ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΔΙΑΤΜΗΜΑΤΙΚΟ ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
«ΣΥΣΤΗΜΑΤΑ ΑΥΤΟΜΑΤΙΣΜΟΥ»

Μεταπτυχιακή Εργασία

Ολοκλήρωση Συστήματος Προγραμματισμού Ρομποτικού Βραχίονα μέσω Φωνητικών Εντολών

Παναγιώτης Μακρυλάκης

Επιβλέπων Καθηγητής: Πανώριος Μπενάρδος

ΑΘΗΝΑ 2023

Περίληψη

Η παρούσα διπλωματική εργασία πραγματεύεται την ολοκλήρωση συστήματος προγραμματισμό του ρομποτικού βραχίονα Staubli RX90L μέσω φωνητικών εντολών στα Αγγλικά. Κύριος σκοπός είναι η διευκόλυνση του προγραμματισμού (online) βιομηχανικών διεργασιών. Ο βραχίονας προγραμματίζεται με βάση τη γλώσσα V^+, οπότε τελικά οι φωνητικές εντολές μας θα πρέπει να μετατραπούν σε συμβατή με το βραχίονα γλώσσα. Το μεγαλύτερο μέρος της εργασίας επιτυγχάνεται με χρήση του λογισμικού MATLAB R2022b.

Στο πρώτο στάδιο της διαδικασίας, οι φωνητικές μας εντολές μετατρέπονται σε γραπτό κείμενο με χρήση των προ εκπαιδευμένων μοντέλων της Microsoft μέσω της υπηρεσίας Microsoft Speech Software Development Kit (SDK). Για την επίτευξη του σκοπού αυτού, δηλαδή μετατροπή φωνητικών προτάσεων σε γραπτό λόγο, σε πραγματικό χρόνο, δημιουργήθηκε ένας μικρός κώδικας σε γλώσσα Python, ο οποίος καλείται μέσα από το MATLAB. Επειδή δεν έχουμε ιδιαίτερο έλεγχο στη μετατροπή ενδέχεται να προκύψουν λάθη η πλειοψηφία των οποίων έχει προβλεφθεί και διορθώνεται αργότερα κατά την επεξεργασία του κειμένου.

Στη συνέχεια ακολουθεί η επεξεργασία του κειμένου και η εξαγωγή αριθμητικών δεδομένων από τα λεγόμενα του χρήστη. Το στάδιο αυτό είναι ιδιαίτερα σημαντικό και είναι καταλυτικής σημασίας για τη σωστή μετατροπή των φωνητικών εντολών σε εντολές V^+. Στη συνέχεια, το σύνολο των εντολών περνάνε μέσα από ένα μοντέλο μηχανικής μάθησης με σκοπό την κατηγοριοποίηση. Το μοντέλο είναι δυαδικό, και σκοπό έχει να ταξινομήσει την κάθε πρόταση ως εντολή η μη-εντολή. Οι εντολές θα περάσουν από μια διαδικασία όπου μέσω παρουσίας συγκεκριμένων λέξεων-κλειδιά κατηγοριοποιούνται ανάμεσα σε 7 προεπιλεγμένες εντολές V^+. Στη συνέχεια αναλύονται οι εντολές δίνοντας στον χρήστη την επιλογή να διορθώσει τυχόν λάθη κατηγοριοποίησης, και βγαίνουν συμπεράσματα όπως το αν απαιτείται επίλυση της αντίστροφης κινηματικής.

Οι εντολές που αφορούν κίνηση σε επιθυμητή θέση με επιθυμητό προσανατολισμό (εντολές: MOVE ή APPRO), πρέπει να αποτυπωθούν ως επιθυμητές γωνίες των αρθρώσεων, άρα απαιτείται επίλυση της αντίστροφης κινηματικής πριν οι εντολές σταλθούν στο βραχίονα. Η αντίστροφη κινηματική επιλύεται με χρήση του Robotics System Toolbox που προσφέρει η MathWorks, και μας επιτρέπει να μετατρέπουμε θέσεις (cartesian points) της μορφής «X, Y, Z, y, p, r» σε θέσεις αρθρώσεων (precision points) της μορφής «q_1, q_2, q_3, q_4, q_5, q_6». Τέλος, παράγεται αυτόματα ένας κώδικας σε γλώσσα V^+ με βάση τις ταξινομημένες εντολές και τα αριθμητικά στοιχεία των φωνητικών εισόδων.

Εκτός από τη λειτουργία μεμονωμένων εντολών που μόλις περιγράψαμε, καταφέραμε να προγραμματίσουμε με φωνητικές εντολές τη βιομηχανική διεργασία pick and place. Κατά τη συγκεκριμένη λειτουργία ο χρήστης πρέπει να χρησιμοποιεί λίγο πιο συγκεκριμένο λεξιλόγιο, αλλά το πλεονέκτημα είναι ότι με τη διάρθρωση λίγων προτάσεων ο χρήστης μπορεί εύκολα να προγραμματίσει μια βιομηχανική διεργασία με πολλές εφαρμογές, που σε άλλη περίπτωση θα απαιτούσε περισσότερο χρόνο για να ολοκληρωθεί.

Όλα τα παραπάνω συνενώθηκαν για τη δημιουργία μιας εφαρμογής με διεπιφάνεια επαφής χρήστη (GUI) με χρήση του MATLAB App Designer η οποία επεκτάθηκε ώστε να δέχεται και γραπτό λόγο ως είσοδο και να καλύπτει περισσότερες ανάγκες. Η όλη διαδικασία είναι μια πρώτη απόπειρα για την επίτευξη των σκοπών μας, αλλά φαίνεται να λειτουργεί ομαλά και αξιόπιστα.

Abstract

This thesis deals with the integration of a NLP based programming system of the Staubli RX90L robotic arm through voice commands in English. The primal objective is to facilitate the on-line programming of industrial processes. The arm is programmed based on the programming language V^+, so eventually our voice commands will need to be converted to a language compatible with the robotic arm. Most of the work is accomplished using MATLAB R2022b software.

In the first stage of the process, voice commands are converted into written text using Microsoft's pre-trained models through the Microsoft Speech Software Development Kit (SDK) service. To achieve this goal, i.e., real-time conversion of voice sentences into written speech, a small code was created in Python language, which is called through MATLAB. Because we have no particular control over the conversion and possible errors may occur, the majority of errors are foreseen and corrected later during the text processing.

Then follows the processing of the text and the extraction of numerical data from the user's spoken words. This stage is particularly important and is crucial for the correct conversion of voice commands into V^+ commands. Then, the set of commands is passed through a machine learning classification model. The model is binary and aims to classify each sentence as a command or non-command. The commands pass through a process that utilizes the existence of specific keywords in order to classify the commands among 7 known V^+ commands. The commands are then analyzed giving the user the option to correct any misclassification errors, and conclusions such as whether an inverse kinematics solution is required are drawn.

Commands regarding motion to a desired position with a desired orientation (commands: MOVE or APPRO) must be expressed as desired joint angles so, inverse kinematics must be solved before the commands are sent to the robotic arm. Inverse kinematics is solved using the Robotics System Toolbox offered by MathWorks, and it allows us to convert points of the form "X, Y, Z, y, p, r" into joint angles (precision points) of the form "q_1, q_2, q_3, q_4, q_5, q_6". Finally, a code in V^+ language is automatically generated based on the classified commands and numeric elements of the voice inputs.

In addition to the loose command operation just described, we were able to voice-program the industrial process called pick and place, as will be discussed next. In this operation the user must use a slightly more specific vocabulary, but the advantage is that by articulating a few sentences the user can easily program an industrial process with many applications that would otherwise require more time to achieve.

All of the above was put together to create a graphical user interface (GUI) application using MATLAB App Designer that was extended to accept textual input and to cover more needs. The whole process is an early attempt to achieve the desired goals, but it seems to be working smoothly and quite reliably.

Ευχαριστίες

Αρχικά, θα ήθελα να ευχαριστήσω ιδιαίτερα τον επιβλέποντα καθηγητή μου, κύριο Πανώριο Μπενάρδο, για την άπταιστη συνεργασία που είχαμε κατά την εκπόνηση της διπλωματικής μου εργασίας. Η καθοδήγηση του ήταν πολύ κρίσιμη σε ερευνητικά και τεχνικά ζητήματα. Θα ήθελα επίσης να τον ευχαριστήσω που μου εμπιστεύτηκε ένα τόσο ενδιαφέρον θέμα, και που πίστεψε στις δυνατότητες μου χωρίς να με γνωρίζει εκ των προτέρων. Με αυτή την εμπειρία είχα τη δυνατότητα να ασχοληθώ με καινούριους για μένα τομείς, να διευρύνω σημαντικά τις γνώσεις μου ως μηχανικός και να βελτιώσω τον τρόπο σκέψης και εργασίας μου.

Επίσης, θα ήθελα να ευχαριστήσω την αγαπημένη μου φίλη και συνάδελφο Μαρία Καρανίκου για τη στήριξη και την αγάπη που μου προσέφερε, και την υπομονή που έκανε μαζί μου όλο αυτόν τον καιρό.

Τέλος, δε θα μπορούσα να παραλείψω να ευχαριστήσω τους γονείς, τα αδέρφια και τους κοντινούς φίλους μου που πάντα είναι πρόθυμοι να μου σταθούν και να μου προσφέρουν ανιδιοτελώς τη στήριξή τους.

Μακρυλάκης Παναγιώτης
Αθήνα, Σεπτέμβριος 2023

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

1 Εισαγωγή

Σύμφωνα με το Robotics Institute (*Robots - The Robotics Institute Carnegie Mellon University*) μπορούμε να ορίσουμε ως ρομπότ ένα επαναπρογραμματιζόμενο πολυλειτουργικό σύστημα, σχεδιασμένο έτσι ώστε να μπορεί να αλληλεπιδρά με το περιβάλλον του και να είναι σε θέση να εκτελέσει κάποια εργασία. Για παράδειγμα να μπορεί να μεταφέρει αντικείμενα, να εκτελεί πολύ ακριβείς κινήσεις ή ακόμα και να απαντάει σε ερωτήσεις. Ένα τέτοιο σύστημα περιλαμβάνει, συνήθως τις ακόλουθες συνιστώσες: ένα μηχανολογικό σύστημα, ένα σύστημα αίσθησης και ένα σύστημα ελέγχου.

Το μηχανολογικό υποσύστημα αποτελείται από μηχανισμούς που επιτρέπουν την κίνηση του ρομπότ, όπως για παράδειγμα οι αρθρώσεις, το σύστημα μετάδοσης κίνησης, οι επενεργητές-κινητήρες, οι οδηγοί, πνευματικά στοιχεία, και άλλα. Όλα μαζί συντελούν στη φυσική υπόσταση και το σώμα του ρομπότ.

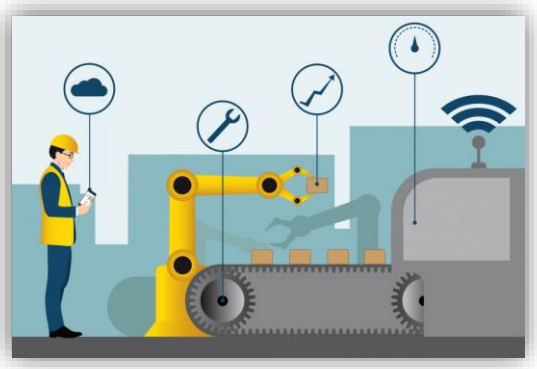
Το αισθητήριο υποσύστημα βοηθάει στη συλλογή πληροφοριών για την κατάσταση (state) των υποσυστημάτων, αλλά και του περιβάλλοντος. Εκτός των άλλων, δέχεται εξωτερικές εντολές, τις επεξεργάζεται, τις μεταφράζει σε ηλεκτρικά σήματα που θα σταλθούν τελικά στους κινητήρες των αρθρώσεων, καθώς, επίσης, παράγει σήματα εξόδου, που θα πληροφορούν το χρήστη για την κατάσταση του συστήματος. Στο υποσύστημα αίσθησης περιλαμβάνονται όργανα μέτρησης, αισθητήρες, ηλεκτρονικά στοιχεία και άλλα.

Το σύστημα ελέγχου αξιοποιεί της πληροφορίες από το υποσύστημα αίσθησης (ανατροφοδότηση: feedback) και δρα έτσι ώστε το ρομπότ να λειτουργεί αποτελεσματικά και με τον επιθυμητό τρόπο. Ο ελεγκτής του ρομπότ επιβλέπει και συντονίζει ολόκληρο το σύστημα, ενώ για τη σχεδίαση και την υλοποίηση του συνήθως απαιτείται ο συνδυασμός γνώσεων από πολλά γνωστικά πεδία όπως είναι η ρομποτική, ο αυτόματος έλεγχος, η τεχνητή νοημοσύνη και η επιστήμη των υπολογιστών.

Όμως, πολλές φορές είναι χρήσιμο ή και απαραίτητο ο προγραμματισμός ενός ρομπότ να είναι φιλικός προς τον μη-εξειδικευμένο χρήστη. Αυτό μπορεί να πραγματοποιηθεί με χρήση φωνητικών εντολών, και αυτό καθίσταται δυνατόν μέσα από επεξεργασία φυσικής γλώσσας (Natural Language Processing: NLP). Για παράδειγμα, οι ηλικιωμένοι άνθρωποι μπορεί να χρειάζονται στο σπίτι τους ένα αναπηρικό καρότσι που ελέγχεται με φωνητικές εντολές (Sivakumar et al., 2013), ένα κινητό ρομπότ που θα τους βοηθάει στις οικιακές δουλειές, ή θα τους υπενθυμίζει να παίρνουν τη φαρμακευτική τους αγωγή την κατάλληλη ώρα. Τα ρομπότ που επιδέχονται φωνητικές εντολές μπορούν επίσης να προσφέρουν ψυχαγωγία και ενημέρωση, τόσο σε ηλικιωμένους με άνοια όσο και σε μικρά παιδιά.

Στα πλαίσια της παρούσας εργασίας, η ουσιαστική χρησιμότητα του προγραμματισμού μέσω φυσικής γλώσσας έγκειται στο πεδίο της βιομηχανικής ρομποτικής. Η βιομηχανική ρομποτική είναι συγκεκριμένο πεδίο της ρομποτικής και ασχολείται αποκλειστικά με τη βιομηχανική παραγωγή. Τα βιομηχανικά ρομπότ είναι εξελιγμένα συστήματα αυτοματισμού που χρησιμοποιούν ηλεκτρονικό υπολογιστή σαν μια βασική συνιστώσα του ελέγχου τους. Σήμερα, οι υπολογιστές αποτελούν ένα βασικό μέρος του βιομηχανικού αυτοματισμού. Με την πάροδο του χρόνου, τόσο η έρευνα όσο και οι εφαρμογές της βιομηχανικής ρομποτικής πληθαίνουν και τώρα πια αποτελεί αναπόσπαστο κομμάτι της βιομηχανίας, συνεπώς δημιουργούνται συνεχώς νέες ανάγκες, απαιτήσεις και προβλήματα προς επίλυση. Δε θα μπορούσε να παραληφθεί η χρησιμότητα ενός ρομποτικού συστήματος σε μια βιομηχανία, καθώς υπάρχουν αρκετές βιομηχανικές διεργασίες που μπορούν να εκμεταλλευτούν τα

προϊόντα της παρούσας εργασίας. Για παράδειγμα, τυπικές εφαρμογές της βιομηχανικής ρομποτικής είναι η μεταφορά εργαλείων/τεμαχίων, η κατασκευή, βαφή, συγκόλληση, η κοπή, και εν γένει η κατεργασία κάποιου βιομηχανικού προϊόντος. Η τεχνολογία μετατροπής εντολών από φυσική γλώσσα σε γλώσσα συμβατή με το βραχίονα επιτρέπει τη διευκόλυνση του σύγχρονου (on-line) προγραμματισμού βιομηχανικών διεργασιών. Στο Σχήμα 1-1 φαίνεται ένα σκίτσο που δείχνει βιομηχανικούς ρομποτικούς βραχίονες καθώς και έναν μηχανικό που επιβλέπει και ελέγχει την όλη διάταξη εκμεταλλευόμενος την τεχνολογία και το βιομηχανικό διαδίκτυο των πραγμάτων (Industrial Internet of Things: IIoT).



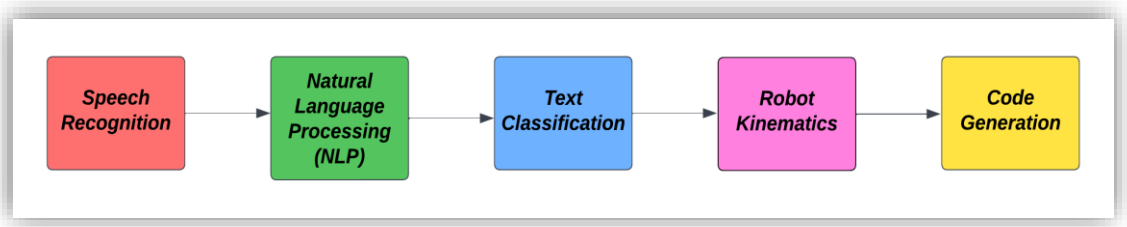
Σχήμα 1-1. Σκίτσο Βιομηχανικών Ρομπότ

1.1 Σκοπός Εργασίας

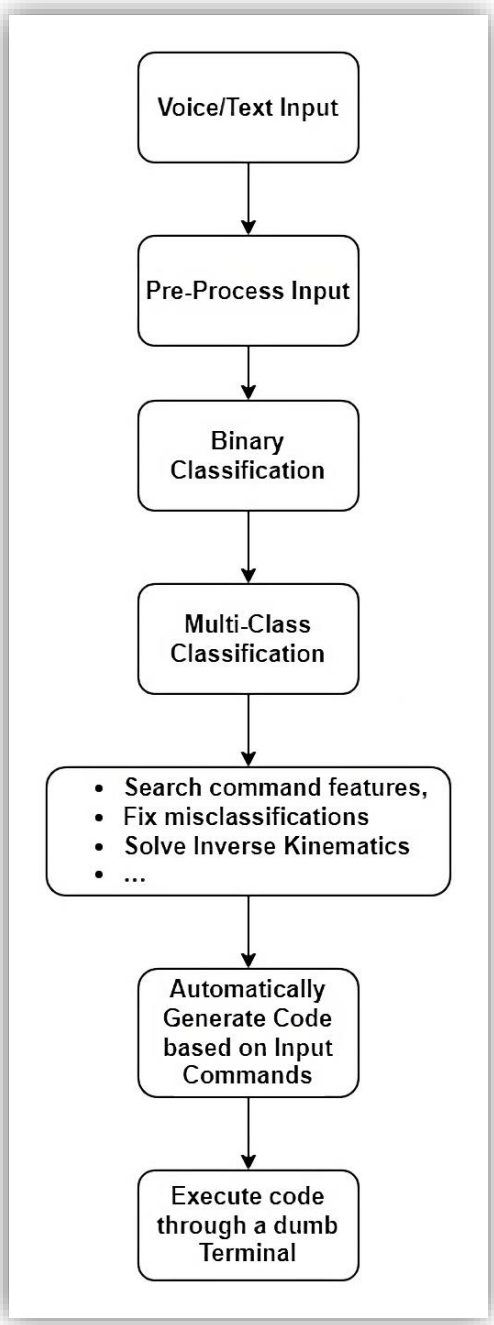
Η παρούσα μεταπτυχιακή εργασία επικεντρώνεται στην ολοκλήρωση συστήματος προγραμματισμού βιομηχανικού ρομποτικού βραχίονα μέσω φωνητικών σε φυσική γλώσσα. Ανάλογα με τη γλώσσα προγραμματισμού στην οποία ανταποκρίνεται ένας βραχίονας, μπορούν να απομονωθούν κατά το δοκούν χρήσιμες εντολές και να ενσωματωθεί γύρω τους ένα σύστημα που θα μετατρέπει τις εντολές του ανθρώπου από φυσική γλώσσα σε συμβατή με το βραχίονα γλώσσα. Σκοπός είναι οι φωνητικές οδηγίες που θα δίνονται στο βραχίονα να αναγνωρίζονται ευφυώς, να αντιστοιχίζονται αποτελεσματικά στις προεπιλεγμένες εντολές και να διευκολύνεται σημαντικά ο προγραμματισμός κινήσεων. Πέρα από την ευκολία που παρέχεται στο χρήστη, με αυτή την προσέγγιση μειώνεται αισθητά και ο χρόνος που απαιτείται για τον προγραμματισμό διεργασιών και για αυτό μπορεί να εφαρμοστεί από πολλές βιομηχανίες. Συνοπτικά ,η διαδικασία ροής του εκάστοτε σήματος φωνητικής εντολής έχει ως εξής:

1. Μετατροπή φωνής σε γραπτό κείμενο.
2. Προεπεξεργασία και συμμόρφωση κειμένου.
3. Ταξινόμηση γραπτού λόγου μέσα σε ένα πλήθος γνωστών εντολών στις οποίες υπακούει το ρομπότ.
4. Επίλυση αντίστροφης κινηματικής για επίτευξη επιθυμητών θέσεων και προσανατολισμών του τελικού στοιχείου δράσης.
5. Εξαγωγή των εντολών σε μορφή συμβατή με τον προγραμματισμό του ρομπότ.
6. Αποστολή των εντολών στον ελεγκτή (μέσω γραμμής εντολών Tera Term) και εκτέλεση του συνόλου των φωνητικών εντολών από τον ρομποτικό βραχίονα.

Στο Σχήμα 1-2 συνοψίζεται η γενική ιδέα που πραγματεύεται η παρούσα εργασία και στο Σχήμα 1-3 παρουσιάζονται ελαφρώς περισσότερες λεπτομέρειες της γενικής ιδέας.



Σχήμα 1-2. Γενική Ιδέα Εργασίας



Σχήμα 1-3. Εμβάθυνση στη Γενική Ιδέα

Συγκεκριμένα, η παρούσα εργασία πραγματεύεται τη δημιουργία μιας εφαρμογής φωνητικών εντολών για on-line προγραμματισμό του βιομηχανικού βραχίονα Staubli RX90L. Στα πλαίσια της παρούσας εργασίας, η γλώσσα στην οποία προγραμματίζεται ο υπό εξέταση βραχίονας ονομάζεται V+. Συγκεκριμένα, από τις πολυάριθμες εντολές της V+ επιλέξαμε να απομονώσουμε 7 εντολές για την επίτευξη των στόχων μας όπως θα εξηγηθεί στην Ενότητα 2.1.5. Από τις πολυάριθμες υπάρχουσες βιομηχανικές διεργασίες, δίνουμε βάση στον προγραμματισμό της διεργασίας παραλαβής και τοποθέτησής του προϊόντος (pick and place) η οποία αποτελεί αναπόσπαστο κομμάτι της βιομηχανικής παραγωγής.

Από τους σκοπούς που εξυπηρετεί το θέμα της παρούσας εργασίας γίνεται εμφανές ότι το σχετικό λεξιλόγιο που θα έχει νόημα να χρησιμοποιεί ο προγραμματιστής/χρήστης είναι, αν μη τι άλλο, περιορισμένο. Αυτό γιατί οι φωνητικές εντολές θα είναι χρήσιμο να σχετίζονται με εντολές εκτέλεσης κάποιας κίνησης ή περιστροφής προς κάποια συγκεκριμένη θέση με συγκεκριμένο προσανατολισμό στο χώρο. Παράλληλα, αν χρησιμοποιείται φωνητική είσοδος, είναι επιθυμητό και ίσως απαραίτητο ο χρήστης να απολαμβάνει μια ευελιξία στον τρόπο ομιλίας και στο λεξιλόγιό του. Γενικά θα πρέπει, στα πλαίσια αναγνώρισης εντολών, ο χρήστης να μη βρίσκεται σε διαρκή επιφυλακή για το ρυθμό ομιλίας του, ούτε να απαιτείται διαρκής παρακολούθηση για τυχών λάθη κατά τη μετατροπή φωνής σε κείμενο για την επίτευξη ενός στόχου. Είναι σημαντικό δηλαδή, όσο είναι δυνατόν, ο χρήστης να μπορεί να ομιλεί φυσικά σαν να απευθυνόταν σε έναν νοήμων άνθρωπο.

Κύριος στόχος είναι να αναπτυχθεί ευφυής τεχνική που θα μπορεί να μετατρέπει τις φωνητικές εντολές του ανθρώπου χειριστή/προγραμματιστή σε κινήσεις του ρομποτικού βραχίονα σε σχεδόν πραγματικό χρόνο. Είναι επίσης ζητούμενο να δημιουργηθεί και μια γραφική διεπαφή για το χρήστη (Graphical User Interface: GUI). Πέραν των μεμονωμένων εντολών, ένας ακόμα σκοπός της παρούσας εργασίας είναι ο χρήστης, κατ' επιλογήν, μέσα από τη διατύπωση λίγων φωνητικών εντολών που θα περιέχουν αριθμητικές τιμές, ο χρήστης να μπορεί να προγραμματίσει τη διαδικασία pick and place.

1.2 Βιβλιογραφική Ανασκόπηση

Ο έλεγχος ρομπότ μέσω φωνητικών εντολών έχει ερευνηθεί εκτεταμένα τις τελευταίες δεκαετίες. Η εργασία (McTear, 2002) βασίζεται σε ενσώματη γραμματική των δομών (Embodied Construction Grammar: ECG) και παρουσιάζει ένα ολοκληρωμένο σύστημα που παρέχει εντολές σε ρομποτικό προσομοιωτή, μέσω διεπαφής φυσικής γλώσσας. Η χρήση του ECG επιτρέπει βαθιά εννοιολογική κατανόηση, για αυτό επεκτείνεται εύκολα σε διαφορετικές γλώσσες (π.χ. Ισπανικά) και παρέχει ένα πλαίσιο ανάλυσης εννοιών όπως κίνηση και χωρικές συσχετίσεις. Η εργασία (Mooney, 2008) ασχολείται με τη διασύνδεση συμβόλων και υποβόσκουσας σημασίας λέξεων (grounded language acquisition). Οι ερευνητές (van Delden & Overcash, 2008) περιγράφουν ποιοτικά και ποσοτικά αποτελέσματα με χρήση των πακέτων Sphinx and MSAPI για τον προγραμματισμό της κίνησης βιομηχανικού ρομποτικού βραχίονα. Το 2010 (Muda et al., 2010) έγινε χρήση των φασματικών συντελεστών Mel Cepstral (MFCC's) ως τεχνική εξαγωγής χαρακτηριστικών (feature extraction), και της μεθόδου δυναμικής χρονικής στρέβλωσης (Dynamic Time Wrapping: DTW) ως τεχνική αντιστοίχισης χαρακτηριστικών (feature matching) ομιλίας. Οι ερευνητές στην εργασία (Tasevski et al., 2013) προσφέρουν λύση για την ενσωμάτωση βιομηχανικού ρομπότ ABB IRB140 σε σύστημα αυτόματης αναγνώρισης φωνής (ASR) και μηχανικής όρασης με αμφίδρομη επικοινωνία. Το ρομπότ ήταν σε θέση να αναγνωρίζει φωνή, αντικείμενα, σχήματα και χαρακτηριστικά

αντικειμένων, και να τα μετακινεί σύμφωνα με φωνητικές εντολές. Το άρθρο (Ondas et al., 2013) αφορά φωνητικό προγραμματισμό του βιομηχανικού ρομπότ SCORPIO και χρησιμοποιούνται ακουστικά μοντέλα βασισμένα σε εκπαιδευμένα κρυφά Μαρκοβιανά μοντέλα (Hidden Markov Models) με χρήση της βάσης δεδομένων SpeechDatE-SK. Επίσης περιλαμβάνει μοντέλο γλώσσας μικρού λεξιλογίου καθώς και βελτίωση ευρωστίας (robustness) αναγνώρισης ομιλίας μέσω μεθόδου φασματικής αφαίρεσης (spectral subtraction) και τροποποιημένου λειτουργικού πλαισίου LIMA (LIMA framework). Η εργασία (Matuszek et al., 2013) πραγματεύεται γραμματική ανάλυση φυσικής γλώσσας, εννοιολογική και θεματική ανάλυση, μεταφράζοντάς την σε εντολές που μπορούν να εφαρμοστούν και να εκτελεστούν από ρομποτικό βραχίονα. Η κύρια συνεισφορά αυτής της εργασίας είναι ότι το σύστημα μεταφράζει φυσική γλώσσας σε κατευθυντήριες εντολές σε άγνωστο (unseen) αρχικά για ρομπότ περιβάλλον μέσω εξαγωγής σχέσεων στα δεδομένα, και όχι μέσω προκαθορισμένης αντιστοίχισης φυσικής γλώσσας σε δράσεις του ρομπότ. Στην εργασία (Christodoulides, 2014) οι ερευνητές ορίζουν λεξικό και κανόνες γραμματικής για τον ορισμό εντολών που περιμένει ο ρομποτικός βραχίονας και παρουσιάζεται το λογισμικό Paaline (ανοιχτού κώδικα) που χρησιμοποιείται για διαχείριση, ανάλυση και οπτικοποίηση γλωσσικών δεδομένων. Στην εργασία (Howard et al., 2014) παρουσιάζεται το μοντέλο γραφήματος κατανεμημένης αντιστοίχισης (Distributed Correspondence Graph: DCG), το οποίο συμπεραίνει το πιο πιθανό σετ περιορισμών σχεδιασμού που αντιστοιχούν σε προκαθορισμένες εντολές φυσικής γλώσσας. Στη συνέχεια, ο σχεδιαστής τροχιάς χρησιμοποιεί τους περιορισμούς (planning constrains) ώστε να βρει μια αλληλουχία δράσεων που προσεγγίζουν τις εντολές εισόδου. Αξίζει να αναφερθεί ότι το 2015 σχεδιάστηκε ένα σύστημα οικιακής υποστήριξης ηλικιωμένων με βάση το εμπορικά διαθέσιμο πακέτο ASR (Panek & Mayer, 2015). Τα ρομπότ ήταν σε θέση να αναγνωρίσουν τη φωνή και την κατεύθυνση από την οποία του ομιλεί ο ηλικιωμένος χρήστης, ειδικά εντός δωματίου χωρίς θόρυβο και αντίλαλο. Οι ερευνητές (Rashid et al., 2017) ελέγχουν ένα ρομπότ μέσω αναγνώρισης φωνής με χρήση Arduino, συνδέοντας κινητό android με ένα ρομπότ-παιχνίδι μέσω Bluetooth. Η διάταξη εκτέλεσης κινήσεων βασίζεται κυρίως σε αισθητήρες και υλισμικό (hardware), και το σύστημα ομιλίας του ρομπότ απαιτεί εκ των προτέρων ηχογράφηση εντολών. Στην εργασία (Misra et al., 2017) εκπαιδεύεται ένα σύνθετο μοντέλο μηχανικής μάθησης και συνδυάζονται μοντέλα επιτηρούμενης μάθησης (Supervised Learning) με μοντέλα μάθησης μέσω ανταμοιβής (Reinforcement Learning). Το σύστημα χαρτογραφεί απευθείας τις εισόδους σε δράσεις, χρησιμοποιώντας ως είσοδο εικόνες και εννοιολογικά διασυνδεδεμένους συνδυασμούς γλώσσας, αντίληψης και κατανόησης εντολής σε μορφή κειμένων. Η έρευνα (Matuszek, 2018) επικεντρώνεται στη διατύπωση εκφράσεων μέσω μηχανικής μάθησης, επιτρέποντας στα ρομπότ να εκπαιδεύονται όσο αλληλοεπιδρούν με χρήστες ενώ παράλληλα λαμβάνουν σημασιολογικά δεδομένα γλώσσας για εντολές και αντικείμενα. Η εργασία (Saini & Joseph, 2022) προσφέρει μια επισκόπηση της μηχανικής μάθησης σε συνδυασμό με την επεξεργασία φυσικής γλώσσας και τη ρομποτική. Η εργασία (Mah et al., 2022) κινείται στη στάθμη της τεχνικής (state-of-the-art) και πραγματεύεται τη διασύνδεση μηχανικής μάθησης, επεξεργασίας φυσικής γλώσσας και το διαδίκτυο των πραγμάτων (Internet of Things: IoT) με σκοπό την επικοινωνία ανθρώπου-μηχανής προς την επίτευξη στόχων αλλά και τη διαχείριση της 4ης βιομηχανικής επανάστασης εν γένει.

Συγκριτικά με τη βιβλιογραφία, η θέση της παρούσας εργασίας δεν ανήκει στη στάθμη της τεχνικής στα επιμέρους πεδία που την πλαισιώνουν, καθώς δεν έχει ως στόχο την ανάπτυξη νέας μεθοδολογίας για επεξεργασία φυσικής γλώσσας. Φυσικά, υπάρχουν σημεία όπου

διακρίνονται οι ιδιαιτερότητες και οι πρωτοτυπίες του εξεταζόμενου συστήματος. Για παράδειγμα, μια ιδιαιτερότητα έγκειται στη διερεύνηση ως προς τον τρόπο κατηγοριοποίησης των προτάσεων η οποία πραγματοποιείται σε 2 στάδια. Επίσης, η εργασία πρωτοτυπεί ως προς την ανάπτυξη της διεπαφής, η οποία παρουσιάζει δυνατότητα γενίκευσης για διαφορετικό ρομποτικό βραχίονα, ικανότητα διόρθωσης αριθμητικών σφαλμάτων ή σφαλμάτων κατηγοριοποίησης. Ακόμα, τονίζεται πως η γραφική διεπαφή προσφέρει δυνατότητα προγραμματισμού του βραχίονα σε επίπεδο διαδικασίας και όχι μεμονωμένων εντολών. Με βάση τη μεθοδολογία που υλοποιήθηκε, παρατηρείται πως μπορεί να προγραμματιστεί οποιαδήποτε ολοκληρωμένη βιομηχανική διεργασία, ανάλογα το βαθμό πολυπλοκότητας. Μέσω αυτής της παρατήρησης καθίσταται εμφανής η προσφορά της παρούσας εργασίας στο πεδίο της βιομηχανικής ρομποτικής. Ως παράδειγμα βιομηχανικής εφαρμογής επιλέχθηκε η διεργασία παραλαβής και τοποθέτησης αντικειμένων (pick and place).

1.3 Δομή Εργασίας

Στο Κεφάλαιο 2 γίνεται μια αναφορά στα στοιχεία θεωρίας που αφορούν την περιγραφή του ρομποτικού βραχίονα Staubli RX90L, της κινηματικής, του χειριστηρίου και της γλώσσας V^+. Στη συνέχεια, πιστοποιείται η επίλυση αντίστροφης κινηματικής, και περιγράφεται η φιλοσοφία της διαδικασίας pick and place. Έπειτα, περιγράφονται στοιχεία αναγνώρισης ομιλίας, επεξεργασίας φυσικής γλώσσας, στατιστικής και τη δυαδικής κατηγοριοποίησης.

Ο ρομποτικός βραχίονας Staubli RX90L περιγράφεται όσον αφορά τη γεωμετρία, τα τεχνικά χαρακτηριστικά, τις αρθρώσεις και τον ελεγκτή του. Περιγράφεται η ευθεία κινηματική σύμφωνα με την τροποποιημένη μέθοδο Denavit-Hartenberg, αλλά και τα αποτελέσματα της επίλυσης της αντίστροφης κινηματικής. Ακόμα, παρουσιάζονται κάποια στοιχεία που αφορούν το τηλεχειριστήριο (teach pendant), καθώς και κάποιες βασικές εντολές που αφορούν προγραμματισμό σε γλώσσα V^+. Στη συνέχεια παρουσιάζονται κάποια αποτελέσματα με σκοπό την πιστοποίηση της επίλυσης αντίστροφης κινηματικής. Έπειτα, περιγράφεται η διαδικασία pick and place, αναφέρονται τα κυριότερα πλεονεκτήματα και μειονεκτήματα αυτής της βιομηχανικής διεργασίας, και παρουσιάζονται αποτελέσματα που σχετίζονται με προσομοίωση κίνησης για εκτέλεση της συγκεκριμένης διεργασίας.

Όσον αφορά την αναγνώριση ομιλίας, περιγράφονται εποπτικά κάποιες ενδεικτικές τεχνικές και αλγόριθμοι, και γίνεται μια ιστορική αναδρομή από την ανάπτυξή της. Επίσης γίνεται αναφορά σε εύχρηστες και δημοφιλείς υπηρεσίες που προσφέρονται για εφαρμογές αναγνώρισης ομιλίας σε πραγματικό χρόνο, οι οποίες χρησιμοποιούν αντίστοιχες μεθόδους και αλγορίθμους, όπως η υπηρεσία Microsoft Speech Software Development Kit (SDK)

Στη συνέχεια γίνεται μια αναφορά στα στοιχεία θεωρίας της επεξεργασίας φυσικής γλώσσας (NLP), στην ιστορία της, στα συστατικά της καθώς και στις εφαρμογές αυτού του πεδίου, ιδίως στη βιομηχανική ρομποτική. Ακόμα, γίνεται αναφορά στους τρόπους αναπαράστασης λέξεων και στις μεθοδολογίες που χρησιμοποιούνται για την ταξινόμηση κειμένων. Οι αλγόριθμοί ταξινόμησης εν γένει μπορεί να υπάγονται τόσο στη γλωσσολογία όσο και στη στατιστική, στη μηχανική μάθηση (Machine Learning) και σε νευρωνικά δίκτυα βαθιάς μάθησης (Deep Learning Neural Networks). Τέλος, γίνεται μια αναφορά σε στοιχεία στατιστικής, έλεγχο υποθέσεων, δυαδική κατηγοριοποίηση SVM και τους δείκτες αξιολόγησης μοντέλων.

Στο Κεφάλαιο 3 περιγράφεται η διαδικασία που ακολουθήθηκε για την ολοκλήρωση του συστήματος, σε όλο το εύρος πραγμάτωσής της. Μετά την εισαγωγή, αρχικά περιγράφεται η φιλοσοφία που ακολουθήθηκε για την κατηγοριοποίηση των προτάσεων εισόδου. Έπειτα, περιγράφεται η λειτουργία της φιλοσοφίας των τμημάτων κώδικα που εκτελούν βασικά βήματα και παρουσιάζονται κατατοπιστικά παραδείγματα με σκοπό τη διευκόλυνση της κατανόησης του αναγνώστη.

Στη συνέχεια, παρουσιάζεται η γραφική διεπαφή (GUI) που κατασκευάστηκε με σκοπό τη συγκρότηση της όλης διαδικασίας υπό μορφή εφαρμογής, φιλική προς το χρήστη. Η εφαρμογή περιλαμβάνει λειτουργία pick and place για τον εύκολο προγραμματισμό εκτέλεσης της συγκεκριμένης βιομηχανικής διεργασίας. Η ροή του προγράμματος pick and place μοιάζει με αυτή που ήδη αναφέρθηκε, αλλά υπάρχουν κάποιες διαφορές όπως το ότι δεν απαιτείται κατηγοριοποίηση εντολών. Απαιτούνται όμως λέξεις-κλειδιά για τον ορισμό των διαφορετικών επιθυμητών διατάξεων.

Στο Κεφάλαιο 4 γίνεται μια σύνοψη και αναφέρονται τα συμπεράσματα που προκύπτουν από την παρούσα εργασία. Επίσης γίνονται προτάσεις για μελλοντικές εργασίες που αφορούν το εξεταζόμενο θέμα.

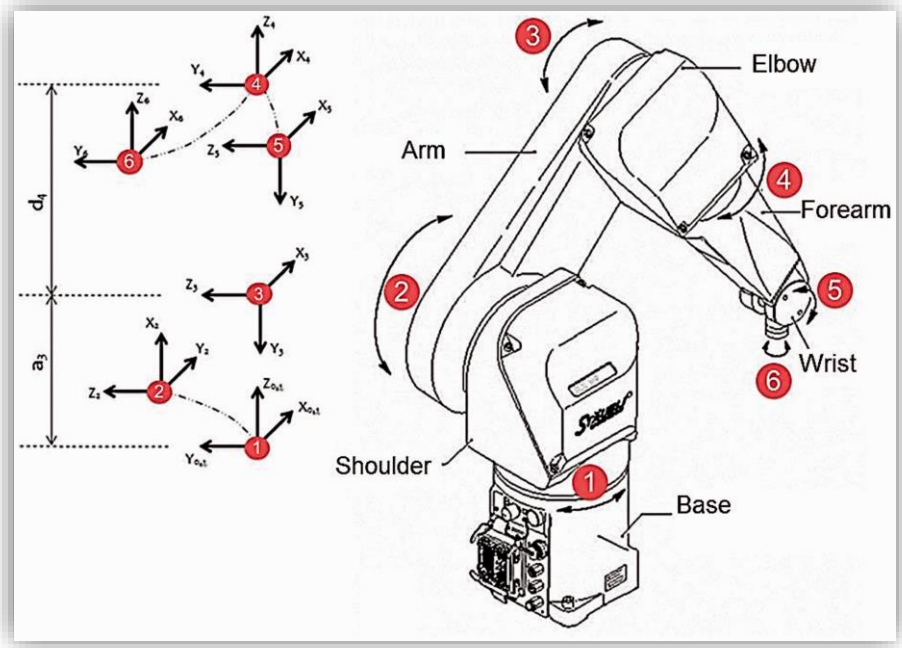
2 Στοιχεία Θεωρίας

2.1 Ρομποτικός Βραχίονας Staubli RX90L

2.1.1 Περιγραφή

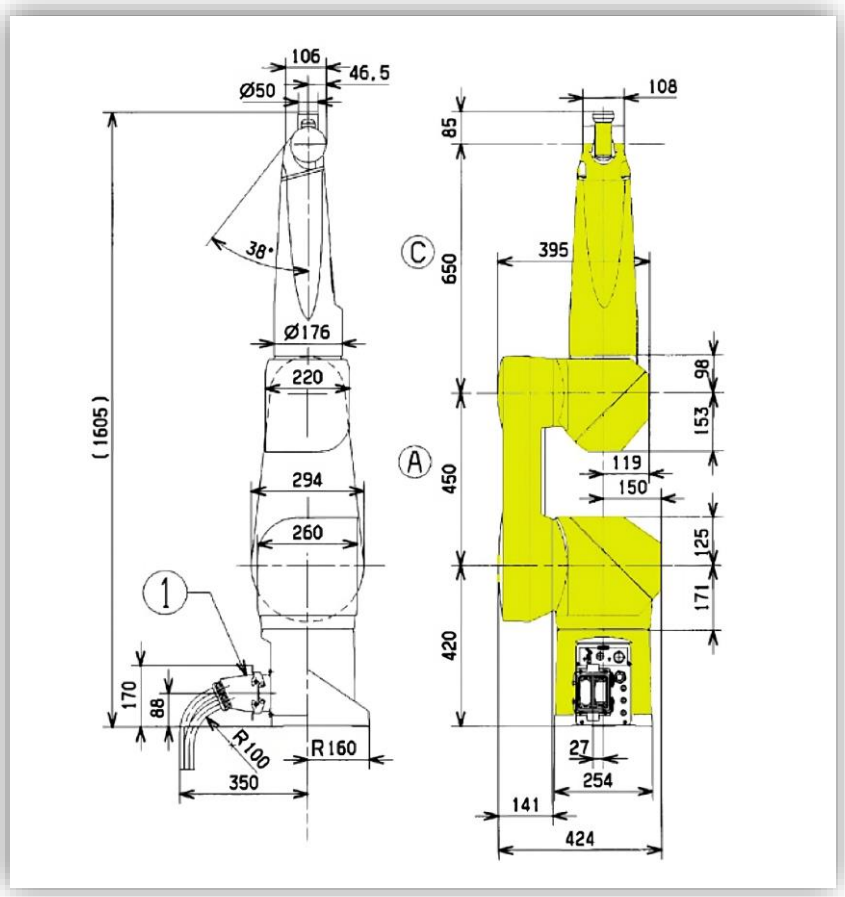
Ο ρομποτικός βραχίονας Staubli RX90L είναι ένα ρομπότ βιομηχανικού τύπου με μεγάλη ακρίβεια κίνησης, ιδανικό για την εκτέλεση πολλών βιομηχανικών διεργασιών. Για παράδειγμα, ο συγκεκριμένος βραχίονας είναι ιδανικός για διεργασίες τύπου pick and place, φινίρισμα, συγκόλληση και συναρμολόγηση προϊόντων. Παραδείγματα βιομηχανιών όπου χρησιμοποιούνται βραχίονες της εταιρίας Staubli είναι οι εξής: αυτοκινητοβιομηχανία, μεταλλουργία, κατασκευή φωτοβολταϊκών και ημιαγωγών, βιομηχανία αναψυκτικών και τροφίμων, φαρμακοβιομηχανία, ιατρική κ.α. Η κίνηση των αρθρώσεων του ρομπότ παράγεται από κινητήρες χωρίς ψήκτρες (brushless motors) συζευγμένους με γωνιοαναλυτές (resolvers). Αυτή η εύρωστη και αξιόπιστη συναρμολόγηση, σε συνδυασμό με ένα σύστημα καταμέτρησης, επιτρέπει τη γνώση της απόλυτης θέσης του ρομπότ οποιαδήποτε στιγμή. Είναι αρκετά ευέλικτος και είναι σε θέση να εκτελέσει μια ποικιλία εφαρμογών, όπως χειρισμός φορτίων (handling of loads), συναρμολόγηση (assembly), διαδικασίες κόλλησης σφαιριδίων (application of adhesive beads), ελέγχου/επιβεβαίωσης (control/check), καθώς και εφαρμογές καθαρών χώρων (clean room applications).

Ο βραχίονας Staubli RX90L αποτελείται από 6 μέλη (συνδέσμους) που συνδέονται ανά δύο μέσω 6 περιστροφικών αρθρώσεων (6R) όπως φαίνεται στο Σχήμα 2-1. Σημειώνεται ότι, καθώς ο συγκεκριμένος βραχίονας είναι ανθρωπομορφικός (articulated robot), οι σύνδεσμοι ονομάζονται όπως τα αντίστοιχα μέλη του ανθρώπινου χεριού (Μίχας & Michas, 2016).



Σχήμα 2-1. 3D Αναπαράσταση Staubli RX90L

Η συναρμολόγηση του ρομπότ εμπεριέχει τα μοτέρ, τα φρένα, μηχανισμούς μετάδοσης κίνησης, δεσμίδες καλωδίων, πνευματικά και ηλεκτρικά κυκλώματα, καθώς επίσης και το σύστημα αντιστάθμισης. Το συναρμολόγημα του βραχίονα αποτελείται από σταθερή και εγκιβωτισμένη δομή (προστασία IP με το πρότυπο NF EN 60529), για να το προστατεύει από εξωτερικές παρεμβάσεις. Ο σχεδιασμός του βασίζεται σε μονάδες μετάδοσης: JCS (Staubli Combined Joint) χρησιμοποιούμενες στις αρθρώσεις 1, 2, 3 και 4. Ο καρπός (wrist) αποτελείται από τις αρθρώσεις 5 και 6. Οι διαστάσεις του Staubli RX90L φαίνονται στο Σχήμα 2-2, και ο Πίνακας 2-1 δείχνει τα τεχνικά χαρακτηριστικά.



Σχήμα 2-2. Διαστάσεις Staubli RX90L

Πίνακας 2-1. Τεχνικά Χαρακτηριστικά Staubli RX90L

Εύρος Έκτασης (mm)	1185
Μάζα (kg)	112
Ονομαστικό Φορτίο (kg)	3.5
Μέγιστο Φορτίο (kg) (nominal speed)	6
Μέγιστη Ταχύτητα Εργασίας (m/s)	11
Βαθμοί Ελευθερίας	6
Ελεγκτής	CS7
Εγκατάσταση	Πάτωμα/Οροφή

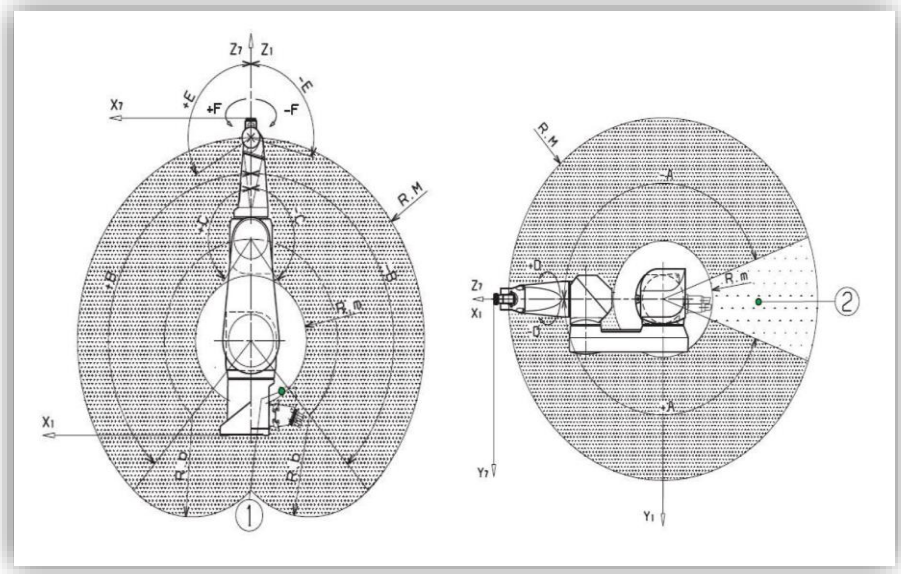
Η κάθε άρθρωση έχει το δικό της κινητήρα και περιστρέφεται ανεξάρτητα εντός των ορίων της, περιστρέφοντας το σύνδεσμο που έπεται αυτής. Ο Πίνακας 2-2 δείχνει τα όρια όλων των αρθρώσεων, τις αρχικές γωνίες (Home Configuration) και τα όρια γωνιακής ταχύτητας.

Πίνακας 2-2. Αρθρώσεις του Staubli RX90L

Άρθρωση	1	2	3	4	5	6
Home (°)	0	-90	90	0	0	0
Όρια (°)	±160	-227.5,45.5	-52.5,232.5	±270	-105,120	±270
Όρια (°/sec)	±177	±150	±143	±201	±160	±390

Τα όρια των αρθρώσεων καθορίζουν το χώρο εργασίας του βραχίονα (Work Space) δηλαδή την περιοχή του τρισδιάστατου χώρου εντός της οποίας μπορεί να βρεθεί το τελικό στοιχείο δράσης (end-effector). Στο Σχήμα 2-3 φαίνεται ο χώρος εργασίας του βραχίονα Staubli RX90L. Προς αποφυγή ιδιομορφιών (singularities) πρέπει να πληρούνται οι παρακάτω συνθήκες:

- Μέγιστη απόσταση μεταξύ αρθρώσεων 2 και 5 (R.M):1100 mm
- Ελάχιστη απόσταση μεταξύ αρθρώσεων 2 και 5 (R. M):401 mm
- Απόσταση μεταξύ αρθρώσεων 3 και 5 (R.b): 650 mm



Σχήμα 2-3. Χώρος Εργασίας του Staubli RX90L

Η καμπίνα του ελεγκτή ελέγχει το ρομπότ μέσω ενισχυτών ισχύος που επικοινωνούν με κάθε άρθρωση του βραχίονα (Μαραγκός & Maragkos, 2018). Η παρεμβολή που χρησιμοποιείται είναι γραμμική. Η ηλεκτρική ενέργεια μετατρέπεται από ένα μπλοκ τροφοδοσίας, το οποίο τροφοδοτεί σε καθένα από τα παραπάνω στοιχεία τις κατάλληλες τάσεις που απαιτούνται για τη σωστή λειτουργία, ξεκινώντας από τριφασική τάση που παρέχεται από το ηλεκτρικό δίκτυο. Τα απαραίτητα καλώδια για την ασφάλεια τοποθετούνται στην πλακέτα ασφαλείας. Όλα αυτά τα στοιχεία λειτουργούν σε κλειστό χώρο, και ο έλεγχος της θερμοκρασίας εξασφαλίζεται από έναν εναλλάκτη θερμότητας αέρα προς αέρα. Δύο μονάδες ανεμιστήρα ψύχουν τα κυκλώματα του ελεγκτή. Το εσωτερικό του CS7 φαίνεται στο Σχήμα 2-4 και το μπροστινό πάνελ του ελεγκτή φαίνεται στο Σχήμα 2-5.

Σχήμα 2-4. Εσωτερικό του CS7

Ο ελεγκτής έχει κατασκευαστεί με αρχιτεκτονική VME (Versa Module Europe bus) και περιλαμβάνει:

- Μονάδα επεξεργασίας τους συστήματος (CPU 030) για έλεγχο.
- Περιφερειακή μονάδα διαχείρισης (SIO), η οποία ελέγχει τον σκληρό δίσκο, τα σήματα εισόδου/εξόδου, τις σειριακές συνδέσεις γενικού σκοπού καθώς και την επικοινωνία με το χειριστήριο του ελεγκτή και τον πίνακα ελέγχου.
- Μονάδα κοινής διεπαφής (MI6), με ενισχυτές ελέγχου των κινητήρων.
- Μονάδα απόλυτης βαθμονόμησης (ACB), όταν ενεργοποιείται το ρομπότ.

Η πλακέτα ασφαλείας περιλαμβάνει όλα τα σήματα και τα εξαρτήματα που διασφαλίζουν τη λειτουργία της εγκατάστασης, κάτω από κατάλληλες συνθήκες ασφαλείας.
Τα χαρακτηριστικά είναι τα ακόλουθα:

- Ελεγκτής τύπου CS7 ελεγχόμενος από έναν 68030 μικροεπεξεργαστή (40 MHZ) και έναν 68882 συνεπεξεργαστή (33 MHZ).
- VME bus (Versa Module Europe bus).
- Μνήμη RAM 4 Mbytes.
- Σκληρός δίσκος των 85 Mbytes το ελάχιστο.
- 3 σκληροί δίσκοι συμβατοί με υπολογιστή (1/2¨).
- 12 είσοδοι /6 έξοδοι από τις οποίες 3 υψηλής ταχύτητας είσοδοι (1msec).
- 4 RS232C/1 RS422/485 σειριακά καλώδια.
- 1 πρότυπο τερματικό οθόνης.
- Χειριστήριο του ρομποτικού βραχίονα.
- Καλώδιο διασύνδεσης του ρομποτικού βραχίονα με τον ελεγκτή (5/10 μέτρα).

Σχήμα 2-5. Μπροστινό Πάνελ του Ελεγκτή

2.1.2 Ευθεία Κινηματική

Η κινηματική είναι η επιστήμη που μελετά την κίνηση των σωμάτων χωρίς να λαμβάνει υπόψιν την αδράνεια και τις δυνάμεις που την προκαλούν. Στη ρομποτική, η κινηματική μελετά την κίνηση μεταξύ των συνδέσμων που απαρτίζουν το ρομπότ, οι οποίοι συνδέονται κινηματικά μέσω των αρθρώσεων. Οι αρθρώσεις εν γένει μπορεί να είναι πρισματικές (διαμήκεις), περιστροφικές ή σφαιρικές. Πιο συγκεκριμένα, στόχος είναι η εξαγωγή της σχέσης μεταξύ των γωνιών των αρθρώσεων με τη θέση και τον προσανατολισμό του τελικού στοιχείου δράσης. Η κινηματική απαρτίζεται από δύο προβλήματα:

1. Την Ευθεία Κινηματική μέσω της οποίας υπολογίζεται η θέση και ο προσανατολισμός του τελικού στοιχείου δράσης για δεδομένες γωνίες αρθρώσεων.
2. Την Αντίστροφη Κινηματική μέσω της οποίας υπολογίζονται οι γωνίες των αρθρώσεων για δεδομένη θέση και προσανατολισμό του τελικού στοιχείου δράσης. Η αντίστροφη κινηματική είναι πολύ πιο δύσκολη και υπολογιστικά απαιτητική για αρκετούς λόγους που θα εξηγηθούν στη συνέχεια.

