ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΣΗΜΑΤΩΝ, ΕΛΕΓΧΟΥ ΚΑΙ ΡΟΜΠΟΤΙΚΗΣ

Διαδραστικός Προσαρμοστικός Έλεγχος Ρομποτικού Τροχήλατου Περιπατητήρα Κινητικής Υποβοήθησης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Δημήτριος Ρακοβίτης

Επιβλέπων : Κωνσταντίνος Τζαφέστας
Αν. Καθηγητής ΕΜΠ

Αθήνα, Οκτώβριος 2021

NATIONAL TECHNICAL UNIVERSITY OF ATHENS
SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING
DIVISION OF SIGNALS, CONTROL AND ROBOTICS

Interactive Adaptive Admittance Control of an Assistive Robotic Rollator for Mobility Assistance

DIPLOMA THESIS

Dimitrios Rakovitis

Supervisor : Constantinos Tzafestas
Assoc. Professor NTUA

Athens, October 2021

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΣΗΜΑΤΩΝ, ΕΛΕΓΧΟΥ ΚΑΙ ΡΟΜΠΟΤΙΚΗΣ

Διαδραστικός Προσαρμοστικός Έλεγχος Ρομποτικού Τροχήλατου Περιπατητήρα Κινητικής Υποβοήθησης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Δημήτριος Ρακοβίτης

Επιβλέπων : Κωνσταντίνος Τζαφέστας
Αν. Καθηγητής ΕΜΠ

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 3η Νοεμβρίου 2021

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Κωνσταντίνος Τζαφέστας	Πέτρος Μαραγκός	Χαράλαμπος Ψυλλάκης
Αν. Καθηγητής ΕΜΠ	Καθηγητής ΕΜΠ	Λέκτορας ΕΜΠ

Αθήνα, Οκτώβριος 2021

(Υπογραφή)

.....
Δημήτριος Ρακοβίτης

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Abstract

This diploma thesis deals with the subject of control of a robotic rollator in physical interaction with the human-user. The project focuses on two research directions. The first concerns the design and training of an Admittance model to drive a smart walker with Force/Torque sensors mounted on its handles. In the second direction, a novel intention-based navigation Controller is proposed that connects the Admittance model with a dynamic local path planner. This shared human-robot control strategy is used to implement an Assistive Mode for driving, while also being extended with an Intention Recognition algorithm for undecidability resolution and a Damping Adaptation technique. The assistance is about guiding the user with safety on dangerous paths, while the undecidability refers to picking the human intented direction of motion on situations where there are multiple possible ways of movement, like in a T-junction. On top of these, the adaptation of the damping coefficient of the Admittance model is proposed, based on the state of the system, to make the human-walker collaboratation more realistic and useful. To evaluate the proposed methods, various experimental results are presented, some for Free Space and some for Constrained Space. In Free Space the driving of the robot is tested with the estimated Admittance model and with adjusted damping, while in Constrained Space specific case scenarios are considered to show the effectiveness of the implemented navigation Controller, methods, and assistive features.

Keywords: Human-Robot Interaction, Assistive Robot, Smart Walker, i-Walk, Admittance Control, Learning from Demonstration, Dynamic Window Motion Planning, Arc-Line Trajectory Generation, Intention Recognition, Adaptive Damping

Περίληψη

Αυτή η διπλωματική εργασία πραγματεύεται το θέμα του ελέγχου ενός ρομποτικού περιπατητήρα σε φυσική αλληλεπίδραση με τον άνθρωπο-χρήστη. Η εργασία εστιάζει σε δύο ερευνητικές κατευθύνσεις. Η πρώτη αφορά την σχεδίαση και την εκπαίδευση ενός μοντέλου Ενεργούς Συμμόρφωσης (Admittance) με σκοπό την οδήγηση ενός ευφυούς περιπατητήρα μέσω αισθητήρων Δύναμης/Ροπής εγκατεστημένων στις λαβές του. Στην δεύτερη ερευνητική κατεύθυνση, προτείνεται ένας νέος Ελεγκτής πλοήγησης που βασίζεται στην πρόθεση του χρήστη και συνδέει το μοντέλο Admittance με έναν δυναμικό τοπικό σχεδιαστή μονοπατιών. Αυτή η από κοινού στρατηγική ελέγχου ανθρώπου-ρομπότ χρησιμοποιείται για την υλοποίηση μιας Λειτουργίας Υποβοήθησης κατά την οδήγηση, ενώ επίσης επεκτείνεται με έναν αλγόριθμο Εντοπισμού Πρόθεσης για την επίλυση της αβεβαιότητας ως προς τη λήψη απόφασης πλοήγησης, καθώς και μία τεχνική Προσαρμοστικής Απόσβεσης. Η βοήθεια που προσφέρεται σχετίζεται με την ασφαλή καθοδήγηση του χρήστη σε επικίνδυνα μονοπάτια, ενώ η αβεβαιότητα ως προς τη λήψη απόφασης πλοήγησης αναφέρεται στην επιλογή της κατεύθυνσης που επιθυμεί ο χρήστης σε περιπτώσεις όπου υπάρχουν πολλαπλοί τρόποι κίνησης, όπως σε κάποια διασταύρωση. Επιπλέον, προτείνεται η προσαρμοστικότητα της παραμέτρου απόσβεσης του μοντέλου Admittance, με βάση την κατάσταση του συστήματος, ώστε να καταστεί η συνεργασία ανθρώπου-περιπατητήρα πιο ρεαλιστική και εύχρηστη. Για να αξιολογηθούν οι προτεινόμενες μέθοδοι παρουσιάζονται ποικίλα πειραματικά αποτελέσματα, κάποια σε Ελεύθερο Χώρο και κάποια σε Περιορισμένο Χώρο παρουσία εμποδίων. Στον Ελεύθερο Χώρο η οδήγηση του ρομπότ δοκιμάζεται με το εκτιμώμενο μοντέλο Admittance και μετά από ρύθμιση της παραμέτρου απόσβεσης, ενώ στον Περιορισμένο Χώρο επιλέγονται συγκεκριμένα σενάρια για να φανεί η αποτελεσματικότητα του υλοποιημένου Ελεγκτή πλοήγησης, των μεθόδων και των βοηθητικών λειτουργιών.

Λέξεις-Κλειδιά: Αλληλεπίδραση Ανθρώπου-Ρομπότ, Βοηθητικό Ρομπότ, Έξυπνος Περιπατητήρας, i-Walk, Έλεγχος Ενεργούς Συμμόρφωσης, Μάθηση μέσω Επίδειξης, Ρομποτικός Σχεδιασμός Κίνησης βάσει Δυναμικού Παραθύρου, Παραγωγή Τροχιάς Τόξου-Ευθείας, Αναγνώριση Πρόθεσης, Προσαρμοστική Απόσβεση

Ευχαριστίες

Θα ήθελα αρχικά να ευχαριστήσω θερμά τον καθηγήτη μου Κωνσταντίνο Τζαφέστα για την ευκαιρία που μου έδωσε να δουλέψω στο εργαστήριο ρομποτικής, για όλες τις συμβουλές και την καθοδήγηση που μου προσέφερε και την επικοινωνία που υπήρχε καθ'όλη την διάρκεια της εκπόνησης της διπλωματικής μου εργασίας. Επίσης, θα ήθελα να πω ένα τεράστιο ευχαριστώ στον υποψήφιο διδάκτορα Αθανάσιο Δομέτιο, για όλες τις ερωτήσεις που μου απάντησε, για όλες τις συζητήσεις που κάναμε, για όλη την συνεισφορά του στο μηχανικό κομμάτι της εργασίας, για την διαθεσιμότητά του, για την καθοδήγηση και τις συμβουλές που μου έδωσε και για το πρότυπο που αποτέλεσε. Επιπλέον, ευχαριστώ όλους τους συναδέλφους με τους οποίους συνεργαστήκαμε και συζητήσαμε.

Εν συνεχεία, να ευχαριστήσω απείρως όλα τα μέλη της οικογένειάς μου για την στήριξη που μου έχουν προσφέρει και την πίστη τους σε έμενα καθ'όλη την διάρκεια της ζωής μου. Ευχαριστώ και είμαι ευγνώμων για τα θεμέλια και την αγάπη που μου έδωσαν, ώστε να μπορώ να είμαι εδώ σημέρα και να γράφω αυτή την διπλωματική εργασία.

Τέλος, ευχαριστώ από καρδιάς τους παιδικούς μου φίλους, τους φοιτητικούς μου φίλους, όλους όσους με θεωρούν φίλο τους και όλους όσους περάσαμε καλές και κακές στιγμές μαζί, καθώς μέσα από αυτούς τους ανθρώπους έχω εξελιχθεί και έχει καθοριστεί η ταυτότητα που έχω σήμερα.

Contents

List of Figures

List of Tables

Chapter 1

Εκτεταμένη Περίληψη

1.1 Υπόβαθρο & Συναφής Έρευνα

Οι άνθρωποι και τα ρομπότ είναι δύο πολύ διαφορετικές οντότητες, η κάθε μία με πλεονεκτήματα και μειωνεκτήματα, όμως αν δουλέψουν μαζί μπορούν να πετύχουν μοναδικά αποτελέσματα. Τέτοια σενάρια είναι εμφανή στο βιομηχανικό περιβάλλον σε εργασίες όπως, μετακίνηση βαριών αντικειμένων [1], τρύπημα/βίδωμα [4], γυάλισμα [5] κλπ, όπου το ρομπότ κάνει τις βαριές δουλειές και ο άνθρωπος το κατευθύνει, ή σε εργασίες συναρμολόγησης όπου ο άνθρωπος κάνει την δύσκολη δουλειά και το ρομπότ του δίνει τα αντικείμενα που χρειάζεται [2, 3]. Σημαντική είναι επίσης η επίδραση στο ιατρικό περιβάλλον, όπως για παράδειγμα στον τομέα της ρομποτικής χειρουργικής [6, 34] όπου η πιθανότητα πλέον για σφάλματα ελαχιστοποιείται χάρη στην ακρίβεια του ρομποτικού χειριστή, ή στον τομέα τον προσθετικών ρομποτικών μελών [7, 8] που ξαναδίνουν ελπίδα.

Τα τελευταία χρόνια έχει αρχίσει να αυξάνεται η παραγωγή ρομποτικών οχημάτων, όπως αναπηρικές καρέκλες [9], περιπατητήρες [10], σκούτερ [33] κλπ, που βοηθάνε ανθρώπους με κινητικά προβλήματα να πλοηγηθούν στο χωρό. Τα ρομπότ αυτά ανήκουν στην ευρύτερη κατηγορία των Βοηθητικών Ρομπότ. Στην ίδια κατηγορία αλλά λίγο διαφορετικά είναι τα ρομπότ συντροφιάς που μπορούν να εκτελούν εργασίες ελέγχου του χώρου και ελέγχονται φωνητικά ή μέσω GUI [12], ή ρομπότ που στοχεύουν στην βελτίωση της ζωής ανθρώπων συγκεκριμένων κατηγοριών, όπως για παράδειγμα τους τυφλούς [11]. Αν και οι εφαρμογές είναι άπειρες, η εργασία αυτή εστιάζει στην κατηγορία των τροχηλατών περιπατητήρων.

Οι έξυπνοι περιπατητήρες είναι κινητά ρομπότ με λαβές που βοηθούν ανθρώπους με κινητικά προβλήματα να περπατήσουν με ασφάλεια. Για να είναι αποτελεσματική μια εφαρμογή βοήθειας βάδισης θα πρέπει το ρομπότ να μπορεί να προβλέψει την *ανθρώπινη πρόθεση*, να *σχεδιάζει εφικτά μονοπάτια* (path planning), και να μπορεί να τα *ακολουθεί* (path following). Το πρώτο μπορεί να γίνει μέσω ελέγχου Ενεργούς Συμμόρφωσης (Admittance) [13], κάμερα απέναντι από τον χρήστη [14], αισθητήρες κλίσης στις λαβές [15] κλπ. Τα επόμενα δύο χρειάζονται για να μπορεί να γίνει μια βέλτιστη και ασφαλής πλοήγηση στο χώρο αποφεύγοντας πιθανές συγκρούσεις και δύσκολα μονοπάτια. Σχετικές δουλειές θα δοθούν στην επόμενη παράγραφο. Πέρα από βοήθεια βάδισης, πιο πολυπλοκές εφαρμογές έχουν υλοποιηθεί, όπως αποφυγή πτώσης με έλεγχο του κέντρου πίεσης, με την βοήθεια μιας φορετής σόλας αισθητήρα φορτίου και λέιζερ που εντοπίζουν την θέση των ποδιών, σε ένα παντοκατευθυντικό περιπατητήρα [16] ή του κέντρου μάζας σε έναν περιπατητήρα με λαβές 2-ΒΕ [17]. Το δεύτερο έχει επίσης χρησιμοποιηθεί για μια εφαρμογή καθιστός-σε-όρθιος [27, 18]. Σε άλλα σενάρια, για να μειωθεί ο ανθρώπινος φόρτος το ρομπότ επιταχύνει/επιβραδύνει σε ανιφοροκατιφόρες [29,30], ή ασκεί βοηθητική ροπή όταν ο άνθρωπος είναι κουρασμένος [19]. Για συγκεκριμένες κατηγορίες

χρηστών, όπως τυφλούς, ιδιαίτεροι αισθητήρες μπορούν να χρησιμοποιηθούν για την πλοήγηση, όπως φορετή δονούμενη ζώνη και δονούμενες λαβές [31], ενώ για να μειωθεί ο φόρτος ανθρώπων με νοητικά προβλήματα, φωνητικές εντολές και οπτικά μενού θα μπορούσαν να χρησιμοποιηθούν [28]. Όλες οι παραπάνω είναι μερικές από την πληθώρα εφαρμογών που θα μπορούσε κανείς να υλοποιήσει, ωστόσο στην ερασία αυτή θα χρησιμοποιηθεί το Βοηθητικό Ρομπότ i-Walk [32], για την κατασκευή ενός μοντέλου Admittance και χρήση του σε μια μέθοδο τοπικής πλοήγησης.

Για να μπορεί ένα ρομπότ να πλοηγηθεί στον χώρο πρέπει να υπάρχει ένας χάρτης του περιβάλλοντος, να μπορεί να προσδιορίσει την πόζα του, να μπορεί να εντοπίσει εμπόδια, να μπορεί να σχεδιάσει μονοπάτια και να τα ακολουθεί. Για τα δύο πρώτα, μπορεί να χρησιμοποιηθεί το πακέτο ROS GMapping που χρησιμοποιεί την τεχνική SLAM [13, 21]. Ο εντοπισμός εμποδίων μπορεί να γίνει με διάφορους αισθητήρες, όπως καμέρα [22], αισθητήρες υπερύθρων [23], συνδυασμός [24], λέιζερ [26] κλπ. Εναλλακτικά, μπορεί να χρησιμοποιηθεί η μέθοδος τοπικού σχεδιασμού μέσω Δυναμικού Παραθύρου (Dynamic Window local Path Planner) [25, 26], με χρήση λέιζερ για εντοπισμό εμποδίων. Όταν είναι γνωστές οι θέσεις των εμποδίων, όλα τα τοπικά εφικτά μονοπάτια μπορούν να υπολογιστούν με χρήση του πακέτου ROS costmap 2D και ένα από αυτά επιλέγεται με βάση την ανθρώπινη πρόθεση. Για να ακολουθήσει το ρομπότ αυτό το μονοπάτι, στα [25, 26] δίνεται γωνιακή εντολή ανάλογη της καμπυλότητάς του, ενώ τυπικά σε συστήματα με έλεγχο Admittance, γίνεται προσαρμογή των παραμέτρων απόσβεσης [19, 20]. Σε αυτή την εργασία οι υλοποιήσεις των [25, 26] θα επεκταθούν και θα συνδεθούν με έναν Ελεγκτή Admittance για τον εντοπισμό της ανθρώπινης πρόθεσης, καθώς αυτή θα είναι μια καινοτόμα εφαρμογή.

1.1.1 Έλεγχος Ενεργούς Συμμόρφωσης

Ένας Ελεγκτής Admittance είναι μια συνάρτηση μεταφοράς που αντιστοιχεί τις δυνάμεις/ροπές, που μπαίνουν ως είσοδοι σε ένα σύστημα, σε εντολές γραμμικής και γωνιακής ταχύτητας. Το πιο σύνηθες δυναμικό μοντέλο, το οποίο χωρίζεται σε ένα γραμμικό και ένα γωνιακό μέρος [13, 19], δίνεται παρακάτω

$$m\dot{u} + b_l u = F_h \tag{1.1}$$

$$J\dot{\omega} + b_a \omega = \tau_h \tag{1.2}$$

όπου m, b_l, J, b_a είναι η επιθυμητή μάζα, γραμμική απόσβεση, ροπή αδράνειας και γωνιακή απόσβεση του ρομπότ, αντίστοιχα, και $F_h = F_{L,h} + F_{R,h}$, $\tau_h = (F_{R,h} - F_{L,h})d$ είναι οι είσοδοι δύναμης και ροπής του συστήματος, όπου $F_{L,h}$, $F_{R,h}$ είναι οι δυνάμεις που εφαρμόζονται στην αριστερή και δεξιά λαβή, αντίστοιχα, και d είναι η απόσταση από τον άξονα περιστροφής του ρομπότ. Με βάση τις εξισώσεις (1.1), (1.2) μπορεί να υπολογισθεί η επιτάχυνση ως

$$\dot{u}_i = \frac{F_{h,i} - b_l u_{d,i}}{m} \tag{1.3}$$

$$\dot{\omega}_i = \frac{\tau_{h,i} - b_a \omega_{d,i}}{J} \tag{1.4}$$

όπου $u_{d,i}$, $\omega_{d,i}$ είναι οι επιθυμητή γραμμική και γωνιακή ταχύτητα, αντίστοιχα, την χρονική στιγμή i και έπειτα οι εντολές ταχύτητας δίνονται από τον τύπο της διαφόρισης

$$u_{cmd} = u_{d,i+1} = u_{d,i} + \dot{u}_i dt \tag{1.5}$$

$$\omega_{cmd} = \omega_{d,i+1} = \omega_{d,i} + \dot{\omega}_i dt \tag{1.6}$$

Αυτός ο Ελεγκτής είναι ιδανικός, καθώς χρειάζεται μόνο δύο παραμέτρους ανατροφοδότησης, την ανθρώπινη δύναμη στην αριστερή και δεξιά λαβή του ρομπότ, μέσω των οποίων υπολογίζονται η δύναμη και η ροπή εισόδου. Αυτός ο τύπος ελέγχου έχει δειχθεί σε παρόμοια συστήματα ότι προσφέρει μια φυσική και άνετη αλληλεπίδραση ανθρώπου-ρομπότ [13,17,19].

Στην συνέχεια, ανάλογα με την επιθυμητή συμπεριφορά περιπατητήρα, οι δυναμικές παράμετροι μπορούν να ρυθμιστούν κατάλληλα. Υψηλές τιμές κάνουν το σύστημα να το αισθάνεται ο χρήστης πιο βαρύ, ενώ χαμηλές πιο ελαφρύ. Συχνά, τα m, J επιλέγονται σταθερά και οι αποσβέσεις χρησιμοποιούνται για την προσαρμογή του συστήματος σε διάφορες καταστάσεις, όπως για παράδειγμα αύξηση του b_a όταν ο χρήστης δεν έχει τον σωστό προσανατολισμό για να ακολουθήσει ένα επιθυμητό μονοπάτι και μείωση του αντίθετα [20], ή αύξηση του b_l όταν πλησιάζει σε αντικείμενα, τοίχους, εμπόδια [19], κλπ. Συχνά, οι παράμετροι επιλέγονται πειραματικά και άρα δεν είναι βέλτιστες.

Για να βρεθούν οι βέλτιστες παράμετροι, πρέπει να γίνει μια σειρά καταγραφών δεδομένων ταχύτητας, επιτάχυνσης και δυνάμεων από τις πιο θεμελιώδεις κινήσεις που πρέπει να μπορεί να εκτελεί το ρομπότ για την εκδοχή του χωρίς κινητήρες. Έτσι, σχηματίζεται ένα σύνολο N δεδομένων που αποτελείται από τριάδες $(u, \dot{u}, F_h)$ και $(\omega, \dot{\omega}, F_h)$, στο οποίο εφαρμόζεται ο αλγόριθμος Ελάχιστων Τετραγώνων για να ταιριάξει η ταχύτητα που προκύπτει από τις σχέσεις (1.1), (1.2) με την πραγματική. Πρακτικά, ελαχιστοποιείται το κόστος

$$I = \frac{1}{2} \sum_{i=1}^{N} e_i^2 \tag{1.7}$$

όπου το σφάλμα για το γραμμικό ή το γωνιακό μοντέλο, αντίστοιχα, δίνεται από τις σχέσεις

$$e_i = u_i - \frac{F_{h,i} - m\dot{u}_i}{b_l} \quad \text{ή} \quad e_i = \omega_i - \frac{\tau_{h,i} - J\dot{\omega}_i}{b_a} \tag{1.8}$$

Η λύση των συστημάτων που προκύπτουν δίνουν τις βέλτιστες δυναμικές παραμέτρους. Χρήση τους στον ελεγκτή των σχέσεων (1.3) – (1.6) θα δώσει την βέλτιστη δυναμική συμπεριφορά του ρομποτικού περιπατητήρα, δηλαδή αυτή που είναι πιο κοντά στην εκδοχή του χωρίς κινητήρες.

1.1.2 Πλοήγηση και Σχεδιασμός Κίνησης

Μετά την κατασκευή του δυναμικού μοντέλου αναλύεται ο Dynamic Window Path Planner που είναι απαραίτητος για την πλοήγηση του ρομπότ στον χώρο. Σε αυτή την υλοποίηση, παράγονται όλα τα μονοπάτια που δεν συγκρούονται με εμπόδια και τοίχους, εντός μια ακτίνας $R = 4\mathrm{m}$ από το κέντρο του ρομπότ. Τα μονοπάτια αυτά είναι τόξα (arcs) που ξεκινούν από την βάση του ρομπότ, όπου αν η καμπυλότητα τους είναι αρκετά μεγάλη αποκόβονται στο σημείο που η εφαπτομένη τους είναι παράλληλη με τον άξονα y του ρομπότ και επεκτείνονται με μία ευθεία (line) μέχρι να φτάσουν τον κύκλο ακτίνας R. Έτσι, προκύπτει ένα σύνολο μονοπατιών arc και arc-line, τα οποία δειγματολειπτούνται ώστε να υπάρχει μια λογική απόσταση μεταξύ τους. Η μέθοδος Arc-Line επιλέγεται, γιατί εντοπίζει ανοίγματα στον χώρο αποτελεσματικά.

Κάθε μονοπάτι χαρακτηρίζεται από τις μεταβλητές *γωνία μονοπατιού* φ και *καμπυλότητα μονοπατιού* κ, που συνδέονται άμεσα μεσω της σχέσης

$$\kappa = \begin{cases} 2\frac{\sin\varphi}{R} & ,|\varphi| < \frac{\pi}{4} \\ \\ \frac{1}{R\cos\varphi} & ,|\varphi| \geq \frac{\pi}{4} \end{cases} \tag{1.9}$$

Στην συνέχεια, τα παραγώμενα μονοπάτια οργανώνονται σε ομάδες (clusters) με βάση τον προορισμό τους. Τα μονοπάτια ενός cluster δεν πρέπει να υπερβαίνουν ένα συγκεκριμένο πλήθος και ανάλογα με αυτό ορίζεται η μεταβλητή *εύρος* του cluster W_C. Άλλες αντιπροσωπευτικές μεταβλητές είναι η *γωνία* του cluster φ_C, η *καμπυλότητά* του cluster κ_C και το *μονοπάτι* του cluster P_C που προκύπτουν από το μέσο μονοπάτι του συνόλου. Προκειμένου το ρομπότ να μπορεί να δει ανοίγματα στον χάρτη έγκαιρα, η διαδικασία της ομαδοποίησης πραγματοποιείται παράλληλα σε δύο επίπεδα, ένα κοντινό με $R = 2\mathrm{m}$ και ένα μακρινό με $R = 4\mathrm{m}$ (groups). Τέλος, πριν ξεκινήσει η διαδικασία επιλογής μονοπατιού, ορίζεται η ανθρώπινη είσοδος ως η έξοδος του μοντέλου Admittance, δηλαδή η γραμμική και γωνιακή ταχύτητα που προκύπτουν με βάση τις σχέσεις (1.5), (1.6) και επιπλέον η γωνία ανθρώπου $\varphi_h = K_{max}\omega_d$, όπου $K_{max} = \frac{\varphi_{max}}{\omega_{max}}$ είναι ένας όρος κανονικοποίησης. Αυτές οι τρεις είσοδοι ορίζονται ως η *ανθρώπινη πρόθεση*.

Όταν πολλα cluster μονοπατιών είναι διαθέσιμα, προκύπτει το πρόβλημα τις αναποφασιστικότητας, κάτι που συμβαίνει κυριώς σε διασταυρώσεις ή χώρους με εμπόδια. Για αυτό, στο [26] προτείνεται ένας μηχανισμός ψηφοφορίας των cluster. Το ρομπότ παρατηρεί την πρόθεση του χρήστη και δίνει πόντους στο κοντινότερο cluster με βάση την σχέση

$$V = \begin{cases} 1 & ,|\varphi_C| < a \\ 1 + 2\frac{|\varphi_h| - a}{b - a} & ,a \leq |\varphi_C| < b \\ 3 & ,b \leq |\varphi_C| \end{cases} \tag{1.10}$$

Όταν μια ομάδα έχει πολύ μεγάλη βαθμολογία ή περάσει ένα ορισμένο χρονικό διάστημα, η διαδικασία παρατήρησης σταματάει και το κορυφαίο cluster επιλέγεται. Εφόσον ένα cluster έχει επιλεχθεί, ο Ελεγκτής πλοήγησης είναι υπεύθυνος για να δώσει τις κατάλληλες εντολές ταχύτητας ώστε το ρομπότ να ακολουθήσει το μονοπάτι της. Ο Ελεγκτής ορίζεται με βάση 4 καταστάσεις (S) που προκύπτουν από το πλήθος των clusters. Αυτές είναι:

- **Normal Motion (S_1):** Υπάρχουν το πολύ 1 cluster για κάθε group.
- **Observing Motion (S_2):** Υπάρχουν περισσότερα από 1 cluster τουλάχιστον σε 1 group. Η διαδικασία ψηφοφορίας ενεργοποιείται και το ρομπότ προσπαθεί να επιλέξει το σωστό cluster συγκριτικά με την ανθρώπινη πρόθεση.
- **Assistive Motion (S_3):** Υπάρχει το πολύ 1 cluster για κάθε group, αλλά αυτά έχουν μέση καμπυλότητα μεγαλύτερη από ένα ορισμένο κατώφλι. Το ρομπότ ελέγχει την γωνιακή ταχύτητα μέχρι η καμπυλότητα του cluster να πέσει κάτω από το κατώφλι.
- **Idle (S_4):** Δεν υπάρχουν εφικτά clusters. Το ρομπότ ελέγχεται αποκλιστικά από τον χρήστη και μπορεί μόνο να περιστραφεί στατικά.

Σε έναν έξυπνο περιπατητήρα πρέπει να υπάρχουν τρία ήδη ελέγχου: από τον άνθρωπο, το ρομπότ και συνδυασμός. Σε κάθε περίπτωση ο άνθρωπος ελέγχει την γραμμική ταχύτητα, ωστόσο η γωνιακή στις S_2, S_3 ελέγχεται από το ρομπότ, στην S_4 από τον άνθρωπο και στην S_1 και από τους δύο. Για να γίνεται από κοινού έλεγχος ορίζεται η γωνία

$$\varphi_{shared} = \alpha\varphi_h + (1-\alpha)\varphi_C \tag{1.11}$$

,όπου το $\alpha \in [0,1]$ είναι ανάλογο του εύρους του cluster. Για μικρότερο εύρος, ο έλεγχος είναι στο ρομπότ και όσο μεγαλώνει ο έλεγχος μεταβιβάζεται περισσότερο στον άνθρωπο. Ενδιάμεσα είναι $\alpha = \frac{W_C}{2} - 1$. Αν υπολογισθεί και η καμπυλότητά κ_{shared} από την σχέση (1.9) τότε ο προτεινόμενος Ελεγκτής πλοήγησης δίνεται από τις σχέσεις

$$u_{cmd} = u_d \tag{1.12}$$

$$\omega_{cmd} = \begin{cases} \beta\omega_d + (1-\beta)u_d \cdot \kappa_{shared} & , |\omega_d| < \omega_{max} \wedge S = S_1 \\ \omega_{max} & , \omega_d \geq \omega_{max} \wedge \beta = 1 \wedge S = S_1 \\ \omega_{min} & , \omega_d \leq \omega_{min} \wedge \beta = 1 \wedge S = S_1 \\ u_d \cdot \kappa_C & , S \in [S_2, S_3] \\ 0.6 \cdot \omega_d & , S = S_4 \end{cases} \tag{1.13}$$

όπου u_d, ω_d είναι οι έξοδοι του μοντέλου Admittance, $\beta = 1$ όταν το εύρος του cluster είναι αρκετά μεγάλο και $\beta = 0$ αντίθετα, και ω_{min}, ω_{max} είναι η ελάχιστη και μέγιστη επιτρεπτή γωνιακή ταχύτητα που προκύπτουν από τα όρια του cluster. Ο άνθρωπος έχει τον έλεγχο της γωνιακής ταχύτητας μόνο αν το εύρος του cluster είναι αρκετά μεγάλο και στην κατάσταση Idle. Σε κάθε άλλη περίπτωση το ρομπότ κατευθύνει τον χρήστη.

1.2 Υλοποίηση

Το ρομπότ που χρησιμοποιήθηκε για αυτή την εργασία, αρχικά ήταν ένα όχημα με 4 κανονικές ρόδες και 2 λαβές. Για να γίνει ρομπότ, τοποθετήθηκαν πάνω στις υπάρχουσες λαβές αισθητήρες δύναμης/ροπής μέσω νέων λαβών που σχεδιάστηκαν και παράχθηκαν με 3D εκτύπωση για να μπορέσουν να τους υποστηρίξουν. Επίσης, τοποθετήθηκαν κινητήρες στις πίσω ρόδες με κωδικοποιητές, ενώ υπήρχαν λέιζερ για χαρτογράφηση και εντοπισμό εμποδίων, οθόνη αλληλεπίδρασης και ο επεξεργαστής.

Αφού τοποθετήθηκαν οι αισθητήρες δύναμης/ροπής, έγινε βαθμονόμηση με δύο τρόπους. Αρχικά, αφαιρέθηκαν οι παρατηρούμενες τιμές κατά την ηρεμία. Έπειτα, παρατηρήθηκε ότι αυτές ελαφρώς αλλάζουν όταν ο χρήστης τοποθετεί τα χέρια του πάνω στις λαβές και τείνει να ξεκινήσει μία κίνηση και είναι διαφορετικές κάθε φορά, ωστόσο παρατηρήθηκε ότι μένουν ίδιες από την αρχή ως το τέλος της κίνησης. Έτσι, σχεδιάστηκε μια μέθοδος εντοπισμού των χεριών πάνω στις λαβές μετά από ηρεμία και οι αρχικές τιμές που παρατηρούνται τότε αφαιρούνται από την μέτρηση του αισθητήρα. Η μέθοδος διαιρεί τα τελευταία δεδομένα δύναμης σε υποσύνολα δεδομένων, υπολογίζει τους μέσους όρους αυτών και την τυπική απόκλιση κάθε δεδομένου από τον μέσο όρο που του αντιστοιχεί και συγκρίνει με την αντίστοιχη κατάσταση κατά την ηρεμία. Αν οι μέσοι όροι διαφέρουν σημαντικά ή αν παρατηρηθεί σημαντική απόκλιση των δεδομένων από την κατάσταση ηρεμίας, σημαίνει ότι ο χρήστης έπιασε τις λαβές.

Στην συνέχεια, είναι σημαντικό να υπάρχει μια καλή εκτίμηση της ταχύτητας του ρομπότ. Αυτό αρχικά έγινε μέσω του λέιζερ που επέστρεφε την θέση και μετά παραγώγιση, ή μέσω IMU και μετά ολοκλήρωση. Ωστόσο, η πρώτη μέθοδος απέτυχε λόγω του χαμηλού ρυθμού λειτουργίας, ενώ η δεύτερη λόγω των σφαλμάτων που ενισχύονται μετά την ολοκλήρωση. Έτσι, η ανατροφοδότηση έγινε μέσω κωδικοποιητών σε υψηλή συχνότητα και η εκτίμηση ήταν αρκετά ακριβής.

Προκειμένου να ξεκινήσει η διαδικασία εκτίμησης του δυναμικού μοντέλου, έγινε χαρτογράφηση του εργαστήριου μέσω του πακέτου ROS GMapping που χρησιμοποιεί το λέιζερ ως

είσοδο και εφαρμόζει την τεχνική SLAM. Έπειτα, έγιναν καταγραφές δεδομένων σε αργή, μέτρια και γρήγορη ταχύτητα για τις κινήσεις: 1. Ευθεία, 2. Όπισθεν, 3. Μπρος-Πίσω, 4. Στροφή 360^o Αριστερά/Δεξιά, 5. Αριστερά-Δεξία, 6. Καμπύλη L Αριστερά/Δεξιά, 7. Καμπύλη S Αριστερά/Δεξιά. Καταγράφηκαν δυνάμεις και ταχύτητες, πέρασαν όλα από φίλτρο ομαλοποίησης Savitzky-Golay και μετά υπολογίστηκαν οι επιταχύνσεις με παραγώγιση.

1.2.1 Εκτίμηση Μοντέλου

Τα δεδομένα που προέκυψαν παρατηρήθηκε ότι είχαν διαφορετικό μήκος για κάθε κίνηση που καταγράφηκε. Για να συγχρονιστούν δοκιμάστηκαν δύο τρόποι, δειγματοληψία και παρεμβολή δεδομένων, και στην συνέχεια εφαρμόστηκε η μέθοδος των Ελάχιστων Τετραγώνων. Και στις δύο περιπτώσεις οι λύσεις των παραμέτρων ήταν πολύ κοντά οπότε, επιλέχθηκε η μέθοδος της παρεμβολής, αφού σε αυτή η πληροφορία αυξάνεται ενώ στην δειγματοληψία μειώνεται. Στην συνέχεια, σχηματίζονται ένα γραμμικό και ένα γωνιακό σύνολο δεδομένων, από τις κινήσεις 1, 2, 3 και 4, 5, 6, αντίστοιχα, και μετά από Ελαχιστοποίηση Τετραγώνων προκύπτουν οι εκτιμήσεις

$$m = 24.03\,\mathrm{kg} \quad \text{and} \quad b_l = 33.33\,\mathrm{Ns/m} \tag{1.14}$$

$$J = 4.15\,\mathrm{kg \cdot m^2} \quad \text{and} \quad b_a = 9.34\,\mathrm{N \cdot s \cdot m} \tag{1.15}$$

Το τελικό μοντέλο μπορεί να αξιολογηθεί συγκρίνοντας την έξοδό του με την πραγματική δυναμική συμπεριφορά του ρομπότ χωρίς κινητήρες για τις διάφορες κινήσεις με διάφορες ταχύτητες. Τα αποτελέσματα δίνονται στο μέρος 5.4.3 και στο Παράρτημα 1. Γενικά, παρατηρείται μία μέση δυναμική συμπεριφορά. Στις πιο αργές κινήσεις οι επιθυμητές ταχύτητες είναι μεγαλύτερες από τις πραγματικές, στις μέτριες είναι σχετικά κοντά και στις γρήγορες είναι μικρότερες. Ωστόσο, αυτή είναι η βέλτιστη δυναμική συμπεριφορά που μπορεί να επιτευχθεί.

1.3 Πείραμα Ελεύθερου Χώρου

Ο σκοπός αυτού του πειράματος είναι να δοκιμαστεί η λειτουργία του εκτιμώμενου μοντέλου Admittance στην πράξη. Ο άνθρωπος εκτελεί τις βασικές κινήσεις με διάφορες ταχύτητες σε έναν χώρο χωρίς εμπόδια και περιορισμούς. Επιπλέον, για αποφυγή εισόδων θορύβου θεωρείται ότι το μοντέλο ενεργοποιείται μόνο όταν οι δυνάμεις/ροπές ξεπεράσουν κάποια ορισμένα κατώφλια. Τα αποτελέσματα δίνονται στο μέρος 6.1.1 και στο Παράρτημα 2. Παρατηρείται ότι το σύστημα λειτουργεί καλά για μικρές και ομαλές μεταβολές της επιθυμητής ταχύτητας ή το πολύ μέτριες. Σε πιο απότομες κινήσεις το ρομπότ δεν μπορεί να ανταποκριθεί γρήγορα λόγω περιορισμών που μπαίνουν από το σύστημα ελέγχου των κινητήρων. Αυτό μπορεί να μπερδέψει τον χρήστη και να εφαρμόσει περισσότερη δύναμη που θα γίνει με μια καθυστέρηση μεγαλύτερη ταχύτητα με αποτέλεσμα να χάσει τον έλεγχο του οχήματος. Αυτά ισχύουν κυρίως για το γραμμικό μοντέλο, καθώς το γωνιακό στις περισσότερες περιπτώσεις παρουσιάζει ικανοποιητική ακρίβεια. Η ακρίβεια μειώνεται ελαφρώς όταν συνδυάζονται γραμμικές και γωνιακές κινήσεις, όπως στις καμπύλες L.

Για να γίνει το σύστημα πιο εύχρηστο δοκιμάστηκε το μοντέλο με μικρότερη παράμετρο γραμμικής απόσβεσης ($b_l = 26.33\ \mathrm{Ns/m}$) και παρατηρήθηκε ότι το ρομπότ δίνει την αίσθηση

ότι είναι πιο ελαφρύ και γρήγορο, κάτι ικανοποιητικό για τα επόμενα πειράματα. Αποτελέσματα φαίνονται στο μέρος 6.1.2.

