*cos(q(4)-
1.5708)*cos(q(5))+65-9*sin(q(4)-1.5708)-
xc)*(70.1646*cos(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*sin(q(10)+1.5708)*sin(q(12))-
47/10*cos(q(10)+1.5708)*sin(q(11))+139/2*cos(q(10)+1.5708)*cos(q(11))-130-
9*sin(q(10)+1.5708)-70.1646*cos(q(4)-1.5708)*cos(q(5))*cos(q(6))+70.1646*sin(q(4)-
1.5708)*sin(q(6))+47/10*cos(q(4)-1.5708)*sin(q(5))-139/2*cos(q(4)-
```

```
1.5708)*cos(q(5))+9*sin(q(4)-1.5708))-(-70.1646*sin(q(5))*cos(q(6))-125/2-
47/10*cos(q(5))-139/2*sin(q(5))-
yc)*(70.1646*sin(q(11))*cos(q(12))+125+47/10*cos(q(11))+139/2*sin(q(11))+70.1646*sin(q
(5))*cos(q(6))+47/10*cos(q(5))+139/2*sin(q(5)))-(70.1646*sin(q(4)-
1.5708)*cos(q(5))*cos(q(6))+70.1646*cos(q(4)-1.5708)*sin(q(6))-47/10*sin(q(4)-
1.5708)*sin(q(5))+139/2*sin(q(4)-1.5708)*cos(q(5))+9*cos(q(4)-1.5708)-zc)*(-
70.1646*sin(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*cos(q(10)+1.5708)*sin(q(12))+47/10*sin(q(10)+1.5708)*sin(q(11))-
139/2*sin(q(10)+1.5708)*cos(q(11))-9*cos(q(10)+1.5708)-70.1646*sin(q(4)-
1.5708)*cos(q(5))*cos(q(6))-70.1646*cos(q(4)-1.5708)*sin(q(6))+47/10*sin(q(4)-
1.5708)*sin(q(5))-139/2*sin(q(4)-1.5708)*cos(q(5))-9*cos(q(4)-
1.5708)))/((70.1646*cos(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*sin(q(10)+1.5708)*sin(q(12))-
47/10*cos(q(10)+1.5708)*sin(q(11))+139/2*cos(q(10)+1.5708)*cos(q(11))-130-
9*sin(q(10)+1.5708)-70.1646*cos(q(4)-1.5708)*cos(q(5))*cos(q(6))+70.1646*sin(q(4)-
1.5708)*sin(q(6))+47/10*cos(q(4)-1.5708)*sin(q(5))-139/2*cos(q(4)-
1.5708)*cos(q(5))+9*sin(q(4)-
1.5708))^2+(70.1646*sin(q(11))*cos(q(12))+125+47/10*cos(q(11))+139/2*sin(q(11))+70.164
6*sin(q(5))*cos(q(6))+47/10*cos(q(5))+139/2*sin(q(5)))^2+(-
70.1646*sin(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*cos(q(10)+1.5708)*sin(q(12))+47/10*sin(q(10)+1.5708)*sin(q(11))-
139/2*sin(q(10)+1.5708)*cos(q(11))-9*cos(q(10)+1.5708)-70.1646*sin(q(4)-
1.5708)*cos(q(5))*cos(q(6))-70.1646*cos(q(4)-1.5708)*sin(q(6))+47/10*sin(q(4)-
1.5708)*sin(q(5))-139/2*sin(q(4)-1.5708)*cos(q(5))-9*cos(q(4)-
1.5708))^2)*(70.1646*sin(q(11))*cos(q(12))+125+47/10*cos(q(11))+139/2*sin(q(11))+70.16
46*sin(q(5))*cos(q(6))+47/10*cos(q(5))+139/2*sin(q(5))))*m*g(3)+(-zc+.0702*sin(q(4)-
1.5708)*cos(q(5))*cos(q(6))+.0702*cos(q(4)-1.5708)*sin(q(6))-.0047*sin(q(4)-
1.5708)*sin(q(5))+.0695*sin(q(4)-1.5708)*cos(q(5))+9/1000*cos(q(4)-1.5708)+1/1000*(-
(70.1646*cos(q(4)-1.5708)*cos(q(5))*cos(q(6))-70.1646*sin(q(4)-1.5708)*sin(q(6))-
47/10*cos(q(4)-1.5708)*sin(q(5))+139/2*cos(q(4)-1.5708)*cos(q(5))+65-9*sin(q(4)-
1.5708)-xc)*(70.1646*cos(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*sin(q(10)+1.5708)*sin(q(12))-
47/10*cos(q(10)+1.5708)*sin(q(11))+139/2*cos(q(10)+1.5708)*cos(q(11))-130-
9*sin(q(10)+1.5708)-70.1646*cos(q(4)-1.5708)*cos(q(5))*cos(q(6))+70.1646*sin(q(4)-
1.5708)*sin(q(6))+47/10*cos(q(4)-1.5708)*sin(q(5))-139/2*cos(q(4)-
