

KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Engineering Science

Personalizing Assistive Robots

A reconfigurable and adaptable shared-autonomy framework and its applications

Santiago Iregui

Personalizing Assistive Robots

A reconfigurable and adaptable shared-autonomy framework and its applications

Santiago IREGUI

Examination committee:
Em. prof. dr. ir. Jean-Pierre Celis, chair
Em. prof. dr. ir. Joris De Schutter, supervisor
Dr. ir. Erwin Aertbeliën, supervisor
(Co-Supervisor)
Prof. dr. ir. Herman Bruyninckx, supervisor
(Co-Supervisor)
Prof. dr. ir. Tinne Tuytelaars
Dr. Gianni Borghesan
Prof. dr. ir. Eric Demeester
Dr. João Silvério
(DLR)

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor of Engineering Science (PhD): Mechanical Engineering

November 2023

© 2023 KU Leuven – Faculty of Engineering Science
Uitgegeven in eigen beheer, Santiago Iregui, Celestijnenlaan 300 box 2420, 3001 Leuven, Belgium (Belgium)

Alle rechten voorbehouden. Niets uit deze uitgave mag worden vermenigvuldigd en/of openbaar gemaakt worden door middel van druk, fotokopie, microfilm, elektronisch of op welke andere wijze ook zonder voorafgaande schriftelijke toestemming van de uitgever.

All rights reserved. No part of the publication may be reproduced in any form by print, photoprint, microfilm, electronic or any other means without written permission from the publisher.

Dedicated to my parents

Preface

When I was in Colombia finishing my master thesis, and wondering what to do next in life, I came across a Facebook post that said, in a very serious and professional “tone" in Spanish, that there was an open PhD position at KU Leuven. After clicking the link, I was fascinated by the description of the project and of the university. So I decided to write to Cristian Alejandro Vergara, who wrote the Facebook post, telling him a bit about me and asking him for his opinion about applying. After the most encouraging reply, I solved the mystery of what I wanted to do after completing my master’s degree.

I decided to write to the contact person of the application, Wilm Decré, while being bit hesitant and afraid about the level of formality I needed to maintain. In the email I expressed my intention of applying and sent him some questions about the application. I was very surprised that, after a few minutes, I received a reply where all my questions were solved in the kindest way possible. That email made me super enthusiastic about applying to a place where people were so kind and respectful.

After following the application procedure I had my first interview. I was pretty nervous and insecure about my "Sofia Vergara" English (read as “Inglich”) accent, and I felt that at any point I would not be able to answer one of the super tough questions that I was expecting. I was wrong. Joris, Erwin and Wilm were in the videocall, and they did not ask me tricky questions at all. All they wanted to know is a bit about myself and my personality, and tell me a bit about the different projects they had. After the first two minutes I lost the fear, and I found myself having a conversation with the kindest people. After a few weeks, I received an email from Joris giving me a warm welcome and offering me the position. I was very thrilled and excited, and I did not hesitate to accept the position.

During my five years in KU Leuven, I immensely appreciated the freedom that I was given in the lab, which copes very well with my stubborn personality. The lab is a place for people to freely explore ideas that sometimes result in minotaur-level mazes, where you continuously smash yourself against the walls while struggling to find a solution. And this is incredibly valuable, because when you find yourself a way out, you realize that you are not the same person as the one who entered the maze. The new person will now be able to find better and faster ways to solve that maze and many others that will show up in the future. This I think is one of the key characteristics of the lab: we are not paper-producing machines, but we are people exploring research ideas and learning in the process. And this is extremely important when the priority is to form good and ethical researchers. Such mentality, combined with many more aspects, makes me very proud of being able to say that I did my PhD in the robotics research team of KU Leuven.

Therefore, I would like to begin by thanking my promotor Joris De Schutter for all the guidance and the things he taught me. I learned an uncountable number of things from him (technical and non-technical), but one that stands out is scientific rigor in every sense of the word. Joris is a very transparent person that does not take any shortcut, and prefers not to publish over publishing something that does not comply with the highest standards. And from him I learned that performing quality research requires time and patience, so that the outcome is something worthy. We spent so much time discussing and making sure that our research was rigorous, that even sometimes discussions about the construction of a single sentence, or even the use of a single word, prolonged for hours. I would also like to thank Joris, because he did not only care about me in academical terms, but also as a person, always making sure that I was OK.

I would like to also thank Erwin Aertbeliën, my co-promotor, who also taught me an uncountable number of things and spent many hours having discussions with me. I knew that, whenever I got stuck, I could reach out to him, and he would have not only one but at least five different options for me to try for solving my problem. His feedback was never based on impositions or authority-based arguments, but always with constructive suggestions where at the end I am trusted with taking the right decisions. Frequently, Erwin's enthusiasm and admirable knowledge resulted in four/five hours meetings that were originally planned for a single hour. After those meetings I did not only have a clearer idea of how I could proceed with my research, but also I temporarily felt like a little expert on the history of robotics,

European history/geography, and Belgian politics. Sincerely, thank you!

I express my gratitude also to Herman Bruyninckx, “former basketball player" as referred to by google’s drop-down suggestions, who gave me lots of inspiration during my PhD. I had the pleasure to have some discussions with him, where he gave me constructive and supportive feedback. Herman is a very direct person, and he will not hesitate to point you towards your methodological flaws and shortcomings with very solid arguments. For this reason, when I began my PhD, I must confess that I was a bit scared of having him as part of my PhD jury. However, at this stage, I am very grateful because this had a very positive impact on my research and formation, and I am very proud that his name appears in my thesis. I would also like to thank Herman for giving me the chance to continue in the lab as a postdoctoral researcher; a position that comes with new challenges and exciting responsibilities.

To João Silvério, I would like to extend my gratitude for participating as member of my PhD examination committee and for coming all the way from Munich for my defense. The insightful discussions and impactful feedback resulted in a thesis with higher quality. He and the rest of the EDAN team (Samuel Bustamante, Gabriel Quere, Jörn Vogel, Freek Stulp, etc.) deserve a very special mention in my thesis for hosting me in their lab during my one-month research visit. I learned a lot from them and I hope to have more projects together in the future. A big thank you to Samuel, who took the initiative and made my research visit and our collaboration possible, and also took the time to show me the highlights of Munich.

To the rest of my examination committee, I would like to extend my deepest gratitude for the constructive feedback and positive comments that I received during the process. To Tinne Tuytelaars, who was also my professor on a course of vision and pattern recognition, I would like to extend a big thank you for her kindness and encouraging feedback during my preliminary defense and also during the rest of my intermediate presentations. A big thank you to Gianni Borghesan as well, who made some important remarks of my thesis and that, even if it took me a long time to address properly, at the end they resulted in a much clearer text. Also I would like to thank him for not revealing during my preliminary defense that he caught me several times playing Call of Duty with Fede on the fourth floor during my thesis writing period. Aditionally, a big thank you to Eric Demeester who took the time to come from Diepenbeek and gave me very useful feedback and insightful questions. Finally, to the chair of my PhD, prof. Jean-Pierre Celis, I would like to express my gratitude for taking the time to ensure that all

the requirements of my PhD are fulfilled and instructing me with the final steps to take for finalizing this journey.

Aditionally, even if Wilm Decré is not officially in my PhD examination committee, he had a big influence in my PhD and deserves a special mention. I have many things to thank him. As I mentioned in the beginning, when I first got a reply from my friend Wilm, I thought that he was a kind person. I was wrong[1]. I discovered that Wilm is not kind, but extremely kind[2]. He spoils us with cutting-edge gadgets and makes sure that we have all the necessary things to work comfortably and happily. With Wilm we were frequently the last ones leaving the office, and we got to talk (and sometimes even to gossip a little...) about a wide variety of topics. It is a friendship based on giving and receiving, as well was having and sharing. And the friendship that we give and have is shared and received[3]. Thank you for all the good laughs and also for all your valuable help during my PhD.

I would also like to acknowledge Carlos Pacho Rodriguez, Luis Muñoz, and Juan Sebastian Nuñez, who where my professors and mentors in Uniandes, Colombia, and that had a big influence in the earlier stages of my career. A big thank you for leading me towards the exciting path of doing research in robotics!

I would like to thank my ROB colleagues, who gave me a lot of inspiration and nice moments during my PhD: Maxim, Ali, Maria, Lander, Ajay, Nicole, Federico, Romulo, Riccardo, Arno, Boris, Renaud, Alexander, Benjamin, Johan, Ruan, Pinhao, Jasper and Louis. Especially, I enjoyed my daily walks to Alma (university restaurant) accompanied by NSFW jokes from Fede, Boris and Wilm. Without forgetting the colleagues that already left the group, I would like to thank Nico Huebel, Niko Tsiogkas, Yudha, Umut, Andrea, Vahid, Johan, Thomas, Juan, Cristian, Sergio, Enea and Marco for all the fun moments and valuable help. Also thanks to my Meco colleagues, specially Dani, Sonia, Alvaricoque, Alejandro, Mathias, Alex, Shamil, Jan, Ajay (yes, also from ROB), Dries, Louis, Flavia, Bastiaan, Wim and Laurens. You guys are all super nice people and I always enjoy crossing paths and smiles with you in the hallways. A special acknowledgement to Cristian and Federico, who closely collaborated with me and contributed to the development of my PhD.

[1] Wilm, please keep reading

[2] I told ya

[3] Dear reader, if you are not following, please do yourself a favor and watch the series Friends!

To my administrative colleagues Marina, Cindy, Marijke, Fabienne, Karin, Regine and Lieve a big applause to you because without your giant help I would not be graduating. Also to my colleagues Jean-Pierre, Tom, Eddy and Gunther, who work in the Electronics and Mechanical workshop, thank you for all the valuable help in this process (specially during the KUKA innovation award). To the ladies and gentlemen working in Alma, who served us the food with a smile and sometimes gave us extra veggies because at my 30s I'm still growing. Also to the cleaning gentleman who dealt with my super messy desk and always said a friendly hello.

I've met so many wonderful people in my life, that there is not enough space and time for mentioning and thanking all of them here. So, I would like to exclaim a big thank you to all the people that have shared their friendship with me during my life in Belgium. Being away from your hometown is not easy, and only the ones that have lived abroad can understand the void that missing your culture, food, friends, family, etc., leaves.

However, have been surrounded by incredible people from all around the world during these five years that I have spent in Belgium, have brought me great memories and joy. Therefore, to continue with this endless preface, I would like to express my deep gratitute (in no particular order) to Juli Prieto, Cata, Estefita bb, Falloncivirís, Vasiliki (my dear wife), Loli, Leja, Bea, Silvi R, Dianis, Silvi P, Juli Parra, Ana, Chiara M., Chiara G., Lau Gómez, Vale, Marlon, Fernando, Annet, Laura Robberechts, Jorge, Alvaro, Natha, Yamid, Marlly ("tu mercecita"), Guido (estrellita), Lau Diaz, Klaas, Julie, Dillamcho (my free lawyer), Linh, Fede, Lie, Salva, Cristian, Elena, Juan, Angeloski, Ludo, Bernardo, etc. I have so many great memories and trips with all of you that I cannot share them all. But I particularly remember the amazing trip to Vietnam with my dear friends Cristian, Lihn and her wonderful father, who, without understanding a single word from each other (he spoke only Vietnamese), managed to get us drunk and give us the most amazing time. Moreover, I will never forget the trip to Amersfoort for the wedding of Marlly and Guido, where I danced until not feeling my feet anymore. I also remember the amazing trip to the Veneto region with my dear friend Angela and her amazing parents, who hosted me and gave me the most delicious Italian food ever. Additionally, with "mi perrito" Fede, we rented a car and traveled through the south of Spain, where we enjoyed the beautiful beaches and landscapes, with a few surprises on the way thanks to the peculiar recommendations of Juan.

Doing a PhD is not easy at all, and I managed to maintain my sanity partially thanks to the people who brought music closer to my life. A big thank you

to the current and former members of our Sint Agatha's band: Flavia, Ruan, Frieder, Rick, Fabien, Juan, Maria, Natha and Silvi, who made me enjoy every rehearsal/gig with their music and company. Also my gratitude to Dani, Diego, Mañe, Babela, Ana, Aurorita, Juan Pardo, David, Jose and Fanor, who have shared with me their incredible talent to interpret music from the Caribbean coast of Colombia. We have shared many jams, concerts and workshops that have brought a little part of Colombia closer to my life. The person that introduced me to this exciting universe of Colombian afro-indigenous music is Camilo Pineda Angola, and deserves my deepest gratitude.

To my girlfriend Flavia, my little Korok who I love with all my heart, a big thank you for all these amazing months together. It has been a tough ride for me in my last year of PhD, and you have been there solid as a rock supporting me and feeding me with healthy food that keep me away from eating daily frozen pizzas (perhaps with a bit of pineapple on them...). You have cheered me up with hugs and Zelda afternoons when I needed it the most. You have made 2023 an incredible year, in spite of all the stress induced by writing this bestseller thesis. I love every bit of you, including your amazing parents, Marinella and Gianni, who have been incredibly kind to me, making me feel as part of the family. I look forward to discovering the world and living new adventures with you!

Now let's cross the Atlantic Ocean and land in the beautiful Republic of Colombia, where I had the fortune to share experiences with many people that had a great influence in my life.

I would like to thank my dear friends that I met during my middle/high school times: Camilo, Panfi, Mapis, Vale, Cheese, Gringo, Angie, Lorenita, Andrade (and his family, that was like my own), Paola M, Paola S, Juanja, Alejandra, Mari, Lili, Lulu (and her lovely family), Andre A, Andre D, Linis, Caro P, Alvarito (and his parents Pili and Alvaro who were like parents to me), Dani, Pablito, Toño, etc. A special mention to Caro P, who is the person who I have texted and video called more in the world by far, and knows a part of me that no-body else does. A special mention also to Panfi, Andrade and Alvarito, who I consider my brothers, and Lulu, who I consider my sister. Even if I don't talk to them so often because of the 8 800 kilometers between us, I love them and I know that I can always count on them. Additionally, thanks to my friends from the university: Apontito, Oscar, Diego, Capi, Charlie, Mapis, Nico, Deivid, Chiti, Ama, Ximenita, Nata R, Castro, Dani and Vivi.

Moving on to my family, I would like to thank my aunt Maria T and uncle Jorge for their important support during my bachelor studies, which made possible that I could do my bachelor studies in Uniandes. Also thanks to my aunt Mavis and uncle Santiago, who have been a great support for me and my parents, and to my cousins Maria Grazia and Maria Andrea who shared great moments with me. My gratitude to my aunt Pili, uncle Rafa and cousin Pipe(s), who were my closest family for a very long part of my life while I was living in Bogotá. Thanks to my aunt Mili and uncle Jesus, who spoiled me a lot during my childhood with amazing presents and are always there in the most important events of my life. Also thanks to my older brother Juanchito who was present in my childhood, who played with me Zelda (a link to the past) and other amazing Nintendo games, and who taught me cool drum grooves. You all have demonstrated that I can count on you “en las buenas y en las malas”.

To my grandparents Hernando, Alvaro, Aurita, Alicia and Mamima, who are not here anymore, I would like to extend a big thank you for so many good memories and for spoiling me when I was a child. I would have loved for you to see me defending my PhD.

Finally, I would like to express my deepest gratitude to my parents Vicky Rincón and Juan Iregui for their immense effort of raising me and teaching me the values that make me the person who I am today. I thank them for all the sacrifices that they had to make for giving me everything in this world. They taught me most of the things I know and thanks to their effort I could make my way towards this important milestone of my life. I could not have asked for better parents. I love you with all my heart.

Leuven, November 2023,
Santiago Iregui Rincón

Acknowledgements

The author gratefully acknowledges the financial support by: the European projects Project Factory-In-a-Day (FIAD) and AI-Prism; the Flanders Make projects YVES_SBO, SmartHandler_ICON and Proud_SBO, Pickit Challenge; and the KU Leuven C1 Project “Neurophysiological investigations of the human, and nonhuman primate brain: from perception to action" (3M180301).

Additionally, the author of this thesis gratefully acknowledges the help and the contributions performed by prof. Peter Janssen, Sofie De Schrijver, Jesus Garcia and Celia Bougou, for their help and their contributions and their collaboration during the BMI experiments and for providing relevant experimental results.

Furthermore, the author of this thesis gratefully acknowledges the help and the valuable contributions of Samuel Bustamante, Gabriel Quere, Jörn Vogel, João Silvério, and other members of the EDAN team at the Institute of Robotics and Mechatronics of the German Aerospace Center (DLR) during the author’s one-month research visit.

Finally, I would like to thank all the people who contributed to my user studies, and whose names I cannot reveal for ethical and legal reasons.

Abstract

The quality of life of people with motor impairments, often caused by spinal cord injuries, is severely reduced. Assistive robotic devices offer an opportunity to improve the quality of life of these people and help them regain a certain level of independence. A completely autonomous solution is often not desired since people have a need to keep their remaining autonomy or recover some of their lost autonomy, while feeling in control as much as possible. Therefore, robotic assistive solutions that share the autonomy between human and computer have been proposed. These solutions assist by granting the users only partial control over the robot.

However, these solutions often provide limited possibilities for personalization and customization, ignoring individual mobility difficulties and preferences. This trend limits people's access to suitable assistive devices that comply with their needs and preferences, since personalization of such devices requires high technical expertise, development time, and therefore high associated costs. Research shows that the lack of personalization also causes that users of these technologies often abandon them, either because the device never fulfilled the user's requirements, or because the requirements changed over time. The lack of user-centered and personalizable robotic devices leaves a clear gap between the features of the available assistive devices and the users' particular needs and preferences.

In order to fill this gap, as a first contribution, a constraint-based reactive assistive framework for the control of assistive robots is proposed, which facilitates the personalization of assistive robotic devices such as table-mounted or wheelchair-mounted robot arms. This enables the robot to comply with the particular needs of users with different levels of mobility. For this purpose, the framework encapsulates reactive assistive behaviors into a set of building blocks that can be created and reused in

different assistive scenarios. For example, these building blocks enable a continuous reaction to input from human-machine interfaces (HMI) and sensor measurements, enabling human-robot interactivity while providing robustness against changes in the environment. These building blocks are also (re-)combined to create personalized assistive strategies and are modulated to shift the level of the provided assistance according to the situation.

As a second contribution, the thesis presents several of reusable and combinable building blocks that can be used to describe and execute personalized assistive strategies that are suitable during activities of daily living. The thesis focuses on activities of daily living that are assumed to occur in unstructured environments (i.e. only little information about the objects and the environment is known beforehand) such that, in the future, they can be applied outside laboratory conditions. Thereby, most of the required information is gathered through measurements and/or estimates of multiple sensors and HMIs, and is used to reactively adapt the provided assistance to deal with changing environments and user intentions.

As a third contribution, the thesis presents extensive experimental validation of the framework and building blocks in eleven widely differing use cases, where the reconfigurability and adaptability features of the framework are highlighted. The use cases illustrate these features by changing aspects of the assistance such as: robotic devices (five robots were used in total), suitable HMIs to interact with the system (e.g. eye tracker, brain-machine interface, SpaceMouse, etc.), sensors to measure the environment (e.g. load cell, 2D and 3D cameras, etc.), type of autonomy (e.g. adaptable autonomy), activities of daily living (e.g. reaching objects), and other important aspects that provide possibilities of personalization according to mobility limitations of targeted users. In addition, a user study was performed to evaluate one of the use cases, where the proposed methods assist users during tasks that involve interaction with articulated objects such as opening doors with unknown dimensions. This study statistically demonstrates the qualitative and quantitative improvements achieved through the proposed methods.

It is hoped that future assistive robotic devices may benefit from this work, and that this will impact the quality of life of people requiring personalized robotic assistance in their daily lives.

Beknopte samenvatting

De levenskwaliteit van mensen met motorische beperkingen, vaak veroorzaakt door ruggenmergletsels, is ernstig verminderd. Ondersteunende robots bieden een kans om de levenskwaliteit van deze mensen te verbeteren en hen te helpen een zekere mate van onafhankelijkheid terug te krijgen. Een volledig autonome oplossing is vaak niet gewenst, aangezien mensen behoefte hebben om hun resterende autonomie te behouden of deels te herstellen. Ze willen zoveel mogelijk controle behouden. Daarom zijn er ondersteunende robot systemen voorgesteld die de autonomie delen tussen mens en computer. Deze oplossingen helpen door de gebruikers slechts gedeeltelijke controle over de robot te geven.

Deze oplossingen bieden echter vaak beperkte mogelijkheden voor personalisatie en maatwerk, waarbij individuele mobiliteitsproblemen en -voorkeuren worden genegeerd. De toegang tot geschikte ondersteunende robots die voldoen aan de juiste behoeften en voorkeuren is immers beperkt doordat personalisatie een hoge technische expertise, een lange ontwikkelingstijd en dus een hoge bijbehorende kost vereist. Onderzoek toont aan dat het gebrek aan personalisatie er ook voor zorgt dat gebruikers van deze technologie deze vaak verlaten, omdat het apparaat nooit voldeed aan de eisen van de gebruiker, of omdat de eisen in de loop van de tijd veranderden. Het gebrek aan op de gebruiker gerichte en personaliseerbare ondersteunende robots zorgt ervoor dat er een duidelijke kloof bestaat tussen de beschikbare ondersteunende robots en de specifieke behoeften en voorkeuren van de gebruikers.

Dit proefschrift overbrugt deze kloof door, als eerste bijdrage, een op beperkingen gebaseerd reactief raamwerk voor te stellen voor de aansturing van ondersteunende robots dat de personalisatie naar verschillende robotconfiguraties vergemakkelijkt, zoals een ondersteunde robotarm die op

vast op tafel of op een gemotoriseerde rolstoel is gemonteerd. Dit maakt het mogelijk om aan de specifieke behoeften van gebruikers met verschillende niveaus van mobiliteit te voldoen. Het voorgestelde raamwerk stelt het reactief gedrag van een ondersteunende robot voor in een reeks bouwstenen die gecreëerd en herbruikt kunnen worden in verschillende scenario's voor ondersteuning. Deze bouwstenen maken een continue reactie mogelijk op input van mens-machine-interfaces (human-machine interfaces, HMI) en input van sensoren die veranderingen in de omgeving meten. Dit zorgt voor robuustheid en laat toe de mens-robot-interactiviteit nauwkeurig te beschrijven en uit te voeren. Deze bouwstenen kunnen herbruikt worden om gepersonaliseerde hulpstrategieën te creëren en gemoduleerd worden om het niveau van de verleende hulp te verschuiven naargelang de situatie.

Als tweede bijdrage presenteert dit proefschrift verschillende herbruikbare en combineerbare bouwstenen die geschikt zijn om gepersonaliseerde hulpstrategieën te beschrijven en uit te voeren voor activiteiten van het dagelijks leven. Het proefschrift richt zich op activiteiten van het dagelijks leven waarvan wordt aangenomen dat ze plaatsvinden in ongestructureerde omgevingen. Er wordt aangenomen dat er vooraf slechts weinig informatie is gekend over de omgeving en voorwerpen in de omgeving, zodat de resultaten mogelijk ook buiten laboratoriumomstandigheden kunnen worden toegepast. Daarom wordt de benodigde informatie verzameld via metingen en/of schattingen van meerdere sensoren en HMIs, en gebruikt om de geboden hulp reactief aan te passen aan veranderende omgevingen en gebruikersintenties.

Als derde bijdrage presenteert het proefschrift uitgebreide experimentele validatie van het raamwerk en de bouwstenen in elf sterk verschillende gebruiksscenario's, waarbij de herconfigureerbaarheid- en aanpasbaarheid wordt onderzocht. De gebruiksscenario's illustreren deze kenmerken door aspecten van de ondersteuning te veranderen, zoals: robotapparatuur (in totaal werden vijf robots gebruikt), geschikte HMI's voor interactie met het systeem (bijv. eyetracker, hersen-machine-interface, SpaceMouse, enz.), sensoren om de omgeving te meten (bijv. loadcel, 2D- en 3D-camera's, enz.), het type autonomie (bijv. aanpasbare autonomie), activiteiten van het dagelijks leven (bijv. het bereiken van objecten) en andere belangrijke aspecten die mogelijkheden tot personalisatie bieden op basis van mobiliteitsbeperkingen van gerichte gebruikers. Daarnaast werd een gebruikersonderzoek uitgevoerd om een van de gebruiksscenario's te evalueren, waarbij de voorgestelde methoden gebruikers assisteren in interactietaken met gelede objecten, zoals bijvoorbeeld nodig voor het

openen van ongekende deuren. Deze studie toont statistisch de kwalitatieve en kwantitatieve verbeteringen aan behaald door het gebruik van de van de voorgestelde methoden.

