Διεθνές Πανεπιστήμιο της Ελλάδος
Τμήμα Μηχανικών Πληροφορικής, Υπολογιστών και Τηλεπικοινωνιών

Ανάπτυξη Συστήματος Off-Line Ελέγχου και Προγραμματισμού Ρομποτικού Βραχίονα 6DOF σε Περιβάλλον MATLAB

Εργασία που υποβλήθηκε στο
Πρόγραμμα Μεταπτυχιακών Σπουδών στη Ρομποτική,
του Διεθνούς Πανεπιστημίου της Ελλάδος,
για τη μερική εκπλήρωση υποχρεώσεων
για το Δίπλωμα Ειδίκευσης στη Ρομποτική

Εκπονήτρια: **Δήμητρα Ζώτου**

Επιβλέπων Καθηγητής: Δημήτριος Σαγρής

Σέρρες, 2021

Ανάπτυξη Συστήματος Off-Line Ελέγχου και Προγραμματισμού Ρομποτικού Βραχίονα 6DOF σε Περιβάλλον MATLAB

Εργασία που υποβλήθηκε στο Πρόγραμμα Μεταπτυχιακών Σπουδών στη Ρομποτική, του Διεθνούς Πανεπιστημίου της Ελλάδος, για τη μερική εκπλήρωση υποχρεώσεων για το Δίπλωμα Ειδίκευσης στη Ρομποτική

Εκπονήτρια: Δήμητρα Ζώτου

Επιβλέπων Καθηγητής: Δημήτριος Σαγρής

Σέρρες, 2021

Υπεύθυνη Δήλωση Φοιτητών:

Η κάτωθι υπογεγραμμένη φοιτήτρια, έχοντας επίγνωση των συνεπειών του Νόμου περί λογοκλοπής, δηλώνει υπεύθυνα ότι είναι συγγραφέας αυτής της Μεταπτυχιακής Εργασίας, αναλαμβάνοντας την ευθύνη επί ολοκλήρου του κειμένου εξ ίσου, έχοντας δε αναφέρει στην Βιβλιογραφία όλες τις πηγές τις οποίες χρησιμοποίησε. Δηλώνει επίσης ότι, οποιοδήποτε στοιχείο ή κείμενο το οποίο έχει ενσωματώσει στην εργασία του προερχόμενο από βιβλία, άλλες εργασίες ή το διαδίκτυο, γραμμένο επακριβώς ή παραφρασμένο, το έχει πλήρως αναγνωρίσει ως πνευματικό έργο άλλου συγγραφέα και έχει αναφέρει ανελλιπώς το όνομά του και την πηγή προέλευσης.

Η Φοιτήτρια:

Δήμητρα Ζώτου

ΕΥΧΑΡΙΣΤΙΕΣ…

Με την ολοκλήρωση της παρούσας διπλωματικής εργασίας, θα ήθελα να ευχαριστήσω όλους όσους συνέβαλαν στην ολοκλήρωσή της.

Ευχαριστώ ιδιαιτέρως τον Δρ. Σαγρή Δημήτριο, για την καθοδήγηση, τις συμβουλές και την πολύτιμη βοήθειά του κατά τη διάρκεια της διπλωματικής μου εργασίας.

Επίσης, ευχαριστώ θερμά την οικογένεια και τους φίλους μου για τη συνεχόμενη στήριξη στη μεταπτυχιακή μου εκπαίδευση.

Σέρρες, Ιούνιος 2021

ΔΗΜΗΤΡΑ ΖΩΤΟΥ

Περίληψη

Σκοπός της παρούσας διπλωματικής εργασίας είναι η ανάπτυξη ενός συστήματος Off-Line ελέγχου και προγραμματισμού ενός ρομποτικού βραχίονα έξι (6) βαθμών ελευθερίας (Degrees of Freedom – DOF) στο περιβάλλον MATLAB.

Δημιουργήθηκε ένα λογισμικό με σκοπό την επίλυση του ευθέως και του αντιστρόφου κινηματικού προβλήματος ενός ρομποτικού βραχίονα. Το λογισμικό είναι αρκετά απλό στη χρήστη του, αφού ο χειριστής του δε χρειάζεται να γράψει κώδικα. Αρκεί να συμπληρωθούν τα απαραίτητα δεδομένα, τα οποία έχει ανάγκη ο αλγόριθμος για να επιλύσει το εκάστοτε μαθηματικό πρόβλημα. Με βάση αυτό, γίνεται ο έλεγχος τον αρθρώσεων για την εύρεση της θέσης του βραχίονα ή το αντίστροφο. Παράλληλα με τη λύση του προβλήματος, θα εμφανίζεται η τρισδιάστατη μορφή του βραχίονα, σε μορφή προσομοίωσης. Έτσι, για το κάθε αποτέλεσμα, ο βραχίονας θα παίρνει τη θέση που θα έπαιρνε ο πραγματικός.

Επίσης, υπάρχει η επιλογή της αποθήκευσης των θέσεων του βραχίονα, ώστε να μπορεί να γίνει η εξαγωγή του NC κώδικα για τον πραγματικό βραχίονα. Ως αποτέλεσμα, μπορεί να γίνει offline χειρισμός και έλεγχος ενός ρομποτικού βραχίονα.

Η μελέτη της εργασίας έγινε στο ρομποτικό βραχίονα RS005L της εταιρείας Kawasaki, ο οποίος βρίσκεται τόσο στο Τμήμα Μηχανικών Πληροφορικής, Υπολογιστών και Τηλεπικοινωνιών, όσο και στο Τμήμα Μηχανολόγων Μηχανικών του Διεθνούς Πανεπιστημίου Ελλάδος στις Σέρρες. Υπάρχει, όμως, η δυνατότητα να χρησιμοποιηθεί το λογισμικό για οποιονδήποτε άλλο σειριακό ρομποτικό βραχίονα ίδιου τύπου, αλλάζοντας τις τιμές των παραμέτρων Denavit-Hartenberg σε φόρμα ελέγχου των παραμέτρων αυτών που διαθέτει η εφαρμογή που αναπτύχθηκε.

English Summary

Thesis Title: Development of an Offline Control System and Programming of a 6DOF Robotic Arm in the MATLAB Environment

Abstract

The purpose of this thesis is the development of an Off-line control system and programming a robotic arm of six (6) degrees of freedom (DOF) in the MATLAB environment.

Software was created to solve the forward and inverse kinematic problem of a robotic arm. The software is quite simple to the user since the operator does not need to write code. The only thing the user needs to do is to fill in the necessary data which the algorithm requires to solve the respective mathematical problem. Based on this, the joints are checked to find the position of the arm or the inverse. Among with the solution of the problem, the three-dimensional form of the robotic arm will appear, in the form of a simulation. Thus, for each result, the arm will take the same position that the real one would take.

There is also the option to save the positions of the robotic arm so that the NC code for the actual arm can be extracted. As a result, a robotic arm can be operated and controlled offline.

The study of this thesis was done on the robotic arm RS005L of the Kawasaki Company, which is located both in the Department of Informatics, Computer and Telecommunications Engineering and the Department of Mechanical Engineering of the International University of Greece in Serres. However, it is possible to use the software for any other serial robotic arm of the same type by changing the Denavit – Hartenberg parameter values into a parameter control form available to the application being developed.

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

1. Εισαγωγή

1.1. Κίνητρο

Από το παρελθόν, ο άνθρωπος έψαχνε τρόπους να διευκολύνει την καθημερινότητά του, να αποφύγει σκληρές και επίπονες διεργασίες. Με την πάροδο των χρόνων έχει καταφέρει την ανάπτυξη και την εξέλιξη ηλεκτρικών συσκευών και μηχανών, οι οποίες έχουν μπει στη ζωή του, κάνοντας επαναλαμβανόμενες και συγκεκριμένες εργασίες.

Έχοντας τα παραπάνω ως κίνητρο, ο άνθρωπος κατάφερε την ύπαρξη των ρομπότ στη βιομηχανία, αλλά και σε πολλούς άλλους τομείς της ζωής. Έτσι, επιτυγχάνεται η μείωση του απαιτούμενου χρόνου απασχόλησης του σε κουραστικές και μονότονες δραστηριότητες, οι οποίες δεν επηρεάζουν μόνο τη σωματική, αλλά και την ψυχική του υγεία. Αυτό συμβαίνει, γιατί τα ρομπότ μπορούν να χρησιμοποιηθούν σε πολλών ειδών επικίνδυνες και πολύπλοκες εργασίες. Ακόμα, μπορούν να υλοποιούν μεγάλου όγκου διεργασίες, πραγματοποιώντας τες σε μεγάλες ταχύτητες, με πολύ μεγάλο βαθμό ακρίβειας.

Στον κλάδο της βιομηχανίας, η ρομποτική έχει εξελιχθεί πολύ και πλέον μπορεί να εμπλακεί σε πάρα πολλούς τομείς. Τα ρομπότ χρησιμοποιούνται σε εφαρμογές, όπως η συναρμολόγηση, η βαφή, η συγκόλληση και η τοποθέτηση των προϊόντων, όπου χρειάζεται. Εκτός από ταχύτητα, ακρίβεια και αντοχή, έχουν το πλεονέκτημα να μπορούν να εκτελούν τις εργασίες που απαιτούνται ακόμα και σε ανθυγιεινές, για τον άνθρωπο, συνθήκες, όπως οι υψηλές θερμοκρασίες ή η ακτινοβολία.

Με έναυσμα τα παραπάνω, η συγκεκριμένη εργασία στηρίχθηκε στην ιδέα της δημιουργίας ενός λογισμικού για τη χρήση ενός βιομηχανικού ρομποτικού βραχίονα συγκεκριμένης γεωμετρίας. Αυτό το λογισμικό, θα μπορεί να χρησιμοποιηθεί από οποιονδήποτε, χωρίς να απαιτούνται γνώσεις προγραμματισμού. Η μόνη προϋπόθεση για την ομαλή και σωστή λειτουργία του λογισμικού είναι η γεωμετρία του βραχίονα, μιας και δεν επιτεύχθηκε η εφαρμογή του για όλους τους πιθανούς βραχίονες.

1.2.Αντικείμενο Εργασίας

Το θέμα της παρούσας εργασίας είναι η ανάπτυξη ενός εκτός σύνδεσης (off-line) συστήματος ελέγχου και προγραμματισμού ενός σειριακού ρομποτικού βραχίονα 6 βαθμών ελευθερίας. Με τον όρο βαθμός ελευθερίας (Degree of Freedom – DOF), στον τομέα της μηχανικής, ονομάζεται ο αριθμός των ανεξάρτητων κινήσεων που διαθέτει ένας μηχανισμός [1].

Ο off-line προγραμματισμός είναι η ανάπτυξη ενός κώδικα για ένα ρομπότ χωρίς αυτός να αναπτύσσεται επί του ίδιου του ρομπότ. Έτσι, με τη χρήση ενός ηλεκτρονικού υπολογιστή,

δημιουργείται ο νέος κώδικας και ύστερα γίνεται η εκτέλεσή του στο ρομπότ. Βασική προϋπόθεση είναι η ύπαρξη ενός λογισμικού που εκτελεί ο χειριστής για να δημιουργήσει τροχιές και τελικά κώδικα, ενώ παράλληλα απαιτείται η ύπαρξη ενός γραφικού μοντέλου, που προσομοιώνει τη γεωμετρία και τις κινήσεις του φυσικού μηχανισμού, ώστε ο χειριστής να έχει καλή εποπτεία των αποτελεσμάτων των εντολών που συντάσσει [2].

Ένα από τα πιο βασικά του πλεονεκτήματα είναι η ευκολία δοκιμών διαφορετικών προσεγγίσεων πάνω στο ίδιο πρόβλημα, κάτι που δε θα μπορούσε να γίνει με χρήση online προγραμματισμού ή τουλάχιστον θα ήταν πολύ χρονοβόρα διαδικασία που θα την καθιστούσε απαγορευτική. Ακόμα, σε περίπτωση προγραμματισμού μίας καινούργιας τροχιάς, το ρομπότ σταματάει τη διαδικασία μόνο κατά τη φόρτωση του νέου κώδικα, με αποτέλεσμα να μειώνεται σημαντικά ο χρόνος αδράνειας του ρομπότ.

Αντίθετα, ο on-line προγραμματισμός ενός ρομπότ απαιτεί τη διακοπή της λειτουργίας του ρομπότ από οποιαδήποτε διαδικασία βρίσκεται και την τοποθέτησή του σε «λειτουργία προγραμματισμού». Υπάρχουν διάφορες τεχνικές online προγραμματισμού, οι οποίες θα αναλυθούν στο επόμενο κεφάλαιο.

Σκοπός είναι η δημιουργία ενός φιλικού προς τον χρήστη λογισμικού, για χειρισμό ενός ρομποτικού βραχίονα. Σε αυτό το πρόγραμμα, αναλόγως με τις ανάγκες και το πρόβλημα του χρήστη, θα εμφανίζεται ένα γραφικό περιβάλλον, όπου θα δηλώνονται οι απαραίτητες μεταβλητές, τις οποίες χρειάζεται ο αλγόριθμος για να επιλύσει το πρόβλημα. Αμέσως μετά θα εμφανίζονται τα αποτελέσματα, μαζί με την τρισδιάστατη γεωμετρική αναπαράσταση του βραχίονα. Για όσες κινήσεις εκτελέσει ο βραχίονας, μετά από εντολή του χρήστη, θα κινείται αμέσως και η αναπαράστασή του.

Το σύστημα ελέγχου και ο προγραμματισμός έγινε πάνω στο ρομποτικό βραχίονα RS005L της εταιρείας Kawasaki, ο οποίος βρίσκεται τόσο στο Τμήμα Μηχανικών Πληροφορικής, Υπολογιστών και Τηλεπικοινωνιών, όσο και στο Τμήμα Μηχανολόγων Μηχανικών, αμφότερα στο Διεθνές Πανεπιστήμιο Ελλάδας στις Σέρρες. Παρόλα αυτά, οι αλγόριθμοι που αναπτύχθηκαν είναι σε παραμετρική μορφή, δηλαδή μπορεί ο χειριστής να αλλάξει όλες τις τιμές και να τους προσαρμόσει εκείνος για κάποιον άλλο σειριακό βραχίονα ίδιου τύπου, αλλά με διαφορετική γεωμετρία μελών. Αφού εκτελεστούν οι επιθυμητές κινήσεις, θα εξάγεται ο αντίστοιχος κώδικας για να περαστεί στον πραγματικό βραχίονα.

1.3. Απόκτηση Γνώσεων

Η αφορμή της παρούσας διπλωματικής εργασίας ήταν η κατασκευή ενός λογισμικού που μπορεί να χειριστεί όχι μόνο έναν, αλλά όλους τους ρομποτικούς σειριακούς βραχίονες έξι βαθμών ελευθερίας, παρόμοιας γεωμετρίας με αυτόν που μελετήθηκε. Η μεγαλύτερη πρόκληση

ήταν να γίνει το λογισμικό όσο πιο εύκολο γινόταν στη χρήση για οποιονδήποτε θελήσει να το εκμεταλλευθεί. Για να μπορέσει να γίνει αυτό θα έπρεπε, αρχικά, να κατασκευαστεί ένα Γραφικό Περιβάλλον Χρήστη, το οποίο θα μπορεί να δέχεται τιμές ως δεδομένα, να υλοποιεί από πίσω όλες τις απαραίτητες μαθηματικές πράξεις και να στέλνει σε αυτό το περιβάλλον τα αποτελέσματα. Παράλληλα, θα έπρεπε να κατασκευαστεί ένας ρομποτικός βραχίονας σε ένα πρόγραμμα που θα επικοινωνεί με το γραφικό περιβάλλον, το οποίο θα λάμβανε και εκείνο τα αποτελέσματά, με σκοπό να υλοποιεί την προσομοίωση.

Κατά τη διάρκεια της έρευνας, της δημιουργίας του λογισμικού και της προσομοίωσης δημιουργήθηκαν προβλήματα. Κάποια ήταν εύκολα στην αντιμετώπισή τους και η λύση δόθηκε άμεσα, ενώ κάποια άλλα όχι, με αποτέλεσμα να γίνει ένας εκ νέου σχεδιασμός.

Όλα τα παραπάνω είχαν ως επακόλουθο την ανακάλυψη νέων δεδομένων και στοιχείων για τα προγράμματα που χρησιμοποιήθηκαν, όπως και για τη λειτουργία ενός ρομποτικού βραχίονα. Έπρεπε να συγγραφούν πολλές γραμμές κώδικα, πολλές συναρτήσεις, ώστε να μπορέσει να υπάρξει αποτέλεσμα.

Έτσι, μέσα απ’ όλη αυτή τη διαδικασία, αποκτήθηκαν πολλές νέες και σημαντικές γνώσεις για το πως λειτουργεί συνολικά ένας ρομποτικός βραχίονας. Ακόμα, η χρήση του περιβάλλοντος MATLAB και η συνεχής βελτίωση του κώδικα που συγγράφηκε, οδήγησαν στη μεγαλύτερη εξοικείωση με το πρόγραμμα.

1.4. Επιθυμητό Αποτέλεσμα

Το επιθυμητό αποτέλεσμα αυτής της εργασίας είναι η δημιουργία ενός λογισμικού, με το οποίο θα μπορεί ο χρήστης να χειριστεί ένα σειριακό ρομποτικό βραχίονα έξι βαθμών ελευθερίας. Θα μπορεί να γίνει online αλλά και offline προγραμματισμός με την εξαγωγή του κώδικα NC. Επίσης, να γίνεται παράλληλα η προσομοίωση της θέσης του βραχίονα, ώστε να γνωρίζει ο χρήστης το αποτέλεσμα. Γι’ αυτό, απαιτείται ο σχεδιασμός ενός βραχίονα με δυνατότητα περιστροφής των μελών του μέσω των περιστροφικών αρθρώσεων.

2. Στάθμη Γνώσεων

Η επιστήμη που ασχολείται με το σχεδιασμό, τον προγραμματισμό και τη λειτουργία μιας αυτοματοποιημένης συσκευής, ή αλλιώς ενός ρομπότ, ονομάζεται ρομποτική. Είναι ένας σχετικά σύγχρονος τομέας και χρησιμοποιεί επιστήμες όπως αυτές της μηχανολογίας, της πληροφορικής και του αυτόματου ελέγχου [3]. Υπάρχουν πολλοί και διάφοροι ορισμοί για το τι είναι ένα ρομπότ. Σύμφωνα με το Ινστιτούτο Ρομποτικής στην Αμερική, ρομπότ ονομάζεται μια επαναπρογραμματιζόμενη μηχανή για πολλαπλές λειτουργίες, με σκοπό τη μεταφορά αντικειμένων για διάφορες εργασίες.

2.1.Ιστορική Αναδρομή

Η λέξη «ρομπότ» προέρχεται από τη σλαβική λέξη ρομπότα (robota), που έχει την έννοια της καταναγκαστικής εργασίας και χρησιμοποιήθηκε για πρώτη φορά από τον θεατρικό συγγραφέα Karel Čapek σε ένα έργο του το 1920 με όνομα *Rossum's Universal Robots (R.U.R)*. Στο έργο, τα ρομπότ εργάζονται για τους ανθρώπους, μέχρι που αποφασίζουν να αντιδράσουν, με αποτέλεσμα να εξαφανίσουν όλη την ανθρωπότητα [4].

Το 1954 σχεδιάστηκε ο πρώτος προγραμματίσιμος ρομποτικός βραχίονας από τους George Devol και Joe Engleberger. Ο πρώτος, μάλιστα, καθιέρωσε και τον όρο «βιομηχανικό ρομπότ» (industrial robot). Το 1961 ξεκίνησε η βιομηχανική εποχή, όπου ο τομέας της ρομποτικής αναπτύχθηκε ραγδαία, με αρχή τον πρώτο βιομηχανικό ρομποτικό βραχίονα με το όνομα "UNIMATE", ο οποίος χρησιμοποιήθηκε στο εργοστάσιο της General Motors [5] με αποτέλεσμα να είναι η πρώτη εταιρία που ενσωμάτωσε τα βιομηχανικά ρομπότ στο εργοστάσιό της [6].

2.2. Χαρακτηριστικά Ρομπότ

Ένα ρομπότ έχει κάποια συγκεκριμένα χαρακτηριστικά. Αρχικά, μπορεί να αισθανθεί το περιβάλλον του (αίσθηση - sensing). Αυτό γίνεται με τη χρήση αισθητήρων, όπως αισθητήρων αφής, φωτός και πίεσης. Ένα ακόμα σημαντικό χαρακτηριστικό είναι η κίνηση που έχει ένα ρομπότ, προκειμένου να μπορεί να έχει αλληλεπίδραση με το περιβάλλον του. Γι' αυτό το λόγο τοποθετούνται τροχοί, πόδια ή οτιδήποτε άλλο χρειάζεται, ώστε να μπορεί να επιτύχει το σκοπό του. Επίσης, ένα ρομπότ θα πρέπει να τροφοδοτείται με ενέργεια και, τέλος, να περιλαμβάνει τη νοημοσύνη που χρειάζεται ώστε να γνωρίζει τι πρέπει να κάνει, η οποία εξαρτάται στον προγραμματισμό του [7].

2.3. Δομή των Ρομπότ

Ένα ρομπότ αποτελείται από συνδέσμους (links) και αρθρώσεις (joints). Σύνδεσμοι είναι τα στερεά σώματα τα οποία δημιουργούν τη μορφή του ρομπότ ενώ οι αρθρώσεις είναι μηχανισμοί που βρίσκονται ανάμεσα από δύο διαδοχικούς συνδέσμους και επιτρέπουν την κίνηση μεταξύ τους. Οι πιο συχνοί τύποι αρθρώσεων είναι η περιστροφική, η πρισματική και η σφαιρική. Η περιστροφική άρθρωση έχει έναν γωνιακό βαθμό ελευθερίας, η πρισματική έναν γραμμικό, ενώ η σφαιρική έχει τρεις βαθμούς ελευθερίας [1].

Ένας ρομποτικός βραχίονας, πέρα από συνδέσμους και αρθρώσεις, περιέχει στο ένα άκρο του μία βάση και στο άλλο άκρο του ένα τελικό σημείο δράσης (end-effector). Η βάση είναι, συνήθως, στερεωμένη στο έδαφος, ενώ το τελικό στοιχείο δράσης είναι κάποιο εργαλείο, το οποίο ονομάζεται τελικό σημείο δράσης (tool center point – TCP). Το εργαλείο εξαρτάται από την εφαρμογή την οποία θα εκτελεί ο βραχίονας. Κάποια από τα πιο συνηθισμένα είναι μία αρπάγη (gripper), συγκολλητικά, ψεκαστήρες.

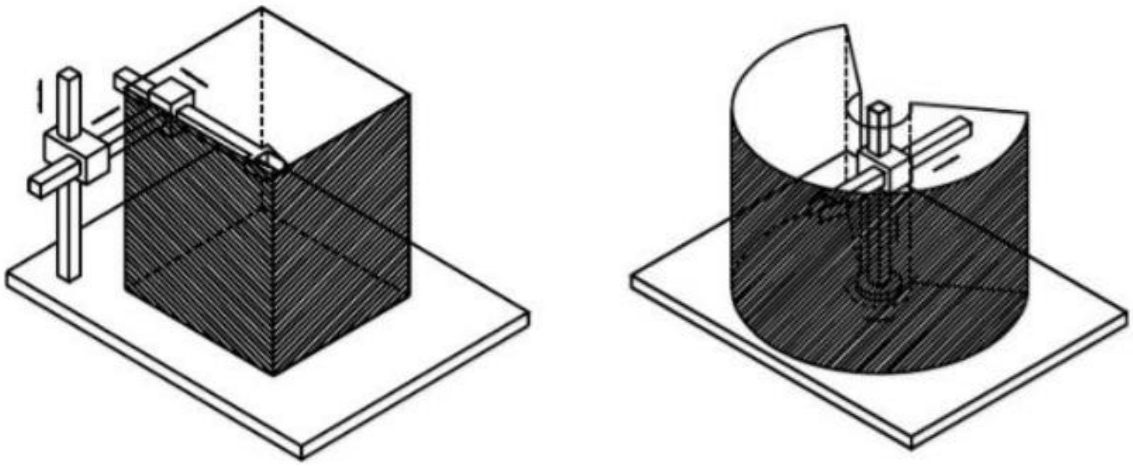
2.4. Κατηγορίες Ρομπότ

Υπάρχουν πολλοί τρόποι κατηγοριοποίησης των ρομπότ, ανάλογα με τα χαρακτηριστικά τους. Ένας από αυτούς είναι ο τύπος κίνησης των ρομπότ, αφού υπάρχουν κάποια που είναι στερεωμένα στο έδαφος, άλλα που έχουν τροχούς, πόδια, ρόδες προκαθορισμένης πλοήγησης, τα μη επανδρωμένα υποβρύχια ρομπότ και, τέλος, τα εναέρια (drones) [8].

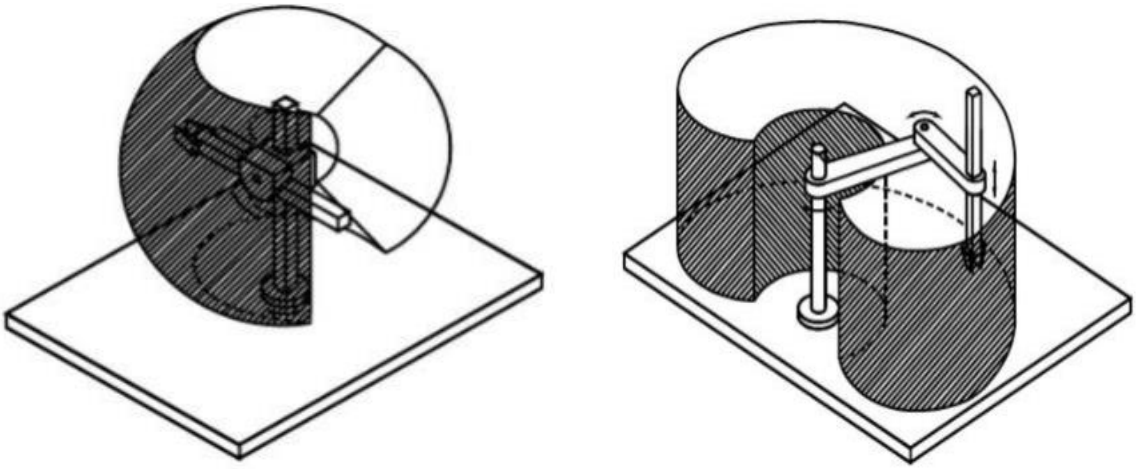
Ένας άλλος τρόπος κατάταξης των ρομπότ είναι ο λόγος χρήσης τους. Υπάρχουν ρομπότ που αξιοποιούνται μόνο σε βιομηχανίες (industrial robots) ενώ άλλα που χρησιμοποιούνται για βοήθεια στο σπίτι (domestic robots), τα οποία ονομάζονται βιομηχανικά και οικιακά ρομπότ αντίστοιχα. Παραμένοντας σε αυτόν τον τρόπο κατάταξης, υπάρχουν αυτά που χρησιμοποιούνται στην ιατρική, δηλαδή τα χειρουργικά ρομπότ, τα ρομπότ ασφάλειας, τα οποία επιτηρούν κάποιες εγκαταστάσεις, όπως επίσης και εκείνα τα οποία είναι αποκλειστικά και μόνο για ψυχαγωγία. Ακόμα, υπάρχουν και τα ρομπότ άμυνας (defense robots) όπου περιλαμβάνουν αισθητήρες υπέρυθρων και μπορούν να αντιδράν ταχύτερα από τον άνθρωπο σε περίπτωση έκτακτης ανάγκης. Τέλος, χάρη στα διαστημικά ρομπότ, η εξερεύνηση άλλων ουράνιων σωμάτων είναι πλέον γεγονός.

Εξαρτώμενα από τη μηχανική δομή τους, τα ρομπότ μπορούν να διαχωριστούν σε πέντε κατηγορίες, καρτεσιανά, κυλινδρικά, σφαιρικά, Scara και αρθρωτά. Ως καρτεσιανά χαρακτηρίζονται τα ρομπότ όπου αποτελούνται από τρεις άξονες XYZ, οι οποίοι μεταξύ τους είναι κάθετοι, δηλαδή ο ένας με τον άλλον σχηματίζουν γωνία 90 μοιρών. Όταν ένα ρομπότ έχει μία γωνιακή και δύο γραμμικές κινήσεις ονομάζεται κυλινδρικό ρομπότ, έχει μία βασική περιστροφική κίνηση και ακολουθεί η ανύψωση και η εμβέλειά του, ενώ δύο γωνιακές και μία

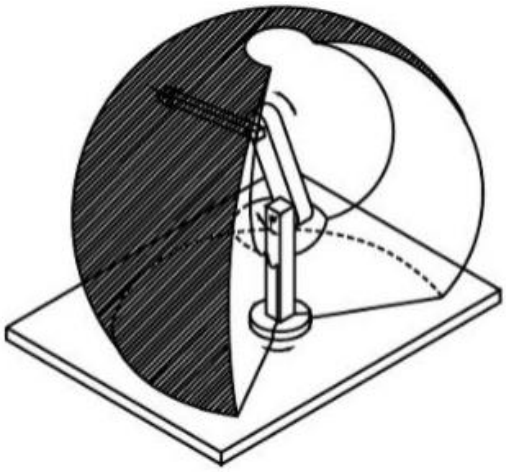
γραμμική κίνηση έχουνε τα σφαιρικά ρομπότ (εικόνα 3), τα οποία έχουν μία βασική περιστροφή, γωνιακή ανύψωση και, τέλος, την εμβέλεια του χώρου. Από την άλλη, τα SCARA (εικόνα 4) αποτελούνται από δύο παράλληλους περιστρεφόμενους συνδέσμους στο επίπεδο. Τέλος, τα αρθρωτά ρομπότ, γνωστά και ως ανθρωπόμορφα, έχουν τρεις συνεχόμενες περιστροφικές κινήσεις [9].



Εικόνα 1: Καρτεσιανό (αριστερά) και Κυλινδρικό (δεξιά) ρομπότ και οι χώροι εργασίας τους [10]



Εικόνα 2: Σφαιρικό ρομπότ (αριστερά) και Scara (δεξιά) και οι χώροι εργασίας τους [10]

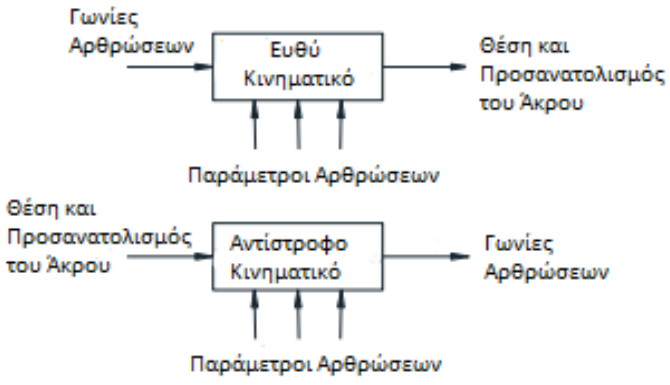


Εικόνα 3: Ανθρωπόμορφο ρομπότ και ο χώρος εργασίας του [10]

2.5. Κινηματική Ανάλυση

Για να μπορέσει ένα ρομπότ να προγραμματιστεί ή να ελεγχθεί χρειάζεται να είναι αντιληπτή η χωρική του έκταση και η αλληλεπίδρασή του με το περιβάλλον, δηλαδή θα πρέπει να γίνει πρώτα η κινηματική του ανάλυση. Ως κινηματική ανάλυση (kinematics) ορίζεται η μελέτη της κίνησης ενός μηχανισμού, χωρίς να λαμβάνονται υπόψιν εξωτερικές δυνάμεις [3].

Η κινηματική ανάλυση συχνά χωρίζεται σε δύο προβλήματα, το ευθύ και το αντίστροφο. Το ευθύ κινηματικό πρόβλημα είναι να βρεθεί η θέση του εργαλείου, γνωρίζοντας τις γωνίες των αρθρώσεων. Στο αντίστροφο κινηματικό πρόβλημα είναι γνωστή η θέση του εργαλείου και αναζητούνται οι γωνίες των αρθρώσεων [11].



Εικόνα 4: Κινηματική Ανάλυση

2.5.1 Denavit – Hartenberg

Η μέθοδος Denavit – Hartenberg είναι μία συνήθης μέθοδος που χρησιμοποιείται για την επιλογή πλαισίων αναφοράς σε εφαρμογές ρομποτικής. Το όνομα της οφείλεται στους Jacques Denavit και Richard S. Hartenberg, οι οποίοι εισηγήθηκαν για εκείνη το 1955. Πλέον, αποτελεί την πιο διαδεδομένη μέθοδο στη βιβλιογραφία της ρομποτικής [12].