1.5708)*cos(q(5))+9*sin(q(4)-1.5708))-(-70.1646*sin(q(5))*cos(q(6))-125/2-
47/10*cos(q(5))-139/2*sin(q(5))-
yc)*(70.1646*sin(q(11))*cos(q(12))+125+47/10*cos(q(11))+139/2*sin(q(11))+70.1646*sin(q
(5))*cos(q(6))+47/10*cos(q(5))+139/2*sin(q(5)))-(70.1646*sin(q(4)-
1.5708)*cos(q(5))*cos(q(6))+70.1646*cos(q(4)-1.5708)*sin(q(6))-47/10*sin(q(4)-
1.5708)*sin(q(5))+139/2*sin(q(4)-1.5708)*cos(q(5))+9*cos(q(4)-1.5708)-zc)*(-
70.1646*sin(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*cos(q(10)+1.5708)*sin(q(12))+47/10*sin(q(10)+1.5708)*sin(q(11))-
139/2*sin(q(10)+1.5708)*cos(q(11))-9*cos(q(10)+1.5708)-70.1646*sin(q(4)-
1.5708)*cos(q(5))*cos(q(6))-70.1646*cos(q(4)-1.5708)*sin(q(6))+47/10*sin(q(4)-
1.5708)*sin(q(5))-139/2*sin(q(4)-1.5708)*cos(q(5))-9*cos(q(4)-
1.5708)))/((70.1646*cos(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*sin(q(10)+1.5708)*sin(q(12))-
47/10*cos(q(10)+1.5708)*sin(q(11))+139/2*cos(q(10)+1.5708)*cos(q(11))-130-
9*sin(q(10)+1.5708)-70.1646*cos(q(4)-1.5708)*cos(q(5))*cos(q(6))+70.1646*sin(q(4)-
1.5708)*sin(q(6))+47/10*cos(q(4)-1.5708)*sin(q(5))-139/2*cos(q(4)-
1.5708)*cos(q(5))+9*sin(q(4)-
1.5708))^2+(70.1646*sin(q(11))*cos(q(12))+125+47/10*cos(q(11))+139/2*sin(q(11))+70.164
6*sin(q(5))*cos(q(6))+47/10*cos(q(5))+139/2*sin(q(5)))^2+(-
70.1646*sin(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*cos(q(10)+1.5708)*sin(q(12))+47/10*sin(q(10)+1.5708)*sin(q(11))-
139/2*sin(q(10)+1.5708)*cos(q(11))-9*cos(q(10)+1.5708)-70.1646*sin(q(4)-
1.5708)*cos(q(5))*cos(q(6))-70.1646*cos(q(4)-1.5708)*sin(q(6))+47/10*sin(q(4)-
1.5708)*sin(q(5))-139/2*sin(q(4)-1.5708)*cos(q(5))-9*cos(q(4)-1.5708))^2)*(-
70.1646*sin(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*cos(q(10)+1.5708)*sin(q(12))+47/10*sin(q(10)+1.5708)*sin(q(11))-
139/2*sin(q(10)+1.5708)*cos(q(11))-9*cos(q(10)+1.5708)-70.1646*sin(q(4)-
1.5708)*cos(q(5))*cos(q(6))-70.1646*cos(q(4)-1.5708)*sin(q(6))+47/10*sin(q(4)-
1.5708)*sin(q(5))-139/2*sin(q(4)-1.5708)*cos(q(5))-9*cos(q(4)-1.5708)))*m*g(2);
    elseif leg==2 || leg==4
        F(1)=(yc-.0702*sin(q(2))*cos(q(3))-.0625-.0047*cos(q(2))-.0695*sin(q(2))-
1/1000*(-(70.1646*cos(q(1)-1.5708)*cos(q(2))*cos(q(3))-70.1646*sin(q(1)-
1.5708)*sin(q(3))-47/10*cos(q(1)-1.5708)*sin(q(2))+139/2*cos(q(1)-
1.5708)*cos(q(2))+65-9*sin(q(1)-1.5708)-
xc)*(70.1646*cos(q(7)+1.5708)*cos(q(8))*cos(q(9))-70.1646*sin(q(7)+1.5708)*sin(q(9))-
47/10*cos(q(7)+1.5708)*sin(q(8))+139/2*cos(q(7)+1.5708)*cos(q(8))-130-
9*sin(q(7)+1.5708)-70.1646*cos(q(1)-1.5708)*cos(q(2))*cos(q(3))+70.1646*sin(q(1)-
1.5708)*sin(q(3))+47/10*cos(q(1)-1.5708)*sin(q(2))-139/2*cos(q(1)-
1.5708)*cos(q(2))+9*sin(q(1)-1.5708))-
(70.1646*sin(q(2))*cos(q(3))+125/2+47/10*cos(q(2))+139/2*sin(q(2))-yc)*(-
70.1646*sin(q(8))*cos(q(9))-125-47/10*cos(q(8))-139/2*sin(q(8))-