Er wordt gehoopt dat toekomstige ondersteunende robots voordeel kunnen halen uit dit onderzoek en dat dit een impact zal hebben op de levenskwaliteit van gebruikers die ondersteunende robots nodig hebben voor hun dagelijks leven.

List of Abbreviations

AO Articulated object. 130–133, 139, 145, 147, 152, 158

BB Building Block. 72, 75, 77–81, 89, 96–98, 106, 111, 113, 128, 161, 162, 165

BMI Brain-Machine Interface. 18, 22, 23, 40, 92–94, 96, 99, 105, 106, 163

DMP Dynamic Movement Primitive. 49

DOF Degrees of Freedom. xxviii, xxxiv, 39, 46, 48, 50, 52–56, 65, 66, 68–71, 84, 92, 108, 161, 177

eTaSL expression-graph based Task Specification Language. 46, 49, 56, 57, 71, 72

GMMs Gaussian Mixture Models. 49

HMI Human-Machine Interface. xxxiii, xxxix, 7–9, 19–31, 33, 35–42, 68–73, 79, 81–83, 85, 86, 89, 90, 94, 161–165, 167, 168, 183

iTaSC Instantaneous Task Specification using Constraints. 48

LfD Learning from Demonstrations. 46, 49, 65, 66, 74, 132

NR Not relevant to be disclosed for the purpose of the discussion. 188–191

PDDL Planning Domain Definition Language. 126

PMF Positioning Mobile with respect to fixed Frame. 49

PPCA Probabilistic Principle Component Analisis. 46, 50, 53, 65, 74, 97, 105

ProMPs Probabilistic Movement Primitives. 70

QR "Quick Response" codes. 87, 187, 188

RAM Reactive Action Model. xxix, 72, 74, 75, 77, 78, 89, 161, 172

RAMo Reactive Approach Motions. 41, 46–48, 56, 62, 66, 167

RVGF Reactive Virtual Guidance Fixture. xxxiv, 75, 77, 78, 82, 83, 85, 87, 96–99, 101, 102, 106, 114, 166, 167

SIM Sensor Interaction Model. xxix, 38, 72–75, 77, 79, 81, 82, 87, 89, 112, 161, 162, 167

SoT Stack of Tasks. 49

TCP Tool Center Point. 192

TRL Technology Readiness Level [50]. 105, 164, 167

UC Use Case. 95

VGF Virtual Guidance Fixture. 70

WHO World Health Organization. 1, 3, 4

List of Symbols

$\boldsymbol{\chi}$	vector containing all task-dependent feature variables of the task specification, which are also decision variables of the optimization problem
$\boldsymbol{\chi}_{\text{am}}$	a vector containing the DOF feature variables related to both, the surface and the tool, of the allowable manifold
$\boldsymbol{\chi}_{\text{lv}}$	vector containing all task-dependent feature variables of the task specification, which are also decision variables of the optimization problem
$\boldsymbol{\chi}_{\text{surf}}$	a vector containing the DOF feature variables related the surface of the allowable manifold
$\boldsymbol{\chi}_{\text{tool}}$	a vector containing the DOF feature variables related the tool of the allowable manifold
$\boldsymbol{\chi}_{B}$	vector containing the feature variables representing the amplitudes of the deviation functions in the binormal direction
$\boldsymbol{\chi}_{N}$	vector containing the feature variables representing the amplitudes of the deviation functions in the normal direction
$\boldsymbol{\Gamma}$	a pose consisting on a position vector and a rotation matrix)
$\boldsymbol{\Sigma}$	covariance Matrix
$\boldsymbol{\Theta}$	a set containing the rotation DOF, associated to the tool, of the allowable manifold
$\boldsymbol{\Phi}$	a set containing the position DOF, associated to the tool, of the allowable manifold

$\boldsymbol{b}$	mean value of the demonstrations
$\boldsymbol{f}_{\text{lm}}$	learned motion model (lm=learned model)
$\boldsymbol{I}$	Identity matrix
$\boldsymbol{p}(\boldsymbol{q})$	position of the end effector (which is function of $\boldsymbol{q}$)
$\boldsymbol{q}$	joint angles
$\boldsymbol{R}(\boldsymbol{q})$	rotation matrix expressing the orientation of the end effector (which is function of $\boldsymbol{q}$)
$\boldsymbol{W}$	matrix containing learned basis functions in the columns
$\hat{w}_i$	weight of constraint i without the regularization factor
μ	regularization factor
${}^{\mathrm{r}_1}_{\mathrm{r}_2}\mathbf{R}$	a rotation matrix in $\mathbb{R}^{3\times3}$ expresses a relative orientation of frame $\{\mathrm{r}_1\}$ with respect to frame $\{\mathrm{r}_2\}$. (some/all indeces might be omitted if the context is clear)
${}_{\mathrm{r}}\boldsymbol{p}^{\mathrm{a,b}}$	a position vector in $\mathbb{R}^3$ that represents the origin of a frame $\{\mathrm{b}\}$ with respect to the origin of a frame $\{\mathrm{a}\}$, expressed with respect to a frame $\{\mathrm{r}\}$ (some/all indeces might be omitted if the context is clear)
${}_{r}\mathbf{t}^{p}$	a twist in $\mathbb{R}^6$ of a moving body is a vector containing, in that order, angular velocity of the moving body and linear velocity of the origin of a frame $\{\mathrm{p}\}$ attached to the moving body, both expressed in a given frame $\{\mathrm{r}\}$ (some/all indeces might be omitted if the context is clear)
${}_{r}\mathbf{w}^{p}$	a wrench in $\mathbb{R}^6$ applied to a body is a vector containing, in that order, applied force and applied moment in the origin of a frame $\{\mathrm{p}\}$ on the body, both expressed in a given frame $\{\mathrm{r}\}$.
s	Normalized progress path variable, $0 \leq s \leq 1$
w_i	weight of constraint i including the regularization factor

Definition of Key Concepts

- **Adaptability:** Property that enables a system to perform **online** modifications in order to comply to changing conditions. In this thesis, the system refers specifically to an assistive robotic device and the changing conditions are related to, for example, the mutable environment, user intention, status of the task, etc.
- **Arbitration:** modulation of the autonomy of the user and the robot.
- **Articulated object:** objects composed by a set of bodies connected through joints/articulations (such as doors, drawers, laptops, jars, etc).
- **Assistive products:** *"Any external product (including devices, equipment, instruments or software), especially produced or generally available, the primary purpose of which is to maintain or improve an individual's functioning and independence, and thereby promote their well-being. Assistive products are also used to prevent impairments and secondary health conditions."* Definition taken from [2].
- **Assistive strategy:** is a task specification in which the robotic system attempts to provide assistance to the user for completing a certain task. In the context of this thesis, the assistive strategy is created by combining different building blocks.
- **Assistive technology:** *"Assistive technology is the application of organized knowledge and skills related to assistive products, including systems and services. Assistive technology is a subset of health technology."* Definition taken from [2].
- **Customization:** refers to the action made by users to modify the system to fit their own needs.

- **Interactive perception:** Perception that requires or exploits interaction. Online Interactive perception is performed when information about the object is acquired through interaction rather than after the interaction has been completed [70].
- **Levels of Autonomy:** Autonomy over different aspects of a task can be allocated (i.e. granted) to the user and/or to the robot. The allocated autonomy can be modulated online, such that it can gradually shift between user and robot. In this thesis, the allocated autonomy and the ability to modulate it is referred to as a level of autonomy. There can be multiple levels of autonomy in parallel that relate to different aspects of the task. These levels of autonomy can be modulated automatically or by user commands.
- **Personalization:** refers to an action done by a stakeholder different from the user to modify the system to fit the user's needs.
- **Pose:** Mathematical representation of the translation and orientation of a rigid body [19].
- **Reactive Robot Behavior:** series of actions (at the continuous level) taken by the robot which tend to follow a nominal plan, but having the ability to reactively deviate from that plan (e.g. when interacting with the human or with the environment). These behaviors are defined through task specifications and executed by closed-loop controllers.
- **Reconfigurability:** Property of a system that enables a stakeholder (e.g. the user, caregiver, etc) to perform **offline** modifications to it in order to comply to new requirements. In this thesis, the system specifically refers to an assistive robotic device and the new requirements are related to the (changing) assistance needs/preferences of a specific user.
- **Screw:** Six-dimensional vector constructed from a pair of three-dimensional vectors [19].
- **Shared Autonomy and Shared Control:** both terms refer to a paradigm used to assist users while performing a task. In this paradigm, the input from the user (e.g. given through an HMI like a joystick) is not taken directly to define the (continuous) commands that the robot should do. Instead, the robotic system takes the desired commands of the user and desired commands produced by the robot itself, and combines them (e.g. with a weighted average) to define the

control actions that are executed in order to fulfill a task. In other words, the control (or the autonomy) over the robot is shared between the user and the robot itself. In this thesis, and in most literature, the terms shared autonomy and shared control are used interchangeably to refer to the same concept.

- **State coordinator:** refers to a software component responsible for coordinating the execution of different states related to a subsystem, e.g. Finite State Machines (FSMs), Petri nets, etc.
- **Task:** refers to an abstraction of an action that occurs repetitively in different activities of daily living and that can be fulfilled by the robot through the execution of a task specification.
- **Task specification:** set of constraints that determine how the states of the robot (at the continuous level) will evolve, derived from how the robot should nominally move and how the robot should interact with the human, the environment, and the workpieces.
- **Twist:** Screw vector containing angular and linear velocity vectors [19].
- **Wrench:** Screw vector containing force and moment vectors [19].

Contents

Abstract xiii

Beknopte samenvatting xv

List of Abbreviations xx

List of Symbols xxii

Definition of Key Concepts xxiii

Contents xxvii

List of Figures xxxiii

List of Tables xxxix

1 Introduction 1

1.1 Problem Statement and Motivation 1

1.2 Research Objectives 6

1.3 Overview of the thesis 6

1.3.1 User-centered assistance and its stakeholders 7

1.3.2 Considered assistive robotic devices 10

1.3.3 Reactive constraint-based shared autonomy 12
1.3.4 Considered use cases 18
1.3.5 Considered HMI and sensors 25
1.4 Configuration of Human-Machine interfaces 29
1.4.1 Configuration of HMI transducers 30
1.4.2 Co-existing interaction modalities 36
1.4.3 User-centered HMI configurations for assistive strategies 37
1.5 Contributions 38
1.5.1 Contribution 1 – reconfigurable and adaptable assistive framework 38
1.5.2 Contribution 2 – assistive behaviors for activities of daily living 39
1.5.3 Contribution 3 – experimental validation of heterogeneous configurations 40
1.6 Thesis Outline 40

2 Generating Reactive Approach Motions Towards Allowable Manifolds using Generalized Trajectories from Demonstrations 45

2.1 Introduction 46
2.2 Related Work 48
2.3 Reactively Adaptable Trajectories 50
2.3.1 Position profile 50
2.3.2 Orientation profile 53
2.4 Allowable Manifolds 53
2.4.1 DOF related to the surface 53
2.4.2 DOF related to the tool 54
2.4.3 Allowable manifold expression 54
2.5 Specification of the Reactive Control 56

2.5.1 Constraints Related to the Allowable Manifolds 57
2.5.2 Constraints Related to the Trajectory Generation . . . 58
2.5.3 Constraints Related to the Environment 59
2.5.4 Additional constraints 60
2.6 Validation Use Cases 60
2.6.1 Grasping of objects with gripper 61
2.6.2 Bin-picking use case 62
2.7 Discussion and Conclusion 65

3 Reconfigurable Constraint-Based Reactive Framework for Assistive Robotics with Adaptable Levels of Autonomy 67

3.1 Introduction 68
3.2 Related Work 70
3.3 Task Specification and Control in eTaSL 71
3.4 Reactive Assistive Behaviors 72
3.4.1 Sensor interaction models (SIMs) 73
3.4.2 Adaptive paths (RAMs) 74
3.4.3 Attraction to intent-inferred targets (RAM) 75
3.4.4 Automatic orientation path (RAM) 75
3.4.5 Reactive Virtual Guidance Fixtures (RAM) 77
3.4.6 Automatic Reactive Approach Motion (RAM) 78
3.5 Levels of Autonomy 79
3.6 Intent Estimation 83
3.7 Validation Use Cases 85
3.8 Discussion and Conclusion 89

4 Rapid development of assistive applications through the use of reconfiguration: presenting two assistive scenarios 91

4.1 Introduction 92
4.2 Assistive Scenario 1: assistance for controlling a robot arm with monkey users through brain-decoded signals 92
4.2.1 Research context and challenges 92
4.2.2 BMI-based robot control: an overview 94
4.2.3 Assistive strategies for BMI decodings 95
4.2.4 Implementation and results 99
4.2.5 Discussion and conclusions 105
4.3 Assistive Scenario 2: assistance for wheelchair-mounted robot arm EDAN 107
4.3.1 Research context and challenges 107
4.3.2 Assistive strategy 109
4.3.3 Intent estimation 114
4.3.4 Control modes 116
4.3.5 Implementation and results 120
4.3.6 Discussion and conclusions 124
4.4 General conclusions 127

5 RAMASA: Robotic Assisted Manipulation of Articulated objects with force-based Shared Autonomy 129

5.1 Introduction 130
5.2 Related Work 131
5.3 Representation of Geom. Primitives and Transformations . . 133
5.4 Force-based shared autonomy 134
5.4.1 Velocity-resolved control module 135
5.4.2 Online interactive perception module 138
5.4.3 Automatic assistance module 146

5.5 User Study 150
5.6 Discussion and Conclusion 157

6 Conclusions and Limitations **161**

A Appendix **169**

A.1 Dynamics and Stability of eTaSL 169
A.1.1 Stability example 171
A.1.2 Automatic Reactive Approach Motion Analysis 172
A.2 Online intent estimation algorithm for assisted reaching tasks 176
A.2.1 Intent estimation plots 176
A.2.2 Combination of online intent estimation and shared autonomy for reaching tasks 180
A.3 HMI configurations 183
A.3.1 SpaceMouse 183
A.3.2 Eye tracker 186
A.3.3 3D camera for hand pointing detection 189
A.3.4 Kinesthetic HMI 191
A.3.5 BMI or BCI 192
A.3.6 Sound HMI (Polycom system) 194
A.3.7 Screen monitor 194

Bibliography **195**

Curriculum Vitae **207**

List of publications **209**

List of Figures

1.1 Demographic charts of EU showing how the population will age by 2100 compared to the year 2021 2

1.2 Aspects of the framework through which customization and personalization of assistive devices can be achieved 7

1.3 Scheme showing a proposed design workflow describing the roles of the main stakeholders involved in the software development and deployment of a user-centered assistive robotic device. 10

1.4 Classification of rehabilitation robots. 11

1.5 Robotic devices used in the use cases presented in this thesis 12

1.6 Task specifications of a door opening task using force and velocity controlled actions, expressed in two alternative task frames, following the task frame formalism. 14

1.7 Snapshots of the human interacting with the robot through an artificial skin. The robot is constrained to be on an allowable manifold 20

1.8 Use cases reconfigured to assist specific mobility limitations during reaching tasks 22

1.9 Diagram that shows how an assistive robotic device (machine) communicates with the user (human) through a collection of HMIs 29

1.10 Conceptual scheme that shows how the output of $H \rightarrow M$ transducers, which represent interpreted human intentions, can be connected to different subsystems of the robot 31

1.11 Scheme showing in what parts of this thesis the different roles assigned to the different stakeholders are discussed, introduced or exemplified. 43

2.1 Snapshots of a human–robot collaborative bin picking application 47

2.2 Learned model to approach to the bin 51

2.3 2D representation of local trajectory deformations 52

2.4 Graphical representation of a tool approaching towards a surface that models the geometry of an object 55

2.5 Example of possible allowable DOF for selected tools 55

2.6 Snapshots for dynamic obstacle avoidance in simulation . . . 62

2.7 Control signals related to the grasping of objects with gripper use-case 63

2.8 Snapshots of the point-cloud visualization in the bin-picking experiment 64

3.1 A scheme of how the components of the framework are connected 73

3.2 Snapshots of how the RVGF adapts its shape for a shifting target 75

3.3 Snapshots of the RVGF adapting to a dynamic obstacle in ahuman-in-the-loop simulation 78

3.4 Multiple paths obtained by solving the simplified dynamics with several initial values 80

3.5 Definition of the ascending or descending piecewise-continuous function in terms of the arbitration parameters 82

3.6 Example of regions depicting the levels of autonomy achieved with intent estimation 85

3.7 Snapshots of the use cases proposed for different target users 86

3.8 Control signals of an experiment performed for validation use case 1 88

3.9 Control signals of an experiment performed for validation use case 4 88

4.1 Control scheme used in the use cases related to the assistive scenario 1 with the brain-machine interface 95

4.2 Photos of the setup used for decodings of the target object . . 99

4.3 Snapshots of a trial of UC6, where the target object is decoded from the monkey's neural activity through the BMI, and subsequently the robot executes a trajectory to grasp it . . . 100

4.4 Snapshot of what the monkey observes through the screen at the beginning of each trial of use case 7, where the end effector of the robot starts resting in the center of a virtual vertical plane while a yellow cue indicator blinks at this same position 101

4.5 Snapshots of reaching three targets during three trials of UC7 without the assistance provided by the RVGF 103

4.6 Snapshots of reaching three targets during three trials of UC7 with the assistance provided by the RVGF 104

4.7 Plots comparing the success rate of the 7 trial sessions without and with assistance 104

4.8 Average of the success rates of all trial sessions, were each session has the same weight regardless of the number of trials 105

4.9 Two examples of tasks that are tackled within the assistive scenario 2 through the use of the EDAN platform. Both tasks involve the pick and place of objects which are placed with a grid-like pattern on a flat surface 109

4.10 2D scheme that illustrates how the user can interact with the GET-ME virtual fixture to switch from a first object to a second object 111

4.11 Illustration of some properties of the GET-ME virtual fixtures 115

4.12 Illustration of the loss function criteria used to predict the most likely object 116

4.13 Control scheme used to control EDAN 118

4.14 Snapshots of UC8. A pick-and-place task with cups is being executed using EDAN with assistance given by GET-ME virtual fixtures 122

4.15 Snapshots of UC9, where a mockup caregiver is hand guiding the robot. Despite the assistive strategy being the same as in the two other use cases (i.e. same building blocks and levels of autonomy, without admittance constraints), hand guiding is enabled with GET-ME virtual fixtures active. 123

4.16 Snapshots of UC10, where a user controlling EDAN (both arm and wheelchair) through the SpaceMouse, with the impedance whole-body controller active, while being assisted by the GET-ME virtual fixtures. 124

4.17 Scheme showing a proposed design workflow describing the roles of the main stakeholders involved in the software development and deployment of a user-centered assistive robotic device. This diagram was duplicated from Fig. 1.3 for convenience of the reader 127

5.1 Scheme that shows how the three modules of the proposed force-based shared autonomy method are interconnected . . 135

5.2 Frames that are relevant for the robot control 135

5.3 Block diagram that summarizes the velocity-resolved control module 138

5.4 Block diagram illustrating the twist generation step of the automatic assistance module 149

5.5 Tasks performed by the participants during the user study . . 151

5.6 Photo showing the setup arrangement in the laboratory for the user study 152

5.7 Average of the NASA TLX weights determined by the second part of the questionnaire, indicating the perceived importance of each of the subscales 154

5.8 Data overlaid on a Boxplot for each qualitative metric, comparing the two variations of the tasks (with and without assistance) . . . 154

5.10 Data overlaid on a Boxplot for each quantitative metric, comparing the two variations of the tasks (with and without assistance). . . . 157

5.11 Difference between the quantitative metrics between executions without and with assistance for the pump task for each participant . . . 157

5.12 Difference between the quantitative metrics between executions without and with assistance for the box task for each participant . . . 158

A.1 Simulation solution with feedforward. . . . 175

A.2 Simulation solution without feedforward. . . . 175

A.3 3D plots to compare the influence on the intent estimation parameters on the likelihood . . . 177

A.4 3D plots to compare the influence on the intent estimation parameters on the posterior probability. . . . 178

A.5 3D plots to compare the influence on the intent estimation parameters on the Mahalanobis distance . . . 179

A.6 3D plots to compare the influence on the intent estimation parameters on the posterior probability. . . . 180

A.7 2D plots to compare the influence on the intent estimation parameters on the shared autonomy regions, by not including the rejection class O_{none} . . . 181

A.8 2D plots to compare the influence on the intent estimation parameters on the shared autonomy regions . . . 182

A.9 Picture of the used SpaceMouse with indication where the different transducers are located. . . . 184

A.10 Picture of the used eye tracker with indication of where the different transducers are located. . . . 190

List of Tables

1.1 Impairment types reported by disabled people in the United Kingdom in 2020/2021 3

1.2 Comparison between the different use cases 26

1.3 Functionalities assigned to different HMIs for the use cases shown in this thesis 28

1.4 Outputs taken from different sensors used in the demonstrated use cases in this thesis 28

2.1 Constraint Parameters 62

3.1 Specification of BBs that create the reactive behaviors used in the validation use-cases 76

3.2 Specification of five adaptable levels of autonomy and the involved weight factors 83

3.3 Validation Use Cases 86

4.1 Comparison between the two use cases related to the BMI decodings 96

4.2 Specification of BBs that are used to generate the automatic approach motion for use case UC6 98

4.3 Specification of BBs that are used to generate the tube-shaped RVGF for use case UC7 98

4.4 Success rate and number of trials for the experiments done without assistance (WOA) and with assistance (WA) 103

4.5 Specification of BBs that are used to generate GET-ME virtual fixtures 112

4.6 Specification of levels of autonomy and the involved weight factors for the GET-ME virtual fixtures 114

4.7 Control modes used in the three use cases related to EDAN . 121

5.1 Chosen selection parameters that compose the selection matrix for each interaction type 148

5.2 One-tailed paired-sample t-test analysis for the qualitative metrics based on the NASA-TLX questionnaire 155

5.3 One-tailed paired-sample t-test analysis for the quantitative metrics 156

A.1 Specification of five levels of autonomy and the involved weight factors for the simplified example 175

A.2 Values used for the parameters of the different example variations to show the influence their influence in the intent estimation algorithm. 177

A.3 Specification of BBs that create the reactive behaviors related to the intent estimation 181

Chapter 1

Introduction

1.1 Problem Statement and Motivation

There is an increasing need of caregivers and assistive products in the world due to a rise in the population that depends on them. One of the reasons is that the percentage of elder population of the world, and in particular of Europe (see figure 1.1), is increasing fast. According to the World Health Organization (WHO) in [3], 1 in 6 people in the world will be aged 60 years or over by 2030. Furthermore, by 2050, the world's population of people aged 60 years and older will double its size, compared to the population of 2022. Also, the number of persons aged 80 years or older is predicted to triple within 2020-2050. These numbers bring many concerns to our society, and one of them is that there will be an increasing necessity (and potentially a shortage) of caregivers and assistive products.