1.4 Πείραμα Περιορισμένου Χώρου

Σε αυτό το πείραμα φαίνεται η αποτελεσματικότητα του συνδυασμού του μοντέλου Admittance με τον Arc-Line Path Planner, μέσω σεναριών που θα εκτελεστούν σε έναν χώρο περιορισμένο από τοίχους και εμπόδια. Το ρομπότ οδηγείται με βάση τον Ελεγκτή των (1.12), (1.13) και επιπλέον με βάση τις 4 καταστάσεις του, ορίζεται η προσαρμοστική απόσβεση

$$b_l = \begin{cases} 26.33 & , S \in [S_1, S_3] \\ 56.33 & , S = s_2 \\ 1033.33 & , S = S_4 \end{cases} \quad (\mathrm{Ns/m}) \tag{1.16}$$

Όταν το ρομπότ βρίσκεται σε Observing Motion η απόσβεση αυξάνεται έτσι ώστε η ταχύτητα να μειώνεται ξαφνικά περίπου κατά 50% και ο χρήστης να αντιληφθεί ότι πρέπει να πάρει μια απόφαση κατεύθυνσης, ενώ για την κατάσταση Idle να μειώνεται περίπου στο 100%. Οι τιμές επιλέχθηκαν πειραματικά. Η προτεινόμενη προσαρμοστική στρατηγική ελέγχου ανθρώπου-ρομπότ ορίζεται από τις σχέσεις (1.12), (1.13), (1.16). Πρακτικά, υλοποιήθηκαν τα σενάρια:

1. Κατεύθυνση προς τον τοίχο υπό γωνία χωρίς εφαρμογή ροπής: Σε αυτό το σενάριο φαίνεται η λειτουργία στην κατάσταση Assistive Motion. Ο χρήστης σπρώχνει το ρομπότ προς τον τοίχο και επειδή το επιλεγμένο μονοπάτι έχει καμπυλότητα μεγαλύτερη από το κατώφλι, που ορίζεται ως $\varphi_{assist} = 0.1 \cdot \varphi_{max}$, το ρομπότ στρίβει το όχημα και κατευθύνει τον χρήστη ώστε να αποφύγει την σύγκρουση. Αποτελέσματα δίνονται στο μέρος 6.2.1.

2. Κατεύθυνση προς τον τοίχο υπό γωνία με εφαρμογή ροπής: Αυτό το σενάριο είναι ίδιο με το προηγούμενο με την διαφορά ότι ο χρήστης ασκεί ροπές ώστε να βγει από το επιλεγμένο μονοπάτι. Το ρομπότ τον δυσκολεύει και τον κατευθύνει πίσω στο σωστό μονοπάτι. Αποτελέσματα δίνονται στο μέρος 6.2.2.

3. Εντοπισμός πρόθεσης σε διασταύρωση Τ: Ο χρήστης εισέρχεται σε διασταύρωση Τ και επιλέγει κατεύθυνση. Εδώ φαίνεται η επίλυση του προβλήματος της αναποφασιστικότητας στην περίπτωση ύπαρξης πολλαπλών εφικτών cluster κίνησης. Αυτό γίνεται μέσω του μηχανισμού ψηφοφορίας των cluster που δίνει πόντους στο cluster που βρίσκεται πιο κοντά στο φ_h, με βάση την σχέση (1.10), για $a = 0.1\varphi_{max}$ και $b = 0.8\varphi_{max}$. Αποτελέσματα δίνονται στο μέρος 6.2.3.

4. Λειτουργία κοντά σε τοίχο: Ο χρήστης πάει πολύ κοντά στον τοίχο και προσπαθεί να σπρώξει το ρομπότ πάνω σε αυτόν. Σε αυτή την περίπτωση δεν υπάρχουν εφικτά μονοπάτια κίνησης και το ρομπότ μπαίνει σε κατάσταση Idle, ενώ η παράμετρος γραμμικής απόσβεσης αυξάνεται ραγδαία. Έτσι, η μόνη εφικτή κίνηση είναι η στατική στροφή μέχρι να εμφανιστούν ξανά εφικτές διαδρομές. Αποτελέσματα δίνονται στο μέρος 6.2.4.

1.5 Συμπεράσματα

Τα αποτελέσματα σε κάθε περίπτωση ήταν ικανοποιητικά, ωστόσο περιορίστηκαν από περιορισμούς της αρχιτεκτονικής του ρομπότ. Όλα τα σενάρια εκτελέστηκαν σε αργή-μέτρια ταχύτητα όπου το σύστημα ελέγχου των κινητήρων λειτουργεί καλά. Ένα σημαντικό πρόβλημα που παρατηρήθηκε είναι ότι κάποιες φορές ο πίσω αριστερά τροχός γλιστρούσε με αποτέλεσμα να προκαλούνται ανεπιθύμητες ταλαντώσεις στο σύστημα.

Στο πείραμα Ελεύθερου Χώρου το ρομπότ είχε μια μέση δυναμική συμπεριφορά σε σχέση με την παρατηρηθείσα. Το γωνιακό μοντέλο ήταν αρκετά ακριβές, ενώ το γραμμικό δεν ανταποκρινόταν κατάλληλα στις κινήσεις με σχετικά υψηλότερη ταχύτητα, λόγω προβλημάτων των κινητήρων. Για τον λόγω αυτό τα πειράματα έπρεπε να γίνουν σε πιο χαμηλές ταχύτητες. Για να γίνει το σύστημα πιο εύχρηστο δοκιμάστηκε να μειωθεί η παράμετρος της γραμμικής απόσβεσης του μοντέλου Admittance, κάτι που ήταν κατάλληλο για τα επόμενα πειράματα εφόσον το σύστημα μπορούσε να κινηθεί πιο γρήγορα.

Στο πείραμα Περιορισμένου χώρου, όλα τα σενάρια ήταν αποτελεσματικά. Στο Assistive Motion, το ρομπότ έστριβε τον χρήστη κατάλληλα, όταν η καμπυλότητα του επιλεγμένου μονοπατιού ήταν πολύ μεγάλη, ενώ αν ο χρήστης προσπαθούσε να αποκλίνει το ρομπότ τον δυσκόλευε και τον επέστρεφε στην σωστή κατεύθυνση. Στο Observing Motion, το πρόβλημα της αναποφασιστικότητας σε διασταυρώσεις λύθηκε μέσω του μηχανισμού ψηφοφορίας και σε κάθε περίπτωση επιλέχθηκε το σωστό cluster, με βάση την πρόθεση του χρήστη που προέκυπτε από το μοντέλο Admittance. Κατά την διαδικασία της ψηφοφορίας η παράμετρος της γραμμικής απόσβεσης προσαρμόστηκε κατάλληλα για να ειδοποιήσει τον χρήστη να πάρει απόφαση, ενώ στο μεταξύ το ρομπότ ήταν υπεύθυνο για την γωνιακή ταχύτητα. Στην κατάσταση Idle, ο χρήστης δεν μπορούσε να σπρώξει το ρομπότ, καθώς η γραμμική έξοδος του μοντέλου Admittance ήταν πάντα μηδενική, λόγω της μεγάλης τιμής της παραμέτρου γραμμικής απόσβεσης. Ωστόσο, μπορούσε μόνο να το στρίψει στατικά ώστε να βρει εφικτά μονοπάτια κίνησης.

Τελικά, παρά τα προβλήματα της δομής του ρομπότ, η σύνδεση μεταξύ μοντέλου Admittance και Dynamic Window Path Planner έγινε και δούλεψε αποτελεσματικά. Η προοπτική για βελτίωση και επέκταση του συστήματος είναι αρκετά υποσχόμενη.

Chapter 2

Background & Related Work

In this chapter, previous studies on the subject of natural collaboration and interaction between humans and robots will be briefly described, mainly focusing on Assistive Robots and robotic rollators.

2.1 Human-Robot Interaction

Humans and robots are two very different entities. Each one has strengths and flaws, but if they work together they can achieve unique results in a very large variety of applications. Industrially, collaborative robots (COBOTs) are designed to reinforce the productivity and reduce the human effort and risk for accidents in hazardous and complex environments. In [1], a human worker guides the path of a very heavy object lifted by a robotic manipulator, thus minimizing the human physical load. In [2], the robot tries to predict the next human intention, using a library of actions and a history of completed tasks, in order to hand over the correct pieces for an assembly task, performed by the human. Similar is the application of [3], where the robot picks objects from starting positions and places them in target positions for the human to use, based on a probabilistic model to predict human behavior, in order to not get in the human way while he is working. Other industrial COBOT applications could be screwing, drilling [4], surface polishing [5] etc. In both scenarios, the human guides a robotic manipulator with minimum effort to achieve a task of high presicion and difficulty, which would tire him very fast if he would do it by himself.

Apart from the industry, in the medical field, robotic surgery through teleoperation has improved the doctor's precision and has eliminated instabilities produced by human hands [6, 34]. In addition, robotic prosthesis has given hope to disabled people and made it possible for humans without legs or arms to walk or catch again [7, 8]. Robotic vehicles, like wheelchairs, scooters [33] and walkers have been recently produced for the motor-impaired and the elderly to simplify their navigation through the environment and reduce their cognitive load [28]. This kind of applications lead to a more specific field of robotics, called Assistive Robots, which help to improve the life of people with disabilities.

2.1.1 Assistive Robots

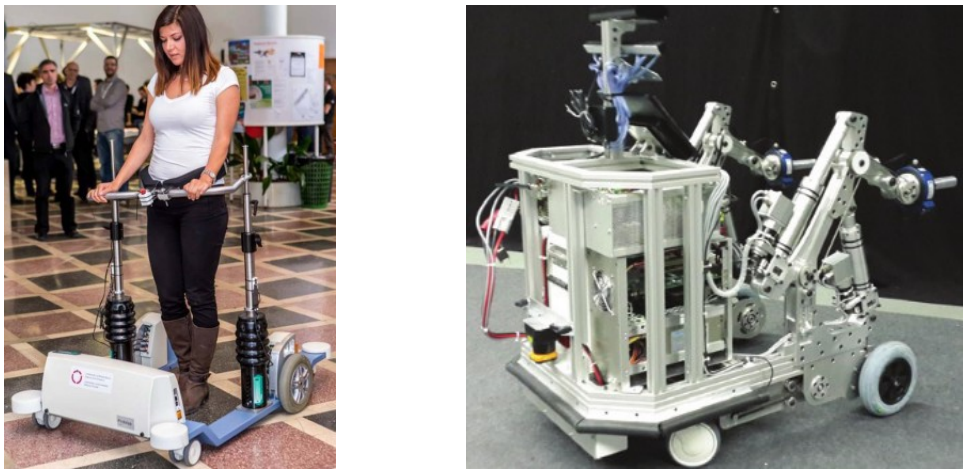
(a) (b) (c) (d)

Fig. 1. (a) Smart Wheelchair [9]. (b) *GuideCane* prototype [10]. (c) *Baxter* arm helping blind human in assembly task [11]. (d) Robot *Hector* from CompanionAble project [12].

Assistive Robotics is a very important field that helps people in need like elderly, injured or disabled, to feel healthy again. These robots could be prosthetics or independent systems that collaborate with their human user to improve his routine and life. An example of such systems is the smart wheelchair (Fig. 1.a) of [9], where the robot takes control of the movement to ensure safety when the user is directing into a narrow passage or near a wall. In [10] a cane robot is used to apply a guidance system to help blind people walk safely (Fig. 1.b), by performing local navigation techniques for avoiding obstacles or hazardous paths. Furthermore, in [11] a Baxter robot helps a blind person in a puzzle assembly process (Fig. 1.c). The robot points its end-effector on the pieces, then the human catches them and waits to be guided to the correct position of assembly, thus the robot becomes his "eyes". Another kind of application is social service or companion robots (Fig. 1.d). These communicate with the user through GUIs or vocally in order to control a smart home/building environment, which could be beneficial for people with mobility difficulties [12]. It is obvious that according to human needs Assistive Robotics' applications could extend to an infinite number of projects. For this specific thesis, the main focus will be on a specific category of Assistive Robots, called smart walkers.

2.1.2 Smart Walkers

Smart walkers are mobile robots with handles that help the elderly or people with mobility limitations or in rehabilitation to walk safely. To make this possible, in any walking aid application, three basic problems must be solved: *human intention estimation*, *path planning*, *path following*. At first, user motion intentions must be detected to let the robot know where he wants to go locally. This has been achieved in many ways in the past, such as through force sensors on the handles to produce velocity intentions through Admittance Control [13], a torso image processing system, using a RGB-D camera facing the user horizontally that builds his torso pose in 3D and then projecting it into a plane facing the robot, hence measuring the plane's position and orientation [14], a user-robot relative position determination by utilizing tilt sensors to compute the human leaning angles relatively to the static robot frame (Fig 2.a) [15] etc. Secondly, an online path planning and path following algorithm needs to be implemented to guide the user through the environment according to his desired global plan, avoiding obstacles



(a) (b)

Fig. 2. (a) The Hoist platform [15]. (b) The MOBOT Mobility assistance platform with actuated handles [17].

and dangerous paths. Related methods for this part will be further analyzed in the navigation approaches section 2.2.

Apart from just walking aid and collision avoidance, more advanced assistive functionalities have been implemented, such as supporting the user to avoid falling over as shown in [16]. For this approach, an omni-directional cane robot was used, which was equipped with a 2-DOF stick fixed on the base of the robot as the handle, a force sensor on the handle for motion control and estimation of human intentions, a LRF to detect the relative distance between the human legs and the robot, and a wearable in-sole load sensor to get the ground reaction force and compute the center of pressure (COP). Then, the COP is observed and if it gets out of the triangle bounds, defined by the feet and the robot base, it means the human is at high risk of falling and the robot gets in a fall prevention mode, where the wheels are stopped and the robot applies the required forces through impedance control, modeling the human as an inverted pendulum to keep him balanced. A fall prevention approach, also based on the inverted pendulum model, was followed in [17], but a rollator with 2- DOF actuated arms was used (Fig. 2.b). In this scenario, the center of mass (COM) was used to obtain the XCOM and thus the required assistive forces and torques were derived, by solving a quadratic programming problem with constraints referring to XCOM staying between the balance boundaries. The same robot was used for the implementation and evaluation of a sit-to-stand transfer assistance method [27, 18], by considering the human as a model with five joints and six rigid bodies and then calculating the optimal external torques that need to be applied on his hands to transfer safely the COM from the zero moment point to the final standing position.

For more generic environments, walking on inclined surfaces has been studied [29, 30]. In this case, to assist the user, when the walker is going upwards it accelerates and when it goes downwards it decelerates. In the same concept of decreasing the human load, an adjustable assistive torque is applied by the robot in [19] based on task difficulty and human fatigue. Focusing on blind people, in [31] the walker sends navigation commands and environment information through vibrating handles and a wearable vibro-tactile belt, when in an autonomous guided navigation mode. However, in a user navigation mode the sensors are only there to inform the user for approaching obstacles and the robot lets the human make the decisions. Cognitive assistance may also play an important role to reduce the human effort and make the human-robot interaction more friendly, through voice commands and visual interfaces [28]. This type of assistance is suitable specifically for people with cognitive problems, like memory loss, visual problems, or for older people which might be to tired to work around the different features of a smart walker platform.

These are just some interesting applications, but many more can be found in the bibliography. For the practical part of this thesis, the i-Walk Assistive Robot [32] will be used, which will be described in section 5.1, for the implementation of an Admittance Control model to obtain human intentions and use them in a local navigation method.

2.2 Navigation Prerequisites & Approaches

A Navigation method, to be complete and practical, requires an up-to-date environment map, a robot localization technique, an obstacle detection method, a path planning and following algorithm. For map building and robot localization, the ROS GMapping package can be utilized, that uses a Simultaneous Localization and Mapping (SLAM) technique [13, 21]. After a static map is known, the robot needs to be able to detect online changes on it, in order to create a local plan. For this purpose a number of sensors could be installed, such as cameras, ultrasonics, lasers and more. Cameras can be used for a vision-based approach as shown in [22], where a 3D point cloud is generated using stereo matching and then filtered to create a 2D local obstacle map. Ultrasonic Sensors (US) provide a cheap alternative, but they lack high precision [23]. Hence, in [24] a camera-US fusion is proposed to calculate obstacles' distance, width and height more accurately. In a different approach, a Dynamic Window local planner is considered in [25, 26], using Laser Range Finder (LRF) for obstacle detection. After an obstacle map is available, a 2D occupancy grid can be produced using the ROS costmap 2D package. Then, every local obstacle-free path can be calculated in a defined radius according to the cell costs. After the paths are available, one of them must be picked according to the user intentions and then the path following problem must be solved. In [25, 26] path following is achieved by giving an angular velocity command that is proportional to the selected path's curvature, but typically for an Admittance controlled system damping modulation has been used to prevent the user from diverging from the path [19, 20]. In addition, if a global goal is available, a global plan can be calculated using optimal algorithms, such as A^* , Dijkstra, artificial potential fields, wavefront etc. In this diploma thesis, the implementations of [25] and [26] will be extended and modified to work with an Admittance Controller, for getting the human input, as it provides an innovative method for navigating locally.

Chapter 3

Admittance Control

Assuming there is a robot of mass m, an Admittance Controller is a transfer function describing how external forces and torques applied on m will be translated into its linear and angular motion commands, most likely its velocity. In other words, it is a force-to-motion control. If this mass is a smart walker with two force sensors mounted on its two handles, respectively, which are grasped by the user hands, the transfer function can vary depending on the desired dynamic behavior.

3.1 Dynamic Model

Typically, to model an admittance controlled system a mass-damper-spring 2nd order system for the linear and angular motion can be considered

$$\mathrm{M}_d\ddot{\mathrm{x}} + \mathrm{B}_d\dot{\mathrm{x}} + \mathrm{K}_d\mathrm{x} = \mathrm{F}_h \tag{1}$$

where M_d, B_d, K_d are the desired inertia, damping and stiffness matrices, respectively, and F_h are the driving forces applied by the user. This kind of model is useful when the position and orientation of the robot are known relatively to a specified map [13] or a position or stiffness control is required. For more general cases where position restrictions are not important, for example because a map is not available or driving in an open space without goal points, the model can be reduced in a mass-damper system as follows

$$\mathrm{M}_d\ddot{\mathrm{x}} + \mathrm{B}_d\dot{\mathrm{x}} = \mathrm{F}_h \tag{2}$$

To simplify the system more, a decoupled model was proposed [13, 19] that divides the system into a linear and an angular subsystem. Thus, the dynamic model can be written as

$$m\dot{u} + b_l u = F_h \tag{3}$$

$$J\dot{\omega} + b_a\omega = \tau_h \tag{4}$$

Where m, b_l, J, b_a are the desired mass, linear damping, inertia and angular damping of the robot, respectively, and

$$F_h = F_{L,h} + F_{R,h} \tag{5}$$

$$\tau_h = (F_{R,h} - F_{L,h})d \tag{6}$$

are the user force and torque inputs, where $F_{L,h}$, $F_{R,h}$ are the human forces on the left and right handle, respectively, and d is the distance from the rotation z-axis of the robot. From equations (3) , (4) reference linear and angular velocities can be produced when the user applies forces or torques to the robot through its handles. Then these velocities can be realized by a low-level controller. This is the model that is going to be used for further analysis on this thesis, because it requires the minimum amount of parameters to work and is suitable for local navigation. This type of control has been shown to provide a natural and comfortable human-walker interaction, as it takes the input forces and it generates a compliant walker behavior [13, 17, 19].

3.2 The Controller

If the Admittance parameters and the linear/angular accelerations are known, the command velocities can be computed from (3), (4) as

$$u_{cmd} = \frac{F_h - m\dot{u}}{b_l} \tag{7}$$

$$\omega_{cmd} = \frac{\tau_h - J\dot{\omega}}{b_a} \tag{8}$$

This approach has been used in [20]. The acceleration term can be obtained through an IMU or could be calculated as the derivative of the motor feedback velocity. However, the acceleration is not really needed to calculate the desired command. Thus, in cases where the calculations are noisy or unreliable, it is proposed that the acceleration is obtained through the dynamic system as

$$\dot{u}_i = \frac{F_{h,i} - b_l u_{d,i}}{m} \tag{9}$$

$$\dot{\omega}_i = \frac{\tau_{h,i} - b_a \omega_{d,i}}{J} \tag{10}$$

where $u_{d,i}$, $\omega_{d,i}$ are the desired linear and angular velocity, respectively, at moment i and then the command velocities are given by the differentiation model

$$u_{cmd} = u_{d,i+1} = u_{d,i} + \dot{u}_i dt \tag{11}$$

$$\omega_{cmd} = \omega_{d,i+1} = \omega_{d,i} + \dot{\omega}_i dt \tag{12}$$

This model is ideal as it requires the minimum of two feedback parameters, the input human left and right forces, through which the total force and torque applied on the robot are calculated.

3.3 Admittance Parameters Regulation & Adaptation

According to the desired walker behavior, the admittance parameters m, b_l, J, b_a can be modulated. From equations (9), (10) it is clear that when the command velocity is very close to zero the accelerations $\dot{u}, \dot{\omega}$ depend only on the mass m and the inertia J, respectively. Thus, these parameters affect the transient state of the system. On the other hand, the linear and angular damping parameters b_l, b_a are important for a good steady state behavior. In general, picking high values for all parameters would make the robot very resistant to changes thus feeling heavier, while picking small values would make it more sensitive thus feeling lighter. Typically, fixed values are picked for m, J empirically and experimentally, while b_l, b_a are used for online walker behavior adaptation as can be seen in [20], where equations (7), (8) were used. In this case, the damping coefficients were highly increased, if the user was diverging from a desired predefined path orientation. In order to guide him back in the correct way, the angular damping coefficient was decreased only if the user intended to turn the robot in the correct direction. In [19] a slightly extended model was used. The dynamics were given by (3), (4) but the input torque consisted of two components, the human and the assistive torque. For this approach, the linear damping parameter was increased, as the robot was approaching an obstacle, with the help of an artificial potential field, while the angular damping parameter was increased if the user tried to exceed the desired predifined path's acceptable orientation limits. Finally, the assistive torque value was modified based on human fatigue. In [13], where equation (1) was used, the damping coefficients were fixed too and the behavior was controlled by the stiffness parameters. It can be seen that, parameters adaptation can be implemented in many ways, depending on the tasks and the level of desired assistance the robot is programmed to provide. Overall, parameters adaptation can provide a very natural human-robot collaboration, but for an assistance to be ideal the user must feel comfortable by having enough control over the system, while being guided with safety [19].

3.4 Learning Admittance Parameters from Demonstrations

As mentioned in the previous section, initially the parameters can be picked empirically or experimentally, but this would not be optimal. Generally, it is desired that when the user has full control of the walker, for example because he is navigating in a very large empty hall far from walls and obstacles, the robot behaves as an unmotorized standard rollator. To find the optimal parameters, at first a number of data must be collected using the unmotorized robot for demonstrations of the most basic movements that the walker must be able to perform. These data are the linear/angular velocities, accelerations, and the human force/torque inputs. Afterwards, a Least Squares algorithm can be used to learn the optimal parameters.

3.4.1 Least Squares

The Least Squares' objective is to adjust the parameters of a model to best fit a dataset, by minimizing the sum of squared errors of the desired fit function. Assuming the dataset consists of N points of $(u, \dot{u}, F_h)$ and $(\omega, \dot{\omega}, F_h)$ trios, the method can be applied to fit the velocity obtained from (3), (4) on the measured velocity. For a decoupled system this must be done twice, once for

the linear and once for the angular model. In this way, the model will be trained to follow the velocity of the dataset with the given force/torque inputs. For the linear model, let the velocity error be

$$e_i = u_i - \frac{F_{h,i} - m\dot{u}_i}{b_l} \tag{13}$$

and the total cost

$$I = \frac{1}{2}\sum_{i=1}^{N} e_i^2 \tag{14}$$

Then, (14) is minimized when

$$\frac{dI}{dm} = 0 \Rightarrow b_l = \frac{\sum F_{h,i}\dot{u}_i - m\sum \dot{u}_i^2}{\sum u_i\dot{u}_i} \tag{15}$$

and

$$\frac{dI}{db_l} = 0 \Rightarrow b_l = \frac{\sum (F_{h,i} - m\dot{u}_i)^2}{\sum u_i(F_{h,i} - m\dot{u}_i)} \tag{16}$$

By solving the system (15), (16) the optimal parameters m, b_l are obtained. Similarly for the angular model, let the velocity error be

$$e_i = \omega_i - \frac{\tau_{h,i} - J\dot{\omega}_i}{b_a} \tag{17}$$

Then, (14) is minimized when

$$\frac{dI}{dJ} = 0 \Rightarrow b_a = \frac{\sum \tau_{h,i}\dot{\omega}_i - J\sum \dot{\omega}_i^2}{\sum \omega_i\dot{\omega}_i} \tag{18}$$

and

$$\frac{dI}{db_a} = 0 \Rightarrow b_a = \frac{\sum (\tau_{h,i} - J\dot{\omega}_i)^2}{\sum \omega_i(\tau_{h,i} - J\dot{\omega}_i)} \tag{19}$$

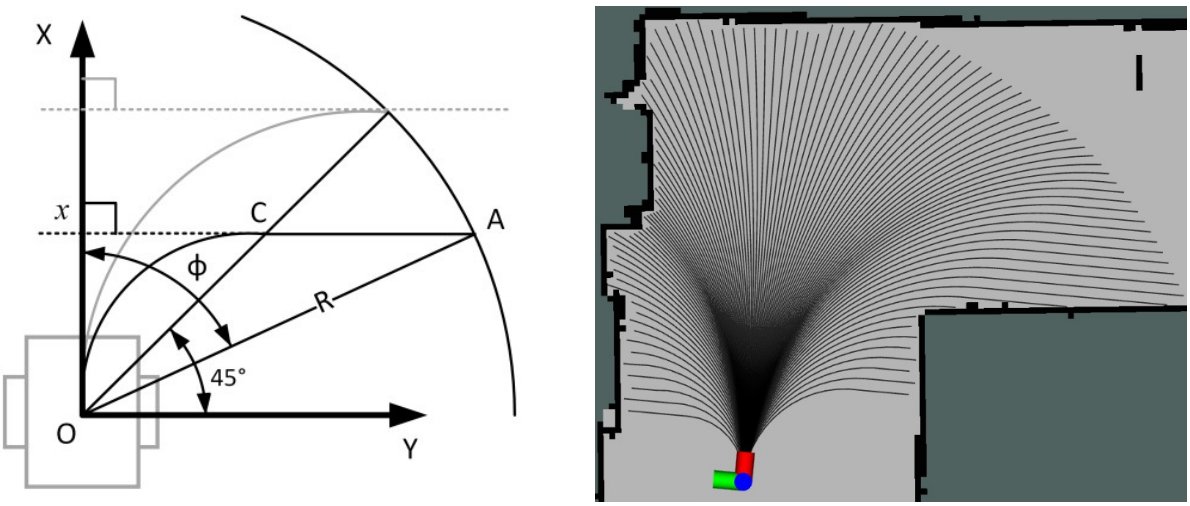
By solving the system (18), (19) the optimal parameters J, b_a are obtained. The solutions can be computed manually, graphically or by a solver library. Having all 4 parameters calculated, by using the Controller of section 3.2, the optimal walker dynamic behavior can be generated, meaning the behavior that is more close to the unmotorized version of the robot.

Chapter 4

Navigation and Motion Planning

After learning the dynamics of the walker, the next important challenge is to guide the user through the environment. In this chapter, the Dynamic Window Approach (DWA) and its modifications, to be able to work along with an Admittance Controller, will be analyzed and explained. The Navigation node will consist of the five subnodes: Human Input, Trajectory Generation, Clustering, Path Planner, Controller.

4.1 Dynamic Window: Arc-Line Trajectory Generation



(a) (b)

Fig. 3. (a) Detailed depiction of Arc-Line paths [25]. (b) Output of the Arc-Line path generation method for radius $R = 4\mathrm{m}$ on a static map.

As explained in [25], the DWA is a widely used local planner which searches for collision-free paths in the input space (u, ω). Given a robot velocity tuple (u_R, ω_R) the algorithm samples paths from the dynamic window $[u_R \pm \Delta T \cdot \dot{u}_{max}, \omega_R \pm \Delta T \cdot \dot{\omega}_{max}]$ and simulates them forward in time for a predefined period, searching for obstacle collisions in a given radius R starting from the robot base. Arc paths with constant curvature κ are generated at each time stump. If possible,

each arc is cut off at the point which its tangent is parallel to the robot's y-axis. Then, their endpoint is extended with a straight line parallel to the tangent, until the limit R is reached (Fig. 3.a). This happens only for the paths that have $|\varphi| > \frac{\pi}{4}$. Otherwise, the arc is cut off at the point where it meets the radius R. Thus, a set of arc and arc-line paths is produced, where each arc has a curvature that is given by

$$\kappa = \begin{cases} 2\frac{\sin\varphi}{R} & , |\varphi| < \frac{\pi}{4} \\ \\ \frac{1}{R\cos\varphi} & , |\varphi| \geq \frac{\pi}{4} \end{cases} \tag{20}$$

where φ is the angle between the path endpoint and the robot's x-axis and can be considered as the path's angle that is given by

$$\varphi = \begin{cases} \arcsin\frac{\kappa R}{2} & , |\varphi| < \frac{\pi}{4} \\ \\ \arcsin\frac{1}{\kappa R} & , |\varphi| \geq \frac{\pi}{4} \end{cases} \tag{21}$$

The produced paths are then sampled to create a fixed spacing between them, thus creating a *path bundle*. The arc-line approach is selected as it provides a good way for finding openings in the map. For a given pose and map, a typical output of the method can be seen in Fig. 3.b, where the robot calculates all feasible paths in a maximum radius of 4 meters from its center. The thick continuous lines are walls and/or obstacles, the light area is free space, the dark area is unknown space, the small thick frame represents the robot's base frame, where the blue dot is its origin, the front red line is its x-axis and the side green one its y-axis, while the thin lines coming out from the robot base are the sampled feasible paths. If the path collides with the wall or an obstacle it is cut off at that point. Otherwise, an opening is detected. After the path bundle is generated, it is essential to pick one path among it to follow.

4.2 Clustering

Before picking a path, the total paths must be organized into groups according to their endpoint destination. Paths that their destination is similar, and they do not collide with walls or obstacles, are going to be in the same group. This process is called *clustering* and each group represents a *cluster*. In order to form clusters, two parameters must be considered; the cluster separation C_{sep} and the minimum cluster span W_{span}. In the words of [25], *"two paths P_i, P_j, where $i > j$, belong to the same cluster if $i - j \leq C_{sep}$"*. Thus, by separating all paths of the path bundle with this condition, clusters $C_{k,l}$ can be produced, where P_k, P_l are the paths of the cluster with the higher and lower angle φ, respectively. Then, for the cluster to be acceptable, its span $W_{k,l}$ must satisfy the condition $W_{k,l} > W_{span}$, where the cluster span $W_{k,l}$ is the length of the chord with angle $\Delta\varphi_{k,l} = \varphi_k - \varphi_l$ and it is defined by trigonometry as

$$W_{k,l} = 2R\sin\frac{\Delta\varphi_{k,l}}{2} \tag{22}$$

This restriction indicates that very small clusters are rejected, as they might correspond to measurement errors or to passages that the robot can not traverse, such as empty spaces under

desks or chairs, small spaces between obstacles, very narrow paths etc. After generating the valid clusters $C_{k,l}$, the path with the mean angle $\varphi_C = \frac{\varphi_l + \varphi_k}{2}$ is selected to represent each one and it is defined as the *cluster path* P_C. Then, The *cluster angle* φ_C can be used to calculate the *cluster curvature* κ_C from equation (20). These two parameters are characterizing each cluster and they are very important for the path planning algorithm that will follow.

4.3 Parallel Clustering

In the previous section, the process of clustering was explained, but not how to exploit the method to improve the overall navigation. During navigation, it is important for the robot to *"see"* near openings in the map ahead of time, so that a safe transition can be made if the user wants to change direction, while also not diverging from its current plan. To achieve that, multiple levels of clustering can be implemented simultaneously. As proposed in [25], a *near* level clustering, containing all paths in the radius $R = 2\text{m}$, is useful for a direct local planning very close to the robot, while a *far* level clustering, containing all paths in the radius $R = 4\text{m}$, can be utilized to detect early the possible ways of the near future motion. The output of the parallel clustering method, for the path bundle of Fig. 3.b, is shown in Fig. 4, where the short thin green lines coming out from the robot base represent the *near* clusters and the longer red ones the *far* clusters. Having these clusters available, the need to pick one of them to follow emerges. There needs to be a method for detecting where the user wants to go, based on the *human input*, and thus selecting the correct cluster as soon as possible to follow its optimal path in order to guide the user in the desired direction effectively.

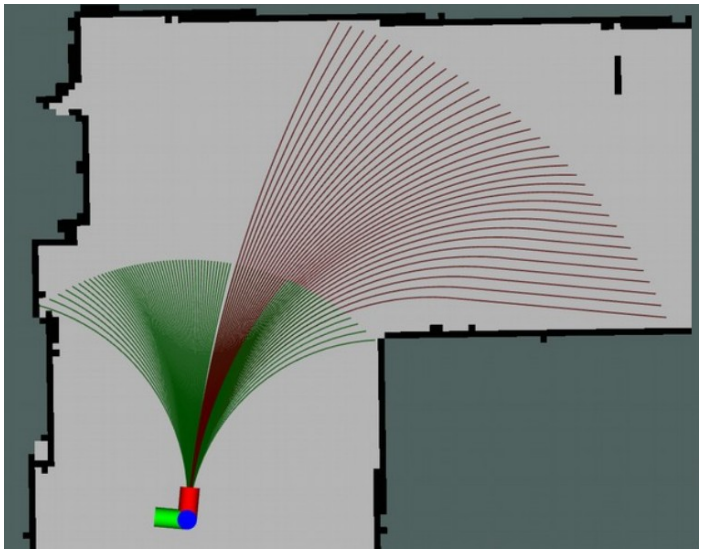


Fig. 4. Parallel two-level clustering for radiuses $R = 2\text{m}$ (green, short) and $R = 4\text{m}$ (red, long), applied on the path bundle shown in Fig. 3.b.

4.4 Human Input

For an Admittance controlled robotic rollator the human input is applied through forces on the walker's handles. Using these forces, the Admittance model generates desired linear and angular velocity commands through equations (11), (12). These commands can then be filtered by the navigation system, instead of giving them directly to the robot. More specifically, as an extension to the work of [26], the input to the navigation system is proposed to be: the linear desired velocity u_d for control of the linear motion, the angular desired velocity ω_d for control of the angular motion and the calculation of a third input; the *human angle* φ_h, which can be used for the detection of the human intended direction of motion, while also for the implementation of a human-robot shared control strategy of the angular motion. The human angle is defined as

$$\varphi_h = K_{max}\omega_d \tag{23}$$

where $K_{max} = \frac{\varphi_{max}}{\omega_{max}}$ is a normalization term, and φ_{max}, ω_{max} are the maximum acceptable path angle and angular velocity, respectively. These three inputs are defined as the *human* intention and their use will be analyzed in the sections to come.

4.5 Path Planning

After a number of clusters are generated, the robot needs to be able to choose one so it can follow its path. This is the *undecidability* problem, which occurs while entering T-junctions, crossroads (Fig. 5) or a space with obstacles, meaning that the robot does not know where the human wants to go, and thus it can not provide a path for following, as there are multiple possible ones. To solve the problem, a cluster score value is established, so that the robot can *"observe"* and compare the clusters to find the one closer to the human intention. In [26] it is proposed that when there are multiple clusters, their scores can be incremented continuously by a vote value given by the following voting function

$$V = \begin{cases} 1 & , |\varphi_C| < a \\ 1 + 2\frac{|\varphi_h| - a}{b - a} & , a \leq |\varphi_C| < b \\ 3 & , b \leq |\varphi_C| \end{cases} \tag{24}$$

where a, b are angle thresholds. This function is suitable for the Admittance approach as it assigns higher votes to clusters with higher curvature, which is proper as the user must apply a noticeable torque to make the robot turn to their direction. At each timestep dt, a vote is given only to the nearest cluster, in comparison with φ_h. The voting procedure stops if all clusters have at least 10 votes and the most voted cluster has a score that is 50% greater than the following's one or if a timeout has passed. Then, the top cluster is selected as the winner of the voting. This procedure can be applied only in a group of clusters, which means that for parallel clustering it should be done for each group once and then compare the winner clusters from all levels. In this case, the cluster with angle closer to φ_h is picked. The final winner cluster is then given to the Controller in order to follow its mean path P_C with angle φ_C and curvature κ_C. This method has been shown in [26] to provide a good solution to the undecidability problem.

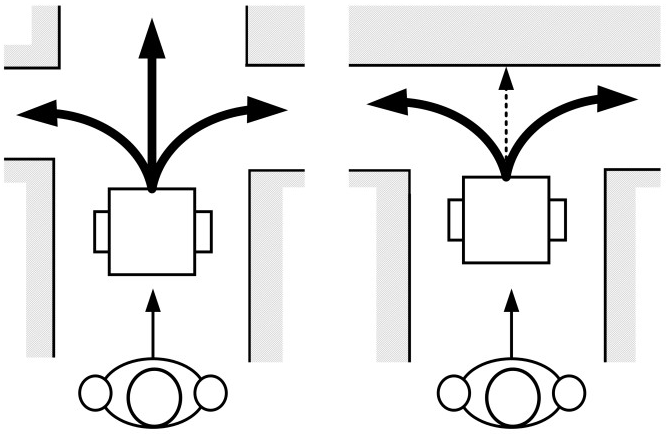


Fig. 5. Undecidable situations: crossroad (left) and T-junction (right) [26].

However, there are not always going to be multiple clusters. Depending on the number and shape of the existing clusters four different robot states (S) are considered:

- **Normal Motion (S_1):** One cluster exists for each group at most.
- **Observing Motion (S_2):** More than one cluster exist at least for one group. The robot considers the human intention to select the correct cluster and controls the angular velocity until then.
- **Assistive Motion (S_3):** One cluster exists for each group at most, but it has curvature greater than a defined threshold. The robot controls the angular velocity until the cluster curvature falls under the threshold.
- **Idle (S_4):** No feasible clusters exist. The robot is fully controlled by the human and it can only be rotated statically.

In cases where there is only one cluster for each group, the one closer to the human angle is picked, while if there are no clusters at all, a NULL path would be the output of the algorithm. These states are important for the definition of the Controller and they will be analyzed further in the following section.