70.1646*sin(q(2))*cos(q(3))-47/10*cos(q(2))-139/2*sin(q(2)))-(70.1646*sin(q(1)-
```

```
1.5708)*cos(q(2))*cos(q(3))+70.1646*cos(q(1)-1.5708)*sin(q(3))-47/10*sin(q(1)-
1.5708)*sin(q(2))+139/2*sin(q(1)-1.5708)*cos(q(2))+9*cos(q(1)-1.5708)-zc)*(-
70.1646*sin(q(7)+1.5708)*cos(q(8))*cos(q(9))-
70.1646*cos(q(7)+1.5708)*sin(q(9))+47/10*sin(q(7)+1.5708)*sin(q(8))-
139/2*sin(q(7)+1.5708)*cos(q(8))-9*cos(q(7)+1.5708)-70.1646*sin(q(1)-
1.5708)*cos(q(2))*cos(q(3))-70.1646*cos(q(1)-1.5708)*sin(q(3))+47/10*sin(q(1)-
1.5708)*sin(q(2))-139/2*sin(q(1)-1.5708)*cos(q(2))-9*cos(q(1)-
1.5708)))/((70.1646*cos(q(7)+1.5708)*cos(q(8))*cos(q(9))-
70.1646*sin(q(7)+1.5708)*sin(q(9))-
47/10*cos(q(7)+1.5708)*sin(q(8))+139/2*cos(q(7)+1.5708)*cos(q(8))-130-
9*sin(q(7)+1.5708)-70.1646*cos(q(1)-1.5708)*cos(q(2))*cos(q(3))+70.1646*sin(q(1)-
1.5708)*sin(q(3))+47/10*cos(q(1)-1.5708)*sin(q(2))-139/2*cos(q(1)-
1.5708)*cos(q(2))+9*sin(q(1)-1.5708))^2+(-70.1646*sin(q(8))*cos(q(9))-125-
47/10*cos(q(8))-139/2*sin(q(8))-70.1646*sin(q(2))*cos(q(3))-47/10*cos(q(2))-
139/2*sin(q(2)))^2+(-70.1646*sin(q(7)+1.5708)*cos(q(8))*cos(q(9))-
70.1646*cos(q(7)+1.5708)*sin(q(9))+47/10*sin(q(7)+1.5708)*sin(q(8))-
139/2*sin(q(7)+1.5708)*cos(q(8))-9*cos(q(7)+1.5708)-70.1646*sin(q(1)-
1.5708)*cos(q(2))*cos(q(3))-70.1646*cos(q(1)-1.5708)*sin(q(3))+47/10*sin(q(1)-
1.5708)*sin(q(2))-139/2*sin(q(1)-1.5708)*cos(q(2))-9*cos(q(1)-1.5708))^2)*(-
70.1646*sin(q(8))*cos(q(9))-125-47/10*cos(q(8))-139/2*sin(q(8))-
70.1646*sin(q(2))*cos(q(3))-47/10*cos(q(2))-139/2*sin(q(2))))*m*g(3)+(-
zc+.0702*sin(q(1)-1.5708)*cos(q(2))*cos(q(3))+.0702*cos(q(1)-1.5708)*sin(q(3))-
.0047*sin(q(1)-1.5708)*sin(q(2))+.0695*sin(q(1)-1.5708)*cos(q(2))+9/1000*cos(q(1)-
1.5708)+1/1000*(-(70.1646*cos(q(1)-1.5708)*cos(q(2))*cos(q(3))-70.1646*sin(q(1)-
1.5708)*sin(q(3))-47/10*cos(q(1)-1.5708)*sin(q(2))+139/2*cos(q(1)-
1.5708)*cos(q(2))+65-9*sin(q(1)-1.5708)-
xc)*(70.1646*cos(q(7)+1.5708)*cos(q(8))*cos(q(9))-70.1646*sin(q(7)+1.5708)*sin(q(9))-
47/10*cos(q(7)+1.5708)*sin(q(8))+139/2*cos(q(7)+1.5708)*cos(q(8))-130-
9*sin(q(7)+1.5708)-70.1646*cos(q(1)-1.5708)*cos(q(2))*cos(q(3))+70.1646*sin(q(1)-
1.5708)*sin(q(3))+47/10*cos(q(1)-1.5708)*sin(q(2))-139/2*cos(q(1)-
1.5708)*cos(q(2))+9*sin(q(1)-1.5708))-
(70.1646*sin(q(2))*cos(q(3))+125/2+47/10*cos(q(2))+139/2*sin(q(2))-yc)*(-
70.1646*sin(q(8))*cos(q(9))-125-47/10*cos(q(8))-139/2*sin(q(8))-
70.1646*sin(q(2))*cos(q(3))-47/10*cos(q(2))-139/2*sin(q(2)))-(70.1646*sin(q(1)-
1.5708)*cos(q(2))*cos(q(3))+70.1646*cos(q(1)-1.5708)*sin(q(3))-47/10*sin(q(1)-
1.5708)*sin(q(2))+139/2*sin(q(1)-1.5708)*cos(q(2))+9*cos(q(1)-1.5708)-zc)*(-
70.1646*sin(q(7)+1.5708)*cos(q(8))*cos(q(9))-
70.1646*cos(q(7)+1.5708)*sin(q(9))+47/10*sin(q(7)+1.5708)*sin(q(8))-
139/2*sin(q(7)+1.5708)*cos(q(8))-9*cos(q(7)+1.5708)-70.1646*sin(q(1)-
1.5708)*cos(q(2))*cos(q(3))-70.1646*cos(q(1)-1.5708)*sin(q(3))+47/10*sin(q(1)-