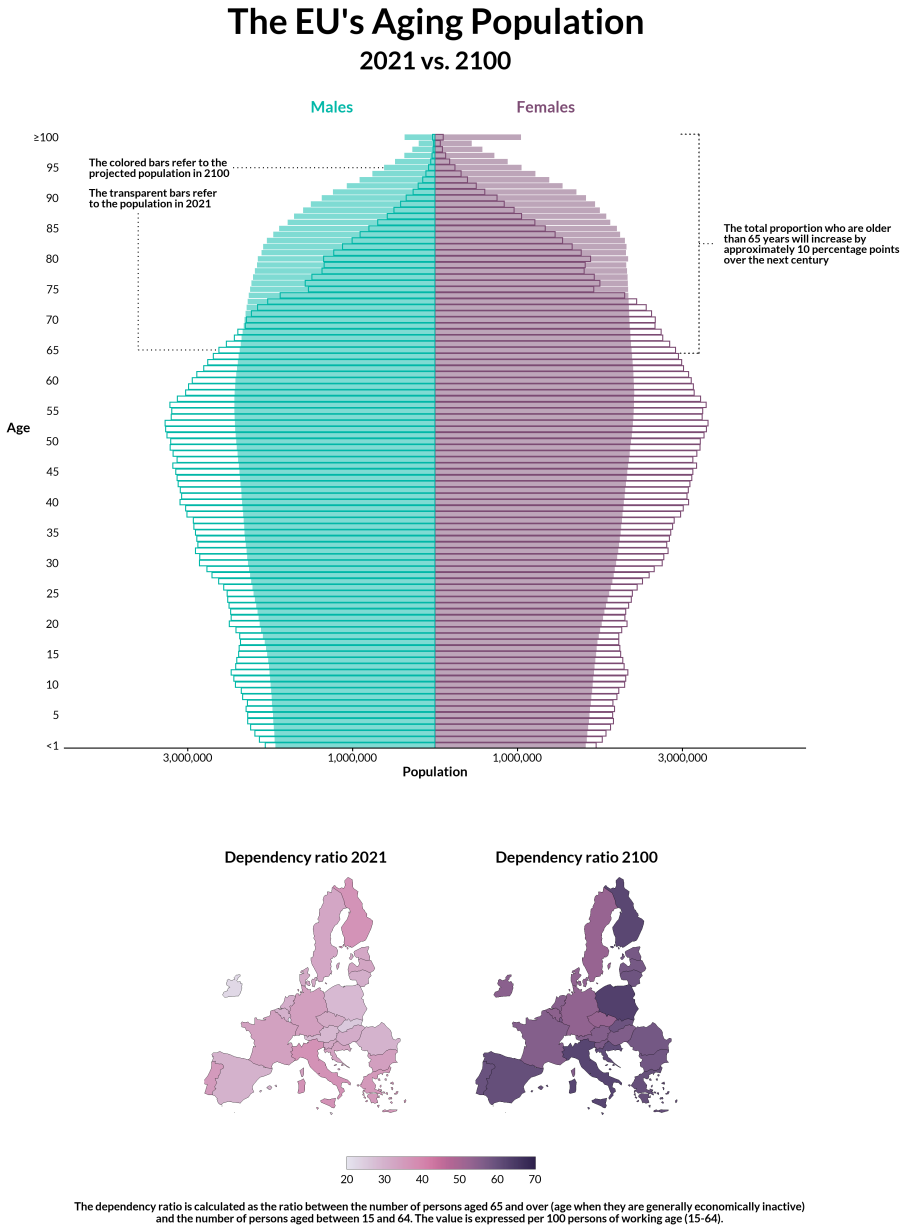


Figure 1.1. Demographic charts of EU showing how the population will age by 2100 compared to the year 2021. The chart was reprinted from [42] and contains data from Eurostat.

Table 1.1. Impairment types reported by disabled people in the United Kingdom in 2020/2021. The table was taken as it is from [62].

Impairment type	Children	Working age	State Pension age	**All ages**
Mobility	15%	42%	63%	**46%**
Stamina/breathing/fatigue	19%	32%	38%	**33%**
Mental health	21%	42%	9%	**29%**
Dexterity	9%	22%	29%	**23%**
Memory	6%	12%	11%	**11%**
Learning	20%	12%	5%	**11%**
Hearing*	5%	7%	17%	**10%**
Vision	5%	7%	13%	**9%**
Social/behavioural	37%	8%	1%	**8%**
Other	21%	23%	23%	**23%**

Elderly people are not the only ones in need for caregivers and assistive products. Another important user group consist of people with motor impairments. There are several diseases, such as degenerative diseases like amyotrophic lateral sclerosis, that can create this need. In addition, tetraplegia, paraplegia, or other types of paralysis caused by injuries in the spinal cord, can make people be dependent on caregivers and/or assistive products to perform activities of daily living such as eating, going to the toilet, dressing, changing location, etc. According to a study done by the WHO in 2013 [1], every year between 250 000 and 500 000 people suffer from a spinal cord injury. In the UK alone, for example, there is an estimate of 14.6 million people (22% of the population) with a disability in 2020/2021, where mobility is the most frequent reported impairment type with 46% (see Table 1.1).

Assistive products are often used to assist all these users in performing activities of daily living, which grant them more autonomy and also reduce partially the constant dependency on caregivers. It is clear that there is a substantial – and an increasing – number of people in need for assistive products and, unfortunately, only a small percentage are active users of them. This is associated with two issues of current assistive technology [49]: *(i)* people do not have access to the technology they need, *(ii)* and the technologies are frequently abandoned.

> *Assistive products enable people to live healthy, productive, independent and dignified lives; to participate in education, the labor market and civic life. Assistive products can also help to reduce the need for formal health and support services, long-term care and the work of caregivers. Without assistive products, people may suffer exclusion, are at risk of isolation and poverty, and may become a burden to their family and on society.*
>
> World Health Organization [2], 2016

According to the WHO [2], only around 10% of people in need have access to these assistive products. This is, among other reasons, due to limited availability, high costs and lack of suitably trained personnel. In addition, the WHO [2] says that the assistive product industry is currently limited and extremely specialized, primarily serving the requirements of high-income settings. This is caused, among others, by **lack of user-centered research**.

The work of [49] also argues that there is a lack of user-centered research and development, and this induces a **mismatch between user needs and provision**. According to the authors, this is one of the main reasons of technology abandonment of many assistive devices. Another study [74] shows that 29.3% of all assistive devices were completely abandoned, while mobility aiding devices were abandoned more frequently than other types. The study found four factors that were inducing this abandonment, in decreasing order of association strength: *(i)* change in user needs or priorities, *(ii)* easy device procurement, *(iii)* poor device performance, *(iv)* and lack of consideration of user opinion in selection.

In our view, these factors can be addressed in the following ways:

(i) **change in user needs or priorities:** This factor occurs because of lack of acknowledgement that user needs and preferences change over time. Thus, immutable products are sold without much possibility of adapting alongside the user. This can be mitigated in different ways, depending on the complexity of the required adaptation. Firstly, products should be configurable by the user (or by a caregiver) at will and in an intuitive way, such that no complex training is required (similarly as one can change the preferred settings of a TV or a computer). And secondly, products should offer reconfigurability, done by the seller or by trained (but still not highly educated) personnel,

such that they can be modified or upgraded to fit the new needs or preferences.

(ii) **easy device procurement:** Surprisingly, the study done in [74] shows that, the easier the product is to obtain from any source, the more likely it is to be abandoned. Devices purchased directly from suppliers such as durable medical equipment dealers, local drug stores, mail-order catalogs, etc, may be easy to obtain but turn out to be less useful if the consumer has inadequate product information. Also, these products are usually mass-produced with homogeneous features for people with certain diagnoses. Thus, the correlation between easy device procurement and inadequate product information alongside non-personalizable equipment should be broken by changing the paradigm towards more user-centered design and commercialization.

(iii) **poor device performance:** According to the study, this factor related to how well the device satisfies the user's expectations for effectiveness, reliability, comfort, ease of use, safety and durability. The more context-specific the assistive device is, the more performance is expected from the device. However, it becomes challenging when assistive devices are supposed to be employable for different situations, (unstructured) environments, users, tasks, etc. The assistive devices of the future should provide good performance by adjusting to variations in these all aspects.

(iv) **lack of consideration of user opinion in selection:** This factor occurs because of lack of acknowledgement of the importance of user preferences and needs when choosing a suitable assistive product. Often the decision is limited or even taken by third parties (e.g. health insurance, family member, etc) rather than by the user, often with strict limitations on budget and product portfolio. However, in practice, it would be very difficult to change the decision-making stakeholders. A way to mitigate this is to make the offered products more easily personalizable. This personalization should be done easily by a commercial service provider stakeholder (e.g. the seller) and, to make it economically feasible, it should not require intervention of highly educated (but still trained) personnel. Evaluation on how the personalization possibilities can cope with the user needs and preferences should be an integral part of the selection process, and it is of course only feasible if the personalization possibilities are present and adequate.

Supported by these ideas, and following our aforementioned view, this thesis proposes a robotic assistive framework composed of a collection of technological tools that, if used properly by the different stakeholders, can contribute to bridging the gap between provision and user needs and preferences. This thesis does not provide user-centered assistive solutions (i.e. developed with end users in the loop), but instead facilitates their development and deployment so that in the future needs and preferences of end users can play a fundamental role.

1.2 Research Objectives

The general objective of this thesis is to develop a control framework that facilitates the commissioning of user-centered robotic assistive systems. These robotic systems should be able to provide physical assistance to users with different needs (e.g. different levels of mobility) and preferences, during activities of daily living while providing an intuitive and safe user-robot interaction. Such assistance should be suited for non-structured and uncertain environments, which are inherent characteristics of places frequented by users of assistive technologies.

The framework should:

• **Specific objective 1** – *allow user personalization,* by enabling re-configurability of the robotic systems to cope with specific user needs and preferences;

• **Specific objective 2** – *adapt on-line to different situations*, by adjusting the provided assistance to handle different situations; and

• **Specific objective 3** – *be validated,* by demonstrating its usability through a series of experiments that mimic heterogeneous realistic assistive scenarios.

1.3 Overview of the thesis

This section gives an overview of the most important features and concepts of the thesis. The content on this section is only introductory, and the details are available in the remaining chapters. We start by discussing the importance of user-centered assistance and which stakeholders play

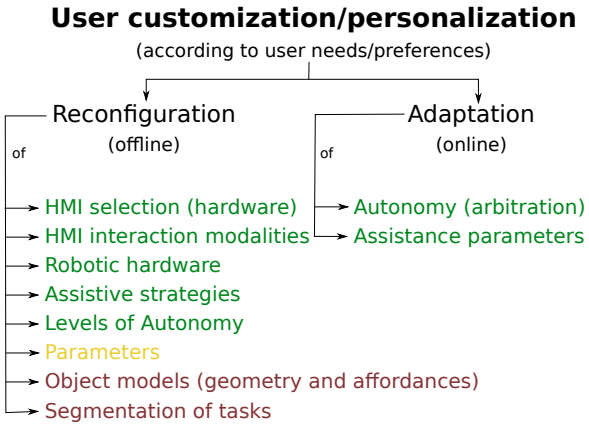


Figure 1.2. Aspects of the framework through which customization and personalization of assistive devices can be achieved. In this thesis, we focus on the green terms and leave out of scope the red ones (which are also important). The “parameters" term is in yellow since, of course, we do select parameters within this thesis but, since it is very general (there are parameters everywhere), formalization related to which parameters should be used for customization/personalization is still required. This formalization is left out of scope of this thesis.

an important role in our view. Then we explain on which type of robotic devices we focus and which type of methods we use for providing the desired assistance. Next we give an overview of the use cases that are used to validate the methods and concepts proposed along the thesis. Finally, we give an overview of the HMIs and sensors that are used for the use cases.

1.3.1 User-centered assistance and its stakeholders

As explained in section 1.2, the main goal of this work is to develop a framework that facilitates the development and deployment of user-centered robotic assistive devices. The main feature that characterizes these user-centered devices is that they offer the possibility of customizing/personalizing some of their aspects to cope with user needs and preferences (e.g. as done in [23]). These needs and preferences can change over time, and it is important to allow easy reconfiguration so that effective assistance can still be provided and the device is not abandoned. In this thesis we do this customization/personalization in a hierarchical way (see fig. 1.2), and we limit the scope to some aspects (bottom layer of fig. 1.2) that we consider important.

As a remark, the aspects mentioned in fig. 1.2 are the ones that we consider the most important to be used as mechanisms for customization and personalization. However, also additional aspects need to be defined when developing and deploying an assistive robotic device.

Due to the complexity of these devices, there are several stakeholders that, based on their expertise, play key roles in the development and the integration of all these aspects. In our view, there are five stakeholders[1] that play the most important roles in the development and deployment of these user-centered assistive devices. These five stakeholders are listed as follows in descending order of required training and expertise:

1. The **system developer(s)** provide(s) flexible software components that can be used to develop several assistive devices that serve several applications and users. Some of these flexible components, referred to as HMI drivers and robot drivers, serve as a bridge to integrate the HMIs and robotic platforms, respectively, within the chosen middleware[2] that interconnects other components that run in the system. Other flexible components that the system developer provides are the assistive building blocks that, when combined properly, create assistive strategies. The formal definition of these building blocks and several examples are given in chapter 3.

2. The **application developer(s)** provide(s) three different libraries that the next stakeholder can choose from to create user-centered integrated solutions. The first library consists of general assistive strategies (whose exact definition is given in chapter 3) that can be reconfigured and adapted to assist multiple users during activities of daily living. Also, a library of different interaction modalities (i.e. how HMI inputs and outputs are interpreted and mapped to/from the assistive robotic device) are provided by this stakeholder. Finally, specific assistive strategies are only compatible with a subset of control modes (e.g. cartesian velocity control, joint position control, joint torque control, etc) that the different robotic devices offer. Thus, this stakeholder provides a third library with the different control modes that are compatible with the developed assistive strategies.

[1] We use a minimalistic approach to define the distribution of the load between the different stakeholders. In practice there can be more stakeholders. Due to the complexity and the involved social and health interactions, there will be many other types of stakeholder. These are not discussed within the scope of this thesis.

[2] The discussion about robotic software architecture is out of the scope of this thesis.

3. The **solution integrator** selects within the libraries provided by the application developer and integrates a user-centered solution that takes into account specific user needs and preferences. This stakeholder first selects which robot and HMIs (hardware) are best suited for the user. Based on the selected robot and HMIs (hardware) suitable for the specific user, specific control modes and interaction modalities (software), respectively, are integrated. Finally, this stakeholder also selects which levels of autonomy, interaction modalities and assistive strategies (a subset of all options) are suitable, and defines how to transition between the different alternatives (e.g. with a touchscreen, voice commands, etc). The solution integrator selects the value of some parameters taking into account the needs and preferences of the user. Some other parameters are exposed to the caregiver/user so that they can do further personalizations/customizations. These parameters should have a formal description such that a graphical user interface can be designed for caregivers and users. This is not further discussed in this thesis.

4. The **caregiver(s)** can assist the user in making some personalization, depending on the mobility limitations of the user.

5. The **user** is the most important stakeholder, since the decisions taken by all stakeholders should take the user needs and preferences into account, at different levels of abstraction.

The user's opinion should be taken into account not only when acquiring the system, but also constantly to cope to changing needs and preferences. Fig. 1.3 shows a way of handling continuous development and improvement, where each stakeholder facilitates the work of other stakeholder closer to the user while also receiving feedback and new requirements from that latter stakeholder. For instance, the user transmits new requirements to the solution integrator who should be able to make changes to cope with the new requirements. If this is not possible with the provided components, the solution integrator needs to transmit the new requirements to the application developer(s) who should make modifications or provide new functionalities. If the application developer(s) cannot find a solution with the available components, the application developer(s) need to transmit the new requirements to the system developer(s) who should make modifications or provide new low-level functionalities. In this way the user is kept inside the development loop at the different levels assigned to the different stakeholders.

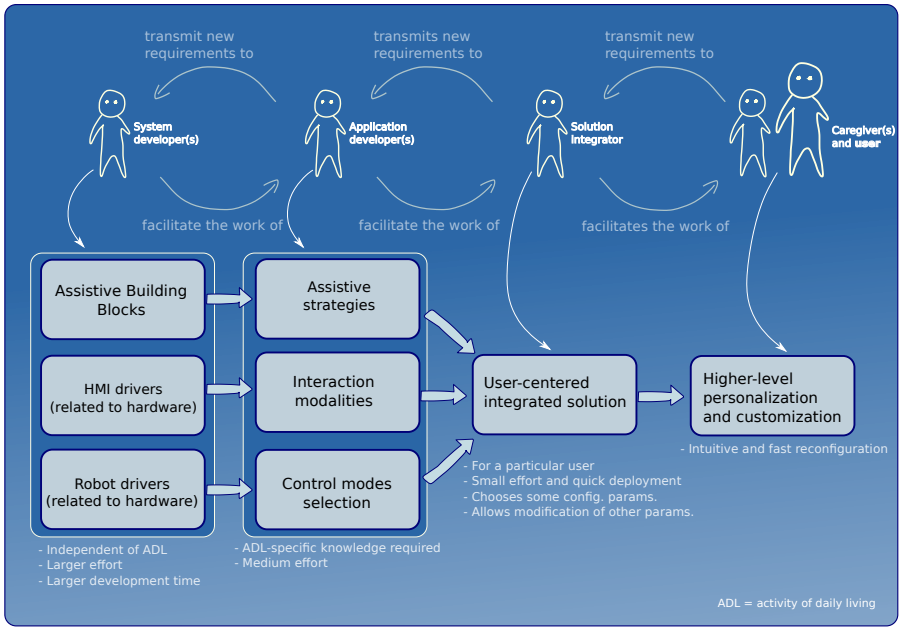


Figure 1.3. Scheme showing a proposed design workflow describing the roles of the main stakeholders involved in the software development and deployment of a user-centered assistive robotic device. The scheme and the role of some stakeholders were inspired by the work found in [99].

1.3.2 Considered assistive robotic devices

In the world of assistive robots there are many types of robots who have as purpose to assist humans with different aspects of their lives. For example, there are socially assistive robots that seek to provide assistance by communicating through speech, facial expressions, and other hands-off interaction strategies. There are also caregiver robots that seek to complement the job that human caregivers do in certain situations in which they feel burdened. Another class of assistive robots are rehabilitation robots, which seek to rehabilitate the human body or to replace or reinforce a lost or weakened function that causes mobility limitations.

Rehabilitation robots can be further be classified according to their usage. Fig. 1.4 shows a possible classification that contains several examples of robotic devices. The scope of this thesis is limited to the robotic devices highlighted in the figure and mainly focuses on robotic arms that are fixed to the world (e.g. on a table). However, the applicability of our methods to a

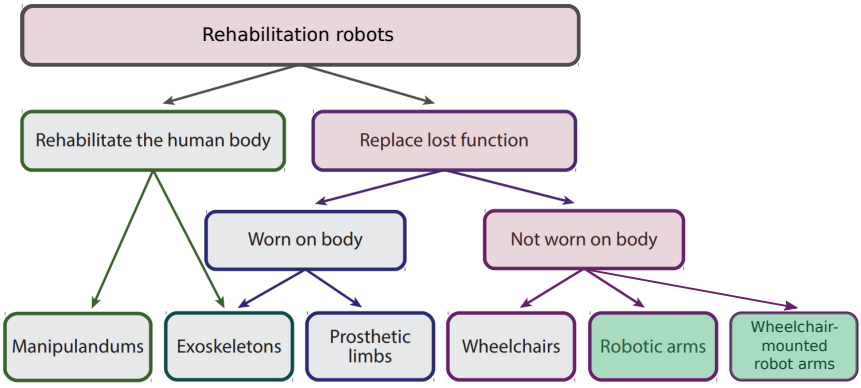


Figure 1.4. Classification of rehabilitation robots. The scope of this thesis is limited to the robots highlighted in green and the types (i.e. parent nodes) highlighted in light pink. Figure taken and adapted from [13].

wheelchair-mounted robot arm is also shown.

As explained in Fig. 1.2, one of the aspects of configuration that this thesis tackles is an easy way of changing towards other robotic devices. This is demonstrated in this thesis by deploying different assistive use cases on five different robotic setups shown in Fig. 1.5. The proposed assistive control methods are decoupled into a robot-agnostic part and a robot-dependent part, which makes the transition between robots easy (as explained later on in subsection 1.3.3). A short description of each robotic setup is given as follows:

- **Kinova Gen 3:** a commercial 7-DOF robotic arm of brand Kinova (sometimes just referred to as the Kinova robot).
- **Franka Panda:** a commercial 7-DOF robotic arm of brand Franka Emika (sometimes just referred to as the Franka robot).
- **UR10:** a commercial 6-DOF robotic arm of brand Universal Robots.
- **EDAN:** a non-commercial system that consist of an 8-DOF robotic arm mounted on an active wheelchair [103]. The use cases presented on this system were done in collaboration with the Institute of Robotics and Mechatronics from the German Aerospace Center (DLR) at their facilities.

- **KUKA iiwa:** a commercial 7-DOF robotic arm of brand KUKA (sometimes just referred to as the iiwa robot)

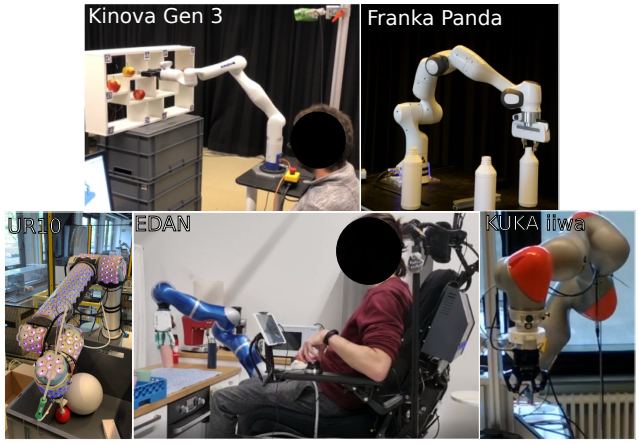


Figure 1.5. Robotic devices used in the use cases presented in this thesis.

1.3.3 Reactive constraint-based shared autonomy

In this subsection we give a high-level introduction about *reactive constraint-based shared autonomy*, which is a core concept used in this thesis. This concept can be decomposed into three terms that independently exist in literature: **reactive**, **constraint-based** and **shared autonomy**. Each of these three terms will be explained in the remaining part of this subsection.

In literature there exist many approaches in which autonomous robots can be programmed in order to execute tasks. A common paradigm which many approaches follow is the so-called Sense-Plan-Act, in which the robot *1)* gathers information about the environment using sensors, *2)* elaborates a plan on how to perform the task, and *3)* finally executes the plan to achieve the task.

In contrast, a more robust way that can deal with changes in the environment, or other uncertainties related to the task, is to instead rely on **reactive** approaches. These approaches continuously use sensor information to react to changes and continuously adapt the defined actions based on control laws.

There are several approaches that offer such reactiveness during task executions. A possibility is to specify nominal actions (e.g. trajectories) in combination with descriptions on how to deviate from these nominal actions based on sensor information. The resulting behaviors due to these combinations are referred to in this thesis as *reactive robot behaviors*.

A quite convenient way of defining such reactive robot behaviors is to model them as a set of constraints. For example, if a robot needs to insert a peg into a hole: *(i)* the peg should maintain alignment with the hole (orientation and position constraints which are part of the nominal plan), *(ii)* the robot should maintain a certain velocity in the direction of insertion (velocity constraint which is part of the nominal plan), *(iii)* the robot should act compliantly to the contact between the peg and the hole since there are uncertainties associated to the location or to manufacturing tolerances of both workpieces (force constraints which are part of the deviations from the nominal plan). Of course this modeling is not unique and the selection of such constraints will influence the behavior and performance of the task execution, with pros and cons under certain situations.

Methods that model a task by using a set of constraints are referred to as **constraint-based** methods. There are different constraint-based methods (some with available software tools) that ease the specification of such constraints and thus of the task (e.g. [27], [88], [69]). However, given the scope of this chapter, only two constraint-based methods are explained here, without going into mathematical details: the *task frame formalism* [21] and *eTaSL* [8]. Later on we make use of both constraint-based methods for the formulation of assistive strategies based on reactive constraint-based shared autonomy.

The first method, the task frame formalism, allows the specification of force-controlled actions to cope with geometric uncertainties that are too large to handle with only passive (i.e. mechanical) compliance. The formalism allows the specification of either force or velocity constraints for each of the axes of a reference frame (both linear and angular), independently. This reference frame will be referred to as the *task frame*, and hence it is selected according to the task at hand. After these actions are defined, a low-level controller is needed to execute the specified forces and velocities.