2.5.2 Παράμετροι Denavit – Hartenberg Του Ρομπότ

Σε κάθε σύνδεσμο i αντιστοιχούνται δύο (2) ζεύγη τιμών. Το πρώτο ζεύγος είναι η απόσταση αi και η γωνία θi που σχηματίζονται μεταξύ του συνδέσμου i-1 και i. Το δεύτερο ζεύγος είναι το μήκος di και η γωνία στρέψης ai του συνδέσμου i. Η κάθε παράμετρος από αυτές που αναφέρθηκαν μπορεί να χαρακτηριστεί κι αλλιώς. Το μήκος αi είναι η απόσταση μεταξύ των μηκών Zi-1 και Zi ως προς τον Xi ενώ το μήκος di είναι η απόσταση από τον Xi-1 έως τον Xi ως προς τον Zi. Όσο αναφορά τις γωνίες περιστροφής, η γωνία θi είναι αυτή μεταξύ των αξόνων Xi-1 και Xi ως προς Zi και η ai είναι αυτή γύρω από τον άξονα xi [3]. Η τελευταία ονομάζεται και γωνία στρέψης (link twist) [11]. Με βάση τα παραπάνω κατασκευάζονται τέσσερις πίνακες.

- Πίνακας περιστροφής γύρω από τον άξονα OW κατά μία γωνία θ [13].

$$Rot_{z_{i-1}}(\theta_i) = \begin{bmatrix} \cos\theta_i & \sin\theta_i & 0 & 0 \\ -\sin\theta i & \cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Πίνακας μετατόπισης γύρω από τον άξονα OW κατά μία απόσταση d [13].

$$Trans_{z_{i-1}}(d_i) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Πίνακας μετατόπισης γύρω από τον άξονα OX κατά μία απόσταση α [13].

$$Trans_{x_i}(\alpha_i) = \begin{bmatrix} 1 & 0 & 0 & \alpha_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Πίνακας περιστροφής γύρω από τον άξονα OX κατά μία γωνία a [13].

$$Rot_{x_i}(a_i) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos a_i & -\sin a_i & 0 \\ 0 & \sin a_i & \cos a_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Πολλαπλασιάζοντας τους τέσσερις παραπάνω πίνακες προκύπτει ο ομογενής πίνακας μετασχηματισμού για δύο γειτονικά συστήματα συντεταγμένων, όπως φαίνεται στη εξίσωση

$$T_A^B = Trans_{x_i}(\alpha_i) * Rot_{x_i}(a_i) * Trans_{z_{i-1}}(d_i) * Rot_{z_{i-1}}(\theta_i)$$

$$= \begin{bmatrix} \cos\theta_i & -\cos a_i \sin\theta_i & \sin a_i \sin\theta_i & \alpha_i \cos\theta_i \\ \sin\theta_i & \cos a_i \cos\theta_i & -\sin a_i \cos\theta_i & \alpha_i \sin\theta_i \\ 0 & \sin a_i & \cos a_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Όταν μεταξύ των συνδέσμων i και i-1 υπάρχει άρθρωση περιστροφής, τότε μεταβάλλεται η γωνία θ_i, ενώ τα άλλα τρία μεγέθη, a_i, d_i και α_i, μένουν σταθερά. Από την άλλη, αν η άρθρωση είναι πρισματική, τότε η μόνη τιμή που μεταβάλλεται είναι η d_i.

2.6 Προγραμματισμός Ρομπότ

Ένας τρόπος χειρισμού ενός ρομποτικού βραχίονα, όπως αναφέρθηκε και παραπάνω, είναι ο προγραμματισμός εκτός σύνδεσης (offline programming – OLP). Με αυτή τη μέθοδο, το ρομπότ και οτιδήποτε άλλο το απαρτίζει, αντιστοιχίζονται και αναπαρίστανται γραφικά με τη μέθοδο της προσομοίωσης [14], [15].

Ένας ακόμα τρόπος προγραμματισμού ενός ρομποτικού βραχίονα είναι με τη βοήθεια ενός χειριστηρίου, το οποίο είναι συνδεδεμένο με το βραχίονα. Έτσι, το ρομπότ μπορεί να μετακινηθεί στις επιθυμητές θέσεις, από σημείο σε σημείο, χρησιμοποιώντας αυτό το χειριστήριο. Καθ' όλη αυτή τη διαδικασία, το ρομπότ απομνημονεύει τις θέσεις, ώστε να μπορεί να το κάνει μόνο του με μεγαλύτερη ταχύτητα [15]. Αυτή η μέθοδος ονομάζεται Διδακτικό μέσο (Teaching Pendant - TP).

Σε διαδικασίες όπως ο ψεκασμός βαφής και χρωμάτων χρησιμοποιούνται ρομποτικοί βραχίονες που είναι, συνήθως, προγραμματισμένοι με την τεχνική του Οδηγούμενου από τη μύτη (Lead by the nose). Σε αυτήν τη μέθοδο απαιτούνται δύο άνθρωποι, ο ένας από τους οποίους θα κρατάει το μηχανισμό του ρομπότ, ενώ ο άλλος θα το απενεργοποιεί. Έτσι, ο πρώτος θα μετακινήσει το ρομπότ στην επιθυμητή θέση, διανύοντας την απαιτούμενη τροχιά. Παράλληλα το ρομπότ θα καταγράφει τις κινήσεις και θα μπορεί να το αναπαράγει αργότερα μόνο του.

Τέλος, ένας ρομποτικός βραχίονας μπορεί να προγραμματιστεί με εντολές θέσεων, όπου μέσα από ένα Γραφικό Περιβάλλον Χρήστη (Graphical User Interface – GUI) θα καθορίζεται και θα επεξεργάζεται η απαιτούμενη θέση X-Y-Z [15].

3. Περιγραφή Του Βιομηχανικού Βραχίονα Kawasaki RS005L

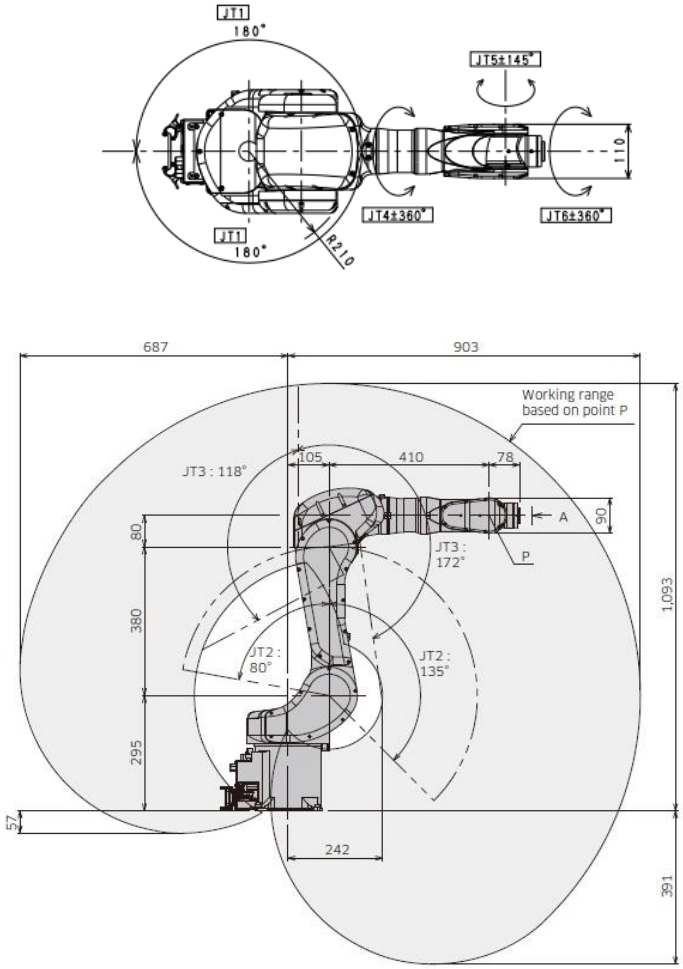
Ο βραχίονας που μελετήθηκε και χρησιμοποιήθηκε στη συγκεκριμένη διπλωματική εργασία είναι ο RS005L της εταιρείας Kawasaki, ο οποίος διατίθεται στο Μεταπτυχιακό Πρόγραμμα Σπουδών Ρομποτικής του Τμήματος Μηχανικών Πληροφορικής, Υπολογιστών και Τηλεπικοινωνιών, στο Διεθνές Πανεπιστήμιο της Ελλάδας (εικόνα 5).

Εικόνα 5: Βιομηχανικός βραχίονας RS005L

3.1 Ο Βραχίονας Kawasaki RS005L

Σύμφωνα με τον κατασκευαστή του, την εταιρεία Kawasaki, ο βραχίονας RS005L είναι ένα ρομπότ υψηλής ταχύτητας αλλά και υψηλής απόδοσης. Ανήκει στη σειρά R (R – Series Robots), όπου χρησιμοποιούνται για εργασίες μικρού έως μεσαίου φορτίου και μπορεί να χρησιμοποιηθεί και μεγάλη ποικιλία εργασιών. Το ωφέλιμο φορτίο που μπορεί να χειριστεί είναι 5 kg, η μέγιστη ακτίνα της κίνησής του φτάνει τα 903 mm και η μέγιστη γραμμική του ταχύτητα είναι 9.300 mm/sec [16].

Ο βραχίονας έχει έξι (6) βαθμούς ελευθερίας, οι οποίοι προέρχονται από τις έξι (6) περιστροφικές αρθρώσεις, από τις οποίες αποτελείται. Στο άκρο του μπορούν να εισαχθούν και να προσαρμοστούν διαφόρων ειδών συσκευές. Στην εικόνα 6 παρουσιάζεται ο χώρος εργασίας του.



Εικόνα 6: Χώρος εργασίας του RS005L [16]

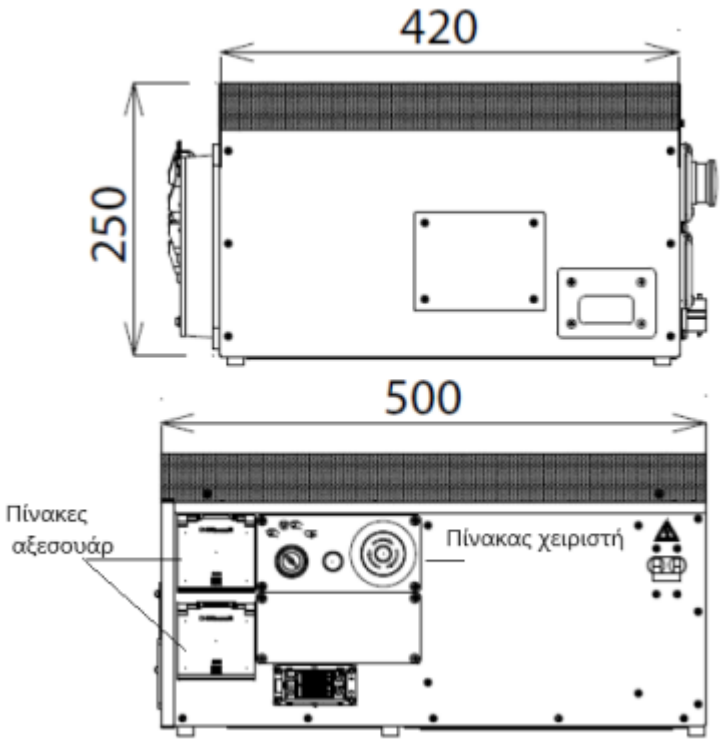
Για να μπορέσει να προγραμματιστεί, αρκεί η αποθήκευση των σημείων της επιθυμητής τροχιάς. Διαφορετικά, μπορεί να προγραμματιστεί με τη γλώσσα AS – Language της Kawasaki.

3.2 Ο Controller

Πέρα από το μηχανικό σύστημα, στα κύρια συστήματα ενός βιομηχανικού ρομποτικού βραχίονα ανήκουν το σύστημα ελέγχου (Controller) και το εξωτερικό χειριστήριο (Teach Pendant).

Εικόνα 7: Controller E71

Ο συγκεκριμένος controller είναι ένα υπερσύγχρονο προϊόν υψηλής τεχνολογίας, συμπαγής και αναβαθμίσιμος μέσω του teach pendant ή ενός υπολογιστή. Είναι φιλικό προς τον χρήστη, παρέχοντας δύο οθόνες για πληροφορίες, όπως, επίσης, η εκκίνησή του γίνεται αυτόματα μέσω της χειροκίνητης μονάδας ελέγχου. Μέσω του προγραμματισμού μπορεί να γίνει σχεδιασμός και υλοποίηση πολύπλοκων συστημάτων με τη βοήθεια των ενσωματωμένων λειτουργιών του λογισμικού του. Χάρη στον επεξεργαστή του, ο controller προσφέρει μεγάλη ταχύτητα στην υλοποίηση των προγραμμάτων, στην φόρτωση δεδομένων και στην αποθήκευσή τους. Τέλος, η συντήρηση του γίνεται αρκετά απλά και εύκολα, προσφέροντας και απομακρυσμένη διάγνωση μέσω Ethernet [16].



Εικόνα 8: Επισκόπηση του Controller [16]

	Περιγραφή
Χειριστήριο Διδασκαλίας (TP)	Παρέχει τα απαραίτητα για τη διδασκαλία του ρομπότ και την επεξεργασία δεδομένων, όπως επίσης την οθόνη για εμφάνιση και χειρισμό πλήθους δεδομένων
Πίνακας Αξεσουάρ	Παρέχονται πύλες για USB και σύνδεση με τον υπολογιστή
Πίνακας Χειριστή	Παρέχει πλήθος διακοπτών για τον χειρισμό του ρομπότ
Σύνδεση TP	Σύνδεσμος για τη σύνδεση του TP

Πίνακας 1: Περιγραφή του Controller [16]

Εικόνα 9: Teach Pendant

Στην εικόνα 10 παρουσιάζονται κάποιοι βασικοί διακόπτες για τη χειροκίνητη λειτουργία του Teach Pendant.

Εικόνα 10: Επισκόπηση του Teach Pendant

3.3 Κώδικας AS

Τα ρομπότ της εταιρείας Kawasaki χρησιμοποιούν τη AS Language ως λογισμικό, με σκοπό είτε την επικοινωνία είτε τον προγραμματισμό τους.

Με βάση τη γλώσσα προγραμματισμού AS, ένα παράδειγμα συγγραφής μιας εντολής, η οποία έχει ως γνωστές τις γωνίες των αρθρώσεων είναι:
JOINT SPEED1 ACCU2 TIMER6 TOOL1 WORK0 CLAMP1 (OFF,0,0,O) OX= WX= # ["θ1", "θ2","θ3", "θ4", "θ5", "θ6"];

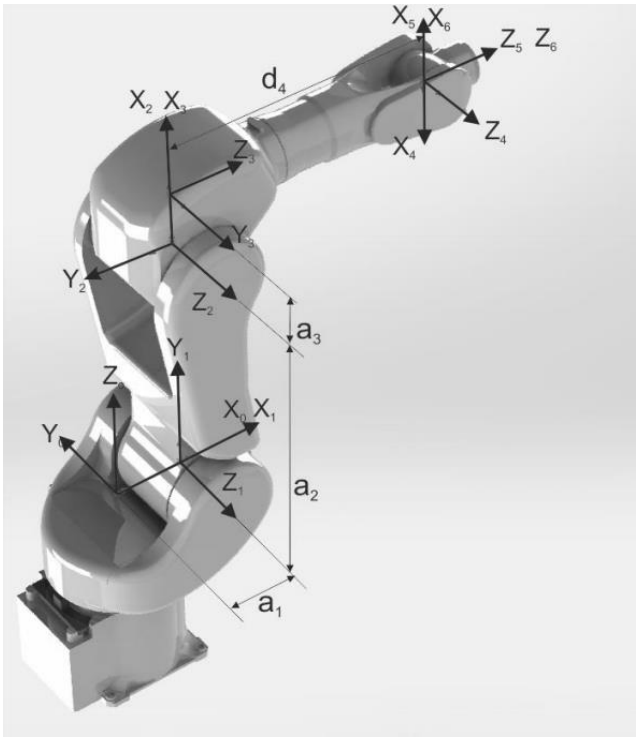
Από την άλλη, εάν είναι γνωστά η θέση και ο προσανατολισμός, συντάσσεται:

LINEAR SPEED1 ACCU2 TIMER0 TOOL1 WORK0 CLAMP1 (OFF,0,0,O) OX= WX= # ["X","Y","Z","O","A","T"]

4. Κινηματική Ανάλυση

4.1 Συστήματα Συντεταγμένων και Παράμετροι κατά Denavit - Hartenberg

Για την επιτυχή χρήση του αλγορίθμου Denavit-Hartenberg υπάρχουν ορισμένα βήματα που πρέπει να υλοποιηθούν. Αρχικά γίνεται η αρίθμηση των αρθρώσεων και έπειτα γίνεται η ανάθεση των Συστημάτων Συντεταγμένων (Σ.Σ) στους συνδέσμους.



Εικόνα 11: Συστήματα συντεταγμένων κατά D-H [17]

Στη συνέχεια, γίνεται ο ορισμός των συντεταγμένων με βάση την εικόνα 11, όπως φαίνεται στον πίνακα 2. Από τη στιγμή που ο βραχίονας αποτελείται μόνο από περιστροφικές αρθρώσεις, όλες οι μεταβλητές είναι σταθερές, πλην των γωνιών θ των αρθρώσεων.

Μέλος	**θ_i (Rz)**	**d_i (Tz)**	**α_i (Tx)**	**a_i (Rx)**
1	**θ1**	0	α_1	a_1
2	**θ2**	0	α_2	0
3	**θ3**	0	α_3	a_3
4	**θ4**	d_4	0	a_4
5	**θ5**	0	0	a_5
6	**θ6**	0	0	0

Πίνακας 2: Παράμετροι D-H

Μέλος	**θ_i (Rz)**	**d_i (Tz)**	**α_i (Tx)**	**a_i (Rx)**
1	**θ1**	0	105	90
2	**θ2**	0	380	0
3	**θ3**	0	80	90
4	**θ4**	410	0	90
5	**θ5**	0	0	90
6	**θ6**	0	0	0

Πίνακας 3: Μεταβλητές D-H για τον βραχίονα Kawasaki RS005L

4.2 Όρια Αρθρώσεων

Από τον κατασκευαστή του βραχίονα ορίζονται τα όρια των γωνιών θ. Σύμφωνα, λοιπόν με την επίσημη ιστοσελίδα της Kawasaki, τα όρια της κάθε γωνίας φαίνονται στον πίνακα 4.

Γωνία	*Min (≤)*	*Max (≥)*
JT1	*-180°*	*+180°*
JT2	*-80°*	*+135°*
JT3	*-172°*	*+118°*
JT4	*-360°*	*+360°*
JT5	*-145°*	*+145°*
JT6	*-360°*	*+360°*

Πίνακας 4: Όρια γωνιών κατά τον κατασκευαστή [16]

Ο υπολογισμός των ορίων των γωνιών JTi διαφέρει από αυτόν κατά Denavit Hartenberg. Έτσι, τα όρια των γωνιών κατά D-H γίνονται όπως παρουσιάζονται στον πίνακα 5.

Γωνία	*Min (≤)*	*Max (≥)*
θ_1	*-180°*	*+180°*
θ_2	*-45°*	*+170°*
θ_3	*-82°*	*+208°*
θ_4	*-180°*	*+540°*
θ_5	*+35°*	*+325°*
θ_6	*-360°*	*+360°*

Πίνακας 5: Όρια γωνιών κατά D-H

Αρχικά, παρουσιάζονται οι πίνακες μετασχηματισμού του κάθε μέλους σύμφωνα με τη μέθοδο Denavit – Hartenberg, χρησιμοποιώντας τις τιμές του πίνακα 2.

$$A_0^1 = \begin{bmatrix} \cos\theta_1 & 0 & \sin\theta_1 & \alpha_1\cos\theta_1 \\ \sin\theta_1 & 0 & -\cos\theta_1 & \alpha_1\sin\theta_1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_1^2 = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & \alpha_2\cos\theta_2 \\ \sin\theta_2 & \cos\theta_2 & 0 & \alpha_2\sin\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_3^4 = \begin{bmatrix} \cos\theta_4 & 0 & \sin\theta_4 & 0 \\ \sin\theta_4 & 0 & -\cos\theta_4 & 0 \\ 0 & 1 & 0 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_4^5 = \begin{bmatrix} \cos\theta_5 & 0 & \sin\theta_5 & 0 \\ \sin\theta_5 & 0 & -\cos\theta_5 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_5^6 = \begin{bmatrix} \cos\theta_6 & -\sin\theta_6 & 0 & 0 \\ \sin\theta_6 & \cos\theta_6 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Το μητρώο A_4^5 συμβολίζει τη θέση και τον προσανατολισμό του πέμπτου μέλους ως προς το τέταρτο και το μητρώο A_5^6 του έκτου ως προς το πέμπτο. Τα υπομητρώα μεταφοράς τους είναι μηδενικά, που σημαίνει πως τα συστήματα συντεταγμένων τους συμπίπτουν. Αυτό είναι και το αναμενόμενο, καθώς οι τρεις τελευταίες γωνίες (θ_4, θ_5 *και* θ_6) ορίζουν τον προσανατολισμό του άκρου του βραχίονα.

Για την αποτελεσματικότερη επίλυση του αντίστροφου κινηματικού, χρησιμοποιείται και το μητρώο A_6^7, το οποίο αναφέρεται στο εργαλείο του ρομπότ. Η θέση και ο προσανατολισμός του παραμένουν σταθερά και το μόνο που χρειάζεται είναι η απόσταση του συστήματος

συντεταγμένων του άκρου από το σύστημα συντεταγμένων του έκτου μέλους. Αυτή συμβολίζεται ως d7 και είναι ίση με 110 mm.

$$A_6^7 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & d_7 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4.3 Ορθή Κινηματική Ανάλυση

Για να βρεθεί η θέση και ο προσανατολισμός του βραχίονα αρκεί η επίλυση του ευθύ κινηματικού προβλήματος. Αυτό επιτυγχάνεται με διαδοχικούς πολλαπλασιασμούς των μητρώων, όπως φαίνεται στον παρακάτω τύπο.

$$A_0^6 = A_0^1 * A_1^2 * A_2^3 * A_3^4 * A_4^5 * A_5^6$$

Για την κινηματική επίλυση χρησιμοποιήθηκαν κάποιες συντομογραφίες για να απλοποιηθούν οι τιμές στους πίνακες. Αυτές είναι οι: $c_i = \cos\theta_i$, $c_{ij} = \cos(\theta_i + \theta_j)$, $s_i = \sin\theta_i$, $s_{ij} = \sin(\theta_i + \theta_j)$.

$$A_0^1 = \begin{bmatrix} c_1 & 0 & s_1 & \alpha_1 * c_1 \\ s_1 & 0 & -c_1 & \alpha_1 * s_1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_0^2 = \begin{bmatrix} c_1 * c_2 & -c_1 * s_2 & s_1 & \alpha_1 * c_1 + \alpha_2 * c_1 * c_2 \\ s_1 * c_2 & -s_1 * s_2 & -c_1 & \alpha_1 * s_1 + \alpha_2 * s_1 * c_2 \\ s_2 & c_2 & 0 & \alpha_2 * s_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_0^3 = \begin{bmatrix} c_1 * c_{23} & s_1 & c_1 * s_{23} & \alpha_1 * c_1 + \alpha_2 * c_1 * c_2 + \alpha_3 * c_1 * c_{23} \\ s_1 * c_{23} & -c_1 & s_1 * s_{23} & \alpha_1 * s_1 + \alpha_2 * s_1 * c_2 + \alpha_3 * s_1 * c_{23} \\ s_{23} & 0 & -c_{23} & \alpha_2 * s_2 + \alpha_3 * s_{23} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_0^4 = \begin{bmatrix} c_1 * c_{23} * c_4 + s_1 * s_4 & c_1 * s_{23} & c_1 * c_{23} * s_4 - s_1 * c_4 & \alpha_1 * c_1 + \alpha_2 * c_1 * c_2 + \alpha_3 * c_1 * c_{23} + d_4 * c_1 * s_{23} \\ s_1 * c_{23} * c_4 - c_1 * s_4 & s_1 * s_{23} & s_1 * c_{23} * s_4 + c_1 * c_4 & \alpha_1 * s_1 + \alpha_2 * s_1 * c_2 + \alpha_3 * s_1 * c_{23} + d_4 * s_1 * s_{23} \\ s_{23} * c_4 & -c_{23} & s_{23} * s_4 & \alpha_2 * s_2 + \alpha_3 * s_{23} - d_4 * c_{23} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_0^5 = \begin{bmatrix} c_5 * (c_1 * c_{23} * c_4 + s_1 * s_4) + s_5 * c_1 * s_{23} & c_1 * s_{23} * s_4 - s_1 * c_4 & s_5 * (c_1 * c_{23} * c_4 + s_1 * s_4) - c_5 * c_1 * s_{23} & \alpha_1 * c_1 + \alpha_2 * c_1 * c_2 + \alpha_3 * c_1 * c_{23} + d_4 * c_1 * s_{23} \\ c_5 * (s_1 * c_{23} * c_4 - c_1 * s_4) + s_5 * s_1 * s_{23} & s_1 * c_{23} * s_4 + c_1 * c_4 & s_5 * (s_1 * c_{23} * c_4 - c_1 * s_4) - c_5 * s_1 * s_{23} & \alpha_1 * s_1 + \alpha_2 * s_1 * c_2 + \alpha_3 * s_1 * c_{23} + d_4 * s_1 * s_{23} \\ c_5 * s_{23} * c_4 - s_5 * c_{23} & s_{23} * s_4 & s_5 * s_{23} * c_4 + c_5 * c_{23} & \alpha_2 * s_2 + \alpha_3 * s_{23} - d_4 * c_{23} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Ο ομογενής πίνακας μετασχηματισμού από το σύνδεσμο 1 έως τον 6 φαίνεται να είναι ο παρακάτω.

$$A_0^6 = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Όπου, αναλυτικά:

$$n_x = c_1 * c_{23} * (c_4 * c_5 * c_6 + s_4 * s_6) + s_1 * (s_4 * c_5 * c_6 - c_4 * s_6) + c_1 * s_{23} * s_5 * c_6$$

$$n_y = s_1 * c_{23} * (c_4 * c_5 * c_6 + s_4 * s_6) - c_1 * (s_4 * c_5 * c_6 - c_4 * s_6) + s_1 * s_{23} * s_5 * c_6$$

$$n_z = s_{23} * (c_4 * c_5 * c_6 + s_4 * s_6) - c_{23} * s_5 * c_6$$

$$o_x = c_1 * c_{23} * (-c_4 * c_5 * c_6 + s_4 * s_6) - s_1 * (s_4 * c_5 * c_6 - c_4 * s_6) - c_1 * s_{23} * s_5 * c_6$$

$$o_y = s_1 * c_{23} * (-c_4 * c_5 * c_6 + s_4 * s_6) + c_1 * (s_4 * c_5 * c_6 - c_4 * s_6) - s_1 * s_{23} * s_5 * c_6$$

$$o_z = s_{23} * (-c_4 * c_5 * c_6 + s_4 * s_6) + c_{23} * s_5 * c_6$$

$$a_x = c_1 * c_{23} * c_4 * s_5 + s_1 * s_4 * s_5 - c_1 * s_{23} * c_5$$

$$a_y = s_1 * c_{23} * c_4 * s_5 - c_1 * s_4 * s_5 - s_1 * s_{23} * c_5$$

$$a_z = s_{23} * c_4 * s_5 + c_{23} * c_5$$

Υπομητρώο Μεταφοράς:

$$p_x = \alpha_1 * c_1 + \alpha_2 * c_1 * c_2 + \alpha_3 * c_1 * c_{23} + d_4 * c_1 * s_{23} = c_1 * (\alpha_1 + \alpha_2 * c_2 + \alpha_3 * c_{23} + d_4 * s_{23})$$

$$p_y = \alpha_1 * s_1 + \alpha_2 * s_1 * c_2 + \alpha_3 * s_1 * c_{23} + d_4 * s_1 * s_{23} = s_1 * (\alpha_1 + \alpha_2 * c_2 + \alpha_3 * c_{23} + d_4 * s_{23})$$

$$p_z = \alpha_2 * s_2 + \alpha_3 * s_{23} - d_4 * c_{23}$$

4.4 Αντίστροφη Κινηματική Ανάλυση

Λύνοντας το αντίστροφο κινηματικό πρόβλημα μπορούν να εντοπιστούν οι τιμές των γωνιών για την επιθυμητή θέση και προσανατολισμό του άκρου. Η επίλυση αυτού του προβλήματος είναι αρκετά πιο περίπλοκη από την επίλυση του ευθέως. Ως γνωστές τιμές, πέρα από τις μεταβλητές X,Y,Z της θέσης, χρειάζονται και οι μεταβλητές O,A,T του προσανατολισμού του εργαλείου.