1.5708)*sin(q(2))-139/2*sin(q(1)-1.5708)*cos(q(2))-9*cos(q(1)-
1.5708)))/((70.1646*cos(q(7)+1.5708)*cos(q(8))*cos(q(9))-
70.1646*sin(q(7)+1.5708)*sin(q(9))-
47/10*cos(q(7)+1.5708)*sin(q(8))+139/2*cos(q(7)+1.5708)*cos(q(8))-130-
9*sin(q(7)+1.5708)-70.1646*cos(q(1)-1.5708)*cos(q(2))*cos(q(3))+70.1646*sin(q(1)-
1.5708)*sin(q(3))+47/10*cos(q(1)-1.5708)*sin(q(2))-139/2*cos(q(1)-
1.5708)*cos(q(2))+9*sin(q(1)-1.5708))^2+(-70.1646*sin(q(8))*cos(q(9))-125-
47/10*cos(q(8))-139/2*sin(q(8))-70.1646*sin(q(2))*cos(q(3))-47/10*cos(q(2))-
139/2*sin(q(2)))^2+(-70.1646*sin(q(7)+1.5708)*cos(q(8))*cos(q(9))-
70.1646*cos(q(7)+1.5708)*sin(q(9))+47/10*sin(q(7)+1.5708)*sin(q(8))-
139/2*sin(q(7)+1.5708)*cos(q(8))-9*cos(q(7)+1.5708)-70.1646*sin(q(1)-
1.5708)*cos(q(2))*cos(q(3))-70.1646*cos(q(1)-1.5708)*sin(q(3))+47/10*sin(q(1)-
1.5708)*sin(q(2))-139/2*sin(q(1)-1.5708)*cos(q(2))-9*cos(q(1)-1.5708))^2)*(-
70.1646*sin(q(7)+1.5708)*cos(q(8))*cos(q(9))-
70.1646*cos(q(7)+1.5708)*sin(q(9))+47/10*sin(q(7)+1.5708)*sin(q(8))-
139/2*sin(q(7)+1.5708)*cos(q(8))-9*cos(q(7)+1.5708)-70.1646*sin(q(1)-
1.5708)*cos(q(2))*cos(q(3))-70.1646*cos(q(1)-1.5708)*sin(q(3))+47/10*sin(q(1)-
1.5708)*sin(q(2))-139/2*sin(q(1)-1.5708)*cos(q(2))-9*cos(q(1)-1.5708)))*m*g(2);
    end
    if leg==1 || leg==4
        F(1)=-F(1);
    end
    %TorqueY
    if leg==1 || leg==3
        F(2)=(zc-.0702*sin(q(4)-1.5708)*cos(q(5))*cos(q(6))-.0702*cos(q(4)-
1.5708)*sin(q(6))+.0047*sin(q(4)-1.5708)*sin(q(5))-.0695*sin(q(4)-1.5708)*cos(q(5))-
9/1000*cos(q(4)-1.5708)-1/1000*(-(70.1646*cos(q(4)-1.5708)*cos(q(5))*cos(q(6))-
70.1646*sin(q(4)-1.5708)*sin(q(6))-47/10*cos(q(4)-1.5708)*sin(q(5))+139/2*cos(q(4)-
1.5708)*cos(q(5))+65-9*sin(q(4)-1.5708)-
xc)*(70.1646*cos(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*sin(q(10)+1.5708)*sin(q(12))-
47/10*cos(q(10)+1.5708)*sin(q(11))+139/2*cos(q(10)+1.5708)*cos(q(11))-130-
9*sin(q(10)+1.5708)-70.1646*cos(q(4)-1.5708)*cos(q(5))*cos(q(6))+70.1646*sin(q(4)-
1.5708)*sin(q(6))+47/10*cos(q(4)-1.5708)*sin(q(5))-139/2*cos(q(4)-
1.5708)*cos(q(5))+9*sin(q(4)-1.5708))-(-70.1646*sin(q(5))*cos(q(6))-125/2-
47/10*cos(q(5))-139/2*sin(q(5))-
yc)*(70.1646*sin(q(11))*cos(q(12))+125+47/10*cos(q(11))+139/2*sin(q(11))+70.1646*sin(q
```

```
(5))*cos(q(6))+47/10*cos(q(5))+139/2*sin(q(5)))-(70.1646*sin(q(4)-
1.5708)*cos(q(5))*cos(q(6))+70.1646*cos(q(4)-1.5708)*sin(q(6))-47/10*sin(q(4)-
1.5708)*sin(q(5))+139/2*sin(q(4)-1.5708)*cos(q(5))+9*cos(q(4)-1.5708)-zc)*(-
70.1646*sin(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*cos(q(10)+1.5708)*sin(q(12))+47/10*sin(q(10)+1.5708)*sin(q(11))-
139/2*sin(q(10)+1.5708)*cos(q(11))-9*cos(q(10)+1.5708)-70.1646*sin(q(4)-
1.5708)*cos(q(5))*cos(q(6))-70.1646*cos(q(4)-1.5708)*sin(q(6))+47/10*sin(q(4)-
1.5708)*sin(q(5))-139/2*sin(q(4)-1.5708)*cos(q(5))-9*cos(q(4)-