Για την επίλυση της ευθείας κινηματικής χρησιμοποιείται ευρέως η τροποποιημένη μέθοδος Denavit-Hartenberg (DH) η οποία αναπτύχθηκε για την περιγραφή της κινηματικής ενός σύνθετου μηχανισμού μέσα από 4 παραμέτρους (Craig, 2005). Η μέθοδος DH δίνει υπό μορφή 4x4 ομογενών μετασχηματισμών, τη θέση και τον προσανατολισμό ενός συνδέσμου (που κινείται από κάποια άρθρωση) ως προς τον προηγούμενο. Ο διαδοχικός (από δεξιά) πολλαπλασιασμός των επιμέρους ομογενών μετασχηματισμών δίνει το συνολικό μετασχηματισμό από τη βάση στο τελικό στοιχείο δράσης. Στην παρούσα εργασία γίνεται χρήση της τροποποιημένης μεθόδου (mDH) η οποία είναι μια επέκταση της αρχικής για πιο περίπλοκες συνδεσμολογίες, προσφέρει μεγαλύτερη ευελιξία στον ορισμό των επιμέρους συστημάτων συντεταγμένων, απλοποιεί την ανάλυση και μειώνει τα σφάλματα. Γενικά, η

μέθοδος mDH δίνει μια ισχυρή και ευέλικτη μέθοδο για μοντελοποίηση και έλεγχο ρομποτικών συστημάτων, επιτρέποντας ακριβή έλεγχο της κίνησης και της συνολικής συμπεριφοράς τους. Κατά την εφαρμογή της τροποποιημένης μεθόδου mDH ακολουθείται η εξής διαδικασία:

1. Προσάρτηση Συστημάτων Συντεταγμένων: Ένα σύστημα συντεταγμένων (ΣΣ) ορίζεται για κάθε σύνδεσμο (Link) και προσαρτάται πάνω στην αντίστοιχη άρθρωση (Joint). Τα συστήματα συντεταγμένων ορίζονται με χρήση συνδυασμού μετασχηματισμών περιστροφής και μετακίνησης στον καρτεσιανό χώρο.
2. Παραμετροποίηση Αρθρώσεων: Κάθε άρθρωση παραμετροποιείται βάσει του τύπου της (πρισματική-περιστροφική-σφαιρική), του άξονα περιστροφής και της απόστασής της από τον άξονα της προηγούμενης άρθρωσης.
3. Παραμετροποίηση Συνδέσμων: Κάθε σύνδεσμος παραμετροποιείται με βάση το μήκος του, τη στροφή του (twist: περιστροφή γύρω από τον άξονα-x του προηγούμενου ΣΣ), την απόστασή του από το προηγούμενο ΣΣ κατά μήκος του κοινού άξονα-x, και της περιστροφής (σε σχέση με το προηγούμενο ΣΣ) γύρω από τον υπό εξέταση άξονα-z.
4. Ομογενείς Μετασχηματισμοί: Με χρήση των παραμέτρων των αρθρώσεων και των συνδέσμων, εξάγεται ένα σύνολο ομογενών μετασχηματισμών μεταξύ των διαδοχικών ΣΣ λόγω κίνησης ή περιστροφής των αρθρώσεων.
5. Ευθεία Κινηματική: Τελικά, η θέση και ο προσανατολισμός του τελικού εργαλείου δράσης υπολογίζονται μέσω πολλαπλασιασμού (από δεξιά) των επιμέρους ομογενών μετασχηματισμών. Ο πολλαπλασιασμός αυτός δίνει έναν ολικό μετασχηματισμό από το παγκόσμιο και σταθερό $ΣΣ_0$ προς το σύστημα συντεταγμένων του τελικού στοιχείου δράσης.

Οι παράμετροι για την κατασκευή του πίνακα mDH είναι οι εξής:

- Μήκη συνδέσμων a_{i-1} : απόσταση z_{i-1}, z_i κατά το x_{i-1}.
- Στρέψεις συνδέσμων α_{i-1} : γωνία z_{i-1}, z_i γύρω από το x_{i-1}.
- Μετατοπίσεις συνδέσμων d_i : απόσταση x_{i-1}, x_i κατά το z_i.
- Γωνίες αρθρώσεων θ_i : γωνία x_{i-1}, x_i γύρω από το z_i.

Αφού υπολογιστούν οι παράμετροι για κάθε σύνδεσμο και άρθρωση, ο μετασχηματισμός ανάμεσα στο $ΣΣ_{i-1}$ προς το $ΣΣ_i$ είναι ο εξής:

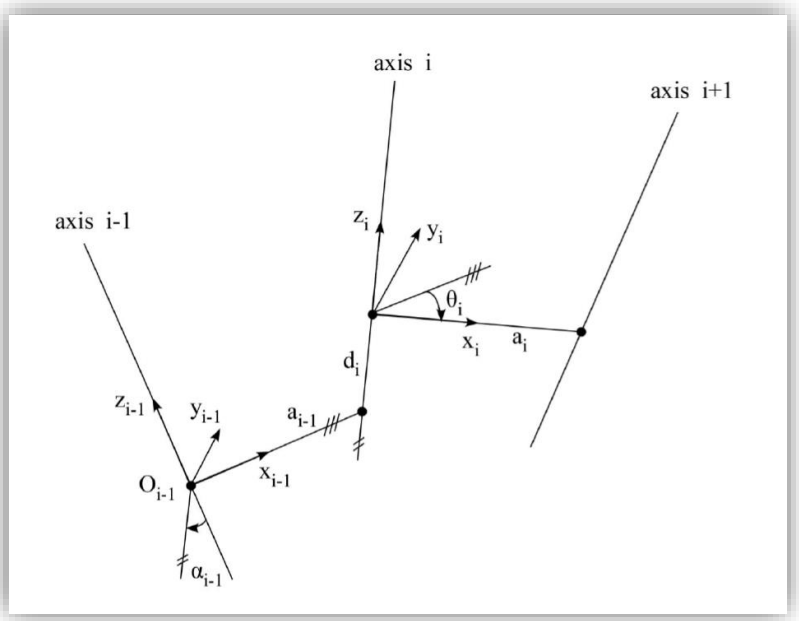
$$T_i^{i-1} = Rot(x_{i-1}, \alpha_{i-1}) * Trans(x_{i-1}, a_{i-1}) * Rot(z_i, \theta_i) * Trans(z_i, d_i) =$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\alpha_{i-1}) & -sin(\alpha_{i-1}) & 0 \\ 0 & sin(\alpha_{i-1}) & cos(\alpha_{i-1}) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_{i-1} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos(\theta_i) & -sin(\theta_i) & 0 & 0 \\ sin(\theta_i) & cos(\theta_i) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \Leftrightarrow$$

$$\Leftrightarrow \boldsymbol{T_i^{i-1}} = \begin{bmatrix} \boldsymbol{cos(\theta_i)} & \boldsymbol{-sin(\theta_i)} & \mathbf{0} & \mathbf{a_{i-1}} \\ \boldsymbol{cos(\alpha_{i-1})\, sin(\theta_i)} & \boldsymbol{cos(\alpha_{i-1})\, cos(\theta_i)} & \boldsymbol{-sin(\alpha_{i-1})} & \boldsymbol{-sin(\alpha_{i-1}) d_i} \\ \boldsymbol{sin(\alpha_{i-1})\, sin(\theta_i)} & \boldsymbol{sin(\alpha_{i-1})\, cos(\theta_i)} & \boldsymbol{cos(\alpha_{i-1})} & \boldsymbol{cos(\alpha_{i-1})\, d_i} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix}$$

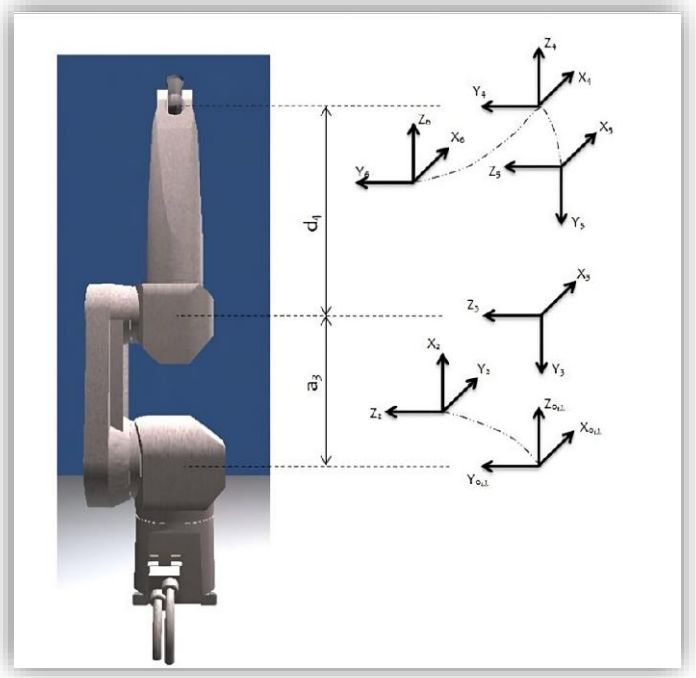
2-1

Ο ολικός ομογενής μετασχηματισμός υπολογίζεται από τον πολλαπλασιασμό των επιμέρους μετασχηματισμών ως: $T_N^0 = T_1^0 T_2^1 \dots T_N^{N-1}$. Τα μεγέθη που χρησιμοποιούνται στη μεθοδολογία mDH αναπαρίστανται γραφικά στο Σχήμα 2-6 (Μίχας & Michas, 2016).



Σχήμα 2-6. Σχηματική Αναπαράσταση τροποποιημένης Denavit-Hartenberg

Στο Σχήμα 2-7 φαίνεται ξανά η διάταξη του Staubli RX90L καθώς και η τοποθέτηση των επιμέρους συστημάτων συντεταγμένων. Βλέπουμε πως για $q_1=0$ το $\Sigma\Sigma_1$ συμπίπτει με το ακλόνητο παγκόσμιο σύστημα συντεταγμένων $\Sigma\Sigma_0$, και ότι το $\Sigma\Sigma_6$ αποτελεί το σύστημα συντεταγμένων της φλάντζας όπου προσάπτεται το τελικό στοιχείο δράσης. Οι διαστάσεις των συνδέσμων δίνονται από τον κατασκευαστή ως $a_3=450$mm και $d_4=650$mm, ενώ το μήκος του τελικού στοιχείου δράσης είναι 85mm.



Σχήμα 2-7. Συστήματα Συντεταγμένων για τον Staubli RX90L

Ο Πίνακας 2-3 είναι ο πίνακας mDH που προκύπτει από εφαρμογή της μεθοδολογίας.

Πίνακας 2-3. Πίνακας mDH για τον βραχίονα Staubli RX90L

Link i	a_{i-1}	α_{i-1}	d_i	θ_i
1	0	0	0	θ_1
2	0	-90	0	θ_2
3	a_3	0	0	θ_3
4	0	90	d_4	θ_4
5	0	-90	0	θ_5
6	0	90	0	θ_6

Στη συνέχεια, με χρήση της εξίσωσης 2-1 δημιουργήθηκαν οι επιμέρους μήτρες ομογενούς μετασχηματισμού μεταξύ των συνδέσμων.

$$A_1^0 = \begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad A_2^1 = \begin{bmatrix} c_2 & -s_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -s_2 & -c_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_3^2 = \begin{bmatrix} c_3 & -s_3 & 0 & a_3 \\ s_3 & c_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad A_4^3 = \begin{bmatrix} c_4 & -s_4 & 0 & 0 \\ 0 & 0 & -1 & -d_4 \\ s_4 & c_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_5^4 = \begin{bmatrix} c_5 & -s_5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -s_5 & -c_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad A_6^5 = \begin{bmatrix} c_6 & -s_6 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ s_6 & c_6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Ο ομογενής μετασχηματισμός του τελικού εργαλείου δράσης ως προς τη βάση προκύπτει:

$$T_6^0 = T_1^0 T_2^1 T_3^2 T_4^3 T_5^4 T_6^5 = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$ όπου:

2-2

$$r_{11} = c_1(c_{23}(c_4c_5c_6 - s_4s_6) + s_{23}c_5c_6) - s_1(s_4c_5c_6 + c_4s_6)$$

$$r_{12} = c_1(c_{23}(-c_4c_5s_6 - s_4c_6) + s_{23}s_5s_6) + s_1(s_4c_5s_6 - c_4c_6)$$

$$r_{13} = c_1(c_{23}c_4s_5 + s_{23}c_5) - s_1s_4s_5$$

$$r_{21} = c_1(c_4s_6 + s_4c_5c_6) + s_1(c_{23}(c_4c_5c_6 - s_4s_6) - s_{23}s_5c_6)$$

$$r_{22} = c_1(c_4c_6 - s_4c_5s_6) + s_1(c_{23}(-c_4c_5s_6 - s_4c_6) + s_{23}s_5s_6)$$

$$r_{23} = c_1s_4s_5 + s_1(c_{23}c_4s_5 + s_{23}c_5)$$

$$r_{31} = -c_{23}s_5c_6 + s_{23}(s_4s_6 - c_4c_5c_6)$$

$$r_{32} = c_{23}s_5s_6 + s_{23}(s_4c_6 + c_4c_5s_6)$$

$$r_{33} = c_{23}c_5 - s_{23}c_4s_5$$

$$p_x = c_1(d_4s_{23} + a_3c_2)$$

$$p_y = s_1(d_4s_{23} + a_3c_2)$$

$$p_z = d_4c_{23} - a_3s_2$$

2.1.3 Αντίστροφη Κινηματική

Όπως αναφέρθηκε, το πρόβλημα της αντίστροφης κινηματικής είναι πιο δύσκολο από την επίλυση της ευθείας κινηματικής, και συνήθως υπάρχουν πολλαπλές λύσεις για δεδομένη θέση και προσανατολισμό του τελικού στοιχείου δράσης. Παρόλα αυτά, σε βραχίονες με 6 βαθμούς ελευθερίας, εκ των οποίων οι τρείς τελευταίοι είναι περιστροφικοί με άξονες που τέμνονται στο ίδιο σημείο, υπάρχει κλειστή λύση (μέθοδος απόπλεξης/αποσύζευξης του Pieper). Σύμφωνα με τη μέθοδο αυτή, το πρόβλημα διασπάται σε δύο απλούστερα προβλήματα, ένα πρόβλημα υπολογισμού θέσης (υπολογισμός των q_1, q_2, q_3) και ένα πρόβλημα υπολογισμού του προσανατολισμού (υπολογισμός των q_4, q_5, q_6). Αυτά ισχύουν αν δε λάβουμε υπόψιν τις διαστάσεις του τελικού εργαλείου και το μετασχηματισμό από την 6η άρθρωση προς αυτό. Έπειτα από αναλυτική επίλυση της αντίστροφης κινηματικής (Μίχας & Michas, 2016), οι γωνίες των αρθρώσεων (όπου $\theta_i \leftrightarrow q_i$) για δεδομένη θέση και προσανατολισμό που προέκυψαν είναι οι εξής:

$$\boldsymbol{\theta_1} = \begin{cases} \boldsymbol{atan2}(\boldsymbol{p_y}, \boldsymbol{p_x}) \\ \boldsymbol{atan2}(-\boldsymbol{p_y}, -\boldsymbol{p_x}) \end{cases}$$

2-3

$$\boldsymbol{\theta_3} = \begin{cases} \boldsymbol{asin}\left(\frac{\boldsymbol{p_x^2}+\boldsymbol{p_y^2}+\boldsymbol{p_z^2}-\boldsymbol{a_3^2}-\boldsymbol{d_4^2}}{2\mathbf{a_3}\boldsymbol{d_4}}\right) \\ -\boldsymbol{asin}\left(\frac{\boldsymbol{p_x^2}+\boldsymbol{p_y^2}+\boldsymbol{p_z^2}-\boldsymbol{a_3^2}-\boldsymbol{d_4^2}}{2\mathbf{a_3}\boldsymbol{d_4}}\right) \\ \boldsymbol{\pi} + \boldsymbol{asin}\left(\frac{\boldsymbol{p_x^2}+\boldsymbol{p_y^2}+\boldsymbol{p_z^2}-\boldsymbol{a_3^2}-\boldsymbol{d_4^2}}{2\mathbf{a_3}\boldsymbol{d_4}}\right) \\ \boldsymbol{\pi} - \boldsymbol{asin}\left(\frac{\boldsymbol{p_x^2}+\boldsymbol{p_y^2}+\boldsymbol{p_z^2}-\boldsymbol{a_3^2}-\boldsymbol{d_4^2}}{2\mathbf{a_3}\boldsymbol{d_4}}\right) \end{cases}$$

2-4

$$\boldsymbol{\theta_2} = \boldsymbol{atan2}\left(-\mathbf{a_3}\boldsymbol{c_3}\boldsymbol{p_z} + (\boldsymbol{d_4} + \mathbf{a_3}\boldsymbol{s_3})(\boldsymbol{c_1}\boldsymbol{p_x} + \boldsymbol{s_1}\boldsymbol{p_y}), (\boldsymbol{d_4} + \mathbf{a_3}\boldsymbol{s_3})\boldsymbol{p_z} + \mathbf{a_3}\boldsymbol{c_3}(\boldsymbol{c_1}\boldsymbol{p_x} + \boldsymbol{s_1}\boldsymbol{p_y})\right) - \boldsymbol{\theta_3}$$

2-5

$$\boldsymbol{\theta_5} = \begin{cases} \boldsymbol{acos}(\boldsymbol{c_1}\boldsymbol{s_{23}}\boldsymbol{r_{13}} + \boldsymbol{s_1}\boldsymbol{s_{23}}\boldsymbol{r_{23}} + \boldsymbol{c_{23}}\boldsymbol{r_{33}}) \\ -\boldsymbol{acos}(\boldsymbol{c_1}\boldsymbol{s_{23}}\boldsymbol{r_{13}} + \boldsymbol{s_1}\boldsymbol{s_{23}}\boldsymbol{r_{23}} + \boldsymbol{c_{23}}\boldsymbol{r_{33}}) \end{cases}$$

2-6

$$\boldsymbol{\theta_4} = \boldsymbol{atan2}(-\boldsymbol{s_1}\boldsymbol{r_{13}} + \boldsymbol{c_1}\boldsymbol{r_{23}}, \boldsymbol{c_1}\boldsymbol{c_{23}}\boldsymbol{r_{13}} + \boldsymbol{s_1}\boldsymbol{c_{23}}\boldsymbol{r_{23}} - \boldsymbol{s_{23}}\boldsymbol{r_{33}})$$

2-7

Αν η q_4 προκύψει στο διάστημα (90,180°) ή αν η λύση προκύψει στο διάστημα (-90,-180°) τότε η λύση είναι αντίστοιχα:

$$\boldsymbol{\theta_{42}} = \begin{cases} -\boldsymbol{2\pi} + \boldsymbol{\theta_4} \\ \boldsymbol{2\pi} + \boldsymbol{\theta_4} \end{cases}$$

2-8

$$\boldsymbol{\theta_6} = \boldsymbol{asin}\big((-\boldsymbol{s_1}\boldsymbol{c_4} - \boldsymbol{c_1}\boldsymbol{s_4}\boldsymbol{c_{23}})\boldsymbol{r_{11}} + (\boldsymbol{c_1}\boldsymbol{c_4} - \boldsymbol{s_1}\boldsymbol{s_4}\boldsymbol{c_{23}})\boldsymbol{r_{21}} + \boldsymbol{s_4}\boldsymbol{s_{23}}\boldsymbol{r_{31}}\big)$$

2-9

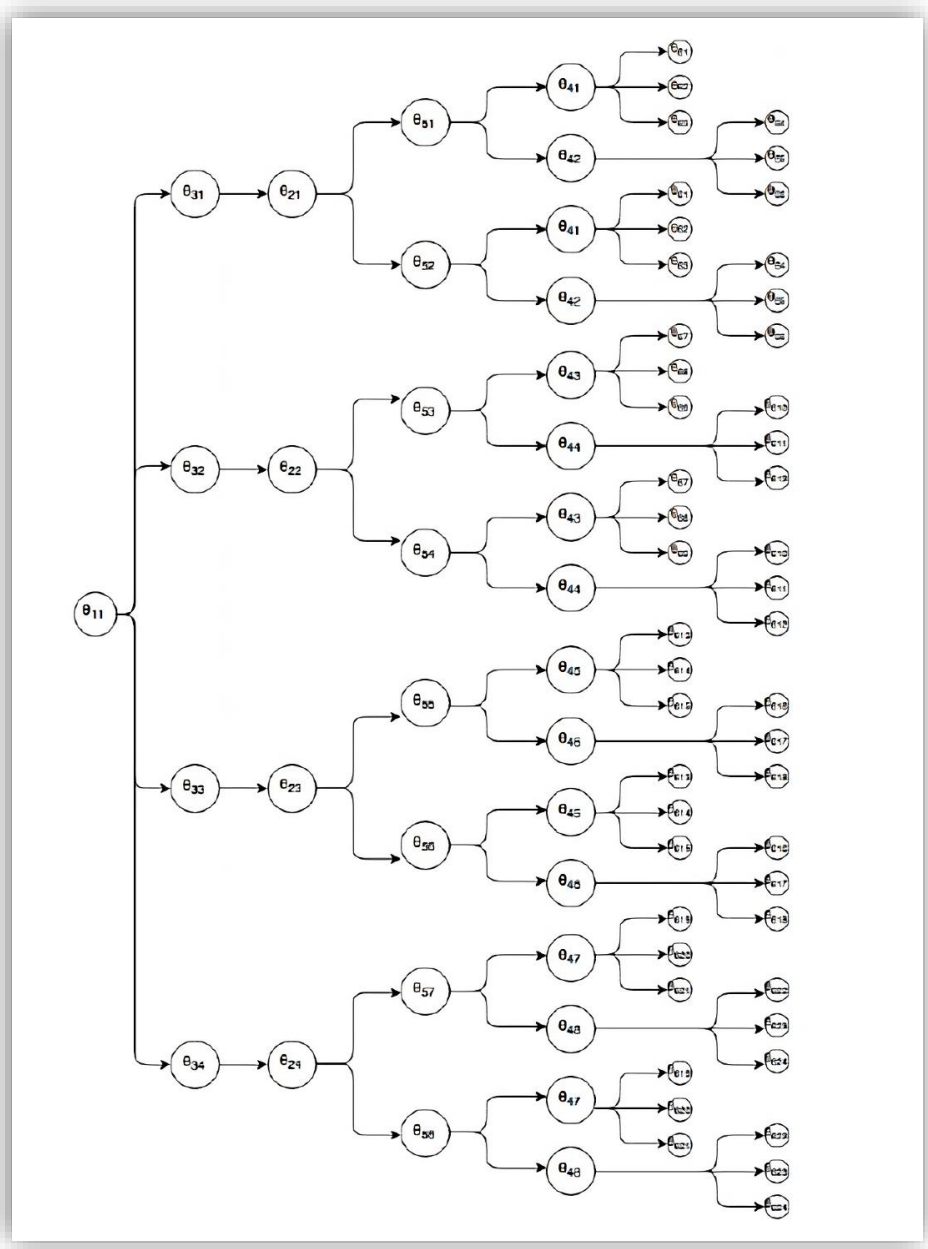
Αντίστοιχα, για να καλύψουμε τα όρια της 6ης άρθρωσης έχουμε τις επιπλέον λύσεις:

$$\boldsymbol{\theta_{62}} = \begin{cases} \boldsymbol{\pi - \theta_6} \\ \boldsymbol{-\pi - \theta_6} \end{cases}$$

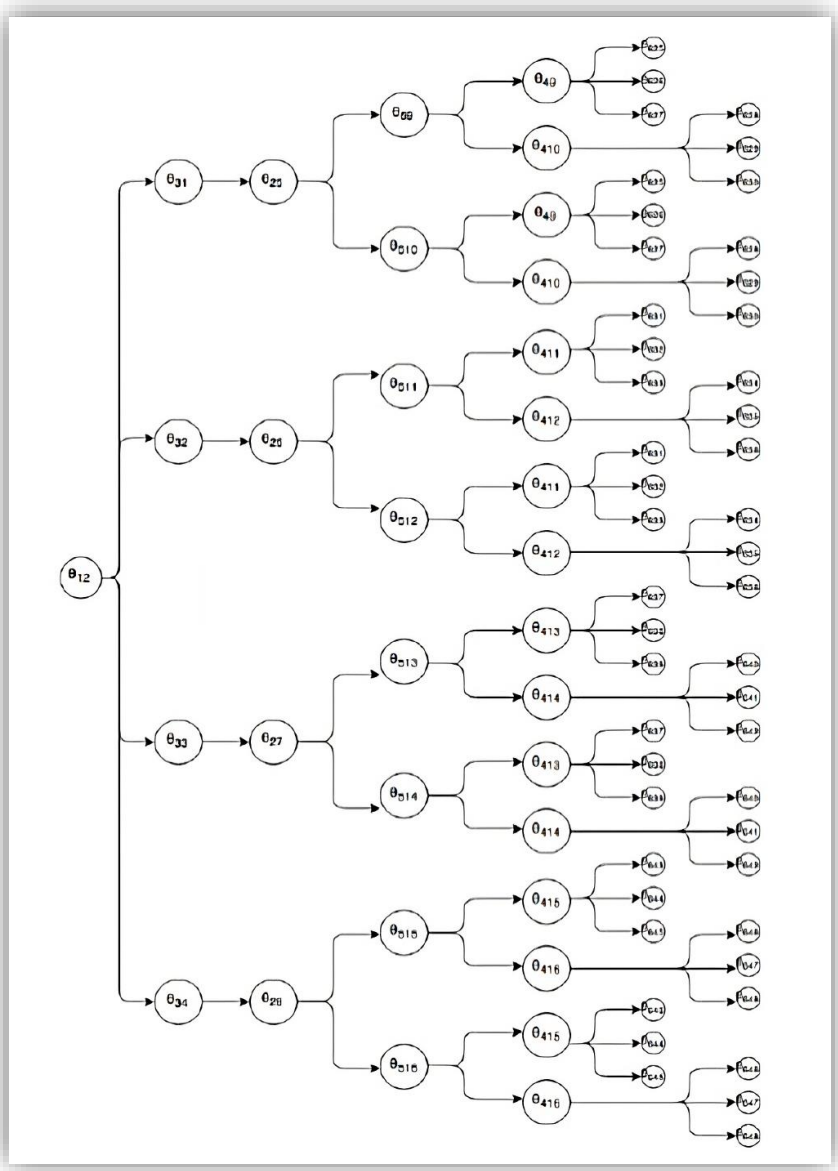
2-10

Όταν ισχύει $\theta_5=0$, τότε οι θ_4, θ_6 δεν υπολογίζονται χωριστά, αντίθετα θέτουμε $\theta_4= 0$ και η όλη περιστροφή ανατίθεται στην άρθρωση 6.

Η πολλαπλότητα των λύσεων είναι εμφανής, και σκοπός είναι η επιλογή της λύσης που συμφέρει περισσότερο. Το σύνολο των 96 πιθανών λύσεων φαίνεται στο Σχήμα 2-8 και στο Σχήμα 2-9 υπό μορφή δέντρων. Τα δέντρα είναι ιδιαίτερα περίπλοκα και απαιτούν προσεκτική εξέταση για την κατανόησή τους.



Σχήμα 2-8. Οι πρώτες 48 λύσεις Αντίστροφης Κινηματικής



Σχήμα 2-9. Οι υπόλοιπες 48 λύσεις Αντίστροφης Κινηματικής

Κατά την επίλυση της αντίστροφης κινηματικής, από τους 96 πιθανούς συνδυασμούς κάποιοι θα απορριφθούν γιατί θα υπερβαίνουν τα όρια των αρθρώσεων, αλλά τελικά πρέπει να επιλεχθεί μια μοναδική λύση. Αυτό γίνεται μέσω ελαχιστοποίησης της συνολικής μεταβολής της τιμής των αρθρώσεων από τις αρχικές συνθήκες, δηλαδή:

$$\boldsymbol{min} \sum_{1}^{6} \boldsymbol{\delta\theta_i}$$

2-11

Όπως θα εξηγηθεί στο Κεφάλαιο 3, ο λόγος που αναφέρθηκε η παραπάνω διαδικασία ήταν καθαρά για λόγους πληρότητας. Κατά την εκπόνηση της εργασίας, η αντίστροφη κινηματική επιλύεται από το MATLAB με χρήση του Robotic Toolbox, όμως θεωρούμε ότι τα αποτελέσματα του αλγόριθμου επίλυσης θα συμφωνούν με αυτά που αναλύθηκαν μόλις. Βέβαια, στην παρούσα εργασία λήφθηκαν υπόψιν και οι διαστάσεις του τελικού στοιχείου δράσης (85mm through fixed joint), οπότε η λύση πράγματι αναμένεται να διαφέρει ελαφρώς από τις λύσεις που μόλις παρουσιάστηκαν.

2.1.4 Χειριστήριο

Στην παρούσα υποενότητα παρουσιάζεται συνοπτικά το χειριστήριο (teach pendant) του ρομποτικού βραχίονα Staubli RX90L (Μαραγκός & Maragkos, 2018). Η μορφή του teach pendant φαίνεται στο Σχήμα 2-10.

Σχήμα 2-10. Χειριστήριο του Staubli RX90L

Ακολουθεί μια εποπτική περιγραφή των λειτουργιών του χειριστηρίου.

Function Keys

Κάτω ακριβώς από την οθόνη του χειριστηρίου φαίνονται τα Function Keys τα οποία επιτρέπουν στον χρήστη/προγραμματιστή να εκτελέσει συγκεκριμένες λειτουργίες. Τα Function Keys είναι τα εξής: EDIT, DISP, CLR ERR, CMD, PROG SET και το κάθε πλήκτρο έχει πάνω αριστερά ένα λαμπάκι LED για να δείχνει την τρέχουσα επιλεγμένη λειτουργία.

Το πλήκτρο EDIT επιτρέπει στο χρήστη να μεταβάλλει τις τιμές πραγματικών μεταβλητών ή μεταβλητών θέσης με χρήση των αριθμητικών πλήκτρων που θα αναφερθούν στη συνέχεια.

Το πάτημα του πλήκτρου DISP εμφανίζει στη οθόνη (με σειρά, από αριστερά προς τα δεξιά) τις τρέχουσες τιμές των γωνιών των αρθρώσεων σε μοίρες (joint angles), την καρτεσιανή θέση του τελικού εργαλείου (world location), την κατάσταση του συστήματος (status), την κατάσταση των ψηφιακών εισόδων/εξόδων (binary I/O) και το τελευταίο σφάλμα που προέκυψε κατά την εκτέλεση (last error).

Το πλήκτρο CLR ERR είναι χρήσιμο κάθε φορά που προκύπτει κάποιο σφάλμα. Κατά την παρατήρηση σφάλματος εμφανίζεται ένα προειδοποιητικό μήνυμα στην οθόνη του pendant και ανάβει το λαμπάκι του συγκεκριμένου πλήκτρου και ο χρήστης πρέπει να πατήσει το συγκεκριμένο πλήκτρο για να διαγραφτεί το προειδοποιητικό μήνυμα και να επιστρέψει το σύστημα στην προηγούμενη κατάσταση.

Με το πάτημα του πλήκτρου CMD εμφανίζεται στη οθόνη (με σειρά, από αριστερά προς τα δεξιά) η επιλογή αυτόματης εκκίνησης (AUTO START), η βαθμονόμηση (CALIB) και η επιλογή αποθήκευσης όλης της μνήμης στο δίσκο (STORE ALL).

Το κουμπί PROG SET εμφανίζει στην οθόνη (με σειρά, από αριστερά προς τα δεξιά) τα εξής: νέο πρόγραμμα (NEW), τον καθορισμό του αριθμού βημάτων (STEP), τον καθορισμό του αριθμού των κύκλων του προγράμματος, την προσαρμογή της ονομαστικής ταχύτητας (CYCLE) και την εκκίνηση του προγράμματος που βρίσκεται στη μνήμη (START). Σε κάθε περίπτωση, για να κάνει την επιλογή του, ο χρήστης χρησιμοποιεί τα User Keys.

Data Entry Keys

Χαμηλά στο κέντρο του χειριστηρίου φαίνονται τα Data Entry Keys μέσω των οποίων ο χρήστης μπορεί να απαντάει σε μηνύματα της οθόνης και να εισάγει ψηφία. Τα πλήκτρα αυτά είναι τα εξής: +/YES, -/NO, DELETE, τα ψηφία 0-9, και το σημείο στίξης (τελεία).

Mode Selection Keys

Κεντρικά στο χειριστήριο φαίνονται τα Mode Selection Keys μέσω των οποίων ο χρήστης μπορεί να αλλάξει mode λειτουργίας. Τα κουμπιά αυτά είναι τα εξής EMERGENCY STOP, RUN/HOLD, DIS PWR, COMP PWR και MAN/HALT.

Το πλήκτρο EMERGENCY STOP χρησιμοποιείται σε κατάσταση έκτακτης ανάγκης για να κόψει το ηλεκτρικό ρεύμα του βραχίονα και να ενεργοποιήσει τα φρένα των κινητήρων. Από τη στιγμή που πατιέται αυτό το πλήκτρο, ο χρήστης για να συνεχίσει πρέπει να γυρίσει το πλήκτρο δεξιόστροφα, μετά να πατήσει το πλήκτρο COMP PWR και να πατήσει το πλήκτρο ARM POWER ON που θα αναβοσβήνει στο μπροστινό πάνελ που φαίνεται στο Σχήμα 2-5.

Το πλήκτρο RUN/HOLD ουσιαστικά είναι πλήκτρο PAUSE/RESUME και όταν διακόπτει τη λειτουργία του συστήματος βγαίνει το μήνυμα HOLD στην οθόνη. Το πλήκτρο DIS PWR κόβει το ηλεκτρικό ρεύμα του Staubli RX90L.

Το πλήκτρο COMP PWR, όπως αναφέρθηκε, ενεργοποιεί το ηλεκτρικό ρεύμα και βάζει το σύστημα σε compute mode (COMP) δηλαδή σε προετοιμασία για ενεργοποίηση του βραχίονα. Όταν το πλήκτρο αυτό πατηθεί, το πλήκτρο ARM POWER ON του μπροστινού πάνελ αναβοσβήνει για 15 δευτερόλεπτα (αν δεν πατηθεί εγκαίρως, πρέπει να γίνουν ξανά όλα από την αρχή). Τέλος, το συγκεκριμένο πλήκτρο πρέπει να πατιέται πριν την εκτέλεση οποιουδήποτε προγράμματος.

Το πλήκτρο MAN/HALT σταματάει το ρομποτικό βραχίονα και εμφανίζει στην οθόνη το μήνυμα E-STOP. Με το πάτημα του συγκεκριμένου πλήκτρου ο βραχίονας περνάει σε manual mode και ο χρήστης μπορεί να μετακινήσει το βραχίονα χειροκίνητα.

Manual Mode Selection

Όταν το σύστημα βρίσκεται σε manual mode το λαμπάκι του MAN/HALT είναι αναμμένο όπως επίσης είναι αναμμένο ένα από τα λαμπάκια του manual mode που υποδεικνύουν στο χρήστη σε ποιο mode βρίσκεται ο βραχίονας. Τα πιθανά modes που μπορεί να επιλέξει ο χρήστης είναι τα εξής: WORLD, TOOL, JOINT, FREE. Όταν ο χρήστης βρίσκεται στο WORLD mode οι κινήσεις γίνονται σύμφωνα με το απόλυτο παγκόσμιο σύστημα συντεταγμένων, ενώ κατά το

TOOL mode οι κινήσεις γίνονται σύμφωνα με το τοπικό σύστημα συντεταγμένων του τελικού εργαλείου. Επίσης, όταν ο χρήστης βρίσκεται στο JOINT mode επιτρέπεται η ανεξάρτητη περιστροφή αρθρώσεων, ενώ κατά το FREE mode οι αρθρώσεις απεμπλέκονται από τους κινητήρες και ο χρήστης μπορεί χειροκίνητα να τοποθετήσει το τελικό εργαλείο κατά το δοκούν, στα πλαίσια της διδασκαλίας του ρομπότ.

Manual Control Keys

Τα πλήκτρα στο δεξί μέρος του χειριστηρίου είναι τα Manual Control Keys και χρησιμεύουν όταν το σύστημα βρίσκεται σε manual mode. Μέσω των Manual Control Keys ο χρήστης μπορεί να επιλέξει τον άξονα κίνησης ή περιστροφής καθώς και την περιστροφή συγκεκριμένων αρθρώσεων. Τα κουμπιά αυτά είναι τα εξής: X/1, Y/2, Z/3, RX/4, RY/5, RZ/6. Τα πλήκτρα X, Y, Z, RX, RY, RZ αφορούν αντίστοιχα κίνηση και περιστροφή κατά τους τρείς άξονες όταν βρισκόμαστε στο WORLD MODE, ενώ τα πλήκτρα 1, 2, 3, 4, 5, 6 επιτρέπουν ανεξάρτητη περιστροφή των αντίστοιχων αρθρώσεων όταν ο χρήστης βρίσκεται στο JOINT MODE.

Speed Bars

Οι μπάρες ταχύτητας επιτρέπουν τη ρύθμιση της ταχύτητας και την κατεύθυνση του ρομποτικού βραχίονα (στα θετικά ή τα αρνητικά). Αν για παράδειγμα, επιλεγεί το πλήκτρο X/1 στο teach pendant, πατώντας το + ή το – στην μπάρα ταχύτητας μετακινείται η φλάντζα του τελικού σημείου δράσης (end-effector) στον θετικό ή αρνητικό άξονα αντίστοιχα. Ο Χρήστης μπορεί να επιλέξει μεγάλη ή μικρή ταχύτητα όσο ο αντίχειράς του απομακρύνεται από τα σύμβολα +/- σε όποια από τις δύο κατευθύνσεις επιθυμεί. Επίσης, το κουμπί SLOW επιτρέπει στο χρήστη να φύγει από το βασικό εύρος ταχυτήτων και να μεταβεί σε κίνηση με πάρα πολύ αργές ταχύτητες.

2.1.5 Γλώσσα V+

Η γλώσσα V+ είναι ένα σύστημα ελέγχου βασισμένο σε υπολογιστή και μια γλώσσα προγραμματισμού ειδικά σχεδιασμένη για να χρησιμοποιείται στα βιομηχανικά ρομπότ της Adept Technology, σε συστήματα όρασης, καθώς και σε συστήματα ελέγχου κίνησης. Η V+ τρέχει σε προγράμματα τύπου «dumb terminals» όπως είναι το Tera Term. Προκειμένου να επιτευχθεί επικοινωνία μεταξύ του ελεγκτή (του βραχίονα) και του υπολογιστή, απαιτείται σύνδεση μέσω σειριακής θύρας RS232. Ο χρήστης έχει την δυνατότητα να επεξεργαστεί και να αποθηκεύει προγράμματα στην μνήμη του ελεγκτή.

Ως σύστημα πραγματικού χρόνου, ο συνεχής υπολογισμός της τροχιάς από τη V+, επιτρέπει σύνθετες κινήσεις να εκτελούνται άμεσα με αποδοτική χρήση της μνήμης του συστήματος, καθώς και με μείωση της πολυπλοκότητας του συστήματος συνολικά. Το σύστημα V+, δημιουργεί συνέχεια εντολές ελέγχου του ρομπότ και μπορεί ταυτόχρονα να αλληλεπιδρά με ένα χειριστή, επιτρέποντας έτσι την δημιουργία και τροποποίηση προγραμμάτων.

Η V+ παρέχει όλη τη λειτουργικότητα των μοντέρνων υψηλού επιπέδου γλωσσών προγραμματισμού, όπως:

- Η κλήση υπορουτίνων.
- Οι δομές ελέγχου.
- Περιβάλλον πολλαπλών εργασιών.
- Εκτέλεση προγραμμάτων αναδρομικά και με επανείσοδο.

Ο editor του controller μπορεί να εκτελέσει τα ακόλουθα:

- Τροποποίηση ενός αποθηκευμένου προγράμματος.
- Εισαγωγή εντολών σε ένα πρόγραμμα.
- Διαγραφή εντολών.

Γενικά, μπορούμε να επεξεργαστούμε τα προγράμματά μας με δύο τρόπους. Ο line editor καλείται με την εντολή **EDIT** ενώ ο full-page editor με την εντολή **SEE**.

- **SEE** prog.name: Με τον τρόπο αυτό, δημιουργείται ένα νέο πρόγραμμα την μνήμη του υπολογιστή με το όνομα που επιλέξαμε (prog.name).
- **EDIT** prog.name: Με αυτή την εντολή, γίνεται επεξεργασία του προγράμματος που είναι ήδη αποθηκευμένο στην μνήμη με το συγκεκριμένο όνομα (prog.name).

Υπάρχουν 3 λειτουργίες όταν χρησιμοποιείται ο editor:

- **COMMAND MODE**: Σε αυτή τη λειτουργία βρίσκεται ο χρήστης αμέσως μετά την εκτέλεση της εντολής SEE prog.name.
- **INSERT MODE**: Με το πάτημα του κουμπιού I στο πληκτρολόγιο ο χρήστης μπαίνει σε αυτήν την λειτουργία, ενώ για να βγει από αυτή ο χρήστης πρέπει να πατήσει το κουμπί ESC,
- **REPLACE MODE**: Με το πάτημα του κουμπιού R στο πληκτρολόγιο ο χρήστης μπαίνει σε αυτήν την λειτουργία, ενώ για να βγει από αυτή ο χρήστης πρέπει να πατήσει το κουμπί ESC.

Στην πρώτη λειτουργία ο χρήστης μπορεί απλά να δει μια λευκή σελίδα εάν το πρόγραμμα δεν υπάρχει στην μνήμη ή να δει το πρόγραμμα εάν αυτό υπάρχει ήδη στην μνήμη. Στη δεύτερη λειτουργία ο χρήστης μπορεί να εισαγάγει ότι επιθυμεί μέσα στο πρόγραμμα, ενώ στην τρίτη λειτουργία ο χρήστης μπορεί να αντικαταστήσει τον χαρακτήρα που βρίσκεται μετά από τον κέρσορα.

Οι βασικές εντολές που συνήθως χρησιμοποιεί ο χρήστης είναι οι εξής:

- **FDIRECTORY** folder.name: Εμφάνιση όλων των αρχείων που εμπεριέχονται στον συγκεκριμένο φάκελο.
- **FLIST** file.name: Εμφανίζει τα περιεχόμενα του φακέλου «file.name».
- **FCOPY** file.name: Αντιγράφει ένα παλιό αρχείο σε ένα νέο.
- **FRENAME** file.name: Μετονομάζει το αρχείο.
- **FDELETE** file.name: Διαγράφει το αρχείο.
- **DEF CD** folder.name: Αλλαγή του φακέλου στον οποίο βρίσκεται ο χρήστης.
- **ENABLE TRACE**: Εμφανίζονται τα ίχνη-βήματα του προγράμματος που εκτελείται.
- **STORE** file.name = prog.name: Αποθηκεύει το πρόγραμμα που υπάρχει ήδη στην μνήμη RAM μέσα στον σκληρό δίσκο με το όνομα που θα δοθεί (file.name).
- **LOAD** file.name: Φορτώνει στην μνήμη RAM το πρόγραμμα «file.name».
- **EXECUTE** prog.name: Εκτέλεση ενός αποθηκευμένου προγράμματος. Το όνομα που έχει οριστεί δεν πρέπει να ξεκινά ποτέ από αριθμό και δεν πρέπει να υπερβαίνει τους 15 χαρακτήρες.
- **SET**: Δήλωση σημείων συντεταγμένων στον χώρο.
    - **SET** (ή **POINT**) #a = #PPOINT(j1_value, j2_value, j3_value, j4_value, j5_value, j6_value) – Ορισμός σημείου ακριβείας, μέσω δήλωσης περιστροφής των αρθρώσεων. Ένα σημείο ακρίβειας (precision point, το σύμβολο «#» υπάρχει μπροστά από το όνομα της θέσης) είναι ένα σημείο του οποίου οι συντεταγμένες δεν είναι εκφρασμένες σε καρτεσιανές συντεταγμένες αλλά σε μοίρες με βάση την απόλυτη θέση της κάθε άρθρωσης.

- **SET** a = TRANS(X_value, Y_value, Z_value, y_value, p_value, r_value) – Δήλωση Καρτεσιανού σημείου, σε χιλιοστά με βάση τους ακλόνητους άξονες X, Y, Z. Τα στοιχεία y_value, p_value, r_value αποτελούν γωνίες Euler (ZYZ) και θα επεξηγηθούν στη συνέχεια.

- **DELAY** time: Σταματά όλες τις κινήσεις του ρομπότ για ορισμένο χρονικό διάστημα.
- **BREAK**: Διακόπτει τη ροή του προγράμματος μέχρι να περατωθεί η τρέχουσα κίνηση του βραχίονα.
- **SPEED** value: Θέτει το ποσοστό της ονομαστικής ταχύτητας των περιστροφικών αρθρώσεων. Αν προστεθεί η λέξη ALWAYS τότε η ταχύτητα παραμένει σταθερή καθ' όλη την διάρκεια εκτέλεσης του προγράμματος (πχ. SPEED 20 ALWAYS)
- **GOTO** label: Αναγκάζει τη ροή του προγράμματος να μεταπηδήσει προς τη γραμμή πού έχει χαρακτηριστεί με την ετικέτα (label), και η ροή συνεχίζει από εκεί κανονικά.
- **CALL** prog.name(arg_list): Καλείται η υπορουτίνα ή συνάρτηση prog.name(arg_list). Η ροή του προγράμματος διακόπτεται μέχρι το πέρας της prog.name και μετά συνεχίζει κανονικά.

Στην παρούσα διπλωματική εξετάστηκαν οι παρακάτω 7 εντολές που σχετίζονται με κίνηση ή και περιστροφή, τις οποίες η εφαρμογή μπορεί να αναγνωρίζει μέσα από τις φωνητικές εντολές του χρήστη:

1. **MOVE** loc.target: Κίνηση (joint-interpolated) του ρομποτικού βραχίονα σε μια συγκεκριμένη θέση (loc.target) ή διάταξη (point or precision point).
2. **APPRO** loc.target, height: Προσέγγιση σε συγκεκριμένο σημείο (loc.target) σε καθορισμένη απόσταση από αυτό (height) κατά τον τοπικό άξονα-z (tool frame).
3. **DEPART** height: Απομάκρυνση από παρούσα θέση, σε συγκεκριμένη απόσταση κατά τον τοπικό άξονα-z (tool frame).
4. **DRIVE** joint_number, degrees, speed: Περιστροφή συγκεκριμένης άρθρωσης κατά συγκεκριμένες μοίρες.
5. **OPENI**: Άνοιγμα της αρπάγης με ταυτόχρονη παύση του προγράμματος μέχρι το τέλος της εκτέλεσης του ανοίγματος.
6. **CLOSEI**: Κλείσιμο της αρπάγης με ταυτόχρονη παύση του προγράμματος μέχρι το τέλος της εκτέλεσης του κλεισίματος.
7. **DO READY**: Επιστροφή σε διάταξη αρχικοποίησης.

Όσον αφορά τον ορισμό ενός καρτεσιανού σημείου, π.χ. a = TRANS(X, Y, Z, y, p, r) χρειάζεται μια επεξήγηση όσων αφορά τα στοιχεία y, p, r που αφορούν τον προσανατολισμό. Σύμφωνα με τον οδηγό χρήσης της V$^+$ ο προσανατολισμός (y, p, r) ορίζεται ως γωνίες Euler μέσω διαδοχικών περιστροφών γύρω από τους τοπικούς άξονες Z-Y-Z. Οι γωνίες Euler ορίζονται εν γένει με πολλούς τρόπους, π.χ. περιστροφή ZYX, και η καλύτερη επιλογή καθορίζεται από το πεδίο εφαρμογής και τους στόχους προς επίτευξη. Ο οδηγός χρήσης αναθέτει στις γωνίες y, p, r τις ονομασίες yaw, pitch, roll, οι οποίες παρόλο που έχουν προέλευση στην αεροπορία χρησιμοποιούνται ευρύτατα και σε εφαρμογές ρομποτικής. Θέλοντας να είμαστε σύμφωνοι με την ονοματολογία του οδηγού χρήσης, από εδώ και στο εξής όποτε αναφερόμαστε στις γωνίες yaw, pitch, roll θα εννοούμε αντιστοίχως τις γωνίες Euler που περιγράφουν διαδοχικές περιστροφές γύρω από άξονες ZYZ.

Η γλώσσα V$^+$ έχει πολλές ιδιαιτερότητες και ενδιαφέρουσες πληροφορίες που μπορεί ο αναγνώστης να αναζητήσει. Για περισσότερες πληροφορίες ο αναγνώστης μπορεί να συμβουλευτεί τους οδηγούς (V$^+$ Reference Guide, 1997) και (V$^+$ User's Guide, 1997).

2.1.6 Πιστοποίηση Αντίστροφης Κινηματικής

Για τον έλεγχο και την πιστοποίηση της λειτουργικότητας του συστήματός μας, πραγματοποιήθηκαν πειραματικές δοκιμές στο εργαστήριο του τομέα Τεχνολογίας των Κατεργασιών της σχολής Μηχανολόγων Μηχανικών του Εθνικού Μετσόβιου Πολυτεχνείου. Δυστυχώς λόγω προβλήματος στη σειριακή επικοινωνία μεταξύ του ελεγκτή του βραχίονα και εξωτερικών συσκευών, δεν ήμασταν σε θέση να εκτελέσουμε αρχείο που παράγεται από την εφαρμογή μας. Αυτό που μπορούσαμε να κάνουμε ήταν, με χρήση του Teach Pendant, να συγκρίνουμε τις τιμές των γωνιών των αρθρώσεων σε συγκεκριμένες διατάξεις, και να τις συγκρίνουμε με αυτές που προκύπτουν από την επίλυση της αντίστροφης κινηματικής στο MATLAB για δεδομένες επιθυμητές συντεταγμένες.
Συγκεκριμένα, θεωρήσαμε ως επιθυμητές θέσεις (mm and degrees) τα στοιχεία που αναγράφει ο Πίνακας 2-4. Με βάση τα στοιχεία αυτά, η αντίστροφη κινηματική δίνει ως γωνίες αρθρώσεων τις τιμές που αναγράφει ο Πίνακας 2-5.

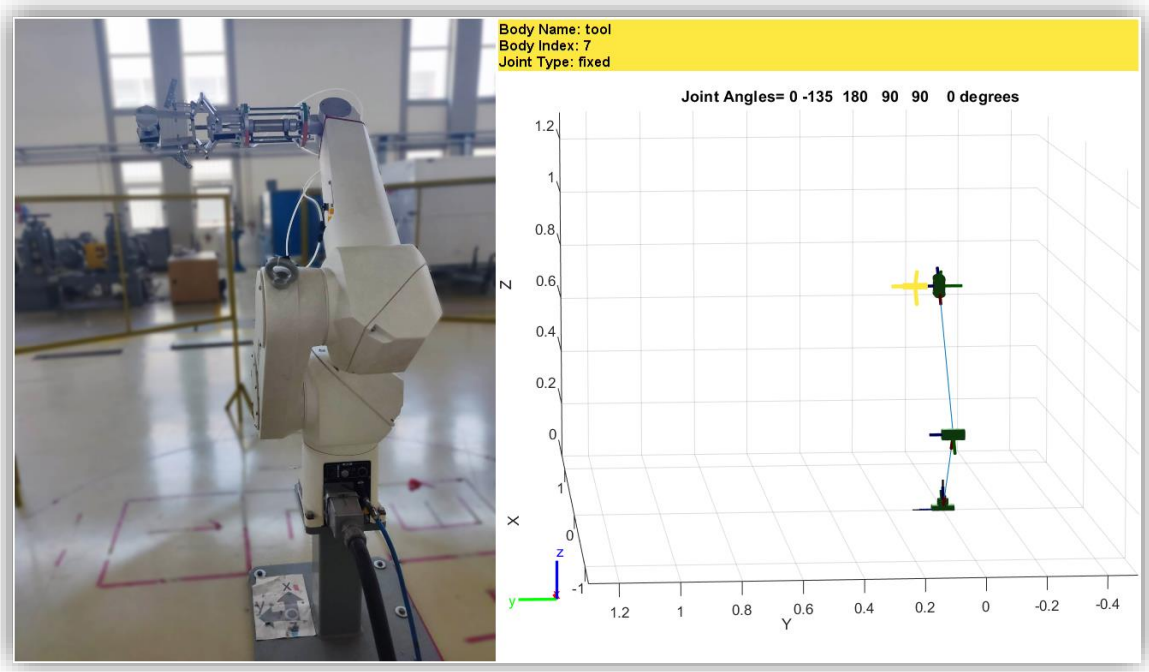
Πίνακας 2-4. Επιθυμητές Θέσεις για Σύγκριση

X	Y	Z	yaw	pitch	roll
141.42	85	777.82	90	90	45
425	425	304.71	30	45	60
226.42	0	777.82	0	90	0
594.836	594.836	793.83	45	0.04	0

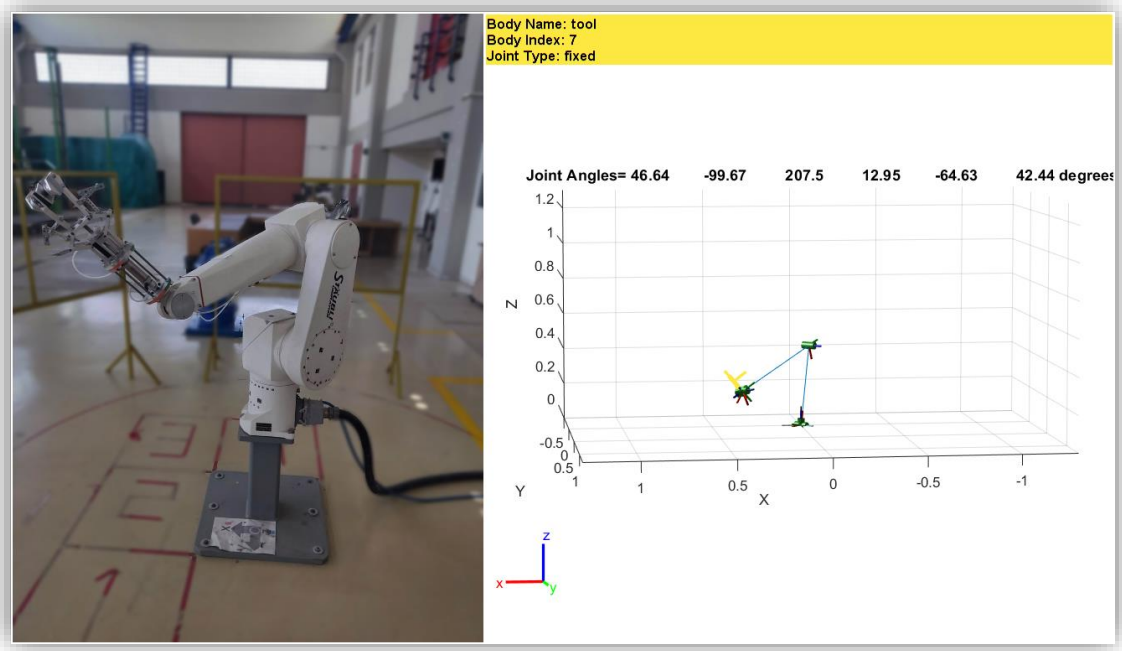
Πίνακας 2-5. Επιθυμητές Γωνίες για Σύγκριση

q_1	q_2	q_3	q_4	q_5	q_6
0	-135	180	90	90	0
46.64	-99.67	207.50	12.95	-64.63	42.44
0	-135	180	0	45	0
45.00	-40.12	90	0	-49.84	0

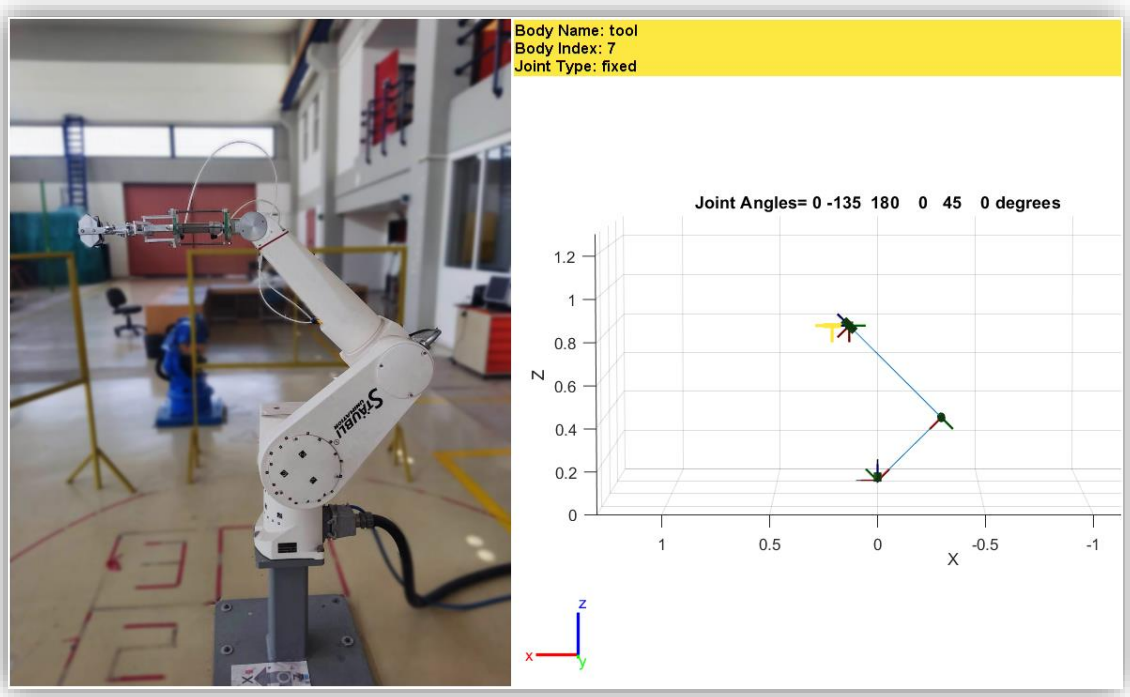
Στα παρακάτω σχήματα, δηλαδή από το Σχήμα 2-11 έως το Σχήμα 2-14, φαίνονται ανά δύο οι συγκρίσεις μεταξύ της πραγματικότητας και των αποτελεσμάτων που προκύπτουν με βάση την αντίστροφη κινηματική στο MATLAB. Για καλύτερη σύγκριση οι φωτογραφίες και τα διαγράμματα φαίνονται από παρόμοιες οπτικές γωνίες. Φαίνεται πως η αντίστροφη κινηματική δίνει ρεαλιστικά αποτελέσματα καθώς οι συγκρινόμενες διατάξεις φαίνονται αρκετά πανομοιότυπες και οι πραγματικές γωνίες των αρθρώσεων συμφωνούν με αυτές που υπολογίστηκαν στο MATLAB. Με βάση τα όσα αναφέρθηκαν, είναι βέβαιο πως το μοντέλο επίλυσης αντίστροφης κινηματικής δίνει ρεαλιστικά αποτελέσματα.