Fig. 1.6 shows two alternative examples of the specification for a door opening task using the task frame formalism [21]. The specification shown in the left of the figure commands a linear velocity along the Z axis of a task frame defined in the handle of the door, while maintaining zero force

or moment (i.e. being compliant) in the other five components (linear and rotational directions along the axes of the task frame). This specification has the advantage that it does not require geometric knowledge of the door (i.e. the hinge is not known), since it relies on active compliance for the end effector to re-orient and follow the circular path related to the door. An alternative task specification is shown in the right of the figure, which uses geometric information of the door in order to specify controlled actions in a task frame located and aligned along the hinge (i.e. angular velocity in Z axis and zero force or moment in the other five directions).

Both specifications deal with geometrical uncertainties: the left with big uncertainties and the right with small uncertainties (i.e. location or alignment errors of the hinge). Since the closed-loop controller needs to deal with these uncertainties using force measurements, the task can be performed faster with the right task specification (hence there is an increase in performance). However, this increase of performance comes with the cost of needing extra geometrical information inherent to a specific door.

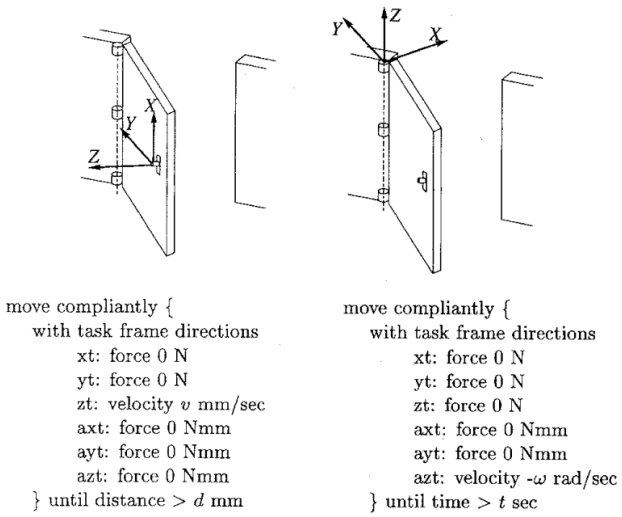


Figure 1.6. Task specifications of a door opening task using force and velocity controlled actions, expressed in two alternative task frames, following the task frame formalism. The figure is reprinted from [21].

The second aforementioned constraint-based approach used in this thesis is eTaSL. This approach offers more flexibility and possibilities for specifying constraints related to the task, to the robot and to the environment. The

most important elements that are used to create a task specification in eTaSL are twofold: expressions and constraints. Firstly, expressions need to be defined, which depend on robot-related variables (i.e. joint positions) and/or variables describing features of the task (referred to as feature variables) and/or time. Secondly, some constraints over these expressions need to be declared.

There are two types of constraints that can be specified for each of the expressions:

- equality or inequality constraints imposed to the expression itself (i.e. position level).
- equality or inequality constraints imposed to the time derivative of the expression (i.e. velocity level).

For example, a commonly used expression is the position of the end effector of the robot (which depends on the joint positions), and different constraints can be imposed to it. One can impose, for instance, the following constraints:

- a position equality constraint, such that it follows a desired trajectory.
- a velocity equality constraint, such that it follows a desired velocity coming from an input device such as a joystick.
- a velocity equality constraint, such that the end effector moves proportionally to measured end effector forces (i.e. admittance constraint) so that it remains compliant.
- position inequality constraints to remain always within a safety region.

All of the above constraints can be declared in the same task specification if desired. Since it is possible that not all of them can be fulfilled at the same time, eTaSL allows the definition of two priority levels. Constraints with the highest priority are referred to as *hard constraints*, since they need to be fulfilled or otherwise the task is considered as unfeasible. Constraints with the lowest priority are referred to as *soft constraints*, and have always an associated weight that indicates their relative importance to each other. For example, if there are (only) two position constraints with the same weight, the resulting behavior is that the expression (e.g. the end effector) will be controlled towards the middle. It is important to mention that this weight

does not influence stability, and it only influences the behavior when there are other conflicting constraints.

Open source code for robot task specification and control using eTaSL is available on [7]. The following non-exhaustive list of features of eTaSL offer composability and, in the case of this thesis, they facilitate the desired reconfigurability of the assistive devices:

- a URDF parser can translate kinematic models of robots into eTaSL expressions and constraints, such as joint position and velocity limits. Since most of the companies (or third party developers) nowadays provide URDF models for most of the robots, this highly facilitates the deployment of tasks in different robotic devices.
- the robot specification, containing kinematic expressions and joint limits, can be decoupled from the task specification, giving the flexibility to reuse the same task specifications for controlling different robotic devices. If decoupled, the task specification is said to be *robot-agnostic*. It is even possible to deploy an application using only a task specification in cartesian space (i.e pose of the end effector) and avoid the need of a kinematic model of the robot, with the disadvantage that constraints related to joint limits will not be taken into account.
- constraints in eTaSL can involve the use of multiple sensors, which facilitate the specification of reactive robot behaviors that interact with the environment and the human in different ways.
- expression graphs are automatically generated from all the defined eTaSL expressions and used to compute the required Jacobian matrices by usign algorithmic (automatic) differentiation. This is not only done for the expressions representing the kinematic model of the robot (obtained from the URDF), but also for all the expressions defined in the task specification.

eTaSL offers more flexibility than the task frame formalism, since it allows constraints of different nature (e.g. position constraints and constraints involving other sensors) and also allows conflicting constraints. In fact, a task specification that follows the task frame formalism can also be specified within eTaSL for specifying force-controlled actions, while it can be combined with additional constraints (as we do in this thesis).

Different works have adopted these two constraint-based paradigms to build frameworks that can facilitate even further the programming of composable

autonomous robotic reactive behaviors for different applications (e.g. [99] and [79]). In this work, we also adopt these two paradigms in order to inherit all the aforementioned features and exploit them to provide reconfigurable and adaptable assistive strategies in the context of assistive robotics.

We provide these assistive strategies by means of **shared autonomy** [91] (also known in literature as shared control). In shared autonomy, the input from the user (e.g. given through an HMI like a joystick) is not used directly to define the (continuous) commands that the robot should execute. Instead, the robotic system combines partial knowledge of the task with the commands of the user to define control actions that are executed in order to fulfill a certain task. This is done reduce the effort of the user while performing the task, while still maintaining autonomy over the task. In other words, the control (or the autonomy) over the robot is shared between the user and the robot itself.

With the framework proposed in this thesis, we leverage the two aforementioned constraint-based approaches to enable multiple reconfiguration and adaptation mechanisms for assistive robotic devices, as shown in Fig. 1.2. Such framework enables the system developer (see Fig. 1.3) to rapidly create composable assistive building blocks (i.e. sets of constraints) that define certain aspects of the assistance. Later on, the application developer selects a combination of those building blocks to specify assistive strategies. These assistive building blocks are defined and further explained in chapter 3.

Autonomy over different aspects of a task, which are associated to these assistive building blocks, can be allocated (i.e. granted) to the user and/or to the robot. The allocated autonomy can be modulated online, such that it can gradually shift between user and robot. In this thesis, the allocated autonomy and the ability to modulate it is referred to as a *level of autonomy*. There can be multiple levels of autonomy in parallel that relate to different aspects of the task. These levels of autonomy can be modulated automatically or by user commands.

Levels of automation is a different term used in literature (e.g. [39]) and it usually refers to an automation taxonomy. Several taxonomies have been proposed in literature (e.g. [39], [108]), which are composed of a finite list of levels that describe how much autonomy is granted to the robotic system (from zero automation to full automation). Nevertheless, in this thesis we do not use the term levels of automation since these taxonomies usually contemplate switching discretely (i.e. not continuously) from one level of automation to another. Instead, we refer to adaptable levels of autonomy,

which are further explained in section 3.5 and illustrated in the rest of the thesis.

It is worth to mention that there is an inconsistency in the terminology used in literature since the terms *levels of autonomy* and *levels of automation* are used interchangeably. For example, in [108], the authors describe six levels of automation for robotic surgery, but the term levels of autonomy is used instead. In contrast, in [39], a taxonomy of 12 levels of automation is given and terminology is consistent with the one used in this thesis. To avoid confusion, we adhere to the definitions given above.

1.3.4 Considered use cases

In this thesis several assistive strategies for different activities of daily living are demonstrated. Some of them were reconfigured to fit potential needs of some users with different mobility limitations. This section gives an overview of the use cases that the reader will encounter throughout the different chapters of the thesis. Such variety of use cases shows the reconfigurability and adaptability that characterizes the methods proposed in this thesis. Each of the use cases exposes different technological challenges that were tackled.

Involving users with real motor impairments in research is a big challenge and requires a high technological readiness level because of practical and ethical concerns. Therefore, we showcased these use cases by using able-bodied users that tried the system in conditions that mimicked the impairments. The impairments were mimicked by using HMIs that are compatible with the mobility limitations (e.g. eyetracker or BMI for a tetraplegic mockup user) and performing the task accordingly (e.g. by not moving the upper and lower limbs during the task execution).

In the remaining part this subsection, explanation about each of the use cases will be given. At the end of the subsection, the reader can find a Table 1.2 that compares and summarizes the use cases.

Reaching Tasks:

Most of the activities of daily living (such as eating, opening a drawer, taking the tv remote control, etc) have a phase that is approaching to an object of interest (e.g. before grasping it) and/or taking an object to a certain place. This phase is referred to in this thesis as a *reaching task*,

since it involves reaching a pose or, more specifically, a manifold of poses (given some tolerances) with the end effector. We present several use cases focusing on reaching tasks, given their importance and recurrence. Several assistive strategies for these tasks will be proposed and reconfigured in order to provide specific assistance in conditions related to the environment (e.g. the objects are in a bin or in a grid-like ordered positioning) and to potential mobility limitations of users.

In **use case 1** (UC1) we demonstrate how the combination of allowable manifolds and an artificial skin can be used to deploy a semi-autonomous bin-picking application. The artificial skin is composed of multiple cells that measure distance to objects (in this case, to the user hand). Each cell has an LED light that changes color when it detects an object in the vicinity (for explainability purposes). This use case can work autonomously if the human decides not to intervene. However, if desired, the human can interact and affect the trajectory of the robot (by means of the artificial skin). For example, the human can pause the robot motion by blocking its progress, or can change the final pose of the robot tool which is constrained to an allowable manifold on the object (see Fig. 1.7). A short description of this use case is presented as follows:

- *environment description*: spherical objects placed inside a bin (i.e. bin-picking).
- *targeted user*: able-bodied.
- *testing user*: able-bodied researcher.
- *robotic device*: UR10 (see Fig. 1.5).
- *HMIs*: artificial skin measuring distance from the hand to the cells.
- *autonomy*: fixed shared autonomy (little autonomy for user).

A more detailed description is given in chapter 2.

In **use case 2** (UC2) we demonstrate an industry-oriented picking application, but this time the system is completely autonomous and, thus, the user cannot intervene with the robot. The robot uses reactive approach motions that are constrained to be on an allowable manifold (as the previous use case), but this time it is used to avoid dynamic obstacles instead of interacting with the human. This use case shows that more autonomy can be given to the system with our methods, until the point of complete autonomy. A short description of this use case is presented as follows:

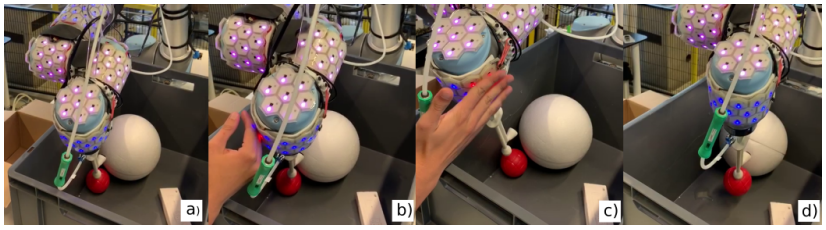


Figure 1.7. Snapshots of the human interacting with the robot through an artificial skin. The robot is constrained to be on an allowable manifold.

- *Environment description*: cylindrical objects placed on top of a table.
- *Targeted user*: none.
- *Testing user*: none.
- *Robotic device*: Kinova Gen 3 (see Fig. 1.5).
- *HMIs*: None.
- *Autonomy*: Autonomous system (no human).

A more detailed description is given in chapter 2. Although in this chapter we only describe it in simulation in order to visually explain how the method works, we did execute it on a real robot.

In **use case 3** (UC3) we demonstrate an application of assistive human-robot interaction (see Fig. 1.8.a). The human can *(1)* specify desired target objects by fingertip pointing, *(2)* change the level of autonomy with voice commands (e.g. make the system advance autonomously), and *(3)* interact with the robot through force (e.g. hand guiding it or pausing its trajectory). The system provides assistance to the user to reach the targets given some constraints, when in hand-guiding mode. The user can also give more autonomy to the robot with a voice command such that it starts an autonomous motion that automatically avoids dynamic obstacles. A short description of this use case is presented as follows:

- *Environment description*: objects are placed on a table and dynamic obstacles are present (and detected with vision).
- *Targeted user*: able-bodied user.
- *Testing user*: able-bodied researcher.
- *Robotic device*: Franka Panda (see Fig. 1.5).

- *HMIs*: Speaker/microphone system, 3D camera and torque sensors.
- *Autonomy*: shared autonomy with adaptable levels of autonomy.

A more detailed description is given in chapter 3.

In **use case 4** (UC4) we demonstrate an application of assistive teleoperation for grasping objects (see Fig. 1.8.b). The user teleoperates the robot using a 6 DOF joystick (SpaceMouse), which allows to control linear and angular velocities of the end effector. The system provides assistance to the user to reach the targets given some constraints. A short description of this use case is presented as follows:

- *Environment description*: objects are placed on a table.
- *Targeted user*: paraplegic user.
- *Testing user*: able-bodied researcher.
- *Robotic device*: Franka Panda (see Fig. 1.5).
- *HMIs*: SpaceMouse.
- *Autonomy*: shared autonomy with adaptable levels of autonomy.

A more detailed description is given in chapter 3.

In **use case 5** (UC5) we demonstrate a feeding application in which users with a low level of mobility can interact with the robot through an eye tracker (see Fig. 1.8.c) The gaze of the user is used to estimate objects of interest (e.g. apples) and used to update the end position of a reactive trajectory online. The user is also able to specify blink gestures (e.g. to start or pause the motion, grasp the object, bring the object to the mouth, etc). A short description of this use case is presented as follows:

- *Environment description*: apples are placed in a shelf and are detected with vision.
- *Targeted user*: tetraplegic user.
- *Testing user*: able-bodied researcher.
- *Robotic device*: Kinova Gen 3 (see Fig. 1.5).
- *HMIs*: Eye tracker and a screen monitor.
- *Autonomy*: shared autonomy with adaptable levels of autonomy.



Figure 1.8. Use cases reconfigured to assist specific mobility limitations during reaching tasks. (a) use case 3, (b) use case 4, (c) use case 5.

A more detailed description is given in chapter 3.

In **use case 6** (UC6) we demonstrate an application targeted for users with low mobility (e.g. tetraplegic) in which neural activity (taken through an invasive BMI) is used to decode the desired target object. Once the robot knows the desired target object, it reaches the object autonomously. Given the medical and ethical implications, and the technological complexity of an invasive BMI, the trials were made by trained non-human primate users in controlled lab conditions. This was done in the context of the KU Leuven C1 project "Neurophysiological investigations of the human, and nonhuman primate brain: from perception to action" (3M180301) with the approval of the KU Leuven ethical committee. A short description of this use case is presented as follows:

- *Environment description*: known and fixed objects.
- *Targeted user*: tetraplegic user.
- *Testing user*: trained non-human primate (i.e. monkey).
- *Robotic device*: Kinova Gen 3 (see Fig. 1.5).
- *HMIs*: BMI decoding target object.
- *Autonomy*: the user provides the target object and the task is executed autonomously.

A more detailed description is given in chapter 4.

In **use case 7** (UC7) we demonstrate an application also targeted to users with low mobility in which neural activity (taken through an invasive BMI) is used to continuously decode desired linear velocities for the end effector of a virtual arm (i.e. in simulation). The system provides assistance to the

user to reach the targets given some constraints, such that the control of the virtual arm (shown on a monitor screen) is facilitated. This was done in the context of the KU Leuven C1 project "Neurophysiological investigations of the human, and nonhuman primate brain: from perception to action" (3M180301) with the approval of the KU Leuven ethical committee. A short description of this use case is presented as follows:

- *Environment description*: known and fixed virtual objects.
- *Targeted user*: tetraplegic user.
- *Testing user*: trained non-human primate (i.e. monkey).
- *Robotic device*: Virtual Kinova Gen 3.
- *HMIs*: BMI decoding linear velocity.
- *Autonomy*: shared autonomy with fixed levels of autonomy.

A more detailed description is given in chapter 4.

In **use case 8** (UC8) we demonstrate an application targeted to users that require the use of a wheelchair but that still have enough mobility in their hands to use a joystick. We use an assistive strategy that attracts the end effector towards objects to be grasped that are close to each other (e.g. in a grid-like spacing). The user controls the robot arm with the joystick, and can perform mode switching in order to control the wheelchair with the same joystick. A short description of this use case is presented as follows:

- *Environment description*: known objects placed in a crate with grid-spaced compartments.
- *Targeted user*: paraplegic user.
- *Testing user*: able-bodied researcher.
- *Robotic device*: EDAN (see Fig. 1.5).
- *HMIs*: SpaceMouse (with mode switching).
- *Autonomy*: shared autonomy with adaptable levels of autonomy.

A more detailed description is given in chapter 4.

In **use case 9** (UC9) we demonstrate a variant of UC8. The application is quite similar, but here we add the posibility of a caregiver to physically interact with the robot (kinesthetically) to provide some help if needed. A short description of this use case is presented as follows:

- *Environment description*: known objects placed in a crate with grid-spaced compartments.
- *Targeted user*: paraplegic user.
- *Testing user*: able-bodied researcher.
- *Robotic device*: EDAN (see Fig. 1.5).
- *HMIs*: SpaceMouse (with mode switching) and torque sensors.
- *Autonomy*: shared autonomy with adaptable levels of autonomy.

A more detailed description is given in chapter 4.

In **use case 10** (UC10) we demonstrate another variant of UC8. The application is quite similar, but here instead of performing mode switching to control the wheelchair and the robot arm separately, we use whole-body control in order to control both at the same time with the same HMI. The SpaceMouse normally controls the robot arm, but when the arm goes outside a region (e.g. it extends or retracts too much) the wheelchair moves accordingly with the same SpaceMouse. This avoids the aforementioned mode switching and facilitates the control of the robotic device in certain scenarios.

A short description of this use case is presented as follows:

- *Environment description*: known objects placed in a crate with grid-spaced compartments.
- *Targeted user*: paraplegic user.
- *Testing user*: able-bodied researcher.
- *Robotic device*: EDAN (see Fig. 1.5).
- *HMIs*: SpaceMouse (with whole body control).
- *Autonomy*: shared autonomy with adaptable levels of autonomy.

A more detailed description is given in chapter 4.

Interacting with Articulated Objects:

Several activities of daily living involve the interaction with articulated objects (such as opening doors, opening drawers, opening laptops, etc). Sometimes this can be done by just pushing the object approximately, but for some tasks a stable grip is required (e.g. opening a fridge or a drawer).

Performing this by teleoperation is quite challenging, since the user needs to provide precise motion subject to the kinematic constraints of the articulated object. Thus, this thesis proposes assistive strategies that facilitate the interaction with such articulated objects.

In **use case 11** (UC11) we demonstrate how we can use such assistance for interacting with articulated objects whose kinematic properties are unknown. A **user study with able-bodied participants** was performed to evaluate and demonstrate the improvement of the proposed assistive method in terms of a series of qualitative and quantitative metrics.

A short description of this use case is presented as follows:

- *Environment description*: unknown articulated objects are present in an unstructured environment.
- *Targeted user*: paraplegic user.
- *Testing user*: non-expert able-bodied users.
- *Robotic device*: KUKA iiwa (see Fig. 1.5).
- *HMIs*: SpaceMouse.
- *Autonomy*: shared autonomy with adaptable levels of autonomy.

A more detailed description is given in chapter 5.

1.3.5 Considered HMI and sensors

This subsection contains a brief overview of the HMIs and sensors used in this thesis. We make a distinction between HMIs and sensors according to the role that each fulfill within an assistive application. HMIs serve to communicate information from the human to the robot or from the robot to the human, while sensors communicate information about the environment to the robot (and not the other way around). The HMIs that communicate information from the human to the robot contain sensors that, instead of measuring data related to the environment, they are expected to measure data related to the human. With this distinction in mind, Table 1.3 and Table 1.4 show the different HMIs and sensors, respectively, used for the use cases of this thesis.

For the HMIs, Table 1.3 shows four different purposes for which the HMIs are used. Use cases under *TCP velocity* have HMIs that influence the

Table 1.2. Comparison between the different use cases. The information about each use case is redundant, since it has already been presented, but it is now re-organized in a table to facilitate the comparison. For the robotic device, see Fig. 1.5.

Use Case	**Environment description**	**Targeted user**	**Testing user**	**Robotic device**	**HMIs**	**Autonomy**
UC1	spherical objects placed inside a bin (i.e. bin-picking)	able-bodied	able-bodied researcher	UR10	artificial Skin measuring distance from the hand to the cells	fixed shared autonomy (little autonomy for user)
UC2	cylindrical objects placed on top of a table	none	none	Kinova Gen 3	none	autonomous system (no human)
UC3	objects are placed on a table and dynamic obstacles are present (and detected with vision)	able-bodied user	able-bodied researcher	Franka Panda	Speaker/microphone system, 3D camera and torque sensors	shared autonomy with adaptable levels of autonomy
UC4	objects are placed on a table	paraplegic user	able-bodied researcher	Franka Panda	SpaceMouse	shared autonomy with adaptable levels of autonomy
UC5	apples are placed in a shelf and are detected with vision	tetraplegic user	able-bodied researcher	Kinova Gen 3	Eye tracker and a screen monitor	shared autonomy with adaptable levels of autonomy
UC6	known and fixed objects	tetraplegic user	trained non-human primate (i.e. monkey)	Kinova Gen 3	BMI (decoding target object)	the user provides the target object and the task is executed autonomously
UC7	known and fixed virtual objects	tetraplegic user	trained non-human primate (i.e. monkey)	Virtual Kinova Gen 3	BMI (decoding linear velocity)	shared autonomy with fixed levels of autonomy
UC8	known objects placed in a crate with grid-spaced compartments	paraplegic user	able-bodied researcher	EDAN	SpaceMouse (with mode switching)	shared autonomy with adaptable levels of autonomy
UC9	known objects placed in a crate with grid-spaced compartments	paraplegic user	able-bodied researcher	EDAN	SpaceMouse (with mode switching) and torque sensors	shared autonomy with adaptable levels of autonomy
UC10	known objects placed in a crate with grid-spaced compartments	paraplegic user	able-bodied researcher	EDAN	SpaceMouse (with whole body control)	shared autonomy with adaptable levels of autonomy
UC11	unknown articulated objects are present in an unstructured environment	paraplegic user	non-expert able-bodied users	KUKA iiwa	SpaceMouse	shared autonomy with adaptable levels of autonomy

velocity of the robot tool. For example, these can be used for *teleoperation* or *kinesthetic guidance*, which allow the user to control the velocity (or desired force) of the robot tool through external interaction (i.e. with an external device such a joystick) or through physical interaction (i.e. with force/torque sensors mounted on the robot), respectively. The use cases under *target indication* have HMIs that influence the target point (in 2D or 3D) that the robot will interpret as desired reaching position. The use cases under *events* have HMIs to trigger, asynchronously, transitions of a finite state machine. Finally, the use cases under *explainability* have HMIs that communicate information from the robot to the human to explain something related to its state.

Also for the sensors, Table 1.4 shows four different purposes for which sensors are used. Use cases under *proprioception* have sensors to measure internal states of the robot, such as joint torques or joint positions. Use cases under *target objects pose* use sensors to measure the pose of target objects in the environment. Use cases under *obstacle detections* are used to measure obstacles either relative to the robot (distance) or globally in the world frame. Finally, use cases under *external wrench* are used to measure directly or indirectly external loads applied to the end effector.