1.5708)))/((70.1646*cos(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*sin(q(10)+1.5708)*sin(q(12))-
47/10*cos(q(10)+1.5708)*sin(q(11))+139/2*cos(q(10)+1.5708)*cos(q(11))-130-
9*sin(q(10)+1.5708)-70.1646*cos(q(4)-1.5708)*cos(q(5))*cos(q(6))+70.1646*sin(q(4)-
1.5708)*sin(q(6))+47/10*cos(q(4)-1.5708)*sin(q(5))-139/2*cos(q(4)-
1.5708)*cos(q(5))+9*sin(q(4)-
1.5708))^2+(70.1646*sin(q(11))*cos(q(12))+125+47/10*cos(q(11))+139/2*sin(q(11))+70.164
6*sin(q(5))*cos(q(6))+47/10*cos(q(5))+139/2*sin(q(5)))^2+(-
70.1646*sin(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*cos(q(10)+1.5708)*sin(q(12))+47/10*sin(q(10)+1.5708)*sin(q(11))-
139/2*sin(q(10)+1.5708)*cos(q(11))-9*cos(q(10)+1.5708)-70.1646*sin(q(4)-
1.5708)*cos(q(5))*cos(q(6))-70.1646*cos(q(4)-1.5708)*sin(q(6))+47/10*sin(q(4)-
1.5708)*sin(q(5))-139/2*sin(q(4)-1.5708)*cos(q(5))-9*cos(q(4)-1.5708))^2)*(-
70.1646*sin(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*cos(q(10)+1.5708)*sin(q(12))+47/10*sin(q(10)+1.5708)*sin(q(11))-
139/2*sin(q(10)+1.5708)*cos(q(11))-9*cos(q(10)+1.5708)-70.1646*sin(q(4)-
1.5708)*cos(q(5))*cos(q(6))-70.1646*cos(q(4)-1.5708)*sin(q(6))+47/10*sin(q(4)-
1.5708)*sin(q(5))-139/2*sin(q(4)-1.5708)*cos(q(5))-9*cos(q(4)-1.5708)))*m*g(1)+(-
xc+.0702*cos(q(4)-1.5708)*cos(q(5))*cos(q(6))-.0702*sin(q(4)-1.5708)*sin(q(6))-
.0047*cos(q(4)-1.5708)*sin(q(5))+.0695*cos(q(4)-1.5708)*cos(q(5))+13/200-
9/1000*sin(q(4)-1.5708)+1/1000*(-(70.1646*cos(q(4)-1.5708)*cos(q(5))*cos(q(6))-
70.1646*sin(q(4)-1.5708)*sin(q(6))-47/10*cos(q(4)-1.5708)*sin(q(5))+139/2*cos(q(4)-
1.5708)*cos(q(5))+65-9*sin(q(4)-1.5708)-
xc)*(70.1646*cos(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*sin(q(10)+1.5708)*sin(q(12))-
47/10*cos(q(10)+1.5708)*sin(q(11))+139/2*cos(q(10)+1.5708)*cos(q(11))-130-
9*sin(q(10)+1.5708)-70.1646*cos(q(4)-1.5708)*cos(q(5))*cos(q(6))+70.1646*sin(q(4)-
1.5708)*sin(q(6))+47/10*cos(q(4)-1.5708)*sin(q(5))-139/2*cos(q(4)-
1.5708)*cos(q(5))+9*sin(q(4)-1.5708))-(-70.1646*sin(q(5))*cos(q(6))-125/2-
47/10*cos(q(5))-139/2*sin(q(5))-
yc)*(70.1646*sin(q(11))*cos(q(12))+125+47/10*cos(q(11))+139/2*sin(q(11))+70.1646*sin(q
(5))*cos(q(6))+47/10*cos(q(5))+139/2*sin(q(5)))-(70.1646*sin(q(4)-
1.5708)*cos(q(5))*cos(q(6))+70.1646*cos(q(4)-1.5708)*sin(q(6))-47/10*sin(q(4)-
1.5708)*sin(q(5))+139/2*sin(q(4)-1.5708)*cos(q(5))+9*cos(q(4)-1.5708)-zc)*(-
70.1646*sin(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*cos(q(10)+1.5708)*sin(q(12))+47/10*sin(q(10)+1.5708)*sin(q(11))-
139/2*sin(q(10)+1.5708)*cos(q(11))-9*cos(q(10)+1.5708)-70.1646*sin(q(4)-
1.5708)*cos(q(5))*cos(q(6))-70.1646*cos(q(4)-1.5708)*sin(q(6))+47/10*sin(q(4)-
1.5708)*sin(q(5))-139/2*sin(q(4)-1.5708)*cos(q(5))-9*cos(q(4)-
1.5708)))/((70.1646*cos(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*sin(q(10)+1.5708)*sin(q(12))-
47/10*cos(q(10)+1.5708)*sin(q(11))+139/2*cos(q(10)+1.5708)*cos(q(11))-130-