4.6 The Controller

In any smart walker application, three control modes should be considered: *human*, *robot* and *shared control*. If only the human had control of the motion the implementation of assistive features would not be possible, while if only the robot had the control there would be times where the user would want to do something different than the robot suggests and it would not let him, resulting in a poor human-robot interaction. Both entities should have a reasonable amount of control for achieving the optimal behavior. In the proposed Navigation method, the Controller gets the output cluster of the planner as input and computes the final linear and angular velocity commands. In any case the human has full control of the linear velocity, while for the angular each robot state corresponds to a control mode ($Ctrl$):

1. Normal Motion → Shared Control
2. Observing Motion → Robot Control
3. Assistive Motion → Robot Control

4. Idle → Human Control

For shared control to be possible and hence pick a *shared path*, a new parameter is defined, called *shared angle* φ_{shared}, which is given by the equation

$$\varphi_{shared} = \alpha\varphi_h + (1-\alpha)\varphi_C \tag{25}$$

where α is regulated according to the control mode. If the control is fully on the human or the robot it is given by

$$\alpha = \begin{cases} 0 & , Ctrl = Robot \\ \\ 1 & , Ctrl = Human \end{cases} \tag{26}$$

Otherwise, if $Ctrl = Shared$, it is given by

$$\alpha = \begin{cases} 0 & , W_C < 2 \\ \\ \frac{W_C}{2} - 1 & , 2 \leq W_C \leq 4 \\ \\ 1 & , 4 < W_C \end{cases} \tag{27}$$

where W_C is the span of the output cluster. The *shared curvature* κ_{shared} can be computed using equation (20). From equations (25), (27), it occurs that, in shared control, when the environment is wide the human can drive the robot alone, while as the space gets tighter the control goes more on the robot so it can safely guide the human. To actually drive the robot, the linear and angular command velocities must be computed. Depending on the four robot states, the proposed Controller is defined as

$$u_{cmd} = u_d \tag{28}$$

$$\omega_{cmd} = \begin{cases} \beta\omega_d + (1-\beta)u_d \cdot \kappa_{shared} & , |\omega_d| < \omega_{max} \wedge S = S_1 \\ \omega_{max} & , \omega_d \geq \omega_{max} \wedge \beta = 1 \wedge S = S_1 \\ \omega_{min} & , \omega_d \leq \omega_{min} \wedge \beta = 1 \wedge S = S_1 \\ u_d \cdot \kappa_C & , S \in [S_2, S_3] \\ 0.6 \cdot \omega_d & , S = S_4 \end{cases} \tag{29}$$

where u_d, ω_d are the outputs of the Admittance model, β depends on the cluster span as

$$\beta = \begin{cases} 0 & , W_C < 4 \\ \\ 1 & , 4 \leq W_C \end{cases} \tag{30}$$

and ω_{min}, ω_{max} are the minimum and maximum acceptable angular velocities derived by the cluster limits as

$$\omega_{min(max)} = u_d \cdot \kappa_{min(max)} \tag{31}$$

where κ_{min}, κ_{max} correspond to the paths of the output cluster with the lower and higher curvature, respectively. From equations (28), (29) it can be seen that in all cases the human has full control of the linear motion, meaning that the robot moves forward only if the user pushes it. For the angular motion, it is clear that in Normal Motion state the human has full control only if the cluster span is wide enough and the desired angular velocity is between the cluster's minimum and maximum limits. As the cluster span shrinks, the human control level decreases as well. Thus, when the cluster span becomes very tight the robot gets the full control, something which also occurs when the human does not apply torque. In Observing Motion state, the robot has the angular control, while the human desired angular velocity is used to determine the human intention and choose a cluster of motion. In Assistive Motion state, the robot controls the angular motion until the curvature of the selected cluster falls under the defined angle threshold. Finally, in Idle state the human has no restrictions in turning, but the desired velocity is scaled down by a factor. In this human-robot interaction system, the human moves the vehicle as he desires and the robot confines its angular movement to assure safety or assistance.

Chapter 5

Experimental Setup

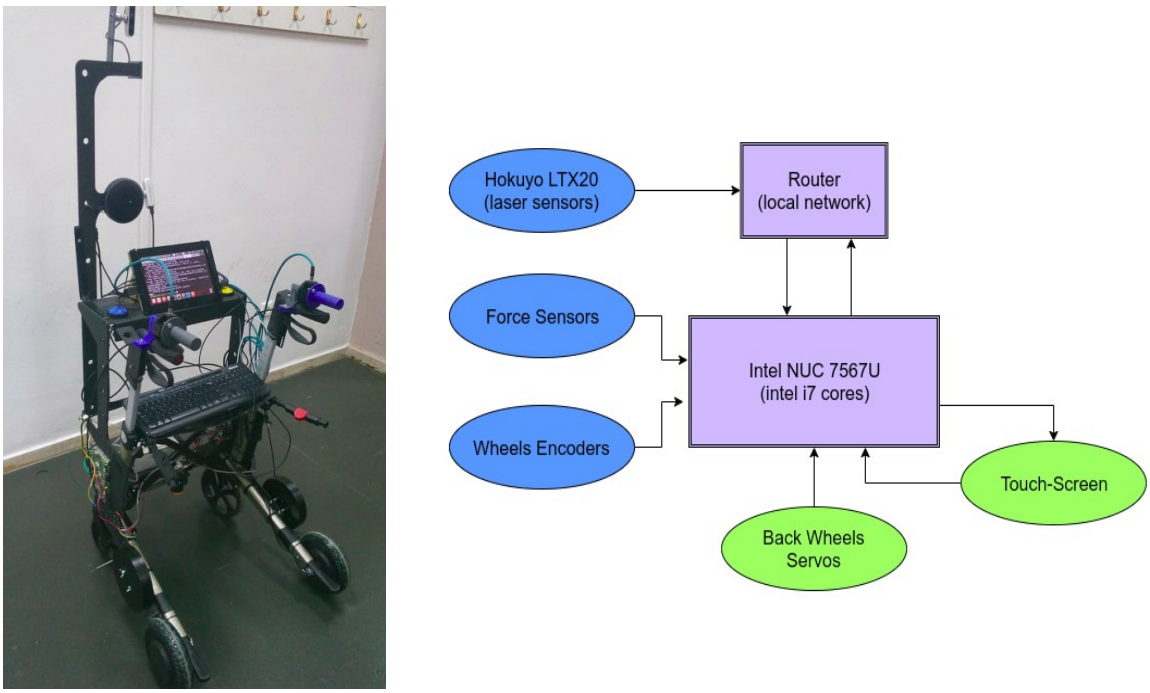
In this chapter, it is explained how the theoretical ideas and models were implemented on the real robotic rollator. The first four sections focus on the architecture, how the force/torque sensors were installed, and velocity feedback methods tried. The next ones describe the necessary procedures required to estimate the unknown Admittance model parameters. Finally, the estimated model is compared with the observed one.

5.1 i-Walk Assistive Robot Architecture

In Fig. 6.a the rollator used for the practical part of this thesis is presented. The hardware that was utilized is part of the one presented in [32] and it is listed below:

1. Force/Torque (F/T) sensors on the handles
2. Hokuyo lidar UST-20LX for navigation
3. Two servomotors on the rear wheels
4. Differential Encoders on all wheels
5. Mini-PC, Intel NUC 7567U (intel i7 cores)
6. 10.1 inches Touch-Screen Display

The F/T sensors are the main device to receive the human input driving forces and give them to an Admittance Controller for intention estimation or control. The laser is used for mapping, online obstacle detection and also is fused with the encoders to form a localization method for computation of the robot's online position and velocity with respect to a defined map. These are the odometry data and are used for the navigation. The robot is able to move through the servomotors on the rear wheels, while all the processing is happening in the intel i7 mini-PC, which serves as the central processing unit of the total structure. Finally, the Touch-Screen Display can be used as a human-robot interface, but for this project it is there only to start/stop the robot and display the navigation output. The total architecture and the connectivity is displayed in Fig. 6.b. The software that makes the programming of the robot possible is the widely used Robot Operating System (ROS), running on Ubuntu Linux.

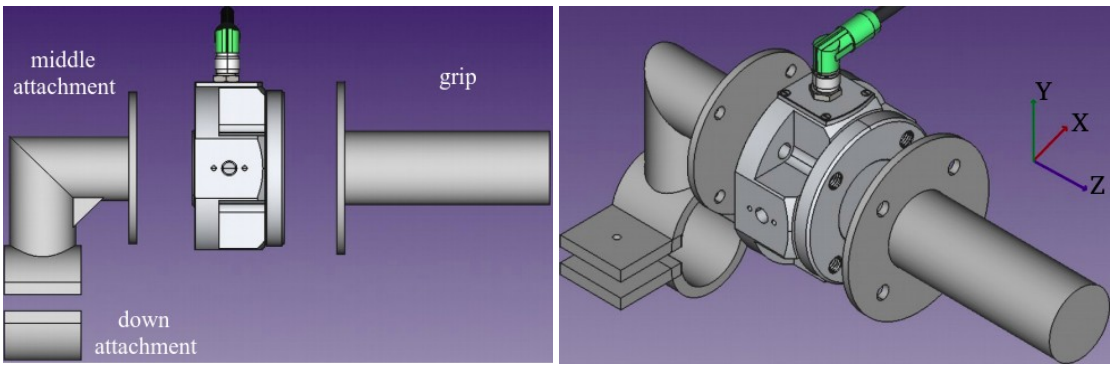


(a) (b)

Fig. 6. (a) The i-Walk Assistive Robot. (b) Hardware architecture.

5.1.1 Mounting Force Sensors on Handles

As the robot structure by itself had no F/T sensors installed, a new handle was designed, in the FreeCad program, and 3D printed to hold the SensONE Bota F/T sensor. This is a 6-DOF device returning 3 force and 3 torque values applied on its internal frame and it was regulated to work on a rate of 100Hz. The handle was designed as practical and economic as possible, and hence it contains three parts; the down attachment, the middle attachment and the grip. The grip has 8cm length and 3cm diameter and in its end it is designed to connect with the sensor through its screw holes. The same goes for the front end of the middle part, while its back end is designed to fit on the already existing robot handle, where the whole structure is fixed through the down attachment. The 3D design can be seen in the first three images of Fig. 7, while the last one shows how the structure was practically placed on the robot. The long straight tube is the grip, the 90º angle part with the triangle nerve is the middle attachment, and the smaller one is the down attachment. The whole structure might not be optimal, but it fits for the requirements of this project.



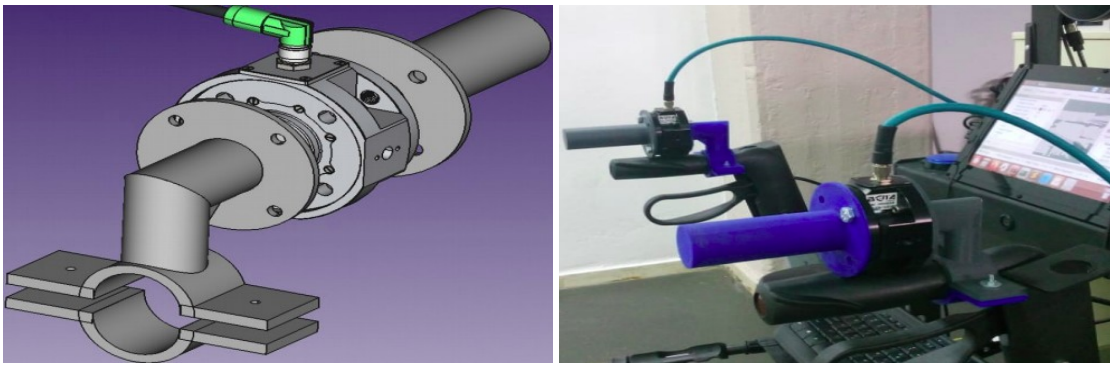


Fig. 7. Design of handle to hold the SensONE F/T sensor and practical placement on the robot after 3D printing.

5.1.2 Calibration

After mounting the F/T sensors on the robot, their bias values change due to gravitational and inertial forces, which depend on the way the structure of the handle is designed and installed, like for example how much tensioned are the screws or how much force is applied due to the weight of the handle and the sensor itself etc. This means that, the values of each sensor in any direction will not be zero if no external load is applied on it, which is not desired. In addition, the initial bias may change as the human drives the robot, due to the hand weight and the motion specific grip of the handles. This problem is solved in two steps. At first, for each sensor, the mean value of the output of each dimension is calculated at rest, while no external forces are applied. Then, these 6 values form the correction vector F_{Bias}, which must be subtracted from the measurement vector to get the real external load. Thus, the sensor measurements must be corrected according to the following formula

$$\mathrm{F}_{Expected} = \mathrm{F}_{Measured} - \mathrm{F}_{Bias} \tag{32}$$

Applying this formula the sensor output at rest will be zero. However, if the human starts using the robot, the bias might shift slightly. Thus, in the second step, the bias occurring from the human hands and the movement must be rejected. To achieve that an online bias correction method was implemented. More specifically, it was observed that when a movement starts the biases are fixed in a value until it is over. If another movement starts the values may be different but they will also be fixed until the end. Thus, to reject this bias its value is computed at the very moment the human grips the handles and the F_{Bias} is modified accordingly. So at that moment, the sensor measurements will go from rest to disturbance in any dimension. To detect that, for every 20 measurements, that is every 0.2 seconds, the mean value of them and their deviation from it is computed. Then those values are kept into vectors of maximum length 4 and 80, respectively. When enough measurements X_i are available, the mean M and deviation s vectors are created as

$$\mathrm{M} = [\frac{X_1 + \cdots + X_{20}}{20}, \ldots, \frac{X_{61} + \cdots + X_{80}}{20}] \tag{33}$$

$$\mathrm{s} = [(X_1 - X_\mu)^2, \ldots, (X_{80} - X_\mu)^2] \tag{34}$$

where X_μ is a final mean value obtained from M. When the sensors are resting, all 4 values of M must be almost the same and that's the value of X_μ. Also, all 80 values of s must be very close to zero. Depending on the sensor noise, some of these values might be slightly higher, but in any case a resting state is determined for each sensor, in order to be able to detect changes from it. So, if any value of M or s are diverging from the resting state values the start of a motion is considered and the bias of that moment is used to correct the F_{Bias}. To apply this method online, it is continuously checked if the robot is back at the resting state. If not the bias is fixed and does not need correction. The total forces and torques applied on the robot can now be calculated from equations (5), (6), where the distance between the handles is measured as $d = 0.243\mathrm{m}$ and $F_{L,h}$, $F_{R,h}$ are the corrected force measurements on Z-axis of the left and right sensor, respectively. It is noted that for each sensor only the force values of Z and Y axis are going to be used, while the rest are discarded. The Z values are used for the calculation of the driving forces and torques, while the Y values for the detection of the human hands on the handles. The bias correction method must be applied to both of these dimensions separately. In Fig. 8, the effectiveness of the

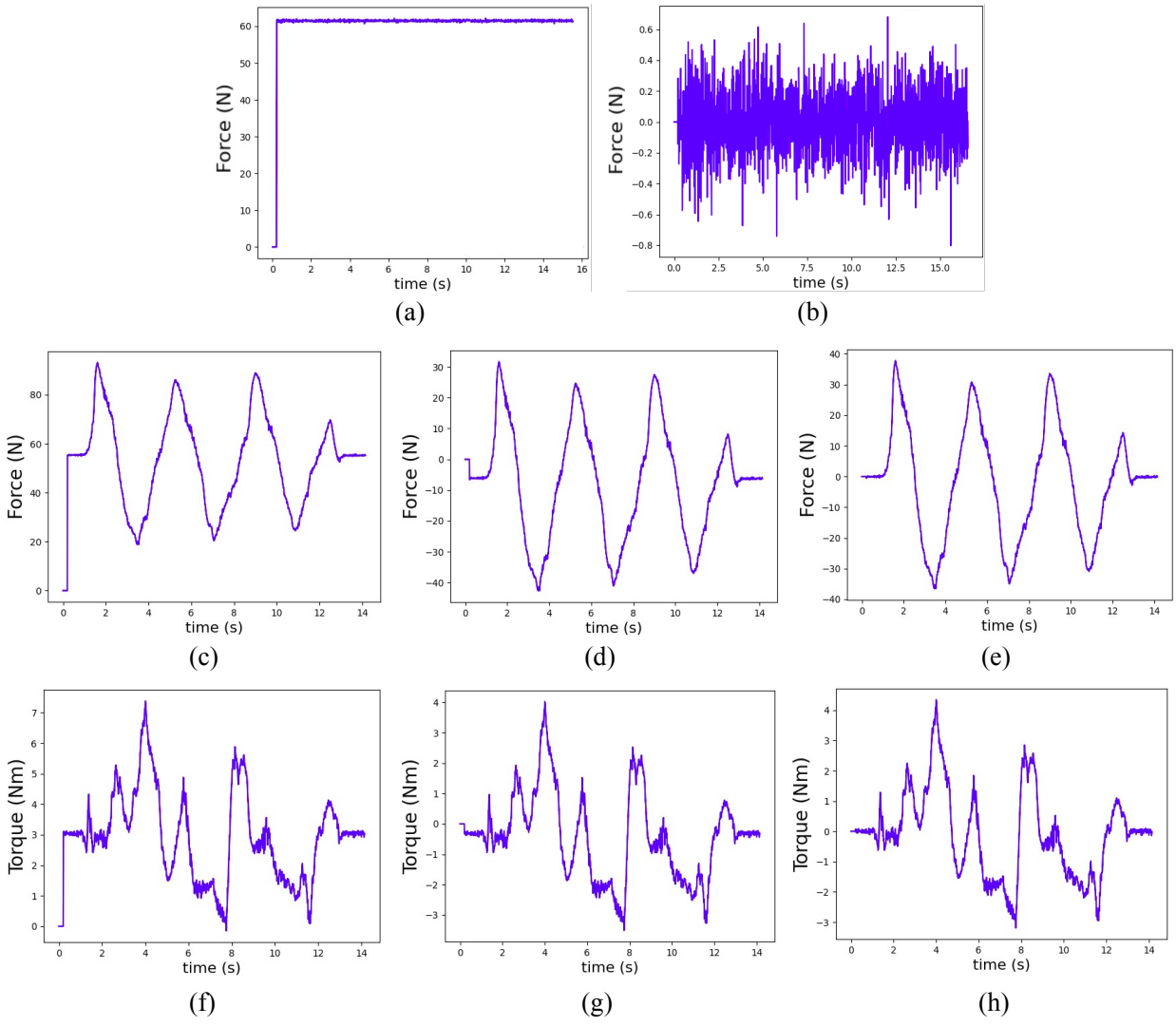


Fig. 8. $Force = F_{L,h} + F_{R,h}$, $Torque = (F_{R,h} - F_{L,h})d$. (a) Resting state force output without bias correction. (b) Resting state force output with one-step bias correction. (c) Back-Forth motion force output without bias correction. (d) Back-Forth motion force output with one-step bias correction. (e) Back-Forth motion force output with two-step bias correction. (f) Back-Forth motion torque output without bias correction. (g) Back-Forth motion torque output with one-step bias correction. (h) Back-Forth motion torque output with two-step bias correction.

method is shown, after applying it on the force measurements of Z-axis of each sensor. While on resting state, the initial force measurement is seen in Fig. 8.a and the correction of step one in Fig. 8.b. A second step correction is not needed as the human does not interfere. For a simple back and forth motion, the initial force measurement can be seen in Fig. 8.c, the step one correction is reflected in Fig. 8.d and the final correction in Fig. 8.e. Here two correction steps are essential, because the human grip changes the bias randomly. The same applies for the torque measurement Fig. 8.f - 8.h. In every scenario the bias is rejected completely.

5.1.3 Velocity Feedback

The next important parameter that needs to be measured is the velocity of the vehicle. Three ways for acquiring velocity were tested: a) LRF and derivative, b) IMU and integration, c) Encoders feedback.

In method (a) the laser is used for localization in a defined static map. The position and the quaternion of the robot were returned in a 5Hz rate. Then the linear and angular velocities were acquired through differentiation, but due to the poor rate they were not continuous and contained steps. To be able to have an estimation of the real velocity, the steps were firstly smoothed with a Savitzky-Golay filter and then approximated with a 4th grade polynomial function (Fig. 9). After comparing different smoothing and denoising filters, like the Butterworth Low Pass of various orders, the Savitzky-Golay filter was proven to be the more appropriate for the smooting of sampling signals coming from optical encoders [35, 36]. However, this is an estimation on the estimation and it is very different from the real velocity, hence that is why a new method was required.

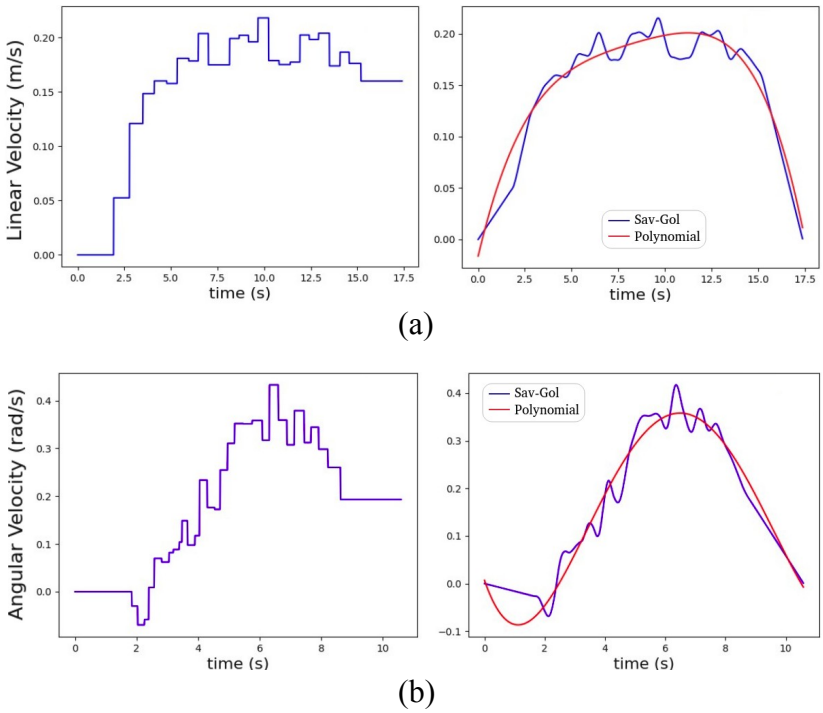
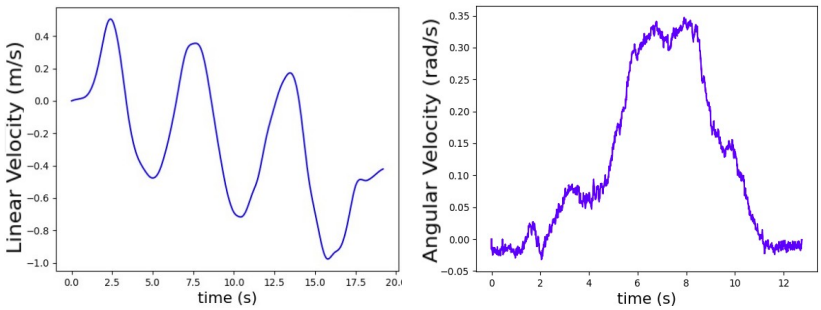


Fig. 9. Linear (a) and angular (b) velocity estimations for a forward and a L Left motion, respectively. LRF to derivative (LEFT) to Savitzky-Golay to 4th grade polynomial (RIGHT).

In method (b) an IMU was installed, which was consisted of an accelometer and a gyroscope. This time the output of the sensor was the linear and angular acceleration and the angular

velocity. Although, the rate was 95Hz the acceleration measurements were quite noisy, so again a Savitzky-Golay filter were used to smooth those graphs. However, even though the angular velocity was a very good estimation (Fig. 10.b), to obtain the linear an integration of the



(a) (b)

Fig. 10. (a) Linear velocity after integration of filtered linear acceleration. (b) Angular velocity as the raw output of the gyroscope.

smoothed linear acceleration should take place. This is problematic as the errors of acceleration are amplified in velocity (Fig. 10.a). Hence, again this method is rejected.

To have the maximum precision, wheel encoders are the best way, as their feedback depends on how much the wheel has been rotated. With this method, even with a low rate of 33.3Hz, as was initially selected, the estimation of the velocity was very realistic. However, to have the best results the encoders were regulated so that a rate of 100Hz could be achieved. This is the most accurate velocity feedback the walker could have and the output is shown in Fig. 11. That is the case only if the surface is smooth and flat, and the wheels are not slipping or stop rotating due to uneven ground. To minimize the possibility of such errors two weights were installed next to the rear wheels to keep them to the ground as much as possible.

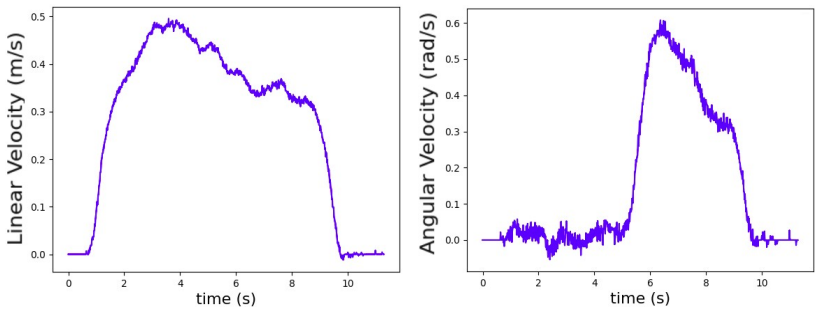


Fig. 11. Encoders' linear and angular velocity feedback in 100Hz rate.

5.2 Mapping

Before starting to collect data for the model training, a map must be defined for all the work to be applied. To create a map from the environment, laser measurements must be given as input to the ROS GMapping package. After a node that uses SLAM is started, the user must navigate the robot very slowly around the walls and the different peculiarities of the space; objects, lengths, widths, corners etc must be carefully detected and measured. After completing the process of mapping, the map of Fig. 12 was generated. This is the NTUA Robotics Lab and that is where

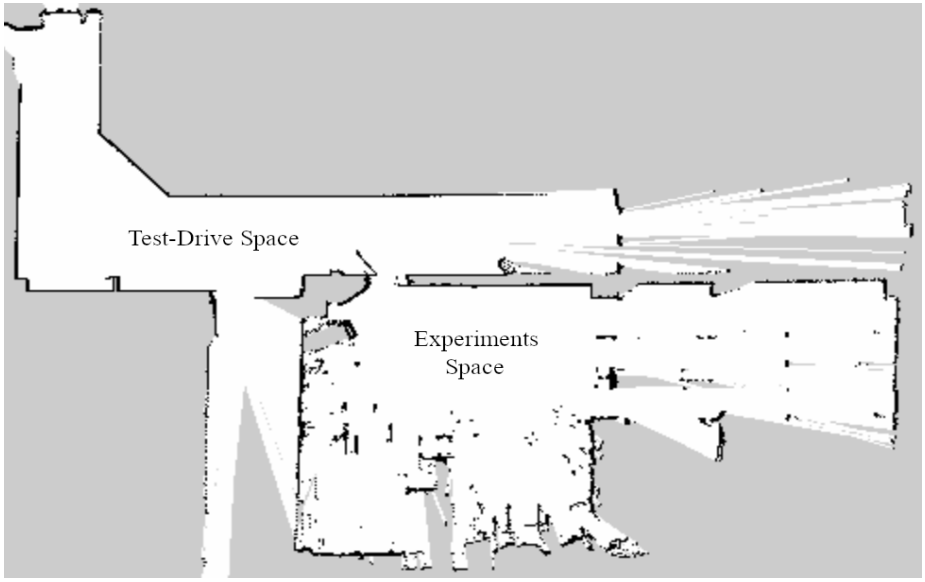


Fig. 12. Robotics Lab map

all the experiments and applications happened. The Experiments Space is the main workspace, as there all the data collections, practical evaluations and experiments are going to take place, while the Test-Drive Space purpose is to evaluate the robot's behavior and responsiveness in a bigger and more realistic environment.

5.3 Data Collection through Recordings

After the hardware, software and workspace of the robot are ready, the collection of force and velocity data may start. This data is later going to be used for the training of the robot, so the most basic movements must be performed. Those are:

1. Forward
2. Backward
3. Back-Forth
4. 360º turn Left/Rightwise
5. Left-Right
6. L curve Left/Rightwise
7. S curve Left/Rightwise

Each movement is illustrated in Fig. 13. For each one, a total of 9 repetitions were done with modulated speed, 3 slow, 3 medium and 3 fast, and slightly different starting and ending positions. In addition, for the curves 6-7 a variation of sharp and smooth turns where made. The data from these simple movements are then going to be processed and forwarded to the Least Squares algorithm, in order to build the optimal Admittance model. The force and velocity data were smoothed by a Savitzky-Golay filter and then acceleration data were obtained by differentiation.

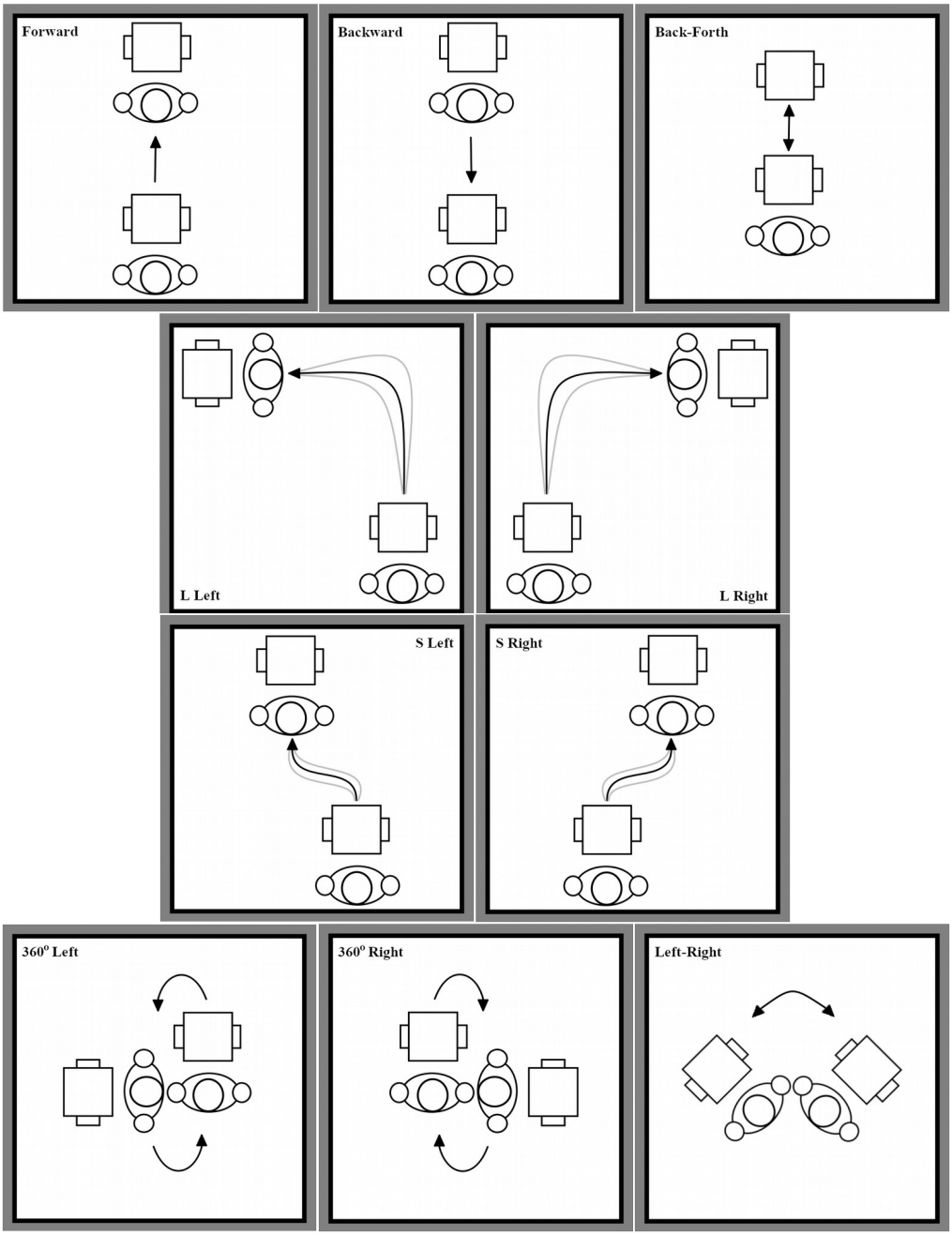


Fig. 13. The basic training movements performed for data collection

5.4 Admittance Model Estimation

5.4.1 Data Pre-processing & Parameters Estimation

After finishing the data recordings it can be seen that different demonstrations have different data lengths as they take different amount of time to be performed. This means that for a learning algorithm there would be more data for the slower demonstrations and less for the faster ones, so

	m (kg)	b_l (Ns/m)
Forward	24.53	19.80
Backward	29.40	33.24
Back-Forth	22.21	41.51

Table 1. Least Squares solutions on the dataset of each basic movement used for the linear model estimation, after sampling.

	m (kg)	b_l (Ns/m)
Forward	24.60	20.10
Backward	29.44	33.95
Back-Forth	22.18	44.78

Table 2. Least Squares solutions on the dataset of each basic movement used for the linear model estimation, after interpolation.

the solutions of the Admittance parameters would adapt more to those with the more information. Thus, before the learning process, every demonstration should have the same length. This can be done in two opposite ways, interpolation and sampling. In the first case, the length of the longer demonstration is picked as the default and then for the shorter ones data are interpolated between the existing points until the max length is reached. In contradiction, for sampling the length of the shorter demonstration is picked and the longer ones are sampled until they reach it. In either way, in the end all demonstrations have the same length and they can be stacked to form a dataset for each basic movement. To compute the linear model parameters with both ways the movements *Forward*, *Backward* and *Back-Forth* are used. If each movement's dataset contains 2 slow, 2 medium and 2 fast demonstrations the Least Squares algorithm, which was explained in section 3.4.1, gives very similar solutions as can be seen in Table 1 and 2. Hence, the interpolation method will be picked to form the datasets as it is slightly better, because the information is increased, while in sampling information is lost. To compute the angular model parameters the demonstrations of *360º Left*, *360º Right* and *Left-Right* are used. In Table 3, the results are shown after interpolation. It is noted that to separate the dynamics, for the linear model, demonstrations with almost zero angular velocity were used, while for the angular model demonstrations with

	J (kg · m^2)	b_a (N · s ·m)
360º Left	5.59	9.77
360º Right	5.77	9.34
Left-Right	3.83	8.54

Table 3. Least Squares solutions on the dataset of each basic movement used for the angular model estimation, after interpolation.

almost zero linear. It can be observed from Tables 2-3 that, depending on the specific movement, the values of the parameters slightly change. So, to get the final estimations, two datasets are

constructed, one for the linear and one for the angular model. This is done by stacking the datasets of their corresponding three basic movements, respectively. Applying the Least Squares algorithm of section 3.4.1 on the two datasets, the following final estimation are obtained

$$m = 24.03\,\mathrm{kg} \quad \text{and} \quad b_l = 33.33\,\mathrm{Ns/m} \tag{35}$$

$$J = 4.15\,\mathrm{kg \cdot m^2} \quad \text{and} \quad b_a = 9.34\,\mathrm{N \cdot s \cdot m} \tag{36}$$

In any case, use of these parameters in an Admittance Controller, will result in a system following an average dynamic behavior in comparison with the various real dynamics that were observed and recorded. The rest of the demonstrations, like the L and S curves were not used for the learning process as they contain both linear and angular movements in varying speeds. For example, in a L curve movement the user starts going forward then reduces its linear velocity to start turning and then increasing it again. This behavior would result in an unclear estimation. So these curves can be used later for the validation of the total estimated dynamic system.

5.4.2 Model Validation

The estimated model given by equations (9) - (12) with the parameters of (35), (36) can be validated by comparing its output with the real observed dynamic behavior of the unmotorized robot, while given the same force/torque input. The results from the model validation (MV) can be seen in the following section 5.4.3. More can be found in Appendix 1.

5.4.3 Model Validation Results

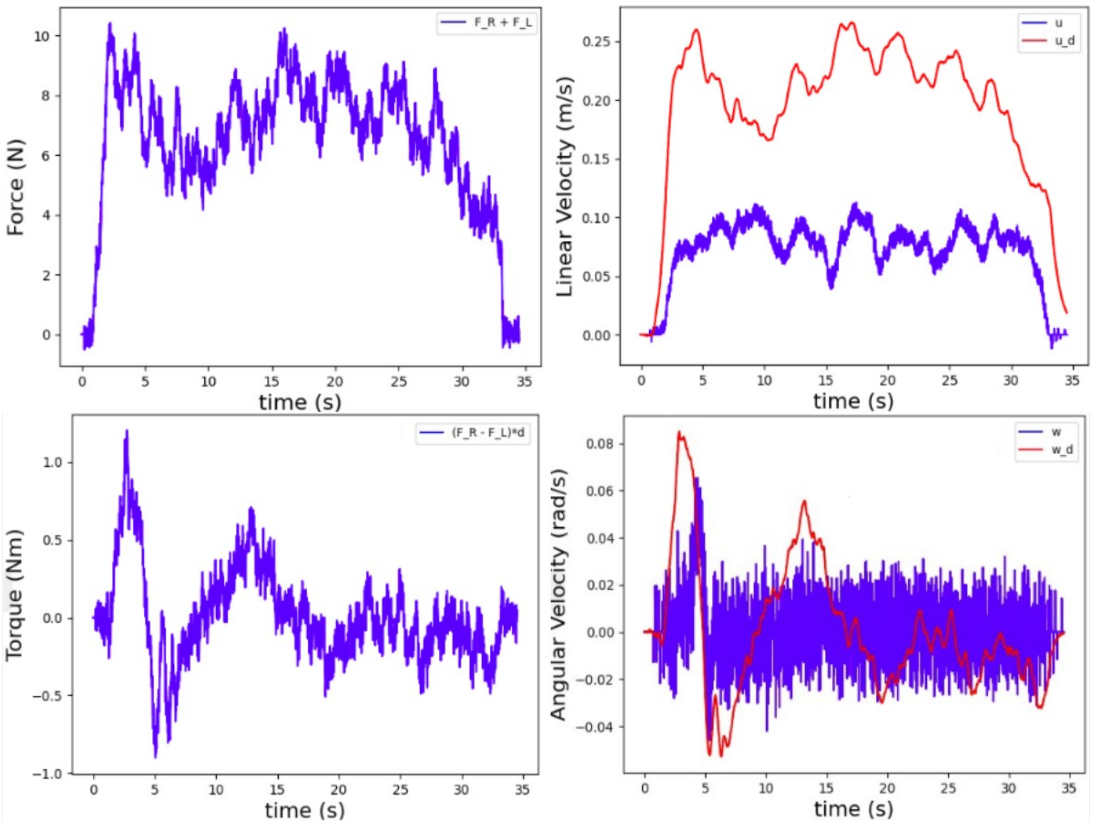


Fig 14. MV: Slow Forward movement. The desired linear velocity is greater than the robot's real one.