Για την εύρεση του προσανατολισμού του άκρου του βραχίονα μπορούν να χρησιμοποιηθούν τρεις διαφορετικοί τρόποι, Euler, Roll-Pitch-Yaw και ως προς το σύστημα της βάσης), όπως παρουσιάζονται στα παρακάτω μητρώα. Στην παρούσα διπλωματική χρησιμοποιήθηκε το μητρώο Roll-Pitch-Yaw.

$$A_{euler}=\begin{bmatrix} \cos 0\cos A\cos T-\sin O\sin T & -\cos T\sin O-\cos O\cos A\sin T & \cos O\sin A & X \\ \cos O\sin T+\cos A\cos T\sin O & \cos O\cos T-\cos A\sin O\sin T & \sin O\sin A & Y \\ -\cos T\sin A & \sin A\sin T & \cos A & Z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_{RPY}=\begin{bmatrix} \cos 0\cos A & \cos O\sin A\sin T-\cos T\sin O & \sin O\sin T+\cos O\cos T\sin A & X \\ \cos A\sin O & \cos O\cos T+\sin O\sin A\sin T & \cos T\sin O\sin A-\cos O\sin T & Y \\ -\sin A & \cos A\sin T & \cos A\cos T & Z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_{BASE}=\begin{bmatrix} \cos T\cos A & \cos T\sin A\sin O-\cos O\sin T & \sin O\sin T+\cos O\cos T\sin A & X \\ \cos A\sin T & \cos O\cos T+\sin O\sin A\sin T & \cos O\sin T\sin A-\cos T\sin O & Y \\ -\sin A & \cos A\sin O & \cos A\cos O & Z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Αφού κατασκευαστεί το παραπάνω μητρώο, γίνεται ο υπολογισμός της θέσης και του προσανατολισμού του έκτου μέλους του ρομπότ. Για να γίνει αυτό, εφαρμόζεται ο τύπος:

$$A_0^6 = A_0^7 * A_7^6$$

Όπου ως A_0^7 χρησιμοποιείται το μητρώο που δημιουργήθηκε παραπάνω και ως A_7^6 ο αντίστροφος πίνακας του A_6^7 και είναι:

$$A_7^6 = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -d_7 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Στη συνέχεια, με γνωστό το μητρώο A06, και ορίζοντας μία θέση και έναν προσανατολισμό για το άκρο του βραχίονα, υπολογίζονται οι γωνίες των αρθρώσεων.

$$p_x = c_1 * (\alpha_1 + \alpha_2 * c_2 + \alpha_3 * c_{23} + d_4 * s_{23}) \quad (1)$$

$$p_y = s_1 * (\alpha_1 + \alpha_2 * c_2 + \alpha_3 * c_{23} + d_4 * s_{23}) \quad (2)$$

$$p_z = \alpha_2 * s_2 + \alpha_3 * s_{23} - d_4 * c_{23} \quad (3)$$

Διαιρώντας τη σχέση (2) με την (1) προκύπτει:

$$\frac{p_y}{p_x} = \frac{s_1 * (\alpha_1 + \alpha_2 * c_2 + \alpha_3 * c_{23} + d_4 * s_{23})}{c_1 * (\alpha_1 + \alpha_2 * c_2 + \alpha_3 * c_{23} + d_4 * s_{23})} = \frac{s_1}{c_1}$$

Η θ_1 υπολογίζεται από την παρακάτω σχέση:

$$\theta_1 = \tan^{-1} \frac{p_y}{p_x}$$

Πλέον, με γνωστή τη γωνία θ_1, γίνεται εφικτή η εύρεση της γωνίας θ_2. Αρχικά πολλαπλασιάζονται οι εξισώσεις (1) με $\cos\theta_1$ και η (2) με $\sin\theta_1$, από τις οποίες προκύπτουν οι (4) και (5) αντίστοιχα.

$$p_x * c_1 = (c_1 * (\alpha_1 + \alpha_2 * c_2 + \alpha_3 * c_{23} + d_4 * s_{23})) * c_1 \quad (4)$$

$$p_y * s_1 = (s_1 * (\alpha_1 + \alpha_2 * c_2 + \alpha_3 * c_{23} + d_4 * s_{23})) * s_1 \quad (5)$$

Στη συνέχεια, γίνεται πρόσθεση κατά μέλη των δύο εξισώσεων (4) και (5), καταλήγοντας στην (6).

$$p_x * c_1 + p_y * s_1 = s_1^2 * (\alpha_1 + \alpha_2 * c_2 + \alpha_3 * c_{23} + d_4 * s_{23}) + c_1^2 * (\alpha_1 + \alpha_2 * c_2 + \alpha_3 * c_{23} + d_4 * s_{23})$$
$$\Rightarrow p_x * c_1 + p_y * s_1 = \alpha_1 + \alpha_2 * c_2 + \alpha_3 * c_{23} + d_4 * s_{23}$$

$$\Rightarrow p_x * c_1 + p_y * s_1 - \alpha_1 - \alpha_2 * c_2 = \alpha_3 * c_{23} + d_4 * s_{23} \quad (6)$$

Στη σχέση (3) χωρίζονται τα γνωστά με τα άγνωστα και προκύπτει η εξίσωση (7).

$$(3) \Leftrightarrow p_z - \alpha_2 * s_2 = \alpha_3 * s_{23} - d_4 * c_{23} \quad (7)$$

Ύστερα, υψώνοντας τις σχέσεις (6) και (7) στο τετράγωνο και αθροίζοντας τις προκύπτει η σχέση (8).

$$(p_x * c_1 + p_y * s_1 - \alpha_1 - \alpha_2 * c_2)^2 + (p_z - \alpha_2 * s_2)^2 = \alpha_3{}^2 + d_4{}^2 \quad (8)$$

Η οποία σχέση είναι της μορφής:

$$A\sin\theta_2 - B\cos\theta_2 = C \quad (9)$$

Όπου το κάθε μέλος είναι το εξής:

$$A = -2 * \alpha_2 * p_z$$
$$B = -2 * \alpha_2 * (p_x * c_1 + p_y * s_1 - \alpha_1)$$
$$C = \alpha_3{}^2 + d_4{}^2 - \alpha_2{}^2 - p_z{}^2 - (p_x * c_1 + p_y * s_1 - \alpha_1)^2$$

Επιλύοντας την εξίσωση (9) προκύπτει η γωνία θ2, η οποία έχει δύο (2) δυνατές λύσεις.

$$\theta_2 = \tan^{-1}\left(\frac{C}{\pm\sqrt{A^2 + B^2 - C^2}}\right) + \tan^{-1}\frac{\mathrm{B}}{A}$$

Η παραπάνω εξίσωση ισχύει μόνο όταν $A^2+B^2>C^2$, αφού η τετραγωνική ρίζα δεν μπορεί να πάρει αρνητικό αριθμό, δηλαδή:

$$(-2 * \alpha_2 * p_z)^2 + (2 * \alpha_2 * (p_x * c_1 + p_y * s_1 - \alpha_1))^2 > (\alpha_3{}^2 + d_4{}^2 - \alpha_2{}^2 - p_z{}^2 - (p_x * c_1 + p_y * s_1 - \alpha_1)^2)^2$$

Αφού έχει βρεθεί και η γωνία θ_2, υπολογίζεται η γωνία θ_3. Από τη σχέση (7), λύνοντας αρχικά ως προς $\cos\theta_{23}$ και, ύστερα, ως προς $\sin\theta_{23}$, προκύπτουν οι σχέσεις (10) και (11) αντίστοιχα.

$$(7) \Rightarrow c_{23} = \frac{-\mathrm{p}_z + \alpha_2 * s_2 + \alpha_3 * s_{23}}{d_4} \quad (10)$$

$$(7) \Rightarrow s_{23} = \frac{\mathrm{p}_z - \alpha_2 * s_2 + d_4 * c_{23}}{\alpha_3} \quad (11)$$

Παίρνοντας τη σχέση (6) και αντικαθιστώντας μία την (10) και μια (11) και λύνοντας ως προς $\sin\theta_{23}$ και $\cos\theta_{23}$ αντίστοιχα, προκύπτουν οι δύο παρακάτω εξισώσεις:

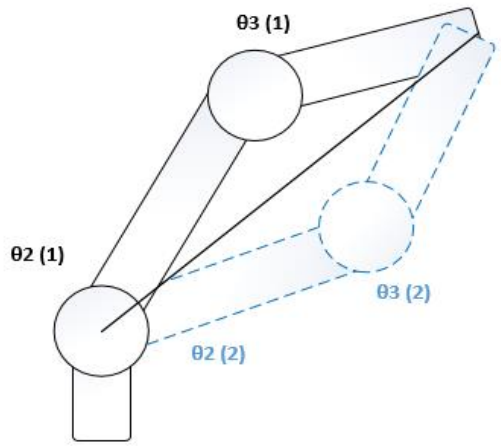
$$(6)\overset{(10)}{\Longrightarrow} s_{23} = \frac{p_x * c_1 + p_y * s_1 - \alpha_1 - \alpha_2 * c_2 - \frac{\alpha_3}{d_4} * (-p_z + \alpha_2 * s_2)}{\frac{\alpha_3^{\,2} + d_4^{\,2}}{d_4}} \tag{12}$$

$$(6)\overset{(11)}{\Longrightarrow} c_{23} = \frac{p_x * c_1 + p_y * s_1 - \alpha_1 - \alpha_2 * c_2 - \frac{d_4}{\alpha_3} * (p_z - \alpha_2 * s_2)}{\frac{\alpha_3^{\,2} + d_4^{\,2}}{\alpha_3}} \tag{13}$$

Διαιρώντας τη σχέση (12) με τη σχέση (13) προκύπτει η γωνία θ_3, όπως φαίνεται παρακάτω.

$$\theta_3 = \tan^{-1}\left(\frac{\frac{p_x * c_1 + p_y * s_1 - \alpha_1 - \alpha_2 * c_2 - \frac{\alpha_3}{d_4} * (-p_z + \alpha_2 * s_2)}{\frac{\alpha_3^{\,2} + d_4^{\,2}}{d_4}}}{\frac{p_x * c_1 + p_y * s_1 - \alpha_1 - \alpha_2 * c_2 - \frac{d_4}{\alpha_3} * (p_z - \alpha_2 * s_2)}{\frac{\alpha_3^{\,2} + d_4^{\,2}}{\alpha_3}}} \right) - \theta_2$$

Παρατηρείται πως υπάρχουν δύο λύσεις για τη γωνία θ_2 και κατ' επέκταση για τη γωνία θ_3. Το πρώτο ζεύγος λύσεων αντιστοιχεί στο να είναι ο «αγκώνας» του βραχίονα επάνω, ενώ το δεύτερο ζεύγος στο να είναι κάτω. Είτε επιλεχθεί η πρώτη λύση είτε η δεύτερη, ο βραχίονας καταλήγει στο ίδιο τελικό σημείο. Αυτό ακριβώς αποτυπώνεται στην εικόνα.



Εικόνα 12: Δύο ζεύγη γωνιών, ίδιο τελικό σημείο

Έχοντας υπολογίσει τις γωνίες θ_1,θ_2 και θ_3, οι οποίες καθορίζουν τη θέση του βραχίονα, πρέπει να υπολογιστούν οι γωνίες θ_4,θ_5 και θ_6, οι οποίες καθορίζουν τον προσανατολισμό του. Για να βρεθούν, θα χρησιμοποιηθεί το υπομητρώο περιστροφής R_3^6. Για να βρεθεί αυτό το υπομητρώο χρειάζονται τα υπομητρώα R_0^3 και R_0^6 του A_0^6.

$$R_3^6 = (R_0^3)^{-1} * R_0^6$$

Όπου το R_3^6 είναι της μορφής:

$$R_3^6 = \begin{bmatrix} n_x & o_x & a_x \\ n_y & o_y & a_y \\ n_z & o_z & a_z \end{bmatrix}$$

Το υπομητρώο R_3^6 μπορεί να υπολογιστεί με τη βοήθεια της παρακάτω εξίσωσης:

$$R_3^6 = R_3^4 * R_4^5 * R_5^6$$

$$R_3^6 = \begin{bmatrix} c_4 * c_5 * c_6 + s_4 * s_6 & -c_4 * c_5 * s_6 + s_4 * c_6 & c_4 * s_5 \\ s_4 * c_5 * c_6 - c_4 * s_6 & -s_4 * c_5 * s_6 - c_4 * c_6 & s_4 * s_5 \\ s_5 * c_6 & -s_5 * s_6 & -c_5 \end{bmatrix}$$

Εάν η γωνία θ_5 είναι ίση με 0° ή 180°, τότε το μητρώο R_3^6 απλοποιείται σημαντικά. Λόγω, όμως, του πεδίου ορισμού της θ_5 ως προς Denavit-Hartenberg, όπου $35° \leq \theta_5 \leq 325°$, η γωνιά θ_5 μπορεί να πάρει μόνο την τιμή 180. Έτσι αν $\theta_5=180$, τότε:

$$R_3^6 = \begin{bmatrix} -c_4 * c_6 + s_4 * s_6 & c_4 * s_6 + s_4 * c_6 & 0 \\ -s_4 * c_6 - c_4 * s_6 & s_4 * s_6 - c_4 * c_6 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_3^6(1,1) = -c_4 * c_6 + s_4 * s_6 = -\cos(\theta_4 + \theta_6) \quad (14)$$

$$R_3^6(2,1) = -s_4 * c_6 - c_4 * s_6 = -\sin(\theta_4 + \theta_6) \quad (15)$$

Διαιρώντας τη σχέση (15) με τη σχέση (14) υπολογίζονται οι γωνίες θ_4 και θ_6.

$$\tan(\theta_4 + \theta_6) = \frac{R_3^6(2,1)}{R_3^6(1,1)} = \frac{-\sin(\theta_4 + \theta_6)}{-\cos(\theta_4 + \theta_6)} \Rightarrow \theta_4 + \theta_6 = \tan^{-1} \frac{R_3^6(2,1)}{R_3^6(1,1)}$$

Όπως φαίνεται από την παραπάνω σχέση, η θ_4 με τη θ_6 δεν μπορούν να υπολογιστούν ξεχωριστά. Έτσι χάνεται ένας βαθμός ελευθερίας. Για την επίλυση της εξίσωσης θεωρείται η θ_4 ίση με το μηδέν (θ_4=0). Διαφορετικά, κρατάει την προηγούμενη τιμή της. Έτσι, μπορεί να υπολογιστεί η γωνία θ_6.

Στην περίπτωση που η θ_5 δεν είναι 180°, τότε:

$$a_z = R_3^6(3,3) = -\cos\theta_5$$

$$\cos\theta_5 = -a_z$$

$$\cos^2\theta_5 + \sin^2\theta_5 = 1$$

$$\sin\theta_5 = \pm\sqrt{1 - a_z^{\,2}}$$

$$\tan\theta_5 = \frac{\pm\sqrt{1 - a_z^{\,2}}}{-a_z} \Rightarrow \theta_5 = \tan^{-1} \frac{\pm\sqrt{1 - a_z^{\,2}}}{-a_z} = \tan^{-1} \frac{I_5 * \sqrt{1 - a_z^{\,2}}}{-a_z}$$

Όπου $I_5 = \pm 1$. Όταν $I_5 = 1$, τότε η λύση που προκύπτει είναι αντεστραμμένου καρπού, ενώ για $I_5 = -1$ είναι μη αντεστραμμένου καρπού.

Πλέον, με γνωστή τη γωνία θ_5, υπολογίζεται η γωνία θ_4.

$$R_3^6(1,3) = a_x = \cos\theta_4 * \sin\theta_5$$

$$R_3^6(2,3) = a_y = \sin\theta_4 * \sin\theta_5$$

$$\theta_4 = \tan^{-1} \frac{I_5 * a_y}{I_5 * a_x}$$

Τέλος, γίνεται ο υπολογισμός της γωνίας θ_6.

$$R_3^6(3,1) = n_z = \cos\theta_6 * \sin\theta_5$$
$$R_3^6(3,2) = o_z = \cos\theta_6 * \sin\theta_5$$
$$\theta_6 = \tan^{-1}\frac{-I_5 * o_z}{I_5 * n_z}$$

5. Προσομοίωση Ρομπότ σε Γραφικό Χώρο

5.1 Γενικά Για Την Προσομοίωση Των Ρομπότ

Προσομοίωση ονομάζεται η αναπαράσταση ή αλλιώς η μίμηση μίας λειτουργίας συστημάτων ή της εξέλιξης διαδικασιών με τη βοήθεια ενός υπολογιστή [18]. Για την επίτευξή της χρησιμοποιείται κάποια μαθηματική περιγραφή ή ένα μοντέλο που αφορά ένα πραγματικό σύστημα.

Υπάρχουν πολλοί λόγοι για τους οποίους χρησιμοποιείται η προσομοίωση μοντέλων. Αρχικά, υπάρχει η δυνατότητα εντοπισμού και επίλυσης προβλημάτων με ασφάλεια. Προσομοιώνοντας ένα μοντέλο, μπορούν να επαληθευτούν λύσεις, δίνοντας έτσι σαφείς πληροφορίες, ακόμα και για πολύπλοκα συστήματα. Επίσης, γίνεται ευκολότερη η εφαρμογή πειραμάτων, με την βοήθεια ψηφιακών αναπαραστάσεων, είτε σε δισδιάστατη είτε σε τρισδιάστατη μορφή. Πολλές επιχειρήσεις χρησιμοποιούν την προσομοίωση, όταν είναι ανέφικτο στο πραγματικό μοντέλο, για πειράματα, αφού με αυτόν τον τρόπο μπορεί να γίνει εξοικονόμηση χρόνου και χρήματος [19].

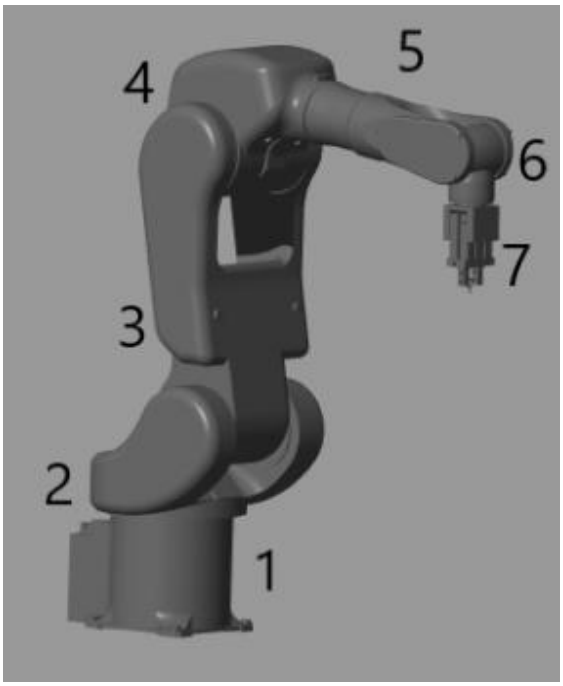
5.2 Προγράμματα Που Χρησιμοποιήθηκαν

5.2.1 SolidWorks

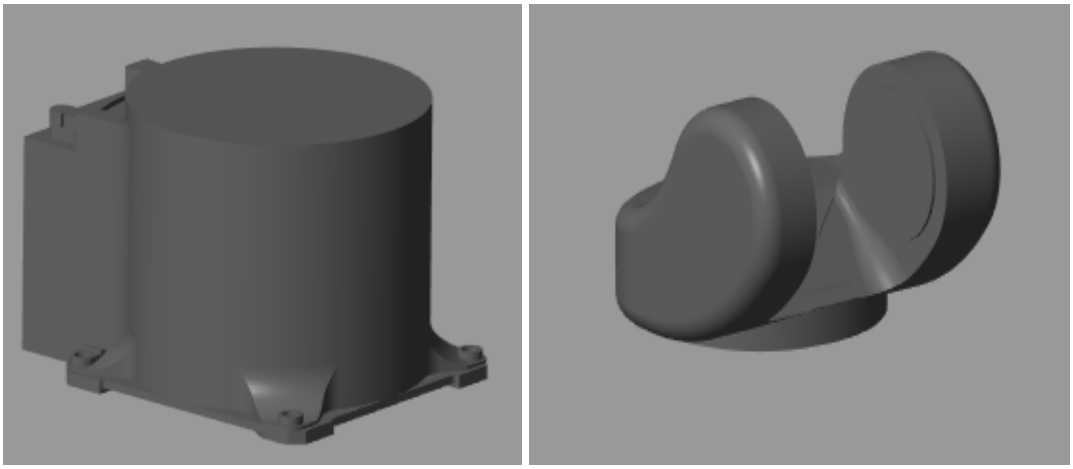
Ο σχεδιασμός του ρομποτικού βραχίονα έγινε στο SolidWorks, με βάση των δεδομένων της επίσημης ιστοσελίδας του βραχίονα [16].

Εικόνα 13: SolidWorks [20]

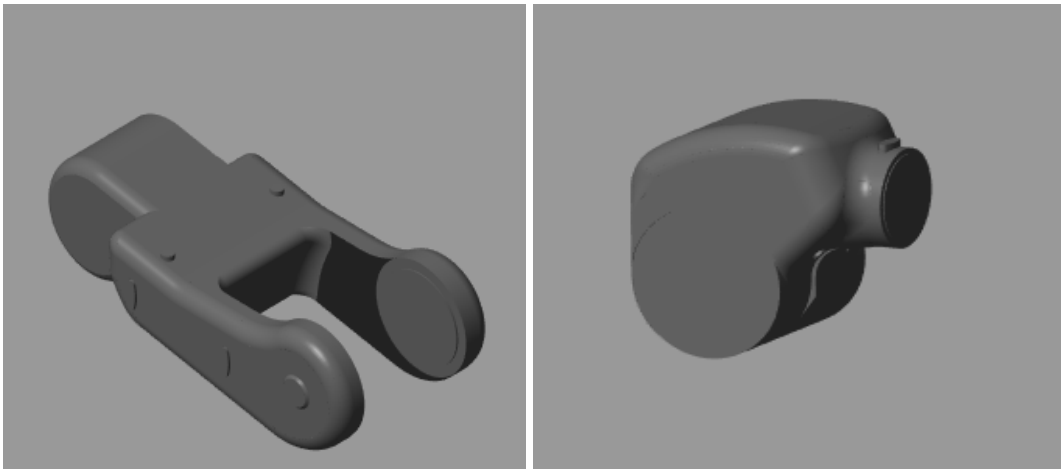
Το SolidWorks είναι ένα πρόγραμμα που προσφέρει δισδιάστατο και τρισδιάστατο σχεδιασμό και χρησιμοποιείται κυρίως για μηχανολογικό σχέδιο [20]. Δίνει τη δυνατότητα απλού αλλά και περίπλοκου σχεδιασμού. Η χρήση του είναι αρκετά διαδεδομένη, σε εκπαιδευτικό αλλά και σε επαγγελματικό επίπεδο. Στη συγκεκριμένη εργασία δεν περάστηκε το σχέδιο του βραχίονα στο Simscape απευθείας από το SolidWorks, αλλά σχεδιάστηκε λεπτομερώς χρησιμοποιώντας τα μπλοκ του Simulink και τα μέλη (parts) του βραχίονα.



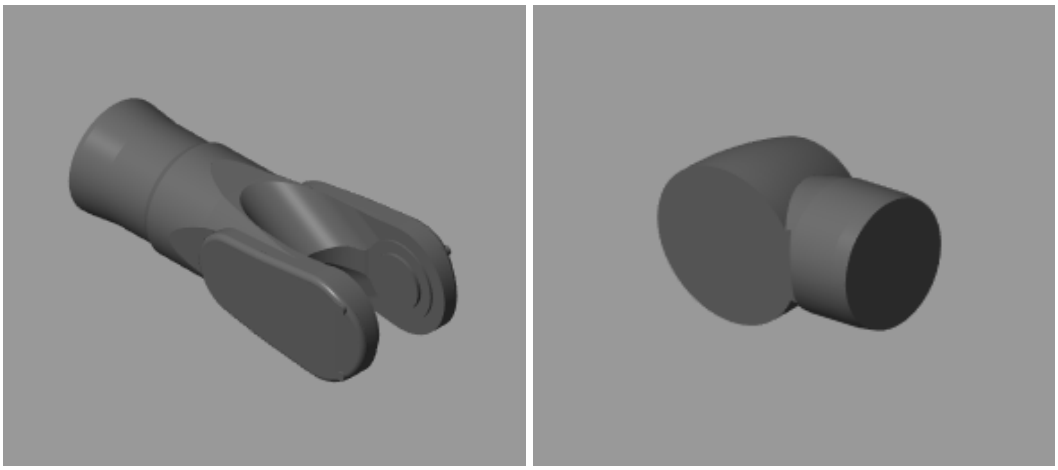
Εικόνα 14: Αρίθμηση μελών του βραχίονα



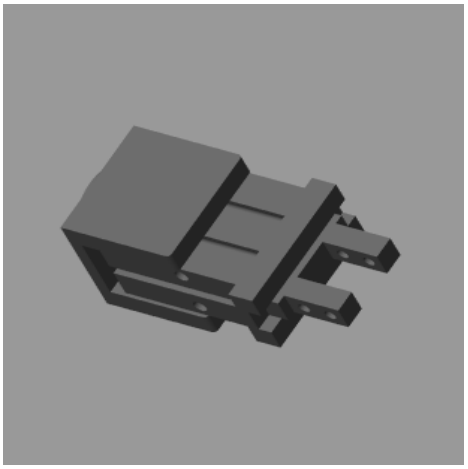
Εικόνα 15: Μέλος 1 και 2



Εικόνα 16: Μέλος 3 και 4



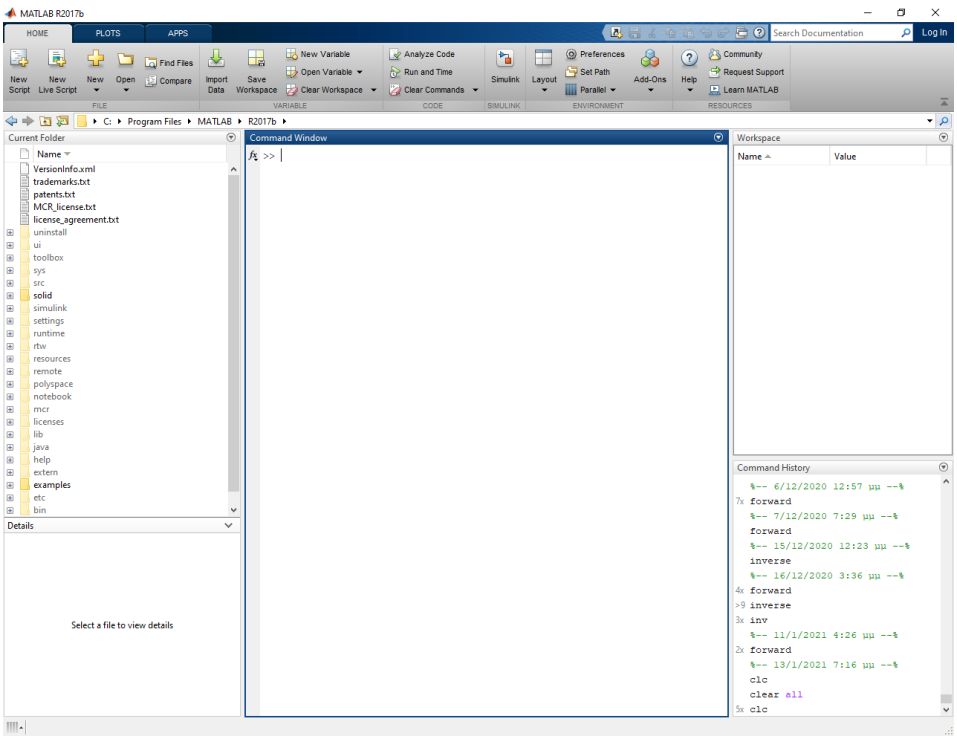
Εικόνα 17: Μέλος 5 και 6



Εικόνα 18: Μέλος 7 (gripper)

5.2.2 MATLAB

Το MATLAB είναι μία πλατφόρμα προγραμματισμού με ποικίλες δυνατότητες, που επιτρέπει σε μηχανικούς, επιστήμονες και όχι μόνο, να δημιουργούν προγράμματα για ανάλυση δεδομένων, να αναπτύσσουν αλγορίθμους και, τέλος, να κατασκευάζουν μοντέλα και εφαρμογές πολλών ειδών, κυρίως για μηχανική εκμάθηση (machine learning) [21]. Το όνομά του είναι συντομογραφία των λέξεων MATrix LABoratory της εταιρείας The MathWorks, Inc [22].



Εικόνα 19: Περιβάλλον MATLAB R2017b

Το MATLAB δίνει τη δυνατότητα δημιουργίας GUI, το οποίο, όπως εξηγήθηκε σε προηγούμενο κεφάλαιο, είναι ένα γραφικό περιβάλλον χρήστη. Αυτό διευκολύνει τη χρήση μίας εφαρμογής, αφού μπορεί να χρησιμοποιηθεί από χρήστες που δε ξέρουν κώδικα. Δηλαδή, μπορεί να κατασκευαστεί μία εφαρμογή, η οποία μπορεί να είναι διαδραστική και να χειρίζεται γραφικά από τον οποιονδήποτε.

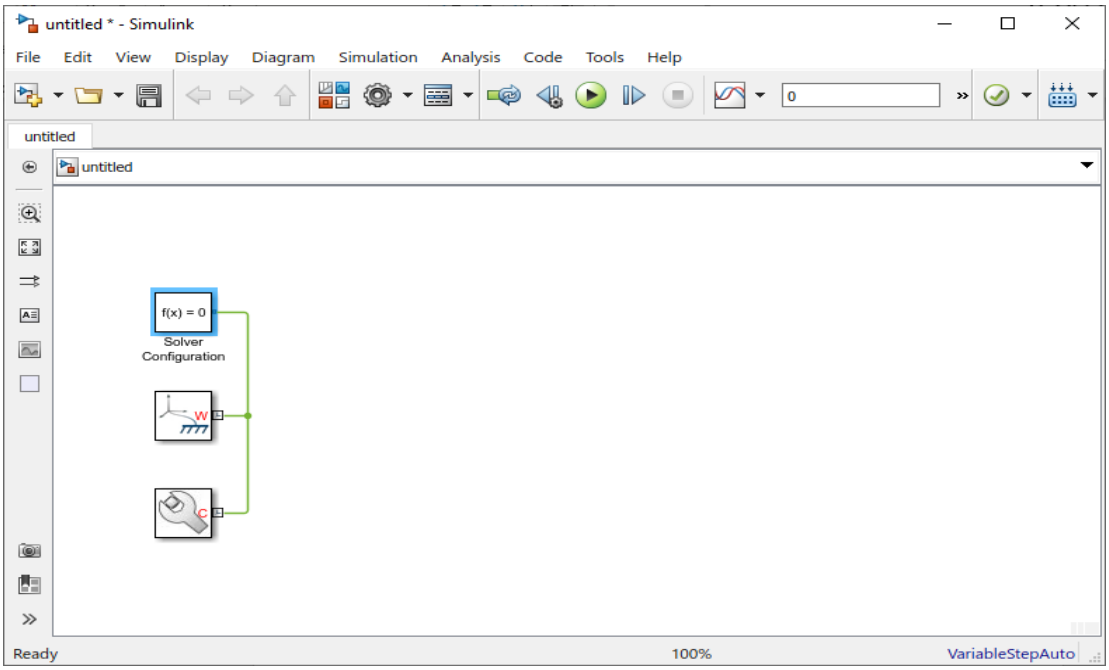
5.2.3 Simulink

Το Simulink είναι ένα πρόγραμμα ενσωματωμένο στο MATLAB. Προσφέρει ένα γραφικό περιβάλλον, το οποίο μπορεί να χρησιμοποιηθεί για προσομοίωση και μοντελοποίηση ενός συστήματος. Ένα από τα προτερήματά του είναι η εύκολη συνεργασία του με το MATLAB, όπως και με όλες τις υπόλοιπες δυνατότητες που εκείνο προσφέρει. Για να μπορέσει να γίνει μία επιτυχής μοντελοποίηση ενός συστήματος, αρκεί η χρήση συγκεκριμένων εργαλείων, βιβλιοθηκών και προσαρμοσμένων μπλοκ. Το συγκεκριμένο πρόγραμμα μπορεί να χειριστεί από απλά έως πολύπλοκα συστήματα [23].

Πιο συγκεκριμένα, για τη μοντελοποίηση και την προσομοίωση του ρομποτικού βραχίονα χρησιμοποιήθηκε το Simscape Multibody, ένα πρόγραμμα το οποίο βρίσκεται στο Simulink και αξιοποιείται για μηχανολογικά συστήματα, όπως ρομπότ και αναρτήσεις οχημάτων. Σε αυτό υπάρχουν μπλοκ τα οποία εκπροσωπούν σώματα, αισθητήρες και άλλα πολλά στοιχεία τα οποία χρησιμοποιούνται για τη διαμόρφωση ενός συστήματος. Είναι εφικτό να περαστούν ολόκληρα

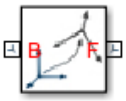
τρισδιάστατα σχέδια σε κομμάτια (parts), όπως, επίσης, οι μάζες, οι αδράνειες οι αρθρώσεις και ό,τι άλλο χρειάζεται για την δυναμική απεικόνιση του μοντέλου [24].

Αρχικά, δημιουργήθηκε ένα νέο αρχείο Simscape Multibody στο Simulink. Εκεί υπάρχουν τρία μπλοκ που είναι απαραίτητα για την προσομοίωση. Το πρώτο περιλαμβάνει πληροφορίες ρυθμίσεων για την προσομοίωση και καθορίζει τις παραμέτρους επίλυσης του μοντέλου. Το δεύτερο δηλώνει το παγκόσμιο σύστημα συντεταγμένων και το τρίτο τη βαρύτητα.



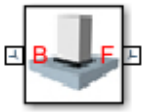
Εικόνα 20: Simscape Multibody

Αμέσως μετά ξεκίνησαν να τοποθετούνται τα απαραίτητα μπλοκ για τη σύσταση του βραχίονα. Παρακάτω παρουσιάζονται και εξηγούνται τα μπλοκ που χρησιμοποιήθηκαν στη συγκεκριμένη προσομοίωση.



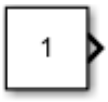
Άκαμπτος μετασχηματισμός (Rigid Transform): Περιστρέφει το πλαίσιο του μπλοκ που ακολουθεί (follower – f) σε σχέση με το προηγούμενο (base - b).

Στερεό (Solid): Σε αυτό το μπλοκ δηλώνονται τα συμπαγή στοιχεία και προστίθονται η γεωμετρία, η αδράνεια και το χρώμα. Σε συνεργασία με το Rigid Transform μπορεί να γίνει η μοντελοποίηση ενός σώματος.



Άρθρωση συγκόλλησης (Weld Joint): Η συγκεκριμένη άρθρωση δεν έχει κανένα βαθμό ελευθερίας και χρησιμοποιείται για να συμπίπτουν δύο σώματα πάντα.

Άρθρωση περιστροφής (Revolute Joint): Αυτό το μπλοκ χρησιμοποιείται για την περιστροφή ενός στερεού ως προς το προηγούμενό του. Έχει έναν βαθμό ελευθερίας και περιστρέφεται ως προς τον άξονα Z.

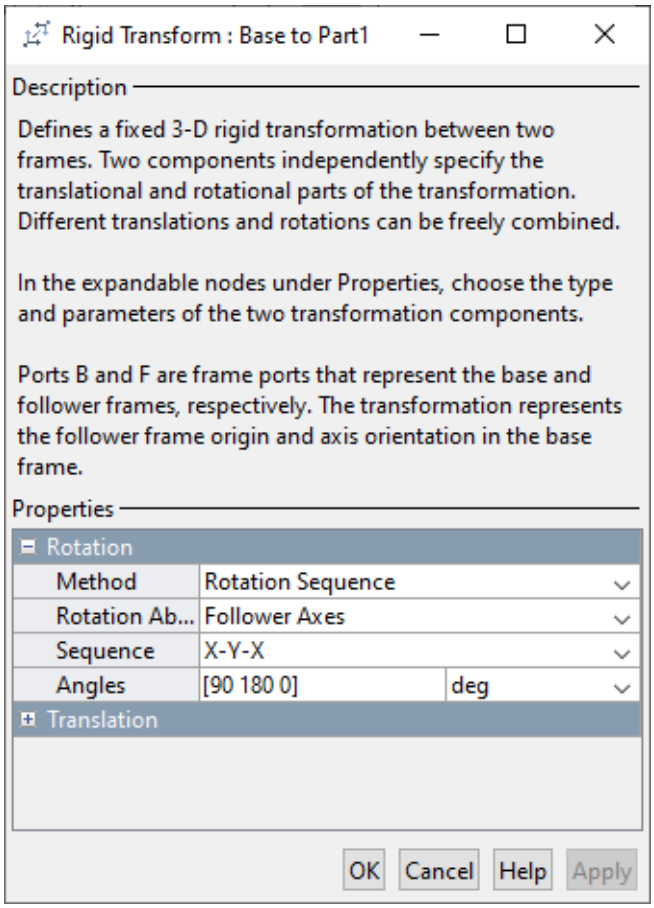


Σταθερά (Constant): Παράγει μία σταθερή τιμή.

Μετατροπέας Simulink-PS (Simulink-PS Converter): Μετατρέπει το simulink σήμα εισόδου σε φυσικό σήμα.