The reader can find more information about the specific HMIs in appendix A.3. More details about how HMIs are used in this thesis for user-centered robotic devices are given in section 1.4.

Table 1.3. Functionalities assigned to different HMIs for the use cases shown in this thesis. The use case (abbreviated as UC) is mentioned only if the transducers involved are used as HMIs (i.e. to communicate information between the human and the robot) and not as regular sensors.

HMI	TCP velocity	target indication	events	explainability
SpaceMouse	UC4, UC8, UC9, UC10, UC11	UC4	UC4, UC11	
kinesthetic[3]	UC3, UC4, UC9			
3D camera		UC3		
Eye tracker		UC5	UC5	
Polycom system			UC3	UC3
BMI	UC7	UC6		
Artificial Skin	UC1			
Screen monitor				UC2, UC3, UC4, UC5, UC6, UC7
tablet touchscreen			UC8, UC9, UC10	UC8, UC9, UC10

Table 1.4. Outputs taken from different sensors used in the demonstrated use cases in this thesis. The use case (abbreviated as UC) is mentioned only if the sensor involved is used to sense something from the environment (and not from the human) and also if the measurement is fed into a closed-loop controller.

Sensor	proprioception	target objects pose	obstacle detections	external wrench
joint torque sensors[4]	UC3, UC4, UC8, UC9, UC10			UC3, UC4
load cell				UC11
3D camera		UC1, UC3, UC4, UC5, UC8, UC9, UC10	UC1, UC3	
Artificial Skin			UC1	
Encoders	All			

[3] A Kinesthetic HMI makes reference to sensors that are used with the purpose of measuring/estimating the wrench exerted by the user to the end effector, and with the purpose of hand guiding it (e.g. a load cell or joint torque sensors). The term comes from the concept of Kinesthetic teaching.

1.4 Configuration of Human-Machine interfaces

One important aspect of user-centered assistive robotic devices is how the user interacts with them. This can be done using Human-Machine Interfaces (HMIs), which are devices composed of hardware and software that enable information flow between both interacting agents (Human and Machine[5]). In this section we explain the key aspects of HMIs and how they can be configured to suit the needs and preferences of users while performing specific tasks. Such configuration is directly connected to the HMI interaction modalities shown in Fig. 1.2 and involves the stakeholders shown in Fig. 1.3.

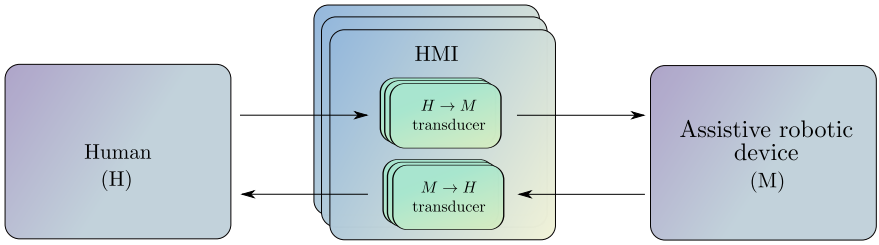


Figure 1.9. Diagram that shows how an assistive robotic device (machine) communicate with the user (human) through a collection of HMIs.

There are two types of information flow that HMIs can provide through the use of transducers: *(i)* information flow from the human (i.e. user) to the machine (denoted by $H \rightarrow M$) that can be used to indicate human intention of changing some aspect of the robot system; and *(ii)* information flow from the machine to the human (denoted by $M \rightarrow H$) that can be used to make the human aware of some aspect of the robotic system. The latter information flow can be used to improve situation awareness, which is known in literature to be very important in robot systems (e.g. [38] and [39]). Both types of information flows can be enabled on the same HMI (e.g. an Xbox controller that vibrates). Fig. 1.9 explains how an assistive robot communicates with the user through a collection of HMIs.

An HMI consists of some/all of the following hardware elements:

[4] In all the applications that use the Franka Panda, proprioception data taken from joint torque sensors (i.e. joint torques) are fed to the Franka impedance controller. In just a couple of cases these joint torques are used to estimate (statically) an external wrench and feed it to an admittance control law generated from specified velocity-level constraints.

[5] In literature the generic term used is *machine*. In the context of this thesis the term machine refers to the computer that is used to control the robot.

1. *human-to-machine ($H \rightarrow M$) transducer(s)*, that convert(s) a physical and measurable value, which is generated due to interaction with the human, to an electrical signal;

2. *machine-to-human ($M \rightarrow H$) transducer(s)*, that convert(s) electrical signals to a physical stimulus that the human perceives through the senses (e.g. actuators for haptics, light bulbs, speakers, etc) for explainability purposes;

3. *signal conditioner(s)*, that transform(s) analog signals that transducers receive/send and converts them into usable and compatible data (e.g. amplifiers, analog filters, saturators, etc);

4. *digital-to-analog converter(s) (DAC)* or *analog-to-digital converter(s) (ADC)*, that make(s) conversions between digital signals that are interpretable and processable by the machine to/from analog signals that translate to physical stimuli;

5. other elements (e.g. ergonomic-related components, processing hardware, etc) that are important but that are out of the scope of the discussion;

Even if the last three elements can sometimes be used to also provide some personalization, we are going to focus on the configuration of the first two elements through the use of software. Thus, the configuration described in the following subsection refers to *which* transducers of the HMI to use and *how* to use them. Both, which and how, are dependent on the application and on user needs/preferences.

It is worth mentioning that each HMI manufacturer enables the user to configure a series of parameters that are specific to the HMI and that can potentially further provide personalization. The configuration referred to in this section includes these specific parameters as part of the whole configuration, but does not focus on it. Instead, we focus on a general abstraction for the common part of the configuration that needs to be defined for all HMIs.

1.4.1 Configuration of HMI transducers

When commissioning and deploying assistive robotic applications, several decisions about the usage configuration of the transducers of the different

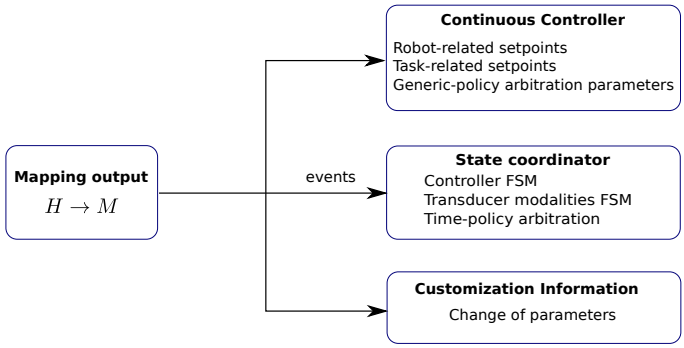


Figure 1.10. Conceptual scheme that shows how the output of $H \rightarrow M$ transducers, which represent interpreted human intentions, can be connected to different subsystems of the robot. This scheme is simplified (i.e. not exhaustive) to fit the purpose of explanation, since in reality various continuous controllers and state coordinators exist and interact with each other based on the configuration (e.g. personalization) provided.

HMIs need to be taken. Since HMIs can have multiple transducers, it is important to select which ones are relevant for the application at hand and how the corresponding signals should be interpreted by the robot system and by the human.

As an example, an Xbox controller, which is a commonly found HMI, can be used in many ways. In the context of robotics, for example, one of its joysticks (a transducer) can be mapped to control certain DOFs of the robot system, while the other joystick can be ignored. In parallel, the buttons (also transducers) can be interpreted to take discrete actions (e.g. close/open the gripper). Furthermore, the vibrator of the controller (another transducer) can be used to indicate information about the state of the robot (e.g. when the force is larger than a safety threshold). Many other possibilities for configuring the transducers of the same HMI for the same application also exist.

It is common in literature to find HMIs that have multiple transducers available, but without explanation on *which* are (or are not) being used for a given application. In addition, detailed information about *how* they are used is often not mentioned or is mentioned ambiguously. Part of *how* the transducers are used is related to how signals of the transducers are mapped. In the case of ($H \rightarrow M$) transducers, the input signals caused by the interaction with the human should be mapped to an interpretable signal to change, discretely or continuously, a state or variable related to the robotic system. In the case of ($M \rightarrow H$), continuous or discrete state

or variable related to the system should be mapped to the transducer in such way that the human can get the information in an interpretable (and hopefully also in an intuitive) way.

Depending on the selected purpose of the $H \rightarrow M$ transducers within the application, their output should be sent to a specific subsystem of the robot (see fig. 1.10). For example, if the transducer is used for controlling the end effector (robot-related setpoint) or the endpoint of an adaptable trajectory (task-related setpoint), the output signal should be sent to a continuous controller. In case that the transducer is used to generate discrete events that should change the state of the robot (e.g. open/close gripper, change control modality, etc), the signal should be sent to a state coordinator (e.g. finite state machine) that is responsible for making the corresponding state transitions. On the other hand, if a transducer (e.g. a capacitive transducer of a touchscreen HMI) is used to reconfigure some aspects of the application/device, the information should be sent to specific configuration handlers to change the corresponding parameters.

For this purpose, it is important to state a clear purpose and interpretation of the input and output signals and, in consequence, define a mapping function from the input to the output. Many types of mapping functions can be used, where the inputs and outputs can be either in a continuous or a discrete space. From simple linear matrix multiplications to deep neural networks, all sorts of mappings could be used, depending on the transducer and on the application. Based on the nature of the interpreted inputs and the desired outputs, the following are common examples of types of mappings that can be specified:

- $\mathbb{R}^{N_{\text{input}}} \rightarrow \mathbb{R}^{N_{\text{output}}}$: mapping from a continuous space of N_{input} real numbers to another space of N_{output} real numbers. An example is a 6-DOF joystick that, by interpreting the input as a desired twist (i.e. linear and angular velocities) and the output as a desired angular velocity[6], maps $\mathbb{R}^6 \rightarrow \mathbb{R}^3$.
- $\mathbb{R}^{N_{\text{input}}} \rightarrow \boldsymbol{S}_{\text{discrete}}$: mapping from a continuous space of N_{input} real numbers to a discrete space of size N_{output}. An example is taking force/torque measurements in $\mathbb{R}^6$ exerted by the robot and activate a light bulb (N_{output}=1) to inform the human when a particular threshold is exceeded.

[6]The linear terms of the twist are ignored, for the sake of the example. Such map is equivalent to perform a map between $se(3) \rightarrow so(3)$.

- $\boldsymbol{S}_{\text{discrete}} \rightarrow \mathbb{R}^{N_{\text{output}}}$: mapping from a discrete space of size N_{input} to a continuous space of N_{output} real numbers. An example is to use a set of two buttons (N_{input}=2) where $\boldsymbol{S}_{\text{discrete}}$ is binary (pressed or not pressed) to continuously open and close a gripper (the gripper moves only while the button is being pressed).
- $\boldsymbol{S}^1_{\text{discrete}} \rightarrow \boldsymbol{S}^2_{\text{discrete}}$: mapping from a discrete space of size N_{input} to a discrete space of size N_{output}. An example is to use one button (N_{input}=1) where $\boldsymbol{S}_{\text{discrete}}$ is binary (pressed or not pressed) to open and close a gripper in a binary domain (entirely open or entirely closed).

In order to facilitate the explanation of these transducer configurations, we propose a meta-model[7] that will be referred to as an *HMI transducer configuration template*. A configuration can then be described using such template for an individual transducer or for a group of transducers that may (or may not) be part of the same HMI. In order to group transducers within the same configuration, it is required that the mapping can be – or needs to be – specified with inputs coming from all the grouped transducers. An example of a group of transducers that, for pragmatism, can be grouped into the same configuration is a collection of buttons that have similar functionalities (e.g. buttons to switch to different operational modes). A situation where transducers need to be grouped into the same configuration is, for example, when sensor fusion techniques are used to map multiple transducer inputs to a combined estimated output.

The HMI transducer configuration template should be filled in transparently so that the different stakeholders can know what to expect when the user interacts with the transducers. The fields of this HMI transducer configuration template are explained as follows[8]:

1. *name*: a unique name identifier that refers to a particular transducer group.
2. *parent HMI*: a unique name that refers to a particular HMI.
3. *information flow*: an indicator that shows whether the transducer passes information from the human to the machine or vice versa.

[7] Further explanation on robot-related meta-models can be found in [20].

[8] A JSON schema could be used to more formally define the transducer configuration template, which provides some implementation advantages, but its explanation is beyond the scope of this section.

4. *DOFs*: an integer number that indicates the number of degrees of freedom of the transducer.

5. *interaction modalities*: a list of possible modalities that are available and can be configured later on to co-exist in parallel and/or to be activated/deactivated during runtime (this is explained later in subsection 1.4.2). To specify each modality the following is required:

    5.1 *identifier*: a unique identifier for referring to that modality.

    5.2 *cause*: a description of the cause (input-related) that will make this modality have a posterior effect.

    5.3 *effect*: a description of the **expected** effect (output-related) that the modality has on the receiving agent of the information flow (i.e. human or machine).

    5.4 *parameters*: a list of parameters related to the usage of the HMI. Some of these parameters are enabled by the manufacturer and some others are enabled by the programmer of the driver (i.e. software to exchange data between the HMI and the robotic system). Even if it is important to include them for a full configuration, in this thesis we ignore them unless it is crucial for the explanation.

    5.5 *mapping*: description of how the DOFs of the transducer are going to be interpreted and mapped for the receiving agent (i.e. human or machine) to act accordingly. This mapping is of course strongly connected to the specified cause and effect, and should be defined consistently to avoid mismatches. To define the mapping the following is required:

        5.5.1 *input interpretation*: a description of how the input is interpreted and, if applicable, in which frame (physical meaning).

        5.5.2 *input units*: a list containing the units of each of the input DOF of the transducer.

        5.5.3 *output interpretation*: a description of how the output will be interpreted and, if applicable, in which frame (physical meaning).

        5.5.4 *output units*: a list containing the units of the elements of the mapping output.

        5.5.5 *function*: a mapping function that maps from the input space to an output space.

5.5.6 *periodicity*: time (in seconds) in which the mapping is performed periodically (inverse to the frequency). A value of 0 indicates that the mapping is performed aperiodically based on triggering events.

The combination of cause and effect defines what the expected human mental model is for the task execution to be successful. If the human's mental model does not match with the specified cause and effect, then frustration and probably unsuccessful task executions will arise as a consequence. Thus, the causes and effects of all modalities should remain transparent and consistent for the user.

The specification of such *transducer configuration* (i.e. filled-in template) can be done in a JSON format for convenience. The transducer configuration related to several HMIs used throughout the use cases presented in this thesis are given in appendix A.3. An example of a transducer configuration is given in Listing 1.1. This example illustrates two common alternatives of interaction modalities, where the user can only control the translation or the orientation of the end effector at the same time. Notice in this example that it is very important to define the input and output interpretations, as the specific linear map directly depends on these definitions and their corresponding units.

Listing 1.1. Transducer configuration of the SpaceMouse joystick for a typical example of mode switching between linear and angular teleoperation of the robot's end effector.

```
{
    "name": "joystick",
    "parent HMI": "SpaceMouse",
    "information flow": H → M,
    "DOFs": 6,
    "interaction modalities":
    {{
        "identifier": "Linear teleoperation",
        "cause": "Linear Interaction with joystick",
        "effect": "Linear motion of end effector",
        "parameters": {...},
        "mapping":
        {
            "Input interpretation": "Desired twist of the
                tcp expressed in a frame attached to the
                joystick",
```

```
                "Input units": [m/s,m/s,m/s,rad/s,rad/s,rad/s],
                "Output interpretation": "Desired linear
                    velocity of the tcp expressed in the tcp
                    frame",
                "Output units": [m/s,m/s,m/s],
                "function": "linear map ℝ^6 → ℝ^3",
                "periodicity": 0.05,
            }
        }
        {
            "identifier": "Angular teleoperation",
            "cause": "Angular Interaction with joystick",
            "effect": "Angular motion of the end effector",
            "parameters": {...},
            "mapping":
            {
                "Input interpretation": "Desired twist of the
                    tcp expressed in a frame attached to the
                    joystick",
                "Input units": [m/s,m/s,m/s,rad/s,rad/s,rad/s],
                "Output interpretation": "Desired angular
                    velocity of the tcp expressed in the tcp
                    frame",
                "Output units": [rad/s,rad/s,rad/s],
                "function": "linear map ℝ^6 → ℝ^3",
                "periodicity": 0.05,
            }
        }}
    }
```

1.4.2 Co-existing interaction modalities

A real robotic assistive device is likely to have multiple HMIs, each having multiple transducers. Furthermore, each transducer can have multiple interaction modalities, and all of them can co-exist in one single application. Multiple interaction modalities related to different transducers can co-exist within the same application. Some of them can exist in parallel and some of them exist as alternative modalities that can be switched when triggering events (e.g. coming from other transducer) occur. They can also have

decoupled or synergetic effects. Examples of pairs of transducers that co-exist with all combinations of effects mentioned above are described as follows:

- *parallel and decoupled effects*: one transducer controls the velocity of the end effector while another transducer (in parallel) controls the gripper.
- *parallel and synergetic effects*: one transducer (e.g. a knob) controls the aperture of a gripper in a continuous space, while another transducer (e.g. a button) controls the same in a discrete space (e.g. opened/closed).
- *alternative and decoupled effects*: one $M \rightarrow H$ transducer (e.g. a screen) can be used to display some information (e.g. effect = the user gets informed about the excerted forces) and, alternatively, can be used to display some other information (e.g. effect = the user gets informed about joint velocities).
- *alternative and synergetic effects*: one transducer next to the robot (e.g. a joystick) controls the velocity of the end effector and alternatively control can be granted to another remote user (e.g. with another joystick).

Typically, the transitions between different interaction modalities and the corresponding triggering events are defined by means of finite state machines.

1.4.3 User-centered HMI configurations for assistive strategies

As explained above, the HMI transducer configuration template can be used to specify multiple interaction modalities. This collection of interaction modalities can be seen as a library that later on the application developer exploits to provide interaction personalization for different applications and various user needs/preferences, as shown in Fig. 1.3. Which modalities co-exist and how they co-exist, depending on the specific user in a certain application, also need to be specified. This can be done, for example, with multiple finite state machines that run in parallel and that transition

between some modalities. Further formalization about how to define these transitions is considered outside the scope of this thesis.

The (preliminary) formalization presented in this section is mostly conceptual, but offers also practical value. For example, it allowed us to use all the aforementioned HMIs for different applications, hence with multiple interaction modalities that complied with particular user mobility limitations. As it will be shown in the next chapters, some $H \rightarrow M$ outputs are used as inputs of a constraint-based task specification (with a corresponding controller) that defines assistive strategies. Thanks to the proposed framework in chapter 3, multiple HMIs can be integrated into a single application and, more importantly, can be easily reconfigured in different ways.

The HMI transducer configurations and the constraints found in the task specification are tightly coupled and should consistently match. More specifically, the interaction modalities that are interpreted as setpoints (see Fig. 1.10) should have corresponding assistive building blocks related to sensor interaction models[9] (SIMs) that are compatible. Mismatches can be avoided if the proposed HMI transducer configurations are filled-in carefully, as they can increase situational awareness of the stakeholders involved during the commissioning and personalization of the assistive robotic device (see Fig. 1.3).

1.5 Contributions

This section summarizes the main contributions of this thesis.

1.5.1 Contribution 1 – reconfigurable and adaptable assistive framework

The first contribution of this work is the proposal of an assistive robotic framework that is:

[9]The sensor interaction models are a type of building block that defines the way in which HMIs and sensors should influence the state of the robotic system. Such building blocks are explained later on in chapter 3.

1. *reconfigurable,* by allowing personalization for different users through the combination of different types of HMIs, assistive strategies, robotic platforms, etc, in a composable way; and
2. *adaptable,* by enabling a systematic way of transitioning between levels of autonomy over decoupled DOF.

This framework provides flexibility to personalize assistive solutions with robotic manipulators and facilitates (re-)configuration to fit changing preferences and needs of users. It also decouples the HMIs (see long list in section 1.3.5) from the behaviors that are used in the different use cases (see long list in section 1.3.4)

Autonomy can be allocated not only over robot-related DOF, but also over decoupled task-related DOF. This, in combination with the offered composability, is a key advantage that gives us more personalization possibilities.

This contribution tackles specific objective 1 and specific objective 2.

1.5.2 Contribution 2 – assistive behaviors for activities of daily living

As a second contribution, we developed several assistive behaviors for assisting users during activities of daily living. These assistive behaviors are specified by using the assistive framework mentioned in *Contribution 1*, and thus they can be easily re-used and reconfigured for different applications and users. The activities of daily living in which we focused on this thesis are performed in unstructured environments (i.e. only little information about the objects and environment is known beforehand) and involve the following actions:

1. reaching or picking objects with arbitrary poses in unstructured environments that make the robot prone to collide with obstacles,
2. interacting with articulated objects, such as doors or drawers, with unknown geometric properties.

Different sensor modalities are used within these behaviors to obtain information about the objects and/or the environment and accordingly adjust the provided assistance.

This contribution tackles specific objective 1 and specific objective 2.

1.5.3 Contribution 3 – experimental validation of heterogeneous configurations

As a third contribution, several experiments were performed to validate the reconfigurability and adaptability of the framework. These experiments demonstrate a wide variety of reconfiguration possibilities on how the proposed framework can be used to personalize the assistance to comply with the mobility limitations of several targeted users. Several use cases that vary the HMIs, the robotic devices, assistive strategies, etc, (see Table 1.2) were developed and deployed during this research. Additionally, several of them have different levels of autonomy, some of which enable on-line adaptation.

We claim that this reconfiguration to personalize/customize assistance provided by assistive devices is facilitated by the proposed framework, and therefore the development and deployment time is reduced. We demonstrate this rapid development and deployment by describing two groups of experiments that were made during strict time limits:

1. A series of experiments were done in collaboration with researchers from the Department of Neuroscience and UZ Leuven (university hospital), with decoded signals measured with an invasive BMI from non-human primate neural activity.
2. A series of experiments were done in a robot manipulator mounted on top of a wheelchair, in collaboration with researchers from the German Aerospace Center (DLR) during a research stay.

This contribution tackles specific objective 3.

1.6 Thesis Outline

Fig. 1.11 shows where some components of this thesis are covered. The outline of the remaining part of this thesis is as follows.

- **Chapter 2** presents a method to generate *Reactive Approach Motions* (RAMo) towards allowable manifolds, enabling interaction with the human and the environment. This chapter provides an initial explanation of the mathematics behind eTaSL and how it can be used to specify such reactive behaviors to, for example, interact with an artificial skin while performing bin picking operations. The reader will get an initial understanding on how to specify robotic reactive behaviors through the use of constraints. Furthermore, it introduces the concept of RAMo, which will be used in chapter 3 for specifying some assistive strategies. The use cases UC1 and UC2 are used to validate the concepts proposed in this chapter.

- **Chapter 3** introduces a reactive and adaptable framework that enables a systematic way of developing user-centered assistive strategies through the specification of constraints. For explanation purposes, we use an alternative mathematical representation of eTaSL, which is equivalent to the one introduced in chapter 2, but more suited to the explanation of this framework. This framework is the backbone of this thesis, which facilitates the desired personalization of assistance. A formalization is presented on how assistive building blocks can be created through the use of constraints. Such building blocks can be either *sensor interaction models* or *reactive action models*. The combination of such building blocks results in the creation of personalized assistive strategies that, for example, enable interaction with multiple HMIs that are suitable for the user. The concepts proposed in this chapter are validated using use cases UC3, UC4 and UC5.

- **Chapter 4** describes the deployment of two assistive applications that benefited from the proposed framework. Both applications had special requirements with strict development timings, and the reconfigurability offered by the framework played a key role and made the development and frequent modifications possible. The first application (use cases UC6 and UC7) was in the context of a Brain-Machine interface which was implanted in monkeys. The second application (use cases UC8, UC9 and UC10) involved the implementation of new assistive strategies on a wheelchair-mounted robot arm, and was developed and deployed during a one-month research stay at the German Aerospace Center (DLR). Challenges and some technical particularities are discussed.