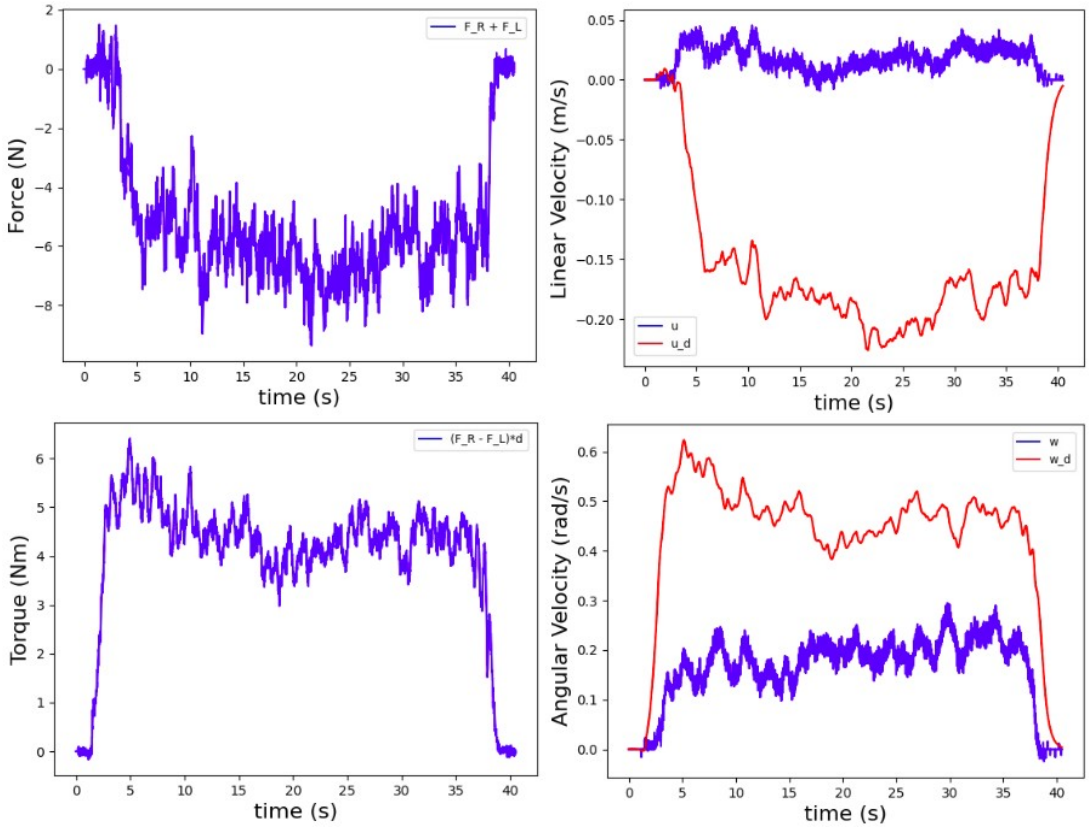


Fig 15. MV: Slow Leftwise 360º turn movement. The desired angular velocity is greater than the robot's real angular velocity.

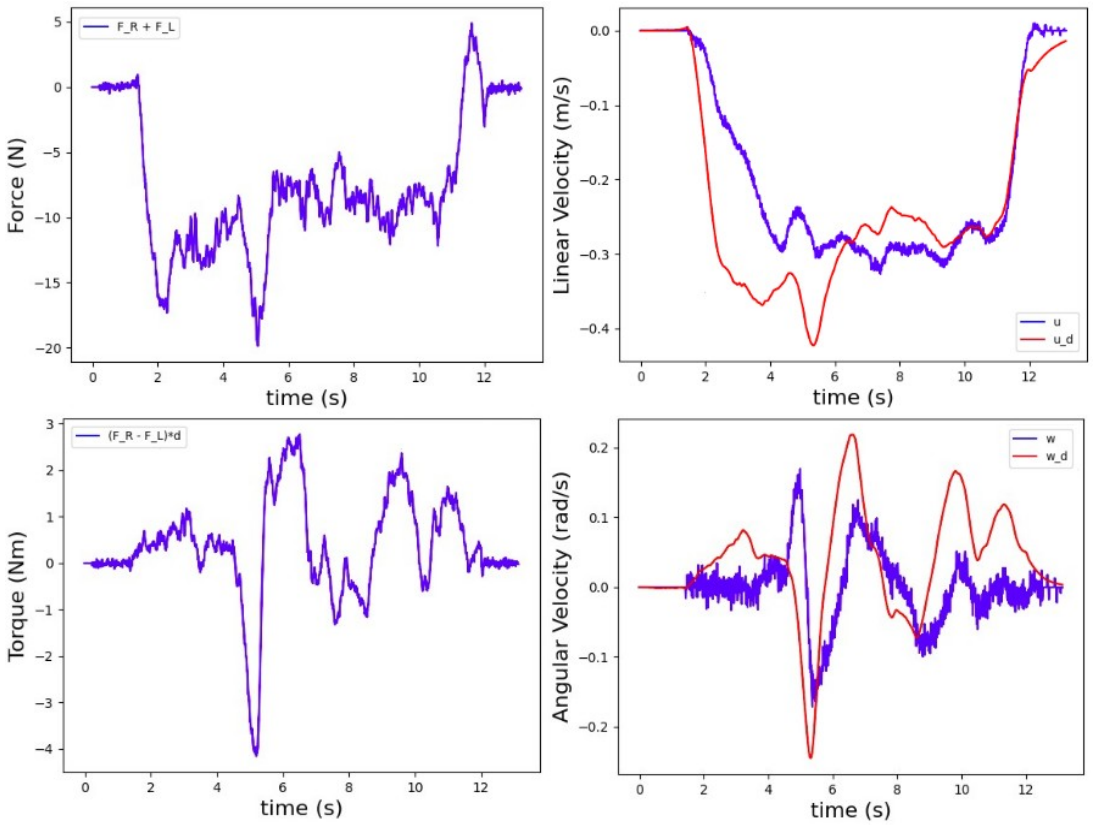


Fig 16. MV: Backward movement with medium speed. The desired linear velocity is very close to the robot's real linear velocity.

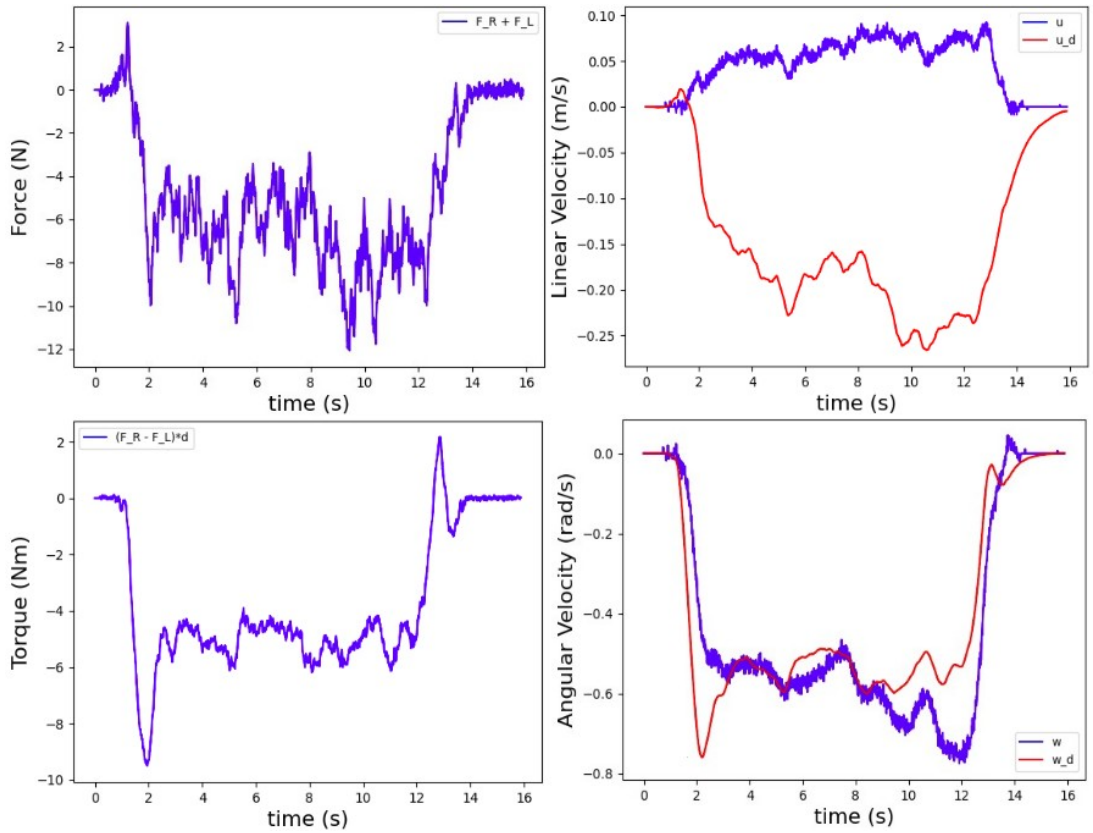


Fig 17. MV: Rightwise 360º turn movement with medium speed. The desired angular velocity is very close to the robot's real angular velocity.

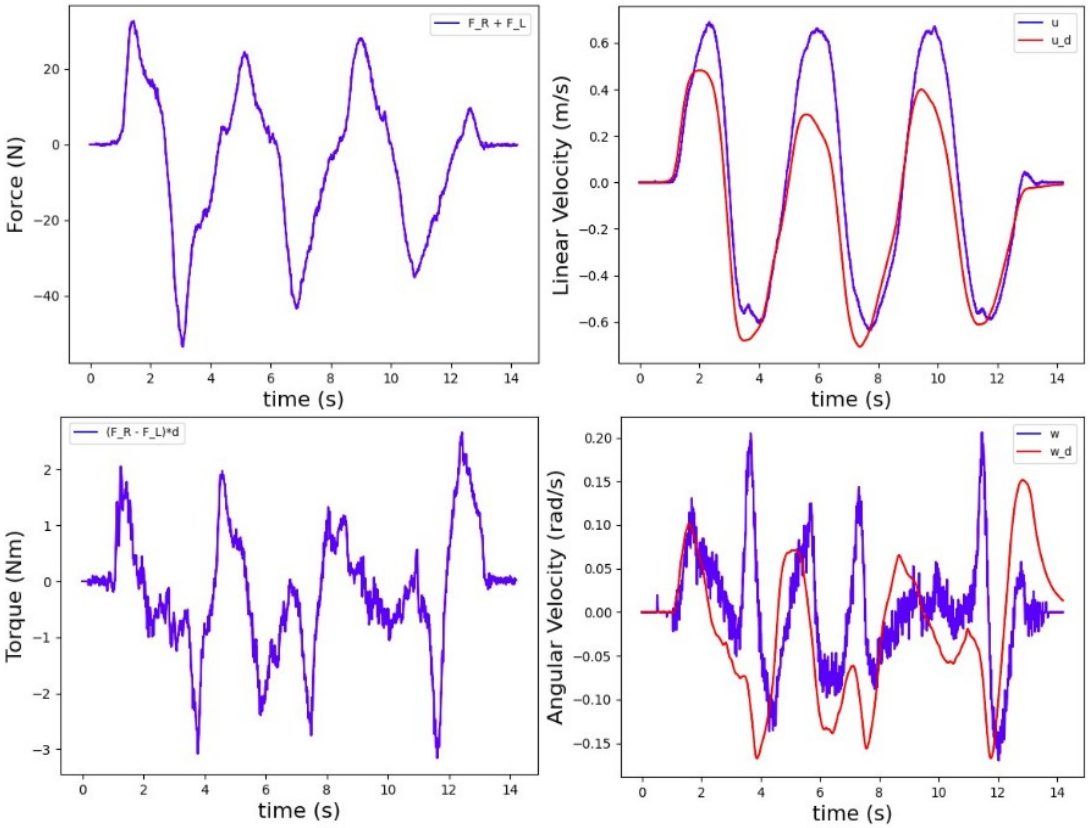


Fig 18. MV: Back-Forth movement with fast speed. The desired linear velocity is lower than the robot's real linear velocity.

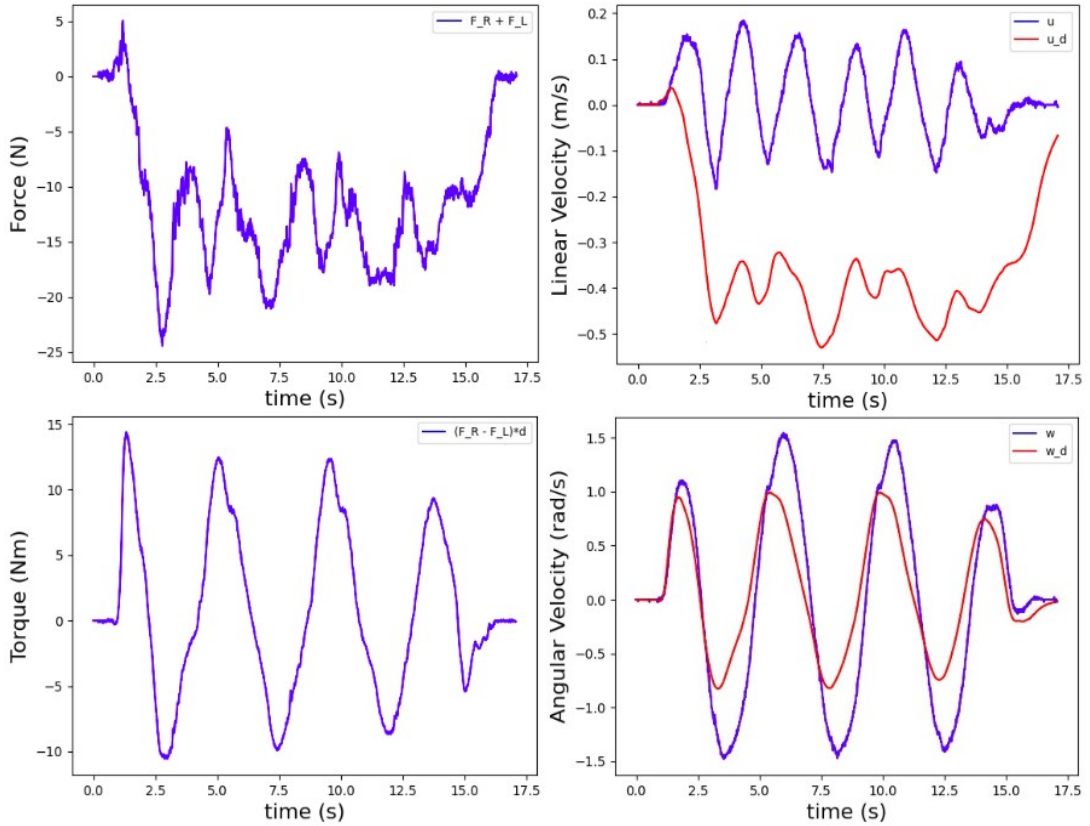


Fig 19. MV: Left-Right turn movement with fast speed. The desired angular velocity is lower than the robot's real angular velocity.

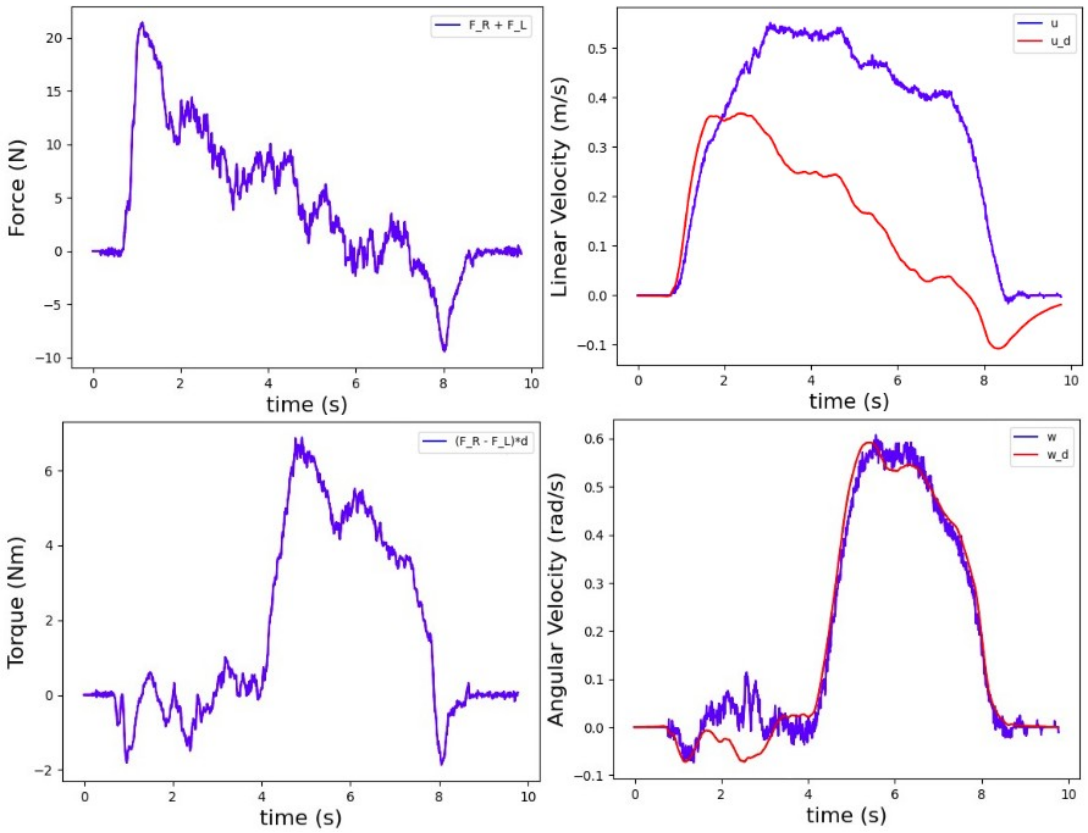


Fig 20. MV: Leftwise L curve movement with medium speed. The desired linear velocity is lower than the robot's real linear velocity. The desired angular velocity is very close to the robot's real angular velocity.

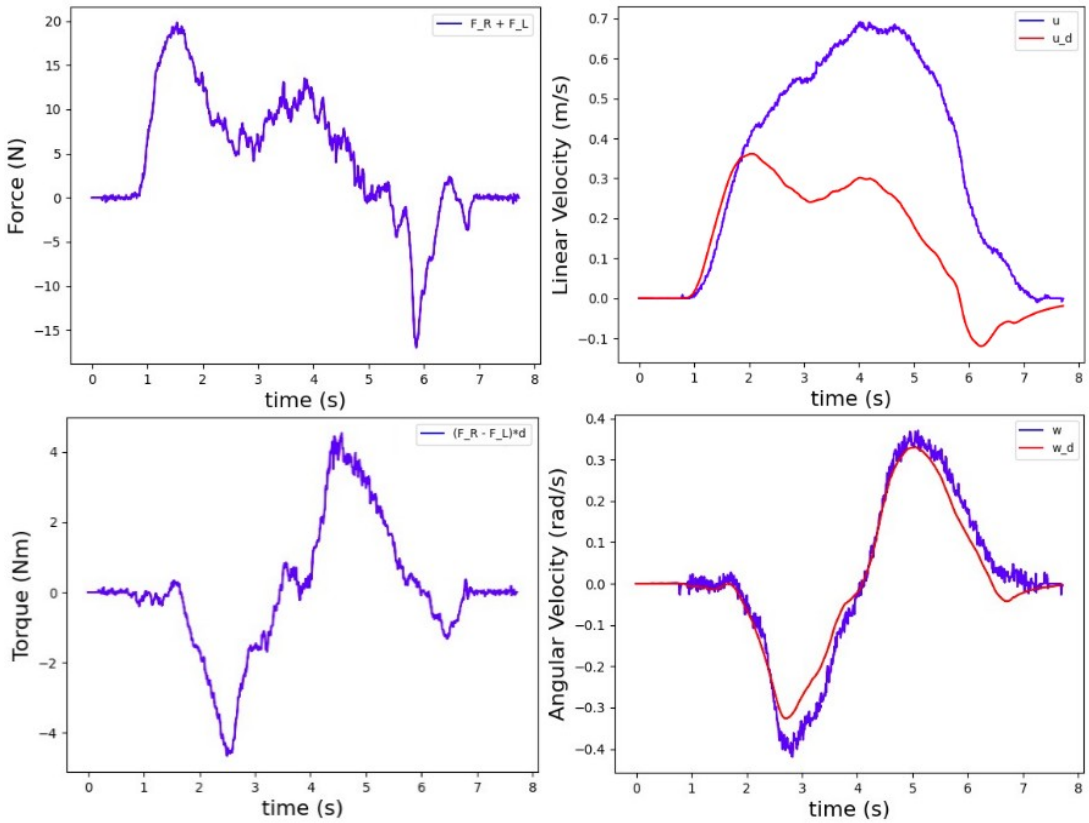


Fig 21. MV: Rightwise S curve movement with medium speed. The desired linear velocity is lower than the robot's real linear velocity. The desired angular velocity is very close to the robot's real angular velocity.

In Figs. 14-19, movements in three different speeds are presented, slow, medium and fast. These are the same movements recorded for learning the dynamic model by driving the unmotorized robot, but also the output of the Admittance Controller is shown for the same force/torque input. In the slow movements (Figs. 14-15) it can be seen that the desired velocity is greater than the observed. For those with medium speed (Figs. 16-17) it is very close and for the faster ones (Figs. 18-19) it is lower. The same conclusions can be derived from the L and S curves, in which both linear and angular motions are happening, with medium-fast speed (Figs. 20-21). For those, the linear desired velocity can not reach the robot's, while the angular desired velocity is very close. This is happening because, as it was observed in section 5.4.1 Tables 2-3, the Least Squares solution gives a linear damping coefficient with large deviation for different movements, while the angular damping coefficient is very similar in any scenario. Thus, most of the angular motions are more accurate than the linear ones. This indicates that the estimated model will always follow an average dynamic behavior and this is the optimal model that can be achieved. In the following chapter this model will be used for the implementation of two different experiment types, one in free space and one in constrained space. In the first case the driving of the robot will be tested, while in the second more advanced assistive features will be analyzed.

Chapter 6

Experiments and Results

In this chapter the results of the experiments that took place will be presented and explained. The structure is divided in two main sections, Free Space Navigation and Constrained Space Navigation. In the first, two scenarios are examined, the driving of the robot with the calculated Admittance parameters and driving with adjusted damping. In the second, four scenarios are proposed to show the effectiveness of the implemented assistive features. Those concern driving on a high curvature path, entering junctions, and adapting the linear damping coefficient. Finally, the findings of the results are summarized in a discussion section.

6.1 Free Space Navigation

The objective of this experiment is to test if the implemented Admittance Controller is suitable for driving the robot. This is a mandatory step in order to later be able to work on more advanced scenarios. For this purpose the human drives the motorized robot in a space without obstacles and constraints, hence this is a Free Space Navigation (FSN). The parts *Test-Drive Space* and *Experiments Space* of the map of Fig. 12 were used, but without considering the walls and obstacles as limitations. The Controller is given by equations (9) – (12) with the parameters of (35), (36). To avoid unwanted input, it is considered that the Controller works only if the user has his hands on the handles of the walker and only if the input forces cross the following thresholds

$$|F_{L,tuple}| = 1.25\,\mathrm{N}\,,\ |F_{R,tuple}| = 1.25\,\mathrm{N}\,,\ |F_{tuple}| = 2.5\,\mathrm{N}$$

$$|\tau_{L,tuple}| = 1.25\,\mathrm{Nm}\,,\ |\tau_{R,tuple}| = 1.25\,\mathrm{Nm}\,,\ |\tau_{L,tuple}| = 2.5\,\mathrm{Nm}$$

Like the previous sections, the basic movements used for the recordings were performed again for the motorized version. Some of the most indicative outputs are presented in Figs. 22-29, organized again in slow, medium and fast movements. The results from this experiment can be seen in section 6.1.1 and in Appendix 2.

6.1.1 Free Space Navigation Results

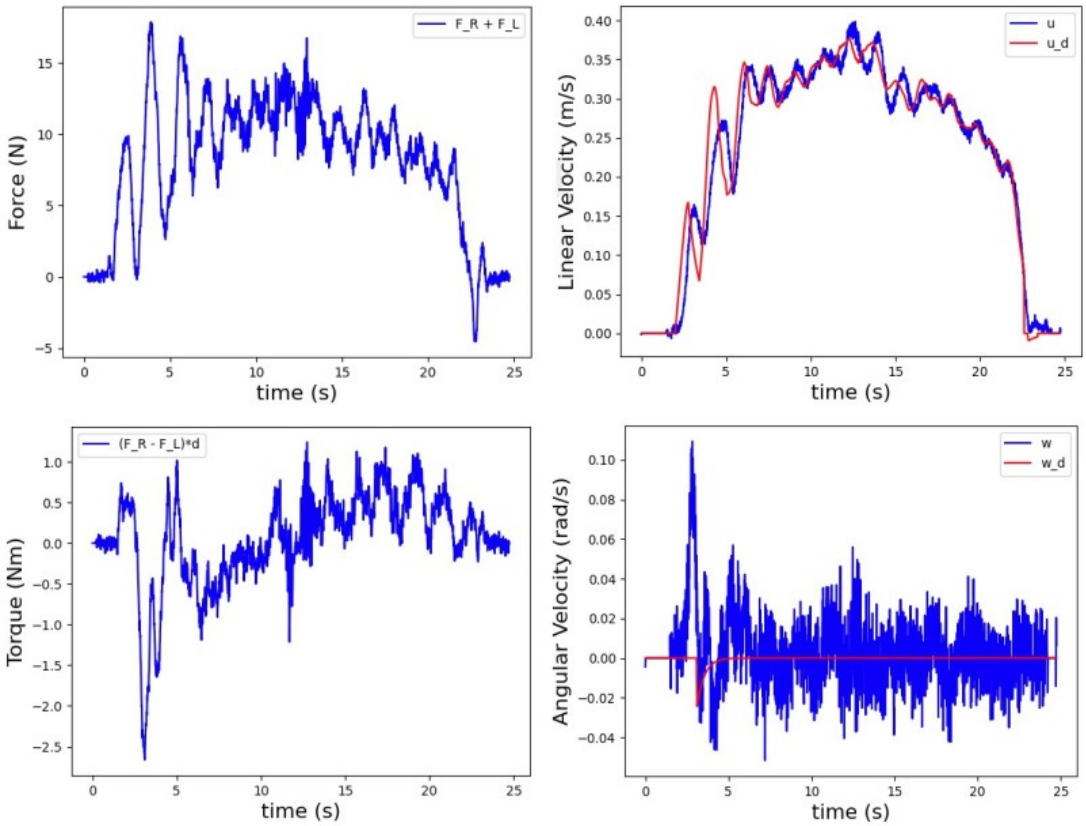


Fig 22. FSN: Slow Forward movement.

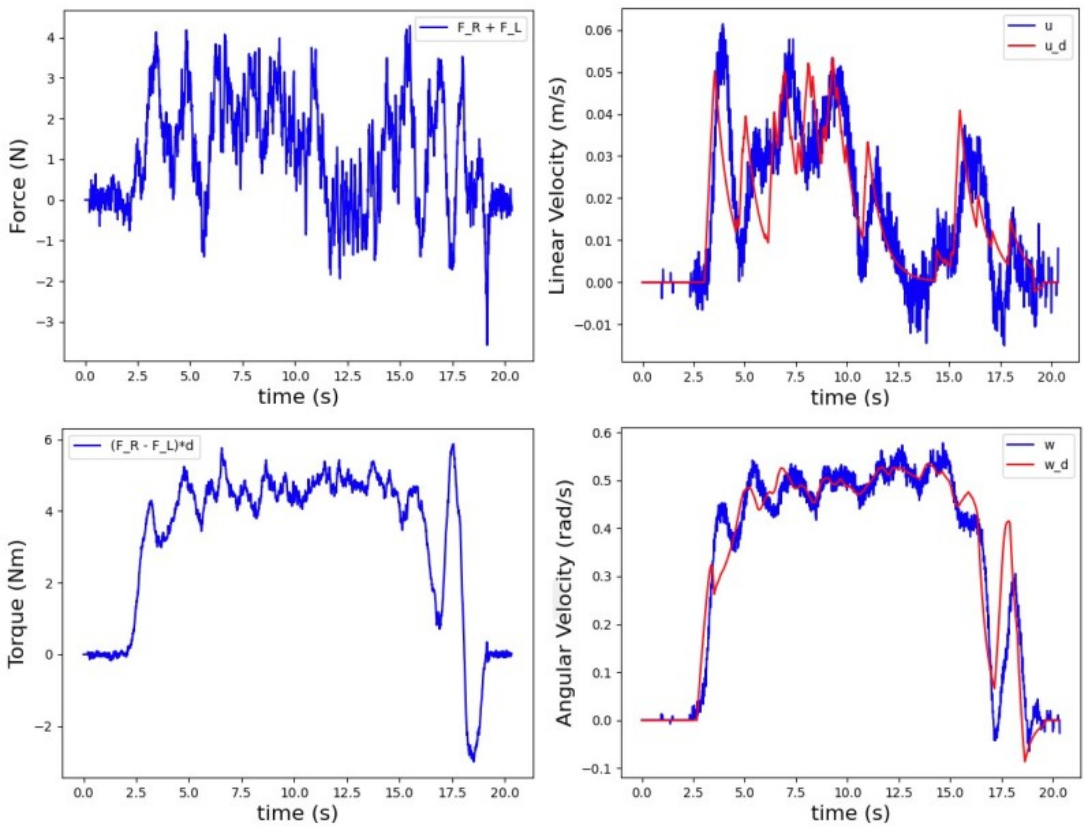


Fig 23. FSN: Slow Leftwise 360º turn movement.

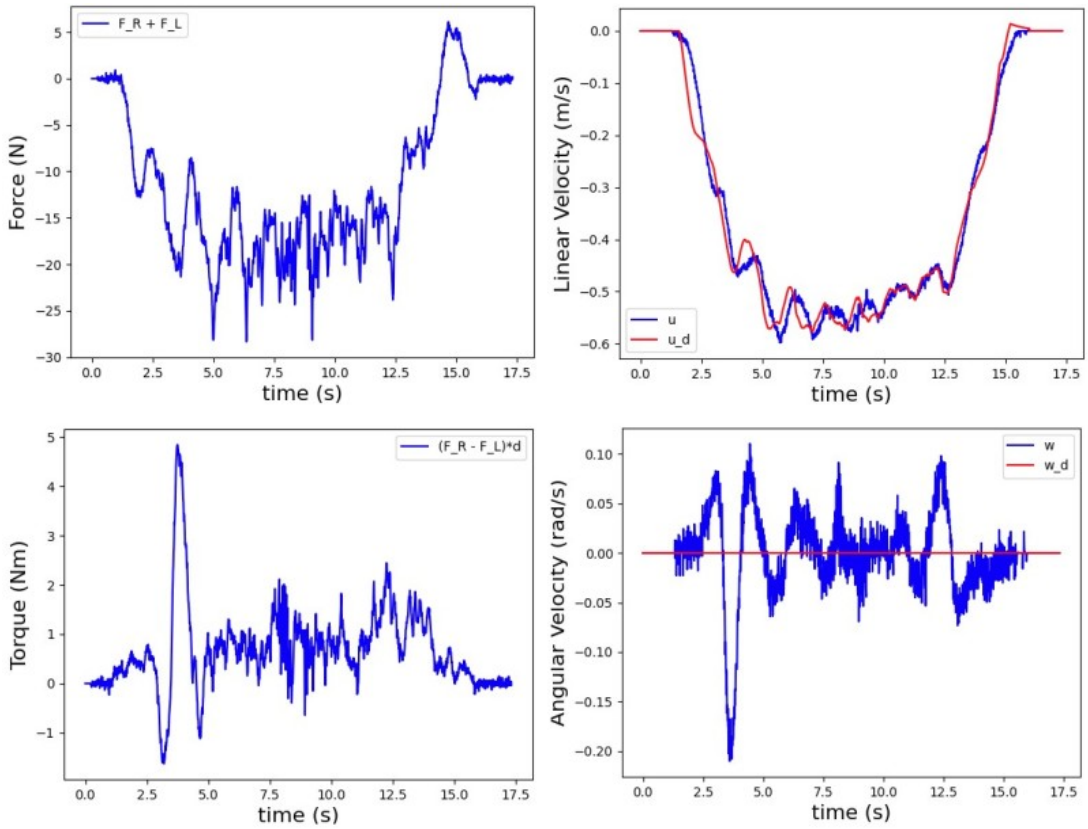


Fig 24. FSN: Backward movement with medium speed.

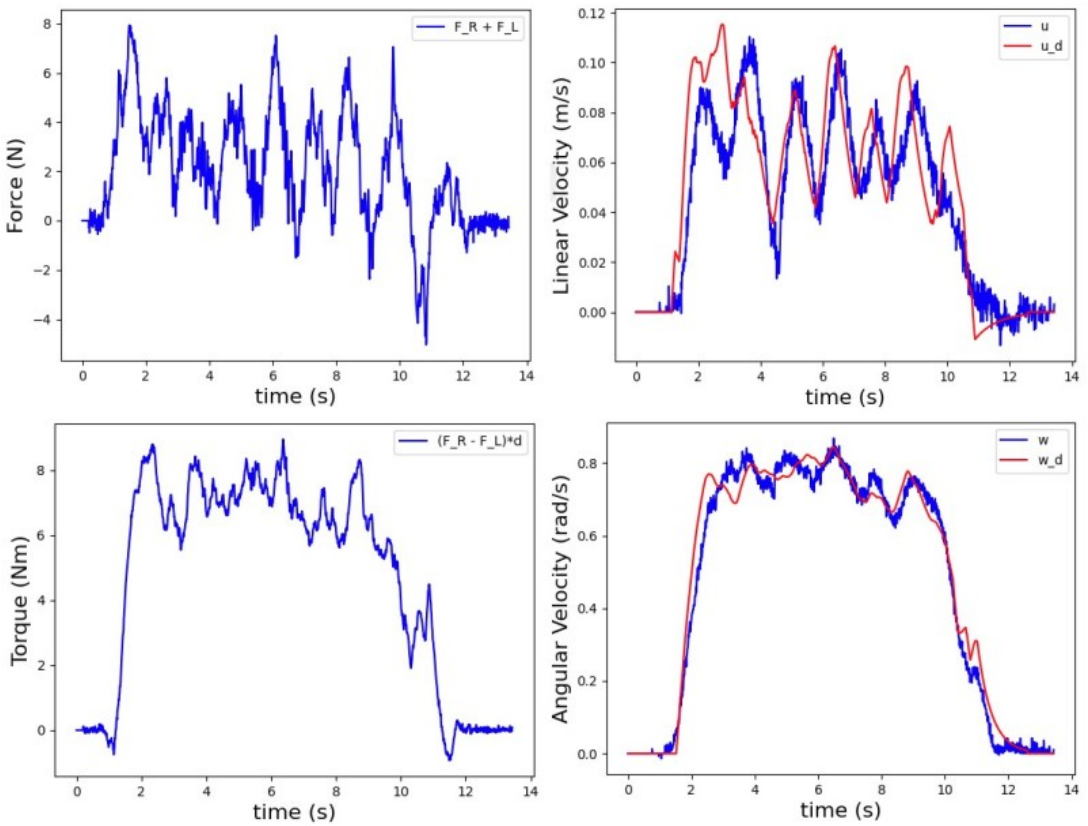


Fig 25. FSN: Leftwise 360º turn movement with medium speed.

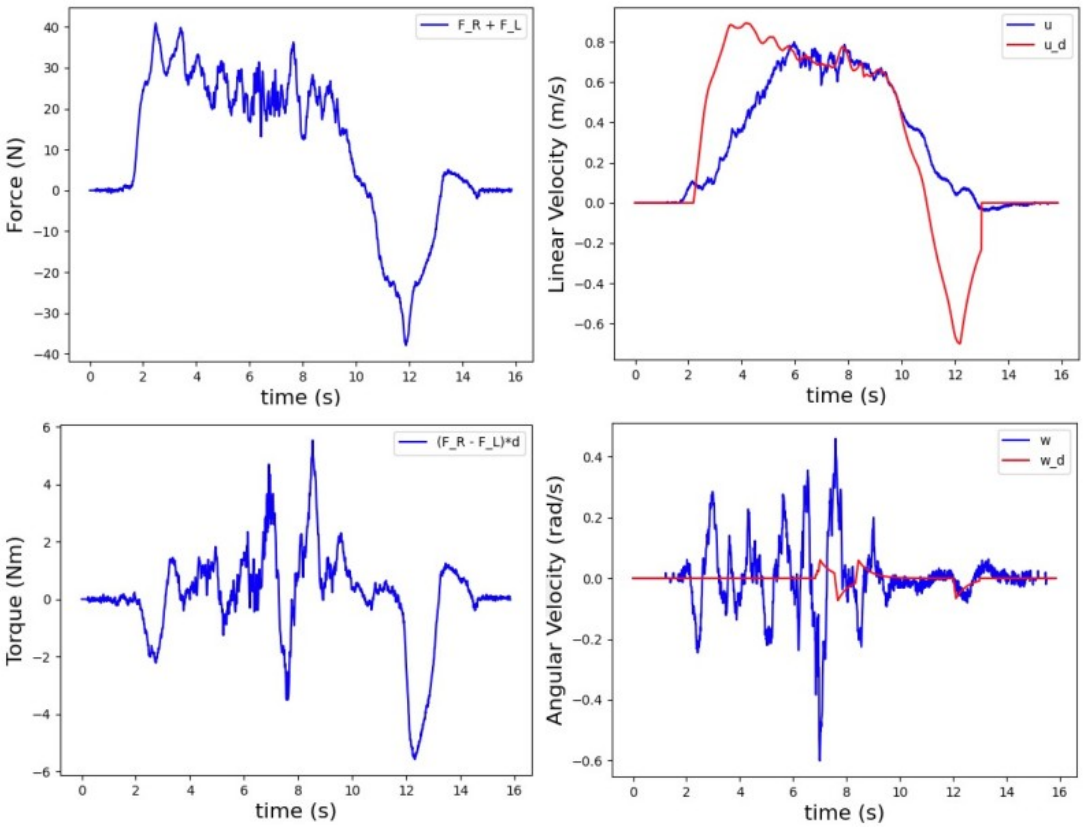


Fig 26. FSN: Forward movement with fast speed

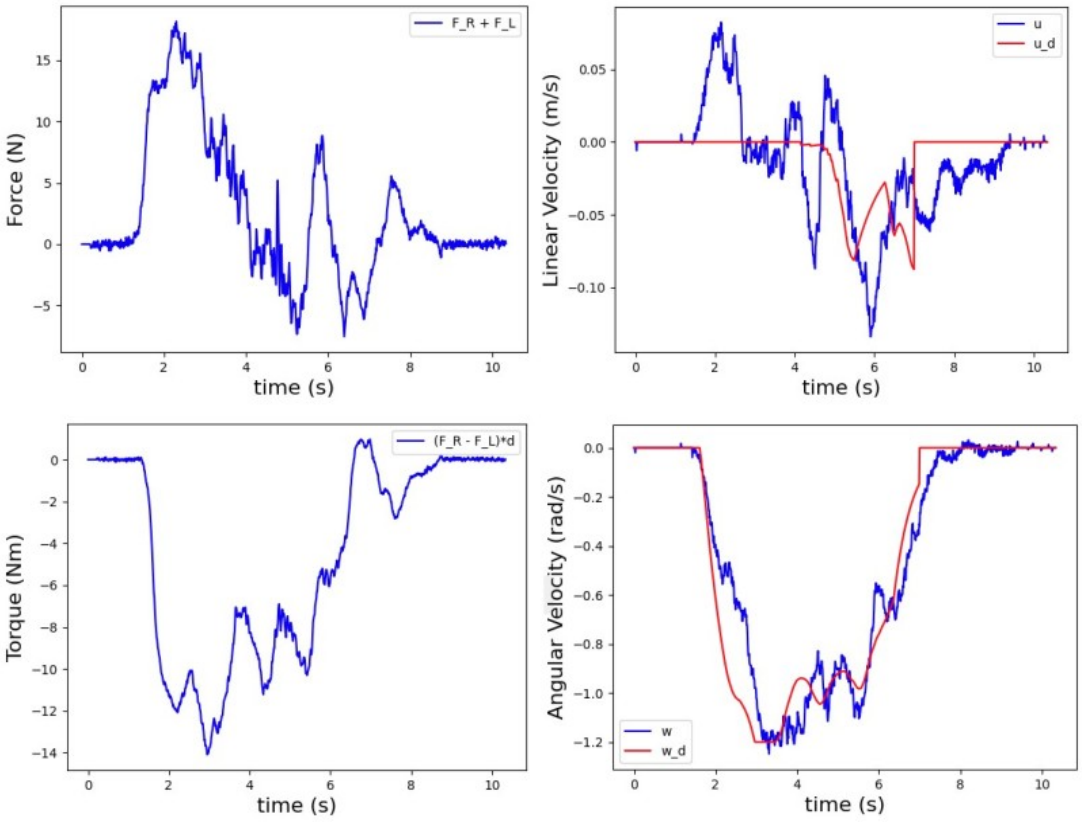


Fig 27. FSN: Rightwise 360º turn movement with fast speed.