Σχήμα 2-11. Σύγκριση Διατάξεων 1



Σχήμα 2-12. Σύγκριση Διατάξεων 2



Σχήμα 2-13. Σύγκριση Διατάξεων 3

Σχήμα 2-14. Σύγκριση Διατάξεων 4

Τέλος, επαναλήφθηκε ακριβώς η ίδια διαδικασία, επιβεβαιώνοντας την ορθότητα της αντίστροφης κινηματικής για 4 ακόμα σημεία που μας ενδιαφέρουν στα πλαίσια της βιομηχανικής διεργασίας pick and place.

2.2 Διεργασία Παραλαβής και Τοποθέτησης Τεμαχίου

2.2.1 Περιγραφή

Η διαδικασία pick and place είναι μια θεμελιώδης εφαρμογή της βιομηχανικής ρομποτικής, στα πλαίσια αυτοματισμών και παραγωγής και ο βραχίονας Staubli RX90L είναι κατάλληλος για την εκτέλεσή της. Η διαδικασία περιλαμβάνει αυτοματοποιημένη διαχείριση και μετακίνηση αντικειμένων κατά τη γραμμή παραγωγής/κατασκευής, και χρησιμοποιείται από την αυτοκινητοβιομηχανία, την παραγωγή ηλεκτρονικών εξαρτημάτων, τη βιομηχανία τροφίμων κ.α. Κατά την εκτέλεση, ένας ρομποτικός βραχίονας αναλαμβάνει την παραλαβή αντικειμένων από μια γνωστή θέση και την τοποθέτησή τους σε μια άλλη γνωστή θέση, συνήθως επαναληπτικά και με μεγάλη ακρίβεια κινήσεων. Ακολουθεί μια εποπτική ανάλυση της υπό εξέταση διεργασίας.

Το πρώτο βήμα στη διαδικασία pick and place είναι η ταυτοποίηση (identification) και ο εντοπισμός (localization) αντικειμένων. Αυτό τυπικά επιτυγχάνεται μέσω αισθητήρων, συστήματος μηχανικής όρασης ή κάποια άλλη μέθοδο εντοπισμού. Τα συστήματα αυτά συλλέγουν πληροφορίες για τα αντικείμενα, όπως τη θέση, τον προσανατολισμό, το μέγεθος και το σχήμα. Προηγμένα συστήματα μηχανικής όρασης αξιοποιούν αλγορίθμους ανάλυσης εικόνων με σκοπό τον καθορισμό της θέσης και των χαρακτηριστικών των αντικειμένων προς μετακίνηση.

Μετά την ταυτοποίηση επιλέγεται κατάλληλη αρπάγη (gripper) ή τελικό στοιχείο δράσης (end effector) με βάση τα χαρακτηριστικά των αντικειμένων. Υπάρχουν πολλοί τύποι αρπάγης όπως μηχανικές αρπάγες με σαγόνια (συχνά αναφέρονται ως δάχτυλα), αρπάγες που λειτουργούν σε κενό (vacuum-based), πνευματικές αρπάγες, μαγνητικές αρπάγες, και εξειδικευμένες αρπάγες για ιδιαιτέρως ευαίσθητα υλικά. Η επιλογή αρπάγης καθορίζεται από χαρακτηριστικά του αντικειμένου όπως υλικό, σχήμα, διαστάσεις, βάρος και ευθραυστότητα. Βασική απαίτηση είναι η αρπάγη να μπορεί να συγκρατεί αντικείμενα με αξιοπιστία, χωρίς παράλληλα να προκαλεί φθορά στα τεμάχια.

Μετά την παραλαβή του τεμαχίου, ο βραχίονας κινείται σε προκαθορισμένη τροχιά προς συγκεκριμένη θέση και προσανατολισμό. Ο σχεδιασμός τροχιάς προγραμματίζεται εκ των προτέρων χειροκίνητα ή προγραμματίζεται αυτόματα με χρήση αλγορίθμων βελτιστοποίησης ή και αποφυγής εμποδίων. Η όλη κίνηση μέχρι την τοποθέτηση ελέγχεται προσεκτικά ώστε να εξασφαλίζεται μεγάλη ακρίβεια και ομαλές κινήσεις, αποφεύγοντας την πρόκληση ζημιών στα αντικείμενα και το γύρω περιβάλλον.

Κατά την άφιξη στη θέση τοποθέτησης του προϊόντος η αρπάγη ανοίγει προς απελευθέρωση του τεμαχίου. Η διαδικασία απαιτεί προσεκτική κίνηση ακριβείας, και έλεγχο ασκούμενης δύναμης για την εξασφάλιση της ασφαλούς τοποθέτησης στην επιθυμητή θέση. Σε αυτό το στάδιο, συστήματα όρασης ή αισθητήρες μπορούν να λειτουργούν πιλοτικά επαληθεύοντας τη σωστή τοποθέτηση και κάνοντας μικρές διορθώσεις εάν απαιτείται.

Η διαδικασία pick and place λειτουργεί επαναληπτικά και μπορεί να εκτελείται με συνεχή τρόπο από το ρομποτικό σύστημα. Ο χρόνος περιόδου (cycle time), δηλαδή ο χρόνος που απαιτείται για μία παραλαβή και μία τοποθέτηση εξαρτάται από αρκετές παραμέτρους όπως το βάρος αντικειμένου, την απόσταση μεταξύ των θέσεων παραλαβής και τοποθέτησης, την περιπλοκότητα της τροχιάς καθώς και την ταχύτητα των κινητήρων που κινούν τις αρθρώσεις του βραχίονα. Η βελτιστοποίηση της περιόδου είναι απαραίτητη για τη μεγιστοποίηση της παραγωγικότητας.

2.2.2 Πλεονεκτήματα – Μειονεκτήματα

Πλεονεκτήματα

- Αυξημένη Απόδοση (efficiency): Η βιομηχανική διαδικασία pick and place προσφέρει σαφώς μεγαλύτερη ταχύτητα και ακρίβεια κινήσεων από αυτή που θα προσέφερε το ανθρώπινο δυναμικό. Τέτοια ρομποτικά συστήματα είναι σε θέση να εκτελούν επαναληπτικές δράσεις με συνέπεια και ακρίβεια, και επιτυγχάνουν αυξημένο ρυθμό παραγωγής και μειωμένες χρονικές περιόδους.
- Αυξημένη Ακρίβεια (accuracy): Τα βιομηχανικά ρομπότ υπερτερούν στην ακρίβεια κινήσεων και τοποθέτησης αντικειμένων. Αυτό οδηγεί σε αυξημένη ποιότητα προϊόντος, μείωση σφαλμάτων και αυξημένη συνέπεια στην παραγωγή/συναρμολόγηση προϊόντων.
- Αυξημένη Ασφάλεια (safety): Με την αυτοματοποίηση της διαδικασίας pick and place, μειώνεται σημαντικά το ρίσκο τραυματισμού λόγω επαναλαμβανόμενης χειροκίνητης μετακίνησης υψηλού φορτίου. Αυτό δημιουργεί ένα ασφαλέστερο περιβάλλον για τους εργαζόμενους.
- Αυξημένη Ευελιξία και Προσαρμοστικότητα: Αυτά τα ρομποτικά συστήματα μπορούν εύκολα να επαναπρογραμματιστούν ώστε να χειρίζονται διαφορετικά αντικείμενα, μεγέθη και σχήματα. Αυτή η ευελιξία προσφέρει στους κατασκευαστές την ικανότητα γρήγορης προσαρμογής σε μεταβαλλόμενες γραμμές παραγωγής καθώς και την ικανότητα βελτιστοποίησης παραγωγής.
- Μείωση Λειτουργικού Κόστους: Παρόλο που η εγκατάσταση βιομηχανικών ρομπότ συνοδεύονται από αρκετά μεγάλο κόστος (αρχικό κόστος επένδυσης), αυτό μακροπρόθεσμα συνοδεύεται από σημαντική μείωση του λειτουργικού κόστους. Η αυξημένη παραγωγή, ο αυξημένος έλεγχος ποιότητας, τα μειωμένα λειτουργικά κόστη (π.χ. μισθός εργάτη που θα εκτελούσε χειροκίνητα τη διεργασία), και η μειωμένη ανάγκη διορθώσεων συμβάλλουν σημαντικά σε μια οικονομικά αποδοτική βιομηχανική διεργασία.
- Αύξηση Παραγωγικότητας (throughput): Η βιομηχανική διαδικασία pick and place λειτουργεί σε αυξημένο ρυθμό και μπορεί να χειρίζεται μεγάλες ποσότητες αντικειμένων. Αυτό αυξάνει σημαντικά την παραγωγικότητα ώστε να ανταπεξέρχεται στις αυξανόμενες βιομηχανικές απαιτήσεις.

Μειονεκτήματα

- Υψηλή Κόστος Επένδυσης: Η εγκατάσταση ρομποτικού συστήματος για pick and place συνοδεύεται από σημαντικό αρχικό κόστος επένδυσης, που περιλαμβάνει την αγορά του βραχίονα, των αρπαγών, συστημάτων μηχανικής όρασης και την ενσωμάτωση στην υπάρχουσα υποδομή. Αυτό το αρχικό κόστος ίσως λειτουργεί ως σημαντικό εμπόδιο σε μικρές ή οικονομικά ασθενείς επιχειρήσεις.
- Περίπλοκος Προγραμματισμός και Ολοκλήρωση: Ο προγραμματισμός και η ενσωμάτωση συστήματος pick and place μπορεί να γίνει ιδιαίτερα περίπλοκη και χρονοβόρα. Απαιτείται γνώση και εξειδίκευση στη ρομποτική, τους αυτοματισμούς και τον προγραμματισμό σε διάφορες γλώσσες, άρα μάλλον είναι απαραίτητη η πρόσληψη εξειδικευμένου προσωπικού.
- Έλλειψη Ευελιξίας σε Περίπλοκα/Αδιευκρίνιστα Σχήματα Αντικειμένων: Παρόλο που η βιομηχανική ρομποτική υπερτερεί στο χειρισμό καθορισμένων μεγεθών και σχημάτων, είναι πιθανό να δυσκολεύεται όταν τα προϊόντα είναι περίπλοκα ή έχουν ακαθόριστο

σχήμα. Άρα πιθανότατα χρειάζονται εξειδικευμένες αρπάγες και πρόσθετος εξοπλισμός για τον αποδοτικό χειρισμό τέτοιων αντικειμένων.

- Περιορισμένη Αισθητήρια Αντίληψη: Παρόλο που τα συστήματα μηχανικής όρασης χρησιμοποιούνται για εντοπισμό αντικειμένων, η αποδοτικότητά τους περιορίζεται από την περιπλοκότητα του περιβάλλοντος, και από ομοιότητες στην εμφάνιση διαφορετικών αντικειμένων. Αυτό μπορεί να οδηγήσει σε σφάλματα τοποθέτησης αν δε δοθεί η δέουσα προσοχή στο σχεδιασμό του συστήματος.
- Μειωμένη Συμβολή Ανθρώπινου Δυναμικού: Η αυτοματοποίηση της διεργασίας pick and place μειώνει σημαντικά την ανάγκη ανθρώπινης εργασίας. Αυτό, παρόλο που αυξάνει την απόδοση και μειώνει τις εργατικές δαπάνες, οδηγεί αναπόφευκτα σε ανεργία και μειωμένες ευκαιρίες εργασίας για πολλούς εργάτες.

Συνολικά, τα πλεονεκτήματα της ρομποτικής διεργασίας pick and place σίγουρα υπερνικούν τα μειονεκτήματα. Παρόλα αυτά, απαιτείται προσεκτικός σχεδιασμός ανάλογα την εφαρμογή, ανάλυση οφέλους-κόστους, και ορθή εγκατάσταση του συστήματος για επιτυχή εφαρμογή.

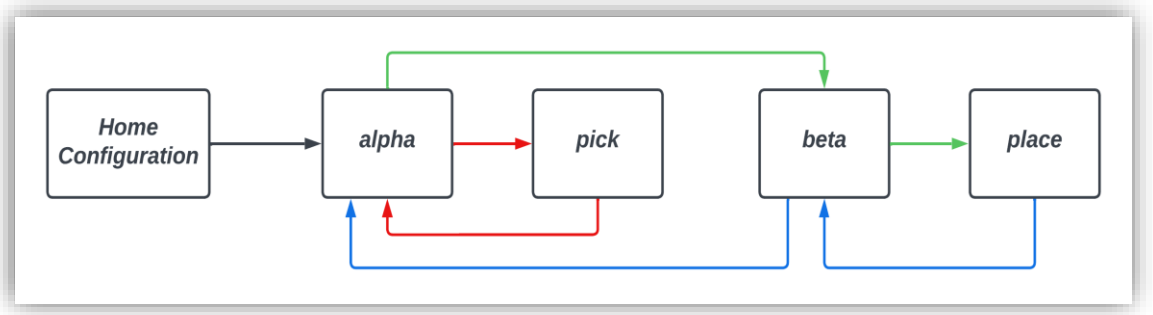
2.2.3 Προσομοίωση Pick and Place

Όπως αναφέρθηκε, λόγω τεχνικών προβλημάτων δεν είμασταν σε θέση να αξιοποιήσουμε πλήρως το βραχίονα για να διαπιστώσουμε τη λειτουργικότητα του συστήματός μας στην πράξη. Για αυτό, με σκοπό την οπτικοποίηση της όλης διαδικασίας και την υποστήριξη των ισχυρισμών μας, πραγματοποιήθηκε προσομοίωση της κίνησης στο MATLAB, με σχεδιασμό τροχιάς. Κατά το σχεδιασμό τροχιάς, από το εκάστοτε σημείο Α στο σημείο Β, θεωρήσαμε τις επιθυμητές τροχιές ως πολυώνυμα $5^{ου}$ βαθμού ώστε να εξασφαλίζεται ομαλότητα κινήσεων. Οι θέσεις από τις οποίες επιθυμούμε να περάσει ο βραχίονας είναι οι τιμές που αναγράφει ο Πίνακας 2-6.

Πίνακας 2-6. Επιθυμητές Θέσεις για Pick and Place

	X	Y	Z	y	p	r
pick	1053.2	0	318.2	0	90	0
place	0	1053.2	318.2	90	90	0
alpha	872.92	0	714.71	0	90	0
beta	0	872.92	714.71	90	90	0

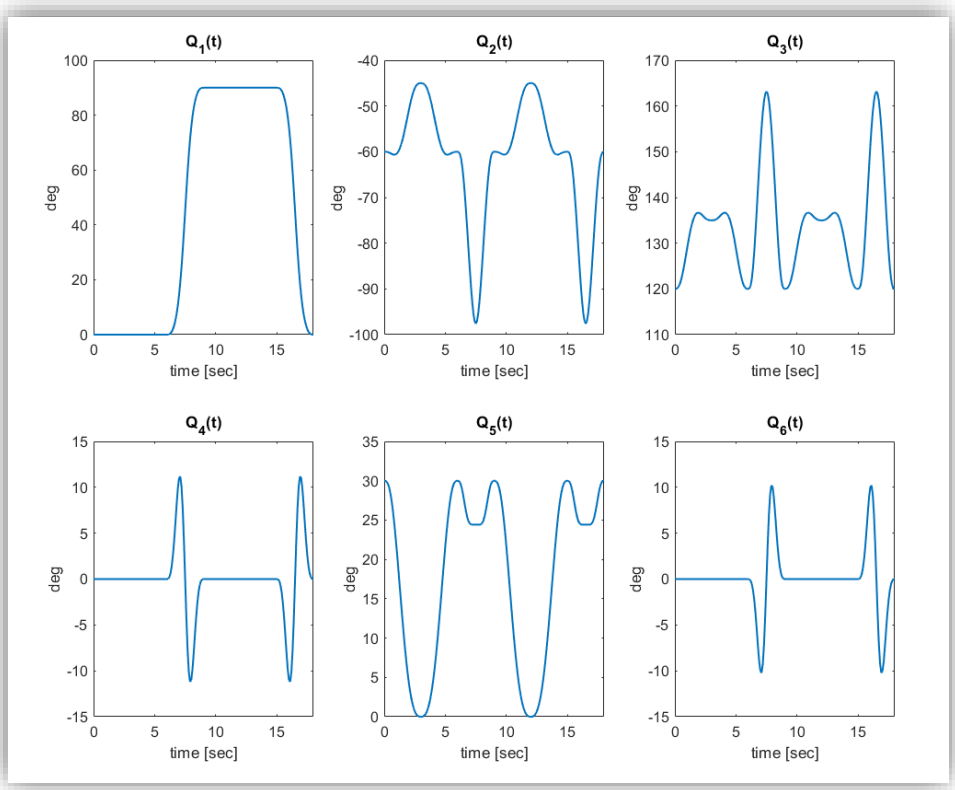
Η αλληλουχία των κινήσεων που εκτελούνται κατά τη συγκεκριμένη διεργασία φαίνονται στο Σχήμα 2-15. Φαίνεται δηλαδή ότι μετά την αρχική διάταξη, ο βραχίονας κάνει μια λούπα ανάμεσα στις θέσεις alpha και pick. Στη συνέχεια, κινείται από τη θέση alpha στη θέση beta και πραγματοποιείται μια λούπα μεταξύ των θέσεων beta και place, πριν εν τέλει επιστρέψει στη θέση alpha. Η χρονική αλληλουχία φαίνεται και από τα χρώματα των βελών, όπου ακολουθείται η γνωστή αλληλουχία RGB (red-green-blue). Στη δική μας προσομοίωση θεωρήσαμε ως αρχική θέση τη θέση alpha.



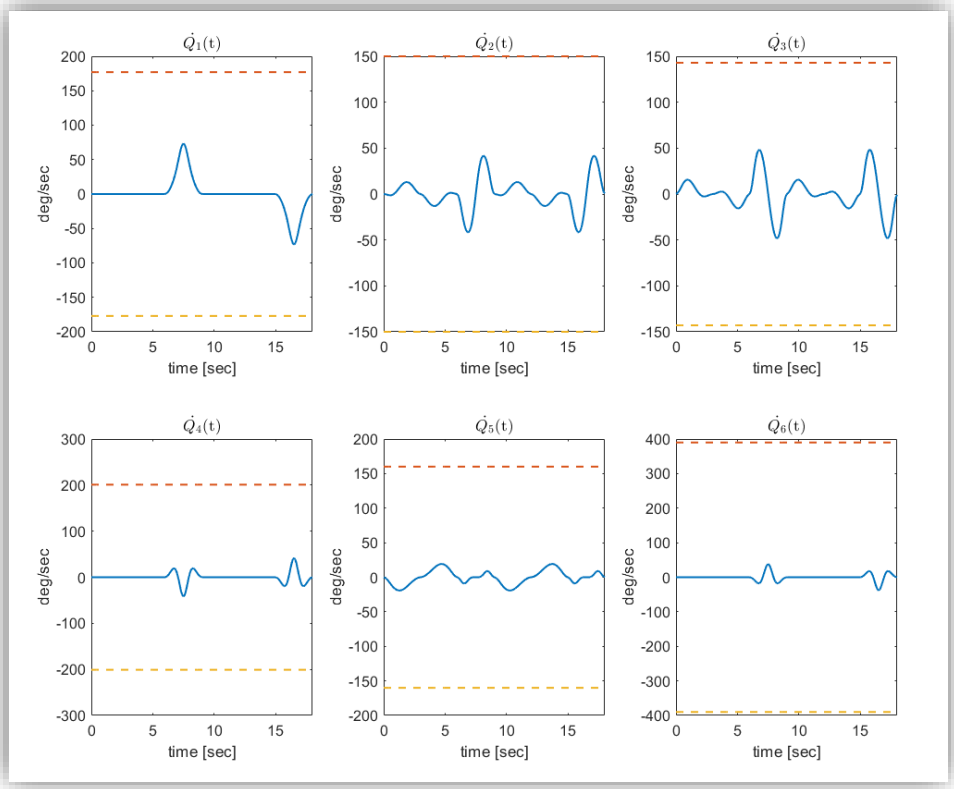
Σχήμα 2-15. Περιγραφή Κινήσεων Pick and Place

Σημειώνουμε ότι σε πραγματική εφαρμογή δε θα μας απασχολούσε καθόλου ο σχεδιασμός τροχιάς, καθώς ο ελεγκτής του βραχίονα θα φρόντιζε το σχεδιασμό και τη βελτιστοποίηση της κίνησης. Τονίζουμε ακόμα πως η παρούσα εργασία επικεντρώθηκε μόνο στην κινηματική του βραχίονα και όχι στη δυναμική. Δηλαδή ενδιαφερόμαστε μόνο για μετατοπίσεις, περιστροφές και ταχύτητες, και δε λαμβάνουμε υπόψιν μάζα, ασκούμενες δυνάμεις και επιταχύνσεις. Για τη συγκεκριμένη προσομοίωση θεωρήσαμε ως αρχική θέση τη θέση alpha, και από εκεί εκτυλίσσεται κίνηση όμοια με αυτή που φαίνεται στο Σχήμα 2-15. Επίσης, θεωρήσαμε χρόνο περιόδου 18 δευτερολέπτων, με χρονικό βήμα dt=0.1 sec, και από τη στιγμή που προσομοιώνουμε 6 διαδοχικές κινήσεις του βραχίονα, θεωρήσαμε ότι η κάθε κίνηση διαρκεί το ένα έκτο (1/6) της περιόδου (κάτι που δεν υπάρχει λόγος να ισχύει στην πραγματικότητα).

Στο Σχήμα 2-16 φαίνεται η χρονική εξέλιξη των γωνιών των αρθρώσεων κατά την προσομοίωση, και στο Σχήμα 2-17 φαίνεται η χρονική παράγωγος των γωνιών. Στα σχήματα αυτά φαίνεται πως οι γωνίες των αρθρώσεων μεταβάλλονται ομαλά, ενώ τα προφίλ χρονικής παραγώγου των αρθρώσεων δεν παρουσιάζουν ασυνέχειες και δεν ξεπερνούν τα επιβαλλόμενα όρια που αναγράφει ο Πίνακας 2-2, συνεπώς ο χρόνος περιόδου (cycle time) που θέσαμε είναι ρεαλιστικός αν μη τοι άλλο.

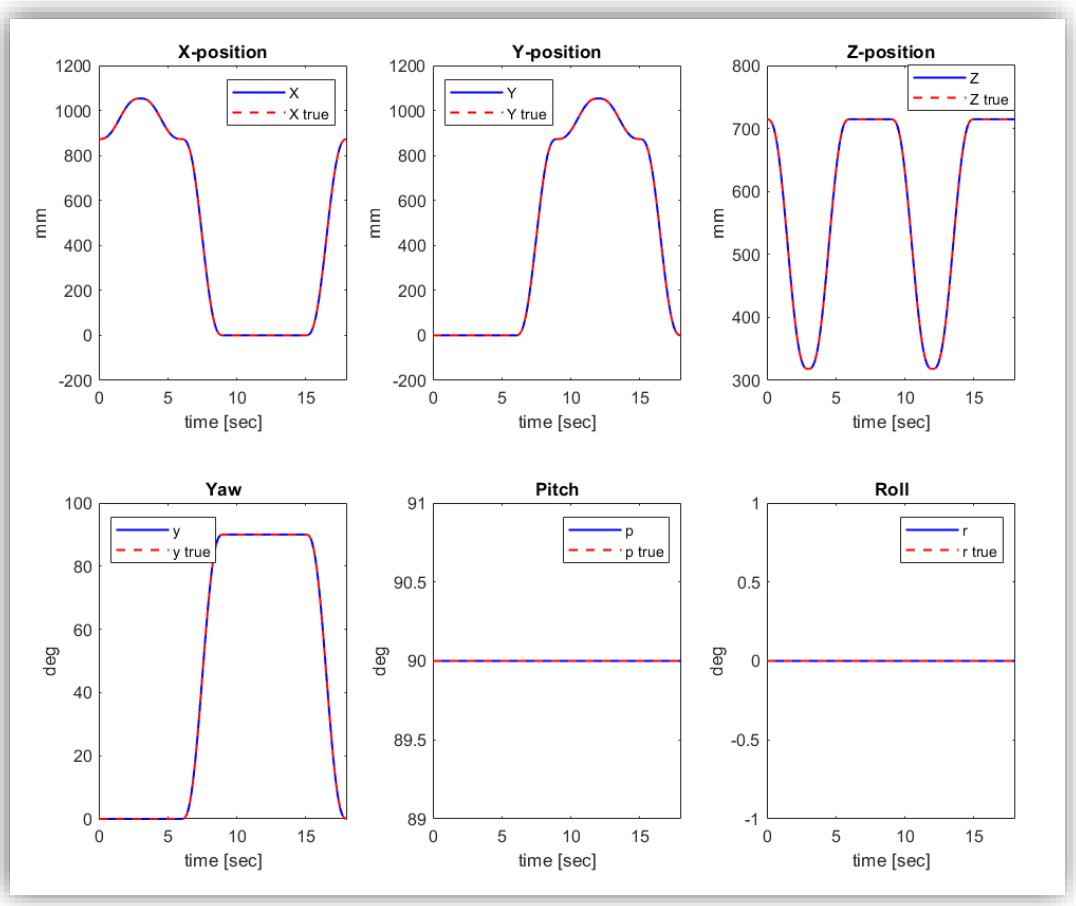


Σχήμα 2-16. Γωνίες Αρθρώσεων Q(t)



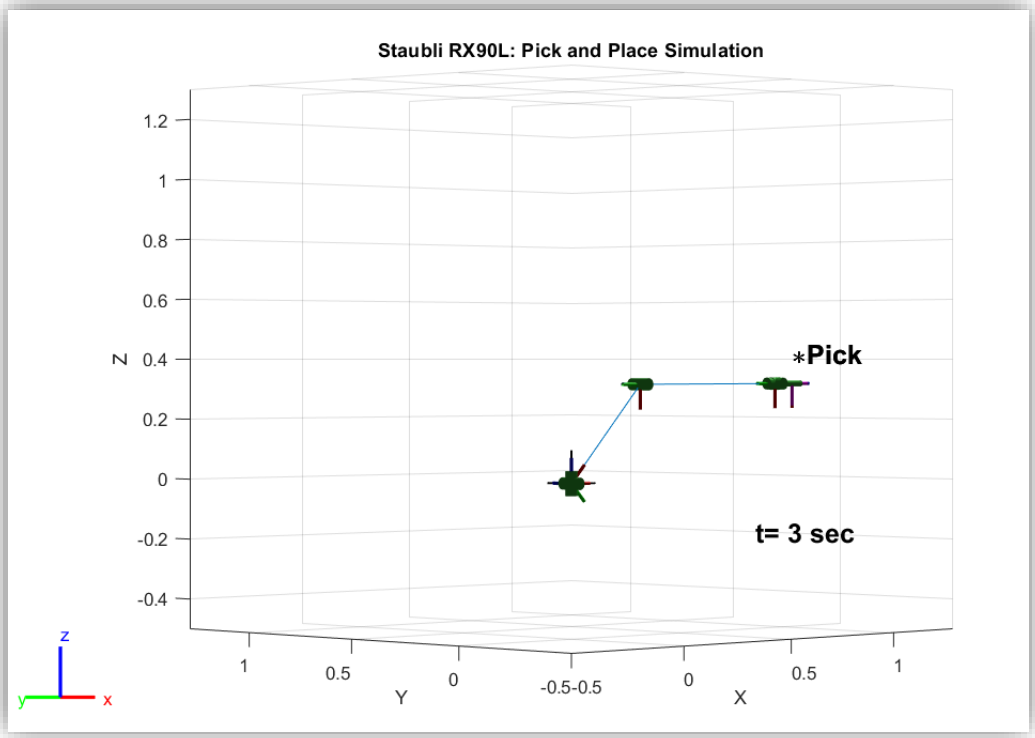
Σχήμα 2-17. Γωνιακές Ταχύτητες Αρθρώσεων

Στο Σχήμα 2-18 φαίνονται τα προφίλ της θέσης και του προσανατολισμού του βραχίονα σε αντιδιαστολή με την επιθυμητή τροχιά που επιβλήθηκε. Είναι ξεκάθαρο ότι η επιθυμητή ομαλή τροχιά ακολουθήθηκε με μεγάλη ακρίβεια, και τα προφίλ των μεγεθών που μας ενδιαφέρουν είναι λείες καμπύλες. Δηλαδή, λόγω της πολυωνυμικής τροχιάς που επιβάλλαμε επιτυγχάνεται συνέχεια ως προς τις ταχύτητες και ανώτερες χρονικές παραγώγους των μεγεθών. Θεωρούμε λοιπόν ότι όλα θα λειτουργούσαν ορθά κατά την πραγματική διεξαγωγή δοκιμής αυτής της διαδικασίας.

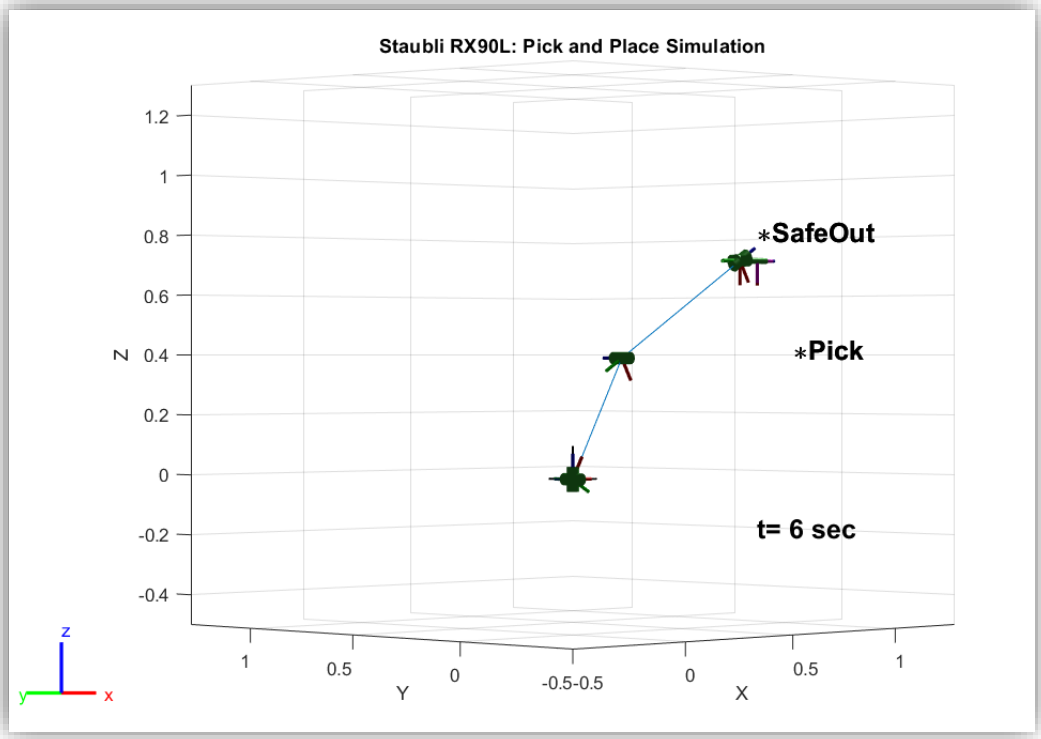


Σχήμα 2-18. Τροχιά (Pick and Place)

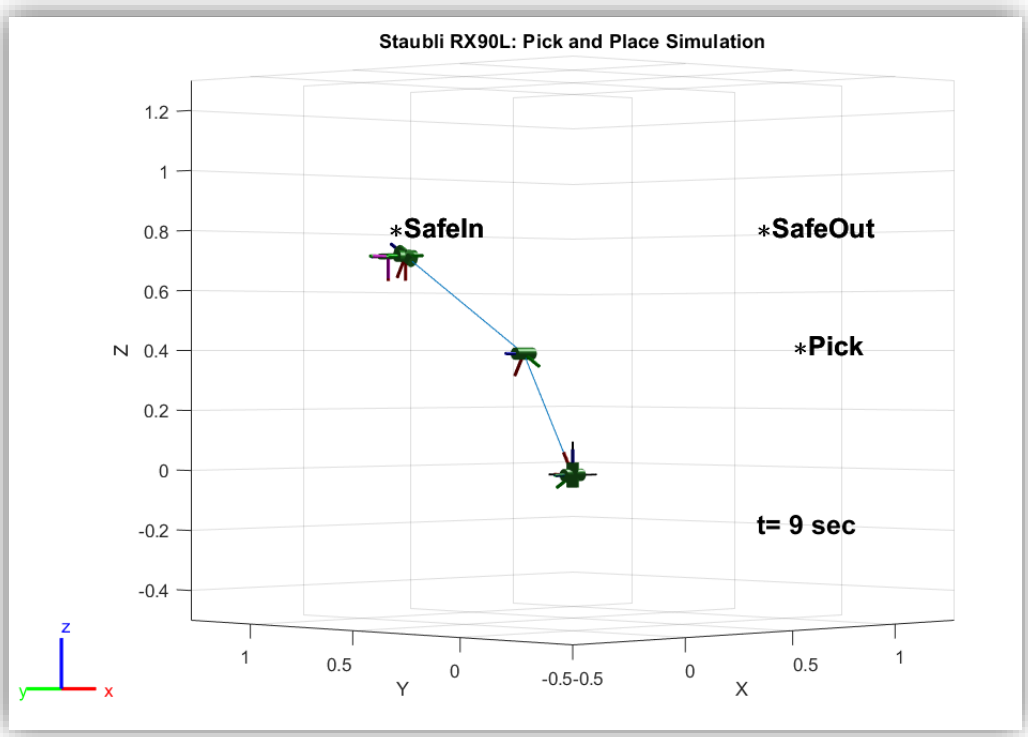
Οι διατάξεις που αναπαριστούν τις 4 ενδιάμεσες θέσης από τις οποίες διέρχεται ο βραχίονας απεικονίζονται διαδοχικά από το Σχήμα 2-19 έως το Σχήμα 2-22, όπου αναγράφονται και οι χρονικές στιγμές κατά τις οποίες ο βραχίονας διέρχεται από τις επιβαλλόμενες θέσεις. Κοιτώντας την προσομοίωση, μπορεί κανείς να φανταστεί πως με τον ορισμό τεσσάρων θέσεων και αξιοποιώντας της εντολές που προσφέρει η γλώσσα V^+, θα μπορούσε σχετικά εύκολα να προγραμματιστεί μια τέτοια διεργασία μέσω φωνητικών εντολών.



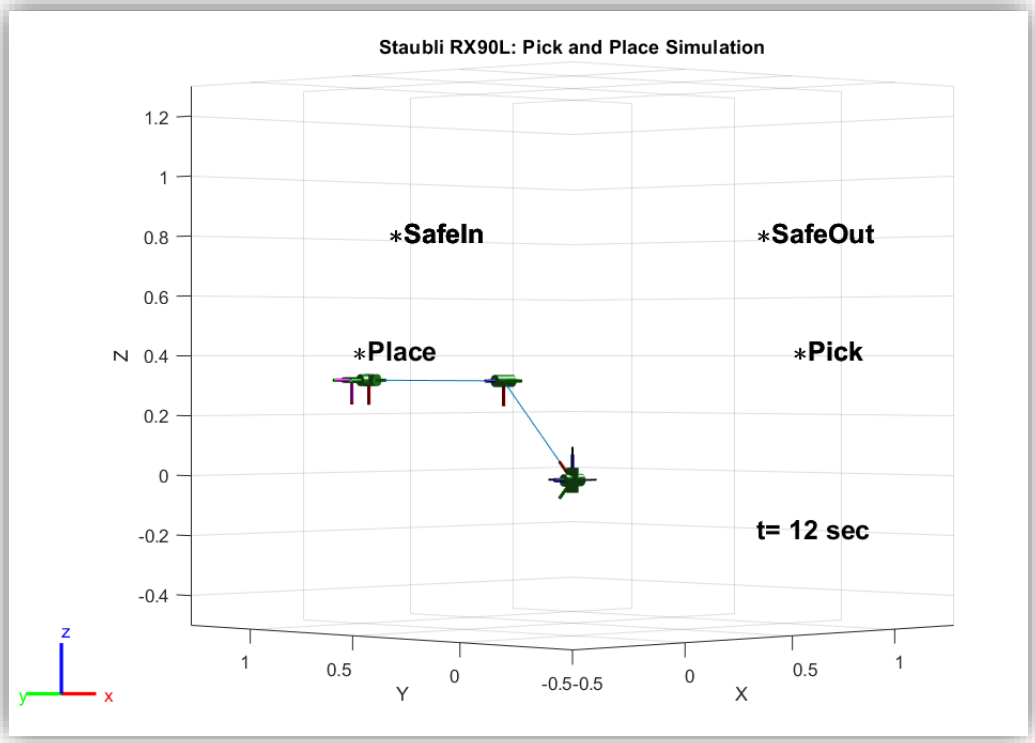
Σχήμα 2-19. Θέση Παραλαβής (pick)



Σχήμα 2-20.Θέση πριν/μετά την Παραλαβή (alpha)



Σχήμα 2-21. Θέση πριν/μετά την Τοποθέτηση (beta)



Σχήμα 2-22. Θέση Τοποθέτησης (place)

2.3 Αναγνώριση Ομιλίας

2.3.1 Εισαγωγή

Η αναγνώριση ομιλίας είναι ένα διεπιστημονικό πεδίο που περιλαμβάνει πληροφορική και υπολογιστική γλωσσολογία. Αναπτύσσει μεθόδους και τεχνολογίες για την αναγνώριση ή και μετάφραση ομιλούμενων λέξεων και τη μετατροπή τους σε γραπτό κείμενο, με κύριο προνόμιο τη δυνατότητα αναζήτησης (searchability). Η τεχνολογία είναι επίσης γνωστή ως automatic speech recognition (ASR), computer speech recognition και speech-to-text. Η αντίστροφη διαδικασία ονομάζεται σύνθεση ομιλίας (speech synthesis).

Οι εφαρμογές της συγκεκριμένης τεχνολογίας είναι αναρίθμητες. Συνήθως περιλαμβάνουν κάποια φωνητική διεπαφή για το χρήστη η οποία μπορεί να επιτρέπει την πραγματοποίηση τηλεφωνικής κλήσης, κάποιον οικιακό αυτοματισμό, την αναζήτηση λέξεων-κλειδιά, την εισαγωγή δεδομένων ή την προετοιμασία δομημένων εγγράφων. Υπάρχει επίσης η αναγνώριση ομιλητή (speaker identification), κάτι που μπορεί να χρησιμοποιηθεί στα πλαίσια της ασφάλειας ως μέσω αυθεντικοποίησης ή επαλήθευσης των στοιχείων του χρήστη.

Από τη σκοπιά της τεχνολογίας, η αναγνώριση ομιλίας έχει μακρά ιστορία που συνοδεύεται από μεγάλα άλματα προόδου, που βασίζονται τόσο στη δημοσίευση ακαδημαϊκών εργασιών όσο και στην παγκόσμια βιομηχανία που ερευνά, χρηματοδοτεί και χρησιμοποιεί σχετικές μεθόδους. Οι πιο πρόσφατες εξελίξεις στο πεδίο σχετίζονται με την ανάπτυξη νευρωνικών δικτύων βαθιάς μάθησης και των μεγάλων βάσεων δεδομένων (big data) (Zhang et al., 2021).

2.3.2 Ιστορική Αναδρομή

Κοιτώντας στο παρελθόν, η ανάπτυξη της αναγνώρισης φωνής μοιάζει με την ανάπτυξη ενός βρέφους, από αναγνώριση λίγων και συγκεκριμένων λέξεων στην ανάπτυξη ενός πλούσιου λεξιλογίου χιλιάδων λέξεων, την απάντηση ερωτήσεων και την ανάλυση συναισθημάτων (Pinola, 2011).

Το 1952 σχεδιάστηκε στα Bell Laboratories το σύστημα Audrey που αναγνώριζε αριθμούς (digits) από ανθρώπινη ομιλία. Μια δεκαετία αργότερα, η International Business Machines Corporation (IBM) παρουσίασε στη διεθνή έκθεση του 1962 το Shoebox, μια μηχανή με ικανότητα αναγνώρισης 16 λέξεων. Παράλληλα, εργαστήρια στην Αμερική, την Ιαπωνία, την Αγγλία και τη Σοβιετική Ένωση δημιούργησαν άλλες διατάξεις προς αναγνώριση φωνής, επεκτείνοντας τη σχετική τεχνολογία στο να μπορεί να αναγνωρίζει φωνήεντα και σύμφωνα. Σήμερα ίσως δεν ακούγονται πολύ σημαντικά βήματα, αλλά στην πραγματικότητα οι προσπάθειες αυτές ήταν ένα τρομερό ξεκίνημα, λαμβάνοντας υπόψιν και την πρωτόγονη κατάσταση των υπολογιστών εκείνης της εποχής.

Τη δεκαετία του 1970 η αναγνώριση φωνής ξεκίνησε την απογείωσή της κυρίως λόγω του ενδιαφέροντος και της επένδυσης του Υπουργείου Άμυνας των Ηνωμένων Πολιτειών. Στα πλαίσια του προγράμματος DARPA Speech Understanding Research (SUR), από το 1971 έως το 1976 σχεδιάστηκε το Harpy speech-understanding system του πανεπιστημίου Carnegie Mellon. Το Harpy μπορούσε να καταλάβει 1011 λέξεις, δηλαδή περίπου το εύρος λεξιλογίου ενός τρίχρονου παιδιού. Ήταν σημαντικό γιατί εισήγαγε μια πιο αποδοτική μέθοδο αναζήτησης, την ακτινωτή αναζήτηση (beam search), που απέδειξε το πεπερασμένο του δικτύου πιθανών προτάσεων. Η ιστορία της αναγνώρισης φωνής είναι στενά συνδεδεμένη με καινοτομίες στις μεθοδολογίες και τεχνολογίες αναζήτησης. Γενικά, τη δεκαετία του ’70 έγιναν μεγάλα βήματα στην ανάπτυξη του πεδίου, όπως η δημιουργία της πρώτης εμπορικής

εταιρίας αναγνώρισης φωνής, της Threshold Technology, και ο σχεδιασμός συστήματος αναγνώρισης πολλαπλών φωνών από τα Bell Laboratories.

Τη δεκαετία του 1980 το λεξιλόγιο αναγνώρισης εκτοξεύτηκε χάρη σε νέες τεχνικές, από μερικές εκατοντάδες λέξεις στην ικανότητα αναγνώρισης χιλιάδων λέξεων. Μια πολύ σημαντική αιτία αυτού του άλματος ήταν η στατιστική μέθοδος που ονομάζεται κρυφά Μαρκοβιανά μοντέλα (Hidden Markov Models: HMM) η οποία, αντί της απλής χρήσης προτύπων λέξεων και αναζήτηση μοτίβων στον ήχο, εκτιμά την πιθανότητα άγνωστοι ήχοι να είναι στην πραγματικότητα λέξεις. Έχοντας πια ένα πλούσιο λεξιλόγιο, η αναγνώριση φωνής άρχισε να δραστηριοποιείται σε εμπορικές εφαρμογές για επιχειρήσεις και βιομηχανίες. Κατάφερε επίσης να μπει στα σπίτια των ανθρώπων με τη μορφή της κούκλας Worlds of Wonder's Julie doll (1987) την οποία τα παιδιά μπορούσαν να εκπαιδεύσουν να αναγνωρίζει τη δική τους φωνή. Δυστυχώς όμως, τα συστήματα αναγνώρισης φωνής εκείνης της εποχής έπρεπε να υπερνικήσουν ένα μεγάλο εμπόδιο. Δεχόντουσαν διακριτή υπαγόρευση, δηλαδή ο χρήστης-εκπαιδευτής έπρεπε να διακόπτει την υπαγόρευση μετά από κάθε λέξη!

Τη δεκαετία του 1990 οι υπολογιστές είχαν πολύ πιο γρήγορους επεξεργαστές και τα συστήματα αναγνώρισης φωνής έγιναν βιώσιμα και για το μέσο άνθρωπο. Το 1990, η Dragon προώθησε το πρώτο προϊόν αναγνώρισης φωνής ευρείας κατανάλωσης, το Dragon Dictate στην τιμή των 9000 δολαρίων, και εφτά χρόνια αργότερα κατέφτασε το βελτιωμένο Dragon Naturally Speaking. Η εφαρμογή αναγνώριζε συνεχή ομιλία, οπότε ο χρήστης μπορούσε να ομιλεί άνετα και φυσικά, με ρυθμό έως 100 λέξεις το λεπτό. Όμως, το πρόγραμμα έπρεπε να εκπαιδευτεί για 45 λεπτά, και ήταν ακόμα αρκετά ακριβό, στην τιμή των 695 δολαρίων. Το 1996 ήταν η άφιξη του πρώτου Voice Portal εν ονόματι VAL, της BellSouth, και ήταν ένα διαδραστικό σύστημα αναγνώρισης φωνής που παρείχε πληροφορίες σχετικά με τα λεγόμενα του χρήστη στο τηλέφωνο.

Μέχρι το 2001, η αναγνώριση φωνής μέσω υπολογιστή είχε ξεπεράσει το 80% σε ακρίβεια (accuracy) αλλά μέχρι το τέλος της δεκαετίας, η εξέλιξη της συγκεκριμένης τεχνολογίας φαινόταν στάσιμη. Τα αποτελέσματα ήταν αποδεκτά για περιορισμένα λεξικά αναφοράς, αλλά τα συστήματα ακόμα «μαντεύαν» με τη βοήθεια στατιστικών μοντέλων ανάμεσα σε ακουστικά παρεμφερείς λέξεις, και όσο διευρύνονταν τα λεξιλόγια και η χρήση του διαδικτύου, τόσο εμφανή γινόντουσαν τα προβλήματα που υπεισέρχονταν.

Η τεχνολογία αναγνώρισης φωνής επανήλθε στο προσκήνιο μέσα από την άφιξη της εφαρμογής Google Voice Action στα κινητά τηλέφωνα android, η οποία στη συνέχεια ως Google Voice Search ενσωματώθηκε και στα κινητά τηλέφωνα iPhone. Το αντίκτυπο της συγκεκριμένης εφαρμογής ήταν τεράστιο για δύο κυρίως λόγους. Πρώτον, τα κινητά τηλέφωνα είναι ιδανικοί φορείς ικανότητας αναγνώρισης φωνής, καθώς η ανάγκη αντικατάστασης των μικρών τους πλήκτρων αποτελούσε κίνητρο για την ανάπτυξη πιο εύχρηστων μεθόδων εισαγωγής εντολών. Δεύτερον, η Google έχει τη δυνατότητα να αποφορτίζει την επεξεργασία ήχου με χρήση των υπολογιστικών νεφών (cloud data centers), και να χρησιμοποιεί όλη αυτή την τεράστια υπολογιστική ισχύ για ανάλυση μεγάλου όγκου δεδομένων. Αυτό ήταν απαραίτητο ώστε να γίνεται αντιστοιχία μεταξύ των λέξεων της φωνής του χρήστη και του τεράστιου όγκου παραδειγμάτων ανθρώπινης ομιλίας που συλλέγεται στο Cloud. Το 2010 η Google πρόσθεσε προσωποποιημένη αναγνώριση (Personalized Recognition) στη φωνητική αναζήτηση των κινητών Android. Έτσι, το λογισμικό μπορούσε να ηχογραφεί της φωνητικές αναζητήσεις του χρήστη, και μπορούσε τελικά να παράγει καλύτερα αποτελέσματα. Ήδη το 2011 η φωνητική αναζήτηση της Google περιλάμβανε αγγλικές λέξεις που προήλθαν από 230 δισεκατομμύρια φωνητικές αναζητήσεις χρηστών!

2.3.3 Περιγραφή Μεθόδων

Εισαγωγή

Η μοντελοποίηση μπορεί να γίνει τόσο με ακουστική μοντελοποίηση (Acoustic Modeling) όσο και με μοντελοποίηση γλώσσας (Language Modeling) καθώς και οι δύο επιλογές παίζουν καθοριστικό ρόλο στους σύγχρονους αλγορίθμους αναγνώρισης ομιλίας. Τα ακουστικά μοντέλα συσχετίζουν ηχητικά σήματα με φωνήματα που χρησιμοποιούνται για ομιλία. Ένα τέτοιο μοντέλο δημιουργείται από εισαγωγή ηχητικών σημάτων και τα αντίστοιχα κείμενα με το περιεχόμενο των σημάτων, και με χρήση λογισμικού κατασκευάζονται στατιστικές αναπαραστάσεις των ήχων που αντιστοιχούν στην κάθε ομιλούμενη λέξη. Αντίστοιχα, τα γλωσσικά μοντέλα δίνουν μια στατιστική κατανομή από αλληλουχία λέξεων και συχνά υιοθετούν τη Μαρκοβιανή υπόθεση για τη σχέση των λέξεων μέσα στην αλληλουχία.

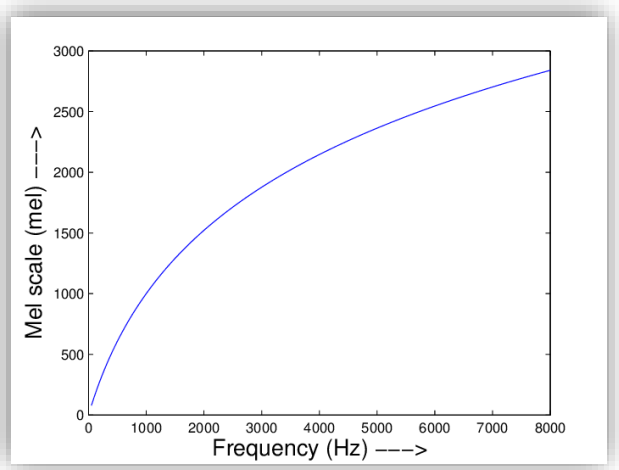
Αρχικά, κατά την επεξεργασία ηχητικού σήματος είναι απαραίτητο να γίνονται αντιληπτές οι παύσεις ανάμεσα στις λέξεις καθώς και να εξαλείφεται ο θόρυβος. Οι λέξεις ξεχωρίζονται από τις παύσεις και τον θόρυβο μέσω της ενέργειας του σήματος καθώς ένα σήμα ανθρώπινης φωνής έχει σαφώς μεγαλύτερη ενέργεια από ένα βουβό σήμα χωρίς ομιλητή. Στη συνέχεια χρησιμοποιούνται διάφοροι αλγόριθμοί προς αναγνώριση λέξεων. Για ρομποτικό έλεγχο πολλοί ερευνητές προτείνουν μεθόδους όπως οι φασματικοί συντελεστές Mel Cepstral (Mel Frequency Cepstral Coefficients: MFCC's) και δυναμική χρονική στρέβλωση (Dynamic Time Warping: DTW). Μετά την προεπεξεργασία χρησιμοποιούνται συχνά οι γραμμικοί προγνωστικοί συντελεστές (Linear Predictive Coefficient: LPC) για εξαγωγή των cepstrum και delta-cepstrum, και κρυφά Μαρκοβιανά μοντέλα (HMM) για εκπαίδευση του μοντέλου της βάσης δεδομένων φωνητικών σημάτων. Στη συνέχεια προτείνεται ο αλγόριθμος Viterbi προς εύρεση της βέλτιστης αλληλουχίας κατάστασης (optimal state sequence) ως δείγμα αναφοράς για αναγνώριση φωνής.

Mel Frequency Cepstral Coefficients (MFCC's)

Η κλίμακα mel, από τη λέξη μελωδία, είναι μια αντιληπτική κλίμακα τόνων που στα αυτιά των ακροατών ισαπέχουν ο ένας από τον άλλο όσον αφορά τη συχνότητα. Η συγκεκριμένη μη-γραμμική κλίμακα χρησιμεύει στην αναγνώριση φωνής γιατί αντιπροσωπεύει καλύτερα την απόκριση του ανθρώπινου συστήματος ακοής. Δηλαδή βασίζεται σε γνωστό κρίσιμο εύρος ζώνης (bandwidth) και συχνότητα του ανθρώπινου αυτιού. Το σημείο αναφοράς μεταξύ της κλίμακας mel και της κλασσικής μονάδας Hertz ορίζεται αναθέτοντας τον αντιληπτικό τόνο των 1000 mels σε τόνο με συχνότητα 1000 Hz, 40 dB πάνω από το κατώτατο όριο ακουστότητας του ακροατή. Πάνω από τα 500 Hz οι ακροατές κρίνουν ότι όλο και πιο μεγάλα διαστήματα παράγουν ίσες αυξήσεις του τόνου. Η καμπύλη της κλίμακας αυτή φαίνεται στο Σχήμα 2-23, και μια σχέση μετατροπής από Hz σε mel (Karjalainen, 1988) είναι η εξής:

$$\boldsymbol{m = 2595\, log_{10}\left(1 + \frac{f}{700}\right)}$$

2-12



Σχήμα 2-23. Κλίμακα mel

Ο όρος cepstrum προήλθε από αντιστροφή των πρώτων τεσσάρων γραμμάτων της λέξης spectrum (φάσμα). Διεργασίες σε cepstrums καλούνται ανάλυση quefrency (quefrency analysis, liftering, ή cepstral analysis). Ο όρος cepstrum (Bogert, 1963) χρησιμοποιείται ως εργαλείο για διερεύνηση περιοδικών δομών σε συχνοτικά φάσματα, οι οποίες παρατηρούνται ως αντίλαλοι ή ανακλάσεις σε αρμονικές συχνότητες.

Το cepstrum περιέχει πληροφορίες για ρυθμούς μεταβολής σε διαφορετικές φασματικές ζώνες και χρησιμοποιείται για τον καθορισμό της θεμελιώδους συχνότητας ανθρώπινης ομιλίας. Για την εξαγωγή του cepstrum χρειάζεται μετασχηματισμός στο πεδίο των quefrency, όπου η ανεξάρτητη μεταβλητή (quefrency) έχει χρονική κλίμακα. Για παράδειγμα, σε ρυθμό δειγματοληψίας σήματος 44100 Hz, παρουσιάζεται μια μεγάλη κορυφή (peak) στο cepstrum του οποίου το quefrency είναι 100 δείγματα (samples), και αυτό υποδεικνύει την ύπαρξη αρμονικής συχνότητας: 44100/100 = 441 Hz.

Οι φασματικοί συντελεστές (MFCC's) αθροιστικά αποτελούν ένα mel-frequency cepstrum (Sahidullah & Saha, 2012) και βρίσκονται ως εξής :

1. Μετασχηματισμός Fourier ενός παραθύρου ενός σήματος.
2. Μετατροπή στην κλίμακα mel, με χρήση τριγωνικής επικάλυψης (overlapping) ή επικάλυψη συνημίτονων.
3. Υπολογισμός λογαρίθμου σε κάθε συχνότητα.
4. Εφαρμογή Διακριτού Μετασχηματισμού Συνημίτονων (Discrete Cosine Transform) (Lap-Pui Chau et al., 2001) σε κάθε συχνότητα, και οι συντελεστές MFCC προκύπτουν ως τα πλάτη του τελικού φάσματος.

Οι φασματικοί συντελεστές MFCC's επιλέγονται ως χαρακτηριστικά (features) εκμάθησης σε εφαρμογές αναγνώρισης ομιλίας μέσω μηχανικής μάθησης, όπως π.χ. συστήματα που αναγνωρίζουν τη διάρθρωση αριθμών. Επιπροσθέτως, η χρήση των συντελεστών αυτών έχει ολοένα και αυξανόμενη εφαρμογή σε ανάκτηση πληροφοριών από μουσική (music information retrieval) όπως κατηγοριοποίηση φύλλου και μετρήσεις ακουστικής ομοιότητας. Σημειώνεται ότι τη δεκαετία του 2000 το European Telecommunications Standards Institute όρισε ένα κανονικοποιημένο αλγόριθμο MFCC's για εφαρμογή στα κινητά τηλέφωνα (European Telecommunications Standards Institute, 2003).

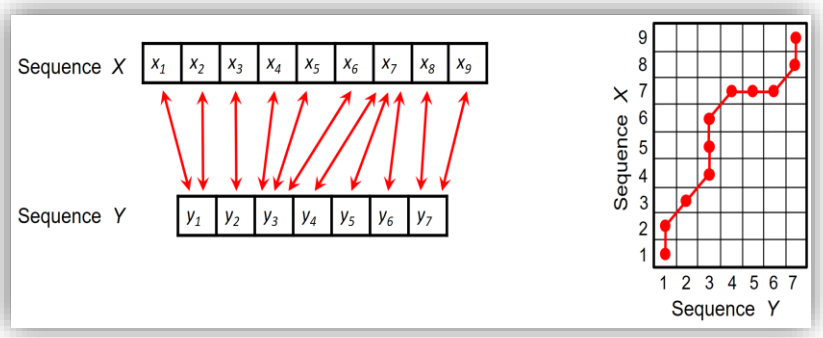
Δυστυχώς ο αλγόριθμος MFCC's δεν είναι πολύ σθεναρός στην επίδραση του θορύβου (noise). Για αυτό, πολλοί ερευνητές προτείνουν την ύψωση των πλατών σε κάποια δύναμη (2 ή 3) με σκοπό τη μείωση της επίδρασης όρων χαμηλής ενέργειας.