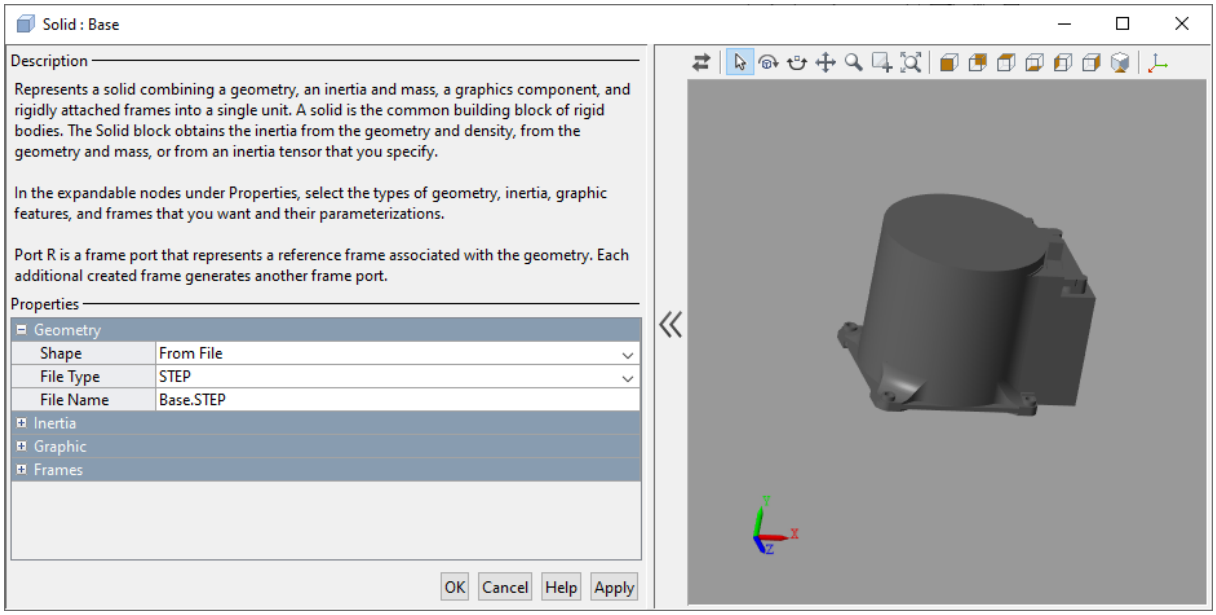
Ρυθμιστής Κέρδους (Slider Gain): Αυτό το μπλοκ καθιστά εφικτή την κλιμάκωση ενός κέρδους κατά τη διάρκεια μίας προσομοίωσης. Για να γίνει αυτό απαιτεί και έναν ρυθμιστή (slider).

Τα Rigid Transforms τοποθετήθηκαν με σκοπό αφενός μεν την επιθυμητή σχεδίαση του βραχίονα στο χώρο, αφετέρου δε την προσαρμογή των συστημάτων συντεταγμένων κάθε μέλους. Είναι, ίσως, η πιο περίπλοκη διαδικασία, αφού το σύστημα συντεταγμένων του ενός μέλους αλλάζει με βάση το προηγούμενο. Έτσι, το κάθε rigid transform είναι υπεύθυνο για την επιθυμητή θέση του κάθε μέλους στον χώρο ως προς το προηγούμενό του, όπως, επίσης, και για την τοποθέτηση των συστημάτων συντεταγμένων με τέτοιον τρόπο, ώστε να μην επηρεάζεται η θέση του και να μπορεί να γίνει η περιστροφή του ως προς τον άξονα z.



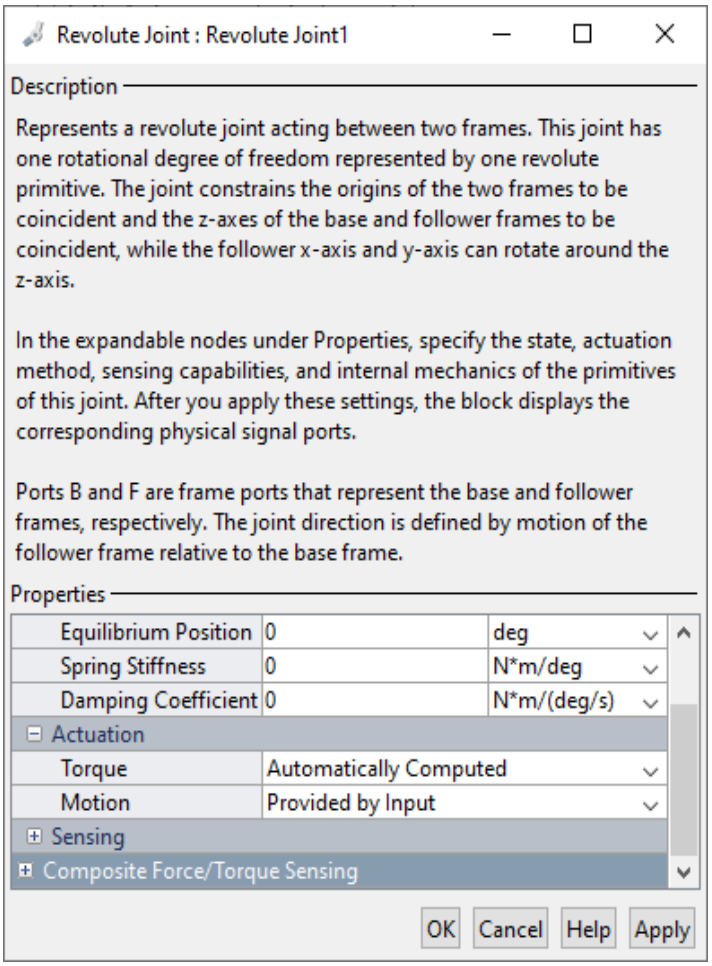
Εικόνα 21: Άκαμπτος μετασχηματισμός (Rigid Transform)

Στο κάθε Solid Block εισέρχεται το σχέδιο .STEP (file format STEP203) του αντίστοιχου part του SolidWorks από το φάκελο του αρχείου. Σε περίπτωση που ο φάκελος αλλάξει θέση, τότε θα πρέπει να γίνει επανατοποθέτηση των σχεδίων στο Block. Για να γίνει αυτό, αρκεί να πατηθεί η επιφάνεια δίπλα στο File Name, όπως φαίνεται στην εικόνα 22.



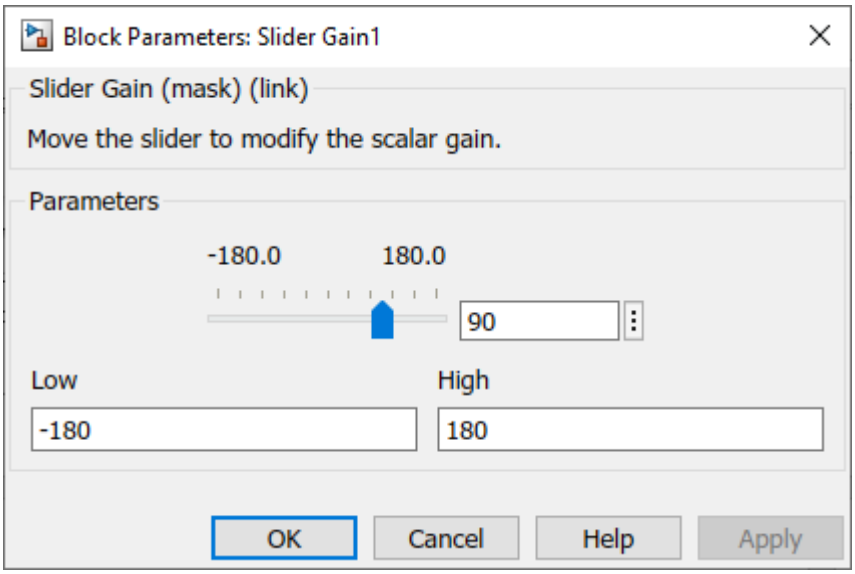
Εικόνα 22: Στερεό (Solid)

Με τη χρήση της περιστροφικής άρθρωσης (Revolute Joint) πετυχαίνεται η περιστροφή των μελών κατά τον άξονα z. Για να μπορεί να γίνει επιτυχώς, θα πρέπει τα συστήματα συντεταγμένων κάθε μέλους να είναι έτσι τοποθετημένα, ώστε ο άξονας z να είναι στη πλευρά προς περιστροφή. Για να ανοίξει η είσοδος του μπλοκ, ώστε να λαμβάνει την επιθυμητή θέση, το Motion ορίστηκε να παίρνει δεδομένα, όπως φαίνεται στην εικόνα 23. Επίσης, ορίστηκαν οι μονάδες να είναι μοίρες (degrees).



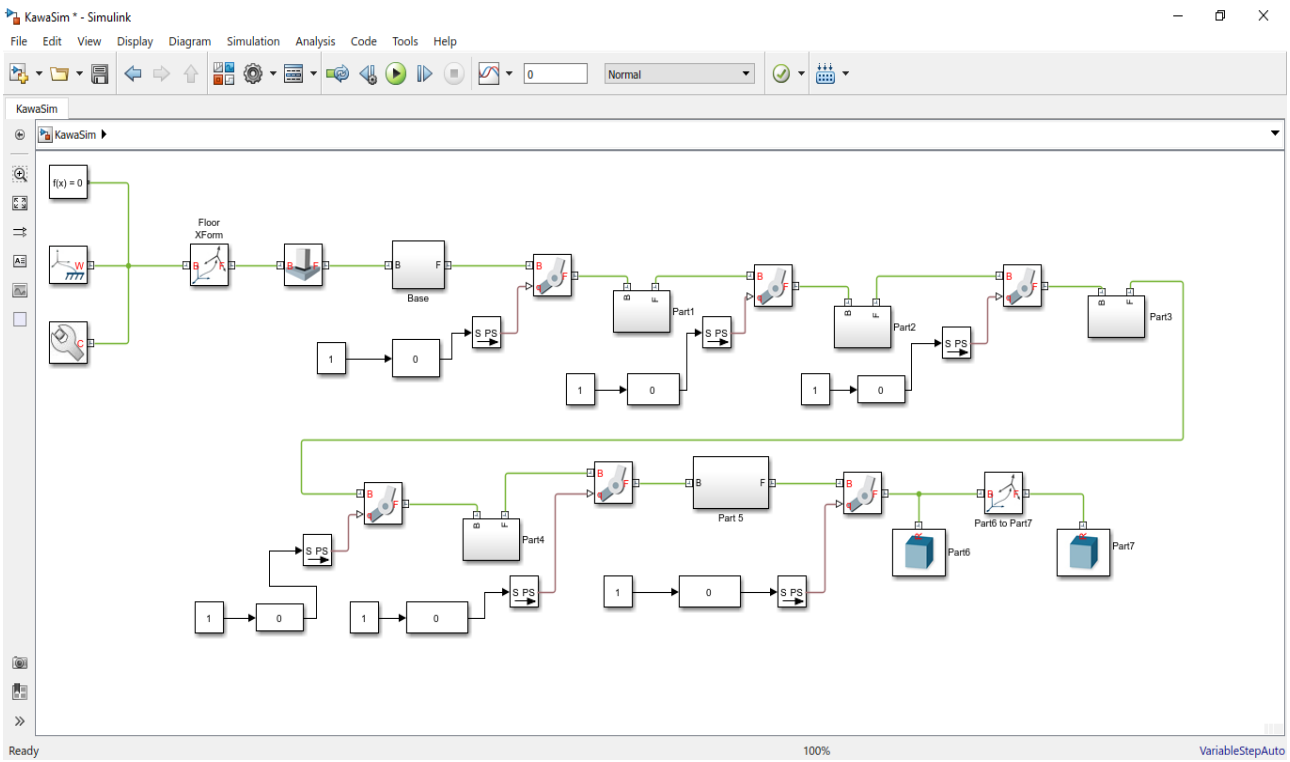
Εικόνα 23: Άρθρωση περιστροφής (Revolute Joint)

Σύμφωνα με τον κατασκευαστή, η κάθε γωνία έχει ένα πεδίο ορισμού. Έτσι, όταν ορίζεται μία τιμή από το χρήστη, θα πρέπει το Simulink να γνωρίζει εάν αυτή είναι ενός ορίων. Με το Slider Gain μπορούν να οριστούν τα ανώτερα και τα κατώτερα όρια της κάθε γωνίας. Σε περίπτωση που η τιμή είναι εκτός, το Slider Gain βγάζει σφάλμα και δε γίνεται η προσομοίωση.



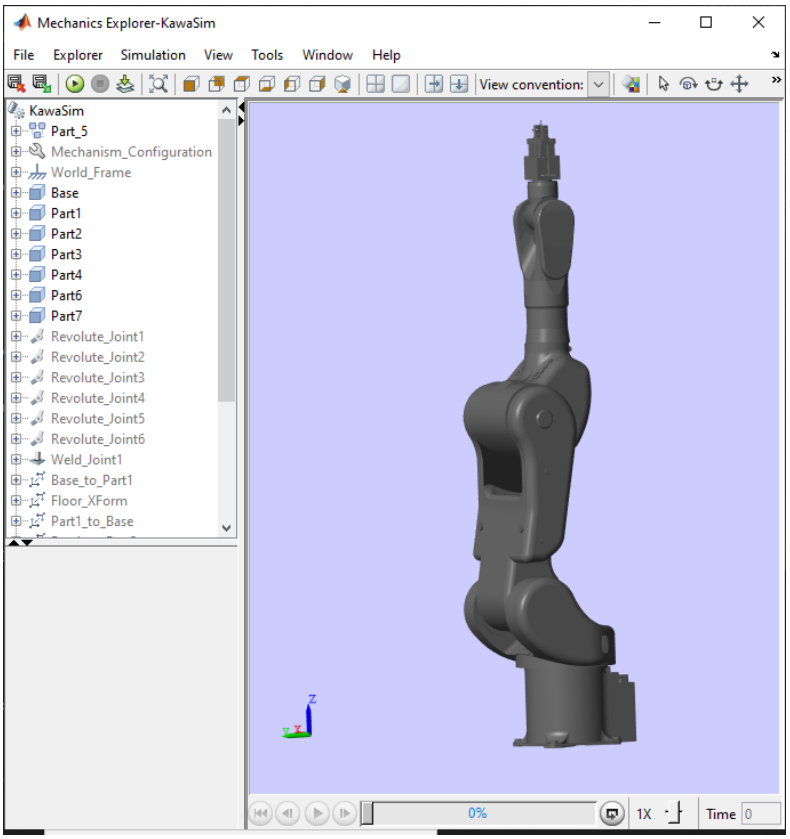
Εικόνα 24: Ρυθμιστής Κέρδους (Slider Gain)

Η τιμή της επιθυμητής γωνίας λαμβάνεται από μία σταθερά (constant), η οποία είναι συνδεδεμένη με το γραφικό μοντέλο που χειρίζεται ο χρήστης. Αυτή στέλνεται από τη σταθερά στο Slider Gain, το οποίο τη φιλτράρει για το αν ανήκει μέσα στα επιτρεπτά όρια. Αμέσως μετά, μεταφράζεται σε φυσικό σήμα, μέσω του μετατροπέα, και στέλνεται στην περιστροφική άρθρωση για να γίνει η κίνηση.



Εικόνα 25: Σχεδιασμός στο Simulink

Την οπτικοποίηση της εφαρμογής την παρέχει το Mechanics Explorer, το οποίο είναι εργαλείο του Simulink. Το συγκεκριμένο πρόγραμμα επιτρέπει την περιήγηση του μοντέλου, δηλαδή την αλλαγή οπτικής γωνίας, τη μεγέθυνση, τη μετακίνηση και άλλα [25].



Εικόνα 26: Οπτικοποίηση βραχίονα

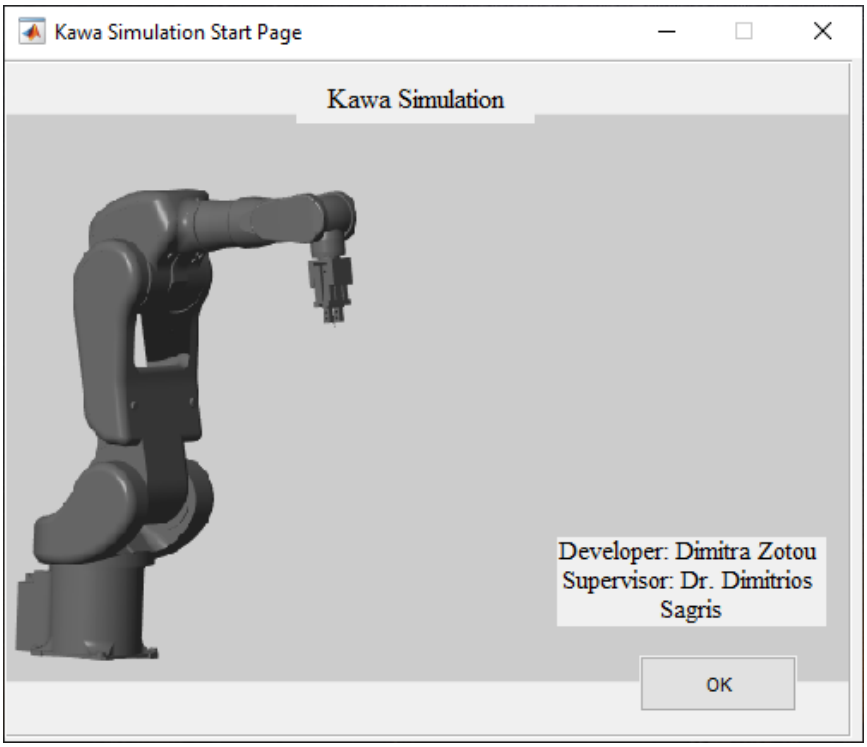
Η εικόνα 26 αναπαριστά τον βραχίονα που κατασκευάστηκε και χρησιμοποιείται για την αναπαράσταση της προσομοίωσης του βραχίονα.

6. Ανάπτυξη Εφαρμογής Χειρισμού του Βραχίονα

6.1 Περιβάλλον Λογισμικού

Για να μπορεί να υλοποιηθεί η προσομοίωση είναι απαραίτητη η εγκατάσταση του MATLAB στον υπολογιστή. Όπως αναφέρθηκε σε προηγούμενο κεφάλαιο, στη συγκεκριμένη διπλωματική χρησιμοποιήθηκε το MATLAB 2017b και το Simulink το οποίο δημιουργήθηκε, τρέχει σε αυτήν την έκδοση. Για να μπορέσει να τρέξει σε κάποια άλλη έκδοση MATLAB δεύτερης γενιάς, θα πρέπει να γίνει εξαγωγή του αρχείου στην επιθυμητή έκδοση.

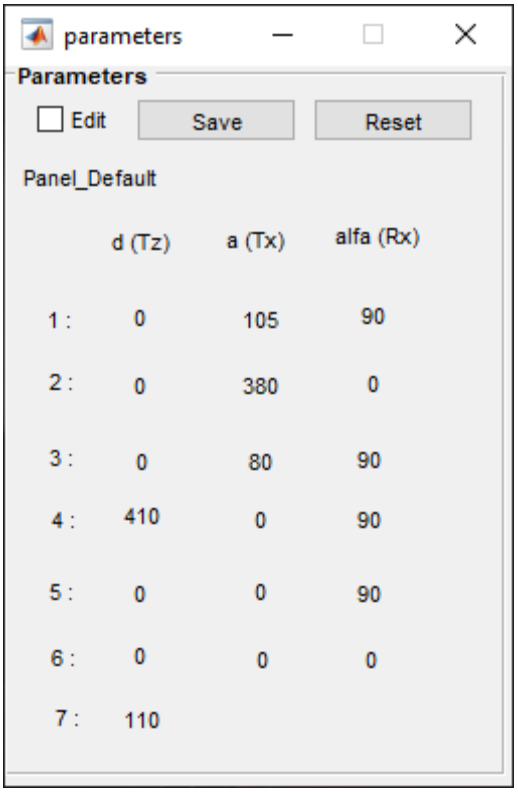
Το λογισμικό που αναπτύχθηκε ονομάζεται Kawa Simulation, όπου Kawa εννοείται Kawasaki. Η εκκίνηση του λογισμικού γίνεται με τη συνάρτηση Kawa_Simulation_Start_Page. Την πρώτη φορά που θα τρέξει απαιτείται λίγος περισσότερος χρόνος, γιατί φορτώνεται ταυτόχρονα το Simulink και το user interface. Το πρώτο πράγμα που εμφανίζεται όταν ανοίξει η φόρμα είναι η αρχική σελίδα τα εφαρμογής, όπως φαίνεται στην εικόνα 27. Πατώντας το πλήκτρο «OK» ξεκινάει η διαδικασία της προσομοίωσης του βραχίονα.



Εικόνα 27: Αρχική Σελίδα Εφαρμογής

Το πρώτο βήμα είναι ο ορισμός των παραμέτρων του βραχίονα. Στη φόρμα που ανοίγει μετά την αρχική σελίδα, όπως φαίνεται στην εικόνα 30, πάνω δεξιά υπάρχει το κουμπί «Parameters», στο οποίο δηλώνονται. Αυτές οι παράμετροι ισχύον για όλες τις μεθόδους

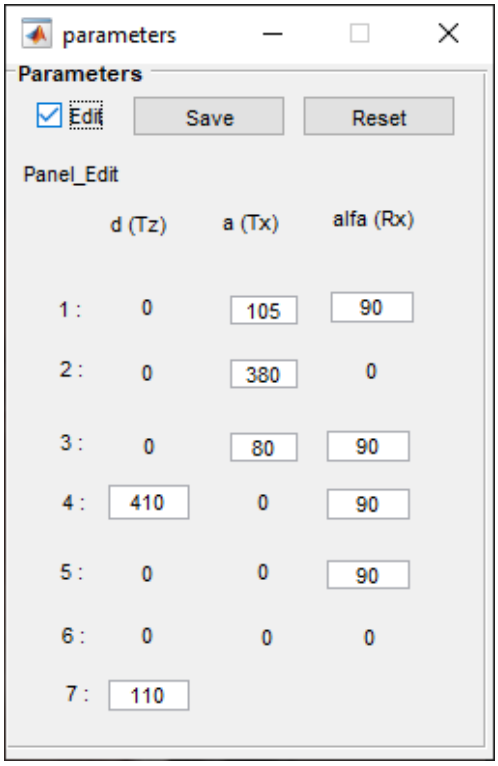
προγραμματισμού και κινηματικών προβλημάτων του προγράμματος. Μόλις πατηθεί το κουμπί, ανοίγει ένα νέο παράθυρο, όπως φαίνεται στην εικόνα 28.



Εικόνα 28: Παράθυρο δηλωμένων παραμέτρων

Εάν ο χρήστης δεν επιθυμεί να αλλάξει τις προκαθορισμένες τιμές, το αφήνει όπως είναι και κλείνει το παράθυρο. Το πρόγραμμα παίρνει απευθείας τις προκαθορισμένες τιμές του βραχίονα. Διαφορετικά, μπορεί να πατήσει το edit (επεξεργασία). Τότε, θα ελευθερωθούν τα κελιά, όπως παρουσιάζεται στην εικόνα 29, που είναι εφικτό να πάρουν άλλες τιμές, ώστε να μπορέσει ο χρήστης να βάλει αυτές που χρειάζεται. Παρατηρείται πως δεν είναι όλα τα κελιά προς επεξεργασία. Αυτό συμβαίνει για να μην είναι εφικτή η αλλαγή της δομής του βραχίονα, παρά μόνο οι διαστάσεις των μελών και ορισμένες σχετικές τοποθετήσεις. Συνεπώς, η ελευθερία

τροποποίησης των παραμέτρων επιτρέπει τη διαχείριση ενός άλλου βραχίονα αλλά της ίδιας μορφής (οικογένειας) σειριακών μηχανισμών.



Εικόνα 29: Παράθυρο δήλωσης νέων παραμέτρων

Σε περίπτωση που έχουν γίνει αλλαγές στις παραμέτρους και είναι επιθυμητό να ξαναγυρίσουν στις προκαθορισμένες, αρκεί να πατηθεί το κουμπί reset.

6.2 Ευθύς Χειρισμός

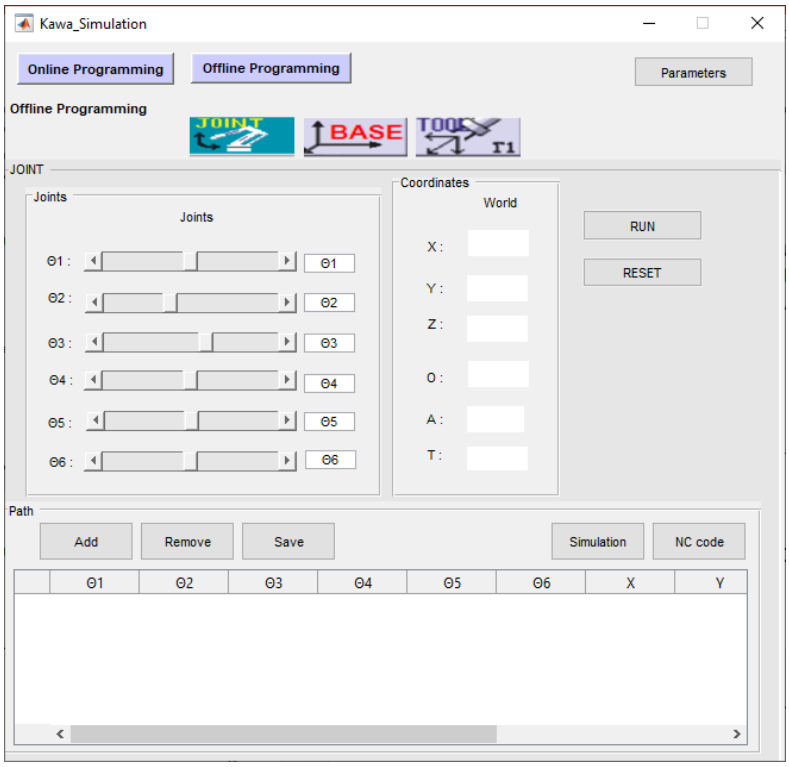
Σε αυτήν την ενότητα θα εξηγηθεί ο τρόπος χρήσης του ευθέως χειρισμού. Στη φόρμα που εμφανίζεται στην εικόνα 30 υπάρχει η επιλογή του τρόπου προγραμματισμού. Εκεί μπορεί να επιλεχθεί αν θα γίνει online ή offline προγραμματισμού. Ο ευθύς χειρισμός γίνεται στο πλήκτρο JOINT, του οποίου η εικόνα είναι από το χειριστήριο του βραχίονα που μελετάται.

6.2.1 Ευθύς Χειρισμός Offline Προγραμματισμού

Η επίλυση του ευθέως κινηματικού γίνεται στη σελίδα JOINT. Ο χρήστης θέτει την επιθυμητή τιμή της κάθε γωνίας σε μοίρες στο κελί που αντιστοιχεί στην κάθε γωνία και πατάει run. Αμέσως, αναγράφονται τα αποτελέσματα X,Y,Z,O,A,T στα άδεια κελιά, στον πίνακα coordinates, όπως επίσης γίνεται η προσομοίωση για τη δεδομένη θέση του βραχίονα στο περιβάλλον του Mechanical Explorer. Εάν ο χρήστης επιθυμεί να αλλάζει τις τιμές των γωνιών με τις μπάρες κυλίσεως (sliders), το αποτέλεσμα αλλάζει αυτόματα, χωρίς να χρειάζεται το κουμπί run. Για μεγαλύτερη ακρίβεια, υπάρχουν τα βελάκια. Σε αυτήν την περίπτωση, χρειάζεται μία

αναμονή μεταξύ των αλλαγών στα sliders, αφού ο αλγόριθμος κάνει τις πράξεις και στέλνει τα αποτελέσματα απευθείας στην προσομοίωση. Με το κουμπί «reset» γίνονται όλες οι τιμές μηδενικές και ο βραχίονας επανέρχεται στην αρχική του θέση.

Τα sliders έχουν οριστεί έτσι, ώστε να μην μπορούν να βγουν εκτός των ορίων των γωνιών που έχουν οριστεί από τον κατασκευαστή (πίνακας 4). Εάν οριστεί γωνία εκτός ορίων στα κενά, τότε το MATLAB θα βγάλει error και δε θα μπορέσει να γίνει η προσομοίωση. Στην περίπτωση που οριστεί μέσα στο κελί κάποιας γωνίας μία τιμή που είναι εκτός πεδίου ορισμού, τότε μετατρέπεται αυτόματα στο ανώτατο ή στο κατώτερο άκρο, αναλόγως με την τιμή που είναι πιο κοντά.



Εικόνα 30: Offline Προγραμματισμός ευθέως κινηματικού

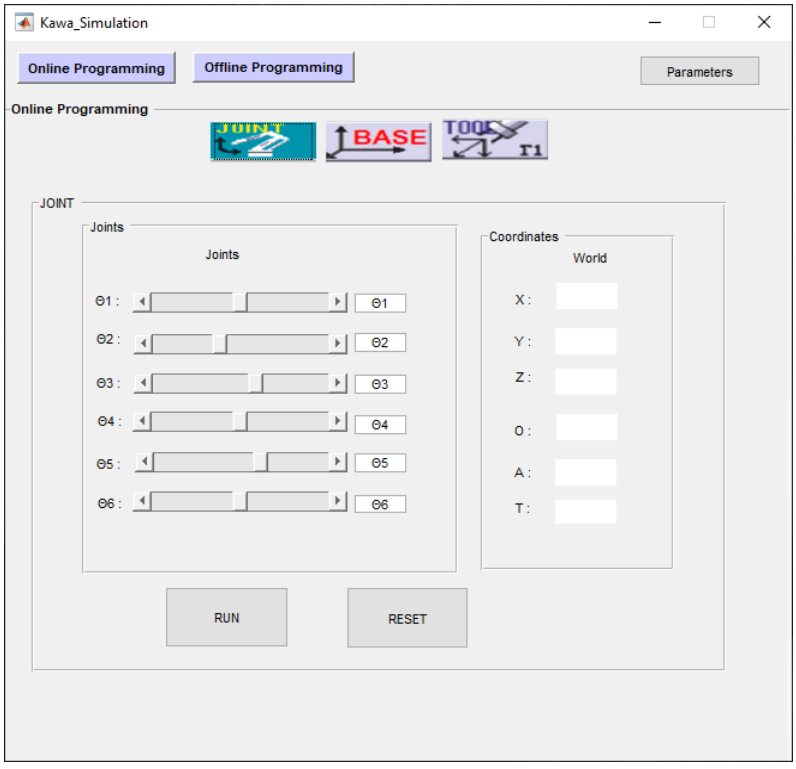
Στον πίνακα Path μπορούν να αποθηκευτούν ή να διαγραφούν θέσεις του βραχίονα. Πιο συγκεκριμένα, υπάρχει η δυνατότητα αποθήκευσης μίας θέσης με το πλήκτρο «Add». Εάν ο χρήστης αλλάξει γνώμη, επιθυμώντας να διαγράψει μία θέση, μπορεί να γίνει με το πλήκτρο «Remove». Όταν καταχωρηθούν όλες οι θέσεις στον πίνακα, με το πλήκτρο «Save» αποθηκεύονται σε ένα αρχείο στο φάκελο του προγράμματος.

Αφού υλοποιηθούν τα παραπάνω, μπορεί να γίνει προσομοίωση των διαδοχικών θέσεων με το «Simulation», όπως, επίσης, να γίνει εξαγωγή του NC κώδικα για τις παραπάνω θέσεις. Το τελευταίο, θα εξηγηθεί περαιτέρω σε επόμενη ενότητα.

6.2.2 Ευθύς Χειρισμός Online Προγραμματισμού

Ο online προγραμματισμός του ευθέως δε διαφέρει και πολύ από τον offline. καθώς και στον online προγραμματισμό, θέτονται οι επιθυμητές τιμές των γωνιών στα αντίστοιχα κελιά του πίνακα Joints και με το run εμφανίζονται οι συντεταγμένες της θέσης του βραχίονα στον πίνακα Coordinates. Επίσης, τα sliders κυμαίνονται μεταξύ των ανώτερων και των κατώτερων δυνατών τιμών που μπορεί να πάρει η κάθε γωνία, ενώ αν οριστεί σε κάποιο κελί μία μη αποδεκτή τιμή, αλλάζει αυτόματα στο ανώτατο ή το κατώτερο όριο αποδεκτών τιμών. Το «reset» έχει ακριβώς την ίδια λειτουργία, δηλαδή επαναφέρει το βραχίονα στην αρχική του θέση, και θέτει όλες τις τιμές των γωνιών στο μηδέν (0).

Στον online προγραμματισμό δε δίνεται η δυνατότητα εξαγωγής NC κώδικα ή αποθήκευσης τιμών. Η μόνη λειτουργία του είναι η προσομοίωση της εκάστοτε θέσης που έχει ορίσει ο χρήστης, βάσει των γωνιών.