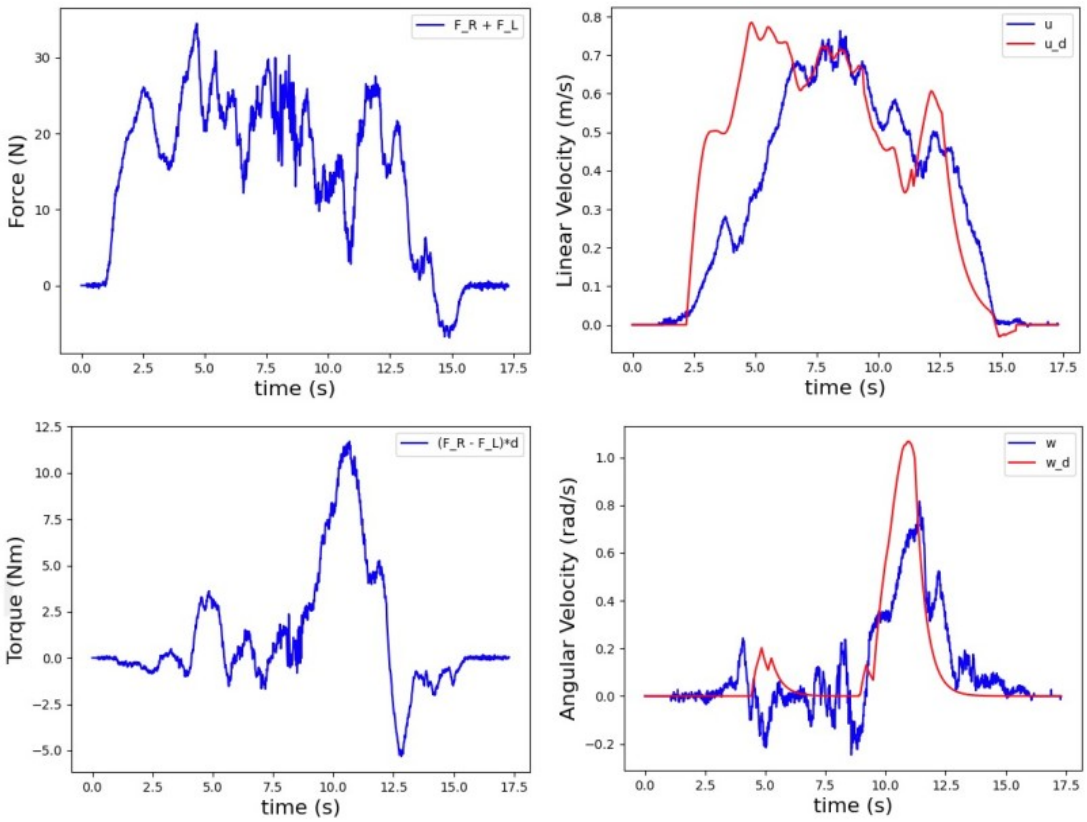


Fig 28. FSN: Leftwise L curve movement with medium-fast speed

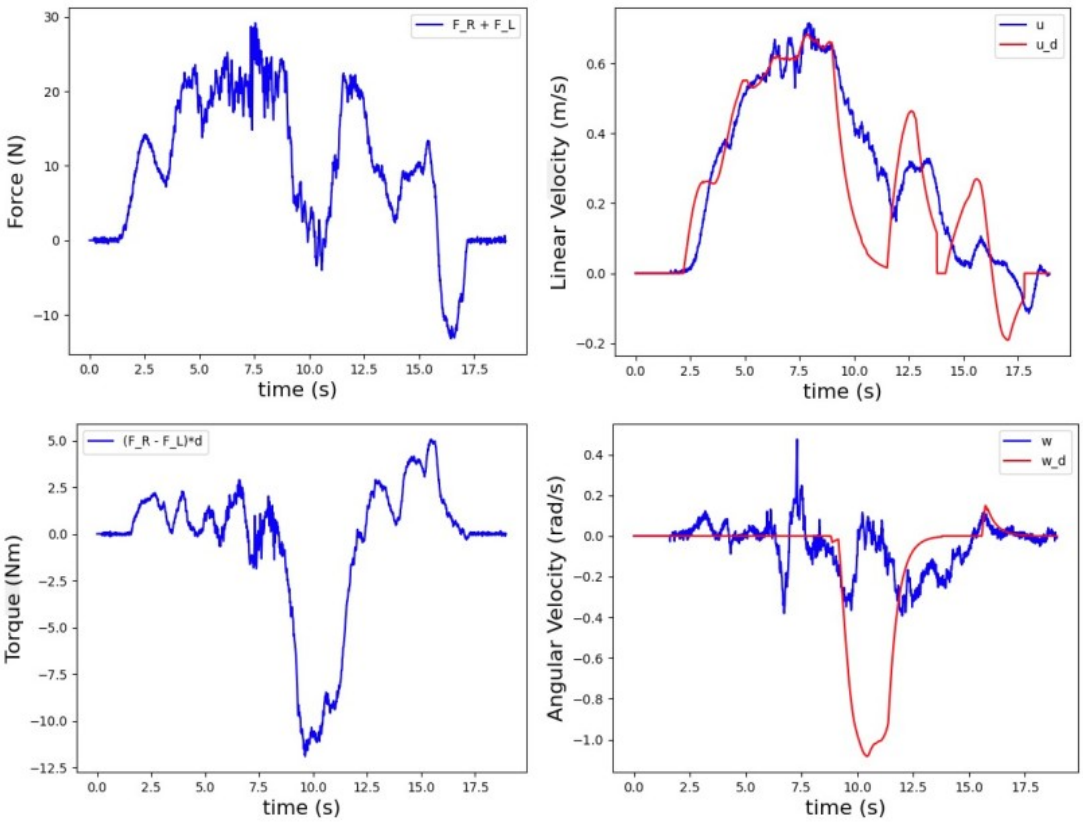
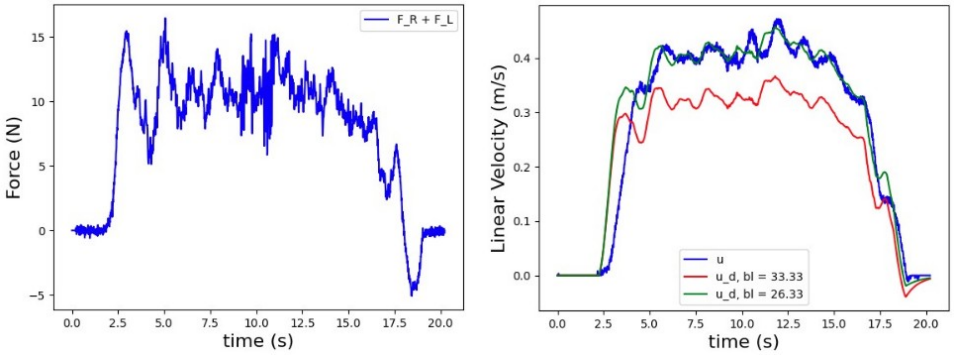


Fig 29. FSN: Rightwise L curve movement with medium speed

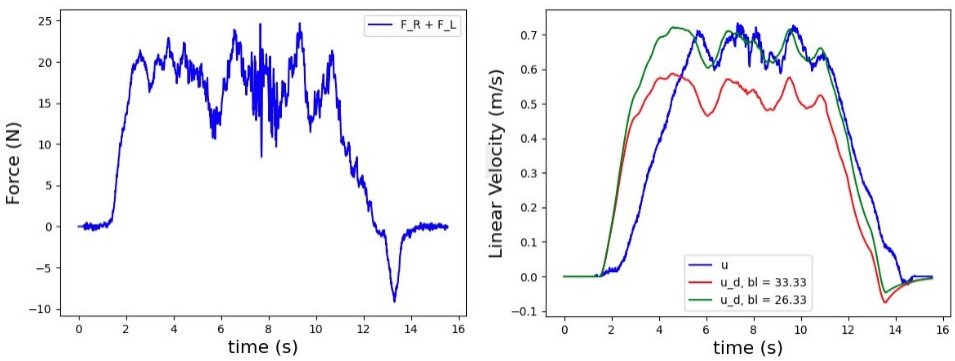
For the linear motions (Figs. 22, 24, 26), in slow (Fig. 22) and medium (Fig. 24) movements the real and desired linear velocities are very close, while in the fast (Fig. 26) movement the robot can not always reach the desired velocity due to low-level motor control limitations. In addition, in these movements even if the desired angular velocity command is almost zero, the real angular behavior indicates that the user either wins the motor and manually turns the robot slightly or the wheels slide or stop rotating for an instance. In contradiction, for the angular motions (Figs. 23, 25, 27), in every scenario real and desired angular velocities are identical. In Figs. 28-29, it can be seen from the left/rightwise L curves that when both linear and angular motions are taking place the accuracy of the system decreases. The robot velocities do not always follow the desired ones. In the left turn an overall acceptable behavior is observed, but in the right turn it can be seen that when the user tries to turn, the handling of the linear behavior gets harder, while also the robot does not follow the angular command. This may happen due to poor user performance or poor low level wheel-motor architecture. To make the turn easier the torque thresholds could be reduced, but this would result in unwanted sensitive turn behavior, especially in only linear motions like forward or backward.

Overall, it can be seen that in abrupt velocity changes the system responds with a small delay resulting in confusing the user and make him apply greater forces, which will later become greater velocities, and thus losing control of the vehicle. So in conclusion, the total system works best for small velocity changes, thus in slow and medium movements where the forces applied are not very strong and abrupt, while faster movements can not be achieved due to low-level constraints. Also, the angular model is very accurate when there are no linear motions, otherwise the performance weakens.

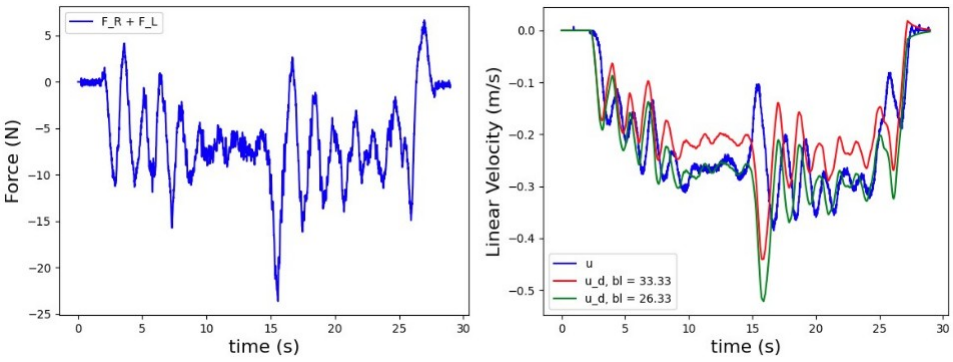
6.1.2 Free Space Navigation Results with Adjusted Damping



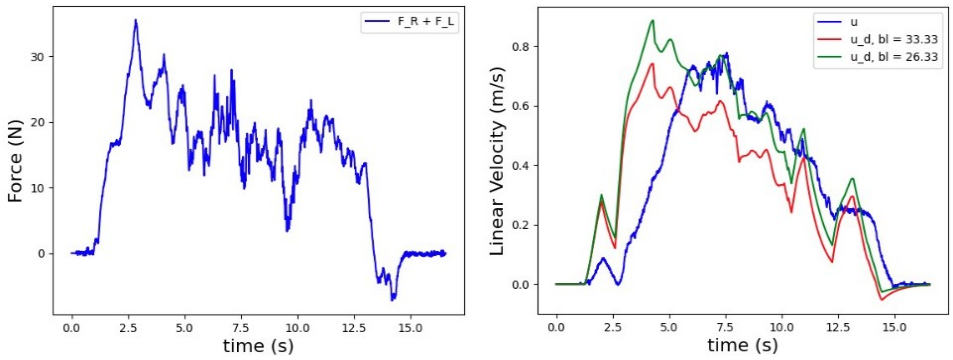
(a)



(b)



(c)



(d)

Fig. 30. Comparison between the adjusted linear damping ($b_l = 26.33$ Ns/m, green*)* and the previously estimated one ($b_l = 33.33$ Ns/m, red). The u corresponds to the velocity resulting from the adjusted damping (blue). (a) Forward slow. (b) Forward medium. (c) Backward slow. (d) Leftwise L turn medium.

In practice, the overall system felt a bit slow in the linear movements, so to try to make the robot more easy to use the linear damping parameter was manually adjusted slightly to fit the needs of the current user. Thus, the comparison between the old ($b_l = 33.33$ Ns/m) and the new ($b_l = 26.33$ Ns/m) linear damping parameter is presented in Fig. 30. The old Admittance model linear output is shown in red, while the new in green. The blue one is the actual velocity of the robot resulting from the adjusted damping. As shown in the previous section, if the old parameter was used the real velocity would follow the red curve, while now it follows the green. It can be observed that for the new damping parameter the robot becomes a little more fast in its linear movements and that is suitable for the experiments to follow, as the driving is easier and the robot feels lighter. This parameter could be further regulated according to the application, the user needs, or the target user base.

6.2 Constrained Space Navigation

In this experiment, the effectiveness of combining the Admittance Controller and the Arc-Line Path Planner will be tested. To do that some assistive features were implemented, based on the output paths of the planner, and tested in a constrained space environment, meaning that it is limited by walls and obstacles. For that purpose the part *Experiments Space* of the map of Fig. 12 was used.

The Navigation of the robot is consisted on its core of four different states, that were previously defined and explained in section 4.5. The robot velocity is commanded by the

Controller of equations (28), (29), which is based on these states. In addition, the following adaptive linear damping is considered

$$b_l = \begin{cases} 26.33 & , S \in [S_1, S_3] \\ 56.33 & , S = s_2 \\ 1033.33 & , S = S_4 \end{cases} \quad (\mathrm{Ns/m}) \tag{37}$$

The proposed shared human-robot control and adaptation strategy is defined by equations (28), (29), (37). When in Normal Motion and the angular velocity input is inside the cluster limits, while the cluster span is wide, the human drives the robot as in free space. If he crosses the minimum or maximum velocity, the limit values are picked. If the input is specifically zero or the span is tight, the robot guides the user to the correct direction. If the span is of medium size the shared path is followed.

When in Observing or Assistive Motion, the angular velocity is fully controlled by the robot, which follows the curvature of the optimal path of the selected cluster, proportionally to the desired linear velocity input given by the human. Specifically in Observing Motion, where there are multiple clusters, like in a junction, the linear damping parameter is increased more than double its previous value, meaning that the robot becomes suddenly more resistive. This adaptation notifies the user that there are multiple routes available and he needs to pick one. The specific value of $b_l = 56.33\ \mathrm{Ns/m}$ is picked experimentally, as it was observed that it scales down the desired velocity around 50%. When in Idle, the human fully controls the angular velocity of the vehicle, but scaled down by a factor, since in this situation a reduced span of the accessible clusters might signal a very tight, cluttered or obstacle-laden space. Thus in addition, the linear damping parameter is increased so much that the desired linear velocity output of the Admittance model is always zero. Again the value of $b_l = 1033.33\ \mathrm{Ns/m}$ was experimentally observed to scale down the velocity around 100%. To test all of the above in practice, the following scenarios where considered:

1. Directing on the wall diagonally without torque application
2. Directing on the wall diagonally with torque application
3. Intention detection in T-junction
4. Function near wall

In the following sections the results from these four scenarios are presented and explained. All of these scenarios were executed in a slow-medium pace where the system works best.

6.2.1 Directing On The Wall Diagonally Without Torque Application

In this scenario the objective is to show the Assistive Motion functionality. The user starts walking with an initial angle of approximately 45^{o} degrees directing on the wall and pushes the robot towards it without applying torque to turn the robot at any time. As he gets nearer to the wall the curvature of the feasible paths increases and becomes greater than a defined threshold, resulting in the robot getting in the Assistive Motion state. The threshold is defined as

$$\varphi_{assist} = 0.1 \cdot \varphi_{max} \tag{38}$$

meaning that when the angle of the selected path exceeds 10% of its maximum value the robot gets in Assistive Motion state. When this happens, the robot has the full control of the angular

velocity until the curvature of the selected path becomes again lower than the threshold. Thus, when the user pushes the vehicle towards the wall it turns properly, guiding him so the collision is avoided without the user applying torques or performing hazardous maneuvers.

In Figs. 31-32, two executions of this scenario are performed, one in slow and one in medium pace. In the first part of each figure, the actual motion that happened on the map is illustrated in four snapshots, so that the produced clusters and selected paths are shown. The clusters are organized in two groups, the far and the near, and they are colored with green, red or blue properly so that they can be clearly visible. Moreover, the cluster path is shown in purple and the shared path in yellow. If only one of those two is visible, it means that they are aligned. In the second part, one can see the force/torque inputs, the real linear/angular velocity of the robot with blue color, the Admittance model outputs with red color and the Controller angular velocity output with green. It can be observed that, in both executions, the user applies very little amount of torque, which is under the torque tuple ($|\tau_{tuple}| = 2.5$ Nm), resulting in the Admittance model giving $\omega_d = 0$ r/s. Thus, the Controller by itself, gives the proper angular velocity command so the robot can follow the selected path, even if the user is not trying to turn. This behavior can be seen in Fig. 31 between 11s-24s and in Fig. 32 between 9s-16s. In the first execution, the real velocity is more accurate, with respect to the commanded, while in the second there is a bit of delay because the motion is more abrupt.

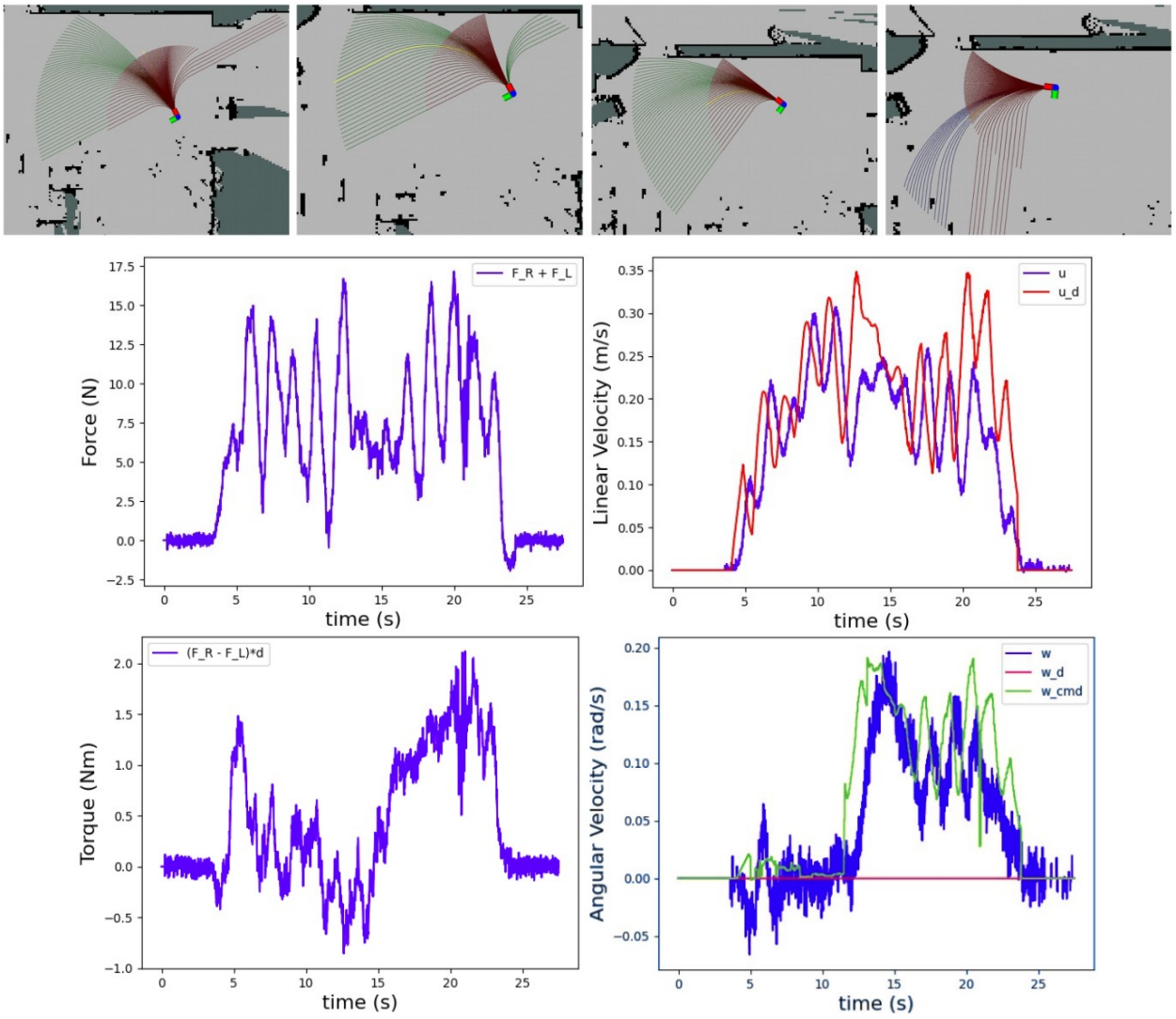


Fig. 31. Execution No. 1: Directing on the wall diagonally without torque application. Slow speed.

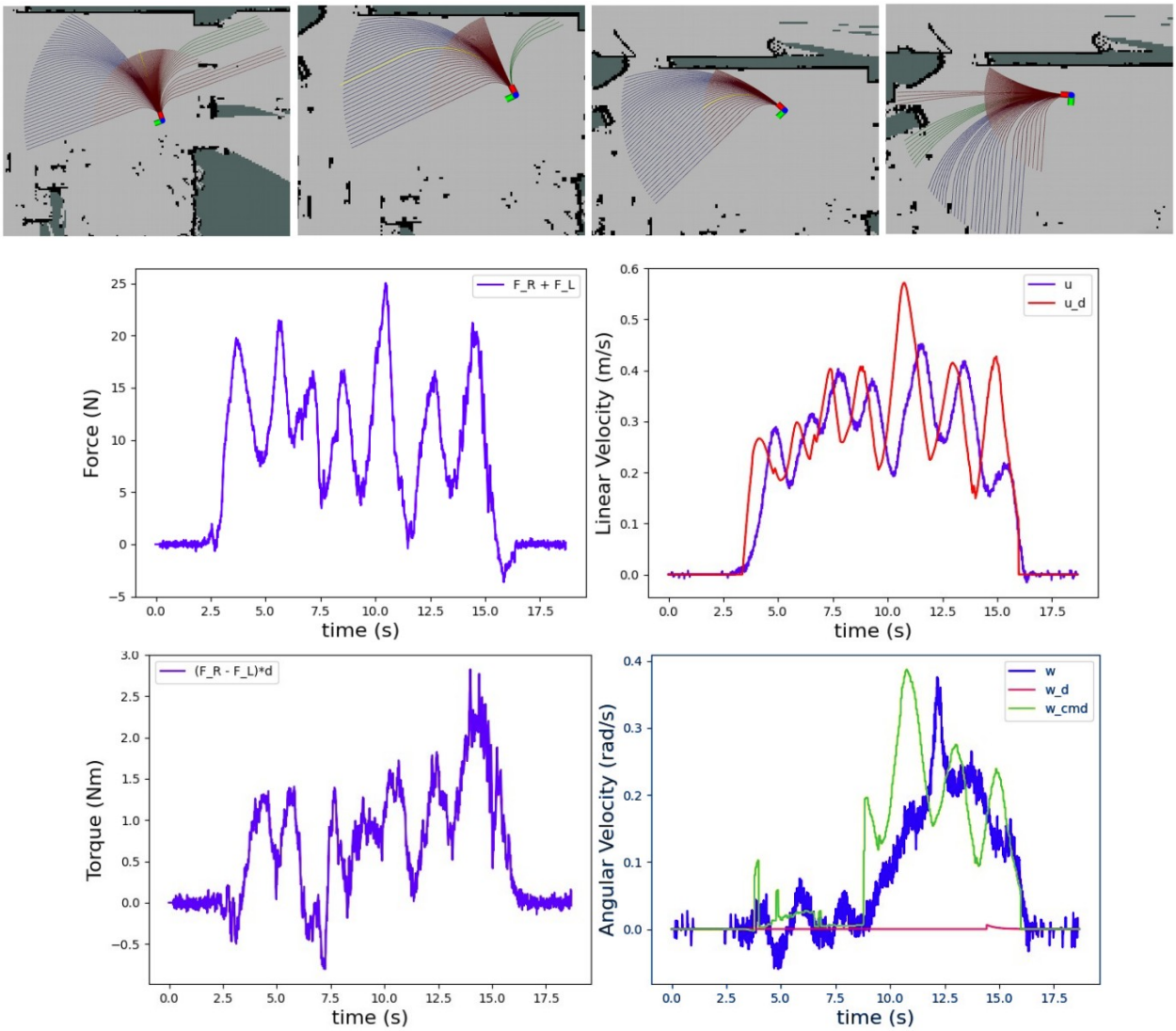


Fig. 32. Execution No. 2: Directing on the wall diagonally without torque application. Medium speed.

6.2.2 Directing On The Wall Diagonally With Torque Application

This scenario is the same as the previous one except that the user applies torques while the robot is in the Assistive Motion state, in order to get out of the guidance path. Those torques are not taken into account and the robot guides back the user into the correct path until the turn or the assistance is completed. In Figs. 33-34, two executions of this scenario are shown. It can be seen in both that, when the user applies torque, the Controller does not follow the Admittance model. On the contrary, the proper angular velocity command is given in order to minimize the divergence from the correct path. From the commanded angular velocity of the robot, one can see that when the user tries to get the robot out of its way, it applies opposite velocity so it can return to the guidance path. In Fig. 33 the assistance begins at $t = 9$s and the robot follows the output curvature until $t = 12$s. Then, between 12s-15s and 18s-22s the user applies torques in both directions to get out of the path with no effect. In the rest of the movement the output path is followed. Similar is the outcome shown in Fig. 34, where the assistance begins again around $t = 9$s and the torques are applied between 13s-15s and 19s-22s.

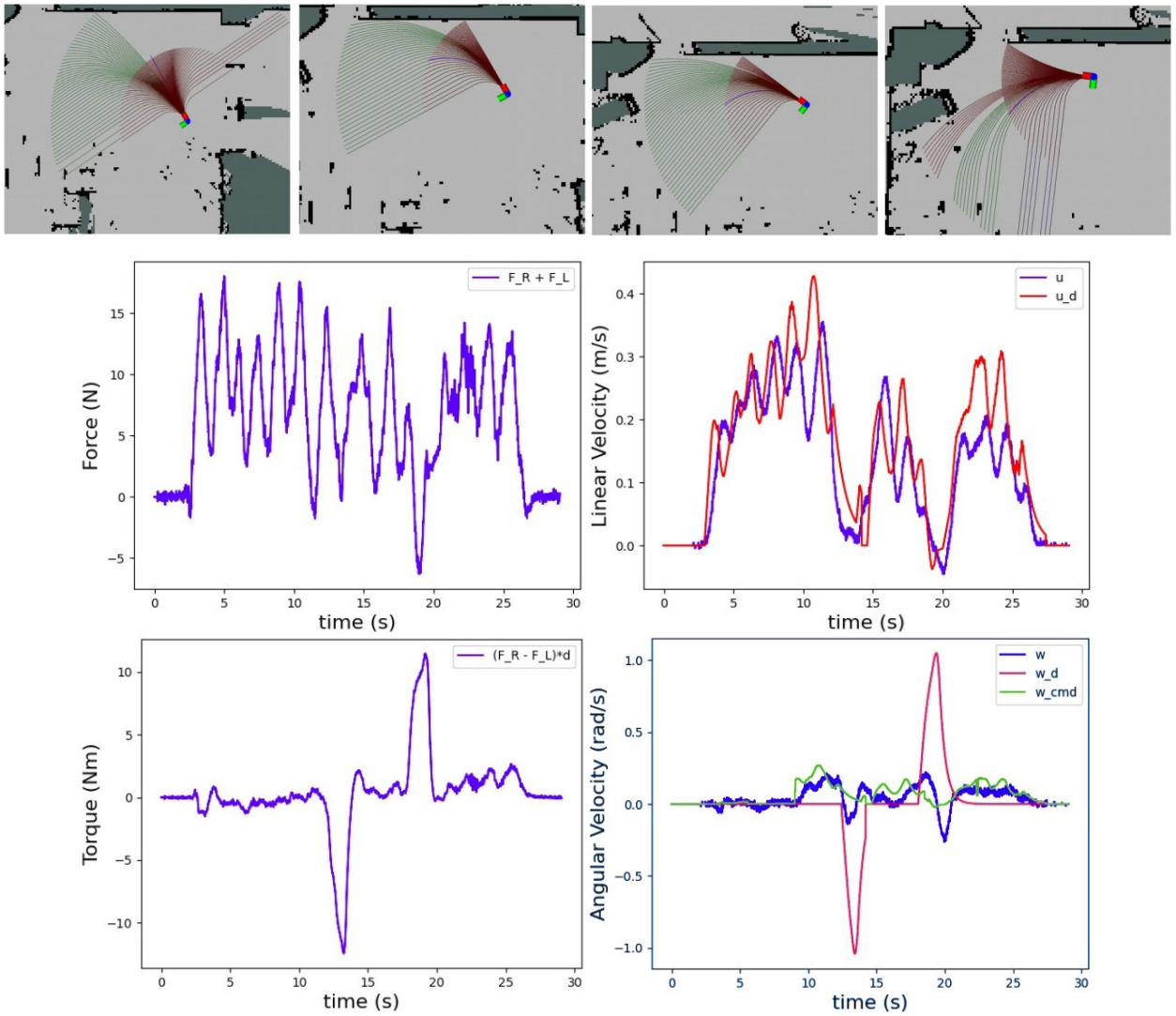
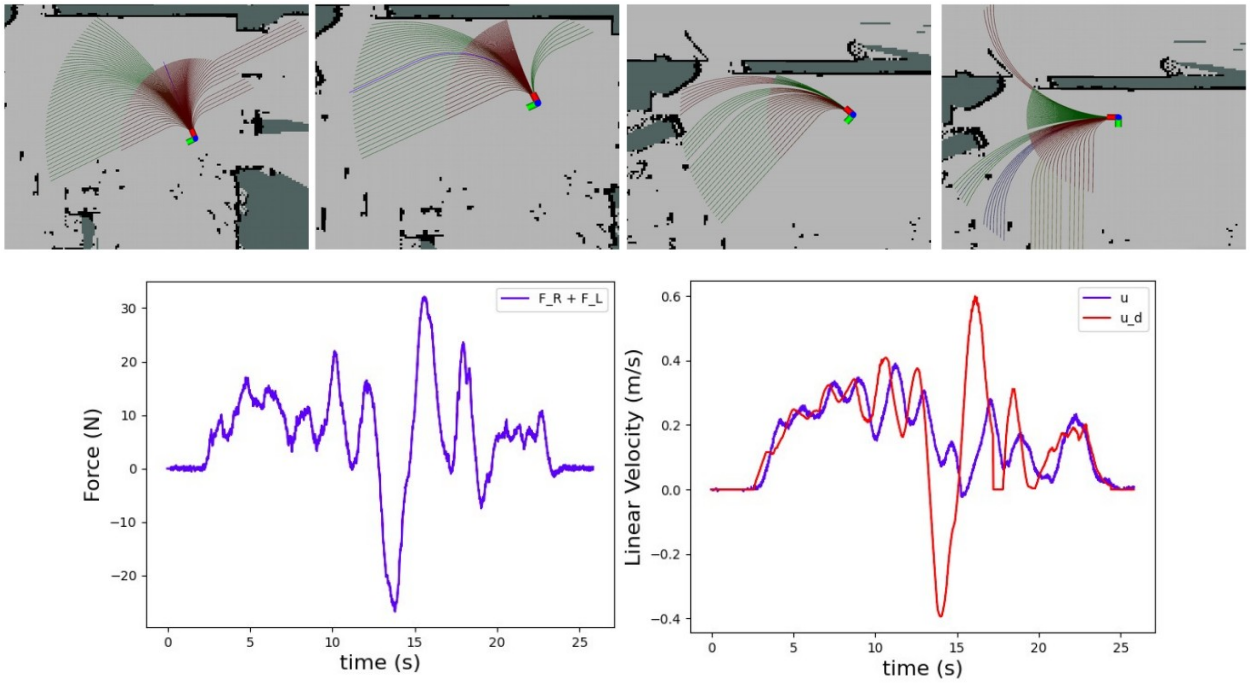


Fig. 33. Execution No. 1: Directing on the wall diagonally with torque application.



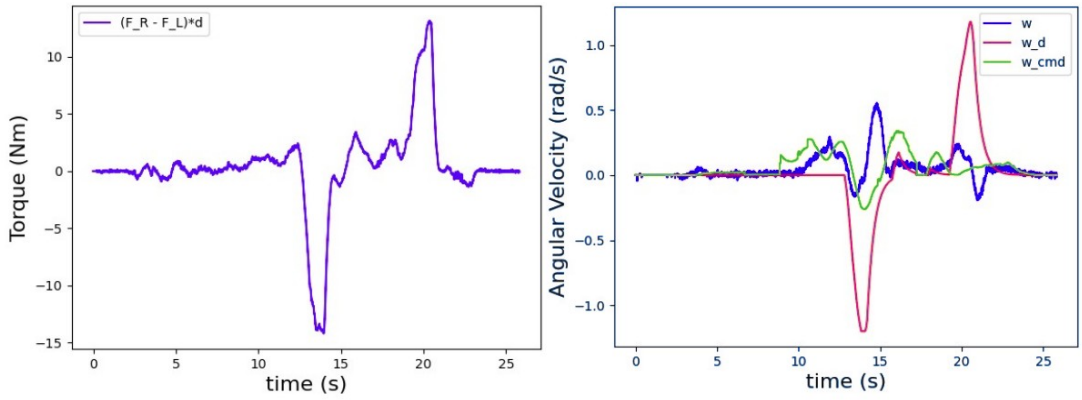
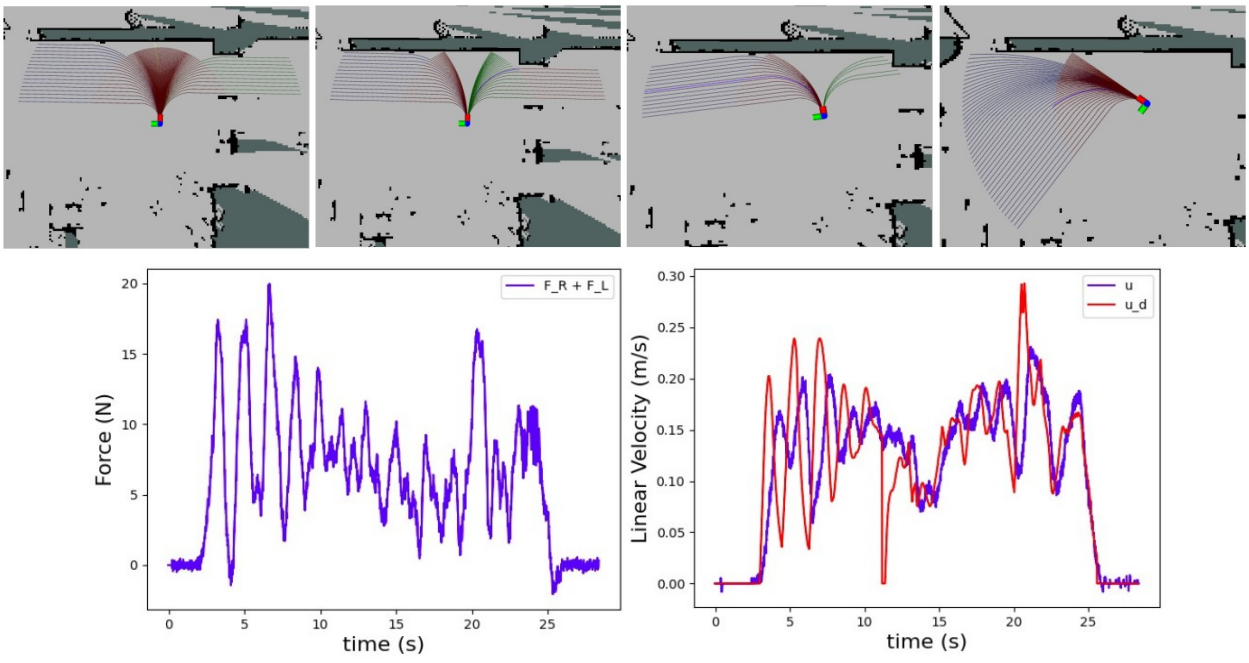


Fig. 34. Execution No. 2: Directing on the wall diagonally with torque application.

6.2.3 Intention Detection In T-Junction

This specific scenario is proposed to examine the method of selecting one among multiple clusters to follow through a cluster voting mechanism, that was presented in section 4.5, and thus solving the undecidability problem. The user enters a T-junction, where there are more than one clusters at least for one group, and the robot gets in the Observing Motion state. While in this state, the output of the Admittance model ω_d is not used as angular velocity command but for the voting of the clusters. The angular velocity is controlled by the robot and the linear by the human through the Admittance model, but with increased linear damping ($b_l = 56.33$ Ns/m). The robot observes the human intention for 5s at most and gives votes to the clusters depending on which one is closer to φ_h, the angle the human desires to have, which is proportional to the desired angular velocity ($\varphi_h = 1.2\omega_d$). In case where $\varphi_h = 0$, priority is given to the cluster with φ_C closer to the robot's angle φ_R. The voting value occurs from equation (24), where $a = 0.1\varphi_{max}$ and $b = 0.8\varphi_{max}$. As explained in section 4.5, when the vote value of one cluster is sufficiently higher than the rest of the clusters, the robot stops observing and selects it.