Δυναμική Χρονική Στρέβλωση (DTW)

Κατά την ανάλυση χρονοσειρών, ο αλγόριθμος DTW μετράει την ομοιότητα μεταξύ 2 χρονικών ακολουθιών που εξελίσσονται με διαφορετική ταχύτητα (Sakoe & Chiba, 1978). Για παράδειγμα, ο αλγόριθμος μπορεί να αναγνωρίζει ομοιότητα στην ομιλία ή στο περπάτημα δύο ανθρώπων ανεξάρτητα από τον προσωπικό τους ρυθμό ή τυχόν επιταχύνσεις στο βήμα τους. Εν γένει, οποιαδήποτε διακριτά δεδομένα, αν μπορούν να μετασχηματιστούν σε γραμμική αλληλουχία, μπορούν να αναλυθούν με χρήση του αλγόριθμου DTW. Μια πολύ σημαντική εφαρμογή του αλγορίθμου είναι και η αναγνώριση φωνής.

Η DTW είναι μια μέθοδος υπολογισμού της βέλτιστης ομοιογένειας μεταξύ 2 δεδομένων ακολουθιών (X,Y), δηλαδή τη βέλτιστη αλληλουχία μετασχηματισμών στην ακολουθία X (ή εισαγωγή ή διαγραφή ή αντικατάσταση στοιχείου ακολουθίας) με σκοπό αυτή να προσεγγίσει την ακολουθία Y. Συχνά γίνεται χρήση τοπικού παραθύρου (μήκους w) προς περιορισμό της απόστασης μεταξύ αντιστοιχισμένων δεικτών ($|i - j| \leq w$, για αντιστοιχισμένα i, j). Φυσικά, υπάρχουν κάποιοι κανόνες και περιορισμοί που πρέπει να ικανοποιούν οι δεδομένες χρονοσειρές:

1. Κάθε δείκτης (index) της 1ης ακολουθίας (X) αντιστοιχεί σε έναν ή περισσότερους δείκτες της 2ης ακολουθίας (Y), και το αντίστροφο.
2. Ο 1ος δείκτης της X πρέπει να αντιστοιχεί στον 1ο δείκτη της Y, αλλά δεν χρειάζεται η αντιστοιχία να είναι μοναδική.
3. Ο τελευταίος δείκτης της X πρέπει να αντιστοιχεί στον τελευταίο δείκτη της Y, αλλά δεν χρειάζεται η αντιστοιχία να είναι μοναδική.
4. Η αντιστοίχιση των δεικτών της X στην Y πρέπει να είναι αύξουσα, και το αντίστροφο. Δηλαδή, αν j>i είναι δείκτες της X, τότε, δεν μπορούν να υπάρχουν δείκτες p>q στην ακολουθία Y ώστε ο δείκτης i να αντιστοιχεί στο δείκτη p και ο δείκτης j να αντιστοιχεί στο δείκτη q, και το αντίστροφο. Το Σχήμα 2-24 προσφέρει οπτική αναπαράσταση του συγκεκριμένου κανόνα καθώς παρατηρείται αύξουσα αντιστοίχιση δεικτών.
5. Για τα μεγέθη των ακολουθιών X, Y (n,m αντίστοιχα) πρέπει να ισχύει $|n - m| \leq w$



Σχήμα 2-24. Αντιστοίχιση Δεικτών (DTW)

Η βέλτιστη ομοιογένεια προκύπτει από αντιστοίχιση που πληροί όλους τους περιορισμούς και ελαχιστοποιεί μια συνάρτηση κόστους (άθροισμα απόλυτων διαφορών). Εκτός από τη βέλτιστη ομοιογένεια ο αλγόριθμος παράγει και το μονοπάτι στρέβλωσης (warping path) πάνω

στο οποίο τα δύο σήματα εμφανίζονται ευθυγραμμισμένα στο χρόνο. Μια γενική εφαρμογή αυτού είναι ο συγχρονισμός ηχητικών σημάτων. Τα μειονεκτήματα είναι ότι ο αλγόριθμος DTW εφαρμόζεται μόνο σε διακριτά δεδομένα, έχει μεγάλο υπολογιστικό φορτίο, και πολλές φορές δεν εγγυάται την τριγωνική ανισότητα. Για αυτό συνήθως χρησιμοποιούνται παραλλαγές ή εναλλακτικές μέθοδοι όπως ο στοχαστικός SDTW και τα κρυφά Μαρκοβιανά μοντέλα.

Κρυφά Μαρκοβιανά Μοντέλα (HMM)

Τα μοντέλα HMM βρίσκουν εφαρμογή σε πεδία όπως θερμοδυναμική, στατιστική μηχανική, οικονομικά, επεξεργασία σήματος, θεωρία πληροφορίας και αναγνώριση μοτίβων (π.χ. ομιλία και γραφικός χαρακτήρας).

Μαρκοβιανή ονομάζεται οποιαδήποτε διεργασία της οποίας η μελλοντική εξέλιξη, με δεδομένες παρούσες συνθήκες, είναι πλήρως καθορισμένη, χωρίς να είναι απαραίτητο να γνωρίζουμε στοιχεία του παρελθόντος. Τα μοντέλα HMM χρησιμοποιούνται ευρέως στην αναγνώριση ομιλίας γιατί ένα ηχητικό σήμα μπορεί να θεωρηθεί τμηματικά στατικό για μικρά χρονικά διαστήματα, π.χ. για μικρή χρονική κλίμακα (~10 msec) η ομιλία μπορεί να θεωρηθεί στατική διεργασία (stationary process). Τα HMM είναι δημοφιλή γιατί εκπαιδεύονται εύκολα, είναι σχετικά απλά και υπολογιστικά εφικτά.

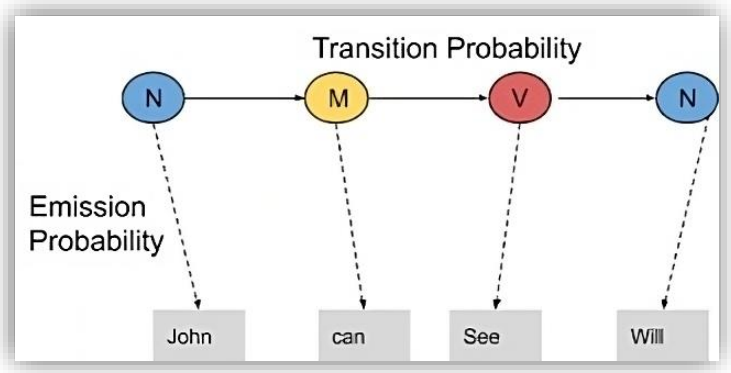
Ένα μοντέλο HMM (Xu et al., 2004) είναι ένα στατιστικό Μαρκοβιανό μοντέλο που θεωρείται ότι αποτελεί Μαρκοβιανή διαδικασία (έστω διαδικασία X) με μη-παρατηρήσιμες καταστάσεις (hidden states). Εξ ορισμού, η μέθοδος HMM απαιτεί την ύπαρξη μιας παρατηρήσιμης διαδικασίας (Y), οι έξοδοι της οποίας επηρεάζονται από τη διαδικασία X κατά γνωστό τρόπο, και σκοπός είναι να εξαχθούν πληροφορίες για τη διαδικασία X μέσω παρατήρησης της εξέλιξης της διαδικασίας Y. Μια ακόμα απαίτηση είναι ότι οι διαδικασίες X και Y συνδέονται μέσα από σχέση αιτιότητας, δηλαδή να αποτελούν ένα αιτιατό σύστημα. Ως έξοδος ενός μοντέλου HMM κάθε χρονική στιγμή είναι η εκ των υστέρων (a-posteriori) πιθανότητα, δηλαδή η πιθανότητα η διεργασία Y να πάρει μια συγκεκριμένη τιμή, δεδομένου ότι η διεργασία X πήρε μια συγκεκριμένη τιμή (με κάποια πιθανότητα).

Στην αναγνώριση ομιλίας, ένα HMM θα δίνει (π.χ. κάθε 10 msec) ως έξοδο ένα πραγματικό διάνυσμα n-διαστάσεων (π.χ. n =10, για μια λέξη). Η έξοδος θα περιλαμβάνει τους συντελεστές των cepstral από τους οποίους θα αξιολογούνται οι πρώτοι συντελεστές, δηλαδή οι πιο σημαντικοί. Το HMM θα τείνει να έχει ως κατάσταση (state) μια στατιστική κατανομή που είναι μίγμα από διαγώνιες Γκαουσιανές συνδιασπορές (Diagonal Covariance Gaussians) που θα δίνουν την πιθανοφάνεια (likelihood) του κάθε παρατηρούμενου διανύσματος, και κάθε λέξη ή φώνημα θα δίνει διαφορετική κατανομή εξόδου. Για μια αλληλουχία λέξεων (δηλαδή μια πρόταση) η έξοδος θα είναι συνένωση (concatenation) των εξόδων του μοντέλου για κάθε λέξη.

Ένα μοντέλο HMM συχνά περιλαμβάνει τα εξής μεγέθη:

- Κρυφές Καταστάσεις (Hidden States π.χ. POS Tags): π.χ. φράση ρήματος (VP), φράση ουσιαστικού (NP), κ.α.
- Πιθανότητα Εκπομπής (Emission Probability): $P(w_i \mid t_i)$ - Πιθανότητα η λέξη να είναι η w_i δεδομένου ότι η ετικέτα είναι η t_i, π.χ. P(book | NP).
- Μήτρα Πιθανότητας Μετάβασης (Transition Probability Matrix): $P(t_{i+1} \mid t_i)$ - Πιθανότητα μετάβασης από μια ετικέτα σε μια άλλη, π.χ. P(VP | NP).
- Παρατηρήσεις: Λέξεις
- Αρχική Πιθανότητα: Πιθανότητα της αρχικής λέξης.

Στο Σχήμα 2-25 φαίνεται ένα παράδειγμα εύρεσης μερών του λόγου (π.χ. ρήμα, ουσιαστικό, επίθετο, κλπ.) με χρήση HMM.

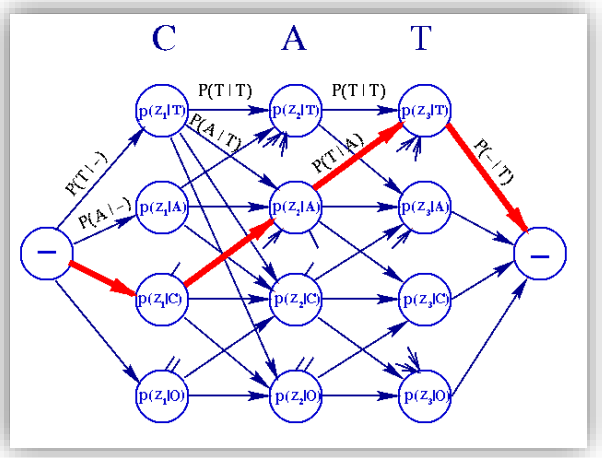


Σχήμα 2-25. POS tagging με χρήση HMM

Αλγόριθμος Viterbi

Ο αλγόριθμος Viterbi (Forney Jr, 2005) χρησιμοποιείται συχνά σε συνδυασμό με ένα μοντέλο HMM και πλέον θεωρείται βασικός αλγόριθμος δυναμικού προγραμματισμού σε προβλήματα βελτιστοποίησης που περιλαμβάνουν πιθανότητες.

Δεδομένης μιας αλληλουχίας παρατηρήσεων, ο αλγόριθμος Viterbi, ως αλγόριθμος δυναμικού προγραμματισμού, βρίσκει την εκτίμηση με την μεγαλύτερη εκ των υστέρων (a-posteriori) πιθανότητα για την πιθανότερη αλληλουχία (Viterbi path) κρυμμένων καταστάσεων που οδηγούν στις δεδομένες παρατηρήσεις. Για παράδειγμα, σε μια εφαρμογή μετατροπής φωνής σε γραπτό κείμενο, το ακουστικό σήμα αντιμετωπίζεται ως μια αλληλουχία παρατηρήσεων και το τελικό κείμενο αντιστοιχεί στην αλληλουχία κρυμμένων καταστάσεων που γενούν το δεδομένο ηχητικό σήμα. Ο αλγόριθμος Viterbi βρίσκει το πιθανότερο κείμενο που αντιστοιχεί σε δεδομένο σήμα. Το Σχήμα 2-26 δείχνει τη βέλτιστη αλληλουχία συμβόλων για αναγνώριση της λέξης CAT (γάτα), με χρήση προεκπαιδευμένου HMM και Viterbi.



Σχήμα 2-26. Βέλτιστη αλληλουχία γραμμάτων (Viterbi)

Μια εναλλακτική του αλγορίθμου Viterbi που έχει μεγάλο πρακτικό ενδιαφέρον είναι ο αναβλητικός αλγόριθμος Viterbi (Lazy Viterbi algorithm). Σε πολλές πρακτικές εφαρμογές, υπό την επίδραση εύλογου ποσοστού θορύβου, ο αναβλητικός Viterbi είναι πολύ πιο γρήγορος

από τον κλασσικό. Αυτό γιατί ο κλασσικός Viterbi λαμβάνει υπόψιν όλα τα πιθανά μονοπάτια που οδηγούν σε δεδομένο αποτέλεσμα ενώ η αναβλητική παραλλαγή διατηρεί μια λίστα προτεραιότητας για τα υπό διερεύνηση μονοπάτια, και τυπικά περιλαμβάνει λιγότερες μαθηματικές πράξεις από τον κλασσικό (ποτέ περισσότερες). Πολλές φορές όμως, η πρακτική εφαρμογή με χρήση κάποιου υλισμικού (hardware) είναι ιδιαίτερα δύσκολη.

2.3.4 Επιλογή Υπηρεσίας Ομιλίας

Όπως αναφέρθηκε, σκοπός της εργασίας είναι η ολοκλήρωση συστήματος αναγνώρισης ομιλίας και όχι η δημιουργία καινούργιου συστήματος. Για την επίτευξη των στόχων της εργασίας υπάρχουν αρκετά διαθέσιμα πακέτα και υπηρεσίες αναγνώρισης ομιλίας που χρησιμοποιούν μεθοδολογίες σαν αυτές που αναφέρθηκαν στην υποενότητα 2.3.3. Παραδείγματα αποτελούν τα δημοφιλή πακέτα Microsoft Speech Software Development Kit (SDK), Google Speech-To-Text, IBM Watson και AssemblyAI. Μετά από διερεύνηση και σύγκριση τέτοιων πακέτων, επιλέχθηκε η υπηρεσία Microsoft Speech Software Development Kit για τους σκοπούς της παρούσας εργασίας. Η επιλογή βασίστηκε στο ότι η συγκεκριμένη υπηρεσία προσφέρεται δωρεάν για ακαδημαϊκούς σκοπούς, η διαδικασία πρόσβασης και εγκατάστασης είναι σχετικά απλή, μετατρέπει σε πραγματικό χρόνο φωνητικά σήματα σε γραπτό λόγο, και προσφέρει δυνατότητα εξατομίκευσης λεξιλογίου.

Για να μπορέσουμε να αξιοποιήσουμε την υπηρεσία που μας ενδιαφέρει, πρέπει αρχικά να κάνουμε εγγραφή (subscription) δωρεάν στο Microsoft Azure καθώς και τη δημιουργία ενός μέσου ομιλίας (Speech Resource).

Κάνουμε χρήση του πακέτου Microsoft Speech Software Development Kit το οποίο αξιοποιεί πολλές από τις δυνατότητες της υπηρεσίας Speech Service και μας δίνει τη δυνατότητα να φτιάξουμε εφαρμογές που ενεργοποιούνται μέσω της φωνής (speech-enabled). Το Speech SDK είναι διαθέσιμο για συνδυασμό με πολλές γλώσσες προγραμματισμού (π.χ. C#, C++, Java, JavaScript, Python κ.α.), σε πολλές πλατφόρμες, και είναι ιδιαίτερα χρήσιμο τόσο για σενάρια πραγματικού χρόνου (real-time) όσο για σενάρια ασύγχρονης αναγνώρισης. Μια πολύ σημαντική διευκόλυνση που μας προσφέρει η συγκεκριμένη υπηρεσία είναι η αναγνώριση ψηφίων, ακόμα και δεκαδικών αριθμών. Φυσικά, είναι πιθανόν να προκύψουν σφάλματα τόσο στους αριθμούς όσο και στις λέξεις, και τα σφάλματα αυτά πρέπει να διορθώνονται, όπως θα εξηγηθεί στην Ενότητα 3.3.2.

Στην παρούσα εργασία, το Microsoft Speech SDK αναλαμβάνει όλο το έργο της μετατροπής φωνητικών εντολών σε γραπτό κείμενο και επιφέρει πολύ μεγάλη συνεισφορά στην επίτευξη των στόχων. Για περισσότερες πληροφορίες, ο αναγνώστης μπορεί να αναζητήσει την πηγή (MSAPI website).

2.4 Επεξεργασία Φυσικής Γλώσσας

2.4.1 Εισαγωγή

Η επεξεργασία φυσικής γλώσσας (Natural Language Processing: NLP) είναι ένα διεπιστημονικό πεδίο που συνδυάζει υπολογιστική γλωσσολογία, επιστήμη υπολογιστών και τεχνητή νοημοσύνη που ασχολείται με αλληλεπιδράσεις μεταξύ της φυσικής γλώσσας και του υπολογιστή. Πιο συγκεκριμένα, το NLP συχνά ασχολείται με τον προγραμματισμό υπολογιστή για την επεξεργασία ή και την κατανόηση περιεχομένου μεγάλου όγκου δεδομένων φυσικής γλώσσας, έχοντας ως είσοδο/έξοδο γραπτό ή και φωνητικό λόγο. Απώτερος σκοπός είναι να καταστεί ο υπολογιστής ικανός να κατανοεί το περιεχόμενο αρχείων συμπεριλαμβανομένου

τυχών μικροδιαφορών και ασαφειών (ambiguities) όσων αφορά το νόημα. Η συγκεκριμένη τεχνολογία είναι ικανή να εξάγει δεδομένα, να κατηγοριοποιεί και να οργανώνει αρχεία, να λαμβάνει αποφάσεις καθώς και να εξάγει ακριβείς περιλήψεις αρχείων. Συνήθεις προκλήσεις του NLP αφορούν αναγνώριση φωνής ή ομιλίας, κατανόηση φυσικής γλώσσας (Natural Language Understanding: NLU) και παραγωγή φυσικής γλώσσας (Natural Language Generation: NLG).

2.4.2 Ιστορική Αναδρομή

Η επεξεργασία φυσικής γλώσσας έχει τις ρίζες της στη δεκαετία του 1950. Ήδη το 1950 δημοσιεύτηκε ένα άρθρο όπου πρότεινε το γνωστό σήμερα Turing Test ως ένα κριτήριο νοημοσύνης για τους υπολογιστές (Turing, 1950). Το Turing Test περιέχει μια διαδικασία που περιλαμβάνει αυτόματη κατανόηση και παραγωγή φυσικής γλώσσας.

Το 1954 έλαβε χώρα το επονομαζόμενο Πείραμα Georgetown που περιλάμβανε πλήρως αυτόματη μετάφραση περισσότερων από 60 λέξεων από τα Ρωσικά στα Αγγλικά (Hutchins, 2004). Οι συγγραφείς ισχυρίζονταν ότι μετά από 3 έως 5 χρόνια η αυτόματη μετάφραση θα θεωρούνταν ένα λυμένο πρόβλημα. Όμως, στην πραγματικότητα η πρόοδος ήταν πολύ πιο βραδεία, και μετά το ALPAC Report το 1966, όπου φανερώθηκε δεκαετής αποτυχία εκπλήρωσης προσδοκιών, η χρηματοδότηση στο συγκεκριμένο πεδίο μειώθηκε δραματικά (Hutchins, 1996). Στη συνέχεια, πραγματοποιήθηκε ελάχιστη έρευνα στην αυτόματη μετάφραση μέχρι τα τέλη της δεκαετίας του 1980, όπου αναπτύχθηκε η πρώτη αυτόματη μετάφραση βασισμένη στη στατιστική.

Τη δεκαετία του 1960 αναπτύχθηκαν κάποια επιτυχή συστήματα επεξεργασίας φυσικής γλώσσας. Για παράδειγμα, το σύστημα SHRDLU του καθηγητή Terry Winograd στο MIT, λειτουργούσε με περιορισμένα λεξιλόγια (blocks worlds) (Ontanon, 2018). Το πρόγραμμα ήταν σε θέση να επικοινωνεί με το χρήστη, να δέχεται εντολές μετακίνησης αντικειμένων και να κατονομάζει συλλογές αντικειμένων. Επίσης, η ELIZA (Weizenbaum, 1966) ήταν μια προσομοίωση όπου χωρίς κανένα στοιχείο για την ανθρώπινη σκέψη και συναίσθημα, συχνά συμπεριφερόταν εντυπωσιακά παρόμοια με έναν άνθρωπο. Για παράδειγμα, αν ένας ασθενής της έλεγε «I have a headache.» η ELIZA ήταν σε θέση να απαντήσει ως «Why do you say that you have a headache?», δηλαδή δίνει μια απλοϊκή μεν απάντηση αλλά αρκετά ανθρωποφανή.

Τη δεκαετία του 1970 πολλοί προγραμματιστές άρχισαν να προγραμματίζουν εννοιολογικές οντότητες (conceptual ontologies) προς τη δόμηση πληροφοριών σε μορφή κατανοητή για τον υπολογιστή. Παραδείγματα είναι η MARGIE (Schank, 1975), η PAM (Wilensky, 1978) κ.α. Επίσης, εκείνη την περίοδο δημιουργήθηκαν και τα πρώτα chatbots όπως το PARRY (Colby, 1974).

Η δεκαετία του 1980 σηματοδοτεί το πέρας των συμβολικών μεθόδων NLP καθώς το ενδιαφέρον επικεντρώθηκε σε ανάλυση με βάση κανόνες (π.χ. γραμματικής), μορφολογία, σημασιολογία και άλλες μεθόδους κατανόησης φυσικής γλώσσας. Μέχρι τα τέλη της δεκαετίας του 1980 είχε γίνει επανάσταση στο NLP με την ανάπτυξη αλγορίθμων μηχανικής μάθησης για επεξεργασία γλώσσας. Η επανάσταση αυτή οφείλεται τόσο στη σταθερή αύξηση της υπολογιστικής ισχύος όσο και τη σταδιακή μείωση κυριαρχίας των θεωριών του Chomsky (Chomsky, 1986) στη γλωσσολογία. Αυτό γιατί τα θεωρητικά θεμέλια των θεωριών του Chomsky αποθάρρυναν τη χρήση βάσεων γλωσσικών δεδομένων (corpus linguistics) που αποτελούν βάση για την προσέγγιση της μηχανικής μάθησης στην επεξεργασία φυσικής γλώσσας.

Τη δεκαετία του 1990 αναπτύχθηκαν πολλές επιτυχημένες μέθοδοι NLP στο πεδίο της αυτόματης μετάφρασης, κυρίως μέσω ερευνητικής συνεισφοράς των μοντέλων ευθυγράμμισης (alignment models) της International Business Machines Corporation (IBM). Τα συστήματα αυτά αξιοποιούσαν δεδομένα από πολύγλωσσες βάσεις δεδομένων (textual corpora), που παράχθηκαν μαζικά από τη Βουλή του Καναδά και την Ευρωπαϊκή Ένωση, λόγω της δια νόμου υποχρέωσης μετάφρασης όλων των κυβερνητικών διαδικασιών σε όλες τις επίσημες γλώσσες των χωρών που συμμετείχαν. Είναι προφανές ότι ο όγκος δεδομένων είναι τεράστιος, όμως τα περισσότερα από αυτά τα συστήματα εξαρτώνταν από το περιεχόμενο και το θέμα της χρησιμοποιούμενης βάσης δεδομένων, και αυτό αποτελούσε μεγάλο εμπόδιο στις μεθόδους αυτές. Ως αποτέλεσμα, η έρευνα επικεντρώθηκε σε μεθόδους που έχουν την ικανότητα να μαθαίνουν αποτελεσματικά από περιορισμένο όγκο δεδομένων.

Μετά το 2000, προς εκμετάλλευση του αυξανόμενου όγκου δεδομένων και του διαδικτύου, η έρευνα επικεντρώθηκε σε αλγόριθμους μη επιτηρούμενης (unsupervised) και ημιεπιτηρούμενης (semi-supervised) μάθησης. Οι αλγόριθμοι αυτού του είδους μπορούν να εκπαιδευτούν μέσα από μίξη ταξινομημένων (labeled/annotated) και μη-ταξινομημένων κειμένων (unlabeled). Εν γένει, η μάθηση χωρίς επίβλεψη είναι πολύ πιο δύσκολη από την επιτηρούμενη μάθηση (supervised learning), και παράγει χειρότερα αποτελέσματα για συγκεκριμένο όγκο δεδομένων, καθώς τα πρώτα μαθαίνουν άγνωστα και κρυμμένα μοτίβα, ενώ τα δεύτερα εκπαιδεύονται με γνωστές απαντήσεις (labels).

Τη δεκαετία του 2010 η μάθηση αναπαράστασης και τα νευρωνικά δίκτυα βαθιάς μάθησης διαδόθηκαν παγκοσμίως στην επεξεργασία φυσικής γλώσσας. Αυτό γιατί άνθισαν παγκοσμίως πολύ καλά αποτελέσματα που έδειξαν ότι τέτοιες τεχνικές (Goldberg, 2016), (*Deep Learning*, n.d.) μπορούν να δώσουν state-of-the-art αποτελέσματα σε πολλές εφαρμογές, όπως μοντελοποίηση γλώσσας (language modeling) και συντακτική ανάλυση (parsing) (Vinyals et al., 2015), (Choe & Charniak, 2016). Αυτό έχει πολύ μεγάλη επίδραση στην ιατρική, όπου το NLP βοηθάει στην ανάλυση σημειώσεων και ηλεκτρονικών ιατρικών δεδομένων (electronic health records).

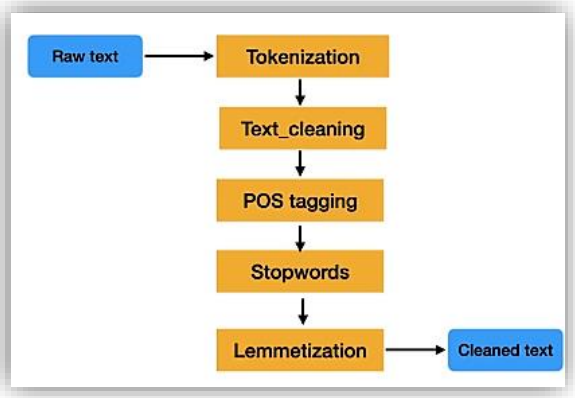
2.4.3 Περιγραφή

Συστατικά του NLP

Το NLP περιλαμβάνει αρκετές τεχνικές και μεθόδους με σκοπό την ανάλυση, την κατανόηση ή και την παραγωγή ανθρώπινης γλώσσας (Natural Language Processing, 2023). Χρησιμοποιούνται τεράστιες και δομημένες βάσεις δεδομένων (corpus) για την εκπαίδευση αλγορίθμων, στα πλαίσια του NLP. Στο Σχήμα 2-27 φαίνεται μια σύνοψη της ορολογίας στα πλαίσια του NLP, όπως ο κατακερματισμός λέξεων (tokenization), η σημασιολογική (semantic) ανάλυση, και η αναγωγή στη ρίζα λέξεων (stemming).

Σχήμα 2-27. Ορολογία NLP

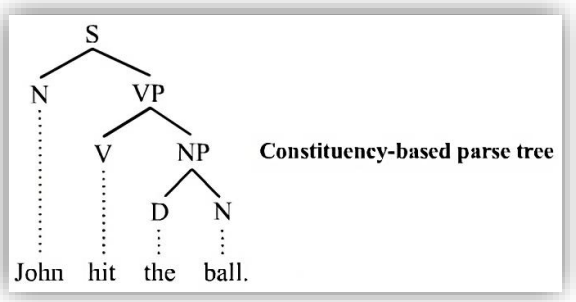
Ένα πρωταρχικό συστατικό του NLP είναι η λεξιλογική ανάλυση (Lexical Analysis), η οποία αφορά τον κατακερματισμό (Tokenization) κειμένων σε μικρές δομικές μονάδες που ονομάζονται tokens. Τα tokens μπορεί να είναι λέξεις, φράσεις, προτάσεις ή και ολόκληρες παράγραφοι, ανάλογα το βαθμό ανάλυσης και λεπτομέρειας που απαιτείται. Η διαδικασία αυτή είναι απαραίτητη για την αποτελεσματική ανάλυση φυσικής γλώσσας, και ανάλογα με τη γλώσσα ανάλυσης, οι δυσκολίες που υπεισέρχονται είναι σημαντικές, όπως γλώσσες όπου οι λέξεις, σε γραπτό λόγο, δεν έχουν κάποιο διαχωριστικό ή κενό διάστημα ανάμεσα τους (π.χ. Ιαπωνικά). Επίσης, πολλές λέξεις όπως σύνδεσμοι, άρθρα κλπ. (π.χ. «και», «το», «σε», κ.α.) μπορεί να παίζουν σημαντικό ρόλο συντακτικά αλλά δεν περιέχουν σημαντικό νόημα σε σχέση με την ουσία της πρότασης στην οποία ανήκουν. Οι λέξεις αυτές στην ορολογία του NLP ονομάζονται stop words και συνήθως αφαιρούνται κατά τα τελικά στάδια της προεπεξεργασίας. Ακόμα υπάρχει και η διεργασία λημματοποίησης (Lemmatization), η οποία επαναφέρει τις λέξεις στη ρίζα τους (π.χ. running, runner…→ run) και αυτό διευκολύνει σημαντικά την ανάλυση. Στο Σχήμα 2-28 φαίνεται μια τυπική διαδικασία καθαρισμού κειμένου.



Σχήμα 2-28. Τυπικός Καθαρισμός Κειμένου

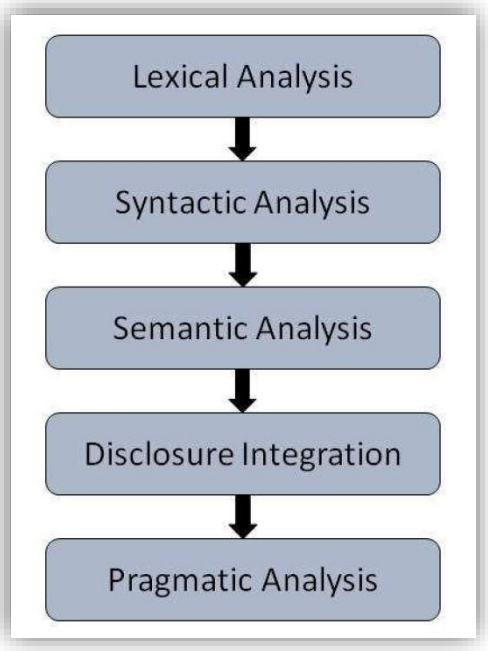
Μια ακόμα πολύ σημαντική διαδικασία στα πλαίσια του NLP είναι η διαδικασία συντακτικής ανάλυσης (Syntactic Parsing) κατά την οποία αναλύεται η γραμματική και η συντακτική δομή του συστήματος λέξεων, π.χ. σε μορφή δέντρου, όπως φαίνεται στο Σχήμα 2-29. Μια τέτοια ανάλυση θα απέρριπτε την πρόταση «The school goes to boy». Στα πλαίσια αυτής της

ανάλυσης εκτελείται και η διαδικασία εύρεσης των μερών του λόγου (Part-of-Speech: POS tagging). Σκοπός αυτής της διεργασίας είναι η κατηγοριοποίηση λέξεων στα πλαίσια της γραμματικής, δηλαδή η κατάταξη λέξεων σε κατηγορίες όπως ρήμα, ουσιαστικό, επίθετο, μετοχή κ.α. Η συγκεκριμένη διαδικασία έπαιξε σημαντικό ρόλο και στην παρούσα εργασία καθώς με βάση αυτή, δόθηκε ιδιαίτερη έμφαση σε ρήματα προτάσεων και σε αριθμητικά ψηφία.



Σχήμα 2-29. Τυπική Συντακτική Ανάλυση

Σειρά έχει η εννοιολογική ανάλυση (Semantic Analysis), η οποία ψάχνει νόημα στις λέξεις του κειμένου, με βάση κάποιο λεξικό. Αυτό επιτυγχάνεται με χαρτογράφηση συντακτικών δομών στο χώρο εργασίας (task domain). Μια τέτοια ανάλυση θα απέρριπτε την πρόταση «I ate a hot ice-cream». Ακολουθεί η ολοκλήρωση γνωστοποίησης (Disclosure Integration), μέσω της οποίας αναδύεται το νόημα διαδοχικών προτάσεων καθώς, το νόημα κάθε πρότασης σχετίζεται με το νόημα της αμέσως προηγούμενης. Τέλος, πραγματοποιείται πραγματολογική ανάλυση (Pragmatic Analysis) κατά την οποία ερμηνεύεται εκ νέου το σύνολο του κειμένου και συσχετίζεται με το νόημα που προκύπτει. Τα παραπάνω συνοψίζονται στο Σχήμα 2-30.



Σχήμα 2-30. Τυπικά Βήματα Ανάλυσης στο NLP

Γενικοί Σκοποί και Εφαρμογές του NLP

Εν γένει, οι στόχοι του NLP είναι οι εξής:

- Να μπορούν οι υπολογιστές να κατανοούν και να παράγουν ανθρώπινη γλώσσα, με τρόπο χρήσιμο και κατανοητό από τους ανθρώπους.
- Να διευκολύνουν την αλληλεπίδραση ανθρώπου-υπολογιστή.
- Να μπορούν οι υπολογιστές να εξάγουν χρήσιμες πληροφορίες από ανθρώπινη φωνή ή κείμενο.

Οι εφαρμογές στα πλαίσια του NLP είναι πολυάριθμες. Για παράδειγμα, υπάρχει η εφαρμογή περίληψης κειμένου, δηλαδή παραγωγή μιας πιο σύντομης εκδοχής του αρχικού κειμένου κρατώντας αυτούσιο το νόημα και τις χρήσιμες πληροφορίες που εμπεριέχονται σε αυτό. Υπάρχει επίσης η αυτόματη μετάφραση, η μοντελοποίηση γλώσσας, η αναγνώριση ομιλίας, η απάντηση ερωτήσεων, και η εξαγωγή πληροφοριών-συμπερασμάτων. Φυσικά, δε θα μπορούσε να παραλειφθεί μια από τις πιο σημαντικές εφαρμογές: η κατηγοριοποίηση εικόνων ή κειμένων (text classification). Η διεργασία αυτή έχει αναρίθμητες πρακτικές εφαρμογές, και είναι δυνατόν στο μέλλον ακόμα και να αναλάβει εξολοκλήρου τη σωστή και εύκολη κατηγοριοποίηση δεδομένων για λογαριασμό των ανθρώπων.

Επίσης δε θα μπορούσε να παραληφθεί η συναισθηματική ανάλυση (Sentiment Analysis) (Kasthuriarachchy et al., 2014), η οποία χρησιμοποιείται ευρέως από εταιρίες ως ανατροφοδότηση για την αξιολόγηση των προϊόντων τους, π.χ. από σχόλια στο διαδίκτυο. Σκοπός αυτής της εφαρμογής είναι να καθοριστεί ο συναισθηματικός τόνος, δηλαδή να εξαχθεί συμπέρασμα για το αν μια πρόταση υποδηλώνει θετικό, αρνητικό ή ουδέτερο συναίσθημα. Για παράδειγμα, μια πρόταση της μορφής: «This movie is really interesting» θα υποδήλωνε θετικό συναίσθημα, ενώ μια πρόταση της μορφής: «This product is not functional at all» θα υποδήλωνε αρνητικό συναίσθημα.

Λοιπά παραδείγματα εφαρμογών στα πλαίσια του NLP είναι η απομάκρυνση των κακόβουλων ή διαφημιστικών email (spam-filtering), η αναγνώριση εικόνας ή κειμένων εντός εικόνας, και το επονομαζόμενο Named Entity Recognition (NER) όπου γίνεται αναγνώριση ονομάτων όπως ονόματα ανθρώπων, περιοχών, οργανισμών ή ημερομηνιών.

Φυσική Γλώσσα και Βιομηχανικές Εφαρμογές

Η ρομποτική και το NLP είναι δύο ισχυρές τεχνολογίες που συχνά συνδυάζονται με σκοπό την ενίσχυση βιομηχανικής παραγωγής, κατεργασιών, διεργασιών και αυτοματισμών. Κατά την εφαρμογή σε βιομηχανικές συνθήκες, το NLP επιτρέπει στις μηχανές να επικοινωνούν και να συνεργάζονται με εργάτες και μηχανικούς πιο αποδοτικά, βελτιώνοντας την παραγωγικότητα, την απόδοση και την ασφάλεια. Οι βιομηχανικές εφαρμογές όπου συνδυάζονται η ρομποτική και το NLP είναι πολυάριθμές, και επιγραμματικά αναφέρονται τα παρακάτω αναμενόμενα οφέλη:

- Φωνητικός Έλεγχος Βιομηχανικού Βραχίονα: Όπως και στην παρούσα εργασία, με χρήση NLP μπορεί να δημιουργηθεί σύστημα αναγνώρισης φωνής και εντολών. Έτσι θα μπορούσε ένας μηχανικός ή ένας τεχνικός να ελέγχει τη συμπεριφορά ενός βραχίονα άμεσα, εκφράζοντας εντολές μέσω φυσικής γλώσσας. Το NLP επιτρέπει φυσική επικοινωνία ανθρώπου-μηχανής, κάτι που διευκολύνει σημαντικά τη συνεργασία σε βιομηχανικό περιβάλλον (Priyadarshana et al., 2022). Οι εργαζόμενοι μπορούν να δίνουν σαφείς οδηγίες ή να επικοινωνούν με βραχίονα, και αυτό υποβοηθά την επιθεώρηση συστημάτων, βελτιώνει την ομαδικότητα και αυξάνει την ακρίβεια εκτέλεσης επαναλαμβανόμενων διεργασιών. Επίσης, το NLP επιτρέπει την

κατανόηση και την επικοινωνία σε αρκετές γλώσσες, και αυτό είναι ιδιαίτερα προσοδοφόρο σε βιομηχανίες παγκόσμιας εμβέλειας που απασχολούν εργαζομένους διαφορετικών εθνικοτήτων. Αυτό καθιστά δυνατή την επεξεργασία αναφορών, οδηγιών και αιτήσεων σε διαφορετικές γλώσσες, κάτι που ενισχύει τη συνεργασία, μειώνει τις σχετικές καθυστερήσεις, και αυξάνει την απόδοση καλλιεργώντας ένα πολυπολιτισμικό εργασιακό περιβάλλον.

- Ανάκτηση Πληροφοριών Στοιχείων Κειμένου: Το NLP μπορεί να αξιοποιηθεί για ανάλυση κειμένων όπως αναφορές, φύλλα συντήρησης και οδηγίες χειριστή. Η αξιοποίηση αυτής της δυνατότητας μπορεί να συμβάλλει στη λήψη αποφάσεων για την αυτοματοποίηση και βελτιστοποίηση θεμάτων εφοδιαστικής (logistics) (Pollettini et al., 2015), αλλά και του κύκλου συντήρησης εξοπλισμού. Με την κατανόηση των πληροφοριών που προκύπτουν, τα συστήματα αυτά μπορούν να συνεισφέρουν στη συντήρηση συστημάτων και στην έγκαιρη πρόβλεψη βλαβών και σφαλμάτων. Επίσης, μέσω ανάλυσης μη-δομημένων κειμένων (text mining), το NLP μπορεί να συμβάλλει ενεργά στη βελτίωση ποιότητας προϊόντων (Rangu et al., 2017), καθώς επιτρέπει βαθύτερη κατανόηση των αναγκών της αγοράς. Όπως και στην παρούσα εργασία, μια διεπαφή προγραμματισμού μέσω φυσικής γλώσσας μπορεί να χρησιμοποιηθεί για να συμπτύξει διαφορετικές λειτουργίες και να προσφέρει ευελιξία και οικονομία χρόνου στον χειριστή.

Συνολικά, ο συνδυασμός NLP και βιομηχανίας ανοίγει το δρόμο σε νέες δυνατότητες για αποδοτικούς και ευφυείς αυτοματισμούς. Ακόμα, ενισχύει την αμεσότητα αλληλεπίδρασης ανθρώπου-ρομπότ και βελτιώνει συνολικά στην παραγωγικότητα και την ασφάλεια διεκπεραίωσης βιομηχανικών διεργασιών.

Τεχνικές Ανάλυσης του NLP

Οι τεχνικές που χρησιμοποιούνται στα πλαίσια του NLP ποικίλουν ανάλογα την εφαρμογή. Οι περισσότεροι από τους αλγορίθμους που αναφέρθηκαν στην Ενότητα 2.3.3 βρίσκουν άμεση εφαρμογή και στα πλαίσια του NLP.

Η παλαιότερη τεχνική βασίζεται σε κανόνες γραμματικής ή συντακτικού, (Rule-based). Σύμφωνα με αυτή την προσέγγιση, οι ειδικοί στη γλωσσολογία και την επιστήμη των υπολογιστών αναπτύσσουν κανόνες και μοτίβα, τα οποία ο υπολογιστής είναι σε θέση να αξιοποιήσει για την ανάλυση φυσικής γλώσσας. Αυτοί οι κανόνες μπορούν να χρησιμοποιηθούν για τη διόρθωση γραμματικών σφαλμάτων, κατηγοριοποίηση κειμένων ή την εξαγωγή συγκεκριμένων πληροφοριών από ένα κείμενο (Mahmud et al., 2015). Δυστυχώς όμως, η προσέγγιση μέσω κανόνων περιορίζεται από τον όγκο, την πολυπλοκότητα και τη μεταβλητότητα της ανθρώπινης γλώσσας με την πάροδο του χρόνου.

Ακολουθούν οι Στατιστικές Μέθοδοι, που βασίζονται σε μοντέλα πιθανοτήτων. Η προσέγγιση αυτή χρησιμοποιεί στατιστικούς αλγορίθμους για την ανάλυση μεγάλου όγκου δεδομένων κειμένου με σκοπό την εύρεση νοήματος και την εκμάθηση μοτίβων ή σχέσεων μεταξύ λέξεων. Τα μοντέλα αυτά συνήθως βασίζονται στη συχνότητα εμφάνισης λέξεων και την ταυτόχρονη εμφάνιση σχετικών μεταξύ τους λέξεων. Μερικές από τις πιο δημοφιλείς μεθόδους αυτής της κατηγορίας περιλαμβάνουν κρυφά Μαρκοβιανά μοντέλα, μοντέλα που βασίζονται σε ακολουθίες λέξεων (n-gram models), μοντέλα μέγιστης εντροπίας (Maximum Entropy Models) (Huang & Zhang, 2009), υπό συνθήκη τυχαία πεδία (Conditional Random Fields), και λανθάνουσα κατανομή Dirichlet (Latent Dirichlet Allocation: LDA).

Τα νευρωνικά δίκτυα βαθιάς μάθησης βρίσκονται σήμερα στη στάθμη της τεχνικής του NLP, και έχουν αυξήσει σημαντικά τη λειτουργικότητα και την ευκολία εκπαίδευσης γλωσσικών μοντέλων. Σκοπός είναι η εκμάθηση μοτίβων και σχέσεων, προσαρμόζοντας τα βάρη των νευρώνων μέσω επαναληπτικής διαδικασίας. Συχνά, τα μοτίβα που μαθαίνει ένα τέτοιο νευρωνικό δίκτυο δεν είναι ευδιάκριτα από τους ανθρώπους. Μερικές από τις πιο δημοφιλείς αρχιτεκτονικές στα πλαίσια του NLP περιλαμβάνουν συνελικτικά νευρωνικά δίκτυα (Convolutional Neural Networks: CNN's) (Li et al., n.d.), ανατροφοδοτούμενα νευρωνικά δίκτυα (Recurrent Neural Networks: RNN's), και μοντέλα μετασχηματισμών (Transformer models) (Wu et al., 2023). Υπάρχουν επίσης οι Υβριδικές μέθοδοι, οι οποίοι συνδυάζουν κανόνες, στατιστικά μοντέλα και νευρωνικά δίκτυα βαθιάς μάθησης με σκοπό την υπερνίκηση των αδυναμιών της κάθε προσέγγισης.

Υπάρχουν επίσης οι Γράφοι γνώσεων (Knowledge Graphs) όπου η γλώσσα αναπαρίσταται ως ένα δίκτυο διασυνδεδεμένων εννοιών και σχέσεων (Khadilkar et al., 2019). Οι Γράφοι γνώσεων συχνά βασίζονται σε σημασιολογικές τεχνολογίες στο διαδίκτυο όπου οργανώνουν πληροφορίες σε δομημένη μορφή, κατανοητή στους υπολογιστές και βρίσκουν εφαρμογή συνήθως σε συστήματα ερώτησης-απάντησης και chatbots.

Αναπαράσταση Λέξεων στο NLP

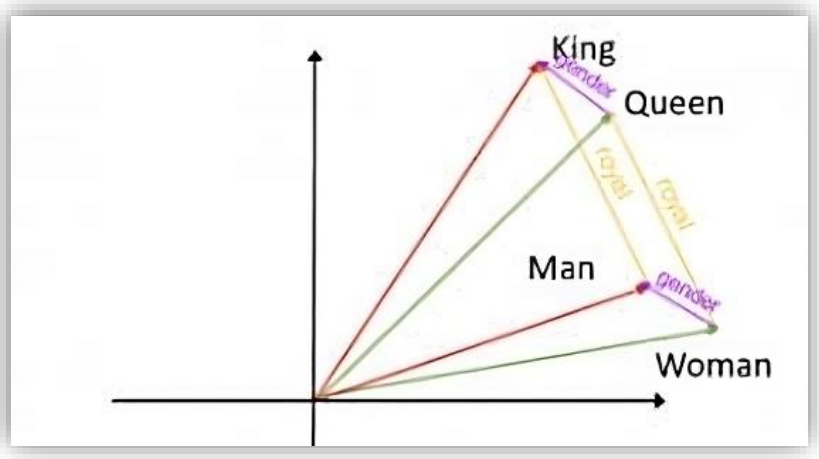
Στα πλαίσια του NLP συχνά πραγματοποιείται αναπαράσταση λέξεων σε μορφή διανύσματος ή πίνακα, ώστε να μπορούν να αναλυθούν στα πλαίσια ενός αλγορίθμου μηχανικής μάθησης. Οι διαφορετικοί τρόποι αναπαράστασης μπορούν να περιγράφουν την παρουσία ή και τη σημασιολογική σχέση των λέξεων.

Η τεχνική της κωδικοποίησης λέξεων (Word Encoding) αναπαριστά μοναδικά την κάθε λέξη ως ένα διάνυσμα με βάση π.χ. τη συντακτική του θέση, τη σημασιολογία, ή τη συχνότητα εμφάνισής της. Για παράδειγμα, η τεχνική One-hot-Encoding αναπαριστά τις λέξεις ως δυαδικά διανύσματα ανάλογα την παρουσία τους ή μέσω αριθμών που σχετίζονται με τη συχνότητα εμφάνισης τους. Μέσω της κωδικοποίησης, κάθε πρόταση ενός αρχείου κειμένων μπορεί να μετατραπεί σε ακολουθία καθορισμένων διαστάσεων και με βάση αυτή να εκπαιδευτεί ένα μοντέλο μηχανικής μάθησης.

Μια ακόμα μέθοδος αναπαράστασης λέξεων είναι το σύνολο (ή σάκος) λέξεων (Bag of Words: BoW) (Cummins et al., 2018), και αναπαριστά την κάθε λέξη ανάλογα τον αριθμό των φορών που υπάρχει στο κείμενο ή μέσω της σταθμισμένης συχνότητας εμφάνισής της (Term Frequency-Inverse Document Frequency: tf-idf). Η αναπαράσταση μέσω tf-idf είναι συνήθως καλύτερη γιατί λαμβάνει υπόψιν μια σταθμισμένη συχνότητα ανάλογα τόσο τη συχνότητα εμφάνισης λέξεων, αλλά και της συχνότητας εμφάνισης της στα υπόλοιπα κείμενα. Η tf-idf κάθε λέξης προκύπτει ως το γινόμενο της συχνότητας εμφάνισης της λέξης σε ένα αρχείο με το αντίστροφο της συχνότητας εμφάνισης της λέξης σε όλα τα υπό εξέταση αρχεία. Σαν αποτέλεσμα προκύπτει ένας αραιός πίνακας (sparse matrix), που ως στοιχεία έχει το tf-idf της κάθε λέξης. Μια επέκταση του συνόλου λέξεων είναι η αναπαράσταση σε σύνολο (ή σάκο) από n-grams (bigram, trigram, etc.) κατά την οποία μελετάμε αλληλουχίες λέξεων (n=2 ή 3) και όχι την παρουσία μεμονωμένων λέξεων. Παρόλο που ασχολούνται μόνο με τη συχνότητα εμφάνισης λέξεων και αγνοούν τη σειρά των λέξεων ή τη συντακτική δομή, αυτές οι μέθοδοι χρησιμοποιούνται ευρέως στην πράξη ειδικά για την κατηγοριοποίηση κειμένων.

Μια πιο εκλεπτυσμένη μορφή αναπαράστασης λέξεων είναι η ενσωμάτωση λέξεων (Word Embeddings) όπου κάθε λέξη αναπαρίσταται ως διάνυσμα πραγματικών αριθμών με συγκεκριμένες διαστάσεις (π.χ. 1x300). Η μετατροπή λέξεων σε διανύσματα που σχετίζονται

με τη σημασία συνήθως είναι αποτέλεσμα προεκπαιδευμένων νευρωνικών δικτύων από προγενέστερες έρευνες, αλλά επιδέχονται επεκτάσεις και προσαρμογή σε συγκεκριμένους στόχους (Kim et al., 2016). Η τεχνική αυτή είναι αρκετά προηγμένη γιατί μας επιτρέπει να βγάζουμε συμπεράσματα ως προς το νόημα των λέξεων, και να κάνουμε αριθμητικές πράξεις με τις λέξεις. Αυτό γιατί η θέση κάθε λέξης στον χώρο των χαρακτηριστικών σχετίζεται με το νόημά της, δηλαδή, λέξεις που είναι κοντά μεταξύ τους στον πολυδιάστατο αυτό χώρο, αναμένεται να έχουν και παρόμοιο νόημα. Για παράδειγμα, όπως φαίνεται στο Σχήμα 2-31, μέσω αυτής της αναπαράστασης θα μπορούσε κανείς να εξάγει συμπεράσματα μέσω αριθμητικών πράξεων όπως: King-Man+Woman≈Queen. Η μέθοδος ενσωμάτωσης λέξεων χρησιμοποιείται συχνά σε εφαρμογές συναισθηματικής ανάλυσης, αυτόματης μετάφρασης και NER. Τα πιο γνωστά προ εκπαιδευμένα μοντέλα μετατροπής ενσωμάτωσης λέξεων είναι τα εξής: Word2Vec (Mikolov et al., 2013), GloVe, και fastText (*English Word Vectors · fastText*, n.d.).



Σχήμα 2-31. Word Embedding Space

Μια ακόμα πιο προηγμένη παραλλαγή αυτής της μεθόδου ονομάζεται ενσωμάτωση λέξεων σε εννοιολογικά πλαίσια (Contextualized Word Embeddings). Η κυριότερη διαφορά με τις συμβατικές ενσωματώσεις λέξεων είναι ότι λαμβάνουν υπόψιν το νόημα της λέξης υπό το πρίσμα του θέματος (context) στο οποίο βρίσκονται. Για παράδειγμα, μπορεί εν γένει η λέξη crazy να σημαίνει τρελός αλλά ένα τέτοιο μοντέλο, μελετώντας π.χ. την πρόταση «We had a great time last night, it was crazy!» θα καταλάβαινε ότι στο συγκεκριμένο θέμα, η λέξη αυτή έχει άλλη σημασία, που παραπέμπει σε πολύ ευχάριστη βραδιά. Τα πιο γνωστά προ εκπαιδευμένα μοντέλα εννοιολογικής ενσωμάτωσης λέξεων/προτάσεων είναι τα μοντέλα ELMo, και BERT (Devlin et al., 2019).

2.5 Στατιστική - Ταξινόμηση - Αξιολόγηση

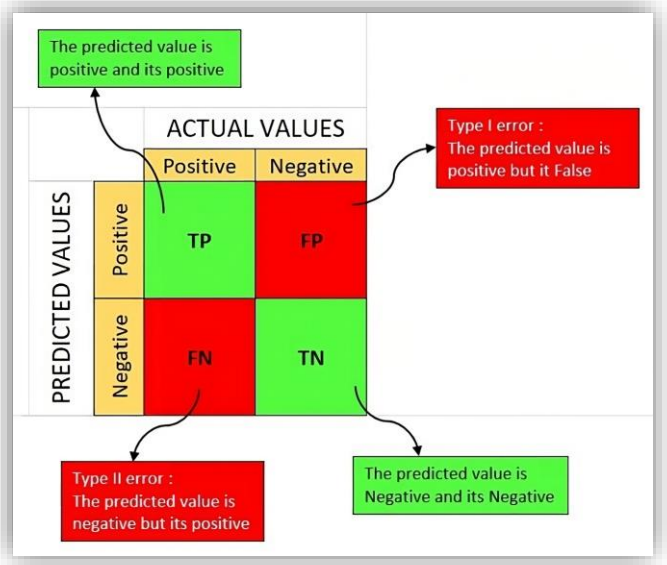
2.5.1 Στοιχεία Στατιστικής - Έλεγχος Υποθέσεων

Η κατηγοριοποίηση δεδομένων και οι φυσικές διαδικασίες καταγραφής και μετρήσεων εμπεριέχουν εγγενώς κάποια τυχαιότητα, οπότε η χρήση της στατιστικής καθίσταται αναπόφευκτη. Οι μεταβλητές ενδιαφέροντος αντιμετωπίζονται ως τυχαίες μεταβλητές που περιγράφονται από κάποια στατιστική κατανομή (π.χ. κανονική, Poisson, Laplace, διωνυμική κ.α.), και η διεξαγωγή συμπερασμάτων υπεισέρχεται συνήθως ως αριθμητικές εκτιμήσεις σε

μορφή αριθμών ή διαστημάτων εμπιστοσύνης (confidence intervals). Συχνά όμως τα συμπεράσματα δεν εκφράζονται αριθμητικά αλλά σχετίζονται με την επιλογή ανάμεσα σε τουλάχιστον δύο αμοιβαίως αποκλειόμενες υποθέσεις (hypothesis testing) (Devore, 2006).

Σε δυαδικά προβλήματα χρησιμοποιούνται δύο αμοιβαίως αποκλειόμενες υποθέσεις: η κενή υπόθεση (Null Hypothesis: H_0) και η εναλλακτική υπόθεση (Alternative Hypothesis: H_1). Ο τρόπος επιλογής ανάμεσα στην H_0 και την H_1 είναι εννοιολογικά παρόμοιος με το πώς ένας δικαστής καταφτάνει στην απόφασή του για την αθωότητα ή την ενοχή ενός κατηγορούμενου. Από τη στιγμή που υπάρχει το τεκμήριο της αθωότητας, ο κατηγορούμενος θεωρείται αθώος μέχρι να εμφανιστούν στοιχεία για την απόδειξη της ενοχής του, για αυτό και η θέση που υποστηρίζει ο κατηγορούμενος αντιστοιχίζεται στην κενή υπόθεση H_0. Σε κάθε περίπτωση, η κενή υπόθεση θεωρείται σωστή μέχρι να εμφανιστούν δεδομένα που αναδεικνύουν την επικράτηση της εναλλακτικής (H_1). Για την απόφαση απόρριψης της H_0 (δηλαδή αποδοχή της H_1) χρησιμοποιούμε στατιστικά μοντέλα, συνήθως η εναλλακτική υπόθεση αναπαριστά τη θεωρία που θέλουμε να ισχύει αν απορρίψουμε την H_0, και αυτό σημαίνει ότι απαιτείται ιδιαίτερη προσοχή στον ορισμό της H_1 ανάλογα τους σκοπούς του εκάστοτε προβλήματος.

Όμως, ακριβώς λόγω της εγγενούς τυχαιότητας, πάντα υπάρχει πιθανότητα σφάλματος για αυτό και έχουν οριστεί δύο είδη σφάλματος: το σφάλμα 1ου είδους και το σφάλμα 2ου είδους. Πιο συγκεκριμένα, στα πλαίσια του συστήματος δικαστής-κατηγορούμενος, το σφάλμα 1ου είδους (false positive or false alarm) αφορά την πιθανότητα φυλάκισης ενός αθώου, ενώ το σφάλμα 2ου είδους (false negative) αφορά την περίπτωση αθώωσης ενός ενόχου. Το Σχήμα 2-32 συνοψίζει τις έννοιες που αναφέρθηκαν σε έναν πίνακα σύγχυσης (confusion matrix).