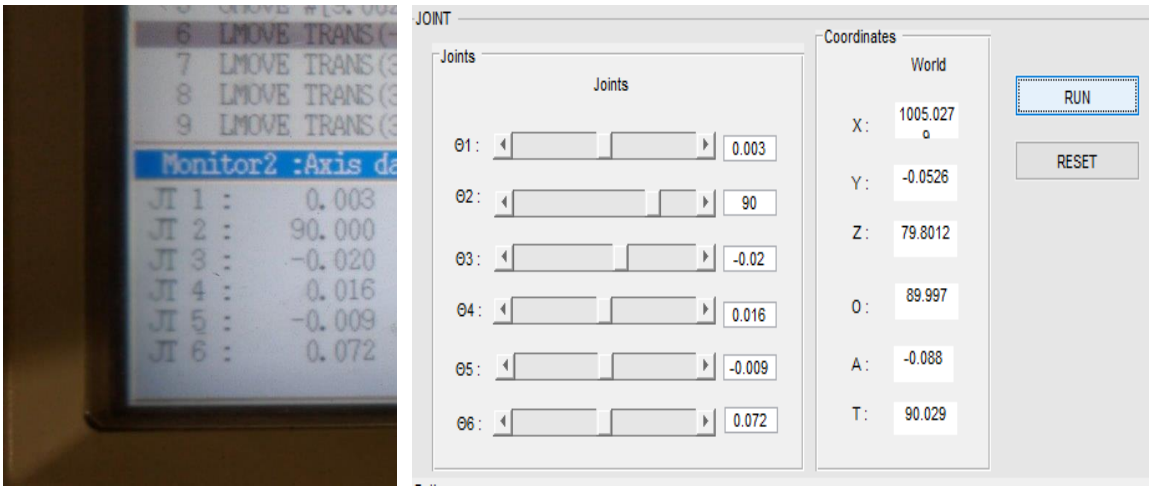


Εικόνα 31: Online Προγραμματισμός ευθέως κινηματικού

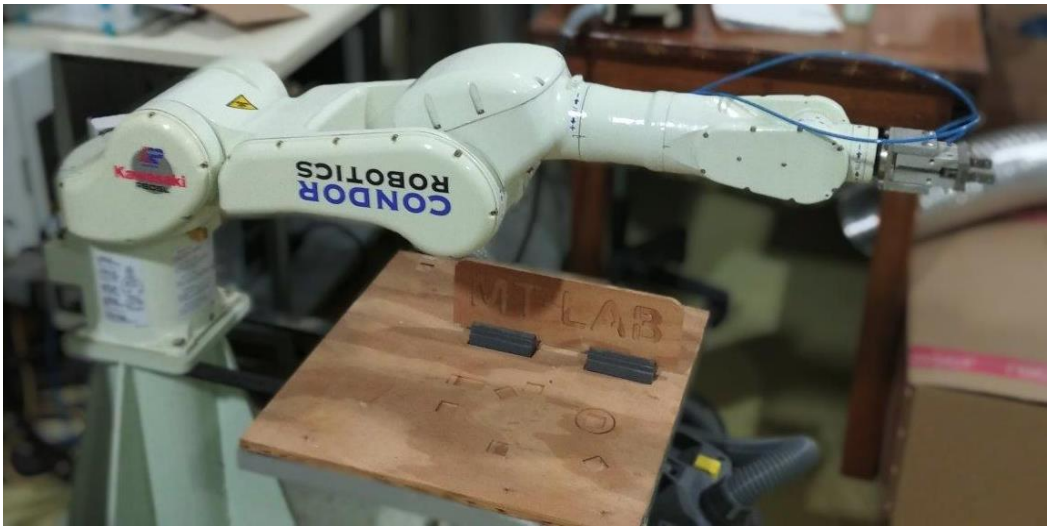
6.2.3 Παραδείγματα του Ευθέως Χειρισμού

Παρακάτω παρουσιάζονται, διαδοχικά, τέσσερις θέσεις του βραχίονα σε σύγκριση με την προσομοίωσή του. Αρχικά, φαίνονται το χειριστήριο (Teach Pendant) και το GUI της προσομοίωσης και, ύστερα, το πως αντιδράει ο βραχίονας σε κάθε περίπτωση.

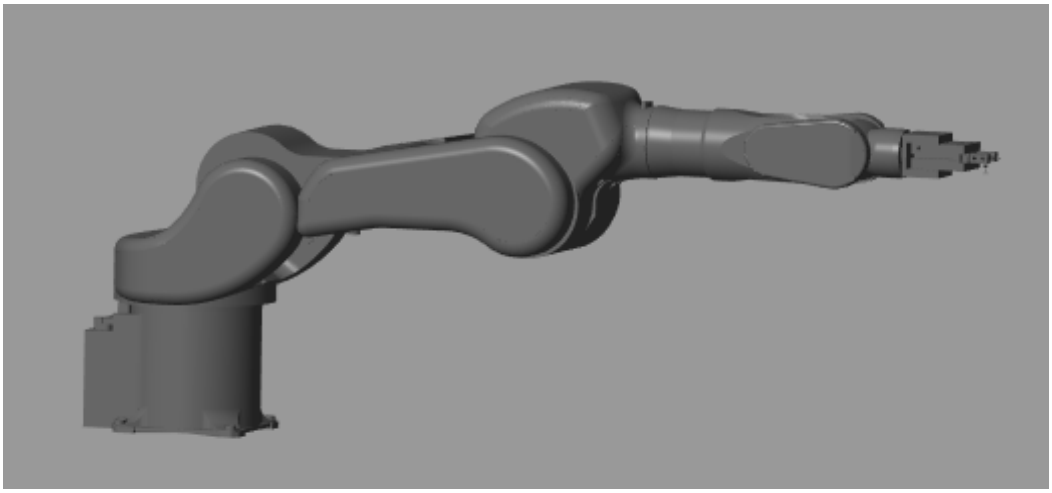
Για τη θέση 1:



Εικόνα 32: Χειριστήριο και GUI θέσης 1



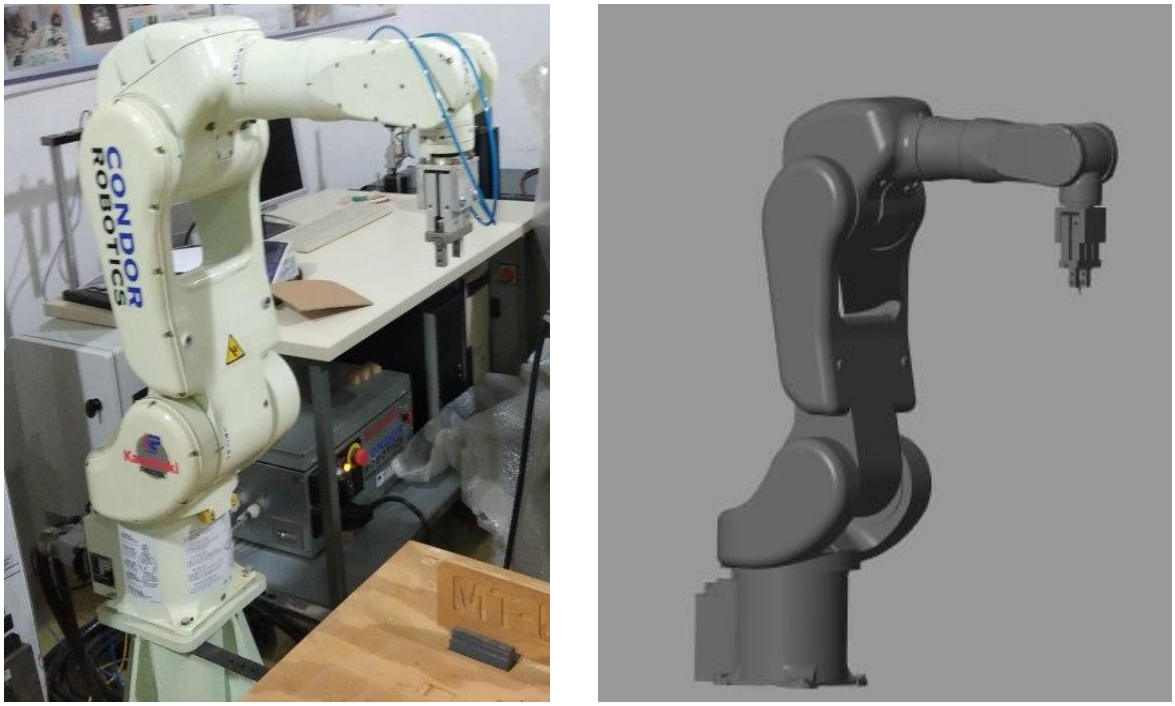
Εικόνα 33: Ο Βραχίονας στη θέση 1



Εικόνα 34: Η προσομοίωση του βραχίονα στη θέση 1

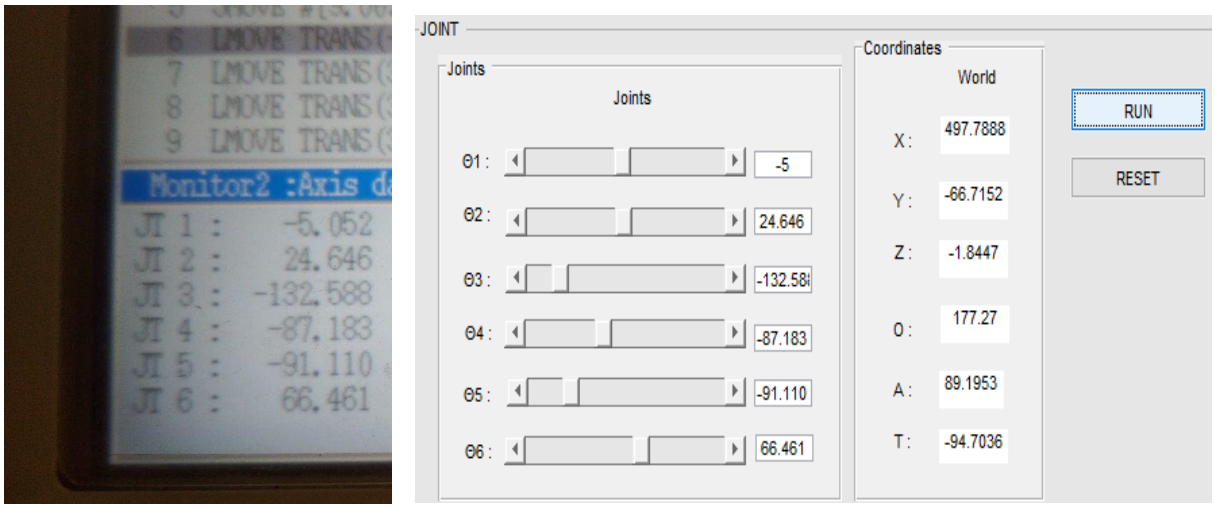
Για τη θέση 2:

Εικόνα 35: Χειριστήριο και GUI θέσης 2



Εικόνα 36: Ο βραχίονας στη θέση 2 και η προσομοίωσή του

Για τη θέση 3:



Εικόνα 37: Χειριστήριο και GUI θέσης 3

Εικόνα 38: Ο βραχίονας στη θέση 3 και η προσομοίωσή του

Για τη θέση 4:

Εικόνα 39: Χειριστήριο και GUI θέσης 4

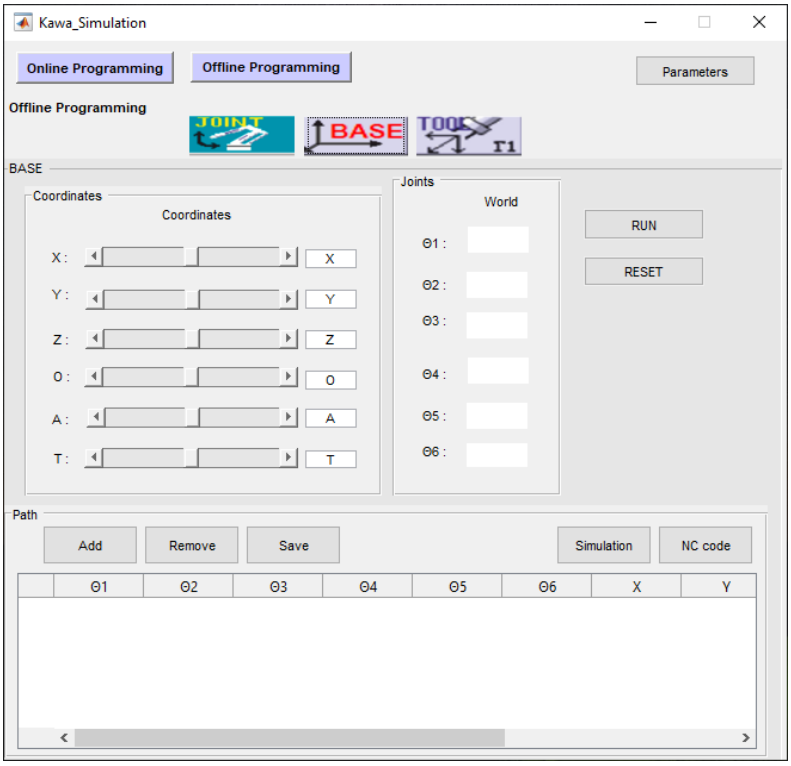
Εικόνα 40: Ο βραχίονας στη θέση 4 και η προσομοίωσή του

6.3 *Αντίστροφος χειρισμός*

Στην παρούσα ενότητα θα αναλυθεί ο τρόπος χρήσης του αντίστροφου χειρισμού. Στη φόρμα που εμφανίζεται στην εικόνα 30 επιλέγεται το πλήκτρο για τη μέθοδο του προγραμματισμού. Επίσης, με το BASE εμφανίζεται η φόρμα για την επίλυση του αντίστροφου κινηματικού. Η εικόνα στο πλήκτρο είναι, επίσης, από το χειριστήριο του ρομπότ.

6.3.1 *Αντίστροφος Χειρισμός Offline Προγραμματισμού*

Η σελίδα BASE περιλαμβάνει την επίλυση του αντίστροφου κινηματικού προβλήματος. Όπως εξηγήθηκε και παραπάνω, είναι γνωστή η επιθυμητή θέση και ο προσανατολισμός του βραχίονα και δίνονται οι γωνίες των αρθρώσεων. Η χρήση του είναι παρόμοια με του ευθέως κινηματικού. Η κυριότερη διαφορά τους είναι ότι ο χρήστης βάζει τα δεδομένα του στον πίνακα Coordinates και με το run λαμβάνει τα αποτελέσματα στον πίνακα Joints. Η χρήση των sliders απαιτούνε περισσότερο χρόνο για να γίνει η προσομοίωση, αφού η προσομοίωση γίνεται απευθείας. Επίσης, τα X,Y,Z,O,A,T δεν έχουν κάποιο πεδίο ορισμού, οπότε ο χρήστης μπορεί να βάλει όποιες τιμές επιθυμεί. Ωστόσο, εάν το αποτέλεσμα κάποιας γωνίας είναι εκτός ορίων κατασκευαστή (πίνακας 4), τότε βγαίνει σφάλμα (error) και δεν μπορεί να γίνει η προσομοίωση.



Εικόνα 41: Offline Προγραμματισμός αντίστροφου κινηματικού

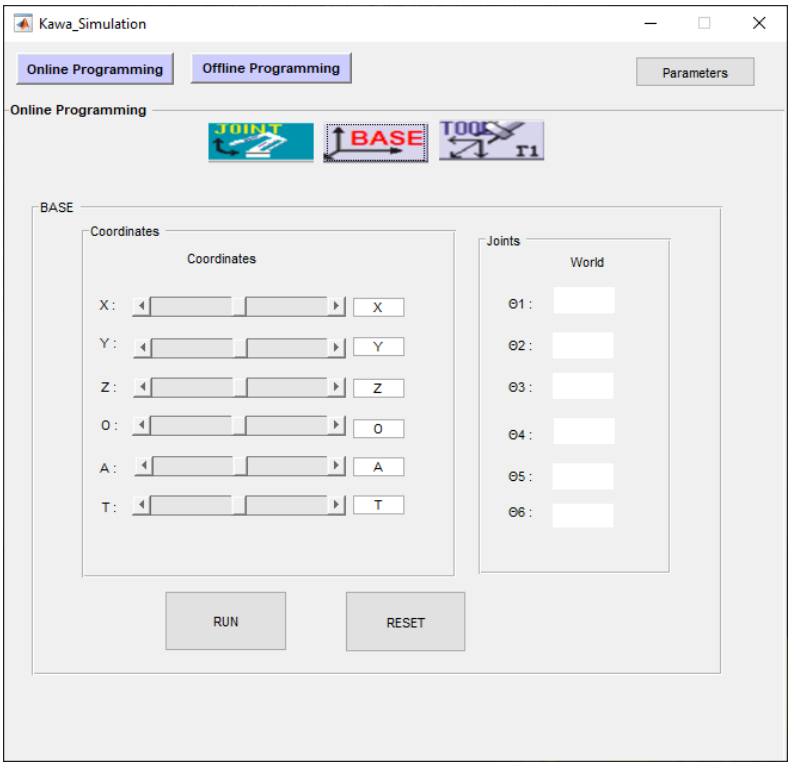
Η προσομοίωση του βραχίονα γίνεται με βάση τα αποτελέσματα των γωνιών. Το πλήκτρο «reset», όπως και ο πίνακας Path έχουν ακριβώς την ίδια λειτουργία με το ευθύ κινηματικό πρόβλημα.

6.3.2 *Αντίστροφος Χειρισμός Online Προγραμματισμού*

Η χρήση του αντίστροφου χειρισμού για τον online προγραμματισμού είναι ακριβώς ίδιος με τον offline. Εισάγονται τα δεδομένα της θέσης στον πίνακα Coordinates και εμφανίζονται τα αποτελέσματα των γωνιών στον πίνακα Joints με το run. Στην περίπτωση χρήσης των sliders

απαιτείται και εδώ περισσότερος χρόνος για την προσομοίωση, η οποία γίνεται αυτόματα, κάθε φορά που πατιέται κάποιο βέλος ή γίνεται χρήση κάποιας μπάρας.

Με το reset γίνεται επαναφορά των τιμών στο μηδέν και ο βραχίονας επιστρέφει στην αρχική του θέση.

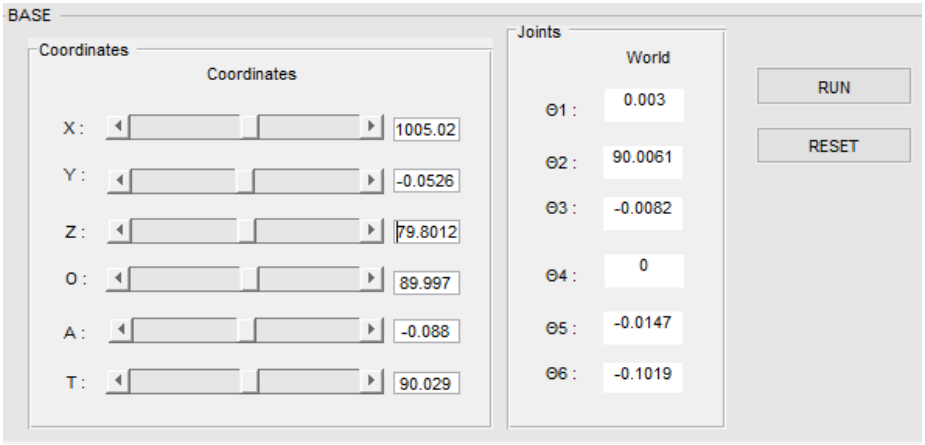


Εικόνα 42: Online Προγραμματισμός αντίστροφου κινηματικού

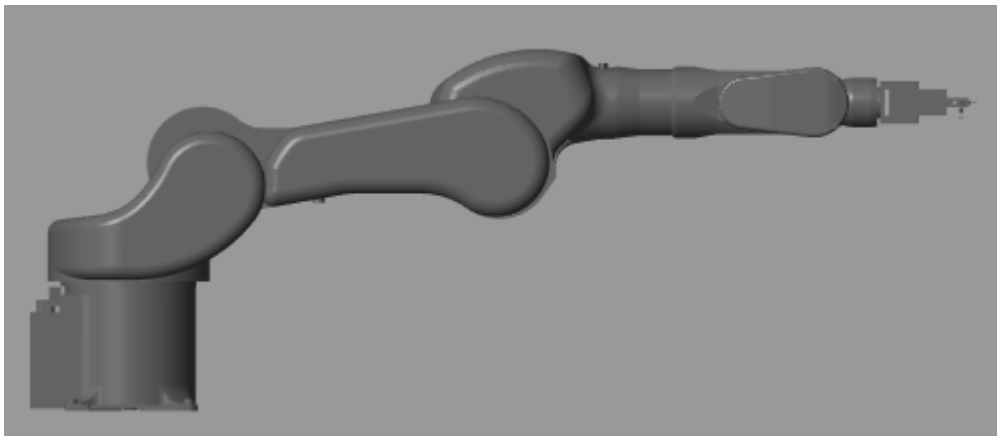
6.3.3 *Παραδείγματα Του Αντίστροφου Χειρισμού*

Στα παρακάτω παραδείγματα, έχουν εισαχθεί ως δεδομένα τα αποτελέσματα του ευθέως χειρισμού. Όπως εξηγήθηκε σε προηγούμενο κεφάλαιο, οι γωνίες θ_2 και θ_3 έχουν δύο (2) λύσεις. Γι' αυτόν το λόγο, υπάρχει πιθανότητα να μην είναι ίδιες από το ευθύ στο αντίστροφο. Παρ 'όλα αυτά, η θέση του τελικού σημείου του βραχίονα θα είναι το ίδιο.

Για τη θέση 1:

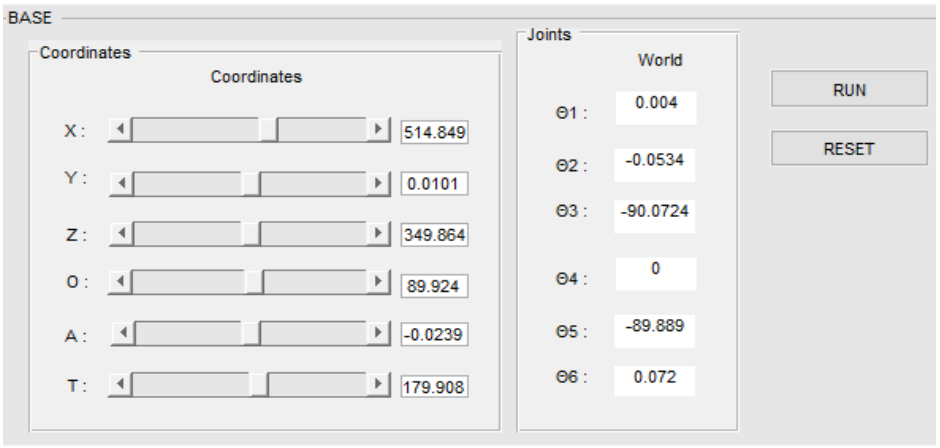


Εικόνα 43: GUI θέσης 1 αντίστροφου χειρισμού



Εικόνα 44: Ο βραχίονας στη θέση 1

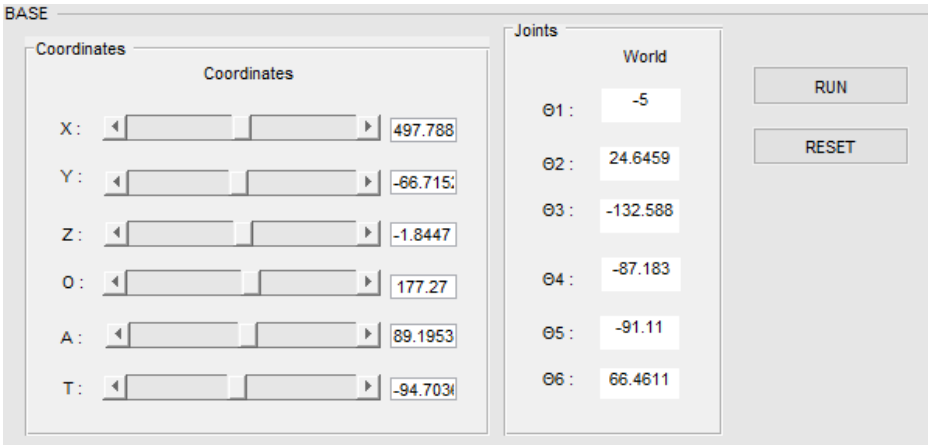
Για τη θέση 2:



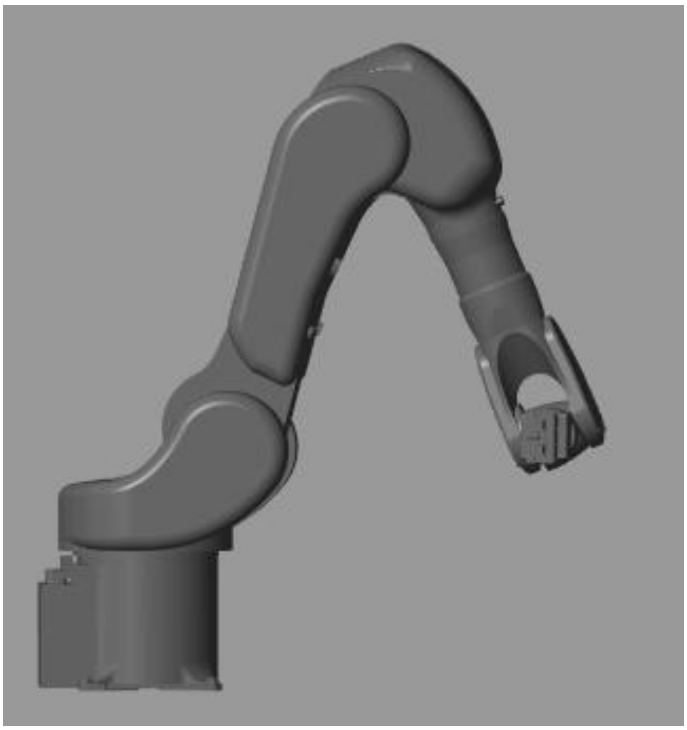
Εικόνα 45: GUI θέσης 2 αντίστροφου χειρισμού

Εικόνα 46: Ο βραχίονας στη θέση 2

Για τη θέση 3:

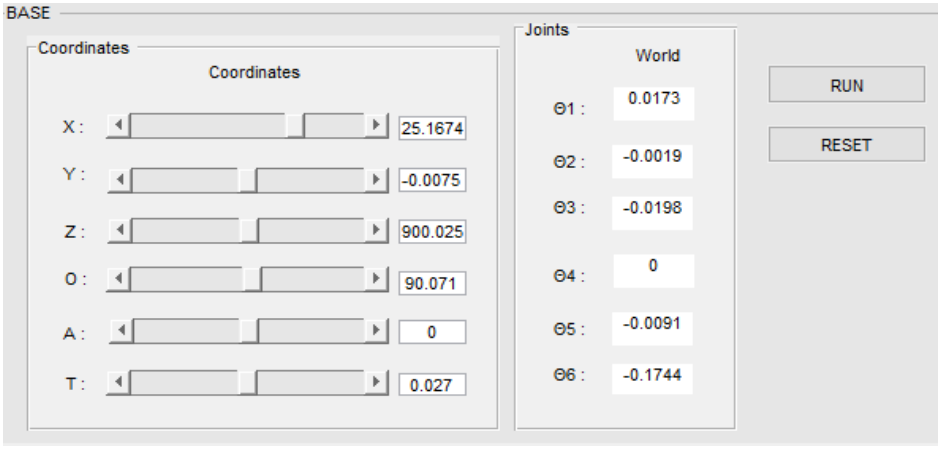


Εικόνα 47: GUI θέσης 3 αντίστροφου χειρισμού

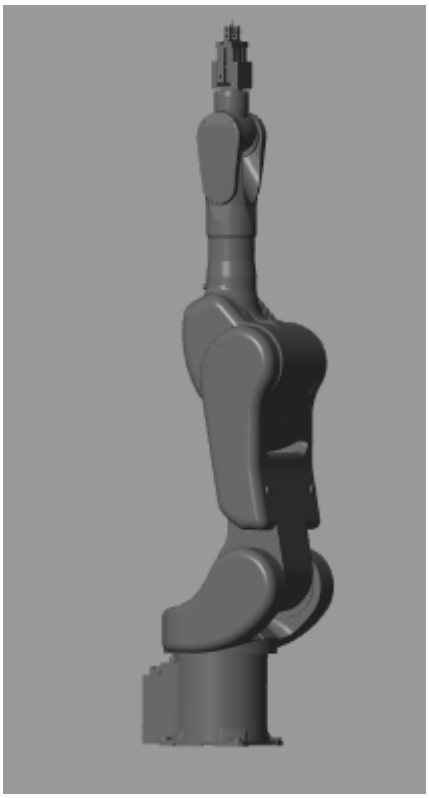


Εικόνα 48: Ο βραχίονας στη θέση 3

Για τη θέση 4:



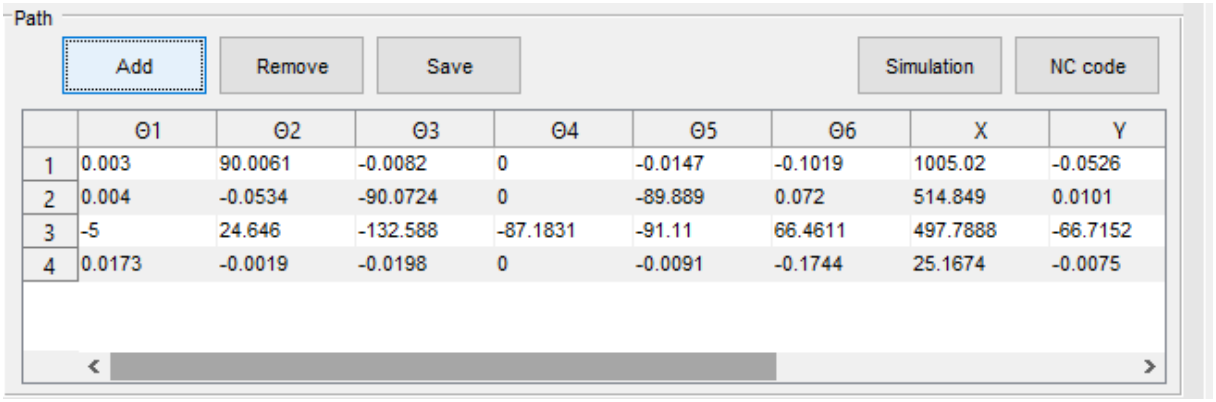
Εικόνα 49: GUI θέσης 4 αντίστροφου χειρισμού



Εικόνα 50: Ο βραχίονας στη θέση 4

6.4 Εξαγωγή NC κώδικα

Όπως αναφέρθηκε σε προηγούμενο κεφάλαιο, στον πίνακα Path αποθηκεύονται οι θέσεις που επιθυμεί ο χρήστης να προσομοιώσει. Στην εικόνα 51 αποθηκεύτηκαν οι τρεις θέσεις που χρησιμοποιήθηκαν στα παραδείγματα της προηγούμενης ενότητας. Για την προσθήκη των θέσεων χρησιμοποιείται το πλήκτρο «Add», ενώ για την αφαίρεση κάποιας το «Remove». Με το «Save» αποθηκεύεται ο πίνακας και αποθηκεύεται ως αρχείο στο φάκελο που είναι το λογισμικό.



	Θ1	Θ2	Θ3	Θ4	Θ5	Θ6	X	Y
1	0.003	90.0061	-0.0082	0	-0.0147	-0.1019	1005.02	-0.0526
2	0.004	-0.0534	-90.0724	0	-89.889	0.072	514.849	0.0101
3	-5	24.646	-132.588	-87.1831	-91.11	66.4611	497.7888	-66.7152
4	0.0173	-0.0019	-0.0198	0	-0.0091	-0.1744	25.1674	-0.0075

Εικόνα 51: Αποθήκευση θέσεων στον πίνακα Path

Η προσομοίωση των θέσεων γίνονται διαδοχικά με το πλήκτρο «Simulation». Τέλος, με το «NC code» εξάγεται ο κώδικας NC και αποθηκεύεται σε ένα .txt αρχείο μέσα στο φάκελο του λογισμικού.