- **Chapter 5** focuses on a new assistive strategy that exploits the proposed framework and that assists the user during interaction with articulated objects, which are commonly found in many activities of daily living. The kinematic properties of such articulated objects are initially unknown, but are estimated online while the robot interacts with the articulation through the use of interactive perception. This chapter combines the constraint-based concepts developed in chapter 3, the task frame formalism and the proposed interactive perception algorithm, to provide assistance that is automatically and seamlessly arbitrated when interactions with novel articulated objects are detected. A **user study** with able-bodied participants was performed in order to validate the proposed assistive strategy through statistical analysis done over collected qualitative and qualitative data. These tasks that the participants did during the user study are based on use case UC11.

- **Chapter 6** contains the overall conclusions about the work presented in this thesis, highlighting the important aspects and limitations. The thesis discusses and concludes each of the chapters individually, and therefore this section is concise and focuses in general key conclusions and limitations.

- **Appendix A.1** contains an explanation about the analysis of the dynamics of the resulting behaviors that are specified using the building blocks described in chapter 3. Some simplified examples of assistive strategies proposed in this thesis are given.

- **Appendix A.2** provides further explanation of the proposed intent estimation algorithm in chapter 3. The influence of the tunable parameters are illustrated by using 2D and 3D plots.

- **Appendix A.3** provides examples on how the HMI transducer configuration template can be used to specify several interaction modalities. In this appendix we provide the different interaction modalities that were used for the different HMIs used in the use cases of this thesis. We also provide a general explanation of these HMIs, and their corresponding transducers.

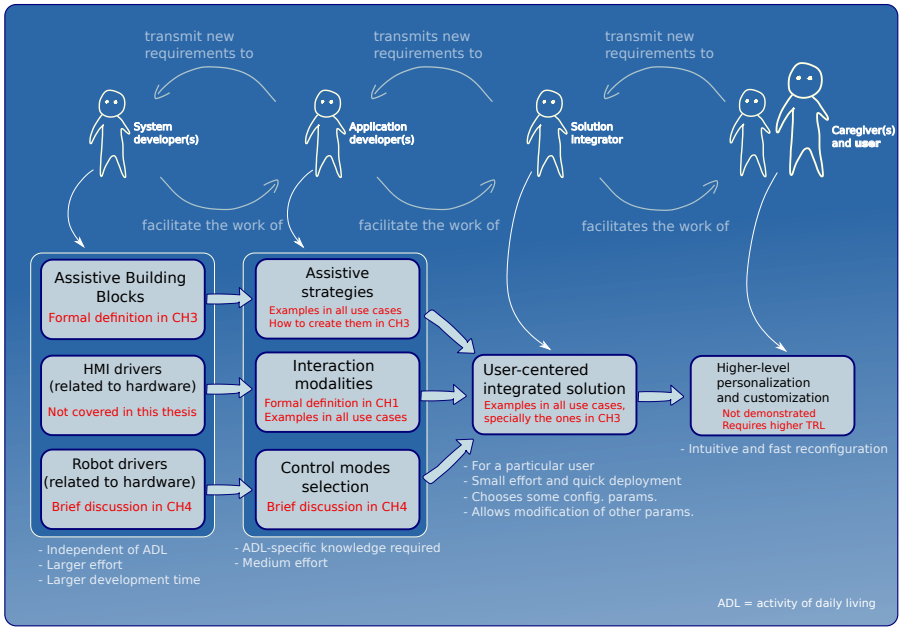


Figure 1.11. Scheme showing in what parts of this thesis the different roles assigned to the different stakeholders are discussed, introduced or exemplified (CH=chapter). This figure was created on top of Fig. 1.3.

Chapter 2

Generating Reactive Approach Motions Towards Allowable Manifolds using Generalized Trajectories from Demonstrations

This chapter is based on [102]: Vergara C., Iregui S., De Schutter J., and Aertbeliën E., Generating Reactive Approach Motions Towards Allowable Manifolds using Generalized Trajectories from Demonstrations, IEEE/RSJ International Conference on Intelligent Robots and Systems, Las Vegas, 2020. While Dr. Cristian Vergara is the main author of this publication, the author of this thesis also made substantial contributions to the development and formalization of the allowable manifolds, the vision system integration, the experimental validation and the writing of the paper, and therefore it is included in this PhD thesis. This paper explains important aspects that were developed for autonomous and semi-autonomous tasks, and that later on will be an integral part of shared-autonomy assistive strategies. Terminology and notation have slightly been adapted to be consistent throughout the thesis. A supplementary video of the chapter can be found in: `https://youtu.be/wz_WT0aoRYM?si=4R-O2C_qvOvhAQko`.

Abstract

There is a high cost associated to the time and expertise required to program complex robot applications with high variability. This is one of the main barriers that inhibit the entry of robotic automation in small and medium-sized enterprises. To tackle the high level of task uncertainty associated with changing conditions of the environment, we propose a framework that leverages a combination between learning from demonstration (LfD) and constraint-based task specification and control. This synergy enables our framework to use LfD to generalize reactive approach motions (RAMo) towards not only a single pose but towards an allowable manifold defined with respect to the object to interact with. As a result, the robot executes the task by following a feasible approach motion generalized from the learned information. This approach motion is generated based on an initial representation of the environment, and it can be reactively adapted in function of current updates of the environment using sensor information. The proposed framework enables the system to deal with applications that involve a high level of uncertainty, increasing the flexibility and robustness, compared to traditional sense-plan-act paradigms.

2.1 Introduction

One of the main challenges of implementing robots in real-world applications is the need for intelligent algorithms that interpret multi-sensor information, thereby enabling robot systems to adapt to dynamic environments. These systems should exhibit the necessary predictability, robustness, and flexibility to deal with geometric uncertainty associated with the robot, task, and environment.

In view of increasing system predictability, in [101], we introduced a combination of the constraint-based methodology *expression-graph based Task Specification Language* (eTaSL) [8] and a Learning from Demonstration approach based on *Probabilistic Principal Component Analysis* (PPCA) [98]. This synergy enables us to use PPCA to generate trajectories defined by a linear combination of learned basis functions weighted by a set of *Degrees Of Freedom* (DOF), while leveraging the definition of eTaSL feature variables to specify these and other DOF associated with the task.

In this paper, we present *Reactive Approach Motions* (RAMo) that further extend the trajectory generation in [1] from specifying a static target pose



Figure 2.1. Snapshots of a human–robot collaborative bin picking application. a) A robot is commanded to align with an augmented surface of the red sphere. During the approach motion, the generated trajectory is reactively adapted to avoid collisions with the bin and the clutter. b) A human is able to move the end effector by interacting with proximity sensors. Collisions with the wall and the clutter are continuously avoided while maintaining alignment with the augmented surface of the red sphere.

toward a dynamic target pose defined on a modeled manifold that can be bounded. A robot programmer can intuitively define this manifold based on inherent task information such as symmetries related to the tool and the work piece. In addition, we increase the flexibility against dynamic obstacles in the environment by enabling local deformations of trajectory segments for instance to enable a tool to circumvent obstacles in different ways.

Hence, as the main contribution of this paper, we extend [101] towards the generation of RAMo, tightly integrating:

1. generation of reactively adaptable trajectories that preserve information provided by human demonstrations while still enabling deformations defined by controlled DOF that affect trajectory segments in their local neighborhood.

2. reactive adaptation of the trajectory goal pose within an allowable manifold defined in function of DOF that can be intuitively specified, bounded and constrained to preferable values. This manifold is

specified using a modular approach where geometric features of the tool, the work piece, and the trajectory are specified separately, while enabling their automatic composition.

The proposed approach increases the flexibility and robustness, enabling the system to deal with applications that involve a high level of uncertainty associated with avoidance of dynamic obstacles.

The framework was evaluated in two use cases that involved the generation and control of RAMo to grasp objects onto which manifolds were defined for the allowable target grasp poses: a simulation use case with dynamic obstacles; and an experiment involving human-robot collaborative bin-picking in which the environment was sensed by an industrial sensor while the human was perceived with proximity sensors as in [100] (see Fig 2.1).

2.2 Related Work

Representations to describe geometrical relations between rigid bodies have been addressed in literature since the early days of robotic research. In [11] the authors proposed mathematical derivations to describe spatial relations between rigid bodies in a goal state. The aforementioned approach inspired the instantaneous Task Specification using Constraints (iTaSC) [27] which extends the Task Frame Formalism [21] towards the definition of geometric body relations as a function of feature frames (virtual frames). This approach enables a systematic constraint-based task specification that deals with complex sensor-based robot behaviors in presence of uncertainties. In [8] the concept of *feature frames* was further extended towards the specification of the robot controller as a function of *feature variables*, which are used as auxiliary variables that enable the specification of DOF of the robot task. This paper exploits the concept of *feature variables* to intuitively define allowable goal-pose manifolds as parametric relationships between rigid bodies, thereby enabling the definition of RAMo that takes into account sensor input as well as robot and environmental constraints.

In view of the search for feasible and stable grasping poses, several methodologies have leveraged the use of manifolds to represent grasping search regions. This is the case in [72], in which the authors used an exhaustive search for a unique stable grasping pose from a predefined set of possibilities. The lack of reactivity to adapt the grasping pose on-line typically limits the robustness of the system against uncertainties. To

deal with this, more advanced systems have used a continuous search for the grasping pose. For instance, the *positioning mobile with respect to fixed frame* (PMF) method [84] introduced allowable manifolds that restrict, totally or partially, motion between two rigid bodies by specifying relations between geometric entities. This approach was further extended in [94] and [93] by also considering relations where the rotation and translation cannot be separated, as well as the composition with additional inequality constraints such as collision avoidance and/or joint limit constraints. Other approaches in literature have leveraged the use of available constraint-based frameworks. This is the case in [17], in which the authors proposed a systematic composition of constraints using eTaSL [8] that takes into account expressions between geometric entities, as well as the runtime monitoring of these constraints. Similarly, in [63] grasping envelopes were defined based on the specification of geometric constraints that are hierarchically solved using the constraint-based framework *stack-of-tasks* (SoT) [69]. Although these methods consider the definition of dynamic goal poses constrained to modeled manifolds, they do not address the generation of reactive approach trajectories that consider context information from the environment. Moreover, in our view, our approach simplifies the commissioning of the task by decomposing the manifold definition in DOF associated with the tool and with the workpiece.

Other methods focused more on the generalization of trajectories that can reactively adapt to changing conditions of the environment while preserving smooth dynamics. For instance, in [78], the authors developed a method to select the appropriate coupling terms of a *Dynamic Movement Primitive* DMP [51] based on a neural network trained with synthetic collision-free trajectories and a minimal representation of the obstacles to dynamically avoid collision in different start-to-goal scenarios. This method nominally generates linear motions and adapts them relying only on a point cloud representation of the environment. Context information of the environment can be added by leveraging LfD, thereby alleviating problems associated with incorrect information from the vision system. For instance, in [58], authors shaped a reactive Lyapunov controller towards an attractor using demonstrations encoded by Gaussian Mixture Models (GMMs). Similarly, in [41], the authors proposed a reinforcement learning algorithm to improve upon demonstrated trajectories when solutions to new situations need to be found, such as for generating collision-free trajectories in dynamic environments. In contrast to these approaches, in this paper, we do not define a unique goal pose for the trajectory, but we specify a dynamic goal pose that is lying on an intuitively modeled manifold.

Few methodologies have addressed the trajectory generation towards goal poses defined on allowable manifolds. For instance, in [35], the authors proposed to generate trajectories that are not constrained to a unique goal pose, but to a *Task Space Region* [14]. In [24], the authors proposed a more reactive approach by leveraging Riemannian geometry to formulate dynamical systems. In contrast to the methods above, our approach leverages the intuitive definition of the task using DOF that can be monitored and controlled. As a result, our system obtains increased predictability and explainability while facilitating the task specification.

2.3 Reactively Adaptable Trajectories

In this section, we introduce adaptable trajectories $\boldsymbol{f}(s, \boldsymbol{\chi}_{\text{path}})$ parameterized in function of DOF corresponding to the normalized path coordinate $0 \leq s \leq 1$, and to a set of variables $\boldsymbol{\chi}_{\text{path}}$ that enable reactive adaptation of the path. These DOF are embedded as feature variables in the control problem in section 2.5. The trajectory is defined by a position profile $\boldsymbol{f}_p(s, \boldsymbol{\chi}_{\text{path}})$ that encodes information generalized from demonstrations, and an orientation profile $\boldsymbol{f}_\psi(s)$ which is modeled.

2.3.1 Position profile

Position paths are generated and reactively adapted within an allowable space encoded from demonstrations, thereby increasing the predictability of our system. We increase flexibility by defining a set of compact basis functions that enable local deformations in the neighborhood of trajectory segments, thus endowing the system with capabilities to circumvent dynamic obstacles not considered in the demonstrations.

The PPCA algorithm is used to formulate a model defined by a linear combination of basis functions learned from N human demonstrations. This model enables generation of trajectories and their reactive adaptation in a predictable way. As described in [101] and [9], it describes a linear space corresponding to the following probabilistic latent model for the position profiles with $D = 3$ signals and their variability:

$$\boldsymbol{f}_{\text{lm}}(s, \boldsymbol{\chi}_{\text{lv}}) = \boldsymbol{W}(s)\boldsymbol{\chi}_{\text{lv}} + \boldsymbol{b}(s), \tag{2.1}$$

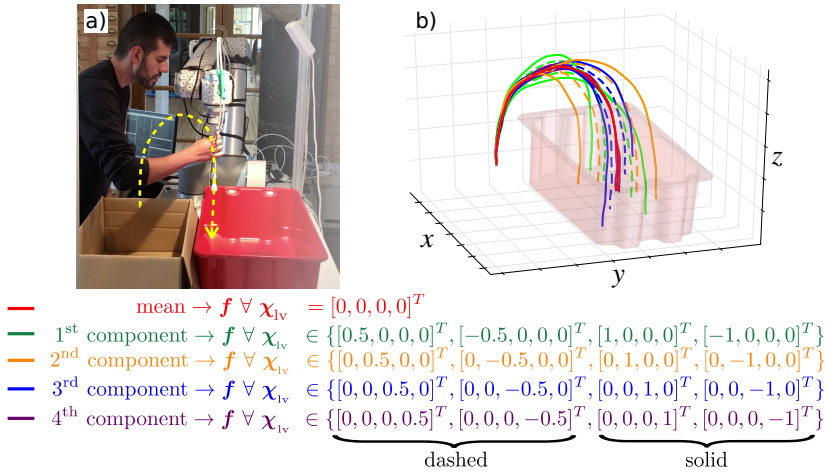


Figure 2.2. Learned model to approach to the bin: a) an operator performs a set of demonstrations to encode the desired position path shape, in this case, entering the bin with a vertical approach; b) graphical representation of the variability encoded in each of the basis functions of the motion model in (2.1).

where $\boldsymbol{W}(s) \in \mathbb{R}^{D\times M}$ contains M basis functions extracted from the set of demonstrations, $\boldsymbol{b}(s) \in \mathbb{R}^D$ corresponds to the mean trajectory of the demonstrations, and $\boldsymbol{\chi}_{\text{lv}} \in \mathbb{R}^M$ contains the latent variables of the model, which are directly embedded as feature variables in the control problem in section 2.5. A graphical representation of the probability distribution of a learned motion model is depicted in Fig 2.2. The figure shows that in this model a larger variability is present along the x-axis, followed by the y-axis, and finally the z-axis.

Local deformations from $\boldsymbol{f}_{\text{lm}}(s, \boldsymbol{\chi}_{\text{lv}})$ are enabled using a piecewise function defined along the trajectory with H bell-shaped ($h = 1, ..., H$) basis functions $a(s)$. To this end, these basis functions are defined as continuously differentiable and compact functions, i.e., each individual basis function only induces deformations in its neighborhood, having no influence outside. These functions can be defined using truncated Gaussians, b-splines or any polynomial that satisfies the mentioned properties. As an example, in this paper we choose polynomials of order 4, parameterized by s-interval widths $\mathrm{w}_h = \{\mathrm{w}_1, \cdots, \mathrm{w}_H\}$ and s-interval centers $c_h = \{c_1, \ldots, c_H\}$; then, we project the polynomials in the normal $\hat{\boldsymbol{N}}_{\boldsymbol{f}}$ and the bi-normal $\hat{\boldsymbol{B}}_{\boldsymbol{f}}$ direction of $\boldsymbol{f}_{\text{lm}}(s, \boldsymbol{\chi}_{\text{lv}})$ modulated by their corresponding amplitude

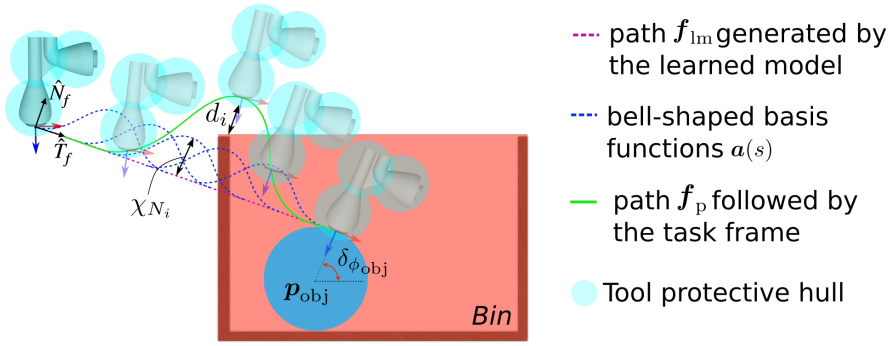


Figure 2.3. 2D representation of local trajectory deformations. The tool moves along f_p to reach the surface of an object located at p_{obj}. Collision between the tool and a bin are avoided by deforming a local section of the path f_{lm} using a piece-wise function defined by a set of bell-shaped functions $a(s)$. The amplitudes of the basis functions χ_{N_i} are computed by constraining distances d_i, measured between the environment and tool protective hulls distributed along f_{lm}, to be larger than a threshold.

$\boldsymbol{\chi}_N = \begin{bmatrix}\chi_{N1} \dots \chi_{NH}\end{bmatrix}^T$ and $\boldsymbol{\chi}_B = \begin{bmatrix}\chi_{B1} \dots \chi_{BH}\end{bmatrix}^T$ as follows:

$$\boldsymbol{f}_{\text{bf}}(s, \boldsymbol{\chi}_N, \boldsymbol{\chi}_B) = \sum_{i=1}^{H} a(s)(\chi_{Ni}\hat{\boldsymbol{N}}_{\boldsymbol{f}} + \chi_{Bi}\hat{\boldsymbol{B}}_{\boldsymbol{f}}) \tag{2.2}$$

$$a(s) = \begin{cases} 16s_h^4 - 32s_h^3 + 16s_h^2 & \text{if } c_h - \frac{\text{w}_h}{2} \leq s \leq c_h + \frac{\text{w}_h}{2} \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

$$\text{with } s_h = \frac{s - (c_h - \frac{\text{w}_h}{2})}{\text{w}_h}. \tag{2.4}$$

The amplitudes $\boldsymbol{\chi}_N$ and $\boldsymbol{\chi}_B$ are DOF embedded as feature variables in the control problem in section 2.5.

To find the tangential $\hat{\boldsymbol{T}}_{\boldsymbol{f}}$, normal $\hat{\boldsymbol{N}}_{\boldsymbol{f}}$ and bi-normal $\hat{\boldsymbol{B}}_{\boldsymbol{f}}$ direction, instead of using of the Frenet-Serret formulas in which the normal component is not well-defined for straight lines, these directions are defined as follows:

$$\hat{\boldsymbol{T}}_{\boldsymbol{f}} = \frac{\frac{d\boldsymbol{f}_{\text{lm}}}{ds}}{||\frac{d\boldsymbol{f}_{\text{lm}}}{ds}||}, \quad \hat{\boldsymbol{N}}_{\boldsymbol{f}} = \frac{\hat{\boldsymbol{d}}_{\text{aux}} \times \hat{\boldsymbol{T}}_{\boldsymbol{f}}}{||\hat{\boldsymbol{d}}_{\text{aux}} \times \hat{\boldsymbol{T}}_{\boldsymbol{f}}||}, \quad \hat{\boldsymbol{B}}_{\boldsymbol{f}} = \hat{\boldsymbol{T}}_{\boldsymbol{f}} \times \hat{\boldsymbol{N}}_{\boldsymbol{f}}, \tag{2.5}$$

where the auxiliary direction $\hat{\boldsymbol{d}}_{\text{aux}}$ is chosen such that $\hat{\boldsymbol{N}}_{\boldsymbol{f}}$ is more likely to be well defined along the entire trajectory, *i.e.*, $\hat{\boldsymbol{d}}_{\text{aux}}$ corresponds to the direction less likely to be parallel to the generated trajectory tangent. Thereby, this direction is chosen by decomposing the tangent vector space of the

demonstrations using PPCA, ordering the principal values in a descending way, and then, choosing the eigenvector corresponding to the last principal value.

Finally, the position profile $\boldsymbol{f}_{\mathrm{p}}(s, \boldsymbol{\chi}_{\mathrm{path}})$ is assembled as follows:

$$\boldsymbol{f}_{\mathrm{p}}(s, \boldsymbol{\chi}_{\mathrm{path}}) = \boldsymbol{f}_{\mathrm{lm}}(s, \boldsymbol{\chi}_{\mathrm{lv}}) + \boldsymbol{f}_{\mathrm{bf}}(s, \boldsymbol{\chi}_N, \boldsymbol{\chi}_B), \tag{2.6}$$

where $\boldsymbol{\chi}_{\mathrm{path}} = \begin{bmatrix} \boldsymbol{\chi}_{\mathrm{lv}}^T & \boldsymbol{\chi}_N^T & \boldsymbol{\chi}_B^T \end{bmatrix}^T$. Fig. 2.3 depicts a graphical representation of this reactive trajectory.

2.3.2 Orientation profile

To reduce the number of demonstrations, as in [101], a reactively adaptable orientation profile $\boldsymbol{f}_{\psi}(s)$ is modeled as a linear interpolation using the axis-angle representation in function of s. This profile spans from a known initial orientation at the beginning of the profile $\boldsymbol{f}_{\psi}(s)|_{s=0}$ towards a variable target orientation at the end of the profile $\boldsymbol{f}_{\psi}(s)|_{s=1}$.

2.4 Allowable Manifolds

The goal pose $\boldsymbol{\Gamma}_G(\boldsymbol{\chi}_{\mathrm{am}}) = \{\boldsymbol{p}_G, \boldsymbol{R}_G\}$ of the generated trajectory is defined within an allowable manifold (hence subscript *am*), specified in function of a set of DOF in which a tool is allowed to move with respect to an object. These DOF are embedded in the control problem in section 2.5 and can be separated into a subset for the surface and another one for the tool ($\boldsymbol{\chi}_{\mathrm{am}} = [{\boldsymbol{\chi}_{\mathrm{surf}}}^T, {\boldsymbol{\chi}_{\mathrm{tool}}}^T]^T$). This separation will be exploited to allow generalization to any combination of surface and tool.

2.4.1 DOF related to the surface

The surface of an object, or a section of it, can be modeled with a continuously differentiable function $\boldsymbol{g}(\mathrm{u}, \mathrm{v}) : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ expressed in frame $\{\mathrm{obj}\}$, where u and v are the generalized coordinates related to the chosen parametrization. In order to determine the goal pose, we first define an intermediate frame whose origin is located at the surface $\boldsymbol{g}(\mathrm{u}, \mathrm{v})$ and one of its axes is always aligned with the normal direction (see Fig. 2.4).