Σχήμα 2-32. Confusion Matrix

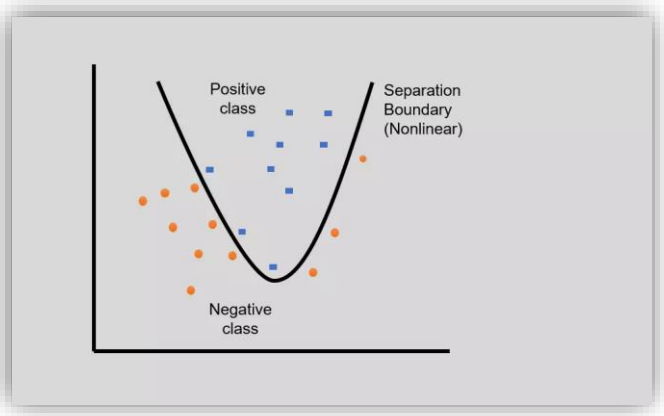
2.5.2 Δυαδική Κατηγοριοποίηση SVM

Όπως αναφέρθηκε, η κατηγοριοποίηση κειμένων βρίσκει μεγάλη εφαρμογή στα πλαίσια του NLP, όπως για φιλτράρισμα κακόβουλης αλληλογραφίας, συναισθηματική ανάλυση και θεματική κατηγοριοποίηση (Cortes & Vapnik, 1995). Η βασική ιδέα είναι ότι αυτά τα μοντέλα εκπαιδεύονται να μπορούν να κατηγοριοποιούν κείμενα σε προκαθορισμένες κατηγορίες (labels, categories ή classes), με βάση χαρακτηριστικά (features) από το περιεχόμενό τους. Το πρώτο βήμα είναι η προεπεξεργασία και η συμμόρφωση κειμένων, και στη συνέχεια η

αναπαράστασή του με μορφή της αρεσκείας μας. Είναι πολύ βασικό να αποφευχθεί η υπερπροσαρμογή (Overfitting) του μοντέλου καθώς στην περίπτωση αυτή το μοντέλο ακολουθεί επακριβώς τα δεδομένα εκπαίδευσης (training) αλλά αδυνατεί να περιγράψει ικανοποιητικά τα δεδομένα επαλήθευσης (testing) και να κάνει αξιόπιστες προβλέψεις σε νέα δεδομένα.

Για τους σκοπούς της εργασίας μας επιλέγεται ο αλγόριθμος Μηχανών Υποστήριξης Διανυσμάτων (Support Vector Machines: SVM), ο οποίος στη βάση του αναπτύχθηκε για προβλήματα δύο κλάσεων. Πιο συγκεκριμένα, ο αλγόριθμος λειτουργεί χαρτογραφώντας τα δεδομένα εισόδου σε έναν πολυδιάστατο χώρο χαρακτηριστικών (feature space), βρίσκοντας το βέλτιστο υπερεπίπεδο (hyperplane) που μεγιστοποιεί τα περιθώρια (margins) ανάμεσα σε 2 κλάσεις, αλλά ταυτόχρονα ελαχιστοποιεί το σφάλμα κατηγοριοποίησης. Τα περιθώρια είναι η απόσταση μεταξύ του βέλτιστου υπερεπιπέδου και των κοντινότερων σε αυτό σημείων από κάθε κλάση. Τα χαρακτηριστικά χρησιμοποιούνται για την κατασκευή του διανύσματος χαρακτηριστικών (feature vector) για κάθε παρατήρηση, και μπορεί να γίνει βέλτιστη επιλογή χαρακτηριστικών μέσω τεχνικών εκμάθησης (Feature Learning). Στόχος είναι, μετά την εκπαίδευση, το μοντέλο να είναι σε θέση να κατηγοριοποιεί νέα και άγνωστα δεδομένα σε μια από τις προκαθορισμένες κλάσεις, δηλαδή να προβλέπει την κλάση στην οποία ανήκει ένα στοιχείο κειμένου. Πολλές φορές είναι απαραίτητο τα δεδομένα να μετασχηματιστούν είτε με βάση τη μέση τιμή και τη διασπορά, είτε στο διάστημα μεταξύ 0 και 1 ώστε οι αποστάσεις των δεδομένων στο χώρο των χαρακτηριστικών (feature space) να έχουν ίδια τάξη μεγέθους. Η μέθοδος SVM γενικεύεται και για κατηγοριοποίηση σε πολλαπλές κλάσεις (π.χ. Error Correcting Output Code: ECOC), όπου κατασκευάζεται ένα σύνολο δυαδικών ταξινομητών, και η ανάλυσή τους δίνει την τελική κατηγοριοποίηση ανάμεσα σε πολλές κλάσεις.

Η μέθοδος SVM συνοδεύεται από πλεονεκτήματα αλλά και μειονεκτήματα. Κύριο πλεονέκτημα μοντέλων SVM είναι ότι μπορούν να διαχειριστούν πολυδιάστατα δεδομένα εισόδου, ακόμα και αν οι διαστάσεις είναι περισσότερες από τις παρατηρήσεις, και γενικεύονται εύκολα σε νέα και άγνωστα δεδομένα. Επίσης, είναι αποτελεσματικό για θορυβώδη δεδομένα καθώς και για δεδομένα που επικαλύπτονται. Στην πιο απλή μορφή της, η μέθοδος SVM σε 2D επίπεδο βρίσκει τη βέλτιστη γραμμική συνάρτηση που διαχωρίζει τα δεδομένα των 2 κλάσεων στο χώρο. Το πιο σημαντικό ίσως πλεονέκτημα είναι ότι, σε συνδυασμό με χρήση συνάρτησης πυρήνα (kernel trick), μπορούμε να κατηγοριοποιήσουμε και μη-γραμμικά διαχωρίσιμα δεδομένα, δηλαδή δεδομένα που δε διαχωρίζονται μέσω ευθείων γραμμών. Ουσιαστικά, με τη μέθοδο αυτή μπορούμε να βρούμε μη-γραμμικά όρια (boundaries) κλάσεων, με εικονικό μετασχηματισμό προς και από κατάλληλο πολυδιάστατο χώρο όπου τα δεδομένα χωρίζονται γραμμικά. Στο Σχήμα 2-33 φαίνεται ένα αυθαίρετο παράδειγμα όπου δύο κλάσεις δεδομένων που διαχωρίστηκαν από μη-γραμμική καμπύλη.



Σχήμα 2-33. Non-Linear SVM Classification

Παρόλα αυτά, η μέθοδος SVM μπορεί να γίνει ιδιαίτερα ακριβή υπολογιστικά, ιδίως για μεγάλο όγκο δεδομένων, και οι υπέρ-παράμετροι (hyperparameter tuning) χρειάζονται προσεκτική ρύθμιση για να έχουμε βέλτιστα αποτελέσματα. Εν γένει, ο αλγόριθμος SVM είναι πολύ ευαίσθητος τόσο στις τιμές των υπέρ-παραμέτρων όσο και στη μορφή της συνάρτησης πυρήνα.

2.5.3 Αξιολόγηση Μοντέλου Κατηγοριοποίησης

Στα πλαίσια της κατηγοριοποίησης, για την αξιολόγηση ενός μοντέλου χρησιμοποιούνται κάποιοι δείκτες (μετρικές) απόδοσης που πηγάζουν από πίνακες σύγχυσης, όπως αυτός στο Σχήμα 2-32, και χρησιμοποιούνται τόσο για δυαδικά προβλήματα όσο και για κατηγοριοποίηση ανάμεσα σε πολλές κλάσεις. Οι δείκτες αυτοί ορίζονται για κάθε κλάση, και συχνά χρησιμοποιείται η μέση τιμή τους για την αξιολόγηση των μοντέλων. Φυσικά, απαιτείται μεγάλη προσοχή καθώς οι δείκτες αυτοί μπορεί να γίνουν παραπλανητικοί, ανάλογα την ποιότητα τον δεδομένων εισόδου, δηλαδή να παρουσιάζουν πολύ καλές τιμές χωρίς το μοντέλο να είναι και στην πραγματικότητα αξιόπιστο.

Για καλύτερη κατανόηση, ας αναλογιστούμε ως παράδειγμα ένα πρόβλημα 2 κλάσεων (έστω κλάσεις A και B). Αξιολογώντας την επιτυχία του μοντέλου ως προς την κλάση A καταμετρούνται οι φορές που τα δεδομένα αναγνωρίστηκαν ορθά ως κλάση A (True Positive: TP), λανθασμένα ως κλάση A (False Positive: FP), ορθά ως κλάση B (True Negative: TN), και εσφαλμένα ως κλάση B (False Negative: FN). Αντίστοιχα, τα μεγέθη αυτά ορίζονται και κατά την αξιολόγηση του μοντέλου όσων αφορά την κλάση B, και το σύνολο όλων αυτών των καταμετρήσεων συνδυάζονται σε έναν πίνακα σύγχυσης.

Ο δείκτης accuracy σχετίζεται με την ακρίβεια των προβλέψεων, και η ερμηνεία του είναι εύκολη, ακόμα και διαισθητικά. Ουσιαστικά είναι ο λόγος των σωστών προβλέψεων προς το σύνολο όλων των προβλέψεων, δηλαδή:

$$\boldsymbol{accuracy} = \frac{\boldsymbol{TP+TN}}{\boldsymbol{TP+TN+FP+FN}}$$

2-13

Ο δείκτης precision (γνωστός και ως: Positive Predictive Value), σχετίζεται με την ευστοχία του μοντέλου, και ουσιαστικά είναι ο λόγος των ορθά θετικών προβλέψεων προς το σύνολο όλων των θετικών προβλέψεων. Όσο λιγότερες είναι οι λανθασμένα θετικές προβλέψεις (FP) για μια κλάση, τόσο το μέγεθος αυτό πλησιάζει τη μονάδα. Αν προκύψει precision=1 για την κλάση A σημαίνει ότι όλα τα δεδομένα που κατηγοριοποιήθηκαν ως κλάση A ήταν πράγματι

αυτής της κλάσης, αλλά δεν παρέχει καμία πληροφορία για τον αριθμό των στοιχείων της κλάσης Α που κατηγοριοποιήθηκαν λανθασμένα σε άλλη κλάση. Η σχέση ορισμού είναι η εξής:

$$\boldsymbol{precision} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FP}}$$

2-14

Ο δείκτης recall της κλάσης Α (γνωστό και ως True Positive Rate: TPR ή Sensitivity), είναι ο λόγος των ορθά θετικών προβλέψεων προς το σύνολο των προβλέψεων που στην πραγματικότητα ανήκουν στην κλάση Α. Συχνά ονομάζεται και ισχύς (power) ελέγχου υποθέσεων. Όσο λιγότερες είναι οι λανθασμένα αρνητικές προβλέψεις (FN) για μια κλάση, τόσο το μέγεθος αυτό πλησιάζει τη μονάδα. Αν προκύψει recall=1 για την κλάση Α σημαίνει ότι όλα τα δεδομένα που κατηγοριοποιήθηκαν ως κλάση Α ήταν πράγματι αυτής της κλάσης, αλλά δεν παρέχει καμία πληροφορία για τον αριθμό των στοιχείων άλλης κλάσης που κατηγοριοποιήθηκαν λανθασμένα στην κλάση Α. Η σχέση ορισμού είναι η εξής:

$$\boldsymbol{recall} = \boldsymbol{power} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FN}}$$

2-15

Πολύ συχνά τα μεγέθη precision και recall δεν είναι χρήσιμα αν αναλυθούν ξεχωριστά, και μια επιλογή είναι ο συνδυασμός τους σε μια σταθμισμένη παράμετρο. Υπάρχουν πολλοί τρόποι συνδυασμού και επιλογής βαρών ανάλογα ποιο από τα δύο μεγέθη παίζει πιο καθοριστικό ρόλο, αλλά στην παρούσα εργασία θα ασχοληθούμε με τον δείκτη f-score. Ο δείκτης f-score είναι ένας σταθμισμένος αρμονικός μέσος των δεικτών precision και recall και η σχέση υπολογισμού που θα χρησιμοποιήσουμε είναι η εξής:

$$\boldsymbol{f_1} = \mathbf{2}\frac{\boldsymbol{recall*precision}}{\boldsymbol{recall+precision}}$$

2-16

Ο δείκτης specificity (True Negative Rate: TNR), αναπαριστά το ρυθμό με τον οποίο ένα μοντέλο ορθώς αναγνωρίζει δεδομένα αρνητικής κλάσης ως αρνητικά. Ως μέγεθος είναι συμπληρωματικό του sensitivity και αναπαριστά την ικανότητα ενός μοντέλου να προβλέπει ορθώς αρνητικές κλάσεις, ενώ αποφεύγει ψευδώς θετικές προβλέψεις, και υπολογίζεται ως εξής:

$$\boldsymbol{specificity} = \frac{\boldsymbol{TN}}{\boldsymbol{TN+FP}}$$

2-17

Ο ρυθμός ψευδώς θετικής ταξινόμησης (False Positive Rate: FPR) ουσιαστικά αποτελεί την πιθανότητα false alarm ή με άλλα λόγια, αποτελεί μέτρο της ισχύος ενός κανόνα λήψης αποφάσεων ως προς το στατιστικό σφάλμα 1^{ου} είδους. Αντίστοιχα, ο ρυθμός ψευδώς αρνητικής ταξινόμησης (False Negative Rate: FNR) ουσιαστικά αποτελεί μέτρο της ισχύος ενός κανόνα λήψης αποφάσεων ως προς το στατιστικό σφάλμα 2^{ου} είδους. Ο ρυθμός αληθώς θετικής ταξινόμησης (True Positive Rate: TPR) ταυτίζεται με τον δείκτη recall, ενώ οι δείκτες False Positive Rate και False Negative Rate υπολογίζονται ως εξής:

$$\boldsymbol{False\ Positive\ Rate} = \mathbf{1} - \boldsymbol{specificity} = \frac{\boldsymbol{FP}}{\boldsymbol{TN+FP}}$$

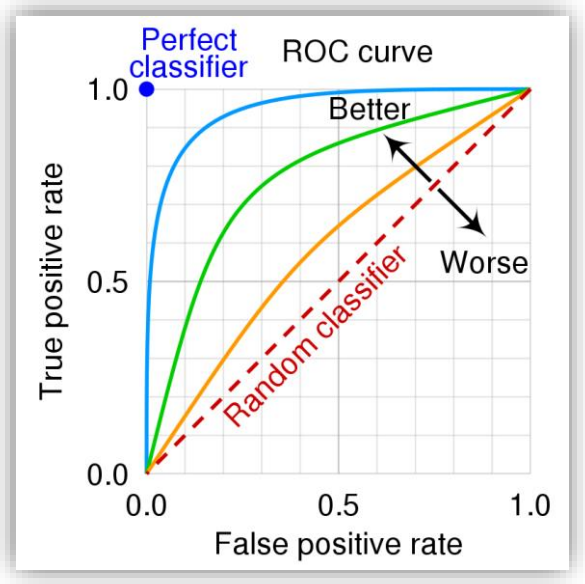
2-18

$$\textbf{\textit{False Negative Rate}} = 1 - \textbf{\textit{sensitivity}} = \frac{FN}{FN+TP}$$

2-19

Από την Εξίσωση 2-18 και την Εξίσωση 2-19 είναι φανερό ότι οι δείκτες αυτοί είναι συμπληρωματικοί, και θέλουμε να τείνουν όσο γίνεται στο μηδέν. Ο συσχετισμός των TPR και FPR δίνει τελικά τη χαρακτηριστική καμπύλη δέκτη, (γνωστή και ως Receiver Operating Characteristic: ROC) η οποία αρχικά χρησιμοποιήθηκε σε στρατιωτικούς δέκτες ραδιοκυμάτων κατά το 2ο παγκόσμιο πόλεμο. Έκτοτε χρησιμοποιείται ευρέως σε πολλά πεδία όπως ιατρική, ραδιολογία, μετεωρολογία, μηχανική μάθηση, εξόρυξη δεδομένων (data mining) και αξιολόγηση στατιστικών μοντέλων κατηγοριοποίησης (classification) ή παλινδρόμησης (regression) (Cai et al., 2010).

Η καμπύλη ROC είναι εργαλείο για την επιλογή βέλτιστου ταξινομητή ανάμεσα σε πολλές υποβέλτιστες επιλογές και αποτελεί φυσικό τρόπο ανάλυσης οφέλους/κόστους κατά τη διαγνωστική λήψη αποφάσεων. Το όφελος αντιστοιχεί σε TPR και το κόστος αντιστοιχεί σε FPR. Μετά την εκπαίδευση ενός μοντέλου και μετά τις προβλέψεις, επιστρέφονται και οι εκ των υστέρων πιθανότητες (ή scores), με βάση τις οποίες έγινε η κατηγοριοποίηση. Οι πιθανότητες που προκύπτουν περιέχουν ένα εσωτερικό όριο (threshold) που διαχωρίζει την μια κλάση από την άλλη, και η επιλογή αυτού του ορίου είναι κρίσιμη καθώς, διαφορετικές τιμές του ορίου θα οδηγούσαν σε διαφορετικές προβλέψεις. Η καμπύλη ROC αναπαριστά τη γραφική παράσταση που συνδέει τα μεγέθη TPR (άξονας-y) και FPR (άξονας-x) για διαφορετικές τιμές του ορίου διάκρισης, όπως φαίνεται στο Σχήμα 2-34. Είναι προφανές ότι ένας τέλειος ταξινομητής θα παρουσίαζε TPR=1 και FPR=0 (100% sensitivity and 100% specificity) για οποιαδήποτε τιμή του threshold, ένας χείριστος ταξινομητής θα παρουσίαζε TPR=0 και FPR=1, ενώ ένας ταξινομητής που ταξινομεί με τυχαίο τρόπο θα κινούνταν στην καμπύλη TPR= FPR. Για τη σύγκριση μεταξύ καμπυλών χρησιμοποιείται συνήθως το εμβαδόν κάτω από την καμπύλη (Area Under Curve: AUC), το οποίο θέλουμε να είναι μέγιστο. Η διαδικασία υπολογισμού του εμβαδού είναι περίπλοκη και κρύβει παγίδες αλλά, η εγγενής MATLAB συνάρτηση *perfcurve* μας επιστρέφει τόσο την καμπύλη ROC όσο και το μέγεθος AUC.



Σχήμα 2-34. Καμπύλη ROC

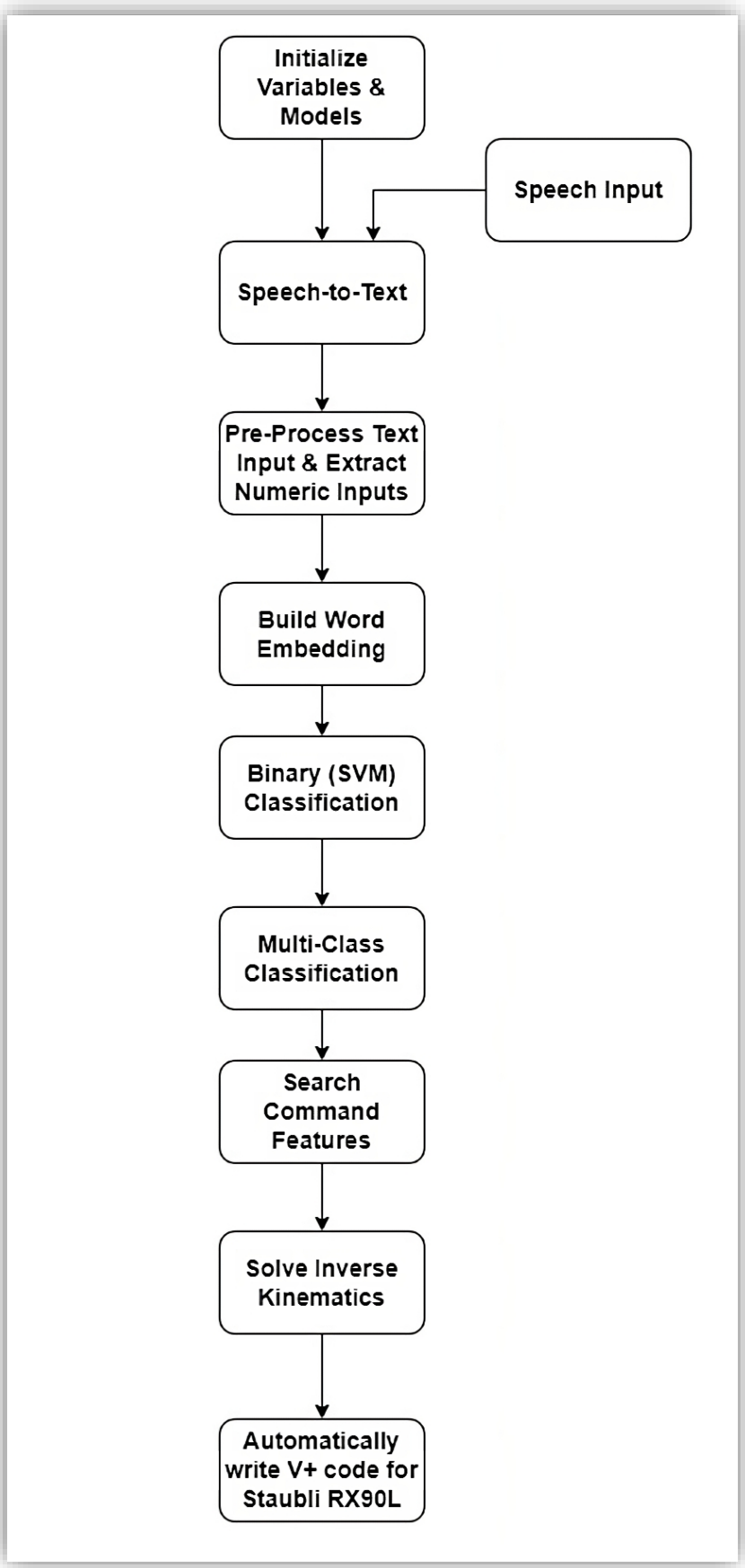
3 Περιγραφή-Παρουσίαση Συστήματος

3.1 Εισαγωγή

Όπως αναφέρθηκε, η παρούσα μεταπτυχιακή εργασία πραγματεύεται την ολοκλήρωση συστήματος προγραμματισμού ρομποτικού βραχίονα μέσω φωνητικών εντολών.

1. Το μεγαλύτερο μέρος της εργασίας υλοποιήθηκε στο MATLAB R2022b, εκτός από τη μετατροπή φωνής σε κείμενο, η οποία πραγματοποιήθηκε με χρήση της Python, καλούμενη εντός του MATLAB. Ο κώδικας Python αξιοποιεί την υπηρεσία Microsoft Speech SDK για μετατροπή φωνής σε κείμενο.
2. Το φωνητικό σήμα, μετά την μετατροπή του σε γραπτό κείμενο, περνάει από προεπεξεργασία με σκοπό τη μορφοποίηση και τον καθαρισμό πιθανών σφαλμάτων κατά την αρχική μετατροπή της φωνής σε κείμενο. Ακολουθεί η εξαγωγή αριθμητικών δεδομένων.
3. Στη συνέχεια, δημιουργείται μια ενσωμάτωση λέξεων (Word Embedding) με σκοπό την ευφυή αναπαράστασή τους.
4. Έπειτα, το σύνολο των προτάσεων του χρήστη περνάει από ένα μοντέλο δυαδικής κατηγοριοποίησης (binary classification) που έχει ως σκοπό να αναγνωρίσει ποιες από τις προτάσεις του χρήστη αντιστοιχούν σε γνωστές εντολές και ποιες είναι άσχετες με τους σκοπούς της εργασίας.
5. Στη συνέχεια, το σύνολο των προτάσεων που πια θεωρείται ότι αντιστοιχούν σε γνωστές εντολές, περνάει από ένα σύνολο κανόνων μοναδικής αντιστοίχισης λέξεων-κλειδιά, με σκοπό την αντιστοίχιση της κάθε πρότασης προς μια εντολή της γλώσσας V^+.
6. Έπειτα, διερευνώνται τα χαρακτηριστικά των εντολών και σε περίπτωση που ο χρήστης το επιθυμεί, μπορεί να ελέγξει και να διορθώσει μία προς μία τις εντολές που έτυχε να κατηγοριοποιηθούν εσφαλμένα.
7. Στη συνέχεια, και εφόσον απαιτείται, γίνεται επίλυση της αντίστροφης κινηματικής του ρομποτικού βραχίονα, έτσι ώστε τελικά να δοθεί εντολή να κινηθεί σε κάποιο precision point.
8. Τέλος, ανάλογα τα όσα προηγήθηκαν στην παρούσα διαδικασία, παράγεται αυτόματα ένα αρχείο που περιέχει το σύνολο των εντολών σε γλώσσα V^+, η εκτέλεση του οποίου πραγματοποιεί τις εντολές που έδωσε φωνητικά ο χρήστης.

Το διάγραμμα ροής της συγκεκριμένης διαδικασίας περιγράφεται σχηματικά στο Σχήμα 3-1.



Σχήμα 3-1. Διάγραμμα Ροής Προγράμματος

3.2 Κατηγοριοποίηση Εντολών

3.2.1 Εκπαίδευση Μοντέλου Δυαδικής Κατηγοριοποίησης

Η κατηγοριοποίηση των εντολών βασίστηκε στη μέθοδο μηχανών υποστήριξης διανυσμάτων (Support Vector Machines: SVM) η οποία περιγράφηκε στην Ενότητα 2.5.2. Όπως αναφέρθηκε στην Ενότητα 2.1.5, για τις φωνητικές εντολές χρειαστήκαμε 7 εντολές από τη γλώσσα V^+. Επίσης, θεωρήσαμε ακόμα μια εντολή Top Level τύπου «OK robot» με την οποία πρέπει να ξεκινάει μια αλληλουχία εντολών.

Για την εκπαίδευση δυαδικού μοντέλου κατασκευάστηκε ένα φύλλο Excel που περιέχει 80 (70+10) προτάσεις, δηλαδή 10 προτάσεις που αφορούν την κάθε εντολή καθώς, και επιπλέον 80 προτάσεις που δεν έχουν καμία σχέση με τον ρομποτικό βραχίονα ή την κίνησή του (συνολικά 160 προτάσεις). Φροντίσαμε με αυτόν τον τρόπο να υπάρχει ίσος αριθμός δεδομένων για κάθε κλάση, δηλαδή τα δεδομένα να είναι ισορροπημένα (balanced). Τα δεδομένα εκπαίδευσης αναγράφονται στο Παράρτημα Β.

Για την πιστοποίηση των μοντέλων εφαρμόσαμε τη μέθοδο επικυρούμενης διασταύρωσης (k-fold cross-validation), δημιουργώντας 4 υποσύνολα από δεδομένα για εκπαίδευση και δοκιμή. Κατά την εφαρμογή τα δεδομένα χωρίζονται τυχαία σε k υποσύνολα (partitions) ίσου μεγέθους. Για καθένα από τα k υποσύνολα χρησιμοποιούνται k-1 υποσύνολα για εκπαίδευση και αυτό που μένει χρησιμοποιείται για επαλήθευση, και η διαδικασία αυτή επαναλαμβάνεται k φορές, αφήνοντας πάντα ένα διαφορετικό υποσύνολο στην άκρη για testing. Στο τέλος μπορεί να προκύψει μια μοναδική εκτίμηση (πρόβλεψη) παίρνοντας το μέσο όρο των επιμέρους προβλέψεων ή την πρόβλεψη που κερδίζει σε πλειοψηφία ανά τα υποσύνολα (folds). Τα δεδομένα χωρίστηκαν σε όλες τις περιπτώσεις ως εξής: ποσοστό για training: 75% (120 προτάσεις) και ποσοστό για testing: 25% (40 προτάσεις).

Αρχικά, θέλαμε να διαπιστώσουμε ποιος είναι ο καλύτερος τρόπος εισαγωγής δεδομένων στο μοντέλο κατηγοριοποίησης. Οι επιλογές που εξετάσαμε, μετά από προεπεξεργασία κειμένου, ήταν οι εξής:

- Word Embedding
- Word Encoding
- Bag of Words

Πριν την εκπαίδευση των μοντέλων βρέθηκαν τα βέλτιστα μοντέλα με χρήση της ρύθμισης *OptimizeHyperparameters* από όπου βρέθηκε η βέλτιστη μορφή συνάρτησης πυρήνα (kernel function), το αν χρειάζεται τυποποίηση (standardization), καθώς και η βέλτιστη τιμή του μεγέθους Box Constraint το οποίο μειώνει την υπερπροσαρμογή.

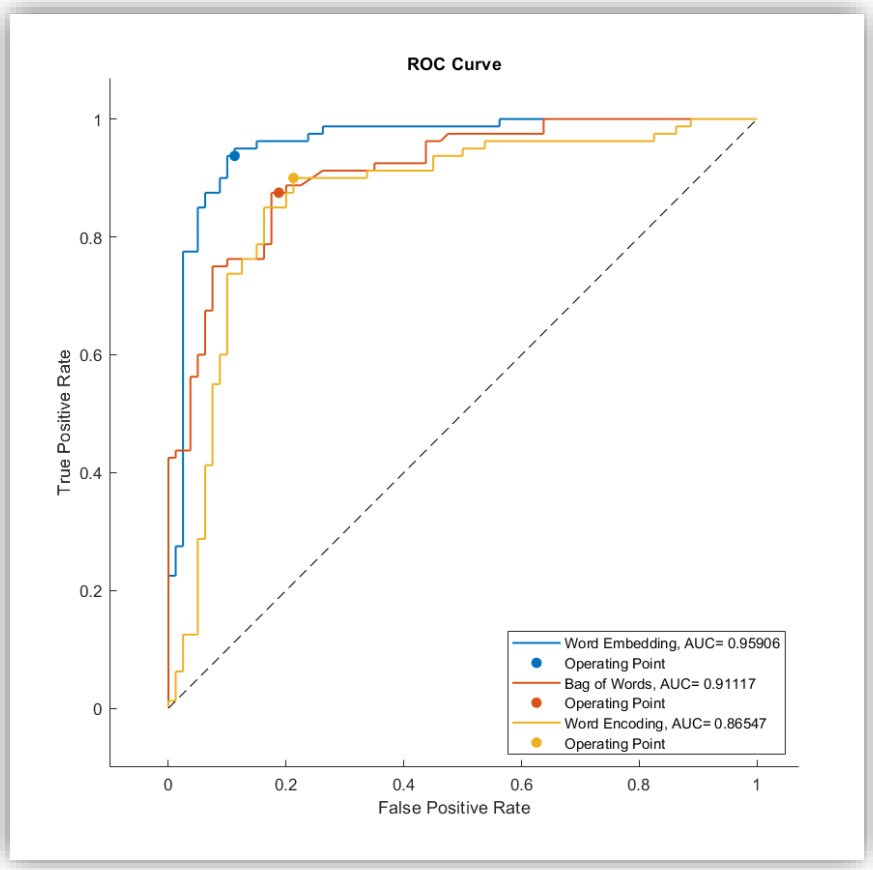
Στη συνέχεια έγινε χρήση της εγγενούς MATLAB συνάρτησης `kfoldLoss` υπολογίστηκε το σφάλμα κατηγοριοποίησης όπου για κάθε υποσύνολο (fold) έγιναν προβλέψεις με χρήση των υπόλοιπων ταξινομητών που εκπαιδεύτηκαν με δεδομένα διαφορετικού υποσυνόλου (out-of-fold). Τις τιμές που προκύπτουν δείχνει ο Πίνακας 3-1.

Πίνακας 3-1. k-fold loss

Word Embedding	Word Encoding	Bag of Words
8.75%	16.25%	15.62%

Για την περεταίρω διερεύνηση της σύγκρισης αξιοπιστίας των μοντέλων κατασκευάστηκε για κάθε μοντέλο η καμπύλη ROC (μέσω της εγγενούς συνάρτησης *perfcurve*) η οποία περιγράφηκε στην Ενότητα 2.5.3. Στο Σχήμα 3-2 φαίνονται οι καμπύλες ROC καθώς και τα βέλτιστα σημεία λειτουργίας των μοντέλων ως προς την κλάση που αντιστοιχεί σε εντολές V^+

(Class: True). Είναι φανερό ότι το SVM μοντέλο που λειτουργεί με βάση Word Embeddings υπερέχει από τα υπόλοιπα ως προς την αξιοπιστία των αποτελεσμάτων του καθώς παρουσιάζει και τη μεγαλύτερη τιμή AUC.



Σχήμα 3-2. Σύγκριση Καμπύλων ROC

Στη συνέχεια, για περαιτέρω έλεγχο ποιότητας των μοντέλων κατασκευάστηκε ένα νέο αρχείο Excel που περιλαμβάνει 100 προτάσεις, 50 από τις οποίες αποτελούν πράγματι εντολές V^+. Τα δεδομένα αναγράφονται επίσης στο Παράρτημα Β. Αναπόφευκτα οι 50 προτάσεις θα μοιάζουν ελαφρώς με τα δεδομένα εκπαίδευσης, καθώς το λεξιλόγιο είναι εγγενώς περιορισμένο, αλλά για πιο ουσιαστική πιστοποίηση, οι 50 άκυρες προτάσεις είναι εντελώς άγνωστες, και περιλαμβάνουν λέξεις που θα μπορούσαν παραπλανητικά να παραπέμπουν σε εντολή (π.χ. My joints are broken). Παραθέτουμε τους πίνακες σύγχυσης που προέκυψαν για το κάθε μοντέλο όπου και πάλι φαίνεται πως το μοντέλο που χρησιμοποιεί word embedding έχει την καλύτερη συμπεριφορά. Σε κάθε περίπτωση, ακόμα και σε άγνωστα δεδομένα το μοντέλο που εκπαιδεύτηκε με ενσωμάτωση λέξεων φαίνεται να προσφέρει τις πιο αξιόπιστες συγκριτικά προβλέψεις. Η κύρια αδυναμία των υπόλοιπων μοντέλων πηγάζει από την αδυναμία ορθής απόρριψης μη-εντολών.

Πίνακας 3-2. Confusion Matrix (word embedding)

Predicted / Actual	True	False
True	49	1
False	3	47

Πίνακας 3-3. Confusion Matrix (word encoding)

Actual \ Predicted	True	False
True	47	3
False	16	34

Πίνακας 3-4. Confusion Matrix (bag of words)

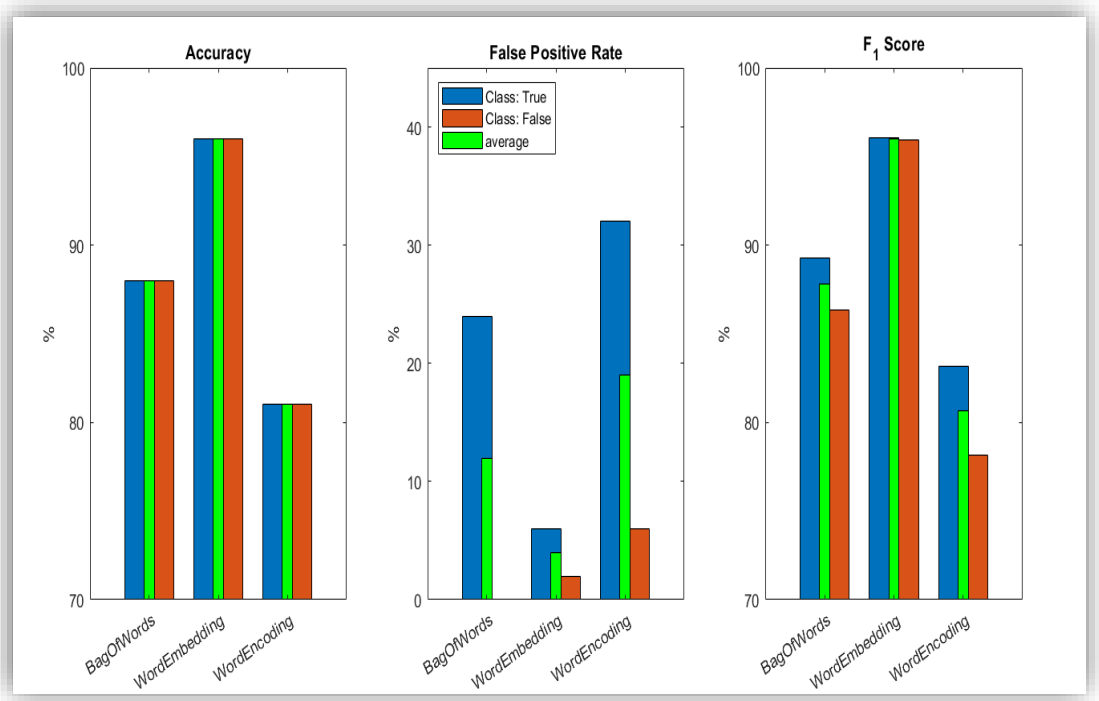
Actual \ Predicted	True	False
True	50	0
False	12	38

Όπως θα εξηγηθεί στη συνέχεια, στην αρχική ιδέα ο χρήστης μπορούσε να επεμβαίνει και να ελέγχει την κατηγοριοποίηση ανάμεσα στις διαφορετικές εντολές V^+, και μπορούσε να απορρίπτει προτάσεις που εσφαλμένα κρίθηκαν ως εντολές από το μοντέλο SVM. Όμως σε αυτή την περίπτωση, αν μια πραγματική εντολή απορριφθεί από το μοντέλο SVM, αυτή χάνεται και ο χρήστης δεν έχει δυνατότητα να την ανασύρει. Για το λόγο αυτό δόθηκε περισσότερη έμφαση στην ορθή αναγνώριση πραγματικών εντολών παρά στην ορθή απόρριψη μη-εντολών. Για αυτό, από τα παρακάτω διαγράμματα τα πιο σημαντικά συμπεράσματα προκύπτουν, πέρα από τον δείκτη ακριβείας (accuracy), από το δείκτη FPR για την κλάση False. Ο δείκτης FPR για την κλάση True αντιστοιχεί σε σφάλμα $1^{ου}$ είδους και ο δείκτης FPR για την κλάση False αντιστοιχεί σε σφάλμα $2^{ου}$ είδους.

Στο Σχήμα 3-3 φαίνονται οι στατιστικοί δείκτες αξιολόγησης των μοντέλων και για τις 2 κλάσεις, καθώς και οι μέσες τιμές (average), που προκύπτουν από τους ανωτέρω πίνακες σύγχυσης. Αρχικά, παρατηρεί κανείς ότι το μοντέλο word embedding έχει στα δεδομένα επαλήθευσης ακρίβεια 96%, και ότι σε όλα τα διαγράμματα η μέση τιμή (των 2 κλάσεων) των δεικτών αυτού του μοντέλου έχει συγκριτικά τις καλύτερες τιμές. Σημειώνεται ότι το τελικό μοντέλο που επιλέχθηκε, χαρακτηρίζεται από τις υπερπαραμέτρους που αναγράφει ο Πίνακας 3-5.

Πίνακας 3-5. Παράμετροι μοντέλου SVM

Kernel Function	Gaussian
Standardize	False
Solver	Iterative Single Data Algorithm (ISDA)
Box Constraint	130.75
Kernel Scale	Auto



Σχήμα 3-3. Σύγκριση Μοντέλων SVM

3.2.2 Κατηγοριοποίηση σε Εντολές V^+

Μετά τη δυαδική κατηγοριοποίηση καταλήγουμε με ένα σύνολο προτάσεων που θεωρήθηκε ότι αναπαριστούν εντολές προς εκτέλεση από το ρομποτικό βραχίονα. Όσον αφορά τις προτάσεις που πράγματι αντιστοιχούν σε εντολές V^+, αρχικά, έγινε προσπάθεια κατασκευής μοντέλων ECOC για κατηγοριοποίηση ανάμεσα σε πολλές κλάσεις. Δυστυχώς τα μοντέλα δεν ήταν ιδιαίτερα αξιόπιστα, για αυτό η ιδέα εγκαταλείφθηκε, και ακολουθήθηκε διαφορετική προσέγγιση.

Λαμβάνοντας υπόψιν ότι στην περίπτωση της παρούσας εργασίας οι στόχοι είναι πολύ συγκεκριμένοι, η θεματολογία και το λεξιλόγιο που μας ενδιαφέρει είναι αρκετά περιορισμένο. Δεν είναι σκοπός μας να φτιάξουμε κάποιο Chatbot, ούτε ένα μοντέλο που να καταλαβαίνει εν γένει την Αγγλική γλώσσα και να γενικεύεται η χρήση του για πολλές θεματολογίες. Ακριβώς επειδή το απαραίτητο λεξιλόγιο είναι εγγενώς περιορισμένο, είναι εύκολο να κατασκευαστούν λίστες λέξεων-κλειδιά που σχετίζονται άμεσα με συγκεκριμένες εντολές, και είναι σημαντικό οι λέξεις που θα χρησιμεύσουν ως λεξιλόγιο να εμφανίζονται σε μια και μόνο μία κλάση. Για παράδειγμα, η λέξη location δεν αποτελεί καλή επιλογή γιατί μπορεί να εμφανίζεται τόσο σε εντολές MOVE όσο και σε εντολές τύπου APPRO ή DEPART. Ουσιαστικά, για κάθε εντολή χρησιμοποιήσαμε όσα περισσότερα συνώνυμα λέξεων και φράσεων γίνεται προκειμένου να δίνεται ευελιξία στο λεξιλόγιο του χρήστη. Ο Πίνακας 3-6 δείχνει το σύνολο των λέξεων και φράσεων που αντιστοιχούν μοναδικά σε κάθε εντολή V^+.

Πίνακας 3-6. Λεξιλόγιο Εντολών

MOVE	APPRO	DEPART	DRIVE	READY	OPENI	CLOSEI	Top Level
move	approach	depart	rotate	reinitialize	open	close	robot
translate	come (closer)	pull (out)	joint	reset	expand	lock	
shift	near	step (away)	turn	revert	free	secure	
stir	draw	get (away)	revolve	return	activate	fasten	
relocate	contact	leave	pivot	restore	unlock	restrict	
proceed	approximate	deviate	degrees	reconfigure	drop	seize	
advance	reach	withdraw	spin	retreat	let (go)	snatch	
transfer	meet	escape	rotation	retire	unretract	catch	
transport	-	exit	swing	initial	unrestrict	hold	
		abandon	twist	starting	release	grab	
			roll	primary	unseal	shut	
			gyrate	original	unload	retract	
			around	origin	place	load	
				safe		pick	
				configuration		collect	
				home			
				back			

Η λέξη «go» λειτουργεί ως μπαλαντέρ, και ανάλογα τις λέξεις εκατέρωθέν της βγαίνει το συμπέρασμα για την αντιστοίχιση εντολής. Για παράδειγμα, η φράση «Go to location …» θα αντιστοιχούσε σε εντολή MOVE, ενώ η φράση «Let go of the item» θα αντιστοιχούσε σε εντολή OPENI, και η φράση «Go back to origin» σε εντολή READY. Με τη συγκεκριμένη προσέγγιση οι εντολές πράγματι αντιστοιχίζονται σωστά σχεδόν σε κάθε περίπτωση, και λαμβάνεται υπόψιν η δυαδική ταξινόμηση από το μοντέλο SVM.

3.3 Περιγραφή Λογικής Κώδικα

3.3.1 Εισαγωγή

Η ροή του κώδικα περιγράφηκε επιγραμματικά στην Ενότητα 3.1. Στην παρούσα ενότητα θα παρουσιάσουμε με μεγαλύτερη λεπτομέρεια την ουσία του κώδικα, και τις επιμέρους συναρτήσεις που αναπτύχθηκαν για την εκτέλεση του στόχου. Σημειώνεται πως ένα μεγάλο τμήμα του κώδικα φαίνεται στο Παράρτημα Α. Οι υπορουτίνες/συναρτήσεις που κατασκευάστηκαν στα πλαίσια της παρούσας εργασίας είναι οι εξής:

- *MAIN_Script.m*
- *MAIN_PickNPlace.m*
- *Initialize.m*
- *create_RX90L_fcn.m*
- *PythonSpeech2Text.m*
- *preprocess.m*

- *repair_text.m*
- *Extract_Numeric_Inputs.m*
- Extract_Numeric_Inputs_PickNPlace.m
- *Build_Embedding.m*
- *Classify.m*
- MyLabelPredictions.*m*
- *SearchCommandFeatures.m*
- *Verify_Commands.m*
- *My_IK_setup.m*
- *My_IK_setup*_PickNPlace.*m*
- *RepairNums.m*
- *Solve_InverseKinematics.m*
- *Extract_file_for_V.m*
- Extract_file_for_V_PickAndPlace.m

3.3.2 Περιγραφή Βασικών Τμημάτων Κώδικα

Initialize.m

Η συγκεκριμένη υπορουτίνα εκτελείται μόνο μια φορά, κατά την πρώτη εκτέλεση της διαδικασίας. Αυτό που κάνει είναι να φορτώνει το εκπαιδευμένο μοντέλο κατηγοριοποίησης εντολών (SVM model) και την ενσωμάτωση λέξεων: fastTextWordEmbedding. Επίσης, προσθέτει το δομοστοιχείο λογισμικού python στο μονοπάτι (path) του MATLAB και καλεί τη συνάρτηση *create_RX90L_fcn.m*, η οποία δημιουργεί το ρομπότ με χρήση του Robotic Toolbox. Το Robotic Toolbox μας επιτρέπει να προκατασκευάσουμε το ρομποτικό βραχίονα με χρήση της μεθόδου mDH, η οποία περιγράφηκε στην Ενότητα 2.1.2. Το ρομπότ που προκύπτει είναι μια οντότητα (object) που έχει ως μέθοδο την εγγενή συνάρτηση *writeAsFunction*, η οποία αυτοματοποιεί την όλη διαδικασία με τη δημιουργία μιας συνάρτησης κατασκευής του βραχίονα (την οποία ονομάσαμε *create_RX90L_fcn*). Ο λόγος για τον οποίο η συγκεκριμένη υπορουτίνα εκτελείται μόνο μια φορά κατά την εκκίνηση, είναι γιατί καθώς το συγκεκριμένο word embedding (*English Word Vectors · FastText*, n.d.) περιέχει περίπου ένα εκατομμύριο λέξεις, αργεί κάποια δευτερόλεπτα να φορτώσει, και αυτό θα προκαλούσε αχρείαστη καθυστέρηση στην όλη διαδικασία αν κάθε φορά ο χρήστης έπρεπε να περιμένει. Με την εφαρμοζόμενη λογική, οι οντότητες αυτές φορτώνονται μια φορά και μετά βρίσκονται ως μεταβλητές στο τοπικό Workspace του MATLAB για όταν ο χρήστης θελήσει να ξαναστείλει εντολές προς το βραχίονα.

PythonSpeech2Text.m

Η υπορουτίνα *PythonSpeech2Text.m* είναι υπεύθυνη για την εκτέλεση του κώδικά python (*real_time_SR_fromMic.py*) που κάνει χρήση του Microsoft Speech SDK, το οποίο περιγράφηκε στην Ενότητα 2.3.4. Καθώς η python τρέχει ασύγχρονα σε σχέση με το MATLAB (out-of-process), στο παράθυρο εντολών του MATLAB φαίνεται σε πραγματικό χρόνο το κείμενο που προκύπτει από κάθε φωνητική είσοδο, όπως θα φαινόταν στη γραμμή εντολών του υπολογιστή αν ο κώδικας python έτρεχε από εκεί.

Σκοπός του χρήστη είναι να δώσει εντολές που σχετίζονται με τις 7 εντολές της V^+ που μας ενδιαφέρουν στα πλαίσια της εργασίας, οι οποίες αναφέρθηκαν στην υποενότητα 2.1.5. Όπως αναφέρθηκε, μια εντολή μπορεί να αφορά τον άνοιγμα ή κλείσιμο της αρπάγης, την περιστροφή συγκεκριμένης άρθρωσης κατά συγκεκριμένες μοίρες, την επιστροφή στην

αρχική θέση (Home Configuration), την προσέγγιση/απομάκρυνση ή και μετάβαση σε συγκεκριμένη θέση με συγκεκριμένο προσανατολισμό.

Μέσω των φωνητικών εντολών, ο χρήστης μπορεί σε κάθε εντολή να περιλάβει μέχρι και 7 αριθμούς ανά εντολή. Αν η αντίστοιχη εντολή αντιστοιχεί σε MOVE ή APPRO τότε, οι τρείς πρώτοι αφορούν την επιθυμητή θέση (σε mm) του τελικού στοιχείου δράσης, οι επόμενοι τρείς αφορούν τον προσανατολισμό σε μοίρες (γωνίες Euler, ZYZ), και ο τελευταίος αφορά την επιθυμητή απόσταση (σε mm) από την τελική θέση, αν η αντίστοιχη εντολή είναι APPRO. Αν η αντίστοιχη εντολή είναι DRIVE, τότε ο 1ος αριθμός αφορά την άρθρωση την οποία θέλουμε να περιστρέψουμε, ο 2ος αφορά τις επιθυμητές μοίρες περιστροφής, και ο 3ος (και προαιρετικός) αριθμός αφορά την ταχύτητα περιστροφής ως ποσοστό (%) της μέγιστης.

Ο κώδικας κατασκευάστηκε με τρόπο που παρέχει ευκολία και ευελιξία στο χρήστη. Όταν ο χρήστης αρθρώσει όλες τις εντολές που επιθυμεί, και πει μια φράση όπως «Please do it now», η ηχογράφηση σταματά, και με βάση το κείμενο που προκύπτει από την ομιλία, συνεχίζεται η ροή του προγράμματος για την εκτέλεση των εντολών. Ακόμα, όταν ο χρήστης θέλει να σταματήσει γιατί αντιλαμβάνεται ότι έχει δώσει εσφαλμένες ή ακατανόητες εντολές, έχει την επιλογή να πει τη φράση «OK stop», και ο κώδικάς σταματάει ενώ η ροή του προγράμματος τερματίζεται. Η δομή της υπορουτίνας *PythonSpeech2Text.m* τρέχει επαναληπτικά, και κάθε φορά που ο χρήστης διακόπτει τη ροή, εμφανίζεται μήνυμα στην οθόνη, στο οποίο ο χρήστης μπορεί μέσα από το πληκτρολόγιο να απαντήσει στο αν θέλει να τερματίσει εντελώς τη διαδικασία ή αν επιθυμεί να αρχίσει πάλι από την αρχή.

Εφόσον η όλη ηχογράφηση κυλήσει ομαλά και σωστά κατά τον χρήστη, η ροή συνεχίζεται κανονικά και το κείμενο, υπό μορφή διανύσματος στήλης ανάλογα με τον αριθμό προτάσεων και ηχογραφημένων σημάτων, υπόκειται σε επεξεργασία.

Παραθέτουμε τους βασικούς κανόνες ηχογράφησης για εκτέλεση φωνητικών εντολών:

- Μία τουλάχιστον πρόταση (π.χ. η 1η πρόταση) πρέπει να περιέχει τη λέξη «robot» (π.χ. «Hello robot.»).
- Η τελευταία πρόταση πρέπει να περιέχει τη φράση «do it now.» προκειμένου να τερματιστεί η ηχογράφηση και να συνεχιστεί η ροή του προγράμματος.
- Ο χρήστης μπορεί να αρθρώσει φράσεις όπως «Delete», «Scratch that» και «Erase that» προκειμένου να διαγράψει την τελευταία του πρόταση αν αντιληφθεί ότι είπε κάτι λάθος και θέλει να το επαναλάβει σωστά.

Στη συνέχεια παραθέτουμε τους κανόνες για τη φωνητική διάρθρωση αριθμητικών δεδομένων κατά την ομιλία:

- Οι αριθμοί πρέπει να διαρθρώνονται ρητά. Για παράδειγμα ο αριθμός 542 πρέπει να διατυπωθεί φωνητικά ως εξής: «five hundred and forty two».
- Κατά τη φωνητική διατύπωση πραγματικών αριθμών (π.χ. δεκαδικούς), ο χρήστης πρέπει πάντα να χρησιμοποιεί τη λέξη «point» για τη δήλωση του δεκαδικού μέρους. Για παράδειγμα, ο αριθμός 1257.9 πρέπει να διατυπωθεί φωνητικά ως εξής: «one thousand two hundred and fifty seven point nine».
- Για τη διατύπωση αρνητικών αριθμών, ο χρήστης πρέπει να πει τη λέξη «minus» ή τη λέξη «negative» πριν τον αριθμό που θέλει να ορίσει.
- Προκειμένου οι αριθμοί να διαχωρίζονται και να μη συγχέονται κατά τη φωνητική είσοδο, είναι χρήσιμο να τοποθετείται μια διαχωριστική λέξη ανάμεσα στα αριθμητικά δεδομένα. Είναι επίσης χρήσιμο η λέξη αυτή να είναι ευδιάκριτη και να μη μοιάζει με άλλες λέξεις. Στα πλαίσια της παρούσας εργασίας επιλέχθηκε η λέξη «slash» που

αντιστοιχεί στο σύμβολο «/». Αρχικά εξετάστηκαν οι λέξεις «dot» και «comma» αλλά δεν είναι πολύ χρήσιμες γιατί μπερδεύονται συχνά με άλλες.

- Κατά τον ορισμό θέσεων, οι χωρικές συντεταγμένες πρέπει να ορίζονται σε χιλιοστά (mm), και οι γωνίες σε μοίρες (degrees).

preprocess.m

Η συνάρτηση αυτή είναι υπεύθυνη για τον καθαρισμό και την προεπεξεργασία του εισαγόμενου κειμένου.

Κατά την εκκίνηση της συγκεκριμένης συνάρτησης καλείται εσωτερικά η συνάρτηση *repair_text.m* η οποία εκτελεί πολύ σημαντικές διορθώσεις στο κείμενο, το οποίο είναι μεταβλητή τύπου string. Αρχικά, πρέπει να αναφερθεί ξανά ότι δεν έχουμε έλεγχο των διεργασιών που εκτελούνται από τη Microsoft για τη μετατροπή της φωνής σε γραπτό κείμενο. Αυτό επιφέρει πιθανά σφάλματα στην ηχογράφηση όπως καθυστέρηση, λάθη στην αναγνώριση αριθμών, ή και λάθος αναγνώριση ακουστικά παρόμοιων λέξεων. Μέσα από πολλές δοκιμές και παρατηρήσεις πιθανών σφαλμάτων κατά την ηχογράφηση, η συνάρτηση *repair_text.m* κατασκευάστηκε ώστε να μπορεί να διορθώσει την πλειοψηφία, αν όχι όλα τα πιθανά σφάλματα. Παραδείγματα σφαλμάτων είναι το μπέρδεμα των λέξεων robot και Robert, three και tree, pick και peak ή desired και desert. Ακόμα, ενώ η συγκεκριμένη υπηρεσία της Microsoft είναι ικανή να αναγνωρίσει αριθμούς, είναι πιθανόν να γίνει κάποιο λάθος και, για παράδειγμα αντί για τον αριθμό 5 να το αναγνωρίσει ως τη λέξη five ή ακόμα χειρότερα να μπερδέψει τη λέξη αυτή με τη λέξη fire. Επίσης, είναι πιθανόν κάποιες φορές η πρόταση του ομιλητή να διακόπτεται πριν τη λήξη της, και το υπόλοιπο κομμάτι της πρότασης να μεταβεί στην αμέσως επόμενη γραμμή. Η συνάρτηση *repair_text.m* επιχειρεί, με βάση κάποια συντακτικά κριτήρια και χρήση ομαλών εκφράσεων (regular expressions), να προλάβει τέτοια λάθη και να ενώσει προτάσεις που κοπήκαν πρόωρα. Αυτό είναι μια κίνηση που διευκολύνει μερικώς το χρήστη, αλλά εισάγει κινδύνους εσφαλμένης συνένωσης εντολών, άρα αστοχίας.

Μετά την εκτέλεση της *repair_text.m* συνεχίζεται η εκτέλεση της συνάρτησης `preprocess.m` η οποία, αρχικά κατασκευάζει κατακερματισμένα αρχεία (tokenized documents) με βάση το ήδη διορθωμένο κείμενο. Στη συνέχεια, γίνεται μια διαδικασία που χρησιμοποιεί κάποιους συντακτικούς κανόνες και λέξεις κλειδιά ώστε, ακόμα και αν ο χρήστης δώσει πολλές εντολές μέσα στην ίδια πρόταση, τελικά η πρόταση να διασπαστεί σε μικρές προτάσεις, μία για κάθε εντολή. Έπειτα, γίνεται χρήση κάποιων εγγενών συναρτήσεων του MATLAB Text Analytics Toolbox που επιδρούν σε tokenized documents, οι οποίες είναι οι εξής:

- *correctSpelling*
- *addPartOfSpeechDetails*
- *removeStopWords*
- *normalizeWords*
- *erasePunctuation*
- *lower*
- *tokenDetails*
- *removeEmptyDocuments*

Τονίζεται ότι λόγω απομάκρυνσης των stop-words στο τέλος της προεπεξεργασίας, οι προτάσεις χάνουν τη συντακτική δομή τους και απομένουν ουσιαστικά ως αλληλουχίες λέξεων που συγκρατούν την ουσία κάθε πρότασης. Τελικά, η συνάρτηση *preprocess.m* επιστρέφει 3 μεταβλητές: τη μεταβλητή του διορθωμένου κειμένου (txt: table variable), τη μεταβλητή

documents (tokenized document), και τη μεταβλητή details (token details) των ιδιοτήτων του tokenized document, με πιο σημαντικές τις ιδιότητες Part of Speech (e.g. verb, noun, numeral) και Type (e.g. digits). Συνοπτικά, η συνάρτηση `preprocess.m` επιτυγχάνει:

- Διόρθωση σφαλμάτων αναγνώρισης λέξεων.
- Διόρθωση σφαλμάτων αναγνώρισης αριθμών.
- Συνένωση προτάσεων που αποκόπηκαν πρόωρα.
- Διάσπαση προτάσεων που περιλαμβάνουν πολλές εντολές.