In Figs. 35-36, the experimental results from two execution of the T-junction scenario are



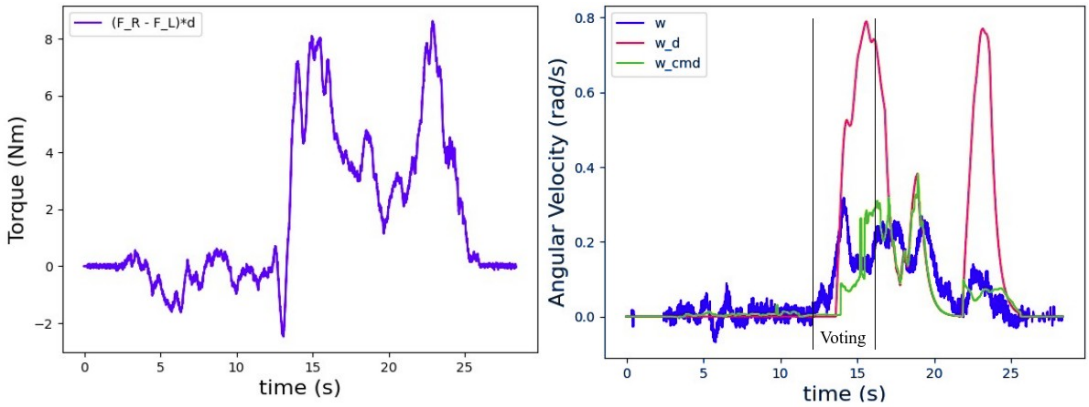
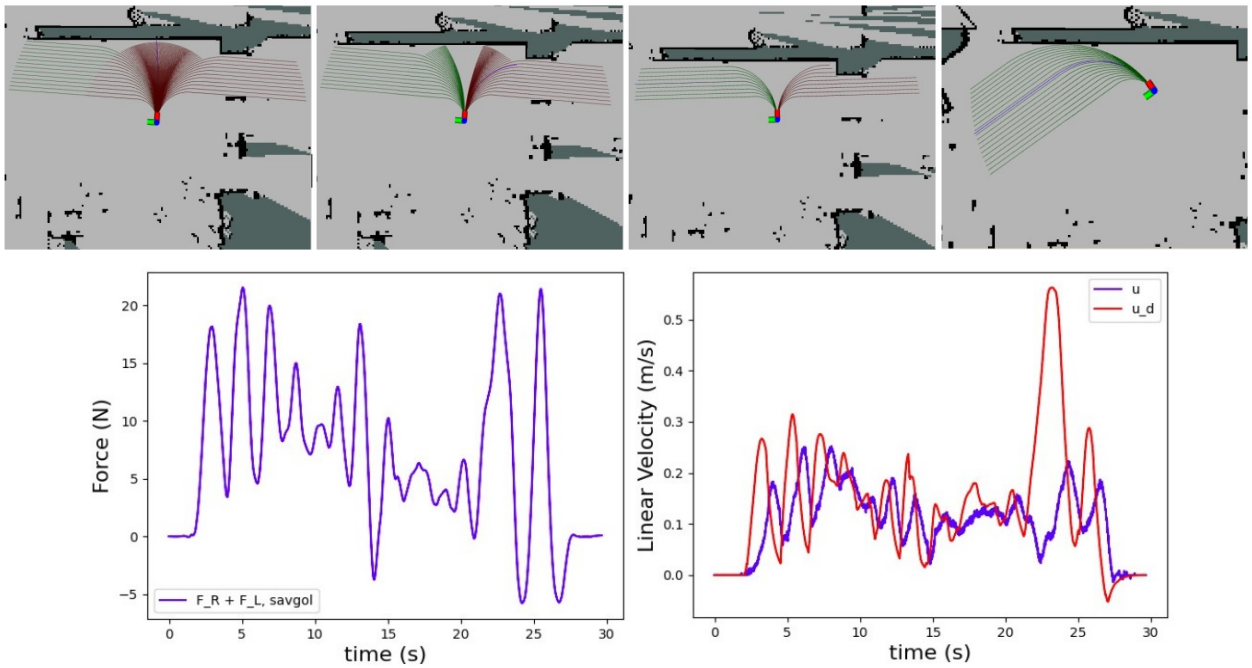


Fig. 35. Execution No. 1: Intention detection in T-juction.

presented. From the snapshots of these figures it can be seen that the robot enters the junction without knowing the desired direction, so in the beginning it picks the cluster that is most likely to be selected according to the robot's orientation. Until the undecidability is resolved, the robot follows the curvature of that cluster. After a few seconds, high intention for left turn is detected so it picks the left cluster. In Fig. 35, the Observing Motion begins at the beginning of the movement as there are three available clusters, front, left, and right. In the beginning the front cluster is picked as it is closer to the φ_R and the undecidability is resolved. When the front cluster is divided in the left and right cluster, as shown in the second snapshot, the right cluster is selected for a moment. However, as the robot goes forward the left cluster is picked and it follows its path until the undecidability is resolved. This can be seen between 12s-16s, where the voting procedure takes place. The user applies positive torque to show his intention to turn left. After $t = 16$s, the left cluster wins the voting and the desired velocity becomes the same as the commanded, as the robot goes back to Normal Motion state.

In Fig. 36, the situation is similar. Here when the front cluster splits, the right cluster is selected for more than a moment. This is visible between 12s-14s, where the robot applies negative angular velocity command. This command is not followed because of the low-level control problems in the right turns. However, later because the right cluster becomes smaller, the left one is picked again. The left cluster's path is followed as the voting takes place between 12s-17s, where the user applies again positive torque to turn left. After this intention is detected,



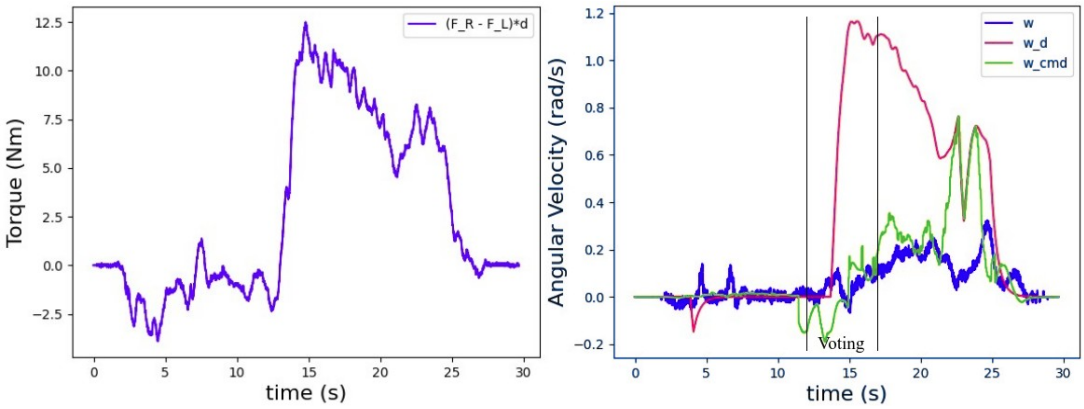
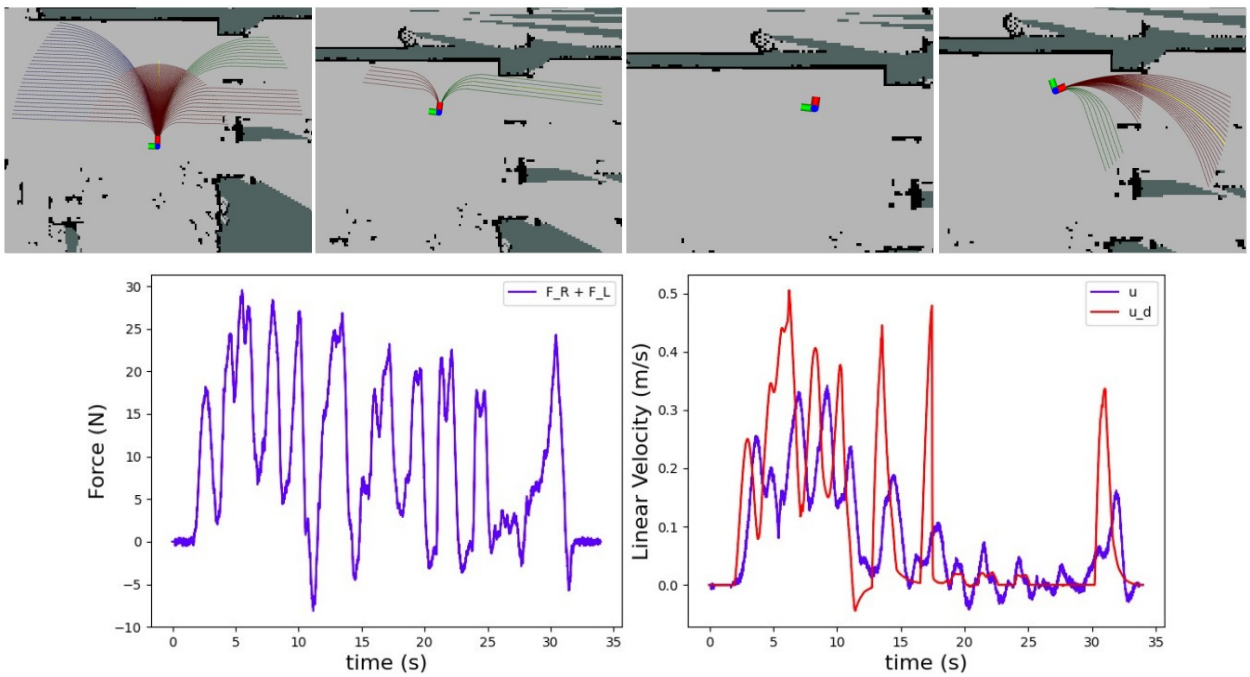


Fig. 36. Execution No. 2: Intention detection in T-juction.

the undecidability is resolved, the robot gets in the Normal Motion state and it follows the desired velocity commanded by the user through the Admittance model, which is filtered by the navigation Controller.

6.2.4 Function Near Wall

For this final scenario the objective is to test how the robot works while in the Idle state. To do that, the user tries to push the robot to hit the wall, but this is not possible as the Path Planner does not return feasible clusters of motion and hence the robot gets in the Idle state. When this happens the linear damping coefficient is highly increased ($b_l = 1033.33$ Ns/m), and the only possible way of movement is for the human to rotate the robot statically, until clusters can be produced. The results can be seen in Fig. 37. In the duration 0s-20s a forward movement directing on the wall with a small negative angle is taking place. In 20s-25s it can be observed that even after forces are applied the linear velocity output of the Admittance model is $u_d = 0$ m/s, because of the huge damping parameter increase. For $t > 25$s, the user turns the robot to the right, paths are generated and the robot gets out of the Idle state.



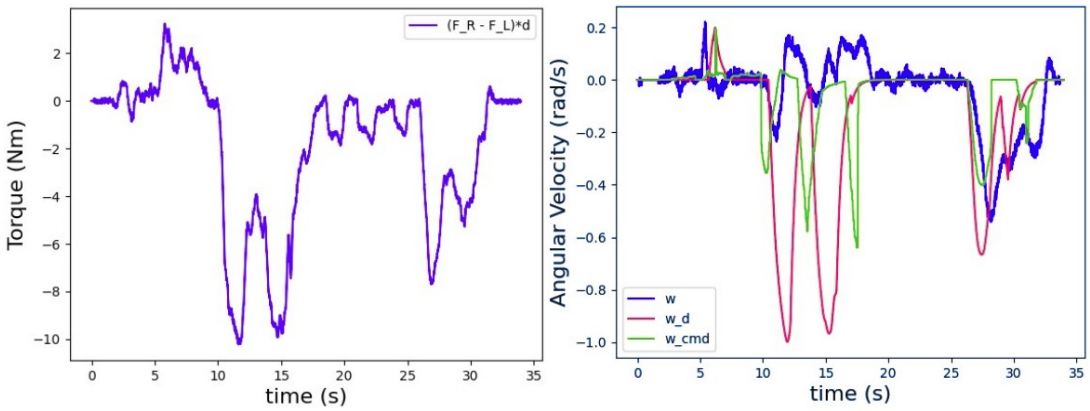


Fig. 37. Execution No. 1: Function near wall.

6.3 Discussion

Overall, everything worked fine but the effectiveness of the methods was limited by the robot's architecture constraints. Every scenario was executed in slow-medium speed where the motor control system worked best. The only noteable problem was that the left rear wheel was sometimes sliding and thus undesired oscillations could occur.

In Free Space Navigation, the robot had the average dynamic behavior, compared to the observed one, and worked best in slow and medium speed movements. Faster or abrupt movements could not be achieved due to low-level control limitations. In this case, the noteable oscillations could make the human-user lose control of the rollator. Furthermore, it was observed that the angular dynamic model is more accurate than the linear, because the angular damping coefficient had much lower deviation in the learning process. Adjusting the linear damping parameter to a slightly lower value made the system a bit more explosive and easier to use. It is noted that depending on the target group or the application this parameter could be further adjusted to achieve better behavior results.

In Constrained Space Navigation, the assistive features were very satisfying. In assistive motion the human-user controlled the linear motion while the robot guided him successfully on a path of high curvature, by giving the angular velocity commands provided by the path planner. If the user tried to diverge with torque appliance the robot would be very hard to turn, as it would try to get him back to the correct path. In the observing motion, the voting mechanism worked as expected and the correct clusters were picked everytime. The angular velocity was controlled by the robot until the human direction intention was detected through the Admittance model output, while the adaptation of the linear damping coefficient made the system significantly slower, for the user to have enough time to react. Then, a cluster was picked and the Controller was considering again the Admittance model for the angular motion control. Finally, in Idle state the user could not push the robot forward as the linear Admittance model would always have zero output linear velocity. However, he could turn the robot around to find new feassible paths of movement.

Finally, despite the restrictions of the system, the connection between the Admittance Model and the Dynamic Window Path Planner worked successfully and effectively. The prospect of improving and expanding the system is quite promising.

Chapter 7

Conclusions & Future Work

7.1 Conclusions

Overall it was a satisfying and exciting journey. A standard rollator with four normal wheels was transformed into a robot. An improvised handle model was constructed to support the force/torque sensors and two wheels with motors and encoders replaced the rear normal wheels. After a large series of trials a good velocity feedback was acquired and the theoretical ideas could start taking place in a real practical environment. Thus, the optimal Admittance model was estimated, by learning from demonstrations of the most basic movements that were performed in the lab. Thus, an Admittance Controller was designed, so that it could be interfaced with an Arc-Line Path Planner, which was slightly modified from its previous implementation. These two implementations matched very well and it was impressive how good they can work together in practice.

The experimental part went as expected. From all the demonstrations the robot had the average observed dynamic behavior and that was the best that could be achievedwith the specific hardware at the time of the experiments. This implemention in combination with the low-level constraints, resulted in a system that works well only in slow and medium speed. But that was fine, as in this pace the experiments on the assistive features, the undecidability resolution, and the damping adaptation worked pretty well.

Apart from the downsides that have been mentioned many times already, this system has shown great potential for future work. The applications one can implement can be countless and can have a great impact in the daily living activities of humans facing motor impairments. As the years pass it is certain that assistance robots will be more and more part of our lives and we should welcome them warmly, as they can highly improve our standard of living.

7.2 Future Work

It is obvious that for this specific system as it is implemented there can be many improvements. For starters, the handles' construction could be modified to make it more stable and thus get more accurate F/T measurements with less noise. Then, the sensor and its calibration should be studied more, as the implemented calibration method is not optimal and there might be cases where after running the robot for too long the measurements are wrong. This also affects the

method of detecting the hands on the handles. Apart from these, as it has been seen the system overall works well in slow and medium pace. Hence, further improvement of the low-level control to respond to the velocity commands faster and more accurately would result in a far better real application performance. To add to this, the implementation of the wheels and their traction should be studied further, as there were moments where they would slide or stop rotating.

Application and scenario wise just the surface of what one can do with a smart robotic walker was scratched. Some interesting scenarios could be: studying how the system responds to an upcoming obstacle, or in a space with many moving obstacles like a hall with people walking nearby. Another case is what happens in indoors and tight spaces like the bathroom, toilet, kitchen, office etc. There, if the current walker was used, the cluster span would be very small or even there would be no clusters at all and the user would not be able to move. Thus, either the same system can be modified to work with more cluster levels with smaller radius or a completely different approach should be considered. For example, for such tight spaces the user could express his intentions through a GUI or vocally and the robot would control the angular velocity until a goal position is reached.

Furthermore, it would be interesting to study the sit-to-stand or stand-to-sit scenarios. For sit-to-stand the robot would have to adapt the damping coefficients properly so it can support the user until he is in the final standing position, while in the stand-to-sit, things are a bit more complicated. The user has to first approach the chair, then start the sitting procedure. For this purpose, at first, a learning procedure should be considered, so that the robot can learn how to guide the user in the optimal path from random starting positions to the goal chair position. The DMPs method could be suitable for this purpose. Then, when the user is in the proper position, the robot would adapt the damping parameters properly so it can support the user until he is completely sitted.

More and more applications can come to one's mind as the daily human needs constantly increase. Just the basics and maybe a little more were mentioned here, but overall this is an interesting active field of research.

Bibliography

[1] Stefan Lichiardopol, Nathan van de Wouw, and Henk Nijmeijer, “Control Scheme for Human-Robot Co-manipulation of Uncertain, Time-varying Loads”, *In Proceedings of the 2009 American Control Conference*, pp. 1485–1490, St. Louis, MO, USA, 2009, doi:10.1109/ACC.2009.5160062

[2] Guilherme Maeda, Aayush Maloo, Marco Ewerton, Rudolf Lioutikov, Jan Peters, “Anticipative Interaction Primitives for Human-Robot Collaboration”, *AAAI Fall Symposium Series. Shared Autonomy in Research and Practice*, pp. 325-330, 2016, Available at: https://aaai.org/ocs/index.php/FSS/FSS16/paper/view/14067. Date accessed: 30 Oct. 2021.

[3] D. Koert, J. Pajarinen, A. Schotschneider, S. Trick, C. Rothkopf and J. Peters, “Learning Intention Aware Online Adaptation of Movement Primitives”, *IEEE Robotics and Automation Letters*, Vol. 4, no. 4, pp. 3719-3726, 2019, doi: 10.1109/ LRA.2019.2928760.

[4] Yusuf Aydin, Doganay Sirintuna, and Cagatay Basdogan, “Towards collaborative drilling with a cobot using admittance controller”, *Transactions of the Institute of Measurement and Control*, Vol. 43, no. 8, pp. 1760-1773, 2021, doi:10.1177/0142331220934643

[5] S. Sánchez Restrepo, G. Raiola, P. Chevalier, X. Lamy and D. Sidobre, “Iterative virtual guides programming for human-robot comanipulation”, *IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, pp. 219-226, 2017, doi: 10.1109/AIM.2017.8014021.

[6] Gyung Tak Sung and Inderbir S. Gill, “Robotic laparoscopic surgery: a comparison of the da Vinci and Zeus systems”, *Urology*, Vol. 68, Issue 6, pp. 893-898, ISSN 0090-4295, 2001, doi: 10.1016/S0090-4295(01)01423-6.

[7] B. E. Lawson, J. Mitchell, D. Truex, A. Shultz, E. Ledoux and M. Goldfarb, “A Robotic Leg Prosthesis: Design, Control, and Implementation”, *IEEE Robotics & Automation Magazine*, Vol. 21, no. 4, pp. 70-81, 2014, doi: 10.1109/MRA.2014.2360303.

[8] D.S.V. Bandara, R.A.R.C. Gopura, K.T.M.U. Hemapala, Kazuo Kiguchi, “Development of a multi-DoF transhumeral robotic arm prosthesis”, *Medical Engineering & Physics*, Vol. 48, pp. 131-141, ISSN 1350-4533, 2017, doi: 10.1016/j.medengphy.2017.06.034.

[9] F. Leishman, O. Horn, G. Bourhis, “Smart wheelchair control through a deictic approach”, *Robotics and Autonomous Systems*, Vol. 58, Issue 10, pp. 1149-1158, ISSN 0921-8890, 2010, doi: 10.1016/j.robot.2010.06.007.

[10] I. Ulrich and J. Borenstein, “The GuideCane-applying mobile robot technologies to assist the visually impaired”, *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, Vol. 31, no. 2, pp. 131-136, 2001, doi: 10.1109/ 3468.911370.

[11] Mayara Bonani, Raquel Oliveira, Filipa Correia, André Rodrigues, Tiago Guerreiro, Ana Paiva, “What My Eyes Can’t See, A Robot Can Show Me: Exploring the Collaboration

Between Blind People and Robots", *In Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility*, pp. 15-27, 2018, doi: 10.1145/3234695.3239330.

[12] P. Mayer, C. Beck and P. Panek, "Examples of multimodal user interfaces for socially assistive robots in Ambient Assisted Living environments", *IEEE 3rd International Conference on Cognitive Infocommunications (CogInfoCom)*, pp. 401-406, 2012, doi: 10.1109/CogInfoCom.2012.6422014.

[13] Sierra Marín, Sergio & Garzon, Mario & Munera, Marcela & Cifuentes G., Carlos, "Human–Robot–Environment Interaction Interface for Smart Walker Assisted Gait: AGoRA Walker", *Sensors,* Vol. 19, no. 13, p. 2897, doi: 10.3390/s19132897.

[14] Afsar, Md. Rayhan & Shen, Tao & Shen, Xiangrong & Zhang, He & Ye, Cang, "Development of a Motorized Robotic Walker Guided by an Image Processing System for Human Walking Assistance and Rehabilitation", *In Procceedings of The 2018 ASME Dynamic Systems and Control (DSC) Conference*, 2018, doi: 10.1115/DSCC2018-9223.

[15] M. Bošnak and I. Škrjanc, "Embedded Control System for Smart Walking Assistance Device", *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, Vol. 25, no. 3, pp. 205-214, 2017, doi: 10.1109/TNSRE.2016.2553369.

[16] P. Di et al., "Fall Detection and Prevention Control Using Walking-Aid Cane Robot", *IEEE/ASME Transactions on Mechatronics*, Vol. 21, no. 2, pp. 625-637, 2016, doi: 10.1109/TMECH.2015.2477996.

[17] Milad Geravand, Wolfgang Rampeltshammer, Angelika Peer, "Control of Mobility Assistive Robot for Human Fall Prevention", *In Proceedings of the 2015 IEEE International Conference on Rehabilitation Robotics (ICORR)*, 2015, doi: 10.1109/ICORR.2015.7281314.

[18] Christian Werner, Milad Geravand, Péter Z Korondi, Angelika Peer, Jürgen M Bauer, and Klaus Hauer, "Evaluating the sit-to-stand transfer assistance from a smart walker in older adults with motor impairments", *Geriatrics and Gerontology International*, Vol. 20, no. 4, pp. 312-316, 2020, doi: 10.1111/ggi.13874.

[19] Geravand, M., Werner, C., Hauer, K.et al., "An Integrated Decision Making Approach for Adaptive Shared Control of Mobility Assistance Robots", I*nternational Journal of Social Robotics*, Vol. 8, Issue 5, pp. 631–648, 2016. doi: 10.1007/s12369-016-0353-z.

[20] Jimenez, Mario & Monllor, Matias & Frizera, Anselmo & Freire, Teodiano & Roberti, Flavio & Carelli, Ricardo, "Admittance Controller with Spatial Modulation for Assisted Locomotion using a Smart Walker", *Journal of Intelligent & Robotic Systems*, Vol. 94, no. 7, pp. 1-17, 2019, doi: 10.1007/s10846-018-0854-0.

[21] G. Grisetti, C. Stachniss and W. Burgard, "Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters", *IEEE Transactions on Robotics*, Vol. 23, no. 1, pp. 34-46, Feb. 2007, doi: 10.1109/TRO.2006.889486.

[22] David Ball, Andrew English, Patrick Ross, Tim Patten, Robert Fitch, Salah Sukkarieh, Andrew Bate, "Vision-based Obstacle Detection and Navigation for an Agricultural

Robot", *Journal of Field Robotics*, Vol. 33, Issue 8, pp. 1107-1130, 2016, doi: 10.1002/rob.21644.

[23] Azeta, Joseph & Bolu, Christian & Hinvi, Daniel & Abioye, Abiodun, "Obstacle detection using ultrasonic sensor for a mobile robot", *IOP Conference Series: Materials Science and Engineering*, Vol. 707, no. 1, 2019, doi: 10.1088/1757-899X/707/1/012012.

[24] Shahdib, Fayaz & Bhuiyan, Md & Hasan, Md Kamrul & Mahmud, Hasan, "Obstacle Detection and Object Size Measurement for Autonomous Mobile Robot using Sensor", *International Journal of Computer Applications*, Vol. 66, pp. 28-33, 2013, doi: 10.5120/11114-6074.

[25] G. P. Moustris and C. S. Tzafestas, "Assistive front-following control of an intelligent robotic rollator based on a modified dynamic window planner", *2016 6th IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, pp. 588-593, 2016, doi: 10.1109/BIOROB.2016.7523689.

[26] G. P. Moustris and C. S. Tzafestas, "Intention-based front-following control for an intelligent robotic rollator in indoor environments", *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1-7, 2016, doi: 10.1109/SSCI.2016.7850067.

[27] Milad Geravand, Peter Zeno Korondi, Christian Werner, Klaus Hauer, Angelika Peer, "Human Sit-to-Stand Transfer Modelling Towards Intuitive and Biologically-Inspired Robot Assistance", *Journal Autonomous Robots*, Vol. 41, Issue 3, pp. 575-592, ISSN 1573-7527, 2017, doi: 10.1007/s10514-016-9553-5.

[28] Frizera-Neto, Anselmo et al., "Empowering and Assisting Natural Human Mobility: The Simbiosis Walker", *International Journal of Advanced Robotic Systems*, Vol. 8, Issue 3, 2011, doi: 10.5772/10666

[29] Annicchiarico, Roberta & Barrué, Cristian & Benedico, T. & Campana, Fabio & Cortés, Ulises & Martínez-Velasco, A., "The i-Walker: an intelligent pedestrian mobility aid", *In Proceedings of the ECAI 2008 - 18th European Conference on Artificial Intelligence*, pp. 708-712, 2008, doi: 10.3233/978-1-58603-891-5-708.

[30] Tausel, Luca & Cifuentes G., Carlos & Rodriguez, Camilo & Frizera, Anselmo & Bastos, Teodiano, "Human-walker interaction on slopes based on LRF and IMU sensors", *In Proceedings of the IEEE RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics*, pp. 227-232, 2014. doi: 10.1109/BIOROB.2014.6913781.

[31] Andreas Wachaja, Pratik Agarwal, Mathias Zink, Miguel Reyes Adame, Knut Möller, Wolfram Burgard, "Navigating bling people with walking impairments using a smart walker", *Journal Autonomous Robots*, Vol. 41, Issue 3, pp. 555-573, ISSN 1573-7527, 2016, doi: 10.1007/s10514-016-9595-8.

[32] Moustris, George & Kardaris, Nikolaos & Tsiami, Antigoni & Chalvatzaki, Georgia & Koutras, Petros & Dometios, Athanasios & Oikonomou, Paris & Tzafestas, Costas & Maragos, Petros & Efthimiou, Eleni & Papageorgiou, Xanthi S. & Fotinea, Stavroula-Evita & Koumpouros, Yiannis & Vacalopoulou, Anna & Karavasili, Alexandra & Nikolakakis, Alexandros & Karaiskos, Konstantinos & Mavridis, Panagiotis, "The I-Walk Assistive Robot", *Human-Friendly Robotics 2020, 13th International Workshop*, pp. 31-45, 2021, doi: 10.1007/978-3-030-71356-0_3.

[33] Dian Wang, Colin Kohler, Andreas ten Pas, Alexander Wilkinson, Maozhi Liu, Holly Yanco, and Robert Platt, “Towards Assistive Robotic Pick and Place in Open World Environments”, *arXiv preprint,* 2019, arXiv:1809.09541[cs.RO].

[34] Schreuder, HWR & Verheijen, R., “Robotic surgery”, *BJOG : An international journal of obstetrics and gynaecology*, Vol. 116. pp. 198-213, 2019. doi: 10.1111/j.1471-0528.2008.02038.x.

[35] M. Karaim, A. Noureldin and T. B. Karamat, “Low-cost IMU Data Denoising using Savitzky-Golay Filters”, *In Proceedings of the 2019 International Conference on Communications, Signal Processing, and their Applications (ICCSPA)*, pp. 1-5, 2019, doi: 10.1109/ICCSPA.2019.8713728.

[36] Śniegocki Henrykm, Charchalis Adam, Dereszewski Mirosław, “Processing of Instantaneous Angular Speed Signal for Detection of a Diesel Engine Failure”, *Mathematical Problems in Engineering*, Vol. 2013, Article ID 659243, 2013, doi: 10.1155/2013/659243

Appendix 1: Model Validation Extra Results

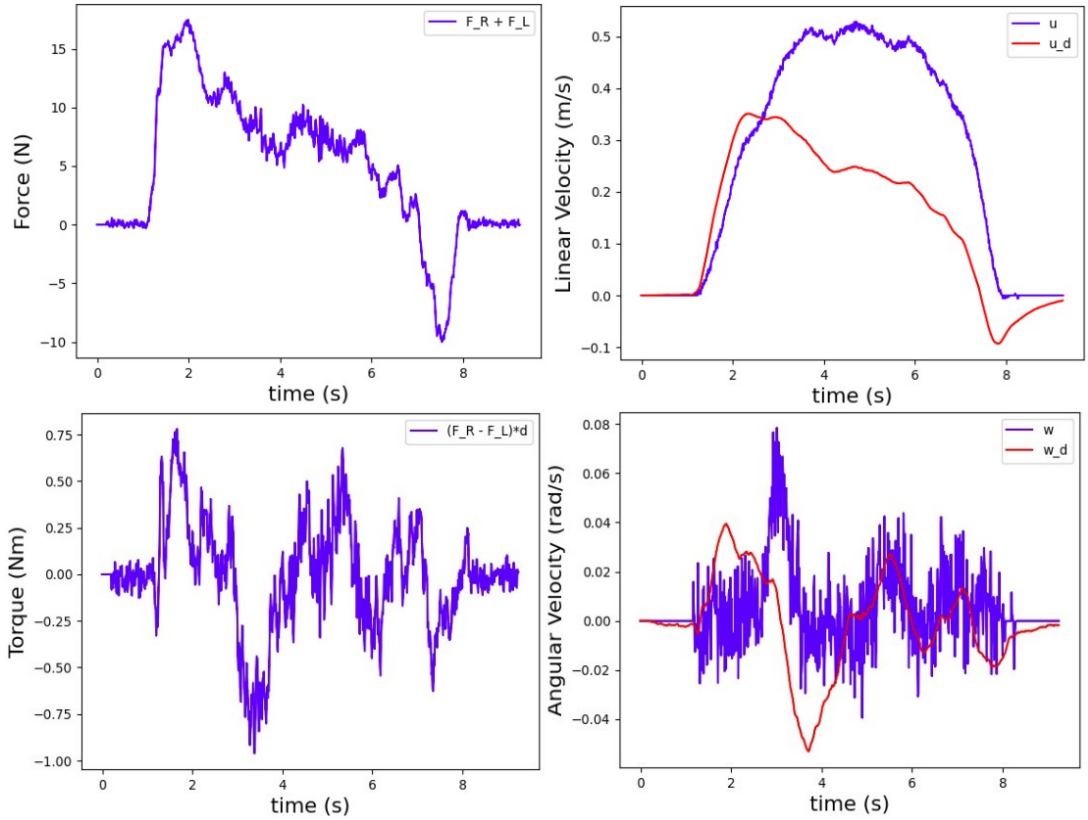


Fig 38. MV: Forward movement with MEDIUM speed.

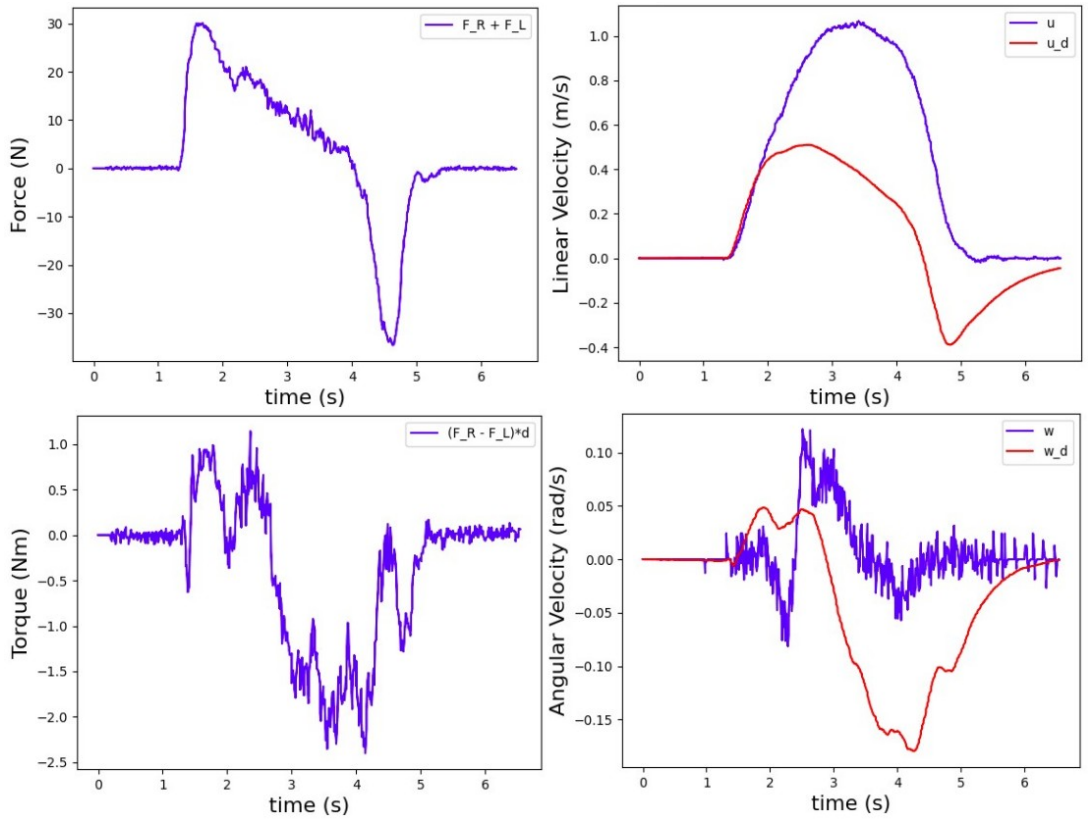


Fig 39. MV: Forward movement with FAST speed.

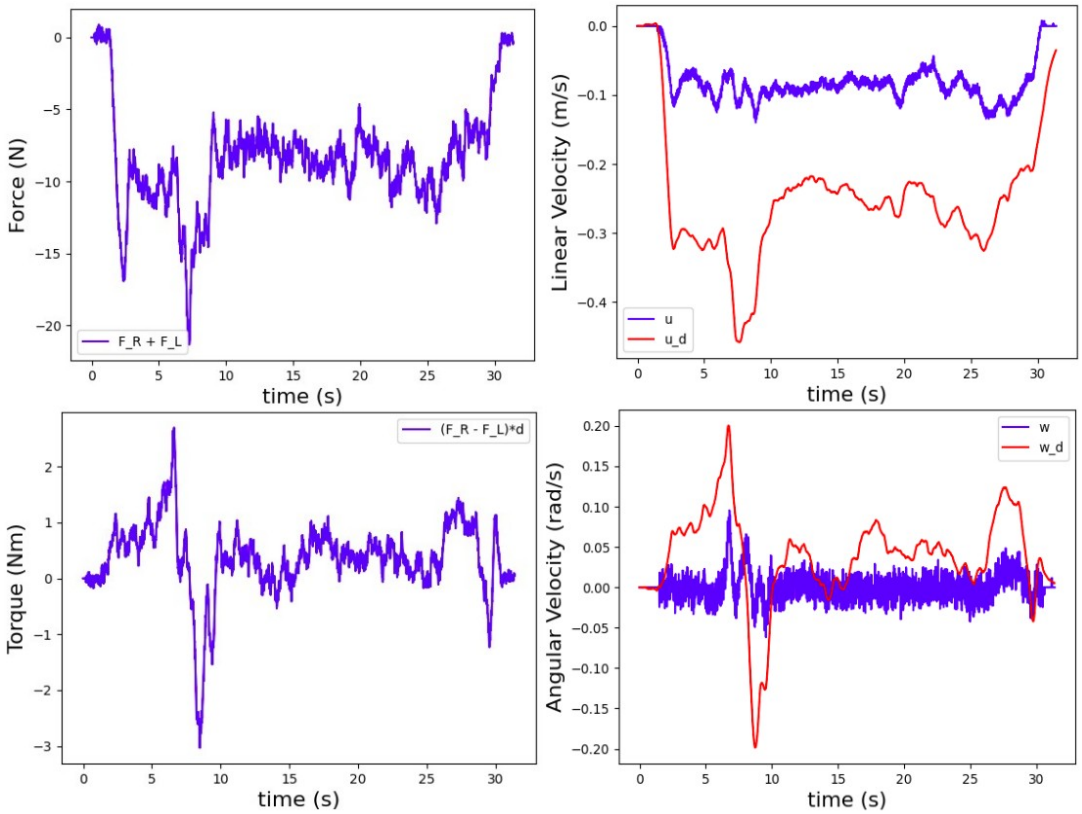


Fig 40. MV: Backward movement with SLOW speed.