In order to give the controller the freedom to adapt the origin of this frame within the surface, the generalized coordinates are embedded as feature variables in the control problem described in section 2.5, i.e. $\boldsymbol{\chi}_{\text{surf}} = [\text{u}, \text{v}]$. Thus, an orientation $\boldsymbol{R}_{\perp}(\text{u}, \text{v}) = [\boldsymbol{x}_{\perp}\ \boldsymbol{y}_{\perp}\ \boldsymbol{z}_{\perp}]$ (the axes are expressed in frame $\{\text{obj}\}$) whose axis $\boldsymbol{z}_{\perp}$ remains always normal to the surface must be defined. This axis can be found with (2.7) and the remaining axes can be chosen arbitrarily. A good option is to choose the axes with (2.8), where $\boldsymbol{R_0} = [\boldsymbol{x}_0\ \boldsymbol{y}_0\ \boldsymbol{z}_0]$ is the initial orientation of the task frame pose, i.e. $\boldsymbol{\Gamma}|_{t=0}$ (see Fig 2.4).

$$\boldsymbol{z}_{\perp} = -\frac{{}^{d\boldsymbol{g}}/_{d\text{u}} \times {}^{d\boldsymbol{g}}/_{d\text{v}}}{||{}^{d\boldsymbol{g}}/_{d\text{u}} \times {}^{d\boldsymbol{g}}/_{d\text{v}}||}, \tag{2.7}$$

$$\boldsymbol{x}_{\perp} = [\boldsymbol{x}_0 - (\boldsymbol{x}_0 \cdot \boldsymbol{z}_{\perp})\boldsymbol{z}_{\perp}] \text{ and } \boldsymbol{y}_{\perp} = \boldsymbol{z}_{\perp} \times \boldsymbol{x}_{\perp} \tag{2.8}$$

2.4.2 DOF related to the tool

Additional DOF can be added according to the properties of the tool and to the specific application. An example, illustrated in Fig. 2.5, shows how additional translational and rotational DOF are defined in some of the axes of the task frame $\{\text{tf}\}$ (i.e. $\boldsymbol{x}_{\text{tf}}$, $\boldsymbol{y}_{\text{tf}}$ and $\boldsymbol{z}_{\text{tf}}$) for a 2-finger gripper and a suction cup. Based on these DOF we create position and rotation sets, $\boldsymbol{\Phi}$ and $\boldsymbol{\Theta}$ respectively, whose elements are function of the DOF (see Fig. 2.5). The orientation sets are composed by elementary rotation matrices $\hat{\boldsymbol{R}}$ that perform a rotation around one of the axes of $\{\text{tf}\}$ according to the related DOF. The vector that contains all the DOF related to the tool is referred to as $\boldsymbol{\chi}_{\text{tool}}$.

2.4.3 Allowable manifold expression

We then find the expression of the allowable manifold in terms of a position (2.9) and an orientation (2.10)

$$\boldsymbol{p}_G(\boldsymbol{\chi}_{\text{am}}) = \boldsymbol{p}_{\text{obj}} + \boldsymbol{R}_{\text{obj}}\, \boldsymbol{g}(\text{u}, \text{v}) + \sum_{\boldsymbol{\varphi} \in \boldsymbol{\Phi}} \boldsymbol{\varphi} \tag{2.9}$$

$$\boldsymbol{R}_G(\boldsymbol{\chi}_{\text{am}}) = \boldsymbol{R}_{\text{obj}}\, \boldsymbol{R}_{\perp}(\prod_{\hat{\boldsymbol{R}} \in \boldsymbol{\Theta}} \hat{\boldsymbol{R}}) \tag{2.10}$$

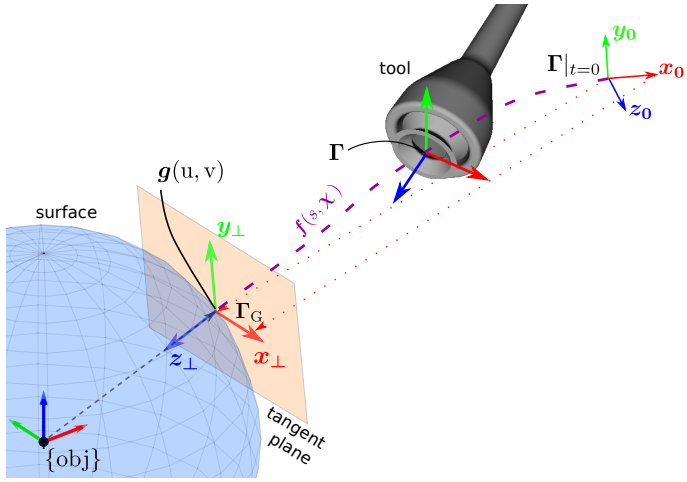


Figure 2.4. Graphical representation of a tool approaching towards a surface that models the geometry of an object. Two DOF of the allowable manifold are associated to the surface while additional DOF are associated to symmetries in the tool. The task frame pose $\boldsymbol{\Gamma}(\boldsymbol{q})$ is commanded to evolve along a reactive trajectory $\boldsymbol{f}(s, \boldsymbol{\chi})$ from an initial pose of the task frame $\boldsymbol{\Gamma}(\boldsymbol{q})|_{t=0}$ towards a goal pose $\boldsymbol{\Gamma}_\mathrm{G}(\boldsymbol{\chi}_\mathrm{am})$. The additional DOF related to the tool are not depicted for ease of understanding.

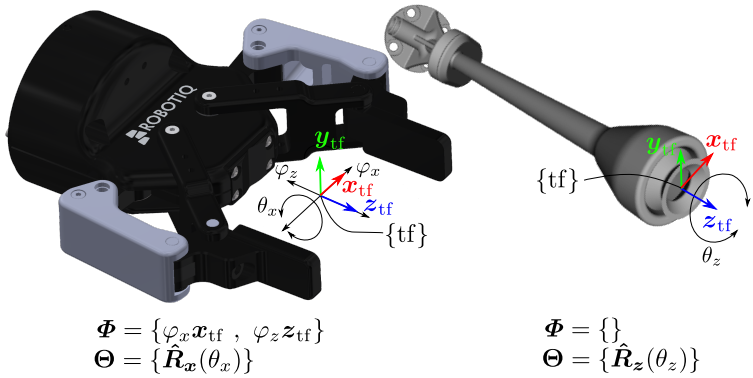


Figure 2.5. Example of possible allowable DOF for selected tools, which are depicted with black arrows. $\boldsymbol{\chi}_\mathrm{tool} = [\varphi_x\ \varphi_z\ \theta_x]$ for the gripper and $\boldsymbol{\chi}_\mathrm{tool} = [\theta_z]$ for the suction cup. The task frame $\{\mathrm{tf}\}$ should be chosen depending on the application, since it will be constrained to reach the allowable manifold at the end of the trajectory (i.e. it can be located with some variation along the $\boldsymbol{z}_\mathrm{tf}$ direction).

where $\boldsymbol{p}_{\text{obj}}$ and $\boldsymbol{R}_{\text{obj}}$ are the position and orientation of the object, expressed in the world frame. In other words, the goal position (2.9) is found by transforming $\boldsymbol{g}(\mathrm{u}, \mathrm{v})$ to the world frame (first two terms) and then giving it translational freedom in some of the axes of the tool (last term). On the other hand, the goal orientation (2.10) is found by multiplying the elements of the rotation set, which gives rotational freedom, and then transforming it to the world frame.

In order to completely define the allowable manifold, inequality constraints are then specified to bound each DOF involved in the definition of the manifold. Some of the bounds depend on the shape properties of both object and tool (e.g. φ_x should be bounded according to the maximum aperture of the gripper and the size of the object). Some others (e.g., u and v) depend on the dimensions of the object. For instance, for a cylindrical surface expressed in cylindrical coordinates, we should bound the coordinate corresponding to the height of the cylinder. It is important to notice that the order of the multiplication in (2.10) affects the bounding values for orientation, and thus it is recommended to choose the task frame alongside the corresponding Euler angle convention, conveniently.

2.5 Specification of the Reactive Control

In this section, we highlight the most relevant aspects from eTaSL [8], and subsequently introduce the control laws that enable the specification of RAMo. However, it is worth to mention that RAMo is not dependent on eTaSL framework per se, but on its implementation of feature variables that allow to define task-related DOF (i.e. other constraint-based frameworks could also be used instead).

The robot control is expressed by means of eTaSL constraints, which are defined as a function of *joint position variables* $\boldsymbol{q}$, *time variable* t, and *feature variables* $\boldsymbol{\chi}$. The latter allow the specification of DOF associated with the task, which can be monitored, controlled, and/or bounded. *eTaSL* formulates the robot behavior as an optimization problem as follows:

$$\underset{\boldsymbol{x}}{\text{minimize}} \qquad \boldsymbol{x}^T \boldsymbol{H} \boldsymbol{x} \tag{2.11a}$$

$$\text{subject to} \qquad \boldsymbol{L_A} \leq \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{U_A} \tag{2.11b}$$

where the argument $\boldsymbol{x}$ is a vector $\begin{bmatrix}\dot{\boldsymbol{q}}^T & \dot{\boldsymbol{\chi}}^T & \boldsymbol{\varepsilon}^T\end{bmatrix}^T$ that contains: the time derivative of the robot joints $\dot{\boldsymbol{q}}$, the time derivative of the feature variables $\dot{\boldsymbol{\chi}}$, and slack variables $\boldsymbol{\varepsilon}$ for each soft constraint. For hard constraints $\boldsymbol{\varepsilon} = \mathbf{0}$. The *weights* $w_i = \mu + \hat{w}_i$ in the diagonal of $\boldsymbol{H}$ enable the system to deal with conflicting (soft) constraints, with a regularization factor μ. $\boldsymbol{L_A}$ and $\boldsymbol{U_A}$ are lower and upper bounds of the constraints described by matrix $\boldsymbol{A}$, respectively. As explained in [101], constraints in (2.11b) can be formulated using a velocity-resolved controller to command a task expression $\boldsymbol{e}(\boldsymbol{q}, \boldsymbol{\chi}, t)$ to evolve towards zero with (2.12), following a first order system with a specified time constant k^{-1}, or to command a task expression $\boldsymbol{g}(\boldsymbol{q}, \boldsymbol{\chi}, t)$ to follow a desired velocity v with (2.13). This formulation is defined as follows:

$$\boldsymbol{J}\begin{bmatrix}\dot{\boldsymbol{q}}\\ \dot{\boldsymbol{\chi}}\end{bmatrix} = -k\boldsymbol{e} - \frac{\partial \boldsymbol{e}}{\partial t} + \boldsymbol{\varepsilon}, \tag{2.12}$$

$$\boldsymbol{J}\begin{bmatrix}\dot{\boldsymbol{q}}\\ \dot{\boldsymbol{\chi}}\end{bmatrix} = \boldsymbol{v} - \frac{\partial \boldsymbol{g}}{\partial t} + \boldsymbol{\varepsilon}, \tag{2.13}$$

where the task function Jacobians $\boldsymbol{J}$ map the time derivatives of states $\boldsymbol{q}$ and $\boldsymbol{\chi}$ into the time derivatives of the task variables $\boldsymbol{e}$. Besides equality constraints, eTaSL also enables the specification of inequality constraints in (2.12) and (2.13).

The formulation of the optimization problem (2.11a) and (2.11b) allows the specification and composition of the following robot behaviors. The number of feature variables involved in (2.11a) is given by the cardinality of the sets defined in sections 2.3 and 2.4, i.e. $|\{s, \boldsymbol{\chi}_{\mathrm{lm}}, \boldsymbol{\chi}_{\mathrm{N}}, \boldsymbol{\chi}_{\mathrm{B}}, \boldsymbol{\chi}_{\mathrm{am}}\}| = 1 + M + 2H + 2 + |\boldsymbol{\chi}_{\mathrm{tool}}|$.

2.5.1 Constraints Related to the Allowable Manifolds

Constraint 1. *Manifold boundaries:* Inequality constraints are used to specify upper and lower bounds for the desired feature variables in $\boldsymbol{\chi}_{\mathrm{am}}$. The use of the feature variables enables an intuitive specification of these manifold boundaries. The constraint can be applied with (2.12) directly on each feature variable or over a mathematical expression $e_i(\boldsymbol{\chi}_{\mathrm{i}})$, where $\boldsymbol{\chi}_{\mathrm{i}} \subseteq \boldsymbol{\chi}_{\mathrm{am}}$. For instance, if the top of a cylinder is desired to be reached, the feature variables related to the equation of the plane must be constrained

with the expression

$$e_1 := \mathrm{u}^2 + \mathrm{v}^2 \leq R^2, \tag{2.14}$$

where R is the radius of the cylinder.

Constraint 2. *Preferable Values:* Additional soft constraints can be added to introduce preferable values $\boldsymbol{V}_{\text{pref}}$ for some of the feature variables involved in the definition of the manifold. These constraints are specified with a low weight to allow deviations in case a possible collision is detected. If specified, the corresponding DoFs will return to this preferable value once the obstacle moves away.

$$\boldsymbol{e}_2 := \boldsymbol{\chi}_{\mathrm{i}} - \boldsymbol{V}_{\text{pref}}, \quad \boldsymbol{\chi}_{\mathrm{i}} \subseteq \boldsymbol{\chi}_{\text{am}} \tag{2.15}$$

2.5.2 Constraints Related to the Trajectory Generation

Constraint 3. *Generation of the nominal trajectory:* A trajectory is generated by commanding its end pose $\boldsymbol{f}(s, \boldsymbol{\chi}_{\text{path}})|_{s=1}$ to coincide with $\boldsymbol{\Gamma}_G(\boldsymbol{\chi}_{\text{am}})$ defined within the manifold. This relation is specified using (2.12) with

$$\boldsymbol{e}_3 := \boldsymbol{f}(s, \boldsymbol{\chi}_{\text{path}})|_{s=1} - \boldsymbol{p}_G(\boldsymbol{\chi}_{\text{am}}) \tag{2.16}$$

This results in the generation of trajectories towards $\boldsymbol{\Gamma}_G(\boldsymbol{\chi}_{\text{am}})$ that can reactively adapt within the combined spaces defined by the demonstrations and the allowable manifold.

Constraint 4. *Deformations along local sections of the trajectory:* In a collision-free motion, the desired behavior corresponds to the tool moving along the path generated by constraint 3, hence, the feature variables $\boldsymbol{\chi}_N$ and $\boldsymbol{\chi}_B$ are commanded to be zero with a soft constraint (2.12).

$$\boldsymbol{e}_4 := \begin{bmatrix} \boldsymbol{\chi}_N^T & \boldsymbol{\chi}_B^T \end{bmatrix}^T \tag{2.17}$$

However, in case an obstacle is perceived along the path, constraint 2.17 is violated by increasing the amplitudes in $\boldsymbol{\chi}_N$ and $\boldsymbol{\chi}_B$ that correspond to the basis functions in the neighborhood of the obstacle (see Fig. 2.3). As a

result, this constraint enables the system to deform $\boldsymbol{f}(s, \boldsymbol{\chi}_{\mathrm{path}})$ to circumvent obstacles.

Constraint 5. *Reactive advancement along the trajectory:* A reactive evolution along the generated trajectory is commanded by specifying the path coordinate s as feature variable and constraining its time derivative $\dot{s}$ to follow a trapezoidal velocity profile v_s defined by a maximum velocity $v_{\max}$ and acceleration $v_{\max}$ [101, 100]. This is formulated using a constraint (2.13) with

$$g_5 := s \quad \text{and} \quad v = v_s \tag{2.18}$$

This results in a reactive behavior that enables the end effector to hold its position or to be driven backwards along the path by performing the appropriate interaction, e.g. by a moving obstacle.

Constraint 6. *Follow motion signals:* The robot motion is generated by commanding $\boldsymbol{\Gamma}(\boldsymbol{q}) = \{\boldsymbol{p}(\boldsymbol{q}), \boldsymbol{R}(\boldsymbol{q})\}$ to coincide with the current instance of the generated trajectory $\boldsymbol{f}(s, \boldsymbol{\chi}_{\mathrm{traj}})$. To this end, the generated trajectory is mapped to a position vector $\boldsymbol{p}_f(s, \boldsymbol{\chi}_{\mathrm{path}})$ and an orientation matrix $\boldsymbol{R}_f(s, \boldsymbol{\chi}_{\mathrm{path}})$, and a constraint (2.12) is applied with:

$$\boldsymbol{e}_{6,p} := \boldsymbol{p}(\boldsymbol{q}) - \boldsymbol{p}_f(s, \boldsymbol{\chi}_{\mathrm{path}}) \tag{2.19}$$

$$\boldsymbol{e}_{6,R} := \boldsymbol{\delta}(\boldsymbol{q}, s, \boldsymbol{\chi}_{\mathrm{path}}) \tag{2.20}$$

where $\boldsymbol{\delta}(\boldsymbol{q}, s, \boldsymbol{\chi}_{\mathrm{path}})$ corresponds to the axis-angle representation of $\boldsymbol{R}(\boldsymbol{q})\boldsymbol{R}_f(s, \boldsymbol{\chi}_{\mathrm{path}})^{-1}$.

2.5.3 Constraints Related to the Environment

Constraint 7. *Point-distance-based collision avoidance:* A collision avoidance behavior is specified by computing distances d_i and $\tilde{d}_i$, which are related to the path and to the tool, respectively. As depicted in Fig. 2.3, this distance is measured from a point cloud representation of the environment towards protective hulls [18] located at the tool (subscript tool) and at H instances of $\boldsymbol{f}(s, \boldsymbol{\chi}_{\mathrm{path}})$ (subscript path). This computation is expedited by representing the point cloud information with k-d trees, representing the protective hull as a set of spheres, and computing the distances d_i and $\tilde{d}_i$ between each sphere and the closest point. Then, d_i and $\tilde{d}_i$ are commanded

to maintain its value above thresholds d_{th} and $\tilde{d}_{\text{th}}$, respectively, by defining a constraint (2.12) with

$$e_{7_{\text{tool}}} := \tilde{d}_i - \tilde{d}_{\text{th}} \geq 0 \tag{2.21}$$

$$e_{7_{\text{path}}} := d_i - d_{\text{th}} \geq 0 \tag{2.22}$$

Therefore, this constraint can act directly upon the motion of $\boldsymbol{\Gamma}(\boldsymbol{q})$ by deviating its progress velocity from the specified velocity v_s, or upon $\boldsymbol{f}(s, \boldsymbol{\chi}_{\text{path}})$ by deforming sections of it even outside of the current neighborhood of $\boldsymbol{\Gamma}(\boldsymbol{q})$. As a result, the generated trajectory (i.e. path and progress) can adapt to circumvent obstacles before the tool reaches their neighborhood.

2.5.4 Additional constraints

Constraint 8. *Joint limits and proximity-based collision avoidance:* Inequality constraints are used to specify bounds on each robot joint position and velocity. Additionally, similarly to [100], a collision avoidance behavior is composed by defining reactive velocities as a function of proximity signals sensed by an artificial skin [73].

2.6 Validation Use Cases

The proposed framework was tested in one simulated and one real use case that involved grasping objects with different geometries. In each use case it was assumed that the following is known: *(i)* the robot platform and its kinematic model (including the tool), *(ii)* the transformation between the vision system and the robot, and *(iii)* the surface model of the objects to pick.

Five demonstrations towards different approach positions $N = 5$ are performed in both use cases to learn the position path using kinesthetic teaching. Fig. 2.2 shows that variability in position can be captured using only four basis functions, therefore, we choose $M = 4$. Moreover, five basis functions are chosen to enable local deformations along the trajectory $H = 5$.

During execution, the information about the pose, the dimensions of the target objects (i.e. $\boldsymbol{g}(\mathrm{u}, \mathrm{v})$), and a point cloud representation of the environment were provided by a vision system (they were known for the simulated use case).

2.6.1 Grasping of objects with gripper

In this use case the task of the robot consists of grasping a target object with the gripper shown in Fig. 2.5. Demonstrations were performed in a real setup to encode a motion model that ensures that the tool always approaches linearly along a vector normal to the object surface, such that the gripper does not collide with it. This use case was performed in a simulated environment so that it facilitates the visualization of the trajectory reactively adapting to avoid obstacles. Instead of using a point cloud as in the next use case, we used a virtual spherical obstacle whose position is controlled with a joystick.

Fig. 2.6 illustrates the response of the generated trajectory when a virtual dynamic obstacle interferes with it. The H instances of $\boldsymbol{f}(s, \boldsymbol{\chi}_{\mathrm{path}})$ are displayed to illustrate how the generated trajectory adapts so that it avoids the obstacle, while the goal pose adapts within the allowable manifold.

Signals related to two trials of the use case are shown in Fig. 2.7. A trial in which a dynamic obstacle causes deviations in the trajectory (as in Fig. 2.6) is compared to a baseline trial in which the obstacle stays still and out of range. When at least one of the collision distances (refer to Constraint 7) is below the defined threshold, the control parameters deviate as follows: *(i)* $\dot{s}$ decreases when the protective hull of the tool is about to collide with the obstacle, *(ii)* $\boldsymbol{p}_{\mathrm{tf}}$ deviates from $\boldsymbol{f}_{\mathrm{lm}}(s, \boldsymbol{\chi}_{\mathrm{lm}})$ due to the local deformations (see (2.6)), *(iii)* the orientation remains almost constant, but slightly deviates because the target orientation adapts within the allowable manifold, *(iv)* $\boldsymbol{\chi}_N$ and $\boldsymbol{\chi}_B$ increase the value of the corresponding basis functions amplitudes, thereby enabling the system to avoid collisions, *(v)* the latent variables $\boldsymbol{\chi}_{\mathrm{lm}}$ of the first three principal components of the learned motion model slightly deviate due to the change in the target position within the allowable manifold. Notice that $\boldsymbol{\chi}_{\mathrm{lm}}$, corresponding to the last principal component, has large deviations when the obstacle moves away. However, notice from Fig. 2.2 that this component has a negligible influence in this particular learned model and thus only three basis functions could have been used instead.

Table 2.1. Constraint Parameters.

	Param.	Value
1	k_1	$2\ \mathrm{s}^{-1}$
	$\hat{w}_1$	50
2	k_2	$1\ \mathrm{s}^{-1}$
	$\hat{w}_2$	0.1
3	k_3	$8\ \mathrm{s}^{-1}$
	$\hat{w}_3$	20
4	k_4	$2.5\ \mathrm{s}^{-1}$
	$\hat{w}_4$	5
5	k_5	0
	$\hat{w}_5$	0.5
6	k_6	$1\ \mathrm{s}^{-1}$
	$\hat{w}_6$	10
7_{tool}	$k_{7_{\mathrm{tool}}}$	$1\ \mathrm{s}^{-1}$
	$\hat{w}_{7_{\mathrm{tool}}}$	50
7_{path}	$k_{7_{\mathrm{path}}}$	$8\ \mathrm{s}^{-1}$
	$\hat{w}_{7_{\mathrm{path}}}$	10

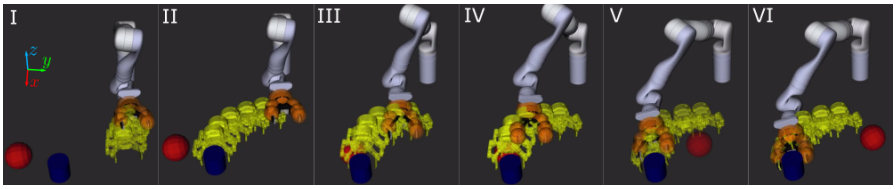


Figure 2.6. Snapshots for dynamic obstacle avoidance in simulation. The blue cylinder represents the object to be grasped, the orange spheres depict the protective hull related to the gripper, the yellow grippers represent the location of the H instances of $\boldsymbol{f}(s, \boldsymbol{\chi}_{\mathrm{traj}})$, and the red sphere is a dynamic obstacle controlled by a joystick.

In Table 2.1, the values of the weights $\hat{w}$ and control gains k for each i-th constraints (see section 2.5 are reported. These values were tuned using intuitive heuristics, based on the relative importance of the constraints and their desired reactiveness. Constraint 2, for example, has a relatively low importance (low weight), since it represents a preferable value that the goal pose will reach whenever there is no obstacle. In contrast, the weights for constraint 1 and 7_{tool} are high, because these constraints are not allowed to be violated. Regarding the tuning of the control gains k_i, constraint 6 and 7_{tool} have low values in order to avoid abrupt movements of the robot, while the gains related to the path generation are higher to enable faster adaptation.