NC_Code - Σημειωματάριο

Αρχείο Επεξεργασία Μορφή Προβολή Βοήθεια

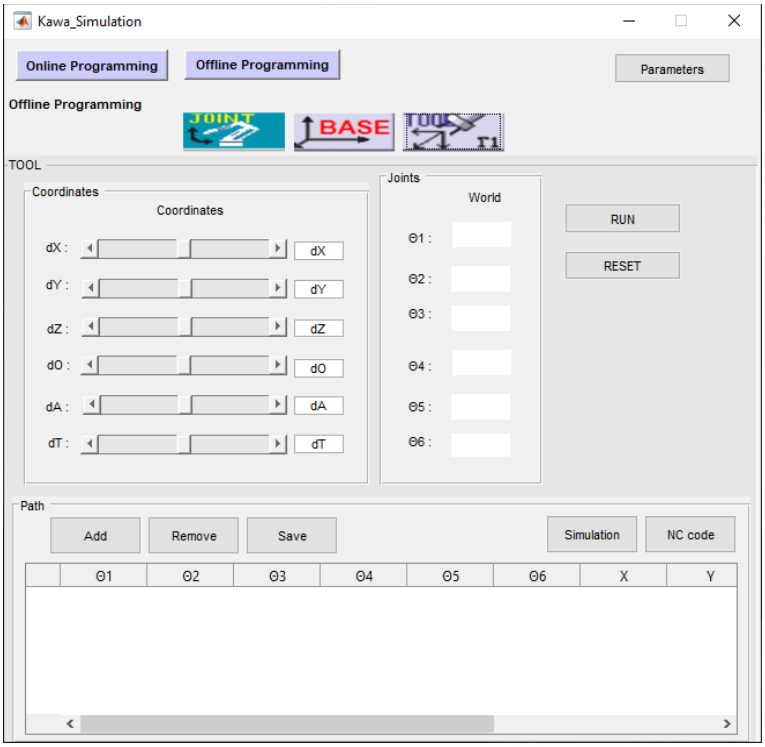
```
.PROGRAM KAWA_SIMULATION
JOINT SPEED1 ACCU2 TIMER0 TOOL1 WORK0 CLAMP1 (OFF,0,0,o) OX= WX= #
[0.00 , 90.01 , -0.01, 0.00, -0.01, -0.10]
JOINT SPEED1 ACCU2 TIMER0 TOOL1 WORK0 CLAMP1 (OFF,0,0,o) OX= WX= #
[0.00 , -0.05 , -90.07, 0.00, -89.89, 0.07]
JOINT SPEED1 ACCU2 TIMER0 TOOL1 WORK0 CLAMP1 (OFF,0,0,o) OX= WX= #
[-5.00 , 24.65 , -132.59, -87.18, -91.11, 66.46]
JOINT SPEED1 ACCU2 TIMER0 TOOL1 WORK0 CLAMP1 (OFF,0,0,o) OX= WX= #
[0.02 , -0.00 , -0.02, 0.00, -0.01, -0.17]
.END
```

Εικόνα 52: Εξαγωγή NC κώδικα

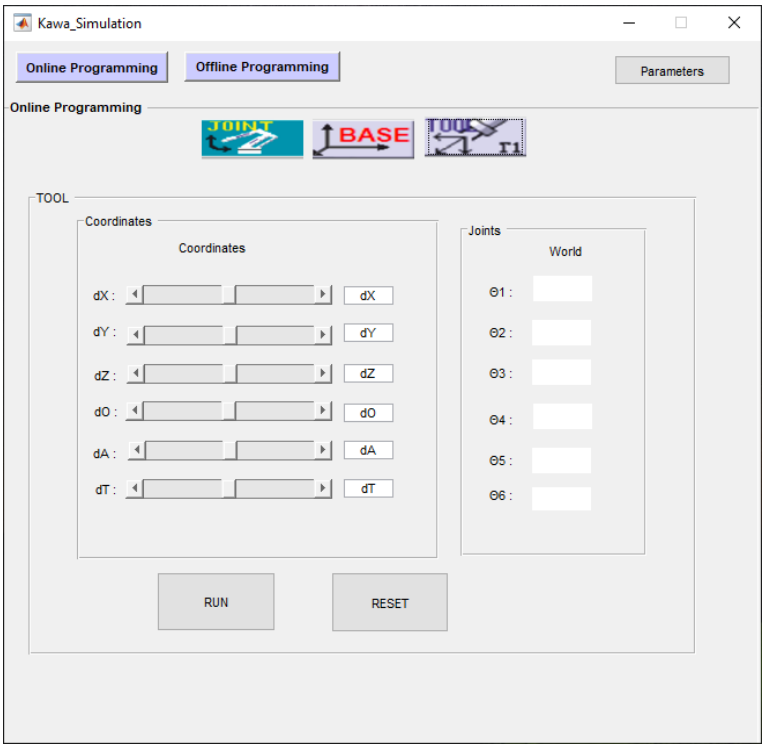
6.5 *Αντίστροφος χειρισμός ως προς την προηγούμενη θέση*

Ο αντίστροφος χειρισμός ως προς την προηγούμενη θέση γίνεται με το πλήκτρο TOOL. Η λογική του είναι παρόμοια με το πλήκτρο BASE, μόνο που ως δεδομένο εισάγεται θέση του τελικού σημείου ως προς το προηγούμενο.

Αυτή η περίπτωση σχεδιάστηκε μόνο γραφικά, αφού δεν ήταν κομμάτι της παρούσας εργασίας. Υπάρχουν όλες οι συναρτήσεις για τη λειτουργία του, όμως δεν έχει γίνει η απαραίτητη επίλυση του μαθηματικού μοντέλου που χρειάζεται το πρόβλημα.



Εικόνα 53: Αντίστροφος χειρισμός ως προς την προηγούμενη θέση offline προγραμματισμού



Εικόνα 54: Αντίστροφος χειρισμός ως προς την προηγούμενη θέση online προγραμματισμού

7. Συμπεράσματα – Προτάσεις Βελτίωσης

Η ολοκλήρωση της παρούσας διπλωματικής εργασίας, που συνδυάζει τη μαθηματική επίλυση του κινηματικού προβλήματος ενός σειριακού ρομποτικού βραχίονα έξι βαθμών ελευθερίας, τον online και offline προγραμματισμό του και την προσομοίωση των κινήσεών του, οδήγησε σε έναν αριθμό συμπερασμάτων.

Αρχικά, αν και αρκετά απαιτητικό, το λογισμικό που δημιουργήθηκε μπορεί να φανεί αρκετά χρήσιμο σε όποιον επιθυμεί να προγραμματίσει τη ροή κινήσεων ενός σειριακού ρομποτικού βραχίονα. Με τη χρήση της προσομοίωσης μπορούν να προσδιοριστούν κινήσεις, χωρίς να είναι απαραίτητη η επαφή με τον πραγματικό βραχίονα. Έτσι, με έναν υπολογιστή και εγκατεστημένο σε αυτόν το MATLAB, μπορεί να γίνει η δουλειά του χρήστη χωρίς να χρειάζεται η φυσική του παρουσία στο ρομπότ.

Επίσης, το περιβάλλον του MATLAB δίνει τη δυνατότητα μαθηματικών υπολογισμών, λύνοντας με ακρίβεια το ευθύ και το αντίστροφο μαθηματικό πρόβλημα. Μπορεί να συνεργαστεί με το Simulink χωρίς πολλές απαιτήσεις. Τέλος, η δημιουργία του γραφικού περιβάλλοντος χρήστη μέσω του ίδιου το MATLAB κάνει τη χρήση του λογισμικού απλή και κατανοητή για οποιονδήποτε θελήσει να το χρησιμοποιήσει.

Υπάρχουν πλεονεκτήματα αλλά και μειονεκτήματα για το παρόν λογισμικό. Ένα πλεονέκτημα του συγκεκριμένου είναι ότι μπορεί να δεχτεί τροποποιήσεις. Αλλάζοντας τις παραμέτρους του, μπορεί να χρησιμοποιηθεί για οποιονδήποτε σειριακό ρομποτικό βραχίονα έξι βαθμών ελευθερίας με παρόμοια γεωμετρία. Επίσης, με λίγες γνώσεις Simulink μπορεί να αλλαχτεί το εργαλείο από αρπάγη σε κάποιο άλλο. Τα μειονεκτήματα του είναι η απαραίτητη ύπαρξη του περιβάλλοντος MATLAB, όπως επίσης και ο μεγάλος χρόνος εκκίνησης του λογισμικού.

Η εργασία μπορεί να εξελιχθεί παραπάνω. Μία πρόταση είναι το λογισμικό να γίνει αυτόνομο και να μη χρειάζεται το περιβάλλον του MATLAB για να λειτουργήσει. Επίσης, μπορούν να συγγραφούν οι συναρτήσεις του πεδίου TOOL στον online και στον offline προγραμματισμό, ώστε να λειτουργεί και εκείνο.

8. Βιβλιογραφία

[1] **Piltan F., Taghizadegan A.,Sulaiman N. B.**, Modeling and Control of Four Degrees of Freedom Surgical Robot Manipulator Using MATLAB/SIMULINK, International Journal of Hybrid Information Technology, Vol.8, No.11 (2015), pp.47-78.

[2] WebSite - www.robodk.com,What Is the Best Way to Program a Robot?, 2018.

[3] **Craig J. John,** Εισαγωγή στη ρομποτική. Μηχανική και αυτόματος έλεγχος, Βιβλίο,Εκδόσεις Τζιόλα, 2009.

[4] WebSite - www.sciencefriday.com, The Origin Of The Word 'Robot', 2011.

[5] WebSite - www.wiki.nus.edu.sg, History of robots, 2008.

[6] WebSite - www.universal-robots.com, Cobots In The Automotive Industry – Automotive Manufacturing, 2018.

[7] WebSite - www.galileo.org, Introduction To Robots.

[8] WebSite - www.arduinobots.wordpress.com, Είδη ρομπότ.

[9] WebSite - www.electronicshub.org, Types of Robots, 2018.

[10] **Siciliano, B., Sciavicco, L., Villani, L., Oriolo, G.**, Robotics, Modelling, Planning and Control, 2009, Springer-Verlag London Limited, Pages 58-65, Pages 105-113, Pages 147-148.

[11] **Radavelli L., Simoni R., De Pieri E. R., Martins D.**, A Comparative Study Of The Kinematics Of Robots Manipulators By Denavit-Hartenberg And Dual Quaternion,

[12] **Εμίρης Δ. Μ., Κουλουριώτης Δ. Ε.**, Ρομποτική, Βιβλίο, 3η Έκδοση, 5η Ανατύπωση, 2015.

[13] **Σαγρής Δ., Μπαλουκτσής Α.,** Σημειώσεις Μαθήματος «Εισαγωγή Στη Ρομποτική Και Στα Αυτόματα Συστήματα», ΠΜΣ Ρομποτικής, 2019.

[14] WebSite - www.blog.robotiq.com**,** What Are the Different Programming Methods for Robots?, 2016.

[15] WebSite - www.automationprimer.com, Robot Programming, 2012.

[16] WebSite - www.robotics.kawasaki.com, RS005L Robot.

[17] **Πλιάκος Β.,** Off-Line Προγραμματισμός Βιομηχανικού Βραχίονα Με Έξι Αρθρώσεις Περιστροφής Kawasaki RS-005L, Διπλωματική εργασία, 2015.

[18] WebSite - www.disigma.gr, Εισαγωγή στην Επιχειρησιακή Έρευνα, Βιβλίο, Β' τόμος, Κεφ. 7, 2010.

[19] WebSite - www.anylogic.com, Why use simulation modeling?.

[20] WebSite - www.solidworks.com, SolidWorks 3D CAD.

[21] WebSite - www.mathworks.com, What is MATLAB?.

[22] WebSite - www.eng.libretexts.org, What is MATLAB?, 2020.

[23] WebSite - www.mathworks.com, Simulink for System Modeling and Simulation.

[24] WebSite - www.mathworks.com, Get Started with Simscape Multibody.

[25] WebSite - www.mathworks.com, Mechanics Explorer.