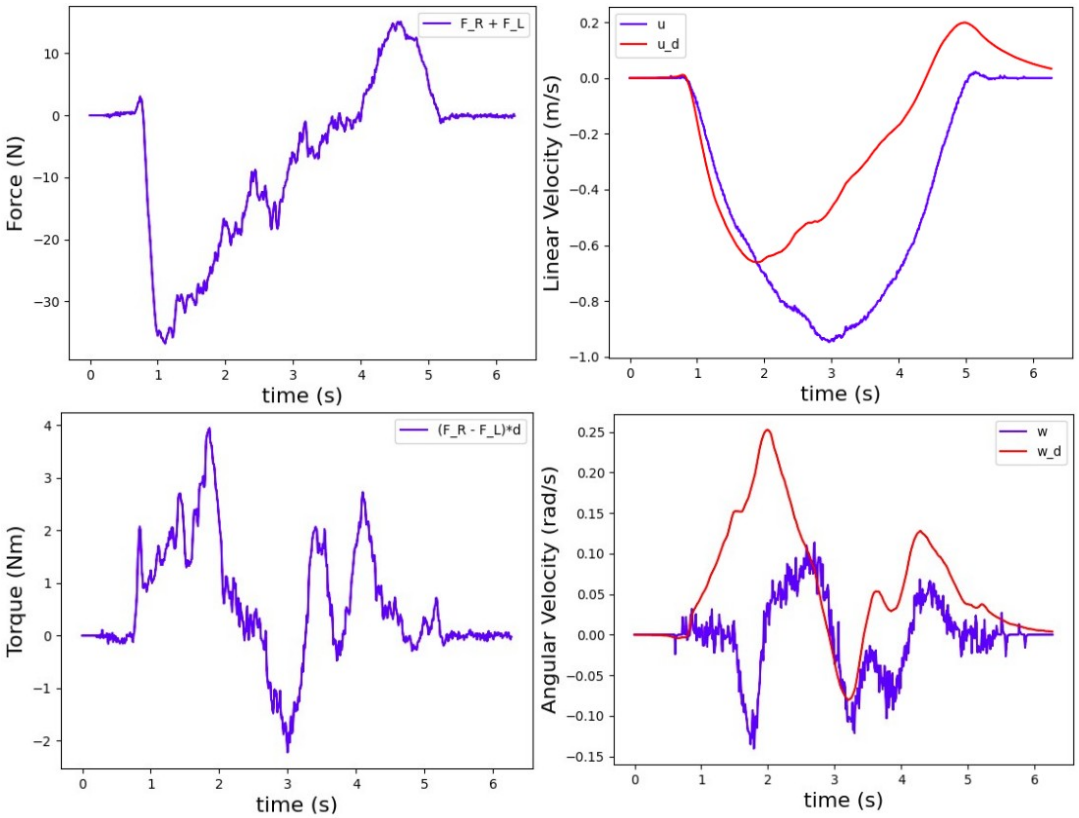


Fig 41. MV: Backward movement with FAST speed.

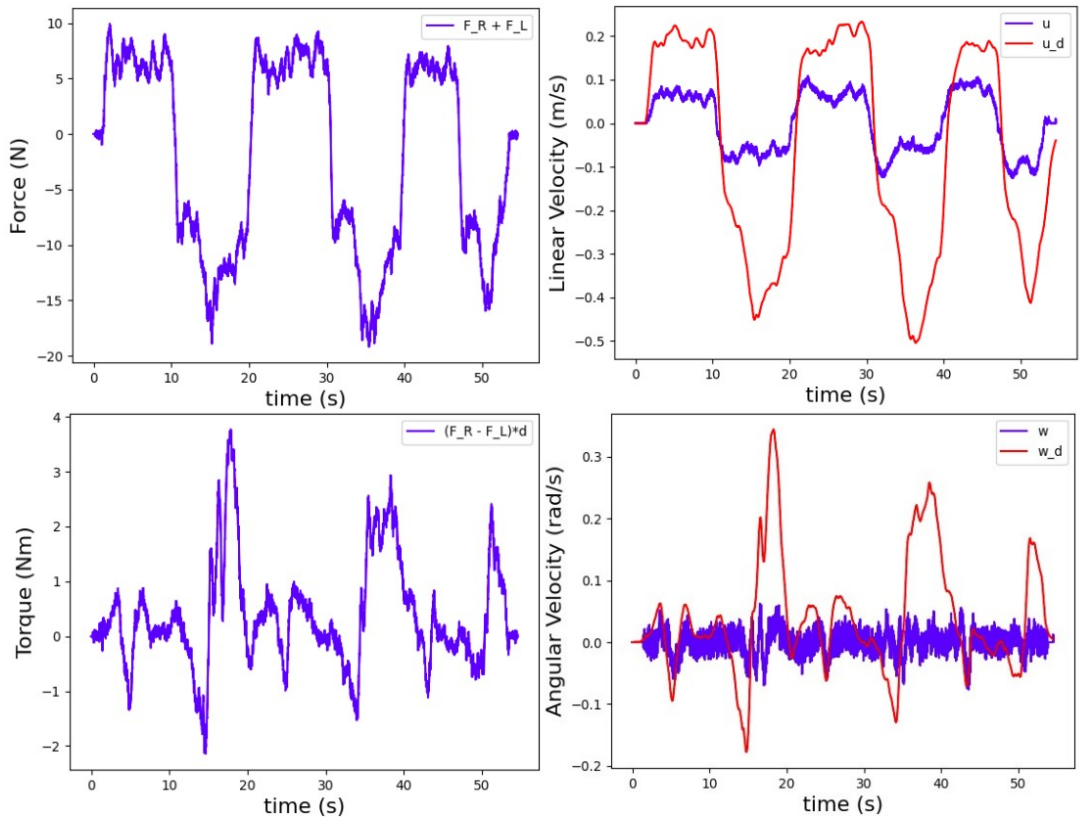


Fig 42. MV: Back-Forth movement with SLOW speed.

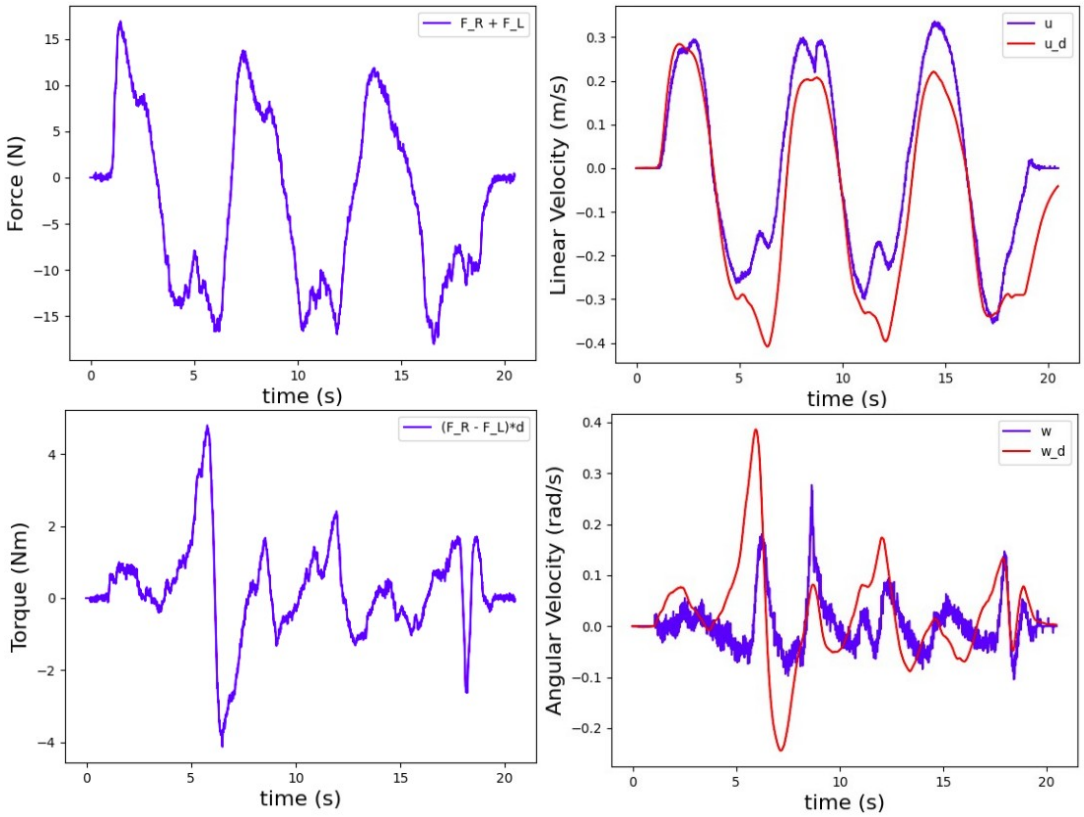


Fig 43. MV: Back-Forth movement with MEDIUM speed.

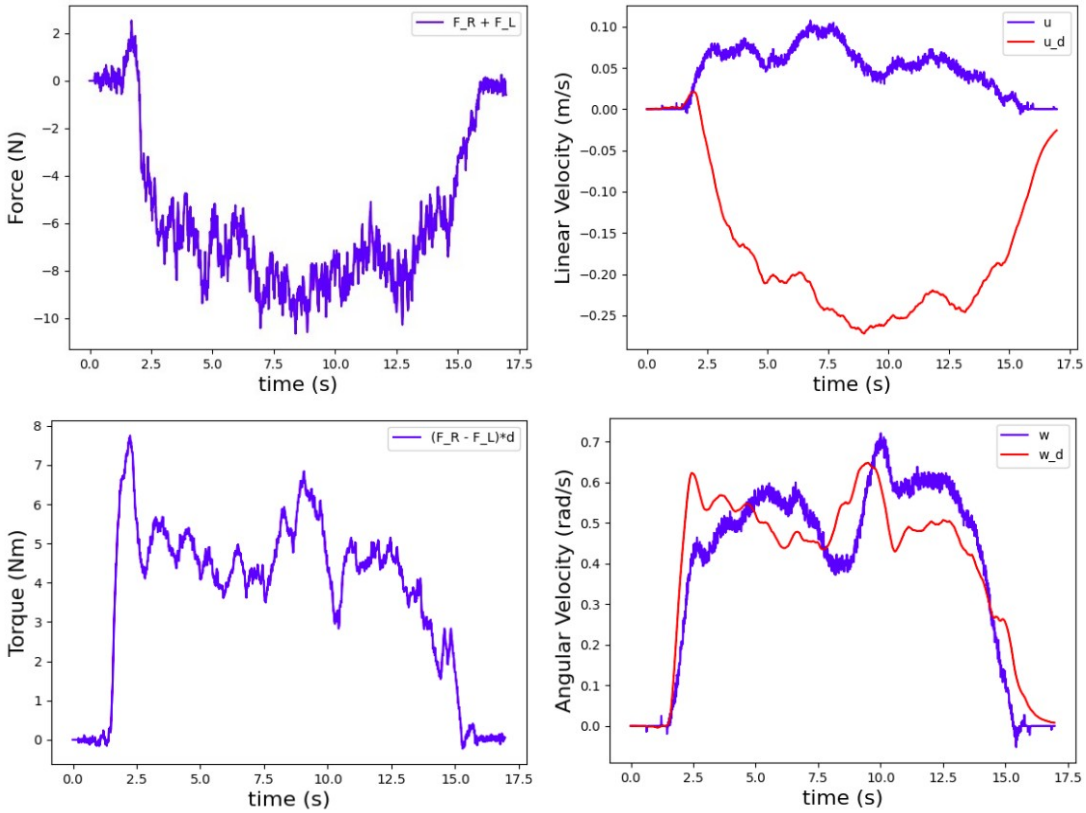


Fig 44. MV: Leftwise 360^0 turn movement with MEDIUM speed.

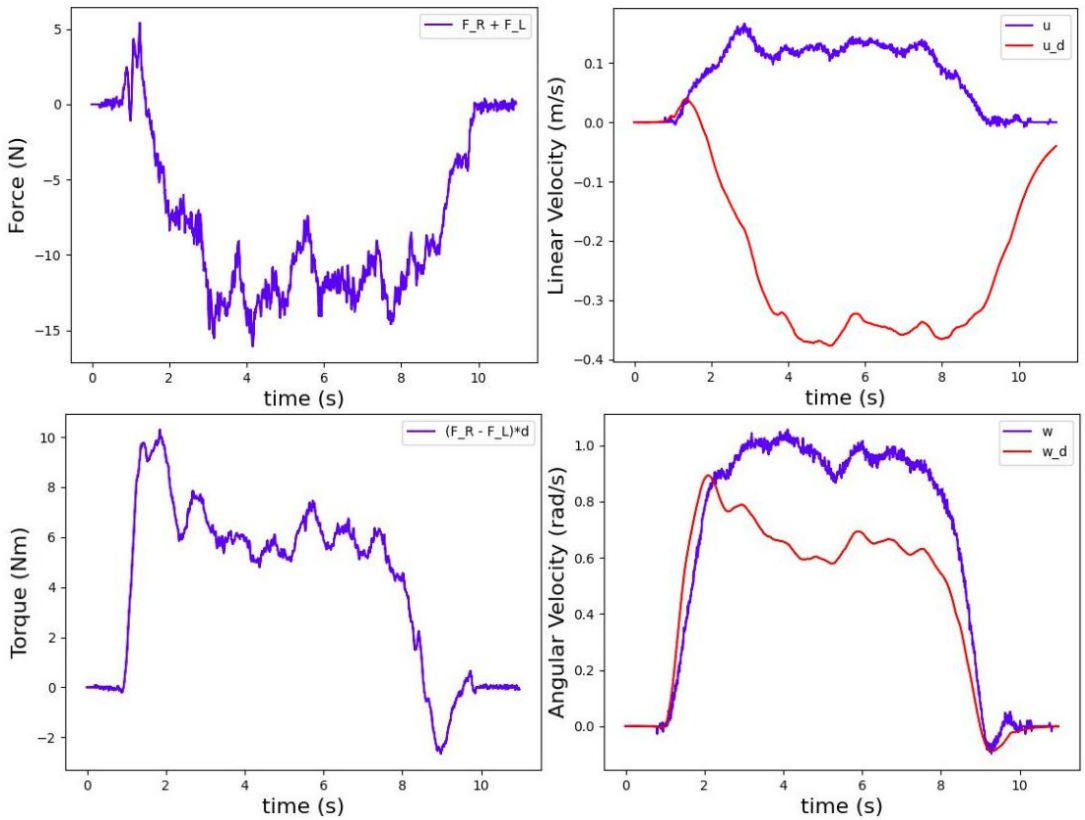


Fig 45. MV: Leftwise 360^0 turn movement with FAST speed.

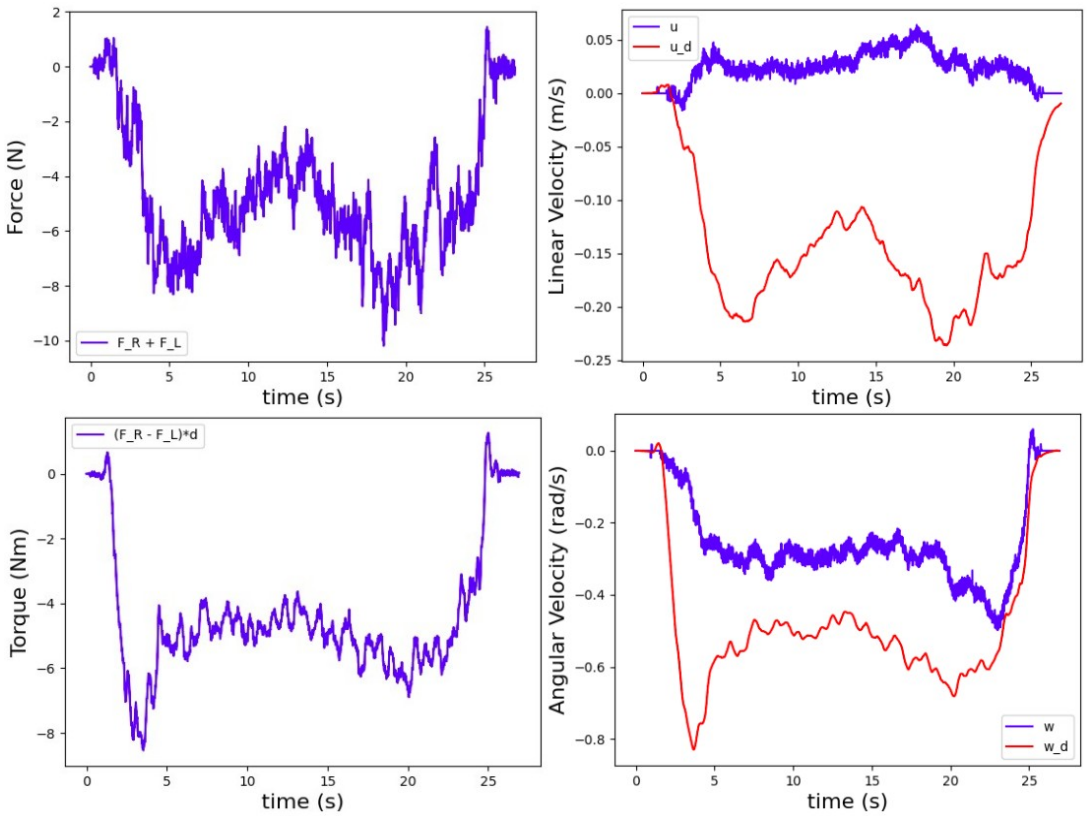


Fig 46. MV: Rightwise 360^0 turn movement with SLOW speed.

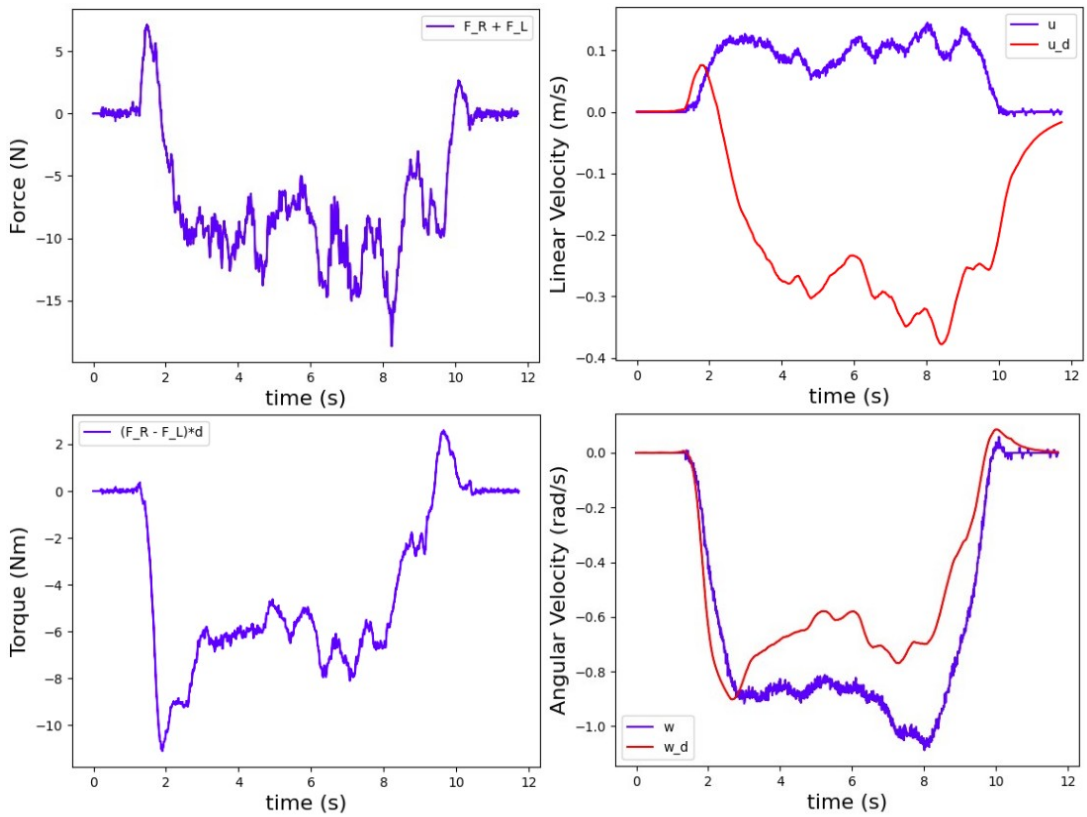


Fig 47. MV: Rightwise 360^0 turn movement with FAST speed.

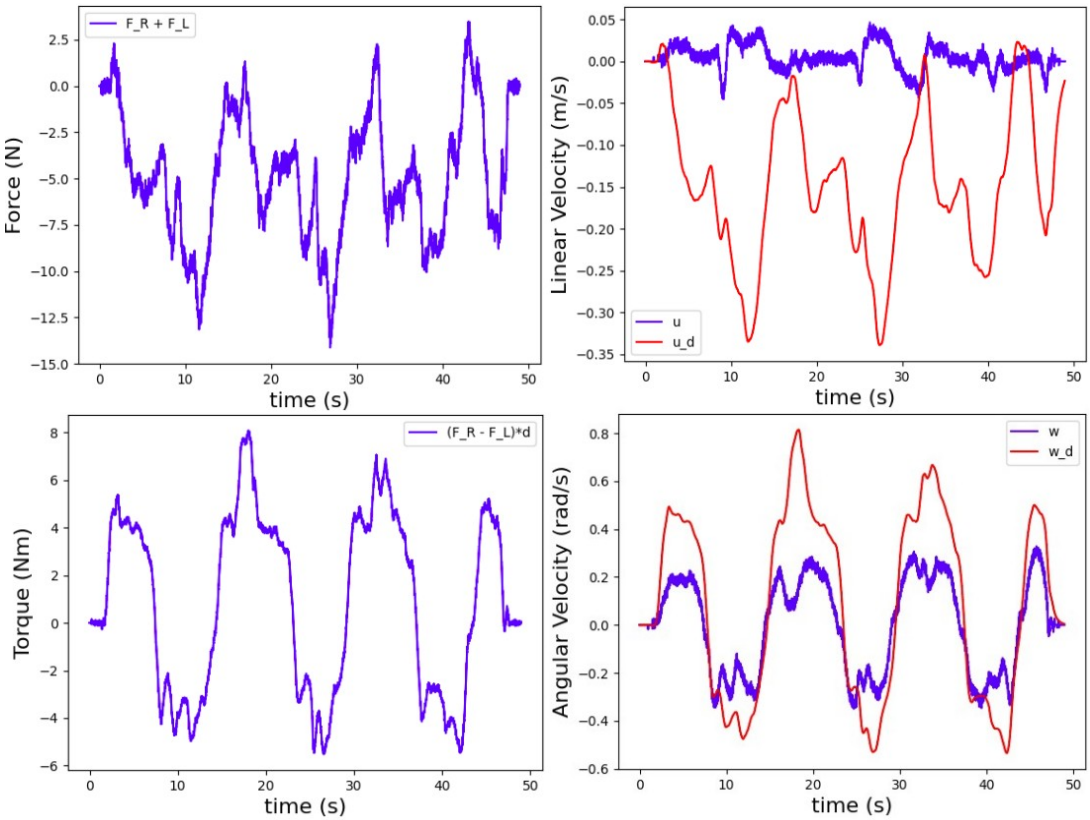


Fig 48. MV: Left-Right turn movement with SLOW speed.

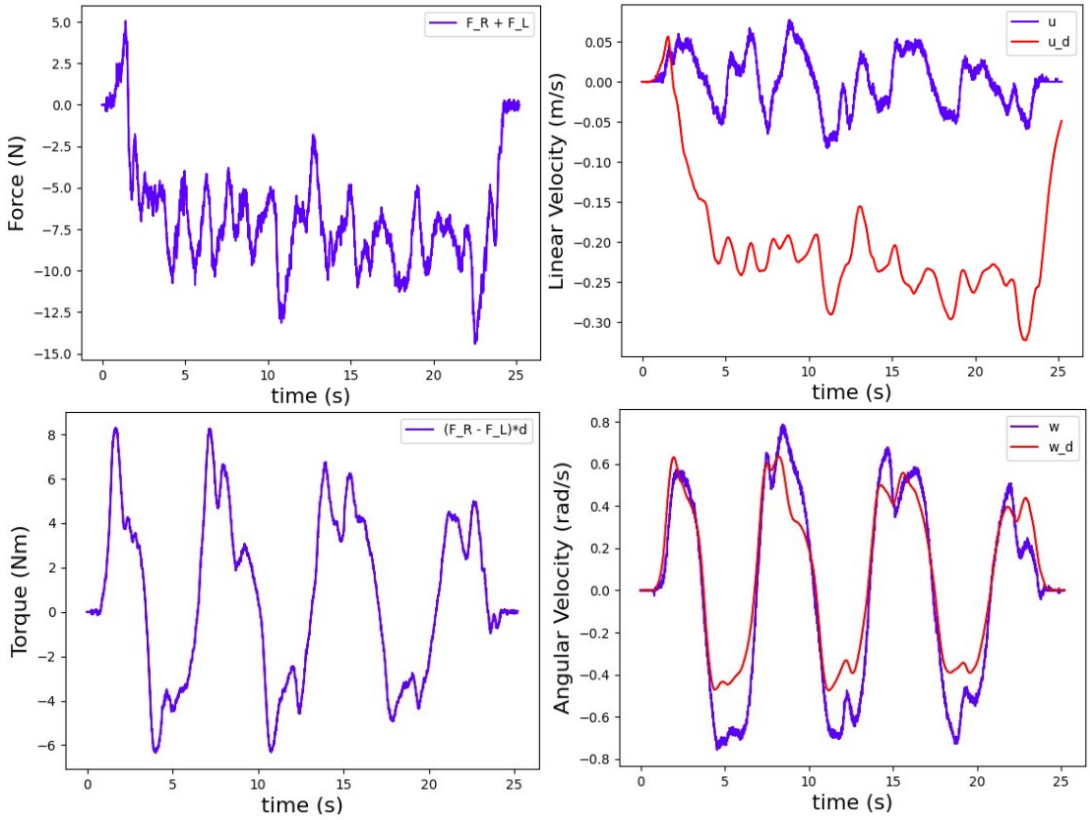


Fig 49. MV: Left-Right turn movement with MEDIUM speed.

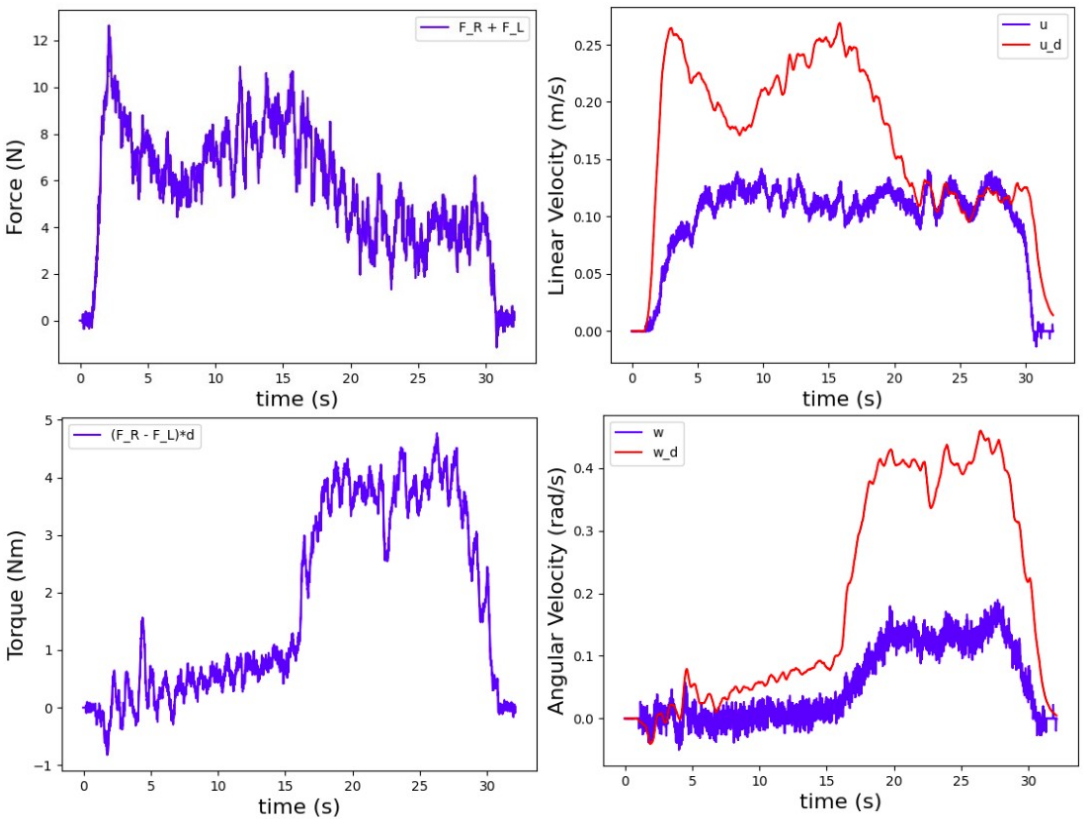


Fig 50. MV: Leftwise L curve movement with SLOW speed.

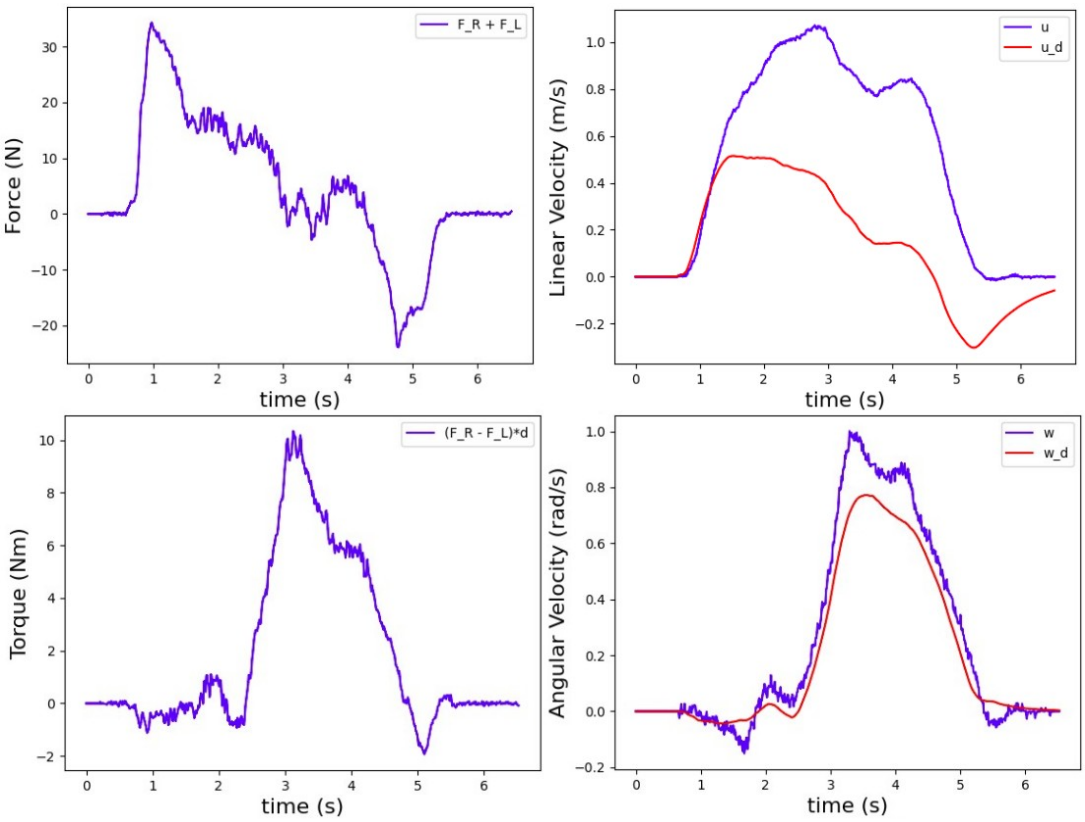


Fig 51. MV: Leftwise L curve movement with FAST speed.

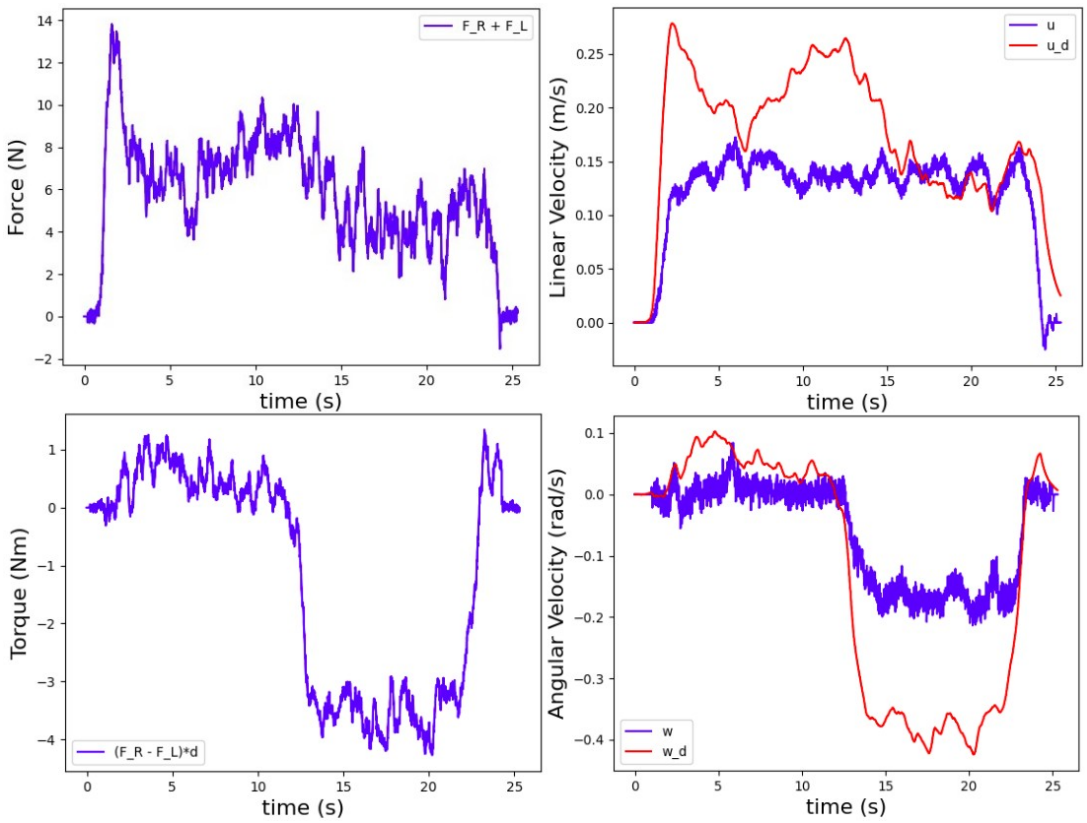


Fig 52. MV: Rightwise L curve movement with SLOW speed.

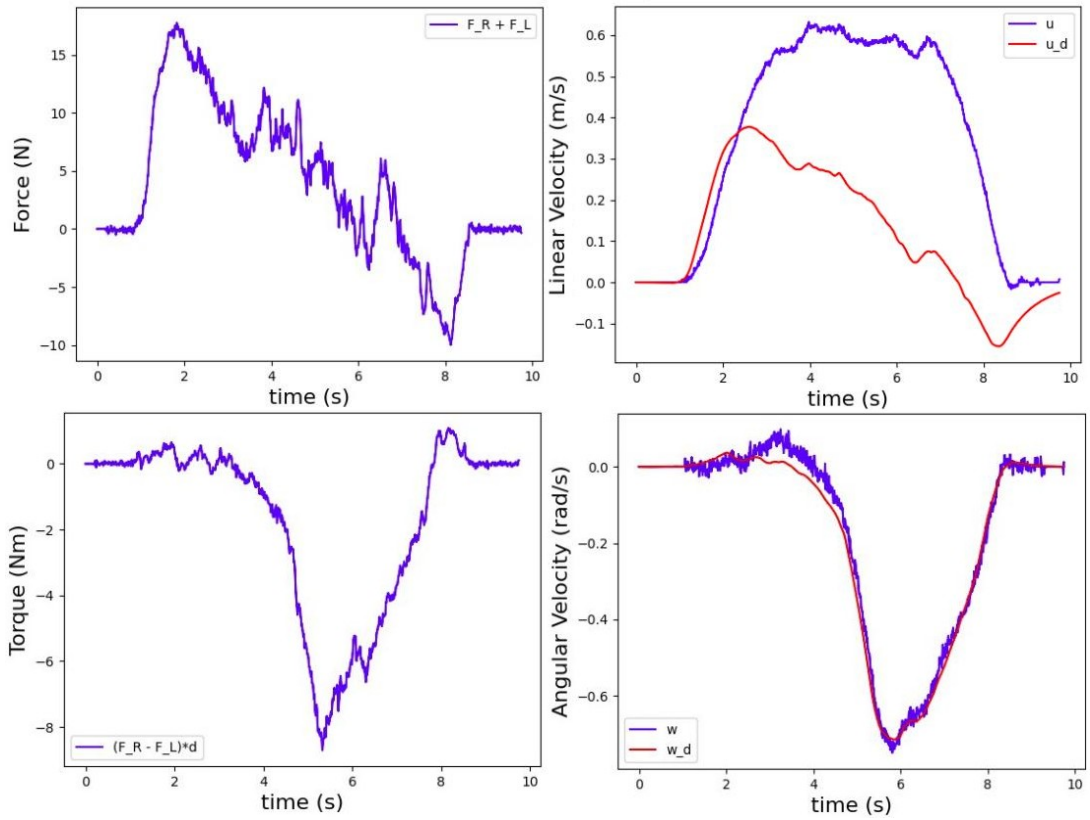


Fig 53. MV: Rightwise L curve movement with MEDIUM speed.

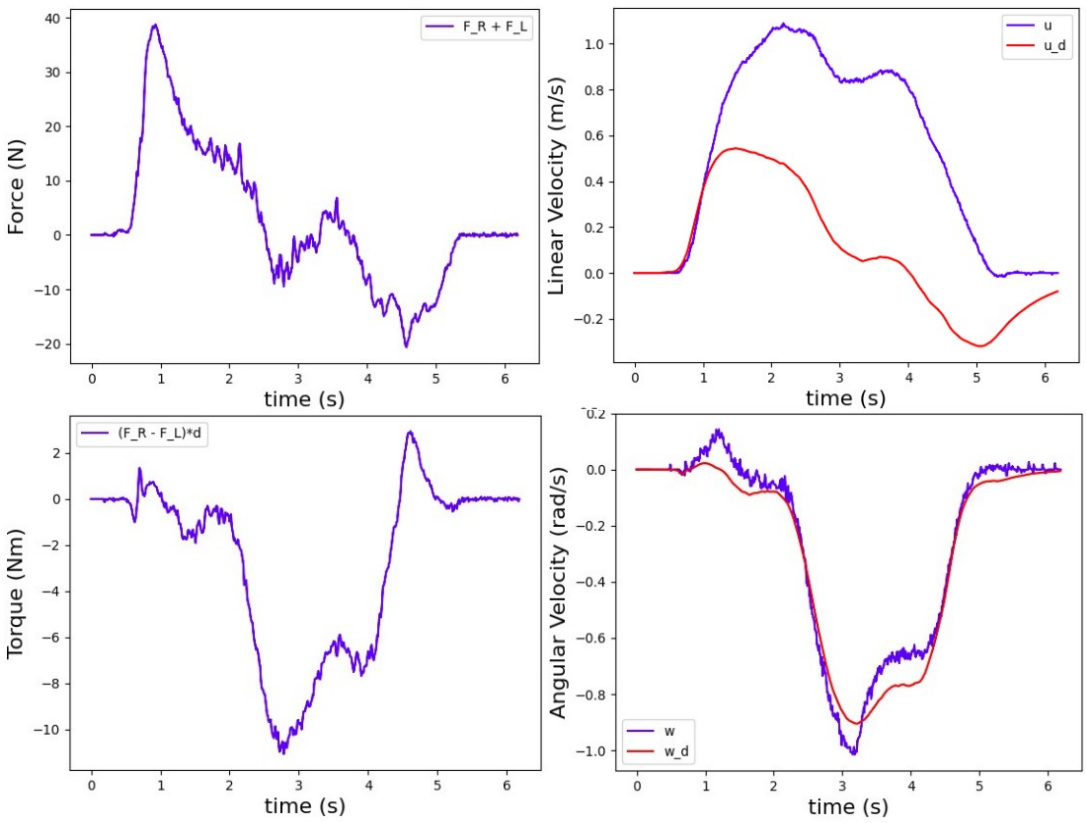


Fig 54. MV: Rightwise L curve movement with FAST speed.