Στο Σχήμα 3-4 φαίνεται ένα στιγμιότυπου ενός κειμένου που παράγεται από την υπηρεσία Microsoft κατά τη διάρκεια ομιλίας του χρήστη, και στο Σχήμα 3-5 φαίνεται η μορφή του κειμένου μετά τη διαδικασία της επεξεργασίας. Παρατηρεί κανείς πως οι προτάσεις που περιέχουν πολλές εντολές διασπώνται σε μικρότερες, ενώ τυχών εντολές που τερματίστηκαν πρόωρα συνενώνονται και πάλι.

```
STARTED
 Session_id: 81fd8fce59c642678e72f8c4e7ed7e79
 Start talking to your microphone:

RECOGNIZED: OK robot approach position 300 slash 300 slash 856.7 slash 45
slash 60 slash 30 at height 100 millimeters.

RECOGNIZED: Move to the desired location and grab the object. Then step away
to distance 25 millimeters and drop the item. After that rotate joint 4 -, 66
degrees and go back to origin.

RECOGNIZED: Do it now.

Terminating...
Closing: ResultReason.RecognizedSpeech
Done
```

Σχήμα 3-4. Αρχική μορφή κειμένου

```
"ok robot"
"approach position 300 300 856.7 45 60 30 height 100"
"move desired location"
"grab object"
"step away distance 25"
"drop item"
"rotate joint 4 minus 66 degree"
"go back origin"
```

Σχήμα 3-5. Μορφή επεξεργασμένου κειμένου

Extract_Numeric_Inputs.m

Μετά το πέρας της *`preprocess.m`*, εκτελείται η συνάρτηση *`Extract_Numeric_Inputs.m`*, η οποία κατασκευάζει έναν αριθμητικό πίνακα που περιέχει όλα τα αριθμητικά δεδομένα που περιλαμβάνει το κείμενο, συμπεριλαμβανομένων των οποιωνδήποτε αρνητικών τιμών. Ο πίνακας που προκύπτει έχει μέγεθος Nx7, όπου N το πλήθος των εντολών που περιέχονται στο κείμενο. Από τη στιγμή που εξάγονται οι αριθμοί από το κείμενο, όπως θα εξηγηθεί στη συνέχεια, δε χρειάζεται πια το κείμενο να περιλαμβάνει αριθμητικά ψηφία.

Build_Embedding.m

Με βάση το εναπομένον κείμενο κατασκευάζεται μια ενσωμάτωση λέξεων (word embedding) με χρήση της συνάρτησης *Build_Embedding.m*. Η μετατροπή λέξεων σε διανύσματα γίνεται με χρήση του προ-εκπαιδευμένου μοντέλου *word2vec* (Mikolov et al., 2013) που προσφέρεται από το MATLAB Text Analytics Toolbox. Με χρήση του συγκεκριμένου word embedding, κάθε λέξη αντιστοιχίζεται σε ένα διάνυσμα γραμμή με μήκος 300 στοιχεία. Ο κάθε αριθμός ως λέξη αντιστοιχίζεται και αυτός σε κάποιο διάνυσμα μεγέθους 1x300, πράγμα που ίσως επηρέαζε την ταξινόμηση (classification). Για αυτό αφαιρέθηκαν προηγουμένως οι αριθμοί από τις προτάσεις, γιατί η παρουσία τους στο κείμενο δε θα είχε κάποια χρήσιμη σημασία.

Για κάθε προεπεξεργασμένη πρόταση κατασκευάζεται ένα μέσο word embedding, παίρνοντας το μέσο όρο των στοιχείων των λέξεων που την αποτελούν. Όπως αναφέρθηκε, αυτή τη τεχνική εξαγωγής διανυσμάτων που αφορούν προτάσεις είναι αρκετά απλοϊκή, αλλά επαρκεί για την επίτευξη των σκοπών της παρούσας εργασίας.

Classify.m

Η συνάρτηση αυτή δέχεται τον αριθμητικό πίνακα ενσωμάτωσης λέξεων (word embedding) που παρήγαγε η *Build_Embedding.m*. Οι γραμμές του πίνακα, όπου καθεμιά αντιστοιχεί σε μια πρόταση, περνάνε από τα 4 εκπαιδευμένα μοντέλα SVM που πιστοποιήθηκαν μέσα από k-fold cross-validation (υποενότητα 3.2.1), και τελικά λαμβάνεται ως πρόβλεψη αυτή που κερδίζει σε πλειοψηφία. Από τη δυαδική κατηγοριοποίηση θα εντοπιστούν οι δείκτες (indices) των προτάσεων που αναγνωρίζονται ως εντολές για κίνηση του βραχίονα.

MyLabelPredictions.m

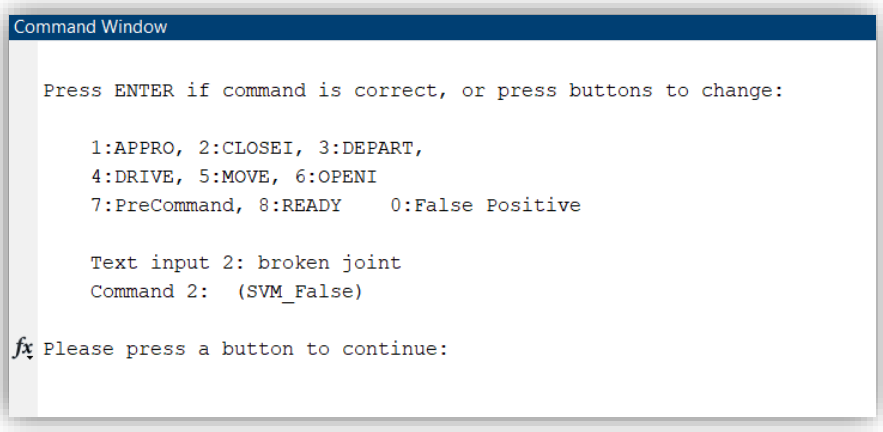
Οι αναγνωρισμένες εντολές θα περάσουν από τη συνάρτηση *MyLabelPredictions.m* η οποία, με βάση την παρουσία των λέξεων που παρουσιάζει ο Πίνακας 3-6, κατηγοριοποιεί τις προτάσεις ανάμεσα στις προεπιλεγμένες εντολές V^+. Αν δε βρεθεί καμία λέξη που περιλαμβάνει ο Πίνακας 3-6 τότε η πρόβλεψη θεωρείται απροσδιόριστη (undefined).

SearchCommandFeatures.m

Εν συντομία, η συνάρτηση *SearchCommandFeatures.m* εξάγει συμπεράσματα όπως:

- Αν υπάρχει εντολή Top Level όπως η φράση: «OK robot».
- Αν υπάρχει ανάγκη επίλυσης της αντίστροφης κινηματικής στα πλαίσια κάποιας εντολής (π.χ. MOVE ή APPRO), και καταγράφει τους δείκτες (indices) των προτάσεων όπου απαιτείται επίλυση.

Αφού γίνει η κατηγοριοποίηση των εντολών, καλείται εσωτερικά η συνάρτηση *Verify_Commands* με σκοπό να ελεγχθεί και να διορθωθούν τυχών λάθη που έγιναν κατά την κατηγοριοποίηση. Η συνάρτηση *Verify_Commands* εμφανίζει στο χρήστη με τη σειρά τις εντολές που δόθηκαν σε συνδυασμό με τη γνωστή εντολή με την οποία αντιστοιχήθηκε, και αν ο χρήστης διαπιστώσει λάθη, έχει τη δυνατότητα με χρήση του πληκτρολογίου να προβεί σε διορθώσεις. Αν η πρόταση έχει κατηγοριοποιηθεί ως μη-εντολή από το δυαδικό μοντέλο SVM, τότε η εντολή θα εμφανίζεται ως κενή. Στο Σχήμα 3-6 φαίνεται ένα στιγμιότυπο από τη γραμμή εντολών του MATLAB όπου ο χρήστης μπορεί σε πραγματικό χρόνο να προβεί σε διορθώσεις κατηγοριοποίησης σε κάθε πρόταση είτε θεωρήθηκε ότι αποτελεί εντολή είτε όχι. Στη συγκεκριμένη περίπτωση, η αρχική πρόταση (I have a broken joint) ορθά κατηγοριοποιήθηκε ως μη-εντολή (SVM_False), οπότε ο χρήστης σε αυτό το σημείο θα πρέπει απλώς να πατήσει το πλήκτρο ENTER για να συνεχίσει στην αξιολόγηση της επόμενης εντολής.



Σχήμα 3-6. Στιγμιότυπο Διόρθωσης Κατηγοριοποίησης

My_IK_setup.m

Η συγκεκριμένη συνάρτηση αρχικά καλεί τη συνάρτηση *RepairNums.m* η οποία παρέχει επιπλέον ευελιξία στο χρήστη μέσω της συμμόρφωσης που εκτελεί στα αριθμητικά δεδομένα, ώστε να είναι συμβατά με τα μεγέθη που δέχεται η αντίστροφη κινηματική. Αρχικά, αν η αντιστοιχισμένη εντολή είναι APPRO, η συνάρτηση τοποθετεί τον τελευταίο αριθμό της υπό μελέτη γραμμής του πίνακα στην 7η στήλη. Επίσης, αν η αντιστοιχισμένη εντολή είναι MOVE ή APPRO, η συνάρτηση *RepairNums.m* φροντίζει η θέση να αφορά τους πρώτους 6 αριθμούς της υπό μελέτη γραμμής του πίνακα με τον εξής τρόπο: Αν ο χρήστης εισάγει λιγότερους από 6 αριθμούς που αφορούν την επιθυμητή θέση και προσανατολισμό, τότε γίνεται συμπλήρωση των υπόλοιπων αριθμών με την τιμή 0 (zero padding). Έτσι, αν για παράδειγμα ο χρήστης δώσει την εντολή: «Approach location four hundred and sixteen point two slash zero slash three hundred eighteen point two at distance one hundred mm.» τελικά ο πίνακας στη συγκεκριμένη γραμμή θα έχει τα στοιχεία: [416.8, 0, 318.2, 0, 0, 0, 100]. Επίσης, τα πρώτα τρία στοιχεία των γραμμών που αφορούν εντολές MOVE ή APPRO διαιρούνται με τον αριθμό 1000 ώστε τελικά οι διαστάσεις τους να είναι σε μέτρα (m). Τέλος, περιορίζονται οι αριθμοί της 7ης στήλη από μια μέγιστη αποδεκτή τιμή 200mm.

Μετά το πέρας της συνάρτησης *RepairNums.m*, η συνάρτηση *My_IK_setup.m* προχωρά σε επίλυση της αντίστροφης κινηματικής, όπου χρειάζεται, αποθηκεύοντας το δείκτη (row index) των γραμμών στους οποίους χρειάστηκε, καλώντας τη συνάρτηση *Solve_InverseKinematics.m*. Μετά την επίλυση της αντίστροφης κινηματικής, επιστρέφεται και μια μεταβλητή που δείχνει αν η επίλυση ήταν επιτυχής (converged solution) ή αν δεν μπόρεσε να συγκλίνει σε βέλτιστη λύση και απλώς μας επέστρεψε την καλύτερη υποβέλτιστη. Στη δεύτερη περίπτωση επιστρέφεται και γραπτό μήνυμα στην οθόνη ότι απέτυχε να επιλύσει την αντίστροφη κινηματική, και δίνεται επιλογή στο χρήστη αν θέλει να διατηρήσει την υποβέλτιστη αυτή λύση ή αν θέλει να την αλλάξει μέσω του πληκτρολογίου, ακόμα και να τερματίσει τελείως τη ροή του προγράμματος.

Solve_InverseKinematics.m

Η συνάρτηση αυτή αποτελεί τον κορμό επίλυσης της αντίστροφης κινηματικής του βραχίονα Staubli RX90L και γίνεται χρήση του *generalizedInverseKinematics* του πακέτου MATLAB Robotic Toolbox.

Η αντίστροφη κινηματική λύνεται με περιορισμούς θέσης, προσανατολισμού και γωνιών αρθρώσεων. Πιο συγκεκριμένα, γίνεται η απαίτηση όλες οι γωνίες των αρθρώσεων να μη ξεπεράσουν τα όρια που περιγράφει ο Πίνακας 2-2, και ο περιορισμός θέσης αφορά την επιθυμητή θέση στην οποία θέλουμε να βρεθεί το τελικό στοιχείο δράσης του βραχίονα.

Ο προσανατολισμός ορίζεται σύμφωνα με τις γωνίες yaw, pitch, roll με διαδοχική περιστροφή ZYZ (V⁺ User's Guide, 1997). Δηλαδή, αρχικά τοποθετείται ένα τοπικό σύστημα συντεταγμένων (ΣΣ) πάνω στο τελικό στοιχείο δράσης, παράλληλο με το σταθερό $ΣΣ_0$, και μετά εφαρμόζεται το επιθυμητό yaw (στροφή γύρω από τον τοπικό άξονα-z), μετά εφαρμόζεται το pitch (στροφή γύρω από τον τοπικό άξονα-y, μετά την περιστροφή yaw) και στο τέλος εφαρμόζεται το επιθυμητό roll (στροφή γύρω από τον τοπικό άξονα-z, μετά τις διαδοχικές περιστροφές yaw και pitch). Στο MATLAB, ο περιορισμός του προσανατολισμού δέχεται quaternions, προφανώς γιατί τα quaternions γενικά είναι πιο συμπαγής αναπαράσταση προσανατολισμού, είναι πιο εύρωστα κοντά σε ιδιομορφίες (singularities), και καθορίζουν μοναδικά τον επιθυμητό προσανατολισμό. Αντίθετα, διαφορετικές γωνίες Euler μπορεί να αναπαριστούν τον ίδιο προσανατολισμό και αυτό δεν είναι ιδιαίτερα βολικό. Έτσι, με αυτά ως δεδομένα επιλύεται η αντίστροφη κινηματική του βραχίονα, ώστε η τελική εντολή να αφορά τον καθορισμό των γωνιών των αρθρώσεων.

Στην επίλυση της αντίστροφης κινηματικής το MATLAB προσφέρει 2 αλγορίθμους: τον προεπιλεγμένο αλγόριθμο Broyden-Fletcher-Goldfarb-Shanno (BFGS) και τον Levenberg-Marquardt (LM). Και οι δύο μέθοδοι είναι Gradient-based επαναληπτικοί αλγόριθμοι ελαχιστοποίησης μιας συνάρτησης κόστους. Οι αλγόριθμοι επιτρέπουν την επιλογή AllowRandomRestarts που χρησιμοποιείται όταν η λύση παραβιάζει περιορισμούς ή όταν η αρχική εκτίμηση είναι πολύ κακή. Επιλέγουμε τον αλγόριθμο BFGS, που είναι quasi-Newton μέθοδος, γιατί είναι πολύ πιο εύρωστος στην εύρεση της βέλτιστης λύσης και είναι πιο αποτελεσματικός σε συνθήκες κοντά στα όρια των αρθρώσεων, ακόμα και για κακή αρχική εκτίμηση, σύμφωνα με το σχετικό εγχειρίδιο του MATLAB (*Inverse Kinematics Algorithms - MATLAB & Simulink*, 2023).

Εν γένει, αν η αντίστροφη κινηματική αποτύχει να δώσει βέλτιστη λύση, ο χρήστης πρέπει να υποψιαστεί ότι οι συγκεκριμένες συντεταγμένες δεν αποτελούν ένα καλώς ορισμένο σημείο ή ότι ίσως περιλαμβάνουν μη-εφικτό (unfeasible) συνδυασμό θέσης και προσανατολισμού.

Extract_file_for_V.m

Η συνάρτηση αυτή παράγει ως έξοδο ένα αρχείο (.txt ή .pg ή ό,τι μορφής επιθυμούμε) το οποίο είναι γραμμένο σε γλώσσα V⁺. Η συνάρτηση αυτή περιλαμβάνει εσωτερικά ορισμένες συναρτήσεις οι οποίες κατά περίπτωση γράφουν το ζητούμενο κώδικα σε γλώσσα συμβατή με τον ρομποτικό βραχίονα (δηλαδή γλώσσα V⁺) ανάλογα τις εντολές και τα αριθμητικά δεδομένα που έδωσε ο χρήστης φωνητικά. Αρχικά, γράφονται κάποιες τυπικές εντολές που σχετίζονται με την αρχικοποίηση και την προετοιμασία του βραχίονα, και στη συνέχεια γράφεται το κομμάτι του κώδικα που σχετίζεται με τις φωνητικές εντολές του χρήστη.

Αξίζει να αναφερθεί ότι όταν η εντολή MOVE εμφανίζεται αμέσως μετά την εντολή APPRO, κάνουμε χρήση της εντολής MOVES η οποία εξασφαλίζει ότι το τελικό εργαλείο δράσης του βραχίονα κινείται σε ευθεία γραμμή κατά την προσέγγιση στο εκάστοτε τεμάχιο. Αντίστοιχα, και με την ίδια λογική, χρησιμοποιείται η εντολή DEPARTS όταν έπεται αμέσως μετά από εντολή CLOSEI. Το τελικό αρχείο μπορεί να εκτελεστεί από ένα πρόγραμμα εξομοιωτή τερματικού ανοιχτού κώδικα όπως το δωρεάν λογισμικό Tera Term.

3.3.3 Εισαγωγικό Παράδειγμα

Ο καλύτερος τρόπος να γίνει κατανοητή η διαδικασία είναι ένα παράδειγμα που περιλαμβάνει όλες τις εντολές που μας ενδιαφέρουν. Οι φωνητικές εντολές μετατρέπονται σε γραπτό κείμενο μέσω του MS Speech SDK, όπως έχει περιγραφεί σε προηγούμενη ενότητα, και προκύπτει το κείμενο που δείχνει ο Πίνακας 3-7. Σε αυτόν τον Πίνακα, τα λάθη κατά τη μετατροπή λόγου σε γραπτό κείμενο έχουν σημειωθεί με κόκκινο χρώμα. Μόλις το παραπάνω κείμενο περάσει από τη συνάρτηση *preprocess.m*, το κείμενο που θα προκύψει δείχνει ο Πίνακας 3-8. Στη συνέχεια, το κείμενο θα περάσει από τη συνάρτηση *Extract_Numeric_Inputs.m* για να εξαχθούν τα αριθμητικά δεδομένα.

Πίνακας 3-7. Αρχικό Κείμενο που προκύπτει από Microsoft Speech-SDK

User Input
OK Robert.
How are you feeling today?
I think the weather today is pretty pleasant.
Approximate location 141.42 slash 85 slash 777.82 slash 90 slash 90 slash 45 at distance 100 mm.
Relocate to dessert location and catch the item.
Abandon target location to distance 50 mm and release the object.
Rotate John three slash minus 46 degrees and go back to home configuration.

Πίνακας 3-8. Διορθωμένο Κείμενο

User Input
ok robot
feel today
think weather today pretty pleasant
approximate location 141.42 85 777.82 90 90 45 distance 100
relocate desired location
catch item
abandon target location distance 50
release object
rotate joint 3 minus 46 degree
go back home configuration

Στη συνέχεια, και με βάση το απομένον κείμενο, μέσω της συνάρτησης *Build_Embedding.m* θα κατασκευαστεί ο πίνακας X, διαστάσεων (10x300) όπου για κάθε πρόταση υπολογίστηκε το μέσο (mean) διάνυσμα (1x300), δηλαδή ο μέσος όρος των διανυσμάτων που αντιστοιχούν στις λέξεις κάθε πρότασης. Ουσιαστικά, κάθε πρόταση ανάγεται στη λέξη της οποίας το διάνυσμα είναι πλησιέστερο στη μέση λέξη της πρότασης. Ο πίνακας X λοιπόν θα περάσει μέσα από το δυαδικό μοντέλο κατηγοριοποίησης (συνάρτηση *`Classify.m`*) το οποίο θα διαπιστώσει ότι οι εντολές στη $2^{η}$ και $3^{η}$ πρόταση είναι άκυρες και δε σχετίζονται με την κίνηση του ρομποτικού βραχίονα. Στη συνέχεια, η συνάρτηση `MyLabelPredictions.m` κατηγοριοποιεί τις προτάσεις ανάλογα την παρουσία λέξεων που παρουσιάζει ο Πίνακας 3-6. Ο Πίνακας 3-9 παρουσιάζει τα αποτελέσματα της επεξεργασίας και της κατηγοριοποίησης, και

ο Πίνακας 3-10 περιλαμβάνει τα αριθμητικά δεδομένα των φωνητικών εντολών που αναγράφει ο Πίνακας 3-8 .

Πίνακας 3-9. Εντολές και multi-class Classification

User Input	V+ Commands
ok **robot**	Top Level
approximate location distance	APPRO
relocate desired location	MOVE
catch item	CLOSEI
abandon target location distance	DEPART
release object	OPENI
rotate joint degree	DRIVE
go **back home configuration**	READY

Πίνακας 3-10. Αριθμητικά Δεδομένα

141.42	85	777.82	90	90	45	100
50						
3	-46					

Στη συνέχεια γίνεται επίλυση της αντίστροφης κινηματικής του βραχίονα κατά την οποία εκτελείται αρχικά η συνάρτηση *RepairNums.m* η οποία θα συμμορφώσει τα στοιχεία που αναγράφει ο Πίνακας 3-10. Με αυτά ως είσοδο, η αντίστροφη κινηματική θα δώσει τις επιθυμητές γωνίες: q_1=0, q_2=-135, q_3=180, q_4=90, q_5=90, q_6=0°,που αφορούν τη 2η και την 3η εντολή. Στο τέλος της διαδικασίας, μέσω της συνάρτησης *Extract_file_for_V.m* θα παραχθεί ένα αρχείο που μεταξύ άλλων, θα περιέχει τον κώδικα που φαίνεται στο Σχήμα 3-7. Παρατηρείται ότι ακριβώς μετά την εκτέλεση εντολής APPRO κάνουμε χρήση και της V+ εντολής ALIGN για να ευθυγραμμίσουμε το τελικό στοιχείο δράσης στον πλησιέστερο τοπικό άξονα πριν την άφιξη στο επιθυμητό σημείο. Επίσης, φροντίζουμε το τελικό στοιχείο δράσης να προσεγγίζει και να απομακρύνεται σε ευθεία γραμμή από το επιθυμητό σημείο (χρήση MOVES και DEPARTS).

```
POINT #loc.target=#PPOINT(0,-135,180,90,90,0);

APPRO loc.target, 100;
ALIGN;
BREAK;
MOVES loc.target;
BREAK;

CLOSEI;

HERE loc.current;
DEPARTS 50;

OPENI;

DRIVE 3, -46, 50;
BREAK;

DO READY;
```

Σχήμα 3-7. Κώδικας V+

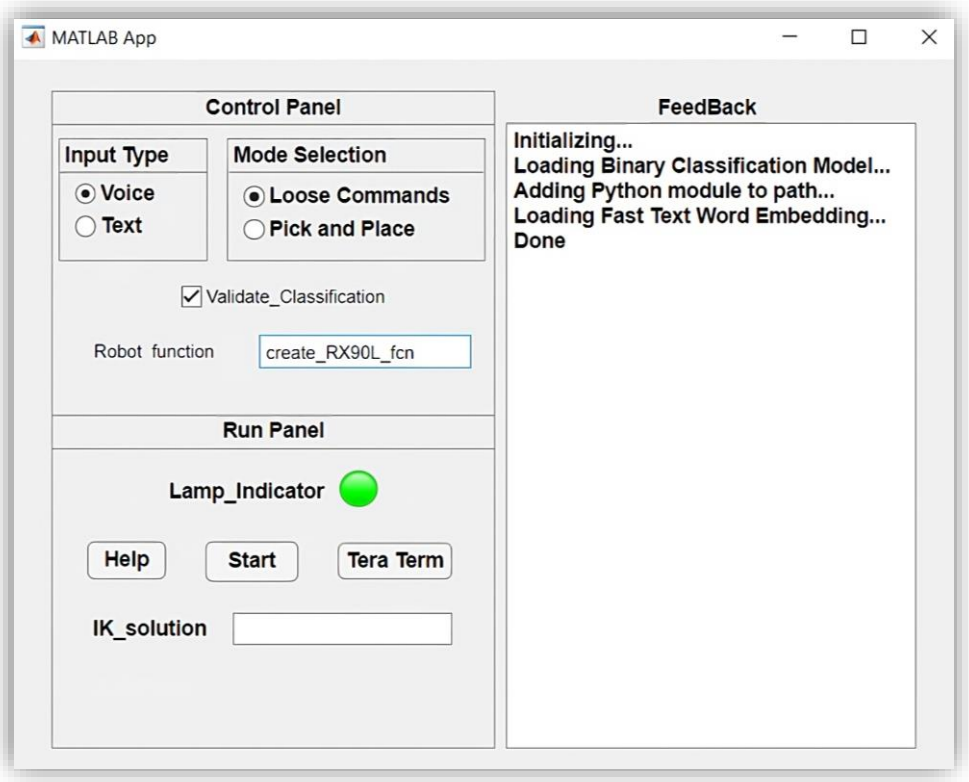
3.4 Ανάπτυξη Εφαρμογής με Γραφική Διεπαφή

3.4.1 Γενική Περιγραφή – Λειτουργία Ελεύθερης Αλληλουχίας Εντολών

Όλη η δομή του κώδικα που περιγράφηκε στην Ενότητα 3.3.2 χρησιμοποιήθηκε για τη δημιουργία ενός MATLAB Application με γραφική διεπαφή για το χρήστη (Graphical User Interface: GUI) η οποία περιλαμβάνει κάποιες επεκτάσεις και βελτιώσεις όπως θα εξηγηθεί στη συνέχεια. Οι βελτιώσεις κρίθηκαν απαραίτητες προκειμένου να ενισχύσουν τη λειτουργικότητα και την ευελιξία της εφαρμογής.

Οι συναρτήσεις που εξηγήθηκαν στην Ενότητα 3.3.2 μετατράπηκαν σε (public) συναρτήσεις, και πολλά στοιχεία που είναι απαραίτητο να είναι ορατά σε όλες τις συναρτήσεις, μετατράπηκαν σε ιδιότητες (properties) της εφαρμογής. Στο Σχήμα 3-8 φαίνεται η δομή της διεπαφής κατά την εκκίνηση, όπως σχεδιάστηκε με χρήση του MATLAB App Designer.

Στη μορφή του κώδικα που περιγράφηκε στην Ενότητα 3.3.2 υπήρχαν κάποιες αδυναμίες που δυσχεραίναν την αποτελεσματικότητα και τη δυνατότητα επέμβασης του χρήστη. Αρχικά, στην πρώιμη μορφή δεν υπήρχε δυνατότητα άμεσης διόρθωσης αριθμητικών δεδομένων, ενώ η διεπαφή επιτρέπει ακριβώς αυτό, μέσα από έναν επεξεργάσιμο πίνακα που παράγεται κατά την εκτέλεση. Ο πίνακας αυτός δίνει επίσης τη δυνατότητα διόρθωσης κατηγοριοποίησης εντολών εύκολα και γρήγορα. Ακόμα, η διεπαφή επιτρέπει την αξιολόγηση της κατηγοριοποίησης σε όλες τις προτάσεις που αρχικά εισήγαγε ο χρήστης, και με αυτό τον τρόπο ελέγχεται περισσότερο το σύνολο των εντολών που θα φτάσουν στον πραγματικό βραχίονα.



Σχήμα 3-8. Γραφική Διεπαφή Χρήστη

Αρχικά δίνεται στο χρήστη η επιλογή ανάμεσα σε φωνητική και γραπτή είσοδο. Η επιλογή αυτή είναι συμπληρωματική ως προς τον πρωταρχικό σκοπό μας, και διευκολύνει σε περίπτωση που ο χρήστης δεν έχει πρόσβαση στο διαδίκτυο ή στην υπηρεσία της Microsoft που χρησιμοποιούμε, ή σε περίπτωση που απαιτείται χρήση της εφαρμογής σε χώρο με έντονη παρουσία θορύβου.

Δίνεται επιλογή λειτουργίας ανάμεσα σε προγραμματισμό ελεύθερης διάρθρωσης εντολών (Loose Commands) και σε προγραμματισμό διεργασίας Pick and Place. Επίσης, δίνεται η επιλογή Validate_Classification που χρησιμεύει μόνο κατά τη λειτουργία Loose Commands, και επιτρέπει τον έλεγχο και τη διόρθωση της κατηγοριοποίησης των εντολών μία προς μία. Τέλος, για λόγους ευελιξίας ,δίνεται η επιλογή στο χρήστη να εισάγει ρομποτικό βραχίονα της επιλογής του, δηλαδή να προκατασκευάσει ένα βραχίονα μέσω του Robotics System Toolbox, και να εισάγει το όνομα της συνάρτησής του στο στοιχείο Robot_function (προαιρετικό). Για λόγους συμβατότητας, αν ο χρήστης θέλει να εισάγει βραχίονα της επιλογής του, οφείλει να ορίσει το σώμα του τελικού στοιχείου δράσης του με το όνομα «tool», και τη βάση ως «base».

Στη συνέχεια, φαίνεται μια συστοιχία τριών πλήκτρων: Help, Start και Tera Term. Ο χρήστης μπορεί αρχικά να πατήσει το πλήκτρο Help, ώστε να λάβει πληροφορίες και οδηγίες για την εφαρμογή και τις εισόδους που δέχεται. Κατά την εκκίνηση της εφαρμογής, το στοιχείο Lamp_Indicator αρχικά είναι κόκκινο και γίνεται πράσινο μόνο όταν έχουν φορτώσει όλα τα απαραίτητα μοντέλα και δεδομένα, όπως το Fast Text Word Embedding. Εφόσον το λαμπάκι γίνει πράσινο, το πάτημα του πλήκτρου Start ξεκινά τη ροή του προγράμματος. Κάτω από τη συστοιχία πλήκτρων φαίνεται το στοιχείο IK_Solution που μας δείχνει αν η αντίστροφη κινηματική επιλύθηκε σωστά ή προέκυψε κάποιο πρόβλημα. Στα δεξιά της διεπαφής, δηλαδή

στο στοιχείο Feedback δίνονται πληροφορίες στο χρήστη σχετικά με την εξέλιξη της ροής του προγράμματος, κάτι σαν τη γραμμή εντολών.

Μετά το πέρας της ροής προγράμματος εμφανίζονται στην οθόνη, ανάλογα το mode λειτουργίας, διάφοροι πίνακες και σχήματα με τα αποτελέσματα και την οπτικοποίηση των επιθυμητών διατάξεων του βραχίονα. Κάθε φορά που εκτελείται η εφαρμογή, ενδιαφέροντα αποτελέσματα (π.χ. κείμενο, εντολές, θέσεις, γωνίες, κατηγοριοποίηση κ.α.) αποθηκεύονται ως μεταβλητές στο Workspace του MATLAB. Το πιο σημαντικό είναι ότι παράγεται ένα αρχείο με το περιεχόμενο των εντολών του χρήστη εκπεφρασμένα σε κώδικα V^+, ώστε πράγματι οι εντολές να μπορούν να εκτελεστούν. Ο χρήστης μπορεί να πατήσει το πλήκτρο Tera Term ώστε να ανοίξει νέο παράθυρο με τη γραμμή εντολών της Tera Term για να εκτελεστούν τελικά οι εντολές από το ρομποτικό βραχίονα.

3.4.2 Περιγραφή Λειτουργίας Pick and Place

Όπως αναφέρθηκε, η βασική ιδέα της εφαρμογής επεκτάθηκε ώστε να μπορεί να προγραμματιστεί τη βιομηχανική εφαρμογή διεργασία pick and place. Οι περισσότερες συναρτήσεις που ορίστηκαν για τη λειτουργία Loose Commands χρησιμοποιήθηκαν σχεδόν αυτούσιες. Μια κύρια διαφορά έγκειται στην εξαγωγή αριθμητικών δεδομένων, για αυτό και κατασκευάστηκε η συνάρτηση *Extract_Numeric_Inputs_PickNPlace.m* η οποία είναι υπεύθυνη για την εξαγωγή και επιδιόρθωση δεδομένων, αλλά και για την εξαγωγή συμπερασμάτων για το τι ακριβώς ορίζεται σε κάθε πρόταση, με βάση τις λέξεις-κλειδιά που αναγράφει ο Πίνακας 3-11. Και σε αυτή την περίπτωση γίνεται συμπλήρωση με μηδενικά (zero padding) στον ορισμό θέσεων με λιγότερα από 6 στοιχεία. Ακόμα, κατασκευάστηκε η συνάρτηση *My_IK_setup_PickNPlace.m* που συμμορφώνει τις μεταβλητές για την επίλυση της αντίστροφης κινηματικής. Μια ακόμα βασική διαφορά είναι ο τρόπος αυτόματης συγγραφής του κώδικα V^+ που απαιτείται για την επίτευξη της συγκεκριμένης βιομηχανικής εφαρμογής, για αυτό και κατασκευάστηκε η συνάρτηση *Extract_file_for_V_PickAndPlace.m*.

Σε αυτή τη λειτουργία δε χρησιμοποιούνται μοντέλα κατηγοριοποίησης αλλά ο χρήστης πρέπει να ορίσει κάποια αριθμητικά δεδομένα προκειμένου να πραγματοποιηθεί ο σκοπός, και το λεξιλόγιό του πρέπει να είναι κάπως πιο περιορισμένο. Υπενθυμίζουμε ότι θέλουμε να προγραμματίσουμε μια αλληλουχία κινήσεων της μορφής που περιγράφηκε στην Ενότητα 2.2.3 στο Σχήμα 2-15.

Ανεξάρτητα με τη σειρά με την οποία ο χρήστης θα ορίσει 4 επιθυμητές θέσεις και τα δεδομένα (αριθμοί x_i), πρέπει να οριστούν τα ακόλουθα, όπως παρουσιάζει ο Πίνακας 3-11. Οι προεπιλεγμένες λέξεις με τις οποίες η διεπαφή αναγνωρίζει το όνομα της θέσης την οποία ορίζει ο χρήστης φαίνονται με παχιά γραμματοσειρά. Δηλαδή, η πρόταση ορισμού θέσης παραλαβής πρέπει να περιέχει μια από τις λέξεις «pick», «collect» και «load», και αντίστοιχα οι υπόλοιπες 3 πρέπει να περιέχουν τις λέξεις «place» ή «unload», «alpha», «beta». Και σε αυτή τη λειτουργία, ο χρήστης πρέπει αρχικά να δώσει στο βραχίονα μια εντολή Top Level (e.g. Hello robot).

Πίνακας 3-11. Ορισμοί χρήστη για pick & place

Ορισμός	Επεξήγηση
Hello robot.	Εντολή Top Level.
Position/Location **pick/collect/load** equals x_1, x_2, x_3, x_4, x_5, x_6.	Θέση παραλαβής τεμαχίου.

Position/Location **place/unload** equals x_1, x_2, x_3, x_4, x_5, x_6.	Θέση εναπόθεσης τεμαχίου.
Position/Location **alpha** equals x_1, x_2, x_3, x_4, x_5, x_6.	Θέση πριν/μετά τη θέση παραλαβής τεμαχίου.
Position/Location **beta** equals x_1, x_2, x_3, x_4, x_5, x_6.	Θέση πριν/μετά τη θέση εναπόθεσης τεμαχίου.
Height equals x_1.	Απόσταση κατά την προσέγγιση / απομάκρυνση.
Distance x equals x_1.	Απόσταση τεμαχίων κατά τη διεύθυνση x.
Distance y equals x_1.	Απόσταση τεμαχίων κατά τη διεύθυνση y.
Repetitions equal x_1 times x_2.	Αριθμός τεμαχίων κατά τις διευθύνσεις x και y αντίστοιχα.

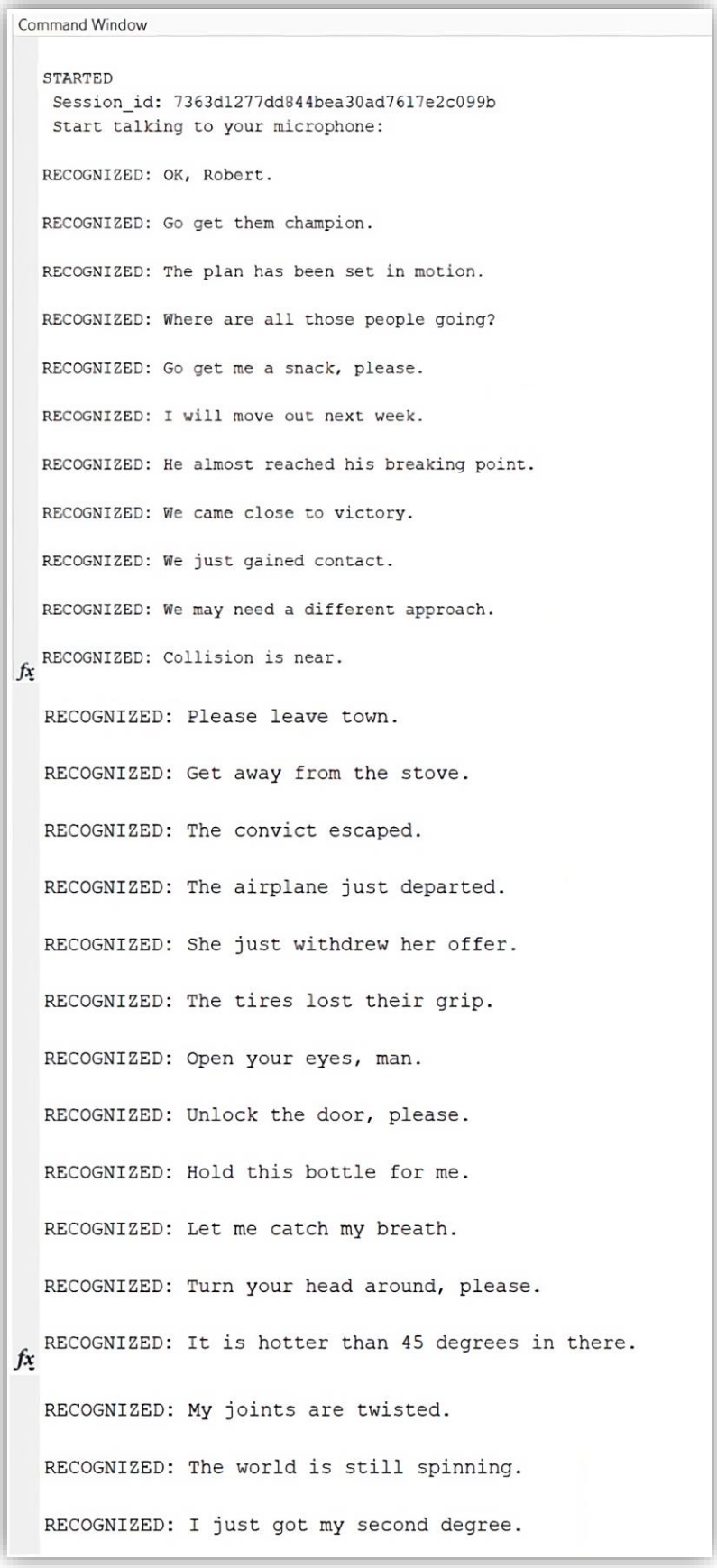
3.5 Παραδείγματα Εφαρμογής

3.5.1 Παραδείγματα Ελεύθερης Λειτουργίας

Παράδειγμα 1

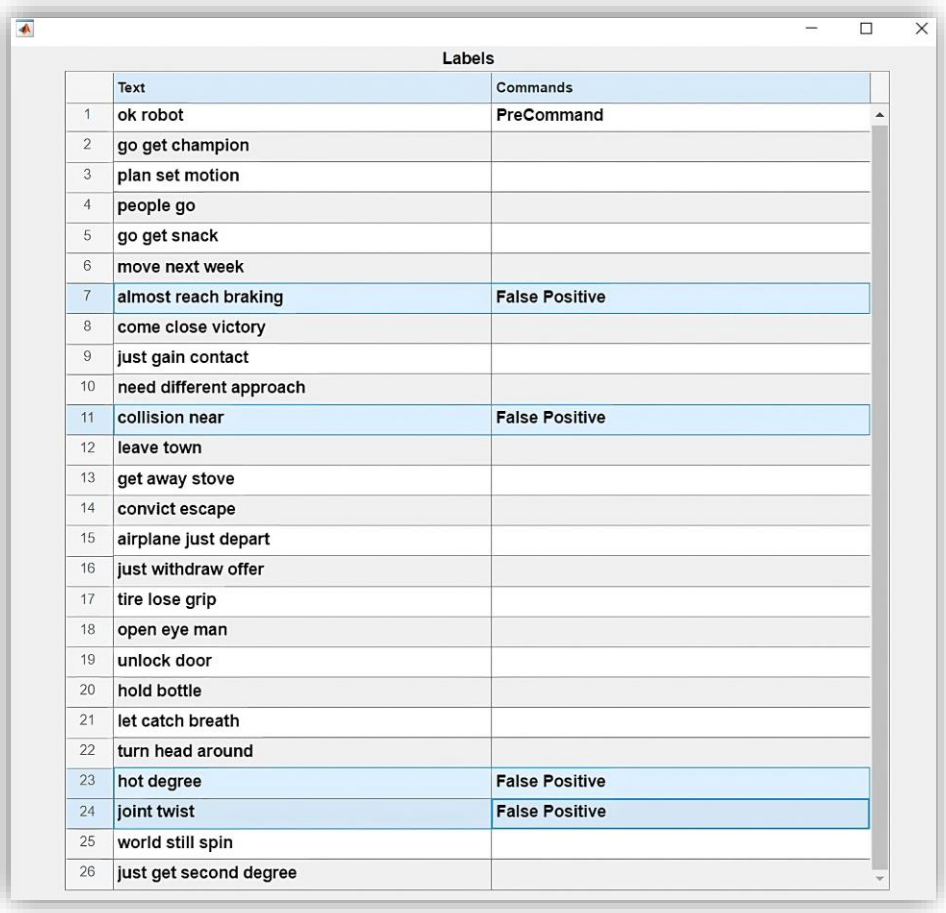
Το παράδειγμα αυτό αξιολογεί την αποτελεσματικότητα της δυαδικής κατηγοριοποίησης αλλά και τη χρησιμότητα της δυνατότητας του χρήστη να επεμβαίνει και να διορθώνει σφάλματα κατηγοριοποίησης. Έστω λοιπόν ότι ο χρήστης επιλέγει φωνητική είσοδο και λειτουργία ελεύθερων εντολών. Μετά το πάτημα του πλήκτρου Start θα εμφανιστεί η γραμμή εντολών του MATLAB όπου θα εμφανίζονται τα λεγόμενα του χρήστη ως γραπτό κείμενο, σε σχεδόν πραγματικό χρόνο. Στο Σχήμα 3-9 φαίνεται η μορφή του κειμένου όπως προκύπτει από την υπηρεσία της Microsoft.

Παρατηρείται ότι οι προτάσεις που φαίνονται στο Σχήμα 3-9 αναπαριστούν άσχετες με τους σκοπούς μας προτάσεις, αλλά περιέχουν λέξεις που ορίστηκαν ως λέξεις-κλειδιά οι οποίες προκαλούν το μοντέλο κατηγοριοποίησης, υπό την έννοια ότι θα μπορούσε το μοντέλο SVM κάποιες προτάσεις εσφαλμένα να τις θεωρήσει ως εντολές. Αμέσως μετά την τελευταία πρόταση που φαίνεται στο Σχήμα 3-9, ο χρήστης πρέπει να αναφέρει τη φράση «Do it now» για να σταματήσει η ηχογράφηση και να συνεχιστεί η εκτέλεση του προγράμματος.



Σχήμα 3-9. Ηχογράφηση Φωνητικών Εντολών σε πραγματικό χρόνο

Στο συγκεκριμένο παράδειγμα έγινε χρήση της επιλογής Validate_Classification για έλεγχο και διόρθωση εσφαλμένων προβλέψεων μία προς μία, και στο Σχήμα 3-10 φαίνεται η τελική κατηγοριοποίηση. Οι προτάσεις που ταξινομούνται ως κενές είναι πράγματι μη-εντολές. Αντίθετα, οι προτάσεις που ταξινομούνται ως False Positive αποτελούν προτάσεις που το δυαδικό μοντέλο SVM θεώρησε εσφαλμένα ως πραγματικές εντολές από τη στιγμή που περιέχουν λέξεις κλειδιά. Για παράδειγμα, η πρόταση 23 κρίθηκε εσφαλμένα ως εντολή επειδή περιέχει τη λέξη-κλειδί «degree». Από το Σχήμα 3-10 φαίνεται πως το μοντέλο λειτουργεί ικανοποιητικά, όπου συνολικά 4 από τις 25 μη-εντολές ταξινομήθηκαν εσφαλμένα ως εντολές προς το ρομποτικό βραχίονα.



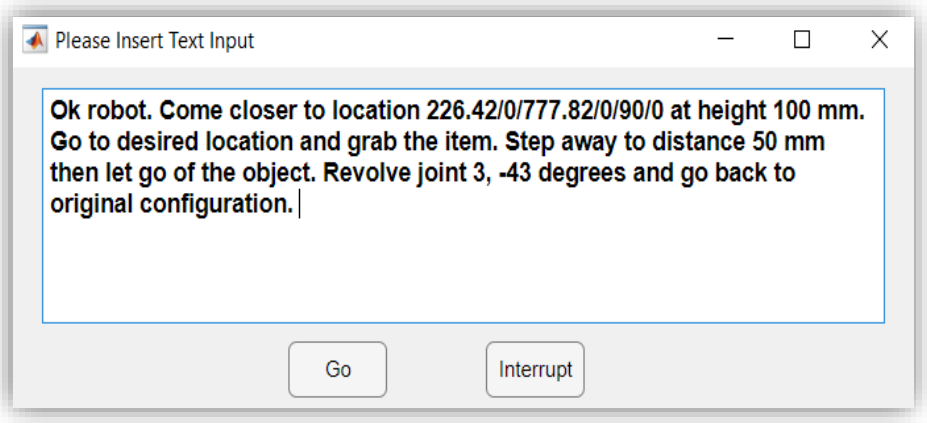
Labels

	Text	Commands
1	ok robot	PreCommand
2	go get champion	
3	plan set motion	
4	people go	
5	go get snack	
6	move next week	
7	almost reach braking	False Positive
8	come close victory	
9	just gain contact	
10	need different approach	
11	collision near	False Positive
12	leave town	
13	get away stove	
14	convict escape	
15	airplane just depart	
16	just withdraw offer	
17	tire lose grip	
18	open eye man	
19	unlock door	
20	hold bottle	
21	let catch breath	
22	turn head around	
23	hot degree	False Positive
24	joint twist	False Positive
25	world still spin	
26	just get second degree	

Σχήμα 3-10. Διορθωμένη Κατηγοριοποίηση

Παράδειγμα 2

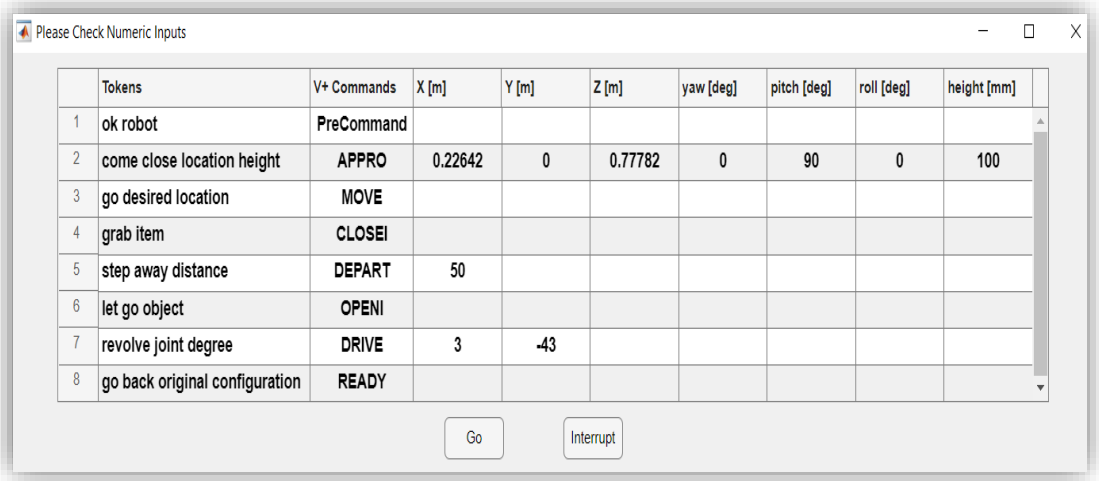
Το παράδειγμα αυτό θα επικεντρωθεί σε πραγματικές εντολές μέσω γραπτού λόγου. Έστω λοιπόν ότι ο χρήστης επιλέγει εισαγωγή κειμένου και λειτουργία ελεύθερων εντολών. Μετά το πάτημα του πλήκτρου Start θα εμφανιστεί στην οθόνη ένα επεξεργάσιμο στοιχείο κειμένου στο οποίο ο χρήστης πληκτρολογεί για παράδειγμα το περιεχόμενο που φαίνεται στο Σχήμα 3-11. Σημειώνεται ότι ο χρήστης μπορεί σε μια πρόταση να συνδυάζει παραπάνω από μια εντολές μέσω συνδετικών λέξεων (π.χ. and), και λόγω της γραπτής εισόδου ο χρήστης θα μπορούσε να χωρίζει τους αριθμούς και με κόμμα. Υπενθυμίζεται ότι ο χρήστης πρέπει οπωσδήποτε να γράφει τους αριθμούς απευθείας ως αριθμητικά ψηφία και όχι ως λέξεις, καθώς στη συγκεκριμένη περίπτωση δε χρησιμοποιείται η υπηρεσία της Microsoft και θα προέκυπτε σφάλμα κατά την εκτέλεση του κώδικα.



Σχήμα 3-11. Εισαγωγή Δεδομένων ως Κείμενο

Στη συνέχεια ακολουθεί δυαδική κατηγοριοποίηση (SVM) ακολουθούμενη από κατηγοριοποίηση που προκύπτει από την παρουσία συγκεκριμένων λέξεων-κλειδιά, όπως περιγράφηκε στην υποενότητα 3.2.2. Μετά την κατηγοριοποίηση και την ανάλυση χαρακτηριστικών των εντολών γίνεται συμμόρφωση των αριθμών (RepairNums) και στη συνέχεια εμφανίζεται ένας επεξεργάσιμος πίνακας όπως φαίνεται στο Σχήμα 3-12. Αυτό δίνει στο χρήστη τη δυνατότητα να επέμβει και στην εισαγωγή των αριθμών, με σκοπό να διορθώσει τυχόν λάθη κατά την υπαγόρευση εντολών, και να προλάβει τυχόν σφάλματα στην κατηγοριοποίηση.

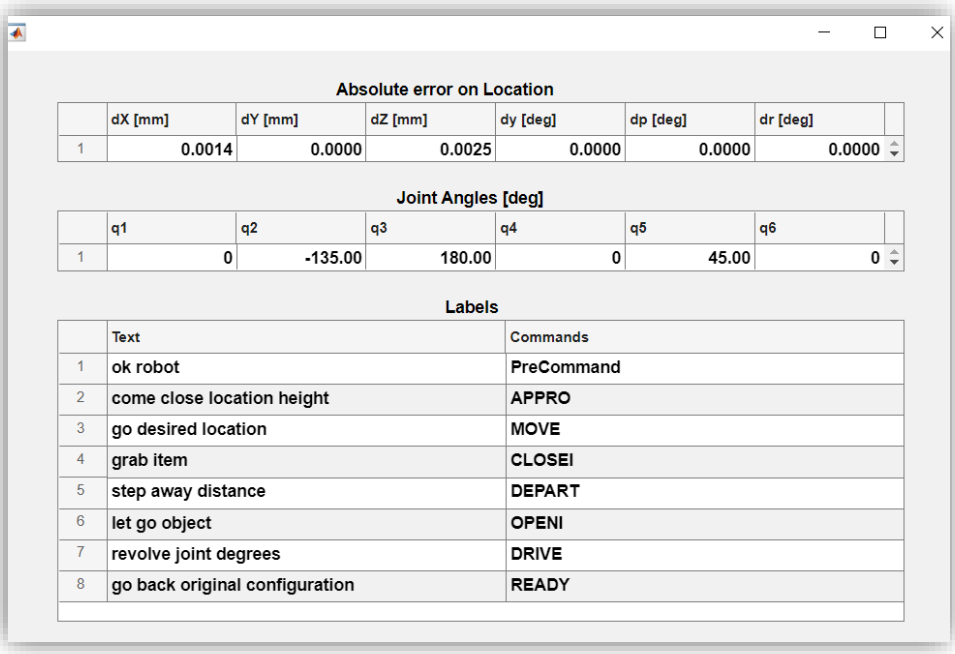
Παρατηρείται λοιπόν ότι ο χρήστης έχει ενεργό έλεγχο στη ροή της διαδικασίας. Συγκεκριμένα, έχει πολλές δικλείδες ασφαλείας κατά τη φωνητική υπαγόρευση και δυνατότητα διόρθωσης σφαλμάτων, δηλαδή μπορεί να διορθώσει τυχόν σφάλματα κατηγοριοποίησης εντολών V+, δηλαδή να αποσύρει προτάσεις που εσφαλμένα έγιναν δεκτές και να ανασύρει προτάσεις που εσφαλμένα απορρίφθηκαν, να διορθώσει τα αριθμητικά δεδομένα, και να μεταβάλλει τις επιθυμητές γωνίες αρθρώσεων του βραχίονα σε περίπτωση που η αντίστροφη κινηματική δε καταφέρει να δώσει βέλτιστα αποτελέσματα.



Σχήμα 3-12. Έλεγχος Αριθμητικών Δεδομένων

Μόλις ο χρήστης τελειώσει την εισαγωγή κειμένου ή τη διόρθωση των δεδομένων, με το πάτημα του πλήκτρου Go συνεχίζεται η ροή προγράμματος. Σε κάθε περίπτωση, ο χρήστης

μπορεί να πατήσει το πλήκτρο Interrupt και να τερματίσει πρόωρα την εκτέλεση. Στο Σχήμα 3-13 φαίνεται ο τελικός πίνακας αποτελεσμάτων για τη συγκεκριμένη είσοδο. Παρατηρούμε ότι τα σφάλματα είναι απειροελάχιστα και ότι η αντίστροφη κινηματική δίνει τα αποτελέσματα που περιμέναμε. Επίσης φαίνονται οι επιμέρους προτάσεις μετά τη διάσπασή τους, και η κατηγοριοποίηση των εντολών, η οποία λειτούργησε αποτελεσματικά. Στο Σχήμα 3-14 φαίνεται σχηματικά η διάταξη του βραχίονα με τις γωνίες που υπολογίστηκαν από την επίλυση της αντίστροφης κινηματικής.