2.6.2 Bin-picking use case

In the second use case, RAMo were used to generate collision-free trajectories that move a tool with a suction cup (see Fig. 2.5) to pick objects from a cluttered tall bin. Further reactivity is enabled in the system to adapt

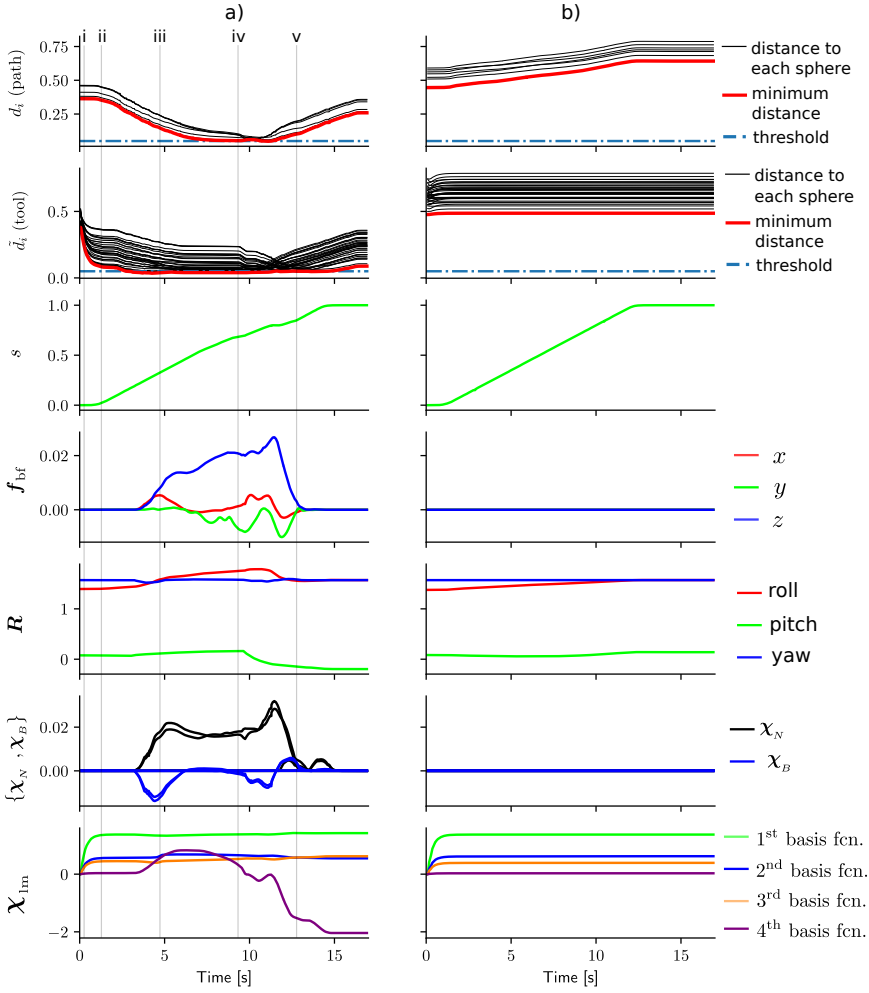


Figure 2.7. Control signals related to the grasping of objects with gripper use-case: a) Signals with dynamic obstacle (see Fig. 2.6) and b) signals without obstacle. The perturbations of the signals, caused due to the obstruction by the obstacle during a part of the trajectory, can be observed. Instances in roman numbers correspond to the snapshots in Fig. 2.6.

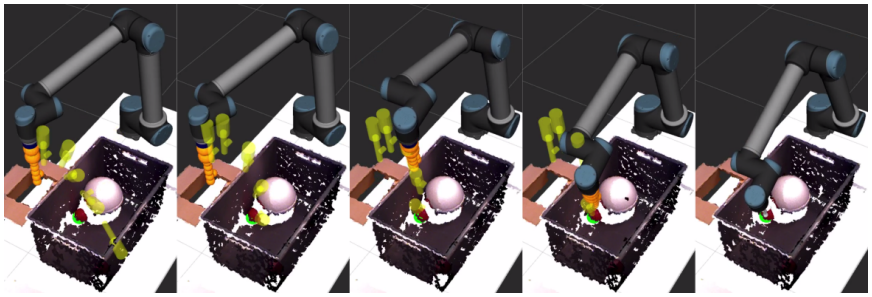


Figure 2.8. Snapshots of the point-cloud visualization in the bin-picking experiment in Fig, 2.1. A full trajectory generated by the learned model in Fig. 2.2 is reactively adapted to command its end pose $\boldsymbol{f}(s, \boldsymbol{\chi}_{\text{path}})|_{s=1}$ to align with the red sphere. Distances between tool representations (yellow cylinders) and the point-cloud representation of the environment are continuously checked, thereby, enabling the system to avoid collisions.

the generated trajectories in case a human is perceived using proximity sensors as in [100].

The learned motion model in Fig. 2.2 was demonstrated in such a way that the tool always approaches vertically to the bin, in order to avoid collisions with the walls. Considering the used tool, the total number of feature variables involved in this application was eighteen.

Fig. 2.1.a shows a tool moving to grasp a sphere in a cluttered environment close to the bin walls. The adaptation of the generated trajectory $\boldsymbol{f}(s, \boldsymbol{\chi}_{\text{path}})$ allows the robot to align the suction cup with the spherical surface while avoiding collisions with the environment. The definition of the allowable manifold gives additional flexibility to the system by enabling the $\boldsymbol{\Gamma}_G(\boldsymbol{\chi}_{\text{am}})$ to adapt to avoid collisions (see Fig. 2.8).

Fig. 2.1.b shows an operator actively interacting with the robot by approaching his hand towards proximity sensors embedded in an artificial robot skin. Distances between the protective hulls and a point cloud representation of the environment as well as the proximity were continuously checked. During this interaction, the suction cup maintained its alignment with an augmented spherical surface $\boldsymbol{g}(\mathrm{u}, \mathrm{v})$ while avoiding collisions with the environment. The augmented spherical surface was used, instead of one that fitted the object, to avoid that the tool touched the object during this demonstration.

2.7 Discussion and Conclusion

This paper extends the synergy between LfD and a constraint-based approach by enabling the generation of RAMo towards specified allowable manifolds. In contrast to pure trajectory planners that only rely on the sensed information, the use of LfD enables our system to mitigate sensor limitations that cause poor visibility, e.g. poor representation of the bin walls due to specular surfaces or occlusions.

The achieved reactiveness enables the robot system to reach the goal while avoiding dynamic obstacles and interacting with humans in cluttered environments. The integration of an artificial skin and a vision system allowed us to exploit the aforementioned capabilities in a safe way for the humans.

In contrast to the state of the art, our approach allows a more intuitive definition of the allowable manifolds, which is performed by selecting position and orientation DOF that depend on the object surface and the tool. The two DOF related to the object surface are used to give freedom at the position level, while maintaining the axis of the tool normal to the surface. On the other hand, the geometric properties of the tool can be exploited by the robot task programmer to add more flexibility to the robot skill, at the expense of increasing the number of optimization variables.

The explicit selection of the task variables to learn and the ones to model, expedite the deployment of bin-picking applications. In comparison to pure model-based approaches, this is achieved because the human can easily abstract and demonstrate the context-dependent motion information relevant to achieve the task. On the other hand, in comparison to end-to-end learning methods, this is achieved due to a drastic reduction in the required number of demonstrations.

The orientation component of the approach motion as well as the evolution profile are modeled. These assumptions reduce the variability that need to be learned, therefore, reducing the number of required demonstrations. The number of independent parameters in PPCA grows linearly with the number of learned signals D as opposed to methods based on Gaussian distributions, in which this number increases quadratically (as in [41]). As a result, the *curse of dimensionality* can be mitigated, encouraging further research to extend PPCA to encode orientation, while maintaining the number of demonstrations N relatively small.

Regarding the definition of the allowable manifold, care must be taken since the representation of $\boldsymbol{g}(\mathrm{u}, \mathrm{v})$ could introduce some singularities. In addition, the arbitrary selection of $\boldsymbol{x}_{\perp}$ in (2.8) can induce a singularity if $\boldsymbol{x}_0$ is parallel to the normal axis $\boldsymbol{z}_{\perp}$.

In case the target objects have a complex geometry, our framework can still be applied by defining the allowable manifold with just DOF related to the tool (provided a goal pose). For example, if a suction cup is mounted, which does not need to be perfectly normal to the surface due to its compliance, infeasible picking poses can be avoided by adapting its orientation.

Future applications of our framework could benefit from a higher level vision-based algorithm which can fit a surface model for the detected object. This could be based on a generic function which can be fitted offline to the point cloud (e.g. a Gaussian Mixture Model), or fitting the most similar shape primitive to it. However, this is out of the scope of our research and we encourage it as future work.

As a conclusion, the proposed method generates RAMo not only preserving information from demonstrations but also reactively adapting to changes in the environment. This reactiveness is achieved by adding freedom in terms of variables related to the task specification. This adds extra flexibility to the system by enabling deviations of the generated trajectory at three different levels: *(i)* deviations in the whole position trajectory that satisfy the information encoded by our LfD method; *(ii)* deviations that are localized at a region of the position trajectory, which extend the flexibility of the learned model; and *(iii)* deviations in the goal pose within an allowable manifold, which is particularly useful for changes in the environment occurring near the object.

In our view, the specification of robot behaviors by monitoring and controlling the task DOF increases the system's predictability and explainability. These aspects are essential to improve the safety and robustness in real robot applications while enabling adaptability to dynamic environments.

Chapter 3

Reconfigurable Constraint-Based Reactive Framework for Assistive Robotics with Adaptable Levels of Autonomy

This chapter is based on [52]: Iregui S., De Schutter J., and Aertbeliën E., Reconfigurable Constraint-Based Reactive Framework for Assistive Robotics With Adaptable Levels of Autonomy, IEEE Robotics and Automation Letters, 2021. Terminology and notation have slightly been adapted to be consistent throughout the thesis. A supplementary video of the chapter can be found in:
`https://youtu.be/LdXKWvSUy3E?si=iQ7joxy3Qijax7nQ`.

Abstract

In this work, we propose a constraint-based reactive framework that allows an easy customization of assistive robotic systems to comply with the particular needs of users with different levels of mobility. This framework enables the specification of modular assistive reactive behaviors which can be easily composed to allocate autonomy over robot-related or task-related

variables. Such assistive behaviors can be systematically modulated online, which results in adaptable levels of autonomy. Modulation can be performed directly by the user through suitable human-machine interfaces (HMIs), or can be performed automatically by the computer. Although our framework also shows potential for handling other types of tasks, we focus on assisting the user with reaching tasks. We experimentally validated the flexibility and applicability of our framework by presenting three assistive use cases, each targeting users with different level of mobility. Each of those cases was easily reconfigured to include various HMIs (e.g. SpaceMouse, eye tracker, etc.), exploiting different assistive strategies. During the experiments, the level of autonomy could be automatically or manually adapted.

3.1 Introduction

The quality of life of people with upper motor disabilities, often caused by spinal cord injuries, is severely reduced. Assistive robotic solutions offer an opportunity to improve the quality of life of these people. In most cases, a completely autonomous solution is not desired as shown in [60]. From a neuro-rehabilitation point of view, people have a need to keep their remaining autonomy or recover some of their lost autonomy. People also want to feel in control as much as possible. Thus, robotic assistive solutions have been proposed that share autonomy between human and computer, and that can assist persons with specific levels of mobility (henceforth referred to as users) in particular scenarios.

However, these solutions often provide limited possibilities for customization, ignoring individual mobility difficulties and preferences. Often, customization is limited to changing the HMI used to drive a subset of the DOF of the robotic manipulator.

In this work, we introduce a framework that increases the flexibility to customize assistive solutions with robotic manipulators. For this purpose, we encapsulate the allocation of autonomy over decoupled DOF into a set of building blocks. These building blocks enable continuous reaction to HMI-input and sensor-input measuring changes in the environment, which offers robustness and continuous human-robot interactivity to the application. These building blocks can be *(i) composed*, which enables the possibility of customizing the assistance according to the particularities of the user and the features of the scenario; and *(ii) modulated*, such that the level

of autonomy over computer-controlled or user-controlled variables can be gradually shifted. The latter can be specified by *HMIs for events* (e.g. buttons) that trigger a transition of certain behaviors, or by a *knob-like HMI* that allows to manually modulate the desired autonomy.

Autonomy can be allocated not only over decoupled robot-related DOF, but also over decoupled task-related DOF. This, in combination with the offered composability, is a key advantage of our framework over previous shared autonomy paradigms, as it gives us more customization possibilities. For instance, the user can have autonomy over robot-related DOF or task-related DOF through various types of HMIs such as: *HMIs for teleoperation* (such as 2D or 3D joysticks), *HMIs for kinesthetic guidance* that interact with the end effector through direct force interaction, and *HMIs for target indication* such as eye-trackers that can indicate the end point of a path. More details about the usage of these types of HMIs, and how they are used in this thesis, are given in section 1.3.5. All these HMIs can be combined simultaneously in one application.

Thus, the main contribution of this work is an assistive framework that is: *(i) adaptable,* by enabling a systematic way of transitioning between levels of autonomy over decoupled DOF; and *(ii) reconfigurable,* by allowing customization for different users through the combination of different types of HMIs, assistive strategies, robotic platforms, etc. As an additional contribution, we introduce novel assistive behaviors that can be composed within the framework to suit the needs of users with different levels of mobility. We experimentally validated the reconfigurability and adaptability of this framework by demonstrating three use cases, targeted for able-bodied, paraplegic and tetraplegic users, respectively.

Following section 3.2 on related work, section 3.3 describes the controller used to specify various behaviors, while section 3.4 describes how to modularly create these behaviors through the definition of constraints. The level of autonomy given by the behaviors can be arbitrated by means of the method presented in section 3.5, which decides based on the monitoring of parameters such as the user intent estimated in section 3.6. We present experimental validation in section 3.7 and conclusions in section 3.8.

3.2 Related Work

Different assistive solutions have been designed with HMIs suitable for certain limitations of mobility. For instance, in [105], tetraplegic users can use an eye tracker for indicating the desired object through their gaze, which makes the robot automatically follow an offline planned path towards the object. Other approaches, like [67], grant more autonomy to users with more mobility and allow them to control a 7 DOF manipulator in a reduced learned latent space controlled by a 2D joystick during object reaching tasks. However, the authors conclude that the latent actions became counter-intuitive and erratic when the state of the robot is not near the training data.

The use of virtual guidance fixtures (VGF) has been explored extensively in assistive robotics, while only recently their adaptation to new situations has been explored. For instance, in [32], the authors proposed a tube-shaped VGF that assists users in performing kinesthetic teaching according to previously demonstrated paths. Although the VGF is activated/deactivated when deviating too much from it, its shape cannot be adapted online according to changes in the environment. The same happens with [4], where the authors generate a VGF from straight lines originating from segmented demonstrated paths, which can be automatically switched on/off when deviating from it. A more adaptive approach to comply with changing environments was proposed in [92], where users can online: generate the geometry of a VGF, switch between generated VGFs, and also adapt their pose. Nevertheless, in [92], such adaptations to comply with changing environments are always handled manually by the user. Reinforcement learning approaches have also been explored to online plan and replan (i.e. to adapt) VGFs, such as in [40] where a mixture of ProMPs is used to represent their paths.

Additionally, constraint-based assistive approaches have been proposed for sharing the autonomy between user and computer. Hierarchical constraints were specified in the Stack of Tasks framework in [95], which make the end effector point autonomously towards a target object while the user teleoperates its position. On the other hand, in [83], constraint-based assistive templates were proposed for specifying non-adaptive shared autonomy behaviors, each belonging to a sequential finite state machine that solves a specific task.

Few customization possibilities have been reported in the state-of-the-art.

For instance, in the approach of [34], online customization of the mapping function of the HMI is enabled and performed by the user through examples. Similarly, in [46], the user is allowed to customize the level of assistance in a shared autonomy system through verbal cues.

3.3 Task Specification and Control in eTaSL

In order to specify the behaviors that compose our assistive framework, we use the constraint-based task specification and control framework eTaSL [8], as explained in section 2.5. In eTaSL[1], the behaviors are expressed by a set of hard (high priority) and soft (low priority) constraints over expressions, which are defined as functions of *joint positions* $\boldsymbol{q}$, *time* t, and *feature variables* that are grouped in vector $\boldsymbol{\chi}$. These feature variables are defined as auxiliary free variables that facilitate the specification of DOF associated with the task. They can be monitored, controlled and/or bounded. eTaSL translates this specification of behavior automatically into the quadratic problem (2.11), which is re-written into an alternative (but equivalent) form for convenience as follows:

$$\underset{\dot{\boldsymbol{Q}},\boldsymbol{\varepsilon}}{\text{minimize}} \qquad \dot{\boldsymbol{Q}}^{\intercal}\mu\boldsymbol{I}\dot{\boldsymbol{Q}} + \boldsymbol{\varepsilon}^{\intercal}\boldsymbol{W}_{\varepsilon}\boldsymbol{\varepsilon} \tag{3.1a}$$

$$\text{subject to} \qquad \boldsymbol{L}_{\boldsymbol{A}} \leq \boldsymbol{A}\dot{\boldsymbol{Q}} \leq \boldsymbol{U}_{\boldsymbol{A}}, \tag{3.1b}$$

where $\boldsymbol{Q} = [\boldsymbol{q}^{\intercal}\ \boldsymbol{\chi}^{\intercal}]^{\intercal}$. For each constraint i, a slack variable ε_i (grouped in $\boldsymbol{\varepsilon} = [\varepsilon_1^{\intercal}, ...\varepsilon_n^{\intercal}]^{\intercal}$) is defined, which penalizes deviations from the soft constraint with a *weight* $w_i = \mu + \hat{w}_i$ (grouped in the diagonal matrix $\boldsymbol{W}_{\varepsilon}$). For hard constraints $\boldsymbol{\varepsilon}_i = \boldsymbol{0}$. The regularization factor μ punishes large $||\dot{\boldsymbol{Q}}||^2$ in case of an underdetermined set of constraints or in the neighborhood of singularities. The constraints (3.1b) ensure that task expressions $\boldsymbol{e}_i(\boldsymbol{q}, \boldsymbol{\chi}, t)$ evolve to a setpoint $\boldsymbol{e}_{s_i}$ by following the first order system (3.2) with time constant k_i^{-1} (denoted as $\boldsymbol{e}_i \rightarrow \boldsymbol{e}_{s_i}$) or evolve to a desired velocity $\boldsymbol{v}_i$ (3.3) (denoted as $\dot{\boldsymbol{e}}_i \rightarrow \boldsymbol{v}_i$)[2].

[1] Open source code, tutorials, examples and a template compatible with multiple commercial robots available at `https://etasl.pages.gitlab.kuleuven.be/`.

[2] Notation of equations (2.12) and (2.13) was slightly adapted in (3.2) and (3.3), respectively, and setpoints $\boldsymbol{e}_{s_i}$ were explicitly included in (3.2). These changes were made for explanation purposes and do not make fundamental changes in the eTaSL formulation.

$$\boldsymbol{J}_i\dot{\boldsymbol{Q}} = -k_i(\boldsymbol{e}_i - \boldsymbol{e}_{s_i}) - {}^{\partial \boldsymbol{e}_i}/_{\partial t} + \boldsymbol{\varepsilon}_i, \tag{3.2}$$

$$\boldsymbol{J}_i\dot{\boldsymbol{Q}} = \boldsymbol{v}_i - {}^{\partial \boldsymbol{e}_i}/_{\partial t} + \boldsymbol{\varepsilon}_i, \tag{3.3}$$

where $\boldsymbol{J}_i = {}^{\partial \boldsymbol{e}_i}/_{\partial \boldsymbol{Q}}$ determine the rows of $\boldsymbol{A}$. For inequality constraints, the equal sign in (3.2) and (3.3) is replaced with $\geq$ or $\leq$. The right hand side terms of (3.2) and (3.3) determine the limits $\boldsymbol{L_A}$ and/or $\boldsymbol{U_A}$. eTaSL performs automatic differentiation to calculate all the involved derivatives, facilitating the reconfigurability of the system. See [8] for details about the derivation of (3.2) and (3.3) and the numerical solution of (3.1). For further insights about the dynamic analysis and stability related to eTaSL, see Appendix A.1.

3.4 Reactive Assistive Behaviors

This section explains how different assistive strategies can be specified using two types of *building blocks* (BBs) which, when combined, create reactive behaviors. The first type, referred to as *reactive action model* (RAM), consists of a set of constraints and variables (task-related and/or robot-related) that define a nominal action (e.g. follow a path) and the ability to deviate from it according to these variables. The second type, referred to as *sensor interaction model* (SIM), consists of a set of constraints that define the interaction between the variables in the RAMs and a sensor or HMI. Both desired and unintended effects that the BBs produce onto each other (e.g. due to environmental disturbances measured with the sensors) are referred to as interactions (see Fig. 3.1). This separation into the two types facilitates the reconfiguration of BBs to assist a user with particular limitations in a task, using suitable HMIs and sensors.

The connection of these two types of BBs with the rest of the framework is shown in Fig. 3.1. The constraints defining the BBs explained in this section are listed in Table 3.1, which are then assembled by eTaSL to automatically generate the controller. To keep the explanation concise, the dependencies of the expressions $\boldsymbol{e}_i$ are only explicitly mentioned when needed for clarity.

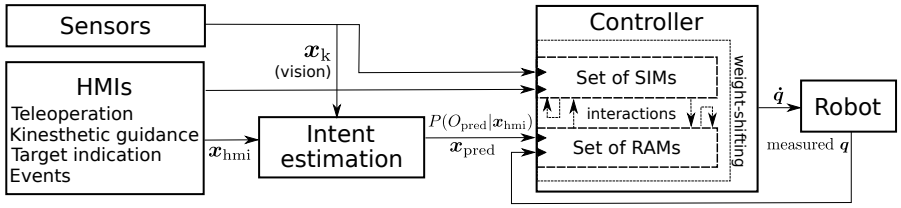


Figure 3.1. A scheme of how the components of the framework are connected. The symbols denoting the signals are explained in section 3.6.

3.4.1 Sensor interaction models (SIMs)

SIMs not only enable the direct interaction between the sensed data and the robot itself, but also with adaptive paths whose shapes can change online. This section presents a non-exhaustive list of SIMs, useful for different applications.

Position SIM: Constraints in the form of $\boldsymbol{e}_i \rightarrow \tilde{\boldsymbol{R}}_{\mathrm{p}}\boldsymbol{x}_{\mathrm{s}}$ can enforce any expression $\boldsymbol{e}_i$ in $\mathbb{R}^{N_p}$ to follow position sensor data $\boldsymbol{x}_{\mathrm{s}}$, mapped by a transformation expression $\tilde{\boldsymbol{R}}_{\mathrm{p}}$. For instance, constraint 4 allows the user to control the end position $\boldsymbol{f}_p|_{s=1}$ of the adaptive path $\boldsymbol{f}_{\mathrm{p}}$ (defined in subsection 3.4.2) through an HMI for target indication whose position input is $\boldsymbol{x}_{\mathrm{hmi}}$. In addition, collision avoidance is implemented by bounding (in)directly measured distances to obstacles using inequality constraints. For instance, constraint 5 constitutes a path obstacle avoidance SIM by keeping distances d_i, measured from an obstacle to j protective spheres located along $\boldsymbol{f}_{\mathrm{p}}$ with equidistant values of the arc length, above a threshold d_{th}. Similarly, a tool obstacle avoidance SIM (constraint 6) keeps distances $\tilde{d}_i$, from an obstacle to h protective spheres distributed along the robot tool, above a threshold $\tilde{d}_{\mathrm{th}}$.

Velocity SIM: Constraints in the form of $\dot{\boldsymbol{e}}_i \rightarrow \tilde{\boldsymbol{R}}_v\,\boldsymbol{v}_{\mathrm{s}}$ enforce the velocity of any expression $\boldsymbol{e}_i$ in $\mathbb{R}^{N_v}$ to follow velocity sensor data $\boldsymbol{v}_{\mathrm{s}} \in \mathbb{R}^{N_v}$, mapped by a transformation expression $\tilde{\boldsymbol{R}}_v$. For instance, constraint 7 enables the end effector, whose position is denoted by $\boldsymbol{p}(\boldsymbol{q})$, to be controlled by an HMI for teleoperation such as a joystick. This can be performed in the world frame ($\tilde{\boldsymbol{R}}_v = \boldsymbol{I}$) or in the tool frame.

Force SIM: Admittance control reacting to measured forces $\boldsymbol{