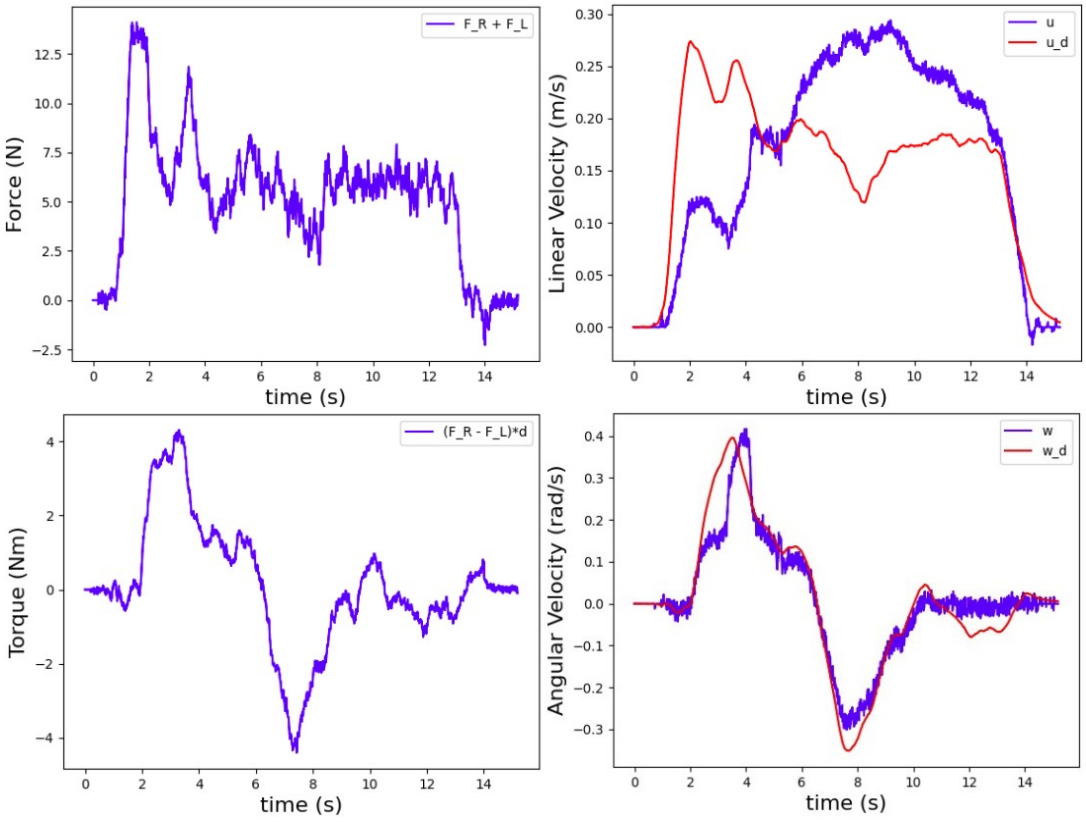


Fig 55. MV: Leftwise S curve movement with SLOW speed.

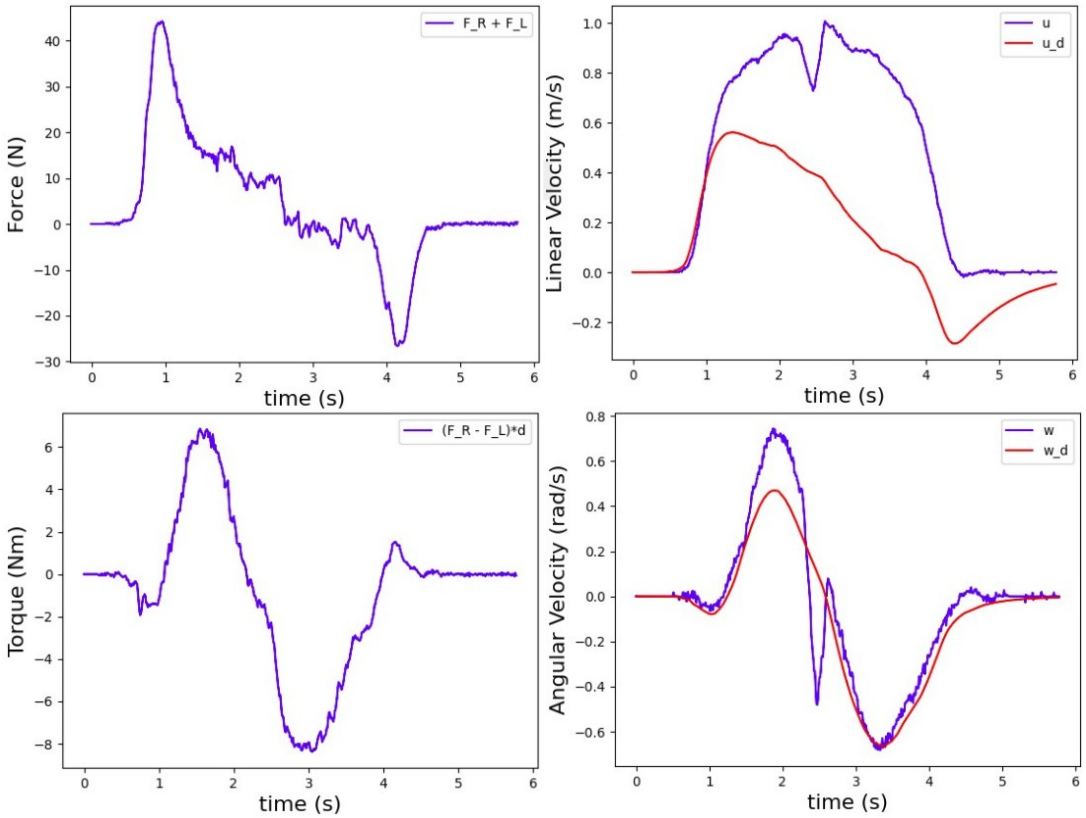


Fig 56. MV: Leftwise S curve movement with FAST speed.

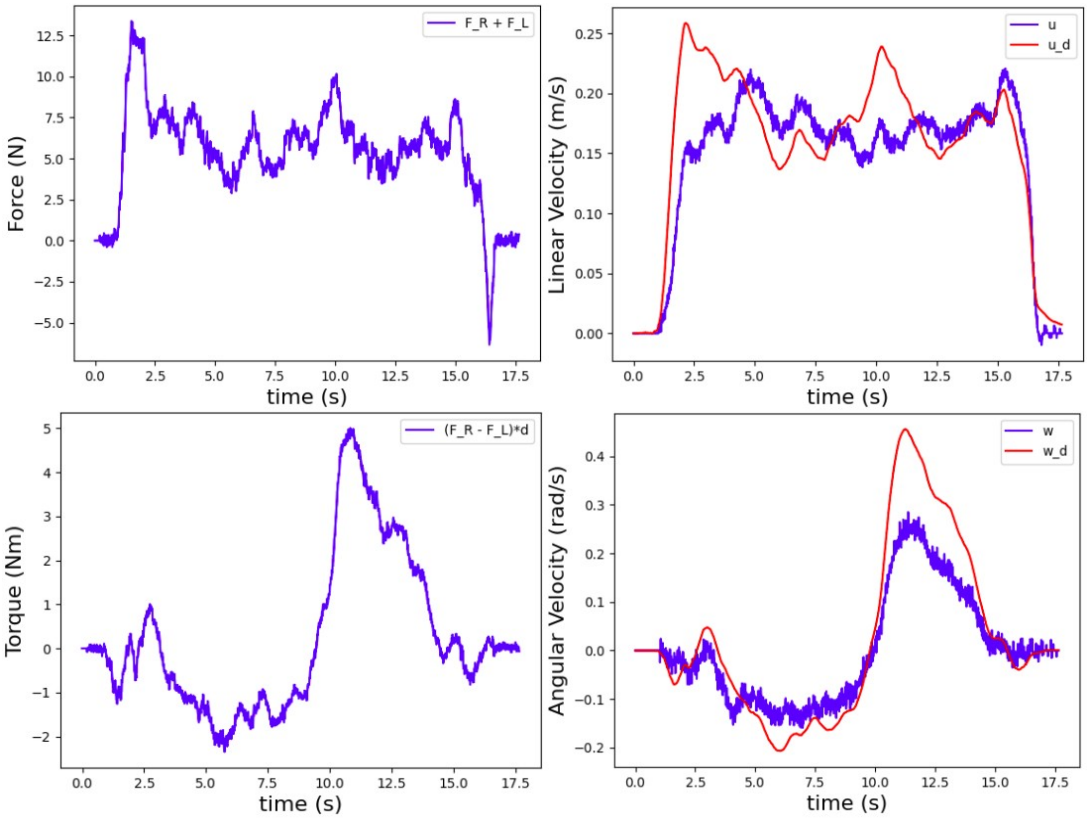


Fig 57. MV: Rightwise S curve movement with SLOW speed.

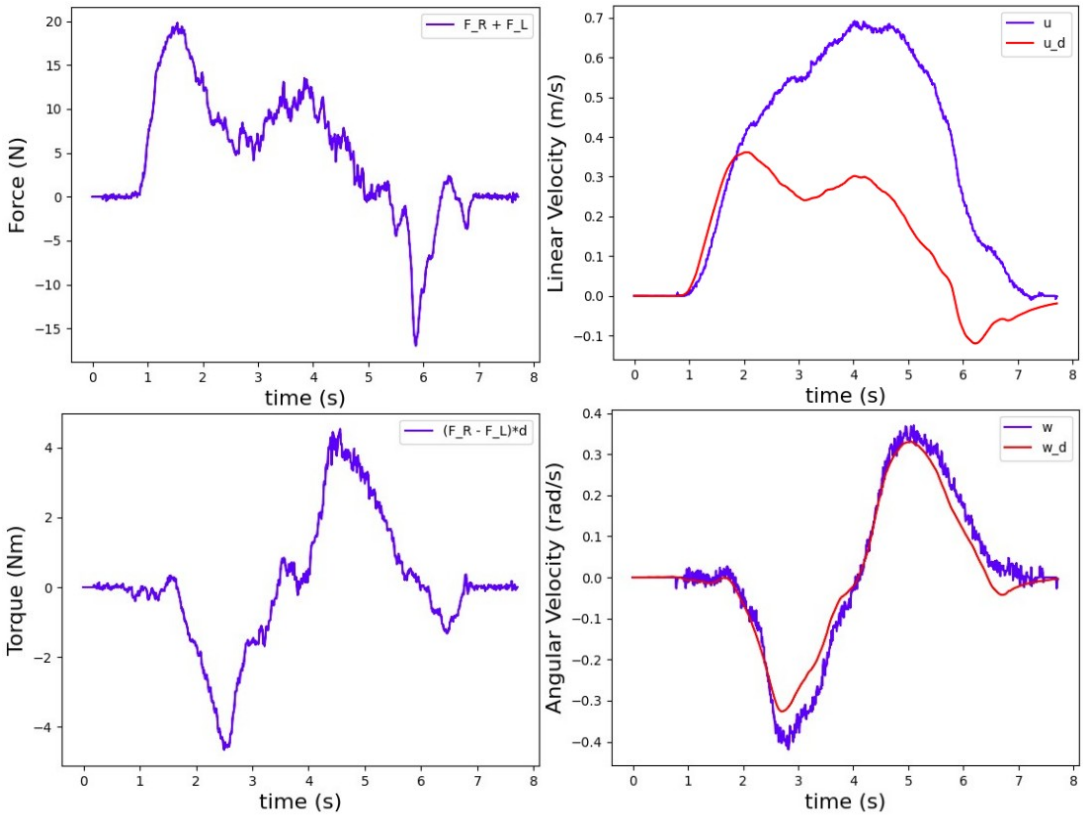


Fig 58. MV: Rightwise S curve movement with MEDIUM speed.

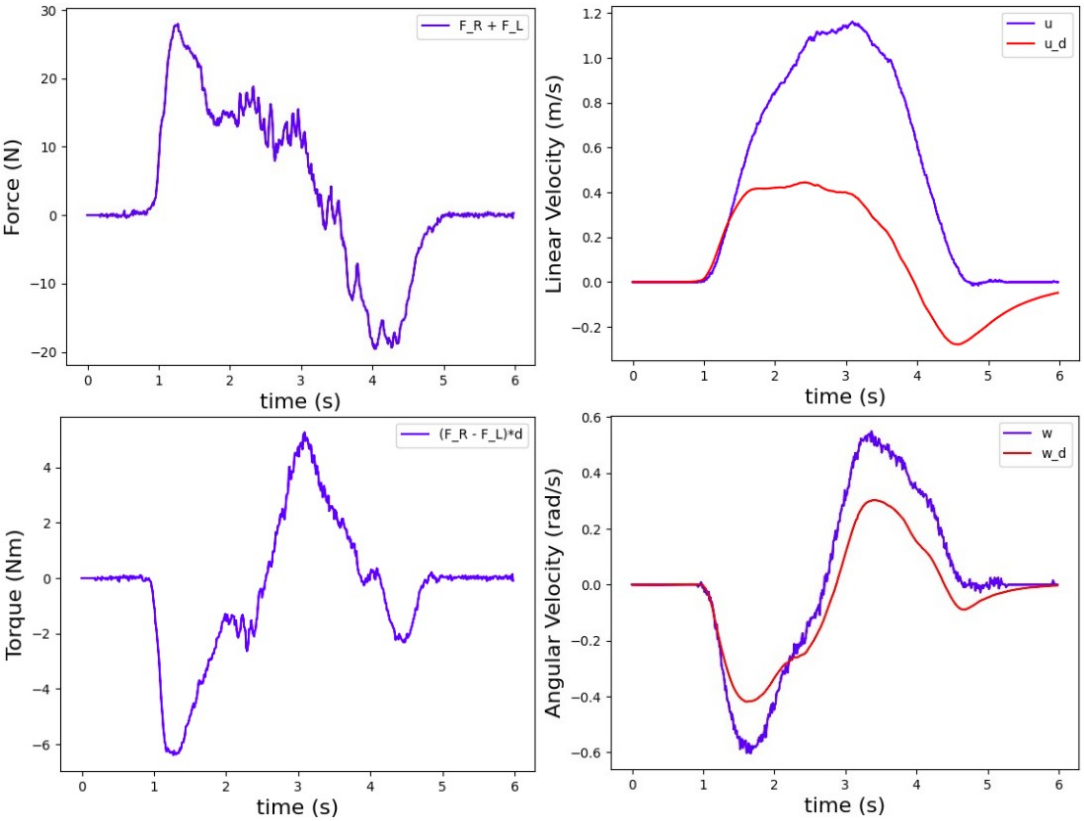


Fig 59. MV: Rightwise S curve movement with FAST speed.

Appendix 2: FS Navigation Extra Results

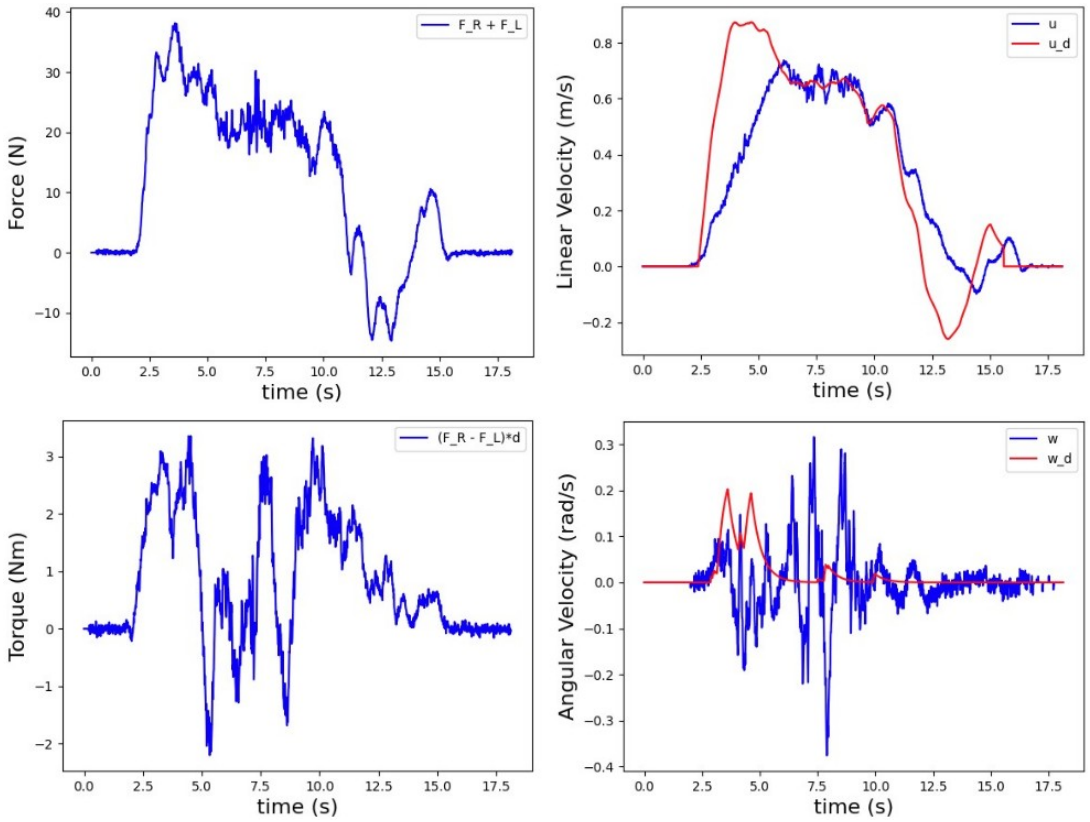


Fig 60. FSN: Forward movement with MEDIUM speed.

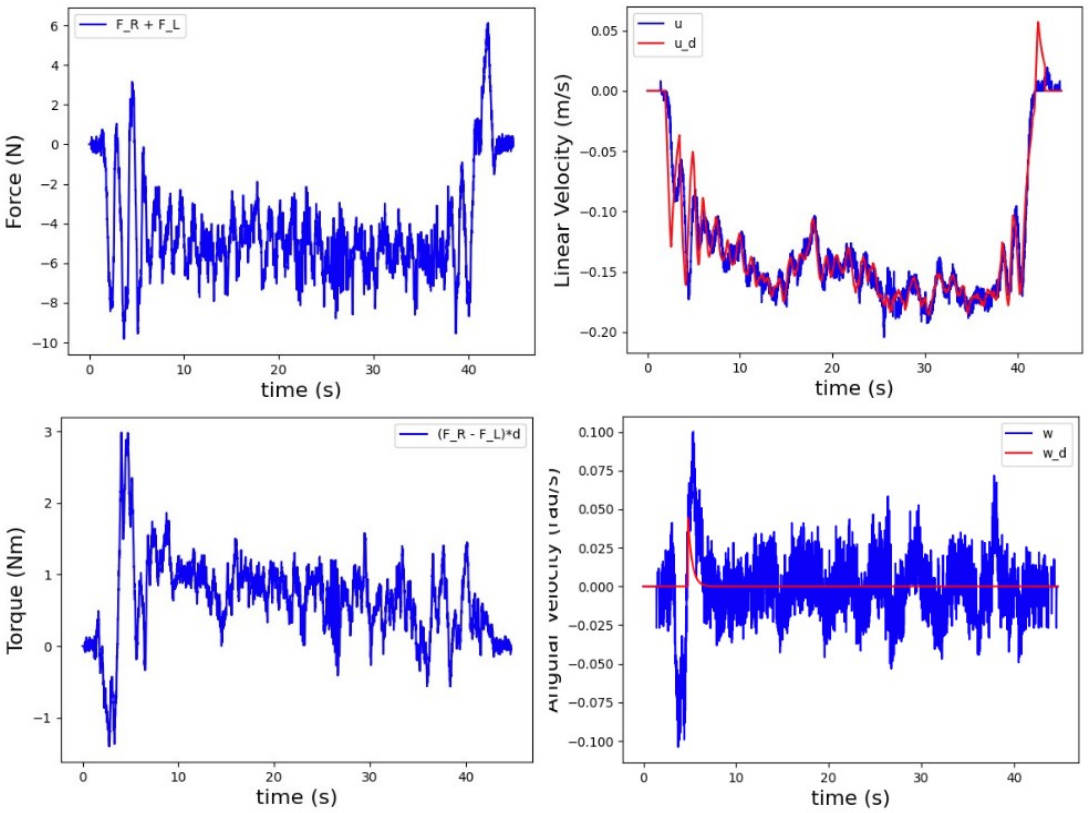


Fig 61. FSN: Backward movement with SLOW speed.

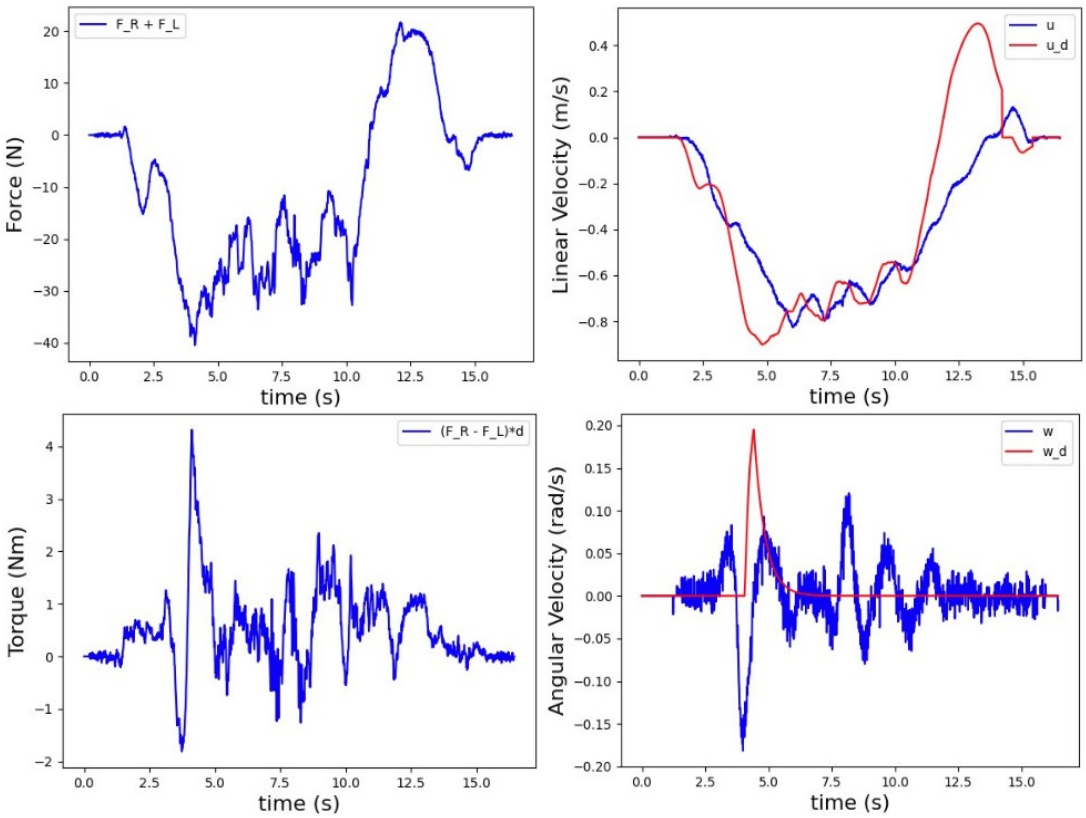


Fig 62. FSN: Backward movement with FAST speed.

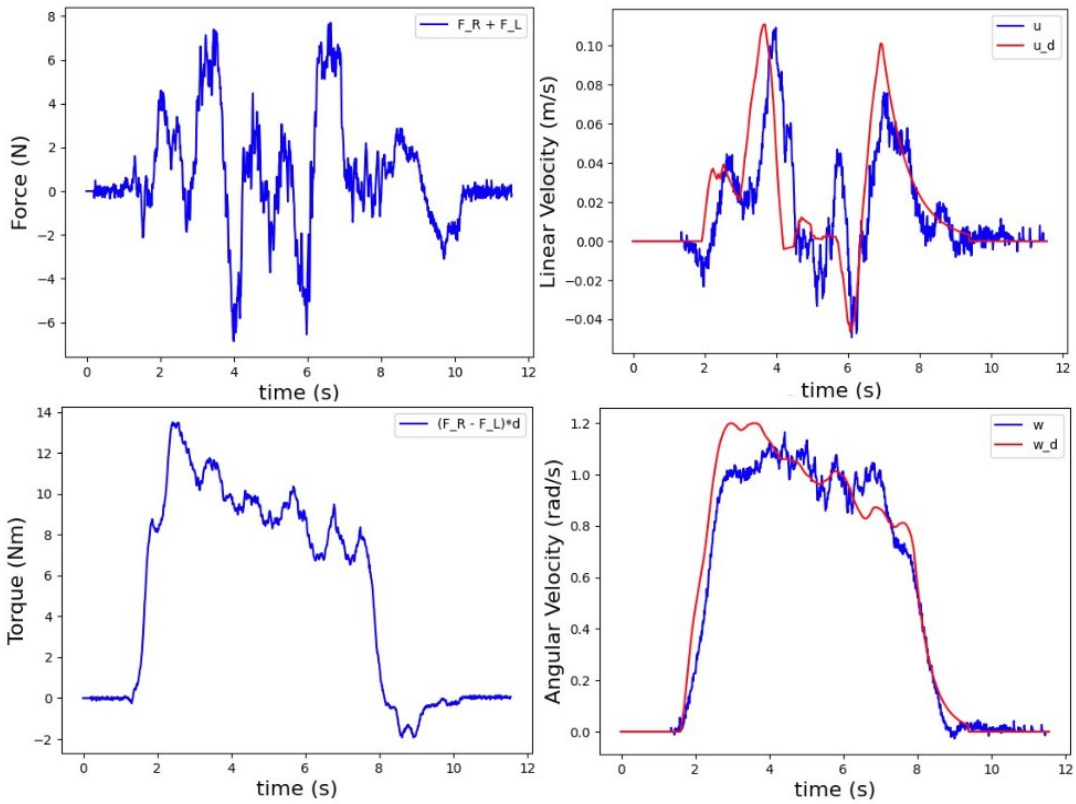


Fig 63. FSN: Leftwise 360º turn movement with FAST speed.

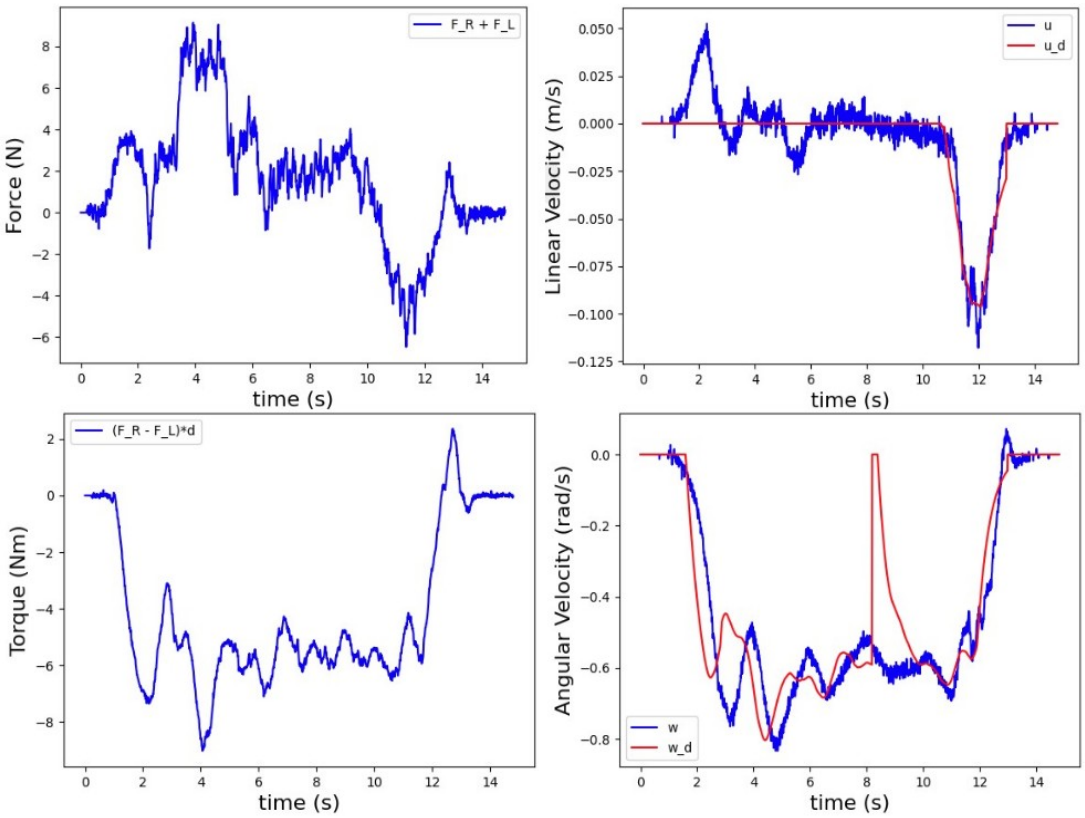


Fig 64. FSN: Rightwise 360º turn movement with SLOW speed.

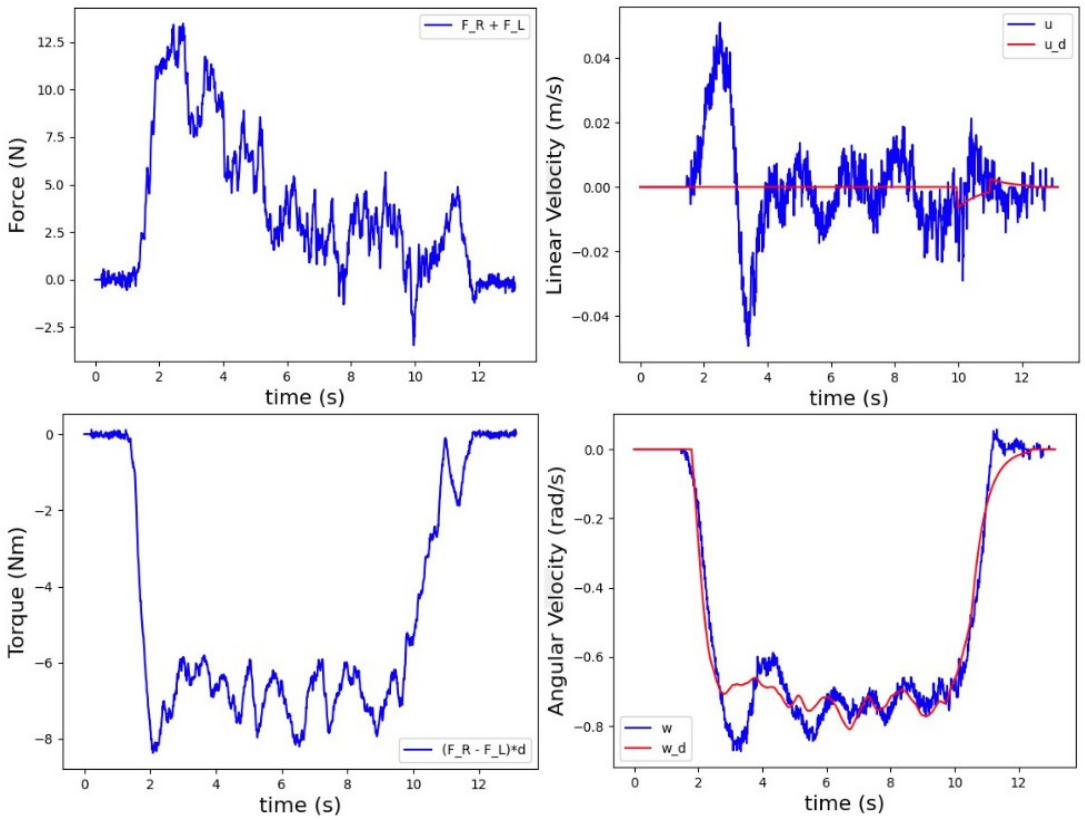


Fig 65. FSN: Rightwise 360º turn movement with MEDIUM speed.

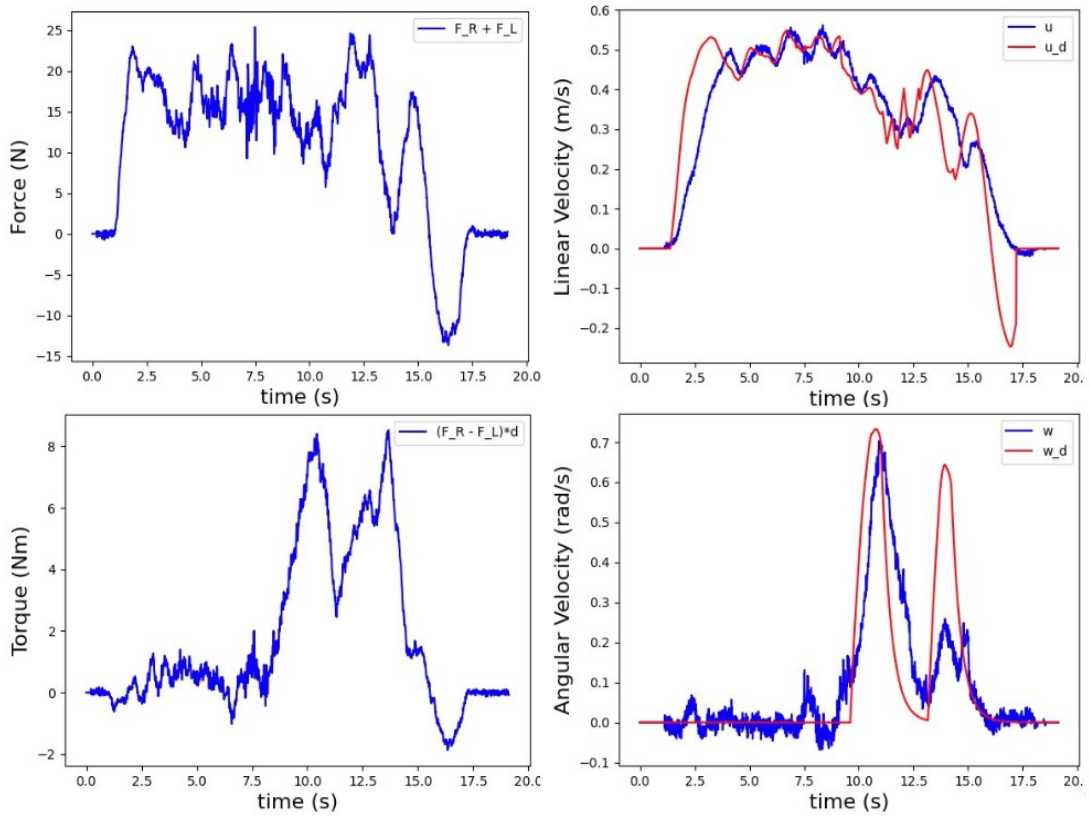


Fig 66. FSN: Leftwise L curve movement with SLOW speed.

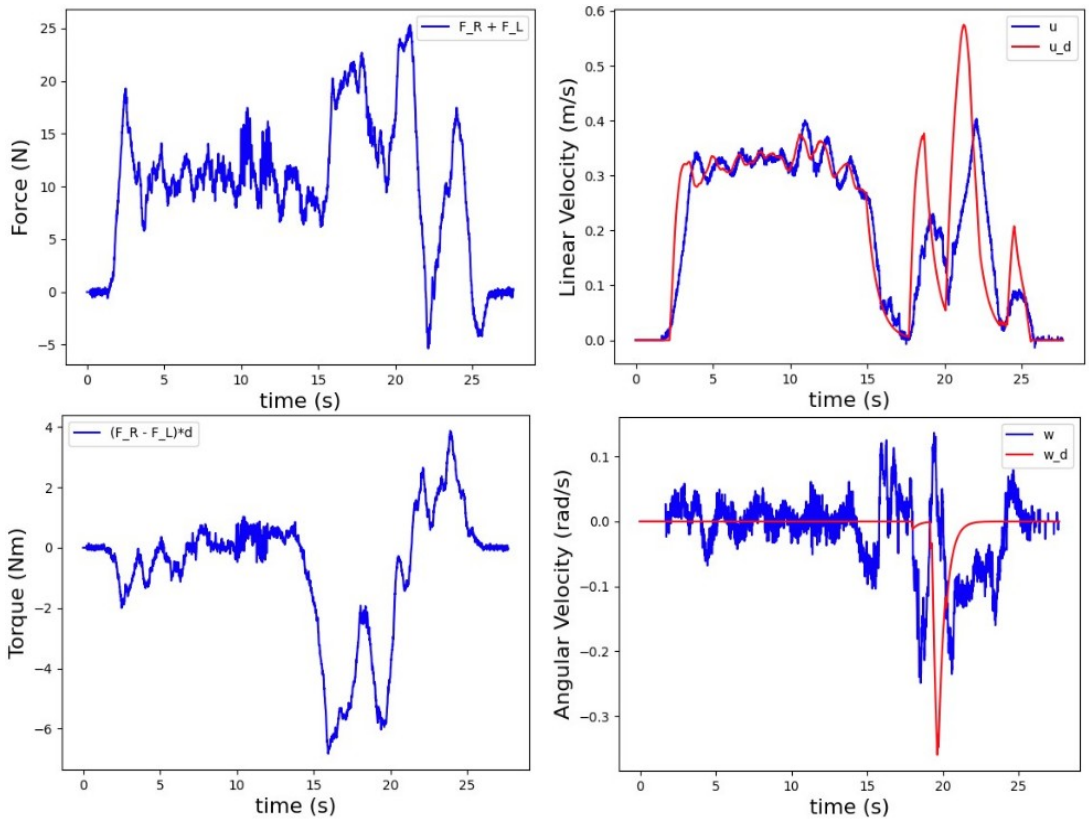


Fig 67. FSN: Rightwise L curve movement with SLOW speed.