Absolute error on Location

	dX [mm]	dY [mm]	dZ [mm]	dy [deg]	dp [deg]	dr [deg]
1	0.0014	0.0000	0.0025	0.0000	0.0000	0.0000

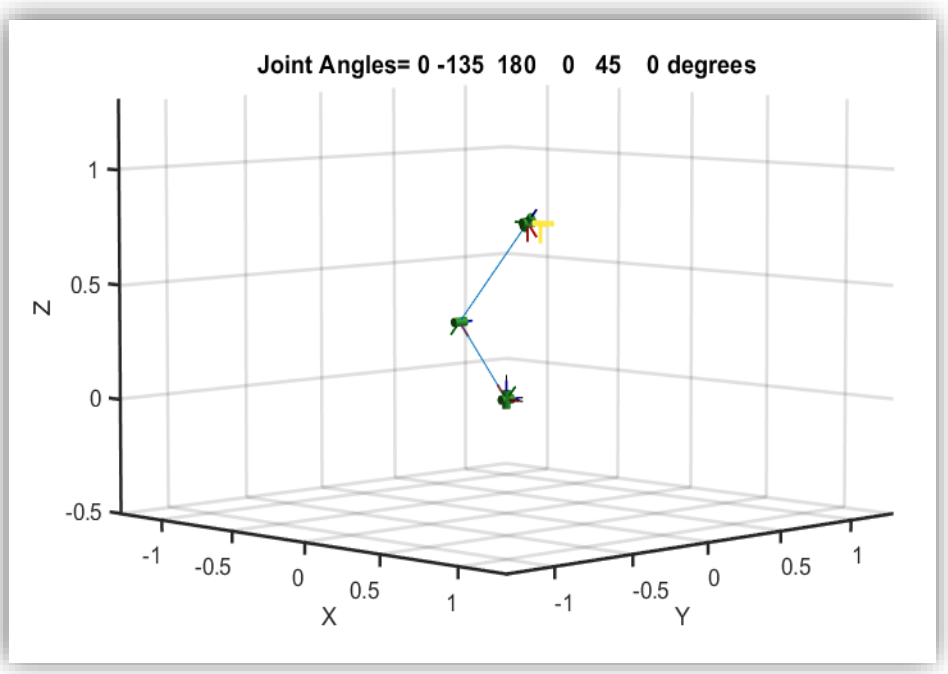
Joint Angles [deg]

	q1	q2	q3	q4	q5	q6
1	0	-135.00	180.00	0	45.00	0

Labels

	Text	Commands
1	ok robot	PreCommand
2	come close location height	APPRO
3	go desired location	MOVE
4	grab item	CLOSEI
5	step away distance	DEPART
6	let go object	OPENI
7	revolve joint degrees	DRIVE
8	go back original configuration	READY

Σχήμα 3-13. Πίνακας Αποτελεσμάτων Παραδείγματος



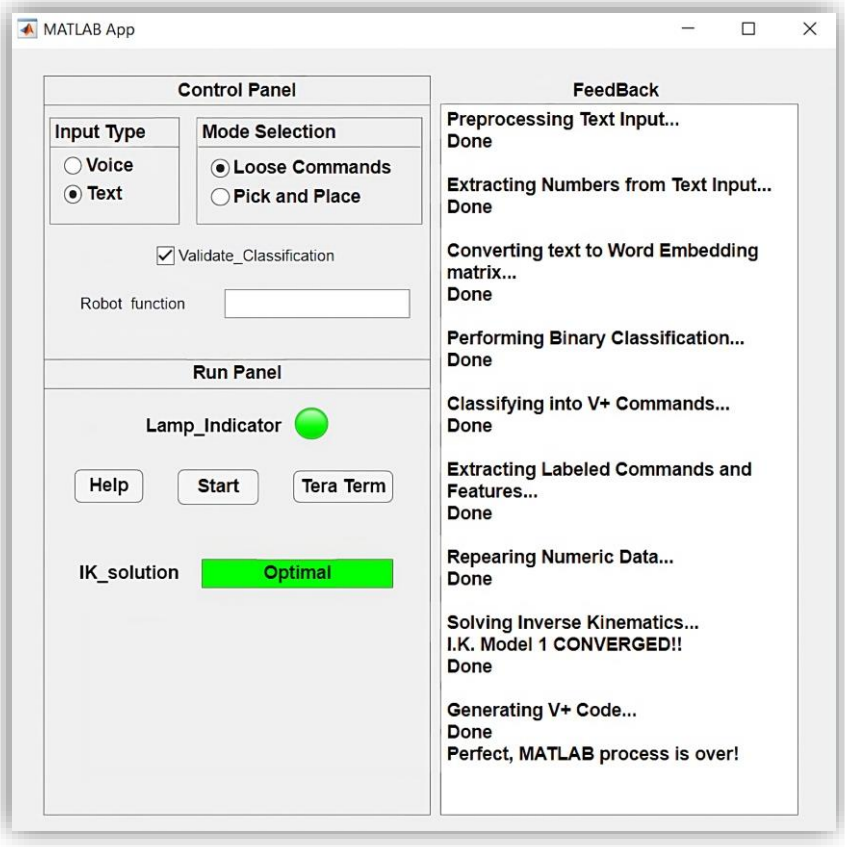
Σχήμα 3-14. Διάταξη Βραχίονα Παραδείγματος

Το παραγόμενο αρχείο από την παραπάνω διαδικασία περιλαμβάνει τον κώδικα που φαίνεται στο Σχήμα 3-15.

```
POINT #loc.target=#PPOINT(0,-135,180,0,45,0);

APPRO loc.target, 100;
ALIGN;
BREAK;
MOVES loc.target;
BREAK;

CLOSEI;

HERE loc.current;
DEPARTS 50;

OPENI;

DRIVE 3, -43, 50;
BREAK;

DO READY;
```

Σχήμα 3-15. Κώδικας V+ (Παράδειγμα Ελεύθερης Λειτουργίας)

Μετά το πέρας της ροής προγράμματος, η διεπαφή της εφαρμογής έχει τη μορφή που φαίνεται στο Σχήμα 3-16.



Σχήμα 3-16. GUI για Loose Commands

3.5.2 Παράδειγμα Λειτουργίας Pick and Place

Έστω λοιπόν ότι ο χρήστης επιλέγει τη λειτουργία pick and place με φωνητική είσοδο, και μετά το πάτημα του πλήκτρου Start, δίνει φωνητικά τα παρακάτω ως είσοδο στην εφαρμογή, όπως παρουσιάζει ο Πίνακας 3-12. Τονίζεται ότι ο χρήστης πρέπει οπωσδήποτε να ακολουθεί τους κανόνες που αναφέρθηκαν στην υποενότητα 3.3.2, όπως το να εκφράζει τους αριθμούς ξεκάθαρα και να τους διαχωρίζει με τη χρήση της λέξης slash.

Πίνακας 3-12. Δεδομένα φωνητικής εισαγωγής για pick & place

OK robot.
Position **pick** equals a thousand and fifty three point two slash zero slash three hundred eighteen point two slash zero slash ninety slash zero.
Position **place** equals zero slash a thousand and fifty three point two slash three hundred eighteen point two slash ninety slash ninety.
Location **alpha** equals eight hundred and seventy two point ninety two slash zero slash seven hundred fourteen point seventy one slash zero slash ninety slash zero.
Position **beta** equals zero slash eight hundred and seventy two point ninety two slash seven hundred fourteen point seventy one slash ninety slash ninety slash zero.
Height equals a hundred and fifty mm.
Distance x equals eighty mm and **distance** y equals ninety mm.
Repetition equals five times four.
Please do it now.

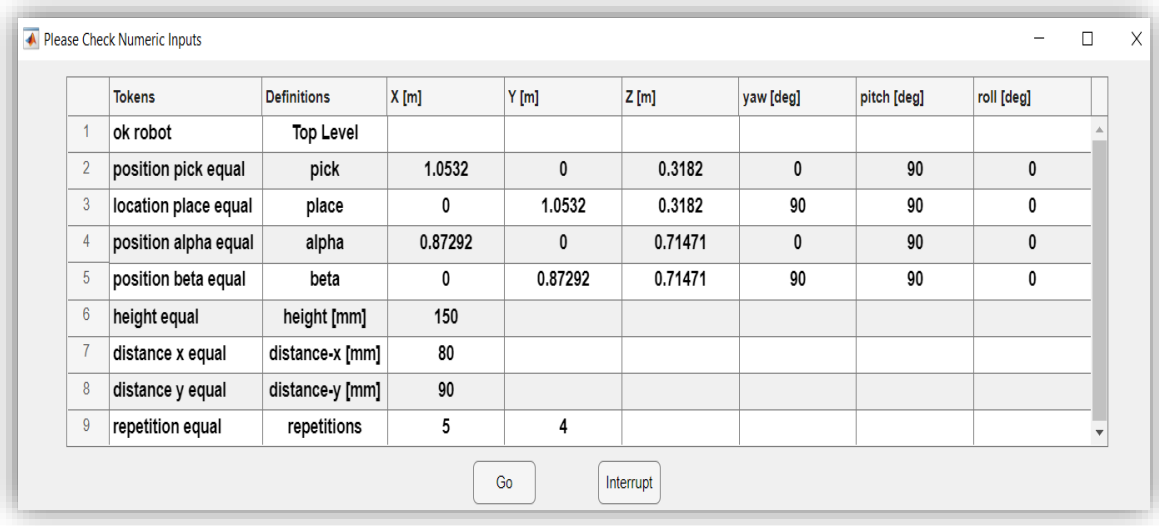
Μέσω της υπηρεσίας Microsoft Speech SDK λαμβάνεται ως αρχικό κείμενο αυτό που φαίνεται στο Σχήμα 3-17 όπου εύκολα παρατηρεί κανείς κάποια σφάλματα της Microsoft στη μετατροπή.

```
STARTED
 Session_id: 12dca48bff4641e499b65140d3dedb60
 Start talking to your microphone:

RECOGNIZED: OK, Robert, position pick equals 1053.2 slash 0 slash 318 point
2/0/90/0.

RECOGNIZED: Location place equals 0 slash 1053.2 slash 318 point 2/90/90/0 Position
alpha equals 872.92 slash 0 slash 714.71 slash 0/90/0.

RECOGNIZED: And position beta equals 0 slash 872.92 slash 714.71 slash 90/90/0.
Height equals 150 millimeters, distance X = 80 and distance y = 90 millimeters.
Repetitions equals 5 * 4.

RECOGNIZED: Do it now.

Terminating...
Closing: ResultReason.RecognizedSpeech
```

Σχήμα 3-17. Αναγνώριση Ομιλίας για Pick & Place

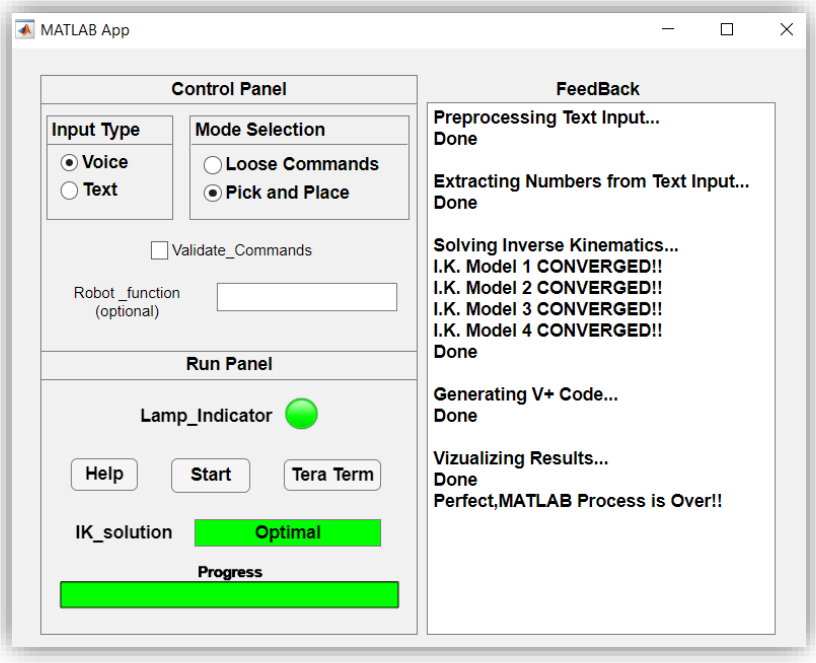
Κατά την αντιστοίχιση στις 4 ενδιάμεσες θέσεις και την εξαγωγή αριθμητικών δεδομένων εμφανίζεται στην οθόνη ένας επεξεργάσιμος πίνακας, όπως αυτός που φαίνεται στο Σχήμα 3-18, με σκοπό τον έλεγχο ή τη διόρθωση σφαλμάτων στην εισαγωγή αριθμών. Σημειώνεται ότι στο συγκεκριμένο παράδειγμα ο αριθμός στην $6^η$ γραμμή αναφέρεται σε ύψος (απόσταση) από επιθυμητές θέσεις, οι αριθμοί στην $7^η$ και $8^η$ γραμμή αναφέρονται σε αποστάσεις αντικειμένων κατά τη διεύθυνση x και y αντίστοιχα, και τα ψηφία στη $9^η$ γραμμή αναφέρονται

στον αριθμό των αντικειμένων κατά τις διευθύνσεις x και y αντίστοιχα. Υπενθυμίζεται ότι η σειρά των προτάσεων δε χρειάζεται να είναι συγκεκριμένη, αρκεί να ακολουθούνται οι κανόνες ορισμού θέσεων κλπ.



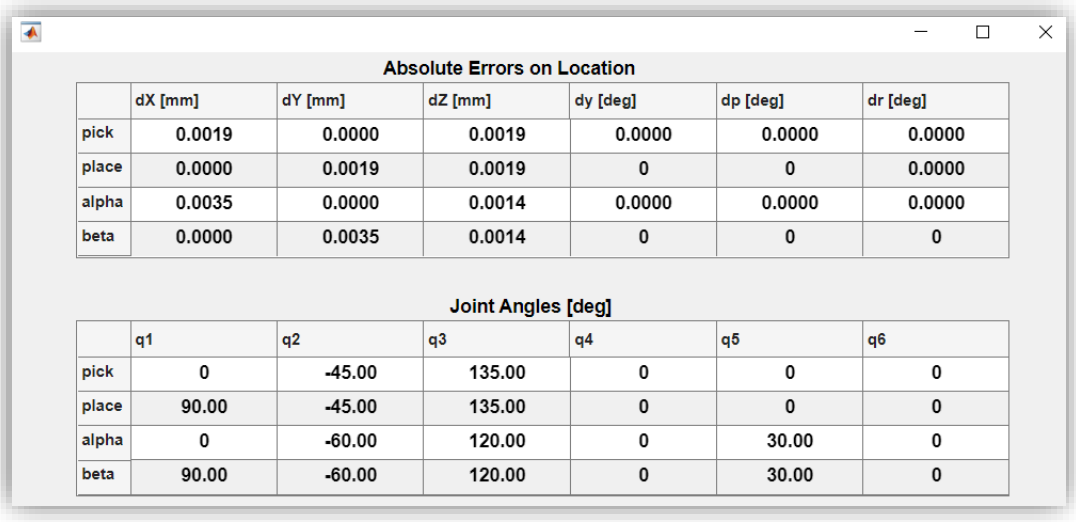
Σχήμα 3-18. Έλεγχος Αριθμητικών Δεδομένων (Pick & Place)

Μετά το πέρας της ροής προγράμματος η διεπαφή της εφαρμογής έχει τη μορφή που φαίνεται στο Σχήμα 3-19.



Σχήμα 3-19. GUI για pick & place

Στο Σχήμα 3-20 φαίνονται τα απόλυτα σφάλματα για τη θέση και τον προσανατολισμό του βραχίονα σε κάθε ενδιάμεση επιθυμητή θέση. Όλα δείχνουν ότι κύλησαν ομαλά και οι επιθυμητές τοποθεσίες επιτεύχθηκαν με αμελητέο σφάλμα.



Absolute Errors on Location

	dX [mm]	dY [mm]	dZ [mm]	dy [deg]	dp [deg]	dr [deg]
pick	0.0019	0.0000	0.0019	0.0000	0.0000	0.0000
place	0.0000	0.0019	0.0019	0	0	0.0000
alpha	0.0035	0.0000	0.0014	0.0000	0.0000	0.0000
beta	0.0000	0.0035	0.0014	0	0	0

Joint Angles [deg]

	q1	q2	q3	q4	q5	q6
pick	0	-45.00	135.00	0	0	0
place	90.00	-45.00	135.00	0	0	0
alpha	0	-60.00	120.00	0	30.00	0
beta	90.00	-60.00	120.00	0	30.00	0

Σχήμα 3-20. Αποτελέσματα pick & place

Μετά το πέρας της εκτέλεσης του προγράμματος, έχει παραχθεί ένα αρχείο (.txt ή .pg ή ό,τι μορφής επιθυμούμε) που περιλαμβάνει το κομμάτι κώδικα που φαίνεται στο Σχήμα 3-21. Αυτό το αρχείο μπορεί να εκτελεστεί μέσω του Tera Term και να εκτελεστούν οι εντολές από το βραχίονα Staubli RX90L.

```
SET #loc.pick=#PPOINT(0,-45,135,0,0,0);

SET #loc.alpha=#PPOINT(0,-60,120,0,30,0);

SET #loc.beta=#PPOINT(90,-60,120,0,30,0);

SET #loc.place=#PPOINT(90,-45,135,0,0,0);

MOVE loc.alpha;

FOR i=0 TO 4
        FOR j=0 TO 3
                SPEED 90;
                APPRO loc.pick:TRANS(i*80,j*90),150;
                ALIGN;
                SPEED 20;
                MOVES loc.pick:TRANS(i*80,j*90);
                BREAK;
                CLOSEI;
                DEPARTS 150;
                SPEED 90;
                MOVE loc.alpha;
                BREAK;
                MOVE loc.beta;
                APPRO loc.place:TRANS(i*80,j*90),150;
                ALIGN;
                SPEED 20;
                MOVES loc.place:TRANS(i*80,j*90);
                BREAK;
                OPENI;
                DEPARTS 150;
                SPEED 90;
                MOVE loc.beta;
                MOVE loc.alpha;
                BREAK;
        END
END

SPEED 90;

MOVE loc.alpha;
```

Σχήμα 3-21. Κώδικας V+ για pick & place

4 Συμπεράσματα και Προτάσεις για Μελλοντική Εργασία

4.1 Σύνοψη

Στα πλαίσια της παρούσας εργασίας έγινε ολοκλήρωση ενός συστήματος προγραμματισμού που αποτελείται από αρκετές συνιστώσες. Έγινε χρήση πακέτου μετατροπής ομιλίας σε κείμενο, χρήση στοιχείων του NLP για συμμόρφωση και καθαρισμό κειμένων, και εκπαιδεύτηκε δυαδικό μοντέλο SVM κατηγοριοποίησης εντολών. Ακόμα, αναπτύχθηκε μοντέλο επίλυσης αντίστροφης κινηματικής, και έγινε διερεύνηση της δυνατότητας προγραμματισμού βιομηχανικών διεργασιών μέσω φωνητικών εντολών. Τέλος, όλες οι συνιστώσες συνδυάστηκαν με τη δημιουργία γραφικής διεπαφής, η οποία προσφέρει στο χρήστη τη δυνατότητα προγραμματισμού ρομποτικού βραχίονα μέσω φωνητικών ή γραπτών εντολών. Επίσης, η γραφική διεπαφή προσφέρει ευελιξία ως προς την επιλογή ρομποτικού βραχίονα και προσφέρει δυνατότητα παρέμβασης του χρήστη τόσο στην κατηγοριοποίηση εντολών όσο και σε αριθμητικά δεδομένα. Καθ' όλη τη διάρκεια εκτέλεσης του προγράμματος, ο χρήστης ενημερώνεται μέσω της διεπαφής για τα βήματα που ακολουθούνται, αλλά και την ομαλή λειτουργία των συνιστωσών του συστήματος, όπως η ποιότητα της λύσης αντίστροφης κινηματικής. Εκ του αποτελέσματος, φαίνεται πως το σύστημα λειτουργεί αξιόπιστα, και οι στόχοι της παρούσας εργασίας επιτεύχθηκαν.

4.2 Συμπεράσματα

Το κύριο μέρος της εργασίας παρουσιάστηκε στο Κεφάλαιο 3, όπου είδαμε τη ροή του σήματος από τη φωνητική είσοδο του χρήστη προς την έξοδο. Η έξοδος είναι ένα αρχείο μέσω του οποίου θα εκτελεστούν οι εντολές του χρήστη από το ρομποτικό βραχίονα. Το παρόν σύστημα αναμένεται να συνεισφέρει σε βιομηχανικές εφαρμογές, αυξάνοντας την παραγωγικότητα, και μειώνοντας το χρόνο προγραμματισμού βιομηχανικών διεργασιών.

Στη συνέχεια, παρουσιάζονται τα κυριότερα συμπεράσματα για κάθε συστατικό του συστήματος.

Μετατροπή φωνής σε κείμενο

Για την επίτευξη των στόχων έγινε χρήση του έτοιμου πακέτου Microsoft Speech SDK. Ο κώδικας που εκμεταλλεύεται τη συγκεκριμένη υπηρεσία είναι γραμμένος σε γλώσσα Python, και καλείται εσωτερικά από τη γραφική διεπαφή που κατασκευάστηκε στο MATLAB. Ο κώδικας αυτός επιτρέπει:

- Αναγνώριση φωνητικών εντολών
- Αναγνώριση αριθμητικών ψηφίων
- Μετατροπή φωνής σε κείμενο, σε πραγματικό χρόνο

Επεξεργασία και Συμμόρφωση Κειμένου

Το στάδιο αυτό είναι υψίστης σημασίας για την ομαλή εκτέλεση των σκοπών της παρούσας εργασίας. Αυτό γιατί, αν προκύψει σφάλμα στο συγκεκριμένο στάδιο, αναπόφευκτα θα προκληθεί εκτροχιασμός του σήματος και τελικά δε θα εκτελεστούν οι εντολές του χρήστη όπως αρχικά επιθυμούσε. Για αυτό, μετά από διερεύνηση και πολλές επαναλήψεις, προστέθηκαν δικλείδες ασφαλείας που επιτρέπουν την παρέμβαση του χρήστη τόσο στα αριθμητικά δεδομένα όσο και στην κατηγοριοποίηση εντολών και ορισμών. Πιο συγκεκριμένα,

εκμεταλλευόμενοι τις εγγενείς συναρτήσεις του MATLAB Text Analytics Toolbox και διάφορες ομαλές εκφράσεις (regular expressions), δημιουργήθηκαν υπορουτίνες διορθωτικής φύσεως που αφορούν τις παρακάτω περιπτώσεις κατά την ηχογράφηση εντολών:

- Διόρθωση σφαλμάτων αναγνώρισης λέξεων.
- Διόρθωση σφαλμάτων αναγνώρισης αριθμών.
- Συνένωση προτάσεων σε περίπτωση ανεπιθύμητης διάσπασης προτάσεων.
- Διάσπαση προτάσεων που περιλαμβάνουν πολλαπλές εντολές.

Αναπαράσταση Λέξεων και Προτάσεων

Κατά την εκπόνηση της εργασίας έγινε σύγκριση των μορφών One-Hot Encoding, Bag of Words και Word Embeddings μέσω αξιολόγησης μοντέλων κατηγοριοποίησης κειμένου. Το μοντέλο που εκπαιδεύτηκε με Word Embeddings υπερτερεί σε σχέση με τα υπόλοιπα, και παρουσιάζει ακρίβεια (accuracy) 96% στα δεδομένα επαλήθευσης καθώς:

- Η αναπαράσταση λέξεων σε μορφή Word Embedding είναι συγκριτικά η πιο ευφυής, δεδομένου ότι δίνει πληροφορίες και για τη γενική σημασία των λέξεων.
- Οι δύο κλάσεις ήταν σχετικά εύκολα διαχωρίσιμες καθώς το νόημα των μισών δεδομένων σχετιζόταν με κίνηση ενός ρομποτικού βραχίονα, ενώ τα υπόλοιπα είχαν νόημα άσχετο με τους σκοπούς της συγκεκριμένης εργασίας.
- Εκ του αποτελέσματος φαίνεται πως τα δεδομένα της κάθε κλάσης είχαν επαρκή απόσταση εντός του χώρου χαρακτηριστικών, και το μοντέλο ήταν σε θέση να βρει τα κατάλληλα όρια (boundaries) ανάμεσα στις 2 κλάσεις.

Κατηγοριοποίηση εντολών

Η κατηγοριοποίηση εντολών πραγματοποιείται σε 2 στάδια. Αρχικά χρησιμοποιείται δυαδικό μοντέλο SVM για κατηγοριοποίηση προτάσεων ως εντολή ή μη-εντολή. Επιλέχθηκε μοντέλο SVM καθώς:

- Μπορεί να διαχειριστεί πολυδιάστατα δεδομένα εισόδου, ακόμα και αν οι διαστάσεις είναι περισσότερες από τις παρατηρήσεις. Υπενθυμίζεται ότι το μοντέλο SVM (με Word Embedding) εκπαιδεύτηκε με 160 παρατηρήσεις, διαστάσεων 1x300.
- Γενικεύεται αξιόπιστα σε νέα και άγνωστα δεδομένα.
- Είναι αποτελεσματικό για τυχόν δεδομένα που επικαλύπτονται.

Όσον αφορά την κατηγοριοποίηση ανάμεσα στις 7 προεπιλεγμένες εντολές V^+:

- Το περιορισμένο λεξιλόγιο που σχετίζεται με τους στόχους της παρούσας εργασίας ευνοεί την εύκολη κατηγοριοποίηση, και εξαλείφει την ανάγκη εκπαίδευσης μοντέλου μηχανικής μάθησης για κατηγοριοποίηση πολλαπλών κλάσεων.
- Έτσι, η περεταίρω κατηγοριοποίηση πραγματοποιήθηκε με βάση την παρουσία συγκεκριμένων λέξεων-κλειδιά που αντιστοιχούν στις προεπιλεγμένες εντολές V^+.

Αντίστροφη Κινηματική

Η επίλυση της αντίστροφης κινηματικής του ρομποτικού βραχίονα επιτρέπει στο χρήστη ορίσει εμμέσως τις επιθυμητές γωνίες αρθρώσεων, και έτσι να ελέγξει τη θέση και τον προσανατολισμό του βραχίονα. Η επίλυση της αντίστροφης κινηματικής αποτελεί κλειδί και στον φωνητικό προγραμματισμό της βιομηχανικής διεργασίας pick and place. Κατασκευάστηκε λοιπόν ένα μοντέλο επίλυσης όπου:

- Τέθηκαν περιορισμοί θέσης, προσανατολισμού, και επιβλήθηκε οι αρθρώσεις να περιστρέφονται εντός των ορίων που παρουσιάζει ο Πίνακας 2-2.

- Η φωνητική είσοδος του χρήστη για ορισμό επιθυμητών σημείων περιλαμβάνει την επιθυμητή θέση (σε χιλιοστά) και τον επιθυμητό προσανατολισμό (σε μοίρες) του βραχίονα μέσω γωνιών Euler (yaw, pitch, roll: ZYZ).
- Ο ορισμός προσανατολισμού δεν ορίζεται μοναδικά, υπό την έννοια ότι διαφορετικές γωνίες Euler μπορεί να αναπαριστούν τον ίδιο προσανατολισμό. Με αυτό ως δεδομένο, είναι πιθανόν στην έξοδο ο βραχίονας να μη βρεθεί ακριβώς στις γωνίες Euler που αρχικά ζήτησε ο χρήστης, αλλά εν τέλει ο προσανατολισμός θα είναι παρεμφερής.
- Από τις δοκιμές που πραγματοποιήθηκαν, τα σφάλματα ήταν αμελητέα οπότε συμπεραίνουμε πως η αντίστροφη κινηματική πράγματι επιλύεται αποτελεσματικά.
- Μετά το πέρας εκτέλεσης του προγράμματος, η γραφική διεπαφή επιστρέφει στο χρήστη γραφήματα που οπτικοποιούν τις επιθυμητές διατάξεις. Με αυτόν τον τρόπο ο χρήστης μπορεί να ελέγξει την ορθότητα των αποτελεσμάτων πριν την εκτέλεση του παραγόμενου αρχείου κώδικα V^+.

4.3 Προτάσεις για Μελλοντική Εργασία

Στα πλαίσια της παρούσας μεταπτυχιακής εργασίας έγινε μια πρώτη απόπειρα ολοκλήρωσης συστήματος προγραμματισμού ρομποτικού βραχίονα μέσω εντολών σε φυσική γλώσσα. Μέσα από προσεκτικό σχεδιασμό των συνιστωσών της εφαρμογής, το σύστημα λειτουργεί αξιόπιστα και φαίνεται να πετυχαίνει τους στόχους της παρούσας εργασίας. Φυσικά πάντα υπάρχει χώρος για βελτίωση, και αυτή η οπτική συχνά οδηγεί σε επέκταση ιδεών, στην εξέλιξη, και στη θωράκιση απέναντι σε πιθανά σφάλματα. Για το λόγο αυτό, στη συνέχεια γίνεται μια σύντομη εμβάθυνση στα σημεία όπου υπάρχουν πολλά υποσχόμενες προοπτικές βελτίωσης:

- Μετατροπή φωνής σε κείμενο:
    - Τα προ-εκπαιδευμένα μοντέλα της Microsoft προσφέρουν μεγάλη διευκόλυνση αλλά στην παρούσα μορφή του συστήματος δεν υπήρχε πρόσβαση στη λειτουργία τους, και αυτό αφήνει περιθώρια σφαλμάτων, ιδίως στη διατύπωση αριθμητικών δεδομένων και στη διάταξη των προτάσεων.
    - Θα μπορούσε ενδεχομένως να γίνει προσπάθεια εξατομίκευσης και να εκπαιδευτεί ένα μοντέλο (custom model) μετατροπής, που θα είναι περισσότερο προσανατολισμένο στους σκοπούς της παρούσας εργασίας.
- Επεξεργασία κειμένου:
    - Ίσως είναι χρήσιμη η προσθήκη δυνατότητας παρέμβασης στην τελική διάταξη προτάσεων, και να δίνεται δυνατότητα εισαγωγής ή και διαγραφής προτάσεων κατά την εκτέλεση του προγράμματος.
- Αναπαράσταση λέξεων και προτάσεων:
    - Η αναπαράσταση ολόκληρων προτάσεων ως ένα διάνυσμα που αντιστοιχεί στη «μέση» λέξη λειτουργεί ικανοποιητικά παρόλη την απλότητά της, αλλά δεν είναι η βέλτιστη επιλογή.
    - Θα μπορούσε να βρεθεί ακόμα πιο ευφυής τρόπος εξαγωγής ενσωμάτωσης προτάσεων (sentence embeddings), όπως με εκπαίδευση ή επέκταση υπάρχοντος μοντέλου BERT. Αυτό θα προκαλούσε δραματική βελτίωση στην αξιοπιστία μοντέλων κατηγοριοποίησης εντολών. Αυτό γιατί η θέση που θα καταλάμβανε η κάθε πρόταση στο χώρο των χαρακτηριστικών θα είχε

καθοριστεί με πιο ευφυή τρόπο, και θα ήταν λιγότερο πιθανό να παρουσιαστεί σφάλμα κατηγοριοποίησης λόγω παρουσίας μεμονωμένων λέξεων.

- Κατηγοριοποίηση εντολών:
    - Το δυαδικό μοντέλο SVM που χρησιμοποιήθηκε λειτουργεί ικανοποιητικά αλλά ενδέχεται να χρήζει βελτίωσης καθώς, η μορφή αναπαράστασης δεδομένων παίζει καθοριστικό ρόλο στην ισχύ ενός μοντέλου.
    - Ίσως η αντικατάσταση της μεθόδου SVM από ένα νευρωνικό δίκτυο βαθιάς μάθησης να προκαλέσει σημαντική βελτίωση στην αξιοπιστία της κατηγοριοποίησης εντολών.

Ο αναγνώστης, αν το επιθυμεί, μπορεί να επεκτείνει τη βασική ιδέα της παρούσας εργασίας και να ενισχύσει την εκτέλεση, καθώς υπάρχουν διαφορετικές οδοί και τεχνικές που μπορεί κανείς να ακολουθήσει για την επίτευξη των στόχων. Συνοψίζοντας τις προτάσεις μας για μελλοντικές εργασίες και έρευνες προτείνεται:

1. Διερεύνηση διαφορετικών τρόπων ορισμού συντεταγμένων, κυρίως όσων αφορά τον προσανατολισμό στον καρτεσιανό χώρο.
2. Αναβάθμιση ευελιξίας, μέσω προσθήκης περισσότερων επιλογών και δυνατοτήτων για το χρήστη.
3. Προσπάθεια εκμετάλλευσης περισσότερων εντολών της γλώσσας προγραμματισμού βραχίονα.
4. Προσπάθεια επέκτασης της ιδέας για προγραμματισμό άλλων βιομηχανικών διεργασιών, όπως διάτρηση οπών.
5. Ενσωμάτωση αντίστοιχου συστήματος σε σύστημα που χρησιμοποιεί και άλλες ευφυείς τεχνικές, όπως μηχανική όραση.
6. Αναβάθμιση για ασύρματη χρήση, όπως ένα τηλεχειριστήριο με οθόνη, που θα δίνει στο χρήστη αντίστοιχες δυνατότητες με τη γραφική διεπαφή της εφαρμογής μας.

5 Βιβλιογραφία

1. Μαραγκός, Χ., & Maragkos, C. (2018). *Προγραμματισμός βιομηχανικού ρομποτικού βραχίονα για αποφυγή σύγκρουσης με συνεργαζόμενο άνθρωπο*. https://doi.org/10.26240/heal.ntua.15076
2. Μίχας, Σ., & Michas, S. (2016). *Προγραμματισμός Τροχιάς και Τηλεπαρακολούθηση Λειτουργίας Ρομποτικού Βραχίονα σε Περιβάλλον Εικονικής Πραγματικότητας με χρήση Αισθητήρων Ηλεκτρονικών Συσκευών Ευρείας Κατανάλωσης.* https://doi.org/10.26240/heal.ntua.10708
3. Adept Technology, Inc. (1997). *V+ Language Reference Guide*.
4. Adept Technology, Inc. (1997). *V+ Language User's Guide, Ver. 12.1*.
5. Bogert, B.P. (1963). *The quefrency analysis of time series for echoes: cepstrum, pseudo-autocovariance, cross-cepstrum and saphe cracking*.
6. Cai Y., M. -Y. Chow, W. Lu and L. Li, "*Evaluation of distribution fault diagnosis algorithms using ROC curves*," *IEEE PES General Meeting*, Minneapolis, MN, USA, 2010, doi: 10.1109/PES.2010.5588154
7. Choe, D. K., & Charniak, E. (2016). *Parsing as Language Modeling. Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2331–2336. https://doi.org/10.18653/v1/D16-1257
8. Chomsky, N. (1986). *Knowledge of Language: Its Nature, Origin and Use*. New York: Praeger.
9. Christodoulides, G. (2014). Praaline: *Integrating Tools for Speech Corpus Research*. 2014, 31–34.
10. Colby, K.M. (1974). *Ten criticisms of parry. SIGART Newsl*., 48, 5-9.
11. Cortes, C., & Vapnik, V. (1995). *Support-vector networks. Machine Learning*, 20(3), 273–297. https://doi.org/10.1007/BF00994018
12. Craig, J. J. (2005). *Mechanics and Control Third Edition*.
13. Cummins, N., Amiriparian, S., Ottl, S., Gerczuk, M., Schmitt, M., & Schuller, B. (2018). Multimodal Bag-of-Words for Cross Domains Sentiment Analysis. 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 4954–4958. https://doi.org/10.1109/ICASSP.2018.8462660
14. Deep Learning. (n.d.). Retrieved March 8, 2023, from https://www.deeplearningbook.org/
15. Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* https://doi.org/10.18653/v1/N19-1423
16. Devore, J. (2006). *A Modern Introduction to Probability and Statistics: Understanding Why and How. Journal of the American Statistical Association*, 101(473), 393–394. https://doi.org/10.1198/jasa.2006.s72
17. English word vectors · fastText. (n.d.). Retrieved March 17, 2023, from https://fasttext.cc/index.html
18. European Telecommunications Standards Institute (2003), *Speech Processing, Transmission and Quality Aspects (STQ); Distributed speech recognition; Front-end feature extraction algorithm; Compression algorithms. Technical standard ES 201 108, v1.1.3*
19. Forney Jr, G. D. (2005). *The Viterbi Algorithm: A Personal History*. http://arxiv.org/abs/cs/0504020
20. Goldberg, Y. (2016). *A Primer on Neural Network Models for Natural Language Processing. Journal of Artificial Intelligence Research*, 57, 345–420. https://doi.org/10.1613/jair.4992

21. Howard, T. M., Tellex, S., & Roy, N. (2014). *A natural language planner interface for mobile manipulators. 2014 IEEE International Conference on Robotics and Automation (ICRA)*, 6652–6659. https://doi.org/10.1109/ICRA.2014.6907841
22. Huang, H., & Zhang, X. (2009). Part-of-speech tagger based on maximum entropy model. 2009 2nd IEEE International Conference on Computer Science and Information Technology, 26–29. https://doi.org/10.1109/ICCSIT.2009.5234787
23. Hutchins, W.J. (2004). *The Georgetown-IBM Experiment Demonstrated in January 1954. In: Frederking, R.E., Taylor, K.B. (eds) Machine Translation: From Real Users to Research. AMTA 2004. Lecture Notes in Computer Science(), vol 3265. Springer, Berlin, Heidelberg*. https://doi.org/10.1007/978-3-540-30194-3_12
24. Hutchins, J. (1996). *ALPAC -- the (in)famous report*.
25. Inverse Kinematics Algorithms—MATLAB & Simulink. (2023). https://www.mathworks.com/help/robotics/ug/inverse-kinematics-algorithms.html
26. Karjalainen, M. (1988). *Speech communication, human and machine: by Douglas O'Shaughnessy, INRS-Telecommunication. Publisher: Addison-Wesley Publishing Company, Route 128, Reading, MA 01867, U.S.A., 1987, ISBN 0-201-16520-1. Signal Processing, 15, 217-218*.
27. Kasthuriarachchy, B. H., De Zoysa, K., & Premaratne, H. L. (2014). Enhanced bag-of-words model for phrase-level sentiment analysis. 2014 14th International Conference on Advances in ICT for Emerging Regions (ICTer), 210–214. https://doi.org/10.1109/ICTER.2014.7083903
28. Khadilkar, K., Kulkarni, S., & Venkatraman, S. (2019). A Knowledge Graph Based Approach for Automatic Speech and Essay Summarization. 2019 IEEE 5th International Conference for Convergence in Technology (I2CT), 1–6. https://doi.org/10.1109/I2CT45611.2019.9033908
29. Kim, J.-K., Tur, G., Celikyilmaz, A., Cao, B., & Wang, Y.-Y. (2016). Intent detection using semantically enriched word embeddings. 2016 IEEE Spoken Language Technology Workshop (SLT), 414–419. https://doi.org/10.1109/SLT.2016.7846297
30. Lap-Pui Chau, Lun, D. P.-K., & Wan-Chi Siu. (2001). *Efficient prime factor algorithm and address generation techniques for the discrete cosine transform. IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 48(10), 985–988. https://doi.org/10.1109/82.974787
31. Li, P., Li, J., & Wang, G. (n.d.). Application of Convolutional Neural Network in Natural Language Processing.
32. Mah, P. M., Skalna, I., & Muzam, J. (2022). *Natural Language Processing and Artificial Intelligence for Enterprise Management in the Era of Industry 4.0. Applied Sciences*, 12(18), 9207. https://doi.org/10.3390/app12189207
33. Mahmud, T., Azharul Hasan, K. M., Ahmed, M., & Thwoi Hla Ching Chak. (2015). A rule-based approach for NLP based query processing. 2015 2nd International Conference on Electrical Information and Communication Technologies (EICT), 78–82. https://doi.org/10.1109/EICT.2015.7391926
34. Matuszek, C., Herbst, E., Zettlemoyer, L., & Fox, D. (2013). *Learning to Parse Natural Language Commands to a Robot Control System*. In J. P. Desai, G. Dudek, O. Khatib, & V. Kumar (Eds.), *Experimental Robotics* (Vol. 88, pp. 403–415). *Springer International Publishing*. https://doi.org/10.1007/978-3-319-00065-7_28
35. Matuszek, C. (2018). *Grounded Language Learning: Where Robotics and NLP Meet. Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, 5687–5691. https://doi.org/10.24963/ijcai.2018/810
36. McTear, M. F. (2002). *Spoken dialogue technology: Enabling the conversational user interface. ACM Computing Surveys*, 34(1), 90–169. https://doi.org/10.1145/505282.505285
37. Microsoft Speech Application Programming Interface (API) and SDK, Microsoft Corporation, http://www.microsoft.com/speech

38. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). *Efficient Estimation of Word Representations in Vector Space.* http://arxiv.org/abs/1301.3781
39. Misra, D., Langford, J., & Artzi, Y. (2017). *Mapping Instructions and Visual Observations to Actions with Reinforcement Learning. Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 1004–1015. https://doi.org/10.18653/v1/D17-1106
40. Mooney, R. J. (2008). *Learning to Connect Language and Perception*.
41. Muda, L., Begam, M., & Elamvazuthi, I. (2010). *Voice Recognition Algorithms using Mel Frequency Cepstral Coefficient (MFCC) and Dynamic Time Warping (DTW) Techniques*. 2(3).
42. Natural language processing. (2023). In Wikipedia. https://en.wikipedia.org/w/index.php?title=Natural_language_processing&oldid=1135665480
43. Ondas, S., Juhar, J., Pleva, M., Cizmar, A., & Holcer, R. (2013). *Service Robot SCORPIO with Robust Speech Interface. International Journal of Advanced Robotic Systems*, 10(1), 3. https://doi.org/10.5772/54934
44. Ontanon, S. (2018). *SHRDLU: A Game Prototype Inspired by Winograd's Natural Language Understanding Work. Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, 14(1), 268–270*. https://doi.org/10.1609/aiide.v14i1.13009
45. Panek, P., & Mayer, P. (2015). *Challenges in Adopting Speech Control for Assistive Robots. In R. Wichert & H. Klausing (Eds.), Ambient Assisted Living* (pp. 3–14). *Springer International Publishing*. https://doi.org/10.1007/978-3-319-11866-6_1
46. Pinola, M. (2011). *Speech Recognition Through the Decades: How We Ended Up With Siri*.https://www.pcworld.com/article/477914/speech_recognition_through_the_decades_how_we_ended_up_with_siri.html
47. Pollettini, J. T., Pessotti, H. C., Filho, A. P., Ruiz, E. E. S., & Junior, M. S. A. (2015). Applying Natural Language Processing, Information Retrieval and Machine Learning to Decision Support in Medical Coordination in an Emergency Medicine Context. 2015 IEEE 28th International Symposium on Computer-Based Medical Systems, 316–319. https://doi.org/10.1109/CBMS.2015.82
48. Powers, D. (2008). Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation. *Mach. Learn. Technol.*, *2*.
49. Priyadarshana, V. P. S. C., Yaparathne, Y. M. P. G. R. R. Y., Jayasooriya, D. B. S., Thennakoon, T. M. T. S. D. B., Jayasekara, A. G. B. P., & Chandima, D. P. (2022). Voice Controlled Robot Manipulator for Industrial Applications. 2022 IEEE 13th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), 0160–0165. https://doi.org/10.1109/IEMCON56893.2022.9946545
50. Rangu, C., Chatterjee, S., & Valluru, S. R. (2017). Text Mining Approach for Product Quality Enhancement: (Improving Product Quality through Machine Learning). 2017 IEEE 7th International Advance Computing Conference (IACC), 456–460. https://doi.org/10.1109/IACC.2017.0100
51. Rashid, H., Ahmed, I. U., Osman, S. B., Newaz, Q., & Reza, S. M. T. (2017). *Design and Implementation of a Voice Controlled Robot with Human Interaction Ability*.
52. Robots—*The Robotics Institute Carnegie Mellon University*. (n.d.). Retrieved April 29, 2023, from https://www.ri.cmu.edu/research/robots/
53. Sahidullah, Md., & Saha, G. (2012). *Design, analysis and experimental evaluation of block based transformation in MFCC computation for speaker recognition. Speech Communication*, 54(4), 543–565. https://doi.org/10.1016/j.specom.2011.11.004
54. Saini, V., & Joseph, N. (2022). *Artificial Intelligence in Robotics Using NLP.*
55. Sakoe, H., & Chiba, S. (1978). *Dynamic programming algorithm optimization for spoken word recognition. IEEE Transactions on Acoustics, Speech, and Signal Processing, 26(1), 43–49*. https://doi.org/10.1109/TASSP.1978.1163055

56. Sivakumar, M. S., Murji, J., Jacob, L. D., Nyange, F., & Banupriya, M. (2013). Speech controlled automatic wheelchair. 2013 Pan African International Conference on Information Science, Computing and Telecommunications (PACT), 70–73. https://doi.org/10.1109/SCAT.2013.7055093
57. *Speech and Language Processing*. (n.d.). Retrieved April 19, 2023, from https://web.stanford.edu/~jurafsky/slp3/
58. Tasevski, J., Nikolic, M., & Miskovic, D. (2013). *Integration of an industrial robot with the systems for image and voice recognition. Serbian Journal of Electrical Engineering*, 10(1), 219–230. https://doi.org/10.2298/SJEE1301219T
59. Turing, A. M. (1950). I.—*COMPUTING MACHINERY AND INTELLIGENCE. Mind, LIX(236), 433–460*. https://doi.org/10.1093/mind/LIX.236.433
60. van Delden, S., & Overcash, B. (2008). *Towards Voice-Guided Robotic Manipulator Jogging*.
61. Vinyals, O., Kaiser, Ł., Koo, T., Petrov, S., Sutskever, I., & Hinton, G. (2015). Grammar as a Foreign Language.
62. Weizenbaum, J. (1966). ELIZA—a computer program for the study of natural language communication between man and machine.
63. Wu, J., Huang, X., Liu, J., Huo, Y., Yuan, G., & Zhang, R. (2023). NLP Research Based on Transformer Model. 2023 IEEE 10th International Conference on Cyber Security and Cloud Computing (CSCloud)/2023 IEEE 9th International Conference on Edge Computing and Scalable Cloud (EdgeCom), 343–348. https://doi.org/10.1109/CSCloud-EdgeCom58631.2023.00065
64. Xu, M., Duan, L.-Y., Cai, J., Chia, L.-T., Xu, C., & Tian, Q. (2004). *HMM-Based Audio Keyword Generation*. In K. Aizawa, Y. Nakamura, & S. Satoh (Eds.), *Advances in Multimedia Information Processing—PCM* 2004 (Vol. 3333, pp. 566–574). *Springer Berlin Heidelberg*. https://doi.org/10.1007/978-3-540-30543-9_71
65. Zhang H., L. Xiao, P. Yan and Q. Xiao, "*Research on Speech Recognition of Power Grid Dispatching Based on Big Data and Deep Learning*," 2021 International Conference on Power System Technology (POWERCON), Haikou, China, 2021, doi: 10.1109/POWERCON53785.2021.9697721

Παράρτημα Α

Κώδικας Python:

```
import os
import azure.cognitiveservices.speech as speechsdk
import time


def recognize_from_microphone():
    # Configure connection
    speech_config = speechsdk.SpeechConfig(subscription=os.environ.get('SPEECH_KEY'), region=os.environ.get('REGION'))
    speech_config.speech_recognition_language = "en-US"
    speech_config.endpoint_id = "https://westeurope.api.cognitive.microsoft.com/sts/v1.0/issuetoken"

    speech_config.set_profanity(speechsdk.ProfanityOption.Raw)  # Do not * trash talking

    #speech_config.set_property(property.SpeechServiceConnection_InitialSilenceTimeoutMs, "1000");
    #speech_config.set_property(SpeechServiceConnection_InitialSilenceTimeoutMs,"1000");

    audio_config = speechsdk.audio.AudioConfig(use_default_microphone=True)
    speech_recognizer = speechsdk.SpeechRecognizer(speech_config=speech_config, audio_config=audio_config)
    phrase_list_grammar = speechsdk.PhraseListGrammar.from_recognizer(speech_recognizer)


    filename = 'Dictionary.txt'
    with open(filename, 'r') as f:
        dict=str(f.read())
        dict=dict.split()

    for word in dict:
        phrase_list_grammar.addPhrase(word)



    done = False
    t = list()


        # Start the process
    speech_recognizer.session_started.connect(lambda evnt: print('\nSTARTED\n Session_id: {}\n Start talking to your
microphone:'.format(evnt.session_id)))
    speech_recognizer.start_continuous_recognition()


    # Define and call recognition and terminating functions
    def on_recognition(evt):
        stop = evt.result.text.__contains__("Stop") or evt.result.text.__contains__("stop") or evt.result.text.__contains__("Do it
now") or evt.result.text.__contains__("do it now")
        print('\nRECOGNIZED: {}'.format(evt.result.text))
        nonlocal t
        t.append(evt.result.text)
        if (stop):
            stop_cb(evt)


    def stop_cb(evt):
        print('\nTerminating...\nClosing: {}'.format(evt.result.reason))
        nonlocal done
        done = True
        speech_recognizer.stop_continuous_recognition()


    # Call the recognised Callback
    speech_recognizer.recognized.connect(on_recognition)

    while not done:
        time.sleep(0.4)

    speech_recognizer.session_stopped.connect(stop_cb)
    speech_recognizer.canceled.connect(stop_cb)

    return t
```

Κώδικες MATLAB:

MAIN_Script.m:

```
if ~exist('init_needed',"var")
    Initialize;
else
    clc
    fprintf("\nNo need to Initialize\n");
end

validateCommands=true;
converged_flg=[];

%% Run Speech2Text

diary ('myDiaryFile.txt')

PythonSpeech2Text;

if ~interrupt_flg
    clc;
    fprintf("Preprocessing Text Input...\n")
    [t0,documents0,details0] = preprocess(txt);
    fprintf("Done\n")

        % Extract Numbers and clear them from text input
    fprintf("\nExtracting Numbers from Text Input...\n")
    [t,documents,details,nums]=Extract_Numeric_Inputs(t0,documents0,details0);
    fprintf("Done\n")

    fprintf("\nConverting txt to Word Embedding...\n")

    [X,documents,sentence2words]=Build_Embedding(documents,emb);

    fprintf("Done\n")


    fprintf("\nPerforming Binary Classification...\n")


    [y_pred,taaab]=Classify(mdl_binary,X,'svm');


    isCommand = (y_pred=="T");
    fprintf("Done\n")

    if any(isCommand)
        fprintf("\nClassifying into V+ Commands...\n")
         X_command=X;
        numbers=nums;
        labelsNew =Classify(mdl,X_command,'ecoc');

        z=MyLabelPredictions(t.Tokens,isCommand);
        tab=table(t.Tokens,sentence2words,isCommand,labelsNew,z,...
            'VariableNames',["Tokens","Mean word","SVM","EMB Class","Class"]);
        fprintf("Done\n")

        fprintf("\nExtracting Labeled Commands and Features...\n")


        [commands,TopLevelFlag,needs_inverse_calc_flag]=SearchCommandFeatures(tab,isCommand,numbers,va
        lidateCommands);
        ind=ismissing(commands);

        % Clean out any Non-Command input from commands and numbers
        if any(ind)
            commands(ind)="";
        end

        fprintf("Done\n")

        if TopLevelFlag
```

```
            if needs_inverse_calc_flag
                fprintf("\nSolving Inverse Kinematics...\n")

                [numbers_new,joint_angles,location,converged_flg,ik_ind]=My_IK_setup(commands,numbers,
                robot);

                if any(converged_flg)
                    fprintf("Done\n")
                else
                    fprintf("No optimal solution found!\n")
                end
            else
                fprintf("\nNo need for Inverse Kinematics\n")
                stopFlow=false;
                numbers_new=numbers;
            end

            %% Determine if You wish to Continue
            if (needs_inverse_calc_flag && ~isempty(converged_flg) && any(~converged_flg))
                for i=1:length(ik_ind)
                    if converged_flg(i)
                        stopFlow=false;
                        continue
                    else
                        loc=location(i,:);
                        angs=joint_angles(i,:);
                        actual_loc=tform2trvec(getTransform(robot,angs*pi/180,'tool','base'));
                        R=tform2rotm(getTransform(robot,angs*pi/180,'tool','base'));
                        Rq=quaternion(R,'rotmat','point');
                        eull=euler(Rq,'ZYZ','frame')*180/pi;

                        fprintf("\nThe desired location is probably unfeasible!!\n\n")
                        fprintf("Desired Location:
%0.3f\t%0.3f\t%0.3f\t%0.3f\t%0.3f\t%0.3f\n",[1000*loc(1:3) loc(4:6)]);
                        fprintf("\nActual Location:
%0.3f\t%0.3f\t%0.3f\t%0.3f\t%0.3f\t%0.3f\n",[1000*actual_loc,eull]);
                        fprintf("Best I.K Solution: %0.3f\t%0.3f\t%0.3f\t%0.3f\t%0.3f\t%0.3f",angs);

                        inpt2=input("\nKeep sub-optimal solution?[y/n] ",'s');
                        if strncmp(inpt2,"n",1) || strncmp(inpt2,"N",1)
                            inpt3=input("\nManually change the solution?[y/n] ",'s');

                            if strncmp(inpt3,"n",1) || strncmp(inpt3,"N",1)
                                fprintf('\nProcess has stopped!\n')
                                stopFlow=true;
                            else
                                inpt4=input("\nPlease insert a row of 6 desired angles in
degrees",'s');

                                inpt4=str2double(string(split(inpt4))');
                                angs=inpt4;
                                if length(angs)<6
                                    angs(length(angs):1:6)=0;
                                end
                                numbers_new(ik_ind(i),1:6)=angs;
                                joint_angles(i,:)=angs;
                                stopFlow=false;
                            end

                        else
                            stopFlow=false;
                        end
                    end
                end

            elseif (~isempty(converged_flg) && all(converged_flg)) ||...
                    (~needs_inverse_calc_flag && isempty(converged_flg))
                stopFlow=false;
                fprintf("No need for I.K. corrections\n")
            end

            if ~stopFlow
                if all(commands=="PreCommand")
                    fprintf("\nYou only gave a TopLevel Command, process is finished\n");
                else
```

```
                    if any(~isnan(numbers(:,1))) ||  (any(commands=="OPENI") ||
any(commands=="CLOSEI"))
                        fprintf("\nGenerating V+ Code...\n")
                        command_list=sort(mdl.ClassNames,"descend");

                        filename="MyStaubliRX90L.txt";
                        Extract_file_for_V(filename,commands,numbers_new,command_list,TopLevelFlag)
                        open MyStaubliRX90L.txt
                        fprintf("Done\n")
                        fprintf("\n\tGood, process is done!\n")
                        fprintf("\n Opening Tera Term...\n")
                        system("C:\Program Files (x86)\teraterm\ttermpro.exe")
                    else
                        fprintf("\nNo Numeric Inputs found!\nNo need to generate V+ code!\n")
                    end

                    if ~isempty(joint_angles)
                        for l=1:size(joint_angles,1)
                            aangs=joint_angles(l,:);
                            position=location(l,:);
                            f=figure;
                            f.Position=[900 100 600 600];
                            f=show(robot, aangs*pi/180,'Collisions','on');
                            hold on;
                            text(gca,position(1),position(2),position(3)+0.2,...
                            '\downarrowTarget','FontWeight','bold','FontSize',15)
                            title(gca,['Joint angles= ',num2str(aangs),'
degrees'],"FontWeight","bold","FontSize",12)
                            f.ZLim=[-0.5 1.3];
                            f.YLim=[-1.3 1.3];
                            f.XLim=[-1.3 1.3];
                            f.View=[135 5];
                            pause(0.1);
                        end
                    end

                end
            else
                fprintf("\nSolution Dismissed, process stopped!\n")
            end
        else
            fprintf("\n You did not give a Top Level Command\n")
        end

    else
        fprintf("\nYou did not give any known commands for Staubli RX90L!\n")
    end

end
```

Initialize.m:

```
clear;clc;close all;

fprintf("\nInitializing...\n");

if count(py.sys.path,pwd)==0
    fprintf("Adding Python module to path...\n")
    insert(py.sys.path,int32(0),pwd)
    pyenv('ExecutionMode', 'OutOfProcess');
end

fprintf("Loading Binary Classification Model...\n")
load 'MDL_binary.mat'

fprintf("Loading Fast Text Word Embedding...\n")
load 'EMB.mat'

robot=create_RX90L_fcn();
init_needed=false;
joint_angles=[];
```

```
location=[];
fprintf("Done\n")
```