9*sin(q(10)+1.5708)-70.1646*cos(q(4)-1.5708)*cos(q(5))*cos(q(6))+70.1646*sin(q(4)-
1.5708)*sin(q(6))+47/10*cos(q(4)-1.5708)*sin(q(5))-139/2*cos(q(4)-
1.5708)*cos(q(5))+9*sin(q(4)-
1.5708))^2+(70.1646*sin(q(11))*cos(q(12))+125+47/10*cos(q(11))+139/2*sin(q(11))+70.164
6*sin(q(5))*cos(q(6))+47/10*cos(q(5))+139/2*sin(q(5)))^2+(-
70.1646*sin(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*cos(q(10)+1.5708)*sin(q(12))+47/10*sin(q(10)+1.5708)*sin(q(11))-
139/2*sin(q(10)+1.5708)*cos(q(11))-9*cos(q(10)+1.5708)-70.1646*sin(q(4)-
1.5708)*cos(q(5))*cos(q(6))-70.1646*cos(q(4)-1.5708)*sin(q(6))+47/10*sin(q(4)-
1.5708)*sin(q(5))-139/2*sin(q(4)-1.5708)*cos(q(5))-9*cos(q(4)-
1.5708))^2)*(70.1646*cos(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*sin(q(10)+1.5708)*sin(q(12))-
47/10*cos(q(10)+1.5708)*sin(q(11))+139/2*cos(q(10)+1.5708)*cos(q(11))-130-
9*sin(q(10)+1.5708)-70.1646*cos(q(4)-1.5708)*cos(q(5))*cos(q(6))+70.1646*sin(q(4)-
1.5708)*sin(q(6))+47/10*cos(q(4)-1.5708)*sin(q(5))-139/2*cos(q(4)-
1.5708)*cos(q(5))+9*sin(q(4)-1.5708)))*m*g(3);
    elseif leg==2 || leg==4
        F(2)=(zc-.0702*sin(q(1)-1.5708)*cos(q(2))*cos(q(3))-.0702*cos(q(1)-
1.5708)*sin(q(3))+.0047*sin(q(1)-1.5708)*sin(q(2))-.0695*sin(q(1)-1.5708)*cos(q(2))-
9/1000*cos(q(1)-1.5708)-1/1000*(-(70.1646*cos(q(1)-1.5708)*cos(q(2))*cos(q(3))-
70.1646*sin(q(1)-1.5708)*sin(q(3))-47/10*cos(q(1)-1.5708)*sin(q(2))+139/2*cos(q(1)-
1.5708)*cos(q(2))+65-9*sin(q(1)-1.5708)-
xc)*(70.1646*cos(q(7)+1.5708)*cos(q(8))*cos(q(9))-70.1646*sin(q(7)+1.5708)*sin(q(9))-
47/10*cos(q(7)+1.5708)*sin(q(8))+139/2*cos(q(7)+1.5708)*cos(q(8))-130-
9*sin(q(7)+1.5708)-70.1646*cos(q(1)-1.5708)*cos(q(2))*cos(q(3))+70.1646*sin(q(1)-
1.5708)*sin(q(3))+47/10*cos(q(1)-1.5708)*sin(q(2))-139/2*cos(q(1)-
1.5708)*cos(q(2))+9*sin(q(1)-1.5708))-
```

```
(70.1646*sin(q(2))*cos(q(3))+125/2+47/10*cos(q(2))+139/2*sin(q(2))-yc)*(-
70.1646*sin(q(8))*cos(q(9))-125-47/10*cos(q(8))-139/2*sin(q(8))-
70.1646*sin(q(2))*cos(q(3))-47/10*cos(q(2))-139/2*sin(q(2)))-(70.1646*sin(q(1)-
1.5708)*cos(q(2))*cos(q(3))+70.1646*cos(q(1)-1.5708)*sin(q(3))-47/10*sin(q(1)-
1.5708)*sin(q(2))+139/2*sin(q(1)-1.5708)*cos(q(2))+9*cos(q(1)-1.5708)-zc)*(-
70.1646*sin(q(7)+1.5708)*cos(q(8))*cos(q(9))-
70.1646*cos(q(7)+1.5708)*sin(q(9))+47/10*sin(q(7)+1.5708)*sin(q(8))-
139/2*sin(q(7)+1.5708)*cos(q(8))-9*cos(q(7)+1.5708)-70.1646*sin(q(1)-
1.5708)*cos(q(2))*cos(q(3))-70.1646*cos(q(1)-1.5708)*sin(q(3))+47/10*sin(q(1)-
1.5708)*sin(q(2))-139/2*sin(q(1)-1.5708)*cos(q(2))-9*cos(q(1)-
1.5708)))/((70.1646*cos(q(7)+1.5708)*cos(q(8))*cos(q(9))-
70.1646*sin(q(7)+1.5708)*sin(q(9))-
47/10*cos(q(7)+1.5708)*sin(q(8))+139/2*cos(q(7)+1.5708)*cos(q(8))-130-
9*sin(q(7)+1.5708)-70.1646*cos(q(1)-1.5708)*cos(q(2))*cos(q(3))+70.1646*sin(q(1)-
1.5708)*sin(q(3))+47/10*cos(q(1)-1.5708)*sin(q(2))-139/2*cos(q(1)-