9. Παράρτημα

```
%function to calculate forward kinematics for both online and offline
%programming
function [x,y,z,o,a,t] = calculate_forward(t1,t2,t3,t4,t5,t6)

%read the a,d,alfa values from the function the_param() which reads the
%values from the gui

[a,d,alfa]=the_param();
a1 = a(1,1); a2 = a(1,2); a3 = a(1,3); a4=a(1,4); a5 = a(1,5); a6=a(1,6);
d1 = d(1,1); d2=d(1,2); d3=d(1,3); d4=d(1,4); d5=d(1,5);d6=d(1,6);
d7=d(1,7);
alfa1 = alfa(1,1); alfa2 = alfa(1,2); alfa3 = alfa(1,3);
alfa4 = alfa(1,4);alfa5 = alfa(1,5); alfa6 = alfa(1,6);

A01 = [cos(t1), -cos(alfa1)*sin(t1), sin(alfa1)*sin(t1), a1*cos(t1)
    sin(t1), cos(alfa1)*cos(t1), -sin(alfa1)*cos(t1), a1*sin(t1)
    0, sin(alfa1), cos(alfa1), d1
    0, 0, 0, 1];

A12 = [cos(t2), -cos(alfa2)*sin(t2), sin(alfa2)*sin(t2), a2*cos(t2)
    sin(t2), cos(alfa2)*cos(t2), -sin(alfa2)*cos(t2), a2*sin(t2)
    0, sin(alfa2), cos(alfa2), d2
    0, 0, 0, 1];

A23 = [cos(t3), -cos(alfa3)*sin(t3), sin(alfa3)*sin(t3), a3*cos(t3)
    sin(t3), cos(alfa3)*cos(t3), -sin(alfa3)*cos(t3), a3*sin(t3)
    0, sin(alfa3), cos(alfa3), d3
    0, 0, 0, 1];

A34 = [cos(t4), -cos(alfa4)*sin(t4), sin(alfa4)*sin(t4), a4*cos(t4)
    sin(t4), cos(alfa4)*cos(t4), -sin(alfa4)*cos(t4), a4*sin(t4)
    0, sin(alfa4), cos(alfa4), d4
    0, 0, 0, 1];

A45 = [cos(t5), -cos(alfa5)*sin(t5), sin(alfa5)*sin(t5), a5*cos(t5)
    sin(t5), cos(alfa5)*cos(t5), -sin(alfa5)*cos(t5), a5*sin(t5)
    0, sin(alfa5), cos(alfa5), d5
    0, 0, 0, 1];

A56 = [cos(t6), -cos(alfa6)*sin(t6), sin(alfa6)*sin(t6), a6*cos(t6)
    sin(t6), cos(alfa6)*cos(t6), -sin(alfa6)*cos(t6), a6*sin(t6)
    0, sin(alfa6), cos(alfa6), d6
    0, 0, 0, 1];

   A67 = [0, 1, 0, 0
    -1, 0, 0, 0
    0, 0, 1, d7
    0, 0, 0, 1];

A02 = A01*A12; A03 = A02*A23; A04 = A03*A34; A05 = A04*A45; A06 = A05*A56;
A07 = A06*A67;
x = A07(1,4);
y = A07(2,4);
z = A07(3,4);

if A07(3,1)==1
    a = 270;
    o = atan2d(-A07(1,2), A07(2,2));
```

```
    t = 0;
elseif A07(3,1)==-1
    a = 90;
    o = -atan2d(A07(1,2),A07(2,2));
    t = 0;
else
   a = atan2d(-A07(3,1), sqrt(A07(1,1)^2+A07(2,1)^2));
   o = atan2d(A07(2,1),A07(1,1));
   t = atan2d(A07(3,2),A07(3,3));
end

end

%%%%%%%%%%%%%%%%
```

```
%function to calculate inverse kinematics for both online and offline
%programming
function [t1,t2,t3,t4,t5,t6] = calculate_inverse(px,py,pz,og,ag,tg)

pi = 3.14159265359;
%read the a,d,alfa values from the function the_param() which reads the
%values from the gui
[a,d,alfa]=the_param();

a1 = a(1,1); a2 = a(1,2); a3 = a(1,3);
d4=d(1,4);d7=d(1,7);

%RPY
a07(1,1) = cos(og)*cos(ag);
a07(2,1) = cos(ag)*sin(og);
a07(3,1) = -sin(ag);
a07(4,1) = 0;

a07(1,2) = cos(og)*sin(ag)*sin(tg)-cos(tg)*sin(og);
a07(2,2) = sin(og)*sin(ag)*sin(tg)+cos(og)*cos(tg);
a07(3,2) = cos(ag)*sin(tg);
a07(4,2) = 0;

a07(1,3) = sin(og)*sin(tg)+cos(og)*cos(tg)*sin(ag);
a07(2,3) = cos(tg)*sin(og)*sin(ag)-cos(og)*sin(tg);
a07(3,3) = cos(ag)*cos(tg);
a07(4,3) = 0;

a07(1,4) = px;
a07(2,4) = py;
a07(3,4) = pz;
a07(4,4) = 1.0;

a76 = [0, -1, 0, 0
    1, 0, 0, 0
    0, 0, 1, -d7
    0, 0, 0, 1];

a06= a07*a76;

x = a06(1,4); y = a06(2,4); z=a06(3,4);

% Th1
if (x==0) && (y==0)
```

```
    dht1 = 0;
    rt1 = 0;
else
    rt1 = atan2(y,x);
    dht1 = atan2d(y,x);
end

%Th2
ab = -(2*a2*z);  %A
b2 =  2*a2*(x*cos(rt1)+y*sin(rt1)-a1);  %B
c2 = a3^2 + d4^2 - a2^2 - z^2 - (x*cos(rt1)+y*sin(rt1) - a1)^2; %C
cth21 = ab^2 + b2^2 - c2^2; %denominator

if (cth21 <0)
    dht2 = 1000;
    dht3 = 1000;
    rt2 = 1000;
    rt3 = 1000;
else
    dht2 = atan2d(c2, -sqrt(cth21)) + atan2d(b2,ab);
    rt2 = dht2*pi/180;
end
%Th3
if dht2 ~= 1000
sth31 = x*cos(rt1)+y*sin(rt1) - a1 - a2*cos(rt2);
sth32 = z - a2*sin(rt2);
cth31 = 2*a3^2 + 2*d4^2;

sth3 = (sth31 - ((a3/d4)*(-sth32)) )/ (cth31/d4);
cth3 = (sth31 - ((d4/a3)*(sth32))) / (cth31/a3);

dht3 = atan2d(sth3,cth3) - dht2;
rt3 = dht3*pi/180;

end


if dht2<-45 || dht2>170 || dht3>208 || dht3<-82

    dht2 = dht2 +360;
    rt2 = dht2*pi/180;
    sth31 = x*cos(rt1)+y*sin(rt1) - a1 - a2*cos(rt2);
    sth32 = z - a2*sin(rt2);
    cth31 = 2*a3^2 + 2*d4^2;

    sth3 = (sth31 - ((a3/d4)*(-sth32)) )/ (cth31/d4);
    cth3 = (sth31 - ((d4/a3)*(sth32))) / (cth31/a3);
    dht3 = atan2d(sth3,cth3) -dht2;
    rt3 = dht3 *pi/180;
end
if dht2<-45 || dht2>170 || dht3>208 || dht3<-82
    dht2 = atan2d(c2, sqrt(cth21)) + atan2d(b2,ab);
    rt2 = dht2*pi/180;

%Th3
if dht2 ~= 1000
sth31 = x*cos(rt1)+y*sin(rt1) - a1 - a2*cos(rt2);
sth32 = z - a2*sin(rt2);
cth31 = 2*a3^2 + 2*d4^2;

sth3 = (sth31 - ((a3/d4)*(-sth32)) )/ (cth31/d4);
cth3 = (sth31 - ((d4/a3)*(sth32))) / (cth31/a3);
```

```
dht3 = atan2d(sth3,cth3) - dht2;
rt3 = dht3*pi/180;
end
end
for k=1:3
    for l=1:3
        base(k,l) = a06(k,l);
    end
end

r03(1,1) = cos(rt1)*cos(rt2 + rt3);
r03(1,2) = sin(rt1);
r03(1,3) = cos(rt1)*sin(rt2+rt3);
r03(2,1) = sin(rt1) * cos(rt2 + rt3);
r03(2,2) = -cos(rt1);
r03(2,3) = sin(rt1)*sin(rt2+rt3);
r03(3,1) = sin(rt2+rt3);
r03(3,2) = 0;
r03(3,3) = - cos(rt2+rt3);

r03inv = transpose(r03);

%R36
r36 = r03inv*base;

az = r36(3,3);
s3 = sqrt(1-az^2);

if (az == 1)
    dht4 = 0;
    dht6 = atan2(r36(2,1),r36(1,1));
    dht5 = 180;
else %I5 =1
    dht5 = atan2d(s3,-az);
    dht4 = atan2d(r36(2,3), r36(1,3));
    dht6 = atan2d(-r36(3,2), r36(3,1));
end

%I5=-1
if dht5>325 || dht5<35
    dht5 = atan2d(-s3,-az);
    dht4 = atan2d(-r36(2,3), -r36(1,3));
    dht6 = atan2d(-(-r36(3,2)), -r36(3,1));
end

if dht5>325 || dht5<35
     dht2 = atan2d(c2, sqrt(cth21)) + atan2d(b2,ab);
     rt2 = dht2*pi/180;
     sth31 = x*cos(rt1)+y*sin(rt1) - a1 - a2*cos(rt2);
     sth32 = z - a2*sin(rt2);
     cth31 = 2*a3^2 + 2*d4^2;

     sth3 = (sth31 - ((a3/d4)*(-sth32)) )/ (cth31/d4);
     cth3 = (sth31 - ((d4/a3)*(sth32))) / (cth31/a3);
     dht3 = atan2d(sth3,cth3) -dht2;
     rt3 = dht3 *pi/180;

     for k=1:3
        for l=1:3
            base(k,l) = a06(k,l);
        end
```

```
    end

r03(1,1) = cos(rt1)*cos(rt2 + rt3);
r03(1,2) = sin(rt1);
r03(1,3) = cos(rt1)*sin(rt2+rt3);
r03(2,1) = sin(rt1) * cos(rt2 + rt3);
r03(2,2) = -cos(rt1);
r03(2,3) = sin(rt1)*sin(rt2+rt3);
r03(3,1) = sin(rt2+rt3);
r03(3,2) = 0;
r03(3,3) = - cos(rt2+rt3);

r03inv = transpose(r03);

%R36
r36 = r03inv*base;

az = r36(3,3);
s3 = sqrt(1-az^2);

    dht5 = atan2d(s3,-az);
    dht4 = atan2d(r36(2,3), r36(1,3));
    dht6 = atan2d(-r36(3,2), r36(3,1));
end

t1 = -dht1;

t2 = 90 - dht2;
if round(t2)==360|| round(t2)==-360
    t2=0;
end
t3 = dht3 - 90;
if round(t3)==360 || round(t3)==-360
    t3=0;
end
t4 = dht4 -180;
if round(t4) == -360
    t4 = 0;
end
if round(t4) == 360
    t4=0;
end
t5 = dht5 - 180;
if round(t5) == -270
    t5 = 90;
end
if round(t5) == 270
    t5 = -90;
end
t6 = dht6;

%Limits
if t1<-180 || t1>180
    f = msgbox('t1 is out of range. -180 <= t1 <= +180.');
end
if t2<-80 || t2>135
    f = msgbox('t2 is out of range. -80 <= t2 <= +135.');
end
if t3<-172 || t3>118
    f = msgbox('t3 is out of range. -172 <= t3 <= +118.');
end
if t4<-360 || t4>360
    f = msgbox('t4 is out of range. -360 <= t4 <= +360.');
```

```
end
if t5<-145 || t5>145
    f = msgbox('t5 is out of range. -145 <= t5 <= +145.');
end
if t6<-360 || t6>360
    f = msgbox('t6 is out of range. -360 <= t6 <= +360.');
end

end
%%%%%%%%%%%%%%%%%%%%
```

```
function [a,d,alfa] = the_param()
S = load('param.mat');

a(1,1) = S.param(1,1);
a(1,2) = S.param(1,2);
a(1,3) = S.param(1,3);
a(1,4) = S.param(1,4);
a(1,5) = S.param(1,5);
a(1,6) = S.param(1,6);
d(1,1) = S.param(2,1);
d(1,2) = S.param(2,2);
d(1,3) = S.param(2,3);
d(1,4) = S.param(2,4);
d(1,5) = S.param(2,5);
d(1,6) = S.param(2,6);
d(1,7) = S.param(2,7);
alfa(1,1) = S.param(3,1);
alfa(1,2) = S.param(3,2);
alfa(1,3) = S.param(3,3);
alfa(1,4) = S.param(3,4);
alfa(1,5) = S.param(3,5);
alfa(1,6) = S.param(3,6);
end
```

```
function varargout =
Kawa_Simulation_Start_Page(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',
mfilename, ...
                   'gui_Singleton',
gui_Singleton, ...
                   'gui_OpeningFcn',
@Kawa_Simulation_Start_Page_OpeningFcn, ...
                   'gui_OutputFcn',
@Kawa_Simulation_Start_Page_OutputFcn, ...
                   'gui_LayoutFcn',  [] ,
...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback =
str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] =
gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
end
%
function
Kawa_Simulation_Start_Page_OpeningFcn(hObjec
t, eventdata, handles, varargin)

filename = ('robotarm.PNG');
a = imread(filename);
```

```
function varargout = Kawa_Simulation(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',
mfilename, ...
                   'gui_Singleton',
gui_Singleton, ...
                   'gui_OpeningFcn',
@Kawa_Simulation_OpeningFcn, ...
                   'gui_OutputFcn',
@Kawa_Simulation_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback =
str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] =
gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
%
function Kawa_Simulation_OpeningFcn(hObject,
eventdata, handles, varargin)
set(handles.uitable1,'Data',cell(0,12))
set(handles.uitable2,'Data',cell(0,12))
set(handles.uitable3,'Data',cell(0,12))

[a,map]=imread('base.png');
I1 = imresize(a, [35 95]);
set(handles.base,'CData',I1);
```

```
axes(handles.axes1);
imshow(a);

handles.output = hObject;
% Update handles structure
guidata(hObject, handles);
end
%
function varargout =
Kawa_Simulation_Start_Page_OutputFcn(hObject
, eventdata, handles)

varargout{1} = handles.output;
end
%
function start_Callback(hObject, eventdata,
handles)

Kawa_Simulation;
close(Kawa_Simulation_Start_Page);
end
%PARAMETERS
function parameters_Callback(hObject,
eventdata, handles)
    parameters
end
%
function varargout = parameters(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',
mfilename, ...
                   'gui_Singleton',
gui_Singleton, ...
                   'gui_OpeningFcn',
@parameters_OpeningFcn, ...
                   'gui_OutputFcn',
@parameters_OutputFcn, ...
                   'gui_LayoutFcn',  [] ,
...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback =
str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] =
gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
end
%
function parameters_OpeningFcn(hObject,
eventdata, handles2, varargin)
handles2.output = hObject;
flag=0;
setappdata(0,'checkbox_value',flag);
% Update handles structure
value = get(handles2.checkbox1, 'Value');
if value == 1
    set(findall(handles2.paneledit, '-
property', 'enable'), 'enable', 'on')
    set(handles2.paneledit, 'visible','on')
    set(findall(handles2.paneldef, '-
property', 'enable'), 'enable', 'off')
    set(handles2.paneldef, 'visible','off')
else
    set(findall(handles2.paneledit, '-
property', 'enable'), 'enable', 'off')
    set(handles2.paneledit, 'visible','off')
    set(findall(handles2.paneldef, '-
property', 'enable'), 'enable', 'on')
    set(handles2.paneldef, 'visible','on')
end

guidata(hObject, handles2);
end
%
```

```
[a,map]=imread('joint.png');
I2 = imresize(a, [35 95]);
set(handles.joint,'CData',I2);

[a,map]=imread('tool.png');
I3 = imresize(a, [35 95]);
set(handles.tool,'CData',I3);

[a,map]=imread('base.png');
I1 = imresize(a, [35 95]);
set(handles.on_base,'CData',I1);

[a,map]=imread('joint.png');
I2 = imresize(a, [35 95]);
set(handles.on_joint,'CData',I2);

[a,map]=imread('tool.png');
I3 = imresize(a, [35 95]);
set(handles.on_tool,'CData',I3);

handles.output = hObject;

open_system('KawaSim');
% Update handles structure
guidata(hObject, handles);

function varargout =
Kawa_Simulation_OutputFcn(hObject, eventdata,
handles)
varargout{1} = handles.output;
%ONLINE PROGRAMMING
function online_program_Callback(hObject,
eventdata, handles)
    set(findall(handles.panel_off, '-
property', 'enable'), 'enable', 'off')
    set(handles.panel_off, 'visible','off')
    set(findall(handles.panel_on, '-property',
'enable'), 'enable', 'on')
    set(handles.panel_on, 'visible','on')
end
%ONLINE JOINT
%online joint functions
function on_joint_Callback(hObject, eventdata,
handles)
set(findall(handles.on_panel_1, '-property',
'enable'), 'enable', 'off')
    set(handles.on_panel_1, 'visible','off')

    set(findall(handles.on_panel_2, '-
property', 'enable'), 'enable', 'on')
    set(handles.on_panel_2, 'visible','on')

    set(findall(handles.on_panel_3, '-
property', 'enable'), 'enable', 'off')
    set(handles.on_panel_3, 'visible','off')
end
%
function on_t1_Callback(hObject, eventdata,
handles)
Value  = get(handles.on_t1, 'Value');
set(handles.on_slid1,'String',
num2str(Value));
end
%
function on_t1_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function on_t2_Callback(hObject, eventdata,
handles)
Value  = get(handles.on_t2, 'Value');
set(handles.on_slid2,'String',
num2str(Value));
end
%
```

```
function varargout =
parameters_OutputFcn(hObject, eventdata,
handles)
varargout{1} = handles.output;
end
%
function checkbox1_Callback(hObject,
eventdata, handles2)
value = get(handles2.checkbox1, 'Value');
if value == 1
    set(findall(handles2.paneledit, '-
property', 'enable'), 'enable', 'on')
    set(handles2.paneledit, 'visible','on')
    set(findall(handles2.paneldef, '-
property', 'enable'), 'enable', 'off')
    set(handles2.paneldef, 'visible','off')

    flag = 1;
    setappdata(0,'checkbox_value',flag);
else
    set(findall(handles2.paneledit, '-
property', 'enable'), 'enable', 'off')
    set(handles2.paneledit, 'visible','off')
    set(findall(handles2.paneldef, '-
property', 'enable'), 'enable', 'on')
    set(handles2.paneldef, 'visible','on')

    flag = 0;
    setappdata(0,'checkbox_value',flag);

end
end

% --- Executes on button press in
resetparam.
function saveparam_Callback(hObject,
eventdata, handles2)

    flag = getappdata(0, 'checkbox_value');
    pi = 3.14159265359;
    if (flag ==1)

        a1_e = str2double(get(handles2.a1_e,
'String'));
        a2_e = str2double(get(handles2.a2_e,
'String'));
        a3_e = str2double(get(handles2.a3_e,
'String'));
        a4_e = str2double(get(handles2.a4_e,
'String'));
        a5_e = str2double(get(handles2.a5_e,
'String'));
        a6_e = str2double(get(handles2.a6_e,
'String'));

        d1_e = str2double(get(handles2.d1_e,
'String'));
        d2_e = str2double(get(handles2.d2_e,
'String'));
        d3_e = str2double(get(handles2.d3_e,
'String'));
        d4_e = str2double(get(handles2.d4_e,
'String'));
        d5_e = str2double(get(handles2.d5_e,
'String'));
        d6_e = str2double(get(handles2.d6_e,
'String'));
        d7_e = str2double(get(handles2.d7_e,
'String'));

        alfa1_e =
str2double(get(handles2.alfa1_e, 'String'))*
(pi/180);
        alfa2_e =
str2double(get(handles2.alfa2_e, 'String'))*
(pi/180) ;
        alfa3_e =
str2double(get(handles2.alfa3_e, 'String'))*
(pi/180);
```

```
function on_t2_CreateFcn(hObject, eventdata,
handles)

if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function on_t3_Callback(hObject, eventdata,
handles)
Value  = get(handles.on_t3, 'Value');
set(handles.on_slid3,'String',
num2str(Value));
end
%
function on_t3_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function on_t4_Callback(hObject, eventdata,
handles)
Value  = get(handles.on_t4, 'Value');
set(handles.on_slid4,'String',
num2str(Value));
end
%
function on_t4_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function on_t5_Callback(hObject, eventdata,
handles)
Value  = get(handles.on_t5, 'Value');
set(handles.on_slid5,'String',
num2str(Value));
end
%
function on_t5_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function on_t6_Callback(hObject, eventdata,
handles)
Value  = get(handles.on_t6, 'Value');
set(handles.on_slid6,'String',
num2str(Value));
end
%
function on_t6_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function on_slid1_Callback(hObject, eventdata,
handles)
set(handles.slid6, 'SliderStep'
,[0.1/(360+360), 2*0.1/(360+360)]);
```

```
        alfa4_e =
str2double(get(handles2.alfa4_e, 'String'))*
(pi/180);
        alfa5_e =
str2double(get(handles2.alfa5_e, 'String'))*
(pi/180);
        alfa6_e =
str2double(get(handles2.alfa6_e, 'String'))*
(pi/180);

        param =
[a1_e,a2_e,a3_e,a4_e,a5_e,a6_e,0;d1_e,d2_e,d
3_e,d4_e,d5_e,d6_e,d7_e;

alfa1_e,alfa2_e,alfa3_e,alfa4_e,alfa5_e,alfa
6_e,0];

    else

        a1_d = str2double(get(handles2.a1_d,
'String'));
        a2_d = str2double(get(handles2.a2_d,
'String'));
        a3_d = str2double(get(handles2.a3_d,
'String'));
        a4_d = str2double(get(handles2.a4_d,
'String'));
        a5_d = str2double(get(handles2.a5_d,
'String'));
        a6_d = str2double(get(handles2.a6_d,
'String'));
        d1_d = str2double(get(handles2.d1_d,
'String'));
        d2_d = str2double(get(handles2.d2_d,
'String'));
        d3_d = str2double(get(handles2.d3_d,
'String'));
        d4_d = str2double(get(handles2.d4_d,
'String'));
        d5_d = str2double(get(handles2.d5_d,
'String'));
        d6_d = str2double(get(handles2.d6_d,
'String'));
        d7_d = str2double(get(handles2.d7_d,
'String'));
        alfa1_d =
str2double(get(handles2.alfa1_d, 'String'))*
(pi/180);
        alfa2_d =
str2double(get(handles2.alfa2_d, 'String'))*
(pi/180);
        alfa3_d =
str2double(get(handles2.alfa3_d, 'String'))*
(pi/180);
        alfa4_d =
str2double(get(handles2.alfa4_d, 'String'))*
(pi/180);
        alfa5_d =
str2double(get(handles2.alfa5_d, 'String'))*
(pi/180);
        alfa6_d =
str2double(get(handles2.alfa6_d, 'String'))*
(pi/180);

        param =
[a1_d,a2_d,a3_d,a4_d,a5_d,a6_d,0;d1_d,d2_d,d
3_d,d4_d,d5_d,d6_d,d7_d;

alfa1_d,alfa2_d,alfa3_d,alfa4_d,alfa5_d,alfa
6_d,0];

    end
    save('param.mat','param');
    clear param
end

 function resetparam_Callback(hObject,
eventdata, handles2)

     set(handles2.checkbox1,'value',0)
st1 = get(handles.on_slid1, 'Value');
set(handles.on_t1,'String', num2str(st1));
t1 = - st1* (pi/180);

st2 = get(handles.on_slid2, 'Value');
set(handles.on_t2,'String', num2str(st2));
t2 = (-st2+90)* (pi/180);

st3 = get(handles.on_slid3, 'Value');
set(handles.on_t3,'String', num2str(st3));
t3 = (st3+90)* (pi/180);

st4 = get(handles.on_slid4, 'Value');
set(handles.on_t4,'String', num2str(st4));
t4 = (st4+180)* (pi/180);

st5 = get(handles.on_slid5, 'Value');
set(handles.on_t5,'String', num2str(st5));
t5 = (st5+180)* (pi/180);

st6 = get(handles.on_slid6, 'Value');
set(handles.on_t6,'String', num2str(st6));
t6 = st6* (pi/180);

[px,py,pz,O,A,T] =
calculate_forward(t1,t2,t3,t4,t5,t6);
set(handles.on_x_px, 'String',
num2str(round(px,4)));
set(handles.on_y_py,
'String',num2str(round(py,4)));
set(handles.on_z_pz, 'String',
num2str(round(pz,4)));
set(handles.on_o_po, 'String',
num2str(round(O,4)));
set(handles.on_a_pa, 'String',
num2str(round(A,4)));
set(handles.on_t_pt, 'String',
num2str(round(T,4)));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(st1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(st2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(st3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(st4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(st5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(st6));
set_param('KawaSim','SimulationCommand','start
');
set_param('KawaSim','SimulationCommand','conti
nue');
set_param('KawaSim','SimulationCommand','pause
');
end
%
function on_slid1_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function on_slid2_Callback(hObject, eventdata,
handles)
set(handles.slid6, 'SliderStep'
,[0.1/(360+360), 2*0.1/(360+360)]);

st1 = get(handles.on_slid1, 'Value');
set(handles.on_t1,'String', num2str(st1));
t1 = - st1* (pi/180);

st2 = get(handles.on_slid2, 'Value');
set(handles.on_t2,'String', num2str(st2));
t2 = (-st2+90)* (pi/180);
```

```
     set(findall(handles2.paneledit, '-
property', 'enable'), 'enable', 'off')
     set(handles2.paneledit,
'visible','off')
     set(findall(handles2.paneldef, '-
property', 'enable'), 'enable', 'on')
     set(handles2.paneldef, 'visible','on')

        a1_d = str2double(get(handles2.a1_d,
'String'));
        a2_d = str2double(get(handles2.a2_d,
'String'));
        a3_d = str2double(get(handles2.a3_d,
'String'));
        a4_d = str2double(get(handles2.a4_d,
'String'));
        a5_d = str2double(get(handles2.a5_d,
'String'));
        a6_d = str2double(get(handles2.a6_d,
'String'));
        d1_d = str2double(get(handles2.d1_d,
'String'));
        d2_d = str2double(get(handles2.d2_d,
'String'));
        d3_d = str2double(get(handles2.d3_d,
'String'));
        d4_d = str2double(get(handles2.d4_d,
'String'));
        d5_d = str2double(get(handles2.d5_d,
'String'));
        d6_d = str2double(get(handles2.d6_d,
'String'));
        d7_d = str2double(get(handles2.d7_d,
'String'));
        alfa1_d =
str2double(get(handles2.alfa1_d, 'String'))*
(pi/180);
        alfa2_d =
str2double(get(handles2.alfa2_d, 'String'))*
(pi/180);
        alfa3_d =
str2double(get(handles2.alfa3_d, 'String'))*
(pi/180);
        alfa4_d =
str2double(get(handles2.alfa4_d, 'String'))*
(pi/180);
        alfa5_d =
str2double(get(handles2.alfa5_d, 'String'))*
(pi/180);
        alfa6_d =
str2double(get(handles2.alfa6_d, 'String'))*
(pi/180);

        param =
[a1_d,a2_d,a3_d,a4_d,a5_d,a6_d,0;d1_d,d2_d,d
3_d,d4_d,d5_d,d6_d,d7_d;

alfa1_d,alfa2_d,alfa3_d,alfa4_d,alfa5_d,alfa
6_d,0];

    save('param.mat','param');
    clear param
 end
 %OFFLINE PROGRAMMING
function offline_program_Callback(hObject,
eventdata, handles)
    set(findall(handles.panel_off, '-
property', 'enable'), 'enable', 'on')
    set(handles.panel_off, 'visible','on')
    set(findall(handles.panel_on, '-
property', 'enable'), 'enable', 'off')
    set(handles.panel_on, 'visible','off')
end
%OFFLINE JOINT
function joint_Callback(hObject, eventdata,
handles)
    set(findall(handles.panel2, '-property',
'enable'), 'enable', 'on')
    set(handles.panel2, 'visible','on')
```

```
st3 = get(handles.on_slid3, 'Value');
set(handles.on_t3,'String', num2str(st3));
t3 = (st3+90)* (pi/180);

st4 = get(handles.on_slid4, 'Value');
set(handles.on_t4,'String', num2str(st4));
t4 = (st4+180)* (pi/180);

st5 = get(handles.on_slid5, 'Value');
set(handles.on_t5,'String', num2str(st5));
t5 = (st5+180)* (pi/180);

st6 = get(handles.on_slid6, 'Value');
set(handles.on_t6,'String', num2str(st6));
t6 = st6* (pi/180);

[px,py,pz,O,A,T] =
calculate_forward(t1,t2,t3,t4,t5,t6);

set(handles.on_x_px, 'String',
num2str(round(px,4)));
set(handles.on_y_py,
'String',num2str(round(py,4)));
set(handles.on_z_pz, 'String',
num2str(round(pz,4)));
set(handles.on_o_po, 'String',
num2str(round(O,4)));
set(handles.on_a_pa, 'String',
num2str(round(A,4)));
set(handles.on_t_pt, 'String',
num2str(round(T,4)));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(st1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(st2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(st3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(st4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(st5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(st6));
set_param('KawaSim','SimulationCommand','start
');
set_param('KawaSim','SimulationCommand','conti
nue');
set_param('KawaSim','SimulationCommand','pause
');
end
%
function on_slid2_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function on_slid3_Callback(hObject, eventdata,
handles)
set(handles.slid6, 'SliderStep'
,[0.1/(360+360), 2*0.1/(360+360)]);

st1 = get(handles.on_slid1, 'Value');
set(handles.on_t1,'String', num2str(st1));
t1 = - st1* (pi/180);

st2 = get(handles.on_slid2, 'Value');
set(handles.on_t2,'String', num2str(st2));
t2 = (-st2+90)* (pi/180);

st3 = get(handles.on_slid3, 'Value');
set(handles.on_t3,'String', num2str(st3));
t3 = (st3+90)* (pi/180);

st4 = get(handles.on_slid4, 'Value');
set(handles.on_t4,'String', num2str(st4));
t4 = (st4+180)* (pi/180);
```

```
    set(findall(handles.panel1, '-property',
'enable'), 'enable', 'off')
    set(handles.panel1, 'visible','off')
    set(findall(handles.panel3, '-property',
'enable'), 'enable', 'off')
    set(handles.panel3, 'visible','off')
end
%
function t1_Callback(hObject, eventdata,
handles)
Value  = get(handles.t1, 'Value');
set(handles.slid1,'String', num2str(Value));
end
%
function t1_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function t2_Callback(hObject, eventdata,
handles)
Value  = get(handles.t2, 'Value');
set(handles.slid2,'String', num2str(Value));
end
%
function t2_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function t3_Callback(hObject, eventdata,
handles)
Value  = get(handles.t3, 'Value');
set(handles.slid3,'String', num2str(Value));
end
%
function t3_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function t4_Callback(hObject, eventdata,
handles)
Value  = get(handles.t4, 'Value');
set(handles.slid4,'String', num2str(Value));
end
%
function t4_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function t5_Callback(hObject, eventdata,
handles)
Value  = get(handles.t5, 'Value');
set(handles.slid5,'String', num2str(Value));
end
%
function t5_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
st5 = get(handles.on_slid5, 'Value');
set(handles.on_t5,'String', num2str(st5));
t5 = (st5+180)* (pi/180);

st6 = get(handles.on_slid6, 'Value');
set(handles.on_t6,'String', num2str(st6));
t6 = st6* (pi/180);

[px,py,pz,O,A,T] =
calculate_forward(t1,t2,t3,t4,t5,t6);

set(handles.on_x_px, 'String',
num2str(round(px,4)));
set(handles.on_y_py,
'String',num2str(round(py,4)));
set(handles.on_z_pz, 'String',
num2str(round(pz,4)));
set(handles.on_o_po, 'String',
num2str(round(O,4)));
set(handles.on_a_pa, 'String',
num2str(round(A,4)));
set(handles.on_t_pt, 'String',
num2str(round(T,4)));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(st1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(st2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(st3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(st4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(st5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(st6));
set_param('KawaSim','SimulationCommand','start
');
set_param('KawaSim','SimulationCommand','conti
nue');
set_param('KawaSim','SimulationCommand','pause
');
end
%
function on_slid3_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function on_slid4_Callback(hObject, eventdata,
handles)
set(handles.slid6, 'SliderStep'
,[0.1/(360+360), 2*0.1/(360+360)]);

st1 = get(handles.on_slid1, 'Value');
set(handles.on_t1,'String', num2str(st1));
t1 = - st1* (pi/180);

st2 = get(handles.on_slid2, 'Value');
set(handles.on_t2,'String', num2str(st2));
t2 = (-st2+90)* (pi/180);

st3 = get(handles.on_slid3, 'Value');
set(handles.on_t3,'String', num2str(st3));
t3 = (st3+90)* (pi/180);

st4 = get(handles.on_slid4, 'Value');
set(handles.on_t4,'String', num2str(st4));
t4 = (st4+180)* (pi/180);

st5 = get(handles.on_slid5, 'Value');
set(handles.on_t5,'String', num2str(st5));
t5 = (st5+180)* (pi/180);

st6 = get(handles.on_slid6, 'Value');
set(handles.on_t6,'String', num2str(st6));
```

```
end
end
%
function t6_Callback(hObject, eventdata,
handles)
Value  = get(handles.t6, 'Value');
set(handles.slid6,'String', num2str(Value));
end

%
function t6_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function slid1_Callback(hObject, eventdata,
handles)
set(handles.slid1, 'SliderStep'
,[0.1/(180+180), 2*0.1/(180+180)]);

st1 = get(handles.slid1, 'Value');
set(handles.t1,'String', num2str(st1));
t1 = - st1* (pi/180);

st2 = get(handles.slid2, 'Value');
set(handles.t2,'String', num2str(st2));
t2 = (-st2+90)* (pi/180);

st3 = get(handles.slid3, 'Value');
set(handles.t3,'String', num2str(st3));
t3 = (st3+90)* (pi/180);

st4 = get(handles.slid4, 'Value');
set(handles.t4,'String', num2str(st4));
t4 = (st4+180)* (pi/180);

st5 = get(handles.slid5, 'Value');
set(handles.t5,'String', num2str(st5));
t5 = (st5+180)* (pi/180);

st6 = get(handles.slid6, 'Value');
set(handles.t6,'String', num2str(st6));
t6 = st6* (pi/180);

[px,py,pz,O,A,T] =
calculate_forward(t1,t2,t3,t4,t5,t6);

set(handles.x_px, 'String',
num2str(round(px,4)));
set(handles.y_py,
'String',num2str(round(py,4)));
set(handles.z_pz, 'String',
num2str(round(pz,4)));
set(handles.o_po, 'String',
num2str(round(O,4)));
set(handles.a_pa, 'String',
num2str(round(A,4)));
set(handles.t_pt, 'String',
num2str(round(T,4)));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(st1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(st2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(st3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(st4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(st5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(st6));
set_param('KawaSim','SimulationCommand','sta
rt');
set_param('KawaSim','SimulationCommand','con
tinue');
t6 = st6* (pi/180);

[px,py,pz,O,A,T] =
calculate_forward(t1,t2,t3,t4,t5,t6);

set(handles.on_x_px, 'String',
num2str(round(px,4)));
set(handles.on_y_py,
'String',num2str(round(py,4)));
set(handles.on_z_pz, 'String',
num2str(round(pz,4)));
set(handles.on_o_po, 'String',
num2str(round(O,4)));
set(handles.on_a_pa, 'String',
num2str(round(A,4)));
set(handles.on_t_pt, 'String',
num2str(round(T,4)));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(st1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(st2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(st3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(st4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(st5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(st6));
set_param('KawaSim','SimulationCommand','start
');
set_param('KawaSim','SimulationCommand','conti
nue');
set_param('KawaSim','SimulationCommand','pause
');
end
%
function on_slid4_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function on_slid5_Callback(hObject, eventdata,
handles)
set(handles.slid6, 'SliderStep'
,[0.1/(360+360), 2*0.1/(360+360)]);

st1 = get(handles.on_slid1, 'Value');
set(handles.on_t1,'String', num2str(st1));
t1 = - st1* (pi/180);

st2 = get(handles.on_slid2, 'Value');
set(handles.on_t2,'String', num2str(st2));
t2 = (-st2+90)* (pi/180);

st3 = get(handles.on_slid3, 'Value');
set(handles.on_t3,'String', num2str(st3));
t3 = (st3+90)* (pi/180);

st4 = get(handles.on_slid4, 'Value');
set(handles.on_t4,'String', num2str(st4));
t4 = (st4+180)* (pi/180);

st5 = get(handles.on_slid5, 'Value');
set(handles.on_t5,'String', num2str(st5));
t5 = (st5+180)* (pi/180);

st6 = get(handles.on_slid6, 'Value');
set(handles.on_t6,'String', num2str(st6));
t6 = st6* (pi/180);

[px,py,pz,O,A,T] =
calculate_forward(t1,t2,t3,t4,t5,t6);

set(handles.on_x_px, 'String',
num2str(round(px,4)));
```

```
set_param('KawaSim','SimulationCommand','pau
se');
end
%
function slid2_Callback(hObject, eventdata,
handles)
set(handles.slid2, 'SliderStep'
,[0.1/(170+45), 2*0.1/(170+45)]);

st1 = get(handles.slid1, 'Value');
set(handles.t1,'String', num2str(st1));
t1 = - st1* (pi/180);

st2 = get(handles.slid2, 'Value');
set(handles.t2,'String', num2str(st2));
t2 = (-st2+90)* (pi/180);

st3 = get(handles.slid3, 'Value');
set(handles.t3,'String', num2str(st3));
t3 = (st3+90)* (pi/180);

st4 = get(handles.slid4, 'Value');
set(handles.t4,'String', num2str(st4));
t4 = (st4+180)* (pi/180);

st5 = get(handles.slid5, 'Value');
set(handles.t5,'String', num2str(st5));
t5 = (st5+180)* (pi/180);

st6 = get(handles.slid6, 'Value');
set(handles.t6,'String', num2str(st6));
t6 = st6* (pi/180);

[px,py,pz,O,A,T] =
calculate_forward(t1,t2,t3,t4,t5,t6);

set(handles.x_px, 'String',
num2str(round(px,4)));
set(handles.y_py,
'String',num2str(round(py,4)));
set(handles.z_pz, 'String',
num2str(round(pz,4)));
set(handles.o_po, 'String',
num2str(round(O,4)));
set(handles.a_pa, 'String',
num2str(round(A,4)));
set(handles.t_pt, 'String',
num2str(round(T,4)));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(st1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(st2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(st3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(st4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(st5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(st6));
set_param('KawaSim','SimulationCommand','sta
rt');
set_param('KawaSim','SimulationCommand','con
tinue');
set_param('KawaSim','SimulationCommand','pau
se');
end
%
function slid2_CreateFcn(hObject, eventdata,
handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9
.9]);
end
end
%
function slid3_Callback(hObject, eventdata,
handles)
set(handles.on_y_py,
'String',num2str(round(py,4)));
set(handles.on_z_pz, 'String',
num2str(round(pz,4)));
set(handles.on_o_po, 'String',
num2str(round(O,4)));
set(handles.on_a_pa, 'String',
num2str(round(A,4)));
set(handles.on_t_pt, 'String',
num2str(round(T,4)));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(st1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(st2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(st3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(st4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(st5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(st6));
set_param('KawaSim','SimulationCommand','start
');
set_param('KawaSim','SimulationCommand','conti
nue');
set_param('KawaSim','SimulationCommand','pause
');
end
%
function on_slid5_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function on_slid6_Callback(hObject, eventdata,
handles)
set(handles.slid6, 'SliderStep'
,[0.1/(360+360), 2*0.1/(360+360)]);

st1 = get(handles.on_slid1, 'Value');
set(handles.on_t1,'String', num2str(st1));
t1 = - st1* (pi/180);

st2 = get(handles.on_slid2, 'Value');
set(handles.on_t2,'String', num2str(st2));
t2 = (-st2+90)* (pi/180);

st3 = get(handles.on_slid3, 'Value');
set(handles.on_t3,'String', num2str(st3));
t3 = (st3+90)* (pi/180);

st4 = get(handles.on_slid4, 'Value');
set(handles.on_t4,'String', num2str(st4));
t4 = (st4+180)* (pi/180);

st5 = get(handles.on_slid5, 'Value');
set(handles.on_t5,'String', num2str(st5));
t5 = (st5+180)* (pi/180);

st6 = get(handles.on_slid6, 'Value');
set(handles.on_t6,'String', num2str(st6));
t6 = st6* (pi/180);

[px,py,pz,O,A,T] =
calculate_forward(t1,t2,t3,t4,t5,t6);

set(handles.on_x_px, 'String',
num2str(round(px,4)));
set(handles.on_y_py,
'String',num2str(round(py,4)));
set(handles.on_z_pz, 'String',
num2str(round(pz,4)));
set(handles.on_o_po, 'String',
num2str(round(O,4)));
```

```
set(handles.slid3, 'SliderStep'
,[0.1/(208+82), 2*0.1/(208+82)]);

st1 = get(handles.slid1, 'Value');
set(handles.t1,'String', num2str(st1));
t1 = - st1* (pi/180);

st2 = get(handles.slid2, 'Value');
set(handles.t2,'String', num2str(st2));
t2 = (-st2+90)* (pi/180);

st3 = get(handles.slid3, 'Value');
set(handles.t3,'String', num2str(st3));
t3 = (st3+90)* (pi/180);

st4 = get(handles.slid4, 'Value');
set(handles.t4,'String', num2str(st4));
t4 = (st4+180)* (pi/180);

st5 = get(handles.slid5, 'Value');
set(handles.t5,'String', num2str(st5));
t5 = (st5+180)* (pi/180);

st6 = get(handles.slid6, 'Value');
set(handles.t6,'String', num2str(st6));
t6 = st6* (pi/180);

[px,py,pz,O,A,T] =
calculate_forward(t1,t2,t3,t4,t5,t6);

set(handles.x_px, 'String',
num2str(round(px,4)));
set(handles.y_py,
'String',num2str(round(py,4)));
set(handles.z_pz, 'String',
num2str(round(pz,4)));
set(handles.o_po, 'String',
num2str(round(O,4)));
set(handles.a_pa, 'String',
num2str(round(A,4)));
set(handles.t_pt, 'String',
num2str(round(T,4)));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(st1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(st2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(st3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(st4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(st5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(st6));
set_param('KawaSim','SimulationCommand','sta
rt');
set_param('KawaSim','SimulationCommand','con
tinue');
set_param('KawaSim','SimulationCommand','pau
se');
end
%
function slid3_CreateFcn(hObject, eventdata,
handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9
.9]);
end
end
%
function slid4_Callback(hObject, eventdata,
handles)
set(handles.slid4, 'SliderStep'
,[0.1/(540+180), 2*0.1/(540+180)]);

st1 = get(handles.slid1, 'Value');
set(handles.t1,'String', num2str(st1));
t1 = - st1* (pi/180);
set(handles.on_a_pa, 'String',
num2str(round(A,4)));
set(handles.on_t_pt, 'String',
num2str(round(T,4)));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(st1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(st2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(st3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(st4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(st5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(st6));
set_param('KawaSim','SimulationCommand','start
');
set_param('KawaSim','SimulationCommand','conti
nue');
set_param('KawaSim','SimulationCommand','pause
');
end
%
function on_slid6_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function on_runj_Callback(hObject, eventdata,
handles)
pi = 3.14159265359;
st1 = str2double(get(handles.on_t1,
'String'));
if (st1>180)
    st1=180; set(handles.on_t1,'String',
num2str(st1)); end
if st1<-180
    st1=-180; set(handles.on_t1,'String',
num2str(st1)); end
set(handles.on_slid1,'Value', st1); t1 = - st1
* (pi/180);
st2 = str2double(get(handles.on_t2,
'String'));
if st2<-80
    st2 = -80; set(handles.on_t2,'String',
num2str(st2)); end
if st2>135
    st2=135; set(handles.on_t2,'String',
num2str(st2)); end
set(handles.on_slid2,'Value', st2); t2 = (-st2
+90) *(pi/180);
st3 = str2double(get(handles.on_t3,
'String'));
if st3<-172
    st3 = -172; set(handles.on_t3,'String',
num2str(st3)); end
if st3>118
    st3=118; set(handles.on_t3,'String',
num2str(st3)); end
set(handles.on_slid3,'Value', st3); t3 =
(st3+90) * (pi/180);
st4 = str2double(get(handles.on_t4,
'String'));
if st4<-360
    st4 = -360; set(handles.on_t4,'String',
num2str(st4)); end
if st4>360
    st4=360; set(handles.on_t4,'String',
num2str(st4)); end
set(handles.on_slid4,'Value', st4); t4 =
(st4+180) * (pi/180);
st5 = str2double(get(handles.on_t5,
'String'));
if st5<-145
```

```
st2 = get(handles.slid2, 'Value');
set(handles.t2,'String', num2str(st2));
t2 = (-st2+90)* (pi/180);

st3 = get(handles.slid3, 'Value');
set(handles.t3,'String', num2str(st3));
t3 = (st3+90)* (pi/180);

st4 = get(handles.slid4, 'Value');
set(handles.t4,'String', num2str(st4));
t4 = (st4+180)* (pi/180);

st5 = get(handles.slid5, 'Value');
set(handles.t5,'String', num2str(st5));
t5 = (st5+180)* (pi/180);

st6 = get(handles.slid6, 'Value');
set(handles.t6,'String', num2str(st6));
t6 = st6* (pi/180);

[px,py,pz,O,A,T] =
calculate_forward(t1,t2,t3,t4,t5,t6);

set(handles.x_px, 'String',
num2str(round(px,4)));
set(handles.y_py,
'String',num2str(round(py,4)));
set(handles.z_pz, 'String',
num2str(round(pz,4)));
set(handles.o_po, 'String',
num2str(round(O,4)));
set(handles.a_pa, 'String',
num2str(round(A,4)));
set(handles.t_pt, 'String',
num2str(round(T,4)));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(st1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(st2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(st3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(st4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(st5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(st6));
set_param('KawaSim','SimulationCommand','sta
rt');
set_param('KawaSim','SimulationCommand','con
tinue');
set_param('KawaSim','SimulationCommand','pau
se');
end
%
function slid4_CreateFcn(hObject, eventdata,
handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9
.9]);
end
end
%
function slid5_Callback(hObject, eventdata,
handles)
set(handles.slid5, 'SliderStep' ,[0.1/(325-
35), 2*0.1/(325-35)]);

st1 = get(handles.slid1, 'Value');
set(handles.t1,'String', num2str(st1));