PythonSpeech2Text.m

```
while (1)
try
    clc;
      tx=py.real_time_SR_fromMic.recognize_from_microphone();

    txt=string(tx)';
    clear tx
    if contains(txt(end),"stop",'IgnoreCase',true)
        clc
        fprintf("\nYou requested an interruption, process suspended!\n")
        interrupt_flg=true;
        inpt=input("\nDo you want to try again? [y/n]\n","s");
        if strncmp(inpt,"n",1) || strncmp(inpt,"N",1)
            clc;
            fprintf("\nProcess Stopped!\n")
            break
        end

    elseif contains(txt(end),"Do it now",'IgnoreCase',true)
        clc
        interrupt_flg=false;
        break
    end

catch e
    e.message
    fprintf("\nSomething is wrong, please fix any issues!\n")
    if(isa(e,'matlab.exception.PyException'))
        fprintf("\nPython exception:\n")
        e.ExceptionObject
    end
   break
end

end
```

preprocess.m:

```
function [textData,documents,details] = preprocess(inpt)

        textData0=inpt;

        % Clean up the text
        textData=repair_text(textData0);

        % Create tokenized documents
        documents = tokenizedDocument(textData(:,1));

        documents = correctSpelling(documents);
        documents = addPartOfSpeechDetails(documents);
        details = tokenDetails(documents);
        i=0;

        while 1             % Sentence loop
                if i<length(documents)
                        i=i+1;
                else
                        break;
                end
                doc=documents(i);
                str=split(string(doc));
                j=0;            % Word index
```

```
                while 1         % Word loop
                    if j<length(str)
                        j=j+1;
                    else
                        break;
                    end
                    ind=str(j)==details.Token;
                    ind=find(ind,1);
                    part=details.PartOfSpeech(ind);

                    cond1=part==categorical("verb");

                    cond2=j>2 && (strncmp(str(j-1),',',1) || strncmpi(str(j-1),'.',1) ||
strcmpi(str(j-1),'and')||...
                        strcmpi(str(j-1),'then') ||strcmpi(str(j-1),'but') || strcmpi(str(j-
1),'or')||...
                        strcmpi(str(j-1),'firstly') || strcmpi(str(j-1),'secondly') ||...
                        strcmpi(str(j-1),'finally') || strcmpi(str(j-1),'before')));

                    cond3=any(strcmpi(str(j),dictionary_verbs));


                    if ((cond2&&cond1) || (cond2&&cond3))
                        if i<length(documents)
                            documents(i+2:end+1)=documents(i+1:end);
                        end
                        documents(i)=tokenizedDocument(join(str(1:j-2)));
                        documents(i+1)=tokenizedDocument(join(str(j:end)));
                        documents = addPartOfSpeechDetails(documents);
                        details=tokenDetails(documents);
                        str=split(string(documents(i)));
                        if j>=length(str)
                            break
                        end
                    else
                        continue
                    end

                    documents = removeEmptyDocuments(documents);
                    documents = addPartOfSpeechDetails(documents);
                    details=tokenDetails(documents);
                    str=split(string(documents(i)));

                    if j>=length(str)
                        break
                    end

                end


                if i>=length(documents)
                    break
                end

            end

            documents = removeEmptyDocuments(documents);

            if contains(documents(end),'Do it now','IgnoreCase',true)
                documents(end)=[];
            end


            %% Clean the final documents

            documents = addPartOfSpeechDetails(documents);
            documents=removeStopWords(documents);
            try
                documents = normalizeWords(documents,'Style','lemma');  % it sometimes mistakes the
language and throws error!
            catch
                documents =documents(:,1);
            end
```

```
            documents = erasePunctuation(documents,'TokenTypes','punctuation');   % Do not delete it
from decimal numbers

            %documents = addDependencyDetails(documents);
            documents=lower(documents);


        documents=removeWords(documents,["mm","millimeter","finally","firstly","secondly","please"," s
        "]);
            documents =replaceWords(documents,"desire","desired");

            documents = removeEmptyDocuments(documents);

            details = tokenDetails(documents);

        details.DocumentNumber=[];details.SentenceNumber=[];details.LineNumber=[];details.Language=[];

            textData=cellstr(documents.joinWords);

            textData=cell2table(textData,'VariableNames',"Tokens");
            textData.Tokens=string(textData.Tokens);

end


function textData=repair_text(textData)

    textData=join(textData);

    textData=replace(textData,"OK,","OK");
    textData=replace(textData,"dessert","desired");
    textData=replace(textData,"shared","desired");
    textData=replace(textData,"right","desired");
    textData=replace(textData,"dessert","desired");
    textData=replace(textData," size "," seize");
    textData=replace(textData," cease "," seize");

    textData=replace(textData," ex ","x");
    textData=replace(textData," why ","y");
    textData=replace(textData,"peak","pick");
    textData=replace(textData,"better","beta");
    textData=replace(textData," oh "," 0 ");
    textData=replace(textData," good ",", go to ");

    textData=replace(textData,","," ");
    textData=replace(textData," end "," and ");


     % Fix to word: move
    expre='\<\w*move\w*';   % Any word containing move. eg. remove->move
    textData=regexprep(textData,expre,"move");

    % Fix to word:robot
    expre='\<[Rr]ob\w*';   % Any word starting with rob
    textData=regexprep(textData,expre,"robot");
    textData=replace(textData,"bro, but","robot");
    textData=replace(textData,"robot","robot. ");

    % Fix many white spaces to a single white space
    expre='\s{2,}';
    textData=regexprep(textData,expre," ");
    expre='\s+[.]';
    textData=regexprep(textData,expre,".");


    % Fix "/" and "comma" to word "slash" and remove it
    expre='\s*\/[\s.]*';
    textData=regexprep(textData,expre," slash ");
    expre='\s*[Cc]omma[\s.]*';
    textData=regexprep(textData,expre," slash ");
    expre='\<\s*[Ss]lash\s*';
    textData=regexprep(textData,expre," ");

     % Fix the negative part of numbers to the word: minus
```

```
    expre='[\s\W]*-[\s\W]*';    % Any "-" followed by a space or a non-alphanumeric(e.g. comma ",")
    textData=regexprep(textData,expre," minus ");

    % Fix to words:joint
    expre='\<[jJ]o[ih]n';
    textData=regexprep(textData,expre,"joint");

    % Fix to words:joint 1
    expre='\<[jJ]oin\w*\s\<[oO][a-z]\w*';
    % Any word strting from join,followed by space and a word starting from "o" (e.g  John on -> joint
1)
    textData=regexprep(textData,expre,"joint 1");

    % Fix to words:joint 2
    expre='\<[jJ]oin\w*\s\<[tT][owal]\w*';
    % Any word strting from join,followed by space and a word starting from "t"followed by "o" or "w"
or "a" or "l"
    textData=regexprep(textData,expre,"joint 2");

    % Fix to words:joint 3
    expre='\<[jJ]oin\w*\s\<[tT][hree]\w*';
    textData=regexprep(textData,expre,"joint 3");


    % Fix to words:joint 4
    expre='\<[jJ]oin\w*\s\<[fF][oauewr]\w*';
    textData=regexprep(textData,expre,"joint 4");

    % Fix to words:joint 5
    expre='\<[jJ]oin\w*\s\<[fF][iverghtnl]\w*';
    textData=regexprep(textData,expre,"joint 5");

    % Fix to words:joint 6
    expre='\<[jJ]oin\w*\s\<[sS][a-z]\w*';
    textData=regexprep(textData,expre,"joint 6");

    % Fix to decimal: .
    expre='\s*[pP]oint[s]*\s*'; %e.g. 0 point 6 -> 0.6
    textData=regexprep(textData,expre,".");

     expre='([.,][.])';
    textData=regexprep(textData,expre,".");

    % Fix digit-words to numbers
    expre1=["\<zero","\<one","\<two\>","\<three",
"\<four\>","\<five","\<six\>","\<seven\>","\<eight\>","\<nine\>","\<ten","\<eleven",...
        "\<twelve",
"\<thirteen","\<fourteen","\<fifteen","\<sixteen","\<seventeen","\<eighteen","\<nineteen"];

expre2=["0","1","2","3","4","5","6","7","8","9","10","11","12","13","14","15","16","17","18","19"];
    textData=regexprep(textData,expre1,expre2,'ignorecase','all');


     % Fix to word:00 (hundred)
    expre='\s*[Hh]undred\s*(and)*\s*';
    textData=regexprep(textData,expre,"00");


     % Fix to word:000 (thousand)
    expre='\s*[Tt]housand\s*(and)*\s*';
    textData=regexprep(textData,expre,"000");

    textData=splitSentences(textData);

    i=0;
    while 1
        i=i+1;
        if i>length(textData)
            break;
        end

        if i<=0
            i=1;
        end
```

```
% Look for unwanted sentences and delete them accordingly

% If a sentence begins with it, delete current and the previous sentence
[st, ~]=regexp(textData(i),'^Scra\w*','ignorecase');  % e.g. Scratch that.
if i>1 && i<=length(textData) && (~isempty(st) && st==1)
    textData(i)=[];
     textData(i-1)=[];
     i=i-2;
     continue
end


% If it exists in the middle of a sentence, delete current sentence
[st, ~]=regexp(textData(i),'scra\w*','ignorecase');
if i>1 && i<=length(textData) && (~isempty(st) && st>1)
    textData(i)=[];
    i=i-1;
    continue
end

% If a sentence begins with it, delete current and the previous sentence
[st, ~]=regexp(textData(i),'^Dele\w*','ignorecase');  % e.g. Delete that.
if i>1 && i<=length(textData) && (~isempty(st) && st==1)
    textData(i)=[];
    textData(i-1)=[];
    i=i-2;
    continue
end

% If it exists in the middle of a sentence, delete current sentence
[st, ~]=regexp(textData(i),'dele\w*','ignorecase');
if i>1 && i<=length(textData) && (~isempty(st) && st>1)
    textData(i)=[];
    i=i-1;
    continue
end

% If a sentence begins with it, delete current and the previous sentence
[st, ~]=regexp(textData(i),'^Eras\w*','ignorecase');  % e.g. Erase that.
if i>1 && i<=length(textData) && (~isempty(st) && st==1)
    textData(i)=[];
    textData(i-1)=[];
    i=i-2;
    continue
end

% If it exists in the middle of a sentence, delete current sentence
[st, ~]=regexp(textData(i),'eras\w*','ignorecase');
if i>1 && i<=length(textData) && (~isempty(st) && st>1)
    textData(i)=[];
    i=i-1;
    continue
end


%% Fix mistakes during voice recording

% Fix digit followed by a word without space
[st, ~]=regexp(textData(i),'\d{1}[a-zA-Z]+[\s*.]');
if ~isempty(st)
    textData(i)=textData{i}(1:st) + " "+ textData{i}(st+1:end);
end

% Fix word followed by a digit without space
[st, ~]=regexp(textData(i),'[a-zA-Z]+\d[\s*.]');
if ~isempty(st)
    textData(i)=textData{i}(1:st) + " "+ textData{i}(st+1:end);
end

 % Connect to previous any sentence starting from a digit, or 2 words and a digit,
[st, ~]=regexp(textData(i),'^(at|to|[Hh]eight|[Dd]istance)\s*\w*\s*\d+','ignorecase','all');
if i>1 && ~isempty(st) && st==1
    [stt, ~]=regexp(textData(i-1),'[.]$');
```

```
    if ~isempty(stt)
        textData{i-1}=textData{i-1}(1:end-1);
        textData(i-1)=textData(i-1) +" "+ textData(i);  % Add it to the previous
    end
    textData(i)=[]; % Delete current sentence
    i=i-2;          % Go back two index steps,
    continue
end

[st, ~]=regexp(textData(i),'\s*\d+','once');
if i>1 && ~isempty(st) && st==1
    textData(i-1)=textData(i-1) +" "+ textData(i);  % Add it to the previous
    textData(i)=[]; % Delete current sentence
    i=i-2;          % Go back two index steps,
    continue
end

[st, ~]=regexp(textData(i),'\s*(orientation)','once','ignorecase');
if i>1 && ~isempty(st) && st==1
    [stt, ~]=regexp(textData(i-1),'[.]$');
    if ~isempty(stt)
        textData{i-1}=textData{i-1}(1:end-1);
        textData(i-1)=textData(i-1) +" "+ textData(i);  % Add it to the previous
    end
    textData(i)=[]; % Delete current sentence
    i=i-2;          % Go back two index steps,
    continue
end



    % These expressions are to seperate sentences in Pick and Place process
expre='[.,]\s*\w*\s*[pP]osition';    % eg.
[st, ~]=regexpi(textData(i),expre,"start");
if ~isempty(st)
    textData(i+2:end+1)=textData(i+1:end);
    textData{i+1}=textData{i}(st+1:end);
   textData{i}=textData{i}(1:st);
end

expre='[.,]\s*\w*\s*[lL]ocation';    % eg.
[st, ~]=regexpi(textData(i),expre,"start");
if ~isempty(st)
    textData(i+2:end+1)=textData(i+1:end);
    textData{i+1}=textData{i}(st+1:end);
   textData{i}=textData{i}(1:st);
end

 expre='[.,]\s*\w*\s*[hH]eight';    % eg.
[st, ~]=regexpi(textData(i),expre,"start");
if ~isempty(st)
    textData(i+2:end+1)=textData(i+1:end);
    textData{i+1}=textData{i}(st+1:end);
    textData{i}=textData{i}(1:st);
end

expre='[.,]\s*\w*\s*[dD]istance';    % eg.
[st, ~]=regexpi(textData(i),expre,"start");
if ~isempty(st)
    textData(i+2:end+1)=textData(i+1:end);
    textData{i+1}=textData{i}(st+1:end);
    textData{i}=textData{i}(1:st);
end

 expre='[.,]\s*\w*\s*[rR]epet';    % eg.
[st, ~]=regexpi(textData(i),expre,"start");
if ~isempty(st)
    textData(i+2:end+1)=textData(i+1:end);
    textData{i+1}=textData{i}(st+1:end);
    textData{i}=textData{i}(1:st);
end


% More splitting of leftover sentences
```

```
        [st, ~]=regexp(textData(i),'[.]\s*[a-z]','ignorecase','all');
        if i<=length(textData) && ~isempty(st)
            textData(i+2:end+1)=textData(i+1:end);
            textData{i+1}=textData{i}(st+1:end);
            textData{i}= textData{i}(1:st);
            i=i-length(st);
            continue
        end


    end
    textData=strtrim(textData);

end
```

Extract_Numeric_Inputs.m:

```
function [txt,documents,details,numbers]=Extract_Numeric_Inputs(txt,~,details)
    N=length(txt.Tokens);
    Ncols=7;
    numbers=nan(N,Ncols);
    max_words=8;
    indx=nan(N,max_words);

    % Turn numeric strings to numbers, and take care of negative numbers

    for i=1:N       % Loop through every document (sentence)
        str=split(txt.Tokens(i));
        n=length(str);
        k=0;
        for j=1:n   % Loop through every word in current sentence
            ind=str(j)==details.Token;
            part=details.PartOfSpeech(ind);
            cond=(part==categorical("numeral") | details.Type(ind)=="digits");
            if ~cond
                continue
            else

                k=k+1;
                if j>1 && k<Ncols && (str(j-1)=="minus"|| str(j-1)=="negative")
                    numbers(i,k)=-double(str(j));
                    indx(i,j)=1;
                else
                    numbers(i,k)=double(str(j));
                    indx(i,j)=1;
                end

            end
        end

    end

    txt.Tokens=replace(txt.Tokens,"."," ");
    txt.Tokens=replace(txt.Tokens,","," ");
    expre='\d*';
    txt.Tokens=regexprep(txt.Tokens,expre," ");
    txt.Tokens=replace(txt.Tokens,"minus"," ");

    txt.Tokens=replace(txt.Tokens,"equal"," ");

    expre='\s{2,}';
    txt.Tokens=regexprep(txt.Tokens,expre," ");

    documents = tokenizedDocument(txt.Tokens);
    documents = removeEmptyDocuments(documents);

    documents = addPartOfSpeechDetails(documents);
    details = tokenDetails(documents);

    details.DocumentNumber=[];details.SentenceNumber=[];details.LineNumber=[];
    details.Type=[];details.Language=[];
    toks=cellstr(documents.joinWords);
    txt=cell2table(toks,'VariableNames',"Tokens");
    txt.Tokens=string(txt.Tokens);
```

```
end
```

Build_Embedding.m:

```
function [meanEmbedding,documents,sentence2words]=Build_Embedding(documents,emb)

    meanEmbedding = zeros(numel(documents),emb.Dimension);

    whatever_vec=word2vec(emb,"whatever");
    for k=1:numel(documents)
        words = split(string(documents(k)));
        wordVectors = word2vec(emb,words);
        meanEmbedding(k,:) = mean(wordVectors,1);
    end

    if any(isnan(meanEmbedding))
        col=meanEmbedding(:,1);
        ind=find(isnan(col));
        for j=1:length(ind)
            meanEmbedding(ind(j),: )=whatever_vec;
        end
    end

    sentence2words=vec2word(emb,meanEmbedding,'Distance','cosine');

    if isrow(sentence2words)
        sentence2words=sentence2words';
    end
end



function [pred,tab]=Classify(mdl,X,kind)

        K=mdl.KFold;

        N=size(X,1);
        pred=strings(N,1);
        tab=strings(size(X,1),K);

        for j=1:K
        tab(:,j) = predict(mdl.Trained{j},X);
        end

        for i=1:N
        t=tab(i,:);
        num_1=sum(double((strcmpi(t,"T"))));

        if num_1/K>=0.5
        pred(i)="T";
        elseif num_1/K<0.5
        pred(i)="F";
        end

        end

        if isrow(pred)
        pred=pred';
        end
end
```

MyLabelPredictions.m:

```
function commands=MyLabelPredictions(t,command_index_SVM)

        N=length(t);
        commands=strings(N,1);
```

```
        move_voc=["move","translate","shift","stir","relocate","proceed","advance","transfer","transpo
        rt"];

        appro_voc=["approach","come","approximate","reach","near","contact","meet"];


        depart_voc=["depart","pull","step","get","leave","deviate","withdraw","escape","exit","abandon
        ","away"];

        drive_voc=["joint","rotate","revolve","pivot","swing","twist","roll","rotation","spin","gyrate
        ","degree","around"];

        ready_voc=["return","restore","reconfigure","retreat","reinitialize","reset","reboot","revert"
        ,"retire",...
            "initial","starting","original","primary","safe","configuration","home","back"];

        open_voc=["open","expand","free","activate","unlock","drop","let","unretract","unrestrict","re
        lease","unseal"];

        close_voc=["close","secure","fasten","shut","seize","snatch","catch","hold","grab"];

        TopLevel_voc="robot";


        for i=1:N
            txt=t(i);
            reg_close=regexpi(txt,"\s*\<close\s*","all");
            reg_turn=regexpi(txt,"\s*\<turn\s*","all");
            reg_lock=regexpi(txt,"\s*\<lock\s*","all");
            reg_retract=regexpi(txt,"\s*\<retract\s*","all");
            reg_restrict=regexpi(txt,"\s*\<restrict\s*","all");
            reg_seal=regexpi(txt,"\s*\<seal\s*","all");
            reg_go=regexpi(txt,"\s*\<go\s*","all");
            reg_draw=regexpi(txt,"\s*\<draw\s*","all");

            expr_move=regexpi(txt,move_voc);
            expr_appro=regexpi(txt,appro_voc);
            expr_depart=regexpi(txt,depart_voc);
            expr_drive=regexpi(txt,drive_voc);
            expr_ready=regexpi(txt,ready_voc);
            expr_open=regexpi(txt,open_voc);
            expr_close=regexpi(txt,close_voc);
            expr_pre=regexpi(txt,TopLevel_voc);

            if any(expr_pre)
                commands(i)="PreCommand";
                continue
            elseif any([expr_move{:}]) || ...
                    (any(~isempty(reg_go)) && (~any([expr_close{:}]) && ~any([expr_ready{:}]) &&
        ~any([expr_open{:}])))
                commands(i)="MOVE";
            elseif any([expr_appro{:}]) || ((any(~isempty(reg_go)) && any([expr_close{:}]))||
        any(~isempty(reg_draw)))
                commands(i)="APPRO";
            elseif any([expr_depart{:}])
                commands(i)="DEPART";
            elseif any([expr_drive{:}]) || any(~isempty(reg_turn))
                commands(i)="DRIVE";
            elseif any([expr_ready{:}])
                commands(i)="READY";
            elseif any([expr_open{:}])
                commands(i)="OPENI";
            elseif any([expr_close{:}]) || any(~isempty(reg_lock)) || any(~isempty(reg_retract)) ||...
                    any(~isempty(reg_restrict)) || any(~isempty(reg_close)) || any(~isempty(reg_seal))
                commands(i)="CLOSEI";
            else
                commands(i)="";
            end

        end
        commands(~command_index_SVM)="";
        commands=categorical(commands);
end
```

SearchCommandFeatures.m:

```
function[commands,TopLevelFlag,needs_inverse_calc_flag]=SearchCommandFeatures(txt,isCommand,numbers,va
lidateCommands)

    N=length(txt.Tokens);
    Ncols=7;
    commands=strings(N,1);

    needs_inverse_calc_flag=false;
    TopLevelFlag=false;

    if validateCommands
        txt=Verify_Commands(txt,isCommand);
    end

    for i=1:N       % Loop through every document (sentence)
        commands(i)=txt.Class(i);
        number=numbers(i,:);
        ind=find(~isnan(number));

        if ~isempty(ind)
            number=number(ind);
            if length(number)>2
                needs_inverse_calc_flag=true;
            end
        else
            continue
        end

    end

    condition=any(commands=="PreCommand");
    if any(condition)
        TopLevelFlag=true;
    end
end
```

Verify_Commands.m:

```
function txt=Verify_Commands(txt,isCommand)

    N=length(txt.Tokens);
    count=0;                % Count corrections to find accuracy

    for i=1:N

          if ~ismissing(txt.Class(i))      % Found dictionary words in text
            if isCommand(i)             % SVM predicted: True
             ...
            else                        % SVM predicted: False
                txt.Class(i)=" ";
            end
        elseif ismissing(txt.Class(i))   % Not dictionary words found in text
            if isCommand(i)
                txt.Class(i)="False Positive?";
            else
                txt.Class(i)=" ";
            end
          end

        fprintf("\nChecking if command %d is correct\n",i)
        fprintf("\nPress ENTER if command is correct, or press buttons to change:\n")
        fprintf("\n\t1:APPRO, 2:CLOSEI, 3:DEPART,\n\t4:DRIVE, 5:MOVE, 6:OPENI\n\t7:PreCommand,
8:READY\t 0:False Positive\t-1:False Negative\n")

        fprintf("\n\tText input %d: ",i)
        fprintf(txt.Tokens(i))
```

```
        fprintf("\n");
        fprintf("\tCommand %d: ",i)
        if ~ismissing(string(txt.Class(i)))
            fprintf(string(txt.Class(i)))
        end

        if ~isCommand(i)
            fprintf("\t(SVM_False)");
        else
            fprintf("\t(SVM_True)");
        end
        fprintf("\n");


        inpt=str2double(input("\nPlease press the corresponding button to continue: ","s"));

        if isempty(inpt) || isnan(inpt)
            continue
        elseif inpt==1
            txt.Class(i)="APPRO";
            count=count+1;
        elseif inpt==2
            txt.Class(i)="CLOSEI";
            count=count+1;
        elseif inpt==3
            txt.Class(i)="DEPART";
            count=count+1;
        elseif inpt==4
            txt.Class(i)="DRIVE";
            count=count+1;
        elseif inpt==5
            txt.Class(i)="MOVE";
            count=count+1;
        elseif inpt==6
            txt.Class(i)="OPENI";
            count=count+1;
        elseif inpt==7
            txt.Class(i)="PreCommand";
            count=count+1;
        elseif inpt==8
            txt.Class(i)="READY";
            count=count+1;
        elseif inpt==0 && iscommand(i)
            fprintf("\nIt was a False Positive command!\n")
            txt.Class(i)="False Positive";
            count=count+1;
        elseif inpt==-1 && ~iscommand(i)
                    fprintf("\nIt was a False Negative command!\n")
                    tx.Class(i)="False Negative";
                    count=count+1;
        end

    end
        acc=1-count/N;
        if acc==0
            fprintf("\nPerfect!\nPrediction Accuracy 100%%\n")
        else
            fprintf("\nCommand Prediction Accuracy %0.3f%%\n",acc*100)
        end
end
```

My_IK_setup.m:

```
function [numbers_new,angles,location,converged_flg,ik_ind]=My_IK_setup(commands,numbers,robot)

        N=length(commands);

        converged_flg=[];
        ik_ind=[];
        angles=[];
        location=[];

        IK_counts=0;
```

```
numbers=RepairNums(numbers,commands);

numbers_new=numbers;

    for i=1:N       % Loop through every document (sentence)
        number=numbers(i,:);
        move_on=any(~isnan(number));
        status=strings(1);

        if ~move_on
            continue
        end


         if commands(i)~="MOVE" && commands(i)~="APPRO"
            continue;
         end

        if commands(i)=="MOVE"
            position=number(1:6);
            IK_counts=IK_counts+1;

            [joint_angles,status]=Solve_InverseKinematics(robot,position);

            if all(abs(joint_angles-[0 -90 90 0 0 0])<=1e-3)
                location(i,:)=[0, 0, 1.185, 0, 0, 0];
            end

            if status~="success"
                converged_flg=[converged_flg;false];
                fprintf("I.K Model %d NOT CONVERGED!\n",IK_counts)
            else
                converged_flg=[converged_flg;true];
                fprintf("I.K. Model %d CONVERGED!!\n",IK_counts)
            end
                location=[location;position];
                ik_ind=[ik_ind,i];
                angles=[angles;joint_angles];
                numbers(i,1:6)=joint_angles;


        elseif (commands(i)=="APPRO")
            position=number(1:6);
            IK_counts=IK_counts+1;

            [joint_angles,status]=Solve_InverseKinematics(robot,position);

            if all(abs(joint_angles-[0 -90 90 0 0 0])<=1e-6)
                location(i,:)=[0, 0, 1.185, 0, 0, 0];
            end

            if status~="success"
                converged_flg=[converged_flg;false];
                fprintf("I.K Model %d NOT CONVERGED!\n",IK_counts)
            else
                converged_flg=[converged_flg;true];
                fprintf("I.K. Model %d CONVERGED!!\n",IK_counts)
            end
               location=[location;position];
               ik_ind=[ik_ind,i];
               angles=[angles;joint_angles];
               numbers(i,1:6)=joint_angles;
        end

    end

    ik_ind=unique(ik_ind,'stable');
    position=unique(position,'rows','stable');
    angles=unique(angles,'rows','stable');
    numbers_new(ik_ind,1:6)=angles;
end
```

RepairNums.m:

```
function nums_new=RepairNums(numbers,commands)

    N=length(commands);
    nums_new=numbers;

    for i=1:N
        command=commands(i);
        number=numbers(i,:);
        ind=find(~isnan(number));

        if isempty(ind)
            continue
        end

        if command=="APPRO"
            if length(ind)<7
                 number(7)=number(ind(end));
                 number(ind(end):6)=0;
            end

            for j=1:7
                if j<=3 && number(j)>1185
                    number(j)=1185;
                end

                if j>3 && j<7 && number(j)>360
                    kk=floor(number(j)/360);
                    number(j)=number(j)-kk*360;
                elseif j>3 && j<7 && number(j)<-360
                    kk=floor(-number(j)/360);
                    number(j)=number(j)+kk*360;
                end

                if j==7 && number(j)>200
                    number(j)=200;
                end
            end

            nums_new(i,:)=number;
            nums_new(i,1:3)=nums_new(i,1:3)/1000;

        elseif command=="MOVE"
            if length(ind)<6
                number(ind(end)+1:6)=0;
            end

            for j=1:6
                if j<=3 && number(j)>1185
                    number(j)=1185;
                end

                if j>3 && number(j)>360
                    kk=floor(number(j)/360);
                    number(j)=number(j)-kk*360;
                elseif j>3 && number(j)<-360
                    kk=floor(-number(j)/360);
                    number(j)=number(j)+kk*360;
                end
            end
            nums_new(i,:)=number;
            nums_new(i,1:3)=nums_new(i,1:3)/1000;
        end
    end
end
```

Solve_InverseKinematics.m:

```
function [joint_angles,status]=Solve_InverseKinematics(robot,target_location,varargin)

% Handle/Transform Input to Appropriate form
    initial_location=[0,0,1.185 , 0.0000  0.0000  0.0000];

    if isempty(target_location) || all(isnan(target_location))
        target_location=initial_location;
    end

    pos=[target_location(1),target_location(2),target_location(3)];
    eull=[target_location(4),target_location(5),target_location(6)]*pi/180;

    % Define Inverse model
    gik = generalizedInverseKinematics("RigidBodyTree",robot,"ConstraintInputs",...
        {'position','orientation','jointbounds'});

    % Define Position Constrain
    positionTgt = constraintPositionTarget('tool');
    positionTgt.TargetPosition=pos;

    % Joint limits, Common on both I.K models
    jointConst = constraintJointBounds(robot);

    % Define Orientation of tool relative to global
    orientationConst = constraintOrientationTarget('tool','ReferenceBody','base');

    % Turn to quaternion
    Rquad=quaternion(eull,'euler','ZYZ','frame');
    orientationConst.TargetOrientation=Rquad.compact;

    q0 = homeConfiguration(robot);

    [joint_angles,solInfo] = gik(q0,positionTgt,orientationConst,jointConst);

    joint_angles=joint_angles*180/pi;
    joint_angles=round(joint_angles,2);

    if abs(joint_angles(5))<=1e-3
        if joint_angles(6)~=0 || joint_angles(4)~=0
            joint_angles(6)=joint_angles(6)+joint_angles(4);
            joint_angles(4)=0;
        end
        if abs(joint_angles(6))==360 || abs(joint_angles(6))==180
            joint_angles(6)=0;
        end
    end

   for i=4:6
       if abs(abs(joint_angles(i))-360)<=1e-6
           joint_angles(i)=0;
       end
   end

    status=solInfo.Status;
end
```

Extract_file_for_V.m:

```
function Extract_file_for_V(filename,commands,numbers)
    arguments
        filename        (:,1)   string
        commands        (:,1)   string
        numbers         (:,7)   double
    end

    N= length(commands);

    fid=fopen(filename,'w');
```

```
fprintf(fid,".PROGRAM Staubli()\n\n");

fprintf(fid,"AUTO $clear.display;\n");

fprintf(fid,"\nSPEED 100, MMPS ALWAYS;\n");

fprintf(fid,"\nCPON ALWAYS;\n");

fprintf(fid,"\nDURATION 0.2 ALWAYS;\n");

fprintf(fid,"\nPARAMETER HAND.TIME = 0.16;\n");

fprintf(fid,"\nOPENI;\n");

fprintf(fid,"\nTOOL TRANS(0,0,85);\n");

fprintf(fid,"\nENABLE TRACE;\n");

fprintf(fid,"\n;/---------------------------------------------/\n");

fclose(fid);


current_loc=[];

for i=1:N

    command=commands(i);
    argument=numbers(i,:);
    ind=~isnan(argument);
    if any(ind)
        argument=argument(ind);
    end

    if command=="PreCommand" || command=="PRECOMMAND"
        continue
    elseif command=="READY"
        READY(filename)
    elseif command=="OPENI"
        OPENI(filename)
    elseif command=="MOVE"
        [target_loc,~]=Define_Final_Ppoint(command,argument,current_loc);
        if (i>1 && commands(i-1)=="APPRO" && any(target_loc~=ready_loc))
            MOVES(current_loc,target_loc,filename);
        else
            if abs(target_loc-ready_loc)<=1e-6
                READY(filename)
            else
            MOVE(current_loc,target_loc,filename);
            end
        end
        current_loc=target_loc;
    elseif command=="DRIVE"
        DRIVE(argument,filename)
    elseif command=="DEPART"
        if (i>1 && (commands(i-1)=="CLOSEI"))
            DEPARTS(argument,filename);
        else
            DEPART(argument,filename);
        end
    elseif command=="CLOSEI"
        CLOSEI(filename)
    elseif command=="APPRO"
        [target_loc,h]=Define_Final_Ppoint(command,argument,current_loc);
        APPRO(current_loc,target_loc,h,filename);
        current_loc=target_loc;
    else
        continue
    end

end

fid=fopen(filename,'a');
```

```
    fprintf(fid,"\n.END\n");
    fclose(fid);

%open(filename)

cd ..

end


%/--------------------------------------------------------------------/

    function varargout=Define_Final_Ppoint(command,argument,current_loc)
    h0=100;
    if command=="MOVE"
        ind=find(~isnan(argument));
        try
            if any(ind)
                q1=argument(1);
                q2=argument(2);
                q3=argument(3);
                q4=argument(4);
                q5=argument(5);
                q6=argument(6);

                Pf=[q1,q2,q3,q4,q5,q6];
                varargout{1}=Pf;
                varargout{2}=[];
            else
                varargout{1}=current_loc;
                varargout{2}=[];
            end

        catch
            varargout{1}=[];
            varargout{2}=[];

        end

    elseif command=="APPRO"
            ind=~isnan(argument);
            try
                if any(ind)
                    q1=argument(1);
                    q2=argument(2);
                    q3=argument(3);
                    q4=argument(4);
                    q5=argument(5);
                    q6=argument(6);
                    h=argument(7);

                    Pf=[q1,q2,q3,q4,q5,q6];
                    varargout{1}=Pf;
                    varargout{2}=h;

                else
                    varargout{1}=current_loc;
                    varargout{2}=h0;
                end
            catch
                varargout{1}=[];
                varargout{2}=[];

            end
    else
        varargout{1}=current_loc;
        varargout{2}=[];
    end

end


function READY(filename,varargin)
    fid=fopen(filename,'a');
```

```
    fprintf(fid,"\nDO READY;\n");
    fclose(fid);
end


function OPENI(filename)
    fid=fopen(filename,'a');
    fprintf(fid,"\nOPENI;\n");
    fclose(fid);
end


function MOVES(P,Pf,filename)

    fid=fopen(filename,'a');
    if isempty(P) || any(Pf~=P)
        q1=Pf(1);
        q2=Pf(2);
        q3=Pf(3);
        q4=Pf(4);
        q5=Pf(5);
        q6=Pf(6);

        fprintf(fid,"\nPOINT
#loc.target=#PPOINT(%0.3f,%0.3f,%0.3f,%0.3f,%0.3f,%0.3f);\n",q1,q2,q3,q4,q5,q6);
    end
    fprintf(fid,"MOVES loc.target;\n");
    fprintf(fid,"BREAK;\n");

    fclose(fid);
end


function MOVE(P,Pf,filename)

    fid=fopen(filename,'a');
    try
        if isempty(P) || any(Pf~=P)
            q1=Pf(1);
            q2=Pf(2);
            q3=Pf(3);
            q4=Pf(4);
            q5=Pf(5);
            q6=Pf(6);

            fprintf(fid,"\nPOINT
#loc.target=#PPOINT(%0.3f,%0.3f,%0.3f,%0.3f,%0.3f,%0.3f);\n",q1,q2,q3,q4,q5,q6);
        end
            fprintf(fid,"MOVE loc.target;\n");
            fprintf(fid,"BREAK;\n");

             fclose(fid);
    catch
         fclose(fid);
         fprintf("\nNo target was defined.Returning to Home Configuration...\n")
         READY(filename)
    end

end
function DRIVE(argument,filename)
    try
        joint=argument(1);
        ang=argument(2);
    catch
        joint=6;
        ang=45;
    end
    if length(argument)==3
        MonitorSpeed=argument(3);
    else
        MonitorSpeed=50;   % 50%
    end

    fid=fopen(filename,'a');
```

```
    if isempty(argument) || any(isnan(argument)) ||abs(argument(1))>6   % If nargin=0,or joint not
specified: zigzag joint 6 to notify user
        joint=6;
        ang=45;
        fprintf(fid,"\n;Warning for missing data in command:\n");
        fprintf(fid,"\nDRIVE %d, %0.3f, %d;\n",joint,ang,MonitorSpeed);
        fprintf(fid,"BREAK;\n");
        fprintf(fid,"\nDRIVE %d, %0.3f, %d;\n",joint,-ang,MonitorSpeed);
        fprintf(fid,"BREAK;\n");

    else
        fprintf(fid,"\nDRIVE %d, %0.3f, %d;\n",joint,ang,MonitorSpeed);
        fprintf(fid,"BREAK;\n");
        fclose(fid);
    end

end


function DEPART(argument,filename)
    ind=~isnan(argument);
    h=argument(ind);
    if isempty(h)
        h=100;
    end
    fid=fopen(filename,'a');
    fprintf(fid,"\nHERE loc.current;\n");
    fprintf(fid,"DEPART %0.3f;\n",h);
end


function DEPARTS(argument,filename)
    ind=~isnan(argument);
    h=argument(ind);
    if isempty(h)
        h=100;
    end
    fid=fopen(filename,'a');
    fprintf(fid,"\nHERE loc.current;\n");
    fprintf(fid,"DEPARTS %0.3f;\n",h);
end


function CLOSEI(filename)
    fid=fopen(filename,'a');
    fprintf(fid,"\nCLOSEI;\n");
    fclose(fid);
end


function APPRO(P,Pf,h,filename)
    fid=fopen(filename,'a');
    try
        if isempty(P) || any(Pf~=P)
            q1=Pf(1);
            q2=Pf(2);
            q3=Pf(3);
            q4=Pf(4);
            q5=Pf(5);
            q6=Pf(6);
            fprintf(fid,"\nPOINT
#loc.target=#PPOINT(%0.3f,%0.3f,%0.3f,%0.3f,%0.3f,%0.3f);\n",q1,q2,q3,q4,q5,q6);
        end
        fprintf(fid,"\nAPPRO loc.target, %0.3f;\n",h);
        fprintf(fid,"ALIGN;\n");
        fprintf(fid,"BREAK;\n");
        fclose(fid);
    catch
         fclose(fid);
         fprintf("\nNo target was defined.Returning to Home Configuration...\n")
         READY(filename)
    end
end
```

For Pick & Place:

Main_Script_PickNPlace.m:

```
if ~exist('init_needed',"var")
    Initialize;
else
    clc
    fprintf("\nNo need to Initialize\n");
end

diary('myDiaryFile.txt')

PythonSpeech2Text;

if ~interrupt_flg
    %% Preprocess
    clc;
    fprintf("Preprocessing...\n")
    [t0,documents,details] = preprocess(txt);

    %% Extract Numbers and clear them from text input
    [pos,pos_names,h,dist,rep,t]=Extract_Numeric_Inputs_PickNPlace(t0,details);
    fprintf("Done\n")

     %% Solve Inverse Kinematics for desired coordinates
    if all(~isnan(pos))
        fprintf("\nSolving Inverse Kinematics...\n")

        [angles,location,actual_locations,converged_flg]=My_IK_setup_PickNPlace(robot,pos,pos_names);

        varnames=["Q1";"Q2";"Q3";"Q4";"Q5";"Q6"];
        tj=array2table(angles,'RowNames',pos_names,'VariableNames',varnames);
        fprintf("Done\n")

        %% Extract file in V+
        filename="MyStaubliRX90L_Pick_and_Place.txt";
        Extract_file_for_V_PickAndPlace(filename,angles,pos_names,h,dist,rep)

        open MyStaubliRX90L_Pick_and_Place.txt

        fprintf("\n Opening Tera Term...\n")
        system("C:\Program Files (x86)\teraterm\ttermpro.exe")
    else
        fprintf("\nNo known commands found for pick and place programming!\n")
    end
end
```

Extract_Numeric_Inputs_PickNPlace.m:

```
function [pos,pos_names,h,dist,rep,t]=Extract_Numeric_Inputs_PickNPlace(t0,details)
    N=length(t0.Tokens);
    Ncols=7;
    numbers=nan(N,Ncols);
    pos=nan(N,Ncols);
    pos_names=strings(N,1);
    h=nan(N,1);
    dist=nan(N,1);
    rep=nan(N,1);
    t=nan(4,1);

    for i=1:N       % Loop through every document (sentence)
        str=split(t0.Tokens(i));
        n=length(str);
        k=0;
        for j=1:n   % Loop through every word in current sentence
            ind=str(j)==details.Token;
            part=details.PartOfSpeech(ind);
            if part~=categorical("numeral")
                continue
            elseif part==categorical("numeral")
                k=k+1;
```

```
                if j>1 && k<Ncols && (str(j-1)=="minus"|| str(j-1)=="negative")
                    numbers(i,k)=-double(str(j));
                else
                    numbers(i,k)=double(str(j));
                end

            end
        end

    end

    k=0;
    l=0;
    for i=1:N
        number=numbers(i,:);
        textData=t0.Tokens(i);
        ind1=find(~isnan(number));
        if isempty(ind1)
            continue
        elseif ~isempty(ind1)
            k=k+1;
            if contains(textData,"height","IgnoreCase",true)
                h(k)=number(ind1);
                continue
            end
            if length(ind1)>2
                l=l+1;
                if contains(textData,"pick") || contains(textData,"collect") ||
contains(textData,"load")
                    pos_names(l)="pick";
                elseif contains(textData,"place") || contains(textData,"placing") ||
contains(textData,"unload")
                    pos_names(l)="place";
                elseif contains(textData,"alpha") || contains(textData,"safe out")
                    pos_names(l)="safe out";
                elseif contains(textData," bet ") || contains(textData,"beta") ||
contains(textData,"safe in")
                    pos_names(l)="safe in";
                end
                pos(l,ind1)=number(ind1);
                if length(pos(l,ind1))<6
                    pos(l,ind1(end)+1:6)=0;
                elseif length(pos(l,ind1))==Ncols
                    h(k)=pos(l,Ncols);
                    pos(l,Ncols)=nan;
                end
                continue
            end

            if contains(textData,"distance","IgnoreCase",true)
                dist(k)=number(ind1(1));
                continue
            end

            if contains(textData,"repetition","IgnoreCase",true) ||...
                    contains(textData,"repeat","IgnoreCase",true)
                rep(k:k+1)=number(ind1);
                continue
            end
        end
    end

     keep=~strncmp(pos_names,"",1);
    pos_names=pos_names(keep);
    if ~isempty(pos_names)
        pos_names=reshape(pos_names,[],1);
    end

    pos=pos(keep,1:6);
    if ~isempty(pos)
        pos=reshape(pos,[],6);
        pos(:,1:3)=pos(:,1:3)/1000;
    end
```

```
    dist=dist(~isnan(dist));
    rep=rep(~isnan(rep));
    h=h(~isnan(h));


    if isempty(h) && all(~isempty(pos))
        h=100;
    elseif length(h)>1
        h=min(max(h),200);
    elseif h<=0
        h=50;
    end

    if isempty(dist)
        dist=[50;50];
    elseif length(dist)==1
        dist(2)=dist(1);
    end

    if isempty(rep)
        rep=[1;1];
    elseif length(rep)==1
        rep(2)=rep(1);
    end

    if ~isempty(pos) && ~isempty(pos_names)
        varnames=["X";"Y";"Z";"yaw";"pitch";"roll"];
        t=array2table(pos,'RowNames',pos_names,'VariableNames',varnames);
    end

end
```

My_IK_setup_PickNPlace.m:

```
function [angles,location,actual_pos,converged_flg]=My_IK_setup_PickNPlace(robot,numbers,pos_names)

    N=size(numbers,1);

    converged_flg=[];
    IK_counts=0;
    angles=[];
    location=[];
    k=0;

    for i=1:N       % Loop through every document (sentence)
        number=numbers(i,:);
        status=strings(1);
        pos_name=pos_names(i);
        poss=number(1:6);
        IK_counts=0;

        if strcmpi(pos_name,"pick")
            loc_ind.pick=i;
            ind=strcmpi(pos_names,"safe out");
            current_pos=numbers(ind,1:6);

        elseif strcmpi(pos_name,"place")
            loc_ind.place=i;
            ind=strcmpi(pos_names,"safe in");
            current_pos=numbers(ind,1:6);

        elseif strcmpi(pos_name,"safe out")
            loc_ind.safe_out=i;
            ind=strcmpi(pos_names,"pick");
            current_pos=numbers(ind,1:6);

        elseif strcmpi(pos_name,"safe in")
            loc_ind.safe_in=i;
            ind=strcmpi(pos_names,"place");
            current_pos=numbers(ind,1:6);
```

```
        end

        [joint_angles,status]=Solve_InverseKinematics(robot,poss,current_pos);

        if status~="success"
            IK_counts=IK_counts+1;
            fprintf("I.K. Model "+ num2str(IK_counts) +" NOT CONVERGED!!")
        else
            converged_flg=[converged_flg;true];
            IK_counts=1;
            k=k+1;
            fprintf("\nI.K. Model %d CONVERGED!!\n",k)
        end

        location=[location;poss];
        angles=[angles;joint_angles];

    end

    location=unique(location,'rows','stable');
    angles=unique(angles,'rows','stable');
    actual_pos=nan(4,6);


    pick_actual=tform2trvec(getTransform(robot,angles(loc_ind.pick,:)*pi/180,'tool','base'));
    R=tform2rotm(getTransform(robot,angles(loc_ind.pick,:)*pi/180,'tool','base'));
    rq=quaternion(R,'rotmat','point');
    eull_pick=euler(rq,'ZYZ','frame')*180/pi;
    actual_pos(loc_ind.pick,:)=[pick_actual,eull_pick];

    place_actual=tform2trvec(getTransform(robot,angles(loc_ind.place,:)*pi/180,'tool','base'));
    R=tform2rotm(getTransform(robot,angles(loc_ind.place,:)*pi/180,'tool','base'));
    rq=quaternion(R,'rotmat','point');
    eull_place=euler(rq,'ZYZ','frame')*180/pi;
    actual_pos(loc_ind.place,:)=[place_actual,eull_place];

    safe_out_actual=tform2trvec(getTransform(robot,angles(loc_ind.safe_out,:)*pi/180,'tool','base'));
    R=tform2rotm(getTransform(robot,angles(loc_ind.safe_out,:)*pi/180,'tool','base'));
    rq=quaternion(R,'rotmat','point');
    eull_safe_out=euler(rq,'ZYZ','frame')*180/pi;
    actual_pos(loc_ind.safe_out,:)=[safe_out_actual,eull_safe_out];

    safe_in_actual=tform2trvec(getTransform(robot,angles(loc_ind.safe_in,:)*pi/180,'tool','base'));
    R=tform2rotm(getTransform(robot,angles(loc_ind.safe_in,:)*pi/180,'tool','base'));
    rq=quaternion(R,'rotmat','point');
    eull_safe_in=euler(rq,'ZYZ','frame')*180/pi;
    actual_pos(loc_ind.safe_in,:)=[safe_in_actual,eull_safe_in];
end



function
Extract_file_for_V_PickAndPlace(filename,configurations,pos_names,height,distance,repetitions)

    if isempty(configurations)
        fprintf("\nNo locations were given!\n")
        return
    end

    conf_pick=configurations(strcmpi(pos_names,"pick"),:);
    conf_place=configurations(strcmpi(pos_names,"place"),:);
    conf_safein=configurations(strcmpi(pos_names,"safe in"),:);
    conf_safeout=configurations(strcmpi(pos_names,"safe out"),:);

    fid=fopen(filename,'w');
        fprintf(fid,".PROGRAM Pick_and_Place()\n\n");
        fprintf(fid,"AUTO $clear.display;\n");
        fprintf(fid,"\nSPEED 100, MMPS ALWAYS;\n");
        fprintf(fid,"\nCPON ALWAYS;\n");
        fprintf(fid,"\nDURATION 0.2 ALWAYS;\n");
        fprintf(fid,"\nPARAMETER HAND.TIME = 0.16;\n");
        fprintf(fid,"\nOPENI;\n");
        fprintf(fid,"\nTOOL TRANS(0,0,85);\n");
```

```
        fprintf(fid,"\nENABLE TRACE;\n");
        fprintf(fid,"\n;/-----------------------------------------------/\n");

    fclose(fid);

    fid=fopen(filename,'a');
        fprintf(fid,"\nSET #loc.pick=#PPOINT(%0.3f,%0.3f,%0.3f,%0.3f,%0.3f,%0.3f);\n",conf_pick);
        fprintf(fid,"\nSET #loc.place=#PPOINT(%0.3f,%0.3f,%0.3f,%0.3f,%0.3f,%0.3f);\n",conf_place);
        fprintf(fid,"\nSET #loc.safein=#PPOINT(%0.3f,%0.3f,%0.3f,%0.3f,%0.3f,%0.3f);\n",conf_safein);
        fprintf(fid,"\nSET
#loc.safeout=#PPOINT(%0.3f,%0.3f,%0.3f,%0.3f,%0.3f,%0.3f);\n",conf_safeout);

        fprintf(fid,"\nMOVE loc.safeout;\n");

        if any(repetitions>1)
            fprintf(fid,"\nFOR i=0 TO %d\n",repetitions(1)-1);
            fprintf(fid,"\tFOR j=0 TO %d\n",repetitions(2)-1);
        else
            fprintf(fid,"i=0\n");
            fprintf(fid,"j=0\n");
        end
        fprintf(fid,"\t\tSPEED 90;\n");
        fprintf(fid,"\t\tAPPRO loc.pick:TRANS(i*%d,j*%d),%d;\n",distance(1),distance(2),height);
        fprintf(fid,"\t\tALIGN;\n");
        fprintf(fid,"\t\tSPEED 20;\n");
        fprintf(fid,"\t\tMOVES loc.pick:TRANS(i*%d,j*%d);\n",distance(1),distance(2));
        fprintf(fid,"\t\tBREAK;\n");
        fprintf(fid,"\t\tCLOSEI;\n");
        fprintf(fid,"\t\tDEPARTS %d;\n",height);
        fprintf(fid,"\t\tSPEED 90;\n");
        fprintf(fid,"\t\tMOVE loc.safeout;\n");
        fprintf(fid,"\t\tBREAK;\n");
        fprintf(fid,"\t\tMOVE loc.safein;\n");
        fprintf(fid,"\t\tAPPRO loc.place:TRANS(i*%d,j*%d),%d;\n",distance(1),distance(2),height);
        fprintf(fid,"\t\tALIGN;\n");
        fprintf(fid,"\t\tSPEED 20;\n");
        fprintf(fid,"\t\tMOVES loc.place:TRANS(i*%d,j*%d);\n",distance(1),distance(2));
        fprintf(fid,"\t\tBREAK;\n");
        fprintf(fid,"\t\tOPENI;\n");
        fprintf(fid,"\t\tDEPARTS %d;\n",height);
        fprintf(fid,"\t\tSPEED 90;\n");
        fprintf(fid,"\t\tMOVE loc.safein;\n");
        fprintf(fid,"\t\tMOVE loc.safeout;\n");
        fprintf(fid,"\t\tBREAK;\n");
        if any(repetitions>1)
            fprintf(fid,"\tEND\n");
            fprintf(fid,"END\n");
        end
        fprintf(fid,"\nSPEED 90;\n");
        fprintf(fid,"\nMOVE loc.safeout;\n");
        fprintf(fid,".END");

    fclose(fid);

end
```

Παράρτημα Β

Δεδομένα εκπαίδευσης SVM:

Tokens	**Class**
Move to target location 146 slash 479 slash 513 slash 10 slash 180 slash 90.	T
Go to the specified position 204 slash 608 slash 953 slash 90 slash 0 slash 45.	T
Translate to location minus 300 slash 400 slash 500 slash 0 slash 180 slash 0.	T
Advance to desired point 342 slash 219 slash 754 slash 0 slash 90 slash 0.	T
Stir to predefined region 320 slash 942 slash 825 slash 43 slash 29 slash 83.	T
Relocate at coordinates 400 slash 600 slash 950 slash 0 slash 0 slash 0.	T
Transport to the specified coordinates.	T
Shift to the final location.	T
Proceed to the desired point.	T
Transfer to position 326 slash 234 slash 785 slash 0 slash 0 slash 0.	T
Joint 2, rotate 67 degrees.	T
Joint 4, roll minus 29 degrees.	T
Joint 1, turn around 81 degrees.	T
Joint 5 revolve 32 degrees.	T
Joint 6 swing 90 degrees.	T
Joint 3 gyrate minus 77 degrees.	T
Joint 6 pivot minus 120 degrees.	T
Joint 6 turn around 45 degrees.	T
Joint 1, do a full rotation.	T
Joint 2, spin for 11 degrees.	T
Approach the location 400 slash 600 slash 950 slash 0 slash 0 slash 0 to distance 50 millimeters.	T
Come closer to position 300 slash 400 slash 550 slash 0 slash 180 slash 0 at distance 25 above.	T
Come near your target at 32 below.	T
Draw closer to location 245 slash 316 slash 863 slash 0 slash 180 slash 0 at distance 55 mm.	T
Approximate the item at distance 25 mm.	T
Meet your target position at height 44 mm.	T
Approach the target at heigh 22 mm.	T
Gain contact to the target 334 slash 215 slash 657 slash 0 slash 0 slash 0 at heigh 23 mm.	T
Reach to location 100 below your target.	T
Draw towards coordinates 500 slash 300 slash 550 slash 0 slash 180 slash 0 slash at height minus 100 mm.	T
Pull out from current location at height 25 mm.	T
Step away from current position at height 55 mm.	T
Depart at height 44 mm.	T
Leave target at height 12 mm.	T
Get away from ongoing location at height 33 mm.	T
Exit from ongoing coordinates at distance 55 mm.	T

Abandon target at height 60 mm.	T
Escape to distance minus 44 mm.	T
Deviate from position of the target at distance 33 mm.	T
Withdraw from your point to distance minus 64 mm.	T
Return to initial location.	T
Restore your starting status.	T
Reconfigure to your primary state.	T
Retreat to safe position.	T
Go back to original configuration.	T
Reinitialize.	T
Reset.	T
Revert to primal configuration.	T
Reboot.	T
Revert to safe configuration.	T
Expand your grip.	T
Free your gripper.	T
Open the gripper.	T
Activate the gripper coil.	T
Let go of the object.	T
Unlock grip.	T
Drop the item.	T
Unretract the tool tip.	T
Release the object.	T
Unseal gripper.	T
Close the gripper.	T
Catch the item.	T
Fasten grip.	T
Lock gripper.	T
Secure the gripper.	T
Restrict the tool tip.	T
Grab the object.	T
Hold item.	T
Seize object.	T
Snatch item.	T
Ok robot	T
Hey Robot.	T
Hi Robot.	T
Yo robot.	T
Hello robot.	T
Morning Robot	T
Good morning Robot.	T
Come on Robot.	T
Good evening Robot.	T
Robot.	T
I am freezing, it is minus 5 degrees.	F

Do you ride a bike?	F
What is the weather like today?	F
I could really use a nap.	F
Whatever.	F
I rollerblade a lot.	F
Do not watch too much TV.	F
I can extract features from speech.	F
Do not ever hit anybody.	F
I will learn more and more about AI and NLP.	F
Can you grab a plate?	F
Did you see the match yesterday in Athens?	F
Where is the dog?	F
Predicting your behavior is a tricky business.	F
I will make you the king of robots.	F
The Japanese yen for commerce is still well known.	F
Little John decided to wear orange today.	F
Tomatoes make great weapons when water balloons are not available.	F
Today we gathered moss for my uncle.	F
He wondered if it could be called a beach if there was no sand.	F
My joint is broken.	F
I cannot turn my head around.	F
The temperature is 54 degrees.	F
Move your legs soldier.	F
He appeared to be confusingly perplexed.	F
I think about going to Africa	F
He ran out of money, so he had to stop playing poker.	F
Charles ate the French fries knowing they would be his last meal.	F
Bill ran from the giraffe toward the dolphin.	F
I was fishing for compliments and accidentally caught a trout.	F
I covered my girl in baby oil.	F
I am counting my calories, yet I really want dessert.	F
Open up, I want to come in.	F
He was all business when he wore his clown suit.	F
Tom got a small piece of pie.	F
25 years later, she still regretted that specific moment.	F
He moved to another apartment.	F
When are you coming?	F
We are getting closer to victory.	F
Let me in, open the door.	F
Remember your place.	F
Whatever.	F
Close the door, I want to sleep.	F
Whatever you need.	F
Combines are no longer just for farms.	F
Do not put peanut butter on your nose.	F

Joe made the sugar cookies.	F
Do you like politics?	F
I am a living furnace.	F
Come on, give me a break.	F
The bees decided to have a mutiny against their queen.	F
What a beautiful back flip.	F
Eating eggs on Thursday for choir practice was recommended.	F
The eye of the tiger is watching.	F
You drive me nuts.	F
My joints are in pain.	F
At that moment she realized she had a sixth sense.	F
It is a skateboarding penguin with a sunhat!	F
Wake up, you must open your eyes.	F
He decided to fake his disappearance to avoid jail.	F
Fasten your seatbelts please.	F
Grab this milk for me please.	F
Hold on a second.	F
My computer cannot reboot.	F
The criminal escaped the scene.	F
We deviated from our initial plans.	F
Leave the rest to my robot.	F
The enemy retreated eventually.	F
Do you want to play catch?	F
He came close to losing it.	F
They could not secure the bomb.	F
Why are terrorists free again?	F
Please, translate this text in German for me.	F
I want to pay in advance.	F
As he looked out the window, he saw a clown walk by.	F
So, we are back to square zero.	F
Turn on the TV please.	F
It caught him off guard.	F
People generally approve of dogs eating cat food.	F
She folded her skirt neatly.	F

Δεδομένα επαλήθευσης SVM:

Tokens	**Class**
Move to location 1.	T
Stir to location 1.	T
Relocate at position 5.	T
Transfer to location 3.	T

Advance to coordinates 3/4/5.	T
Proceed to location 3.	T
Shift to target location.	T
Rotate joint 3/minus 45 degrees.	T
Revolve joint 4.	T
Spin 45 degrees	T
Pivot joint 2/33 degrees.	T
Twist joint 1 for 56 degrees.	T
Swing joint 6 / minus 12.	T
Roll 45 degrees on joint 6.	T
Approach position 1 at height 22 mm.	T
Come close to location 1.	T
Approximate position 2.	T
Reach coordinates 3/4/5.	T
Meet location 2 at distance 33 mm.	T
Go towards location 2 at distance 44.	T
Gain contact with coordinates 3/4/5 at 66 mm.	T
Depart from position 1 at height 22 mm.	T
Pull out from location 1.	T
Step away from position 2 to distance 11 mm.	T
Leave coordinates 3/4/5 to 25.	T
Withdraw from location 2 at distance 33 mm.	T
Abandon location 2 at distance 44.	T
Escape coordinates 3/4/5 at 66 mm.	T
Go back to origin.	T
Return to original configuration.	T
Reinitialize.	T
Reset.	T
Restore initial configuration.	T
Revert to safety.	T
Reconfigure to primal position.	T
Close your gripper.	T
Grab the item.	T
Hold the item.	T
Catch object.	T
Snatch item.	T
Seize item.	T
Secure the object.	T
Open your gripper.	T
Free the item.	T
Drop the item.	T
Let go of object.	T
Release object.	T
Unlock gripper.	T
Unseal the object.	T

Ok robot.	T
What a lovely day!	F
I look like a robot.	F
My joint is broken.	F
I cannot turn my head around.	F
The temperature is 54 degrees.	F
Move your legs soldier.	F
Come on, give me a break.	F
Go to Africa for a month.	F
We are getting closer to victory.	F
Let me in, open the door.	F
Remember your place.	F
Whatever.	F
Please return the money you took.	F
Fasten your seatbelts please.	F
Grab this milk for me please.	F
Hold on a second.	F
My computer cannot reboot.	F
The criminal escaped the scene.	F
We deviated from our initial plans.	F
Leave the rest to my robot.	F
The enemy retreated eventually.	F
Do you want to play catch?	F
He came close to losing it.	F
They could not secure the bomb.	F
Why are terrorists free again?	F
Please, translate this text in German for me.	F
I want to pay in advance.	F
Where is the company's new department?	F
The old man had no choice but to retire.	F
Oh, what a plot twist they got there!	F
Roll the dice and see what happens.	F
Did the drone reach its target?	F
I need to transfer my money elsewhere.	F
Let us pray to our sweet Lord Jesus!	F
Drop the attitude, it is not helping.	F
The car did a whole rotation in the air.	F
Do you know what the spin of a particle is?	F
How can someone abandon his kids?	F
I cannot get enough of this meal.	F
That player really stepped his game up.	F
That prisoner should be released.	F
Someone once said to seize the moment.	F
Close your eyes and sleep.	F
They were safe when the explosion took place.	F

We lost contact with our coordinators.	F
The company expanded dramatically.	F
Please, take the night shift for me.	F
Public transport is quite safe nowadays.	F
Do you go bowling?	F
I approximated my peak.	F