1.5708)*cos(q(2))+9*sin(q(1)-1.5708))^2+(-70.1646*sin(q(8))*cos(q(9))-125-
47/10*cos(q(8))-139/2*sin(q(8))-70.1646*sin(q(2))*cos(q(3))-47/10*cos(q(2))-
139/2*sin(q(2)))^2+(-70.1646*sin(q(7)+1.5708)*cos(q(8))*cos(q(9))-
70.1646*cos(q(7)+1.5708)*sin(q(9))+47/10*sin(q(7)+1.5708)*sin(q(8))-
139/2*sin(q(7)+1.5708)*cos(q(8))-9*cos(q(7)+1.5708)-70.1646*sin(q(1)-
1.5708)*cos(q(2))*cos(q(3))-70.1646*cos(q(1)-1.5708)*sin(q(3))+47/10*sin(q(1)-
1.5708)*sin(q(2))-139/2*sin(q(1)-1.5708)*cos(q(2))-9*cos(q(1)-1.5708))^2)*(-
70.1646*sin(q(7)+1.5708)*cos(q(8))*cos(q(9))-
70.1646*cos(q(7)+1.5708)*sin(q(9))+47/10*sin(q(7)+1.5708)*sin(q(8))-
139/2*sin(q(7)+1.5708)*cos(q(8))-9*cos(q(7)+1.5708)-70.1646*sin(q(1)-
1.5708)*cos(q(2))*cos(q(3))-70.1646*cos(q(1)-1.5708)*sin(q(3))+47/10*sin(q(1)-
1.5708)*sin(q(2))-139/2*sin(q(1)-1.5708)*cos(q(2))-9*cos(q(1)-1.5708)))*m*g(1)+(-
xc+.0702*cos(q(1)-1.5708)*cos(q(2))*cos(q(3))-.0702*sin(q(1)-1.5708)*sin(q(3))-
.0047*cos(q(1)-1.5708)*sin(q(2))+.0695*cos(q(1)-1.5708)*cos(q(2))+13/200-
9/1000*sin(q(1)-1.5708)+1/1000*(-(70.1646*cos(q(1)-1.5708)*cos(q(2))*cos(q(3))-
70.1646*sin(q(1)-1.5708)*sin(q(3))-47/10*cos(q(1)-1.5708)*sin(q(2))+139/2*cos(q(1)-
1.5708)*cos(q(2))+65-9*sin(q(1)-1.5708)-
xc)*(70.1646*cos(q(7)+1.5708)*cos(q(8))*cos(q(9))-70.1646*sin(q(7)+1.5708)*sin(q(9))-
47/10*cos(q(7)+1.5708)*sin(q(8))+139/2*cos(q(7)+1.5708)*cos(q(8))-130-
9*sin(q(7)+1.5708)-70.1646*cos(q(1)-1.5708)*cos(q(2))*cos(q(3))+70.1646*sin(q(1)-
1.5708)*sin(q(3))+47/10*cos(q(1)-1.5708)*sin(q(2))-139/2*cos(q(1)-
1.5708)*cos(q(2))+9*sin(q(1)-1.5708))-
(70.1646*sin(q(2))*cos(q(3))+125/2+47/10*cos(q(2))+139/2*sin(q(2))-yc)*(-
70.1646*sin(q(8))*cos(q(9))-125-47/10*cos(q(8))-139/2*sin(q(8))-
70.1646*sin(q(2))*cos(q(3))-47/10*cos(q(2))-139/2*sin(q(2)))-(70.1646*sin(q(1)-
1.5708)*cos(q(2))*cos(q(3))+70.1646*cos(q(1)-1.5708)*sin(q(3))-47/10*sin(q(1)-
1.5708)*sin(q(2))+139/2*sin(q(1)-1.5708)*cos(q(2))+9*cos(q(1)-1.5708)-zc)*(-
70.1646*sin(q(7)+1.5708)*cos(q(8))*cos(q(9))-
70.1646*cos(q(7)+1.5708)*sin(q(9))+47/10*sin(q(7)+1.5708)*sin(q(8))-
139/2*sin(q(7)+1.5708)*cos(q(8))-9*cos(q(7)+1.5708)-70.1646*sin(q(1)-
1.5708)*cos(q(2))*cos(q(3))-70.1646*cos(q(1)-1.5708)*sin(q(3))+47/10*sin(q(1)-
1.5708)*sin(q(2))-139/2*sin(q(1)-1.5708)*cos(q(2))-9*cos(q(1)-
1.5708)))/((70.1646*cos(q(7)+1.5708)*cos(q(8))*cos(q(9))-
70.1646*sin(q(7)+1.5708)*sin(q(9))-
47/10*cos(q(7)+1.5708)*sin(q(8))+139/2*cos(q(7)+1.5708)*cos(q(8))-130-
9*sin(q(7)+1.5708)-70.1646*cos(q(1)-1.5708)*cos(q(2))*cos(q(3))+70.1646*sin(q(1)-
1.5708)*sin(q(3))+47/10*cos(q(1)-1.5708)*sin(q(2))-139/2*cos(q(1)-
1.5708)*cos(q(2))+9*sin(q(1)-1.5708))^2+(-70.1646*sin(q(8))*cos(q(9))-125-
47/10*cos(q(8))-139/2*sin(q(8))-70.1646*sin(q(2))*cos(q(3))-47/10*cos(q(2))-
139/2*sin(q(2)))^2+(-70.1646*sin(q(7)+1.5708)*cos(q(8))*cos(q(9))-