t1 = - st1* (pi/180);

st2 = get(handles.slid2, 'Value');
set(handles.t2,'String', num2str(st2));
t2 = (-st2+90)* (pi/180);
    st5 = -145; set(handles.on_t5,'String',
num2str(st5)); end
if st5>145
    st5=145; set(handles.on_t5,'String',
num2str(st5)); end
set(handles.on_slid5,'Value', st5); t5 =
(st5+180) * (pi/180);
st6 = str2double(get(handles.on_t6,
'String'));
if st6<-360
    st6 = -360; set(handles.on_t6,'String',
num2str(st6)); end
if st6>360
    st6=360; set(handles.on_t6,'String',
num2str(st6)); end
set(handles.on_slid6,'Value', st6); t6 = st6 *
(pi/180);


[px,py,pz,O,A,T] =
calculate_forward(t1,t2,t3,t4,t5,t6);

set(handles.on_x_px, 'String',
num2str(round(px,4)));
set(handles.on_y_py,
'String',num2str(round(py,4)));
set(handles.on_z_pz, 'String',
num2str(round(pz,4)));
set(handles.on_o_po, 'String',
num2str(round(O,4)));
set(handles.on_a_pa, 'String',
num2str(round(A,4)));
set(handles.on_t_pt, 'String',
num2str(round(T,4)));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(st1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(st2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(st3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(st4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(st5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(st6));
set_param('KawaSim','SimulationCommand','start
');
set_param('KawaSim','SimulationCommand','conti
nue');
set_param('KawaSim','SimulationCommand','pause
');
end
%
function on_resetj_Callback(hObject,
eventdata, handles)
t = 0;
t1 = num2str(t);
%sliders
set(handles.on_slid1, 'Value', 0);
set(handles.on_slid2,  'Value', 0);
set(handles.on_slid3,  'Value', 0);
set(handles.on_slid4,  'Value', 0);
set(handles.on_slid5,  'Value', 0);
set(handles.on_slid6,  'Value', 0);
%gonies t
set(handles.on_t1, 'String', t);
set(handles.on_t2,  'String', t);
set(handles.on_t3,  'String', t);
set(handles.on_t4,  'String', t);
set(handles.on_t5,  'String',t);
set(handles.on_t6,  'String', t);
%times x,y,z,o,a,t
set(handles.on_x_px, 'String', t);
set(handles.on_y_py, 'String', t);
set(handles.on_z_pz, 'String', t);
set(handles.on_o_po, 'String', t);
set(handles.on_a_pa, 'String', t);
set(handles.on_t_pt, 'String', t);
```

```
st3 = get(handles.slid3, 'Value');
set(handles.t3,'String', num2str(st3));
t3 = (st3+90)* (pi/180);

st4 = get(handles.slid4, 'Value');
set(handles.t4,'String', num2str(st4));
t4 = (st4+180)* (pi/180);

st5 = get(handles.slid5, 'Value');
set(handles.t5,'String', num2str(st5));
t5 = (st5+180)* (pi/180);

st6 = get(handles.slid6, 'Value');
set(handles.t6,'String', num2str(st6));
t6 = st6* (pi/180);

[px,py,pz,O,A,T] =
calculate_forward(t1,t2,t3,t4,t5,t6);

set(handles.x_px, 'String',
num2str(round(px,4)));
set(handles.y_py,
'String',num2str(round(py,4)));
set(handles.z_pz, 'String',
num2str(round(pz,4)));
set(handles.o_po, 'String',
num2str(round(O,4)));
set(handles.a_pa, 'String',
num2str(round(A,4)));
set(handles.t_pt, 'String',
num2str(round(T,4)));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(st1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(st2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(st3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(st4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(st5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(st6));
set_param('KawaSim','SimulationCommand','sta
rt');
set_param('KawaSim','SimulationCommand','con
tinue');
set_param('KawaSim','SimulationCommand','pau
se');
end
%
function slid5_CreateFcn(hObject, eventdata,
handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9
.9]);
end
end
%
function slid6_Callback(hObject, eventdata,
handles)
set(handles.slid6, 'SliderStep'
,[0.1/(360+360), 2*0.1/(360+360)]);

st1 = get(handles.slid1, 'Value');
set(handles.t1,'String', num2str(st1));
t1 = - st1* (pi/180);

st2 = get(handles.slid2, 'Value');
set(handles.t2,'String', num2str(st2));
t2 = (-st2+90)* (pi/180);

st3 = get(handles.slid3, 'Value');
set(handles.t3,'String', num2str(st3));
t3 = (st3+90)* (pi/180);

st4 = get(handles.slid4, 'Value');
set(handles.t4,'String', num2str(st4));
```

```
set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t));

set_param('KawaSim','SimulationCommand','updat
e');
end
%BASE ONLINE
function on_base_Callback(hObject, eventdata,
handles)
    set(findall(handles.on_panel_1, '-
property', 'enable'), 'enable', 'on')
    set(handles.on_panel_1, 'visible','on')

    set(findall(handles.on_panel_2, '-
property', 'enable'), 'enable', 'off')
    set(handles.on_panel_2, 'visible','off')

    set(findall(handles.on_panel_3, '-
property', 'enable'), 'enable', 'off')
    set(handles.on_panel_3, 'visible','off')
end
%
function on_px_Callback(hObject, eventdata,
handles)
Value  = get(handles.on_px, 'Value');
set(handles.on_sliderx,'String',
num2str(Value));
end
%
function on_px_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function on_py_Callback(hObject, eventdata,
handles)
Value  = get(handles.on_py, 'Value');
set(handles.on_slidery,'String',
num2str(Value));
end
%
function on_py_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function on_pz_Callback(hObject, eventdata,
handles)
Value  = get(handles.on_pz, 'Value');
set(handles.on_sliderz,'String',
num2str(Value));
end
%
function on_pz_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
t4 = (st4+180)* (pi/180);

st5 = get(handles.slid5, 'Value');
set(handles.t5,'String', num2str(st5));
t5 = (st5+180)* (pi/180);

st6 = get(handles.slid6, 'Value');
set(handles.t6,'String', num2str(st6));
t6 = st6* (pi/180);

[px,py,pz,O,A,T] =
calculate_forward(t1,t2,t3,t4,t5,t6);

set(handles.x_px, 'String',
num2str(round(px,4)));
set(handles.y_py,
'String',num2str(round(py,4)));
set(handles.z_pz, 'String',
num2str(round(pz,4)));
set(handles.o_po, 'String',
num2str(round(O,4)));
set(handles.a_pa, 'String',
num2str(round(A,4)));
set(handles.t_pt, 'String',
num2str(round(T,4)));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(st1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(st2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(st3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(st4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(st5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(st6));
set_param('KawaSim','SimulationCommand','sta
rt');
set_param('KawaSim','SimulationCommand','con
tinue');
set_param('KawaSim','SimulationCommand','pau
se');
end
%
function slid6_CreateFcn(hObject, eventdata,
handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9
.9]);
end
end
%
function runj_Callback(hObject, eventdata,
handles)
pi = 3.14159265359;
%read the angles
st1 = str2double(get(handles.t1, 'String'));
if (round(st1)>180)
    st1=180;
    set(handles.t1,'String', num2str(st1));
end
if st1<-180
    st1=-180;
    set(handles.t1,'String', num2str(st1));
end
set(handles.slid1,'Value', st1); t1 = - st1
* (pi/180);
st2 = str2double(get(handles.t2, 'String'));
if st2<-80
    st2 = -80;
    set(handles.t2,'String', num2str(st2));
end
if st2>135
    st2=135;
    set(handles.t2,'String', num2str(st2));
end
end
%
function on_po_Callback(hObject, eventdata,
handles)
Value  = get(handles.on_po, 'Value');
set(handles.on_slidero,'String',
num2str(Value));
end
%
function on_po_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function on_pa_Callback(hObject, eventdata,
handles)
Value  = get(handles.on_pa, 'Value');
set(handles.on_slidera,'String',
num2str(Value));
end
%
function on_pa_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function on_pt_Callback(hObject, eventdata,
handles)
Value  = get(handles.on_pt, 'Value');
set(handles.on_slidert,'String',
num2str(Value));
end
%
function on_pt_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function on_sliderx_Callback(hObject,
eventdata, handles)
px = get(handles.on_sliderx, 'Value');
set(handles.on_px,'String', num2str(px));
py = get(handles.on_slidery, 'Value');
set(handles.on_py,'String', num2str(py));
pz = get(handles.on_sliderz, 'Value');
set(handles.on_pz,'String', num2str(pz));
sog = get(handles.on_slidero, 'Value');
set(handles.on_po,'String', num2str(sog)); og
= sog* (pi/180);
sag = get(handles.on_slidera, 'Value');
set(handles.on_pa,'String', num2str(sag)); ag
= sag *(pi/180);
stg = get(handles.on_slidert, 'Value');
set(handles.on_pt,'String', num2str(stg)); tg
= stg* (pi/180);

[t1,t2,t3,t4,t5,t6] =
calculate_inverse(px,py,pz,og,ag,tg);

set(handles.on_t1p, 'String', round(t1,4));
set(handles.on_t2p, 'String', round(t2,4));
set(handles.on_t3p, 'String', round(t3,4));
set(handles.on_t4p, 'String', round(t4,4));
set(handles.on_t5p, 'String', round(t5,4));
set(handles.on_t6p, 'String', round(t6,4));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t1));
```

```
set(handles.slid2,'Value', st2); t2 = (-st2
+90) *(pi/180);
st3 = str2double(get(handles.t3, 'String'));
if st3<-172
    st3 = -172;
    set(handles.t3,'String', num2str(st3));
end
if st3>118
    st3=118;
    set(handles.t3,'String', num2str(st3));
end
set(handles.slid3,'Value', st3); t3 =
(st3+90) * (pi/180);
st4 = str2double(get(handles.t4, 'String'));
if st4<-360
    st4 = -360;
    set(handles.t4,'String', num2str(st4));
end
if st4>360
    st4=360; set(handles.t4,'String',
num2str(st4));  end
set(handles.slid4,'Value', st4); t4 =
(st4+180) * (pi/180);
st5 = str2double(get(handles.t5, 'String'));
if st5<-145
    st5 = -145; set(handles.t5,'String',
num2str(st5));  end
if st5>145
    st5=145; set(handles.t5,'String',
num2str(st5));  end
set(handles.slid5,'Value', st5); t5 =
(st5+180) * (pi/180);
st6 = str2double(get(handles.t6, 'String'));
if st6<-360
    st6 = -360; set(handles.t6,'String',
num2str(st6));  end
if st6>360
    st6=360; set(handles.t6,'String',
num2str(st6)); end
set(handles.slid6,'Value', st6); t6 = st6 *
(pi/180);

%call function with the results
[px,py,pz,O,A,T] =
calculate_forward(t1,t2,t3,t4,t5,t6);
%set the values in gui
set(handles.x_px, 'String',
num2str(round(px,4)));
set(handles.y_py,
'String',num2str(round(py,4)));
set(handles.z_pz, 'String',
num2str(round(pz,4)));
set(handles.o_po, 'String',
num2str(round(O,4)));
set(handles.a_pa, 'String',
num2str(round(A,4)));
set(handles.t_pt, 'String',
num2str(round(T,4)));
%set the values in simulation
set_param('KawaSim/Slider Gain1', 'Gain',
num2str(st1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(st2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(st3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(st4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(st5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(st6));
set_param('KawaSim','SimulationCommand','sta
rt');
set_param('KawaSim','SimulationCommand','con
tinue');
set_param('KawaSim','SimulationCommand','pau
se');
end
%
```

```
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t6));

set_param('KawaSim','SimulationCommand','start
')
end
%
function on_sliderx_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function on_slidery_Callback(hObject,
eventdata, handles)
px = get(handles.on_sliderx, 'Value');
set(handles.on_px,'String', num2str(px));
py = get(handles.on_slidery, 'Value');
set(handles.on_py,'String', num2str(py));
pz = get(handles.on_sliderz, 'Value');
set(handles.on_pz,'String', num2str(pz));
sog = get(handles.on_slidero, 'Value');
set(handles.on_po,'String', num2str(sog)); og
= sog* (pi/180);
sag = get(handles.on_slidera, 'Value');
set(handles.on_pa,'String', num2str(sag)); ag
= sag *(pi/180);
stg = get(handles.on_slidert, 'Value');
set(handles.on_pt,'String', num2str(stg)); tg
= stg* (pi/180);

[t1,t2,t3,t4,t5,t6] =
calculate_inverse(px,py,pz,og,ag,tg);

set(handles.on_t1p, 'String', round(t1,4));
set(handles.on_t2p, 'String', round(t2,4));
set(handles.on_t3p, 'String', round(t3,4));
set(handles.on_t4p, 'String', round(t4,4));
set(handles.on_t5p, 'String', round(t5,4));
set(handles.on_t6p, 'String', round(t6,4));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t6));

set_param('KawaSim','SimulationCommand','start
')
end
%
function on_slidery_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function on_sliderz_Callback(hObject,
eventdata, handles)
px = get(handles.on_sliderx, 'Value');
set(handles.on_px,'String', num2str(px));
```

```
function add2_Callback(hObject, eventdata,
handles)
%add a new row in the results matrix of
joint
   j1 = get(handles.t1, 'String'); j2 =
get(handles.t2, 'String'); j3 =
get(handles.t3, 'String');
   j4 = get(handles.t4, 'String'); j5 =
get(handles.t5, 'String'); j6 =
get(handles.t6, 'String');
   x = get(handles.x_px, 'String'); y =
get(handles.y_py, 'String'); z =
get(handles.z_pz, 'String');
   o = get(handles.o_po, 'String'); a =
get(handles.a_pa, 'String'); t =
get(handles.t_pt, 'String');

   C = [{j1} {j2} {j3} {j4} {j5} {j6} {x}
{y} {z} {o} {a} {t}];
   current_data = get(handles.uitable2,
'data');
   current_data = [current_data ; C];

set(handles.uitable2,'data',current_data);
end
%
function remove2_Callback(hObject,
eventdata, handles)
   oldDat = get(handles.uitable2,'Data');
   nRows = size(oldDat,1);
   dat = cell(nRows-1,12);
   dat= oldDat(1:nRows-1,:);
   set(handles.uitable2,'Data',dat)
   guidata(hObject,handles)
end
%
function save2_Callback(hObject, eventdata,
handles)
   data = get(handles.uitable2,'Data');
   data_joint = str2double(data);
   save('data_joint.mat','data_joint');
   clear data_joint
end
%
function sim2_Callback(hObject, eventdata,
handles)
t1=0; t2=0; t3=0; t4=0; t5=0; t6=0;
U = load('data_joint.mat');
[r,c] = size(U.data_joint);

for i=1:r
    t1 = U.data_joint(i,1);
    t2 = U.data_joint(i,2);
    t3 = U.data_joint(i,3);
    t4 = U.data_joint(i,4);
    t5 = U.data_joint(i,5);
    t6 = U.data_joint(i,6);

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t6));
set_param('KawaSim','SimulationCommand','sta
rt');
set_param('KawaSim','SimulationCommand','con
tinue');
set_param('KawaSim','SimulationCommand','pau
se');
end
end
%
```

```
py = get(handles.on_slidery, 'Value');
set(handles.on_py,'String', num2str(py));
pz = get(handles.on_sliderz, 'Value');
set(handles.on_pz,'String', num2str(pz));
sog = get(handles.on_slidero, 'Value');
set(handles.on_po,'String', num2str(sog)); og
= sog* (pi/180);
sag = get(handles.on_slidera, 'Value');
set(handles.on_pa,'String', num2str(sag)); ag
= sag *(pi/180);
stg = get(handles.on_slidert, 'Value');
set(handles.on_pt,'String', num2str(stg)); tg
= stg* (pi/180);

[t1,t2,t3,t4,t5,t6] =
calculate_inverse(px,py,pz,og,ag,tg);

set(handles.on_t1p, 'String', round(t1,4));
set(handles.on_t2p, 'String', round(t2,4));
set(handles.on_t3p, 'String', round(t3,4));
set(handles.on_t4p, 'String', round(t4,4));
set(handles.on_t5p, 'String', round(t5,4));
set(handles.on_t6p, 'String', round(t6,4));


set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t6));

set_param('KawaSim','SimulationCommand','start
')
end
%
function on_sliderz_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function on_slidero_Callback(hObject,
eventdata, handles)
px = get(handles.on_sliderx, 'Value');
set(handles.on_px,'String', num2str(px));
py = get(handles.on_slidery, 'Value');
set(handles.on_py,'String', num2str(py));
pz = get(handles.on_sliderz, 'Value');
set(handles.on_pz,'String', num2str(pz));
sog = get(handles.on_slidero, 'Value');
set(handles.on_po,'String', num2str(sog)); og
= sog* (pi/180);
sag = get(handles.on_slidera, 'Value');
set(handles.on_pa,'String', num2str(sag)); ag
= sag *(pi/180);
stg = get(handles.on_slidert, 'Value');
set(handles.on_pt,'String', num2str(stg)); tg
= stg* (pi/180);

[t1,t2,t3,t4,t5,t6] =
calculate_inverse(px,py,pz,og,ag,tg);

set(handles.on_t1p, 'String', round(t1,4));
set(handles.on_t2p, 'String', round(t2,4));
set(handles.on_t3p, 'String', round(t3,4));
set(handles.on_t4p, 'String', round(t4,4));
set(handles.on_t5p, 'String', round(t5,4));
set(handles.on_t6p, 'String', round(t6,4));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t1));
```

```
function nc2_Callback(hObject, eventdata,
handles)

data_joint = load('data_joint.mat');
formatSpec = '[%4.2f , %4.2f , %4.2f, %4.2f,
%4.2f, %4.2f]\n';
[r,c] = size(data_joint.data_joint);
fileID = fopen('NC_Code.txt','w');
fprintf(fileID,'.PROGRAM
KAWA_SIMULATION\n');
for i=1:r
    fprintf(fileID,'JOINT SPEED1 ACCU2
TIMER0 TOOL1 WORK0 CLAMP1 (OFF,0,0,o) OX=
WX= # \n');
    fprintf(fileID, formatSpec,
data_joint.data_joint(i,1:6));
end
fprintf(fileID, '.END\n');
fclose(fileID);
end
%
function resetj_Callback(hObject, eventdata,
handles)
t = 0;
t1 = num2str(t);
%sliders
set(handles.slid1, 'Value', 0);
set(handles.slid2,  'Value', 0);
set(handles.slid3,  'Value', 0);
set(handles.slid4,  'Value', 0);
set(handles.slid5,  'Value', 0);
set(handles.slid6,  'Value', 0);
%gonies t
set(handles.t1, 'String', t);
set(handles.t2,  'String', t);
set(handles.t3,  'String', t);
set(handles.t4,  'String', t);
set(handles.t5,  'String',t);
set(handles.t6,  'String', t);
%times x,y,z,o,a,t
set(handles.x_px, 'String', t);
set(handles.y_py, 'String', t);
set(handles.z_pz, 'String', t);
set(handles.o_po, 'String', t);
set(handles.a_pa, 'String', t);
set(handles.t_pt, 'String', t);


set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t));

set_param('KawaSim','SimulationCommand','upd
ate');
end
%OFFLINE BASE
function base_Callback(hObject, eventdata,
handles)
    set(findall(handles.panel1, '-property',
'enable'), 'enable', 'on')
    set(handles.panel1, 'visible','on')
    set(findall(handles.panel2, '-property',
'enable'), 'enable', 'off')
    set(handles.panel2, 'visible','off')
    set(findall(handles.panel3, '-property',
'enable'), 'enable', 'off')
    set(handles.panel3, 'visible','off')
end
%
function px_Callback(hObject, eventdata,
handles)
```

```
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t6));

set_param('KawaSim','SimulationCommand','start
')
end
%
function on_slidero_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function on_slidera_Callback(hObject,
eventdata, handles)

px = get(handles.on_sliderx, 'Value');
set(handles.on_px,'String', num2str(px));
py = get(handles.on_slidery, 'Value');
set(handles.on_py,'String', num2str(py));
pz = get(handles.on_sliderz, 'Value');
set(handles.on_pz,'String', num2str(pz));
sog = get(handles.on_slidero, 'Value');
set(handles.on_po,'String', num2str(sog)); og
= sog* (pi/180);
sag = get(handles.on_slidera, 'Value');
set(handles.on_pa,'String', num2str(sag)); ag
= sag *(pi/180);
stg = get(handles.on_slidert, 'Value');
set(handles.on_pt,'String', num2str(stg)); tg
= stg* (pi/180);

[t1,t2,t3,t4,t5,t6] =
calculate_inverse(px,py,pz,og,ag,tg);

set(handles.on_t1p, 'String', round(t1,4));
set(handles.on_t2p, 'String', round(t2,4));
set(handles.on_t3p, 'String', round(t3,4));
set(handles.on_t4p, 'String', round(t4,4));
set(handles.on_t5p, 'String', round(t5,4));
set(handles.on_t6p, 'String', round(t6,4));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t6));

set_param('KawaSim','SimulationCommand','start
')
end
%
function on_slidera_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function on_slidert_Callback(hObject,
eventdata, handles)
```

```
Value  = get(handles.px, 'Value');
set(handles.sliderx,'String',
num2str(Value));
end
%
function px_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function py_Callback(hObject, eventdata,
handles)
Value  = get(handles.py, 'Value');
set(handles.slidery,'String',
num2str(Value));
end
%
function py_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function pz_Callback(hObject, eventdata,
handles)
Value  = get(handles.pz, 'Value');
set(handles.sliderz,'String',
num2str(Value));
end
%
function pz_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function og_Callback(hObject, eventdata,
handles)
Value  = get(handles.og, 'Value');
set(handles.slidero,'String',
num2str(Value));
end
%
function og_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function ag_Callback(hObject, eventdata,
handles)
Value  = get(handles.ag, 'Value');
set(handles.slidera,'String',
num2str(Value));
end
%
function ag_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function tg_Callback(hObject, eventdata,
handles)
```

```
px = get(handles.on_sliderx, 'Value');
set(handles.on_px,'String', num2str(px));
py = get(handles.on_slidery, 'Value');
set(handles.on_py,'String', num2str(py));
pz = get(handles.on_sliderz, 'Value');
set(handles.on_pz,'String', num2str(pz));
sog = get(handles.on_slidero, 'Value');
set(handles.on_po,'String', num2str(sog)); og
= sog* (pi/180);
sag = get(handles.on_slidera, 'Value');
set(handles.on_pa,'String', num2str(sag)); ag
= sag *(pi/180);
stg = get(handles.on_slidert, 'Value');
set(handles.on_pt,'String', num2str(stg)); tg
= stg* (pi/180);

[t1,t2,t3,t4,t5,t6] =
calculate_inverse(px,py,pz,og,ag,tg);

set(handles.on_t1p, 'String', round(t1,4));
set(handles.on_t2p, 'String', round(t2,4));
set(handles.on_t3p, 'String', round(t3,4));
set(handles.on_t4p, 'String', round(t4,4));
set(handles.on_t5p, 'String', round(t5,4));
set(handles.on_t6p, 'String', round(t6,4));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t6));

set_param('KawaSim','SimulationCommand','start
')
end
%
function on_slidert_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function on_runb_Callback(hObject, eventdata,
handles)

px = str2double(get(handles.on_px, 'String'));
set(handles.on_sliderx,'Value', px);
py = str2double(get(handles.on_py, 'String'));
set(handles.on_slidery,'Value', py);
pz = str2double(get(handles.on_pz, 'String'));
set(handles.on_sliderx,'Value', pz);
sog = str2double(get(handles.on_po,
'String')); set(handles.on_slidero,'Value',
sog); og = sog * (pi/180);
sag = str2double(get(handles.on_pa,
'String')); set(handles.on_slidera,'Value',
sag); ag = sag * (pi/180);
stg = str2double(get(handles.on_pa,
'String')); set(handles.on_slidert,'Value',
stg); tg = stg * (pi/180);

[t1,t2,t3,t4,t5,t6] =
calculate_inverse(px,py,pz,og,ag,tg);

set(handles.on_t1p, 'String', round(t1,4));
set(handles.on_t2p, 'String', round(t2,4));
set(handles.on_t3p, 'String', round(t3,4));
set(handles.on_t4p, 'String', round(t4,4));
set(handles.on_t5p, 'String', round(t5,4));
set(handles.on_t6p, 'String',round(t6,4));
```

```
Value  = get(handles.tg, 'Value');
set(handles.slidert,'String',
num2str(Value));
end
%
function tg_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function sliderx_Callback(hObject,
eventdata, handles)
px = get(handles.sliderx, 'Value');
set(handles.px,'String', num2str(px));
py = get(handles.slidery, 'Value');
set(handles.py,'String', num2str(py));
pz = get(handles.sliderz, 'Value');
set(handles.pz,'String', num2str(pz));
sog = get(handles.slidero, 'Value');
set(handles.og,'String', num2str(sog)); og =
sog* (pi/180);
sag = get(handles.slidera, 'Value');
set(handles.ag,'String', num2str(sag)); ag =
sag *(pi/180);
stg = get(handles.slidert, 'Value');
set(handles.tg,'String', num2str(stg)); tg =
stg* (pi/180);

[t1,t2,t3,t4,t5,t6] =
calculate_inverse(px,py,pz,og,ag,tg);

set(handles.t1p, 'String', round(t1,4));
set(handles.t2p, 'String', round(t2,4));
set(handles.t3p, 'String', round(t3,4));
set(handles.t4p, 'String', round(t4,4));
set(handles.t5p, 'String', round(t5,4));
set(handles.t6p, 'String',round(t6,4));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t6));

set_param('KawaSim','SimulationCommand','sta
rt')
end
%
function sliderx_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9
.9]);
end
end
%
function slidery_Callback(hObject,
eventdata, handles)
px = get(handles.sliderx, 'Value');
set(handles.px,'String', num2str(px));
py = get(handles.slidery, 'Value');
set(handles.py,'String', num2str(py));
pz = get(handles.sliderz, 'Value');
set(handles.pz,'String', num2str(pz));
sog = get(handles.slidero, 'Value');
set(handles.og,'String', num2str(sog)); og =
sog* (pi/180);

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t6));

set_param('KawaSim','SimulationCommand','start
')
end
%
function on_resetb_Callback(hObject,
eventdata, handles)
t = 0;
set(handles.on_t1p, 'String', t);
set(handles.on_t2p, 'String', t);
set(handles.on_t3p, 'String', t);
set(handles.on_t4p, 'String', t);
set(handles.on_t5p, 'String', t);
set(handles.on_t6p, 'String', t);
set(handles.on_px, 'String', t);
set(handles.on_py, 'String', t);
set(handles.on_pz, 'String', t);
set(handles.on_po, 'String', t);
set(handles.on_pa, 'String', t);
set(handles.on_pt, 'String', t);
set(handles.on_sliderx, 'Value', 0);
set(handles.on_slidery,  'Value', 0);
set(handles.on_sliderz,  'Value', 0);
set(handles.on_slidero,  'Value', 0);
set(handles.on_slidera,  'Value', 0);
set(handles.on_slidert,  'Value', 0);

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t));

set_param('KawaSim','SimulationCommand','updat
e');
end
%TOOL ONLINE & OFFLINE FUNCTIONS (EMPTY)
%OFFLINE FUNCTIONS
function tool_Callback(hObject, eventdata,
handles)
    set(findall(handles.panel2, '-property',
'enable'), 'enable', 'off')
    set(handles.panel2, 'visible','off')
    set(findall(handles.panel1, '-property',
'enable'), 'enable', 'off')
    set(handles.panel1, 'visible','off')
    set(findall(handles.panel3, '-property',
'enable'), 'enable', 'on')
    set(handles.panel3, 'visible','on')
end
%
function dx_Callback(hObject, eventdata,
handles)
end
%
function dx_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
```

```
sag = get(handles.slidera, 'Value');
set(handles.ag,'String', num2str(sag)); ag =
sag *(pi/180);
stg = get(handles.slidert, 'Value');
set(handles.tg,'String', num2str(stg)); tg =
stg* (pi/180);

[t1,t2,t3,t4,t5,t6] =
calculate_inverse(px,py,pz,og,ag,tg);

set(handles.t1p, 'String', round(t1,4));
set(handles.t2p, 'String', round(t2,4));
set(handles.t3p, 'String', round(t3,4));
set(handles.t4p, 'String', round(t4,4));
set(handles.t5p, 'String', round(t5,4));
set(handles.t6p, 'String',round(t6,4));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t6));

set_param('KawaSim','SimulationCommand','sta
rt')
end
%
function slidery_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9
.9]);
end
end
%
function sliderz_Callback(hObject,
eventdata, handles)
px = get(handles.sliderx, 'Value');
set(handles.px,'String', num2str(px));
py = get(handles.slidery, 'Value');
set(handles.py,'String', num2str(py));
pz = get(handles.sliderz, 'Value');
set(handles.pz,'String', num2str(pz));
sog = get(handles.slidero, 'Value');
set(handles.og,'String', num2str(sog)); og =
sog* (pi/180);
sag = get(handles.slidera, 'Value');
set(handles.ag,'String', num2str(sag)); ag =
sag *(pi/180);
stg = get(handles.slidert, 'Value');
set(handles.tg,'String', num2str(stg)); tg =
stg* (pi/180);

[t1,t2,t3,t4,t5,t6] =
calculate_inverse(px,py,pz,og,ag,tg);

set(handles.t1p, 'String', round(t1,4));
set(handles.t2p, 'String', round(t2,4));
set(handles.t3p, 'String', round(t3,4));
set(handles.t4p, 'String', round(t4,4));
set(handles.t5p, 'String', round(t5,4));
set(handles.t6p, 'String',round(t6,4));


set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t4));
    set(hObject,'BackgroundColor','white');
end
end
%
function dy_Callback(hObject, eventdata,
handles)
end
%
function dy_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function dz_Callback(hObject, eventdata,
handles)
end
%
function dz_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function do_Callback(hObject, eventdata,
handles)
end
%
function do_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function da_Callback(hObject, eventdata,
handles)
end
%
function da_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
%
function dt_Callback(hObject, eventdata,
handles)
end
%
function dt_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function sliderdx_Callback(hObject, eventdata,
handles)
end
%
function sliderdx_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
```

```
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t6));

set_param('KawaSim','SimulationCommand','sta
rt')
end
%
function slidero_Callback(hObject,
eventdata, handles)
px = get(handles.sliderx, 'Value');
set(handles.px,'String', num2str(px));
py = get(handles.slidery, 'Value');
set(handles.py,'String', num2str(py));
pz = get(handles.sliderz, 'Value');
set(handles.pz,'String', num2str(pz));
sog = get(handles.slidero, 'Value');
set(handles.og,'String', num2str(sog)); og =
sog* (pi/180);
sag = get(handles.slidera, 'Value');
set(handles.ag,'String', num2str(sag)); ag =
sag *(pi/180);
stg = get(handles.slidert, 'Value');
set(handles.tg,'String', num2str(stg)); tg =
stg* (pi/180);

[t1,t2,t3,t4,t5,t6] =
calculate_inverse(px,py,pz,og,ag,tg);

set(handles.t1p, 'String', round(t1,4));
set(handles.t2p, 'String', round(t2,4));
set(handles.t3p, 'String', round(t3,4));
set(handles.t4p, 'String', round(t4,4));
set(handles.t5p, 'String', round(t5,4));
set(handles.t6p, 'String',round(t6,4));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t6));

set_param('KawaSim','SimulationCommand','sta
rt')
end
%
function slidera_Callback(hObject,
eventdata, handles)
px = get(handles.sliderx, 'Value');
set(handles.px,'String', num2str(px));
py = get(handles.slidery, 'Value');
set(handles.py,'String', num2str(py));
pz = get(handles.sliderz, 'Value');
set(handles.pz,'String', num2str(pz));
sog = get(handles.slidero, 'Value');
set(handles.og,'String', num2str(sog)); og =
sog* (pi/180);
sag = get(handles.slidera, 'Value');
set(handles.ag,'String', num2str(sag)); ag =
sag *(pi/180);
stg = get(handles.slidert, 'Value');
set(handles.tg,'String', num2str(stg)); tg =
stg* (pi/180);

[t1,t2,t3,t4,t5,t6] =
calculate_inverse(px,py,pz,og,ag,tg);

set(handles.t1p, 'String', round(t1,4));
set(handles.t2p, 'String', round(t2,4));
set(handles.t3p, 'String', round(t3,4));
set(handles.t4p, 'String', round(t4,4));
set(handles.t5p, 'String', round(t5,4));
end
%
function sliderdy_Callback(hObject, eventdata,
handles)
end
%
function sliderdy_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function sliderdz_Callback(hObject, eventdata,
handles)
end
%
function sliderdz_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function sliderdo_Callback(hObject, eventdata,
handles)
end
%
function sliderdo_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function sliderda_Callback(hObject, eventdata,
handles)
end
%
function sliderda_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
function sliderdt_Callback(hObject, eventdata,
handles)
end
 %
function sliderdt_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function add3_Callback(hObject, eventdata,
handles)
%add a new row in the results matrix of tool
   j1 = get(handles.t1d, 'String'); j2 =
get(handles.t2d, 'String'); j3 =
get(handles.t3d, 'String');
   j4 = get(handles.t4d, 'String'); j5 =
get(handles.t5d, 'String'); j6 =
get(handles.t6d, 'String');
   x = get(handles.dx, 'String'); y =
get(handles.dy, 'String'); z = get(handles.dz,
'String');
   o = get(handles.do, 'String'); a =
get(handles.da, 'String'); t = get(handles.dt,
'String');

   C = [{j1} {j2} {j3} {j4} {j5} {j6} {x} {y}
{z} {o} {a} {t}];
```

```
set(handles.t6p, 'String',round(t6,4));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t6));

set_param('KawaSim','SimulationCommand','sta
rt')
end
%
function slidert_Callback(hObject,
eventdata, handles)
px = get(handles.sliderx, 'Value');
set(handles.px,'String', num2str(px));
py = get(handles.slidery, 'Value');
set(handles.py,'String', num2str(py));
pz = get(handles.sliderz, 'Value');
set(handles.pz,'String', num2str(pz));
sog = get(handles.slidero, 'Value');
set(handles.og,'String', num2str(sog)); og =
sog* (pi/180);
sag = get(handles.slidera, 'Value');
set(handles.ag,'String', num2str(sag)); ag =
sag *(pi/180);
stg = get(handles.slidert, 'Value');
set(handles.tg,'String', num2str(stg)); tg =
stg* (pi/180);

[t1,t2,t3,t4,t5,t6] =
calculate_inverse(px,py,pz,og,ag,tg);

set(handles.t1p, 'String', round(t1,4));
set(handles.t2p, 'String', round(t2,4));
set(handles.t3p, 'String', round(t3,4));
set(handles.t4p, 'String', round(t4,4));
set(handles.t5p, 'String', round(t5,4));
set(handles.t6p, 'String',round(t6,4));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t6));

set_param('KawaSim','SimulationCommand','sta
rt')
end
%
function runb_Callback(hObject, eventdata,
handles)

px = str2double(get(handles.px, 'String'));
set(handles.sliderx,'Value', px);
py = str2double(get(handles.py, 'String'));
set(handles.slidery,'Value', py);
pz = str2double(get(handles.pz, 'String'));
set(handles.sliderx,'Value', pz);
sog = str2double(get(handles.og, 'String'));
set(handles.slidero,'Value', sog); og = sog
* (pi/180);
sag = str2double(get(handles.ag, 'String'));
set(handles.slidera,'Value', sag); ag = sag
* (pi/180);
   current_data = get(handles.uitable3,
'data');
   current_data = [current_data ; C];
   set(handles.uitable3,'data',current_data);
end
%
function remove3_Callback(hObject, eventdata,
handles)
   oldDat = get(handles.uitable3,'Data');
   nRows = size(oldDat,1);
   dat = cell(nRows-1,12);
   dat= oldDat(1:nRows-1,:);
   set(handles.uitable3,'Data',dat)
   guidata(hObject,handles)
end
%
function save3_Callback(hObject, eventdata,
handles)
   data = get(handles.uitable3,'Data');
   data_tool = str2double(data);
   save('data_tool.mat','data_tool');
   clear data_tool
end
%
function sim3_Callback(hObject, eventdata,
handles)
end
%
function nc3_Callback(hObject, eventdata,
handles)
end
%ONLINE FUNCTIONS
function on_tool_Callback(hObject, eventdata,
handles)
    set(findall(handles.on_panel_1, '-
property', 'enable'), 'enable', 'off')
    set(handles.on_panel_1, 'visible','off')

    set(findall(handles.on_panel_2, '-
property', 'enable'), 'enable', 'off')
    set(handles.on_panel_2, 'visible','off')

    set(findall(handles.on_panel_3, '-
property', 'enable'), 'enable', 'on')
    set(handles.on_panel_3, 'visible','on')
end
%
function on_dx_Callback(hObject, eventdata,
handles)
end
%
function on_dx_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function on_dy_Callback(hObject, eventdata,
handles)
end
%
function on_dy_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function on_dz_Callback(hObject, eventdata,
handles)
end
%
function on_dz_CreateFcn(hObject, eventdata,
handles)
```

```
stg = str2double(get(handles.tg, 'String'));
set(handles.slidert,'Value', stg); tg = stg
* (pi/180);

[t1,t2,t3,t4,t5,t6] =
calculate_inverse(px,py,pz,og,ag,tg);

set(handles.t1p, 'String', round(t1,4));
set(handles.t2p, 'String', round(t2,4));
set(handles.t3p, 'String', round(t3,4));
set(handles.t4p, 'String', round(t4,4));
set(handles.t5p, 'String', round(t5,4));
set(handles.t6p, 'String',round(t6,4));

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t6));

set_param('KawaSim','SimulationCommand','sta
rt')
end
%
function add1_Callback(hObject, eventdata,
handles)
%add a new row in the results matrix of base
   j1 = get(handles.t1p, 'String'); j2 =
get(handles.t2p, 'String'); j3 =
get(handles.t3p, 'String');
   j4 = get(handles.t4p, 'String'); j5 =
get(handles.t5p, 'String'); j6 =
get(handles.t6p, 'String');
   x = get(handles.px, 'String'); y =
get(handles.py, 'String'); z =
get(handles.pz, 'String');
   o = get(handles.og, 'String'); a =
get(handles.ag, 'String'); t =
get(handles.tg, 'String');
   C = [{j1} {j2} {j3} {j4} {j5} {j6} {x}
{y} {z} {o} {a} {t}];
   current_data = get(handles.uitable1,
'data');
   current_data = [current_data ; C];

set(handles.uitable1,'data',current_data);
end
%
function remove1_Callback(hObject,
eventdata, handles)
   oldDat = get(handles.uitable1,'Data');
   nRows = size(oldDat,1);
   dat = cell(nRows-1,12);
   dat= oldDat(1:nRows-1,:);
   set(handles.uitable1,'Data',dat)
   guidata(hObject,handles)
end
%
function sim1_Callback(hObject, eventdata,
handles)
t1=0; t2=0; t3=0; t4=0; t5=0; t6=0;
U = load('data_base.mat');
[r,c] = size(U.data_base);

for i=1:r
    t1 = U.data_base(i,1);
    t2 = U.data_base(i,2);
    t3 = U.data_base(i,3);
    t4 = U.data_base(i,4);
    t5 = U.data_base(i,5);
    t6 = U.data_base(i,6);
```

```
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function on_do_Callback(hObject, eventdata,
handles)
end
%
function on_do_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function on_da_Callback(hObject, eventdata,
handles)
end
%
function on_da_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function on_dt_Callback(hObject, eventdata,
handles
end
%
function on_dt_CreateFcn(hObject, eventdata,
handles)
if ispc &&
isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end
%
function on_sliderdx_Callback(hObject,
eventdata, handles)
end
%
function on_sliderdx_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function on_sliderdy_Callback(hObject,
eventdata, handles)
end
%
function on_sliderdy_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function on_sliderdz_Callback(hObject,
eventdata, handles)
end
%
function on_sliderdz_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
```

```
set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t1));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t2));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t3));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t4));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t5));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t6));
set_param('KawaSim','SimulationCommand','sta
rt');
set_param('KawaSim','SimulationCommand','con
tinue');
set_param('KawaSim','SimulationCommand','pau
se');
end
end
%
function nc1_Callback(hObject, eventdata,
handles)

data_base = load('data_base.mat');
formatSpec = '[%4.2f , %4.2f , %4.2f, %4.2f,
%4.2f, %4.2f]\n';
[r,c] = size(data_base.data_base);
fileID = fopen('NC_Code.txt','w');
fprintf(fileID,'.PROGRAM
KAWA_SIMULATION\n');
for i=1:r
    fprintf(fileID,'JOINT SPEED1 ACCU2
TIMER0 TOOL1 WORK0 CLAMP1 (OFF,0,0,o) OX=
WX= # \n');
    fprintf(fileID, formatSpec,
data_base.data_base(i,1:6));
end
fprintf(fileID, '.END\n');
fclose(fileID);
end
%
function resetb_Callback(hObject, eventdata,
handles)
t = 0;
set(handles.t1p, 'String', t);
set(handles.t2p, 'String', t);
set(handles.t3p, 'String', t);
set(handles.t4p, 'String', t);
set(handles.t5p, 'String', t);
set(handles.t6p, 'String', t);
set(handles.px, 'String', t);
set(handles.py, 'String', t);
set(handles.pz, 'String', t);
set(handles.og, 'String', t);
set(handles.ag, 'String', t);
set(handles.tg, 'String', t);
set(handles.sliderx, 'Value', 0);
set(handles.slidery,  'Value', 0);
set(handles.sliderz,  'Value', 0);
set(handles.slidero,  'Value', 0);
set(handles.slidera,  'Value', 0);
set(handles.slidert,  'Value', 0);

set_param('KawaSim/Slider Gain1', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain2', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain3', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain4', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain5', 'Gain',
num2str(t));
set_param('KawaSim/Slider Gain6', 'Gain',
num2str(t));

set_param('KawaSim','SimulationCommand','upd
ate');
end
```

```
end
%
function on_sliderdt_Callback(hObject,
eventdata, handles)
end
%
function on_sliderdt_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
function on_sliderda_Callback(hObject,
eventdata, handles)
end
%
function on_sliderda_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function on_sliderdo_Callback(hObject,
eventdata, handles)
end
%
function on_sliderdo_CreateFcn(hObject,
eventdata, handles)
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
end
%
function on_run3_Callback(hObject, eventdata,
handles)
end
%
function on_reset3_Callback(hObject,
eventdata, handles)
end
%
```

ΔΙΕΘΝΕΣ ΠΑΝΕΠΙΣΤΗΜΙΟ ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΠΑΝΕΠΙΣΤΗΜΙΟΥΠΟΛΗ ΣΕΡΡΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ,

ΥΠΟΛΟΓΙΣΤΩΝ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

ΣΤΗ ΡΟΜΠΟΤΙΚΗ