70.1646*cos(q(7)+1.5708)*sin(q(9))+47/10*sin(q(7)+1.5708)*sin(q(8))-
139/2*sin(q(7)+1.5708)*cos(q(8))-9*cos(q(7)+1.5708)-70.1646*sin(q(1)-
1.5708)*cos(q(2))*cos(q(3))-70.1646*cos(q(1)-1.5708)*sin(q(3))+47/10*sin(q(1)-
1.5708)*sin(q(2))-139/2*sin(q(1)-1.5708)*cos(q(2))-9*cos(q(1)-
1.5708))^2)*(70.1646*cos(q(7)+1.5708)*cos(q(8))*cos(q(9))-
70.1646*sin(q(7)+1.5708)*sin(q(9))-
47/10*cos(q(7)+1.5708)*sin(q(8))+139/2*cos(q(7)+1.5708)*cos(q(8))-130-
9*sin(q(7)+1.5708)-70.1646*cos(q(1)-1.5708)*cos(q(2))*cos(q(3))+70.1646*sin(q(1)-
1.5708)*sin(q(3))+47/10*cos(q(1)-1.5708)*sin(q(2))-139/2*cos(q(1)-
1.5708)*cos(q(2))+9*sin(q(1)-1.5708)))*m*g(3);
    end
    if leg==3 || leg==4
        F(2)=-F(2);
    end
end
function xPos=xLegPos(q,x_coor)
    for i=1:4
        leg=i;
        if leg==2
            xPos(i,1)=70.1646*cos(q(4)-1.5708)*cos(q(5))*cos(q(6))-70.1646*sin(q(4)-
1.5708)*sin(q(6))-4.7*cos(q(4)-1.5708)*sin(q(5))+139/2*cos(q(4)-1.5708)*cos(q(5))+65-
9*sin(q(4)-1.5708)-x_coor(2);
```

```
        elseif leg==4
            xPos(i,1)=70.1646*cos(q(10)+1.5708)*cos(q(11))*cos(q(12))-
70.1646*sin(q(10)+1.5708)*sin(q(12))-
4.7*cos(q(10)+1.5708)*sin(q(11))+69.5*cos(q(10)+1.5708)*cos(q(11))-65-
9*sin(q(10)+1.5708)-x_coor(4);
        elseif leg==1
            xPos(i,1)=70.1646*cos(q(1)-1.5708)*cos(q(2))*cos(q(3))-70.1646*sin(q(1)-
1.5708)*sin(q(3))-4.7*cos(q(1)-1.5708)*sin(q(2))+69.5*cos(q(1)-1.5708)*cos(q(2))+65-
9*sin(q(1)-1.5708)-x_coor(1);
        elseif leg==3
            xPos(i,1)=70.1646*cos(q(7)+1.5708)*cos(q(8))*cos(q(9))-
70.1646*sin(q(7)+1.5708)*sin(q(9))-
4.7*cos(q(7)+1.5708)*sin(q(8))+69.5*cos(q(7)+1.5708)*cos(q(8))-65-9*sin(q(7)+1.5708)-
x_coor(3);
        end
    end

end
function [y1,y2,y3,y4]=yLegPos(q)
        y2=-70.1646*sin(q(5))*cos(q(6))-125/2-4.7*cos(q(5))-139/2*sin(q(5));
        y4=70.1646*sin(q(11))*cos(q(12))+62.5+4.7*cos(q(11))+69.5*sin(q(11));
        y1=70.1646*sin(q(2))*cos(q(3))+62.5+4.7*cos(q(2))+69.5*sin(q(2));
        y3=-70.1646*sin(q(8))*cos(q(9))-62.5-4.7*cos(q(8))-69.5*sin(q(8));
end

function [yPosP,y1,y2,y3,y4]=ycons(q,qP)
     [y1,y2,y3,y4]=yLegPos(q);
     [yP1,yP2,yP3,yP4]=yLegPos(qP);
     d1=3;
     yPosP(1)=(y1-yP1)^2-d1;
     yPosP(2)=(y2-yP2)^2-d1;
     yPosP(3)=(y3-yP3)^2-d1;
     yPosP(4)=(y4-yP4)^2-d1;
end
```

send_to_aibo.m

```
function send_to_aibo(q,aibo)
    q=q*180/pi;
    Str_legs=str2mat(  'legLF1.val', 'legLF2.val', 'legLF3.val', 'legRF1.val'
,'legRF2.val', 'legRF3.val', 'legRH1.val', 'legRH2.val','legRH3.val','legLH1.val'
,'legLH2.val' ,'legLH3.val','neck.val','headPan.val','headTilt.val');
    cmd = [];
    % Compose the command string
    for i=1:15
        cmd = [cmd ' & ' Str_legs(i,:) '=' num2str(q(i))];
    end
    cmd = cmd(4:size(cmd, 2));
    cmd=[cmd ' smooth:50ms'];
    urbiSend (aibo, [cmd ';']);
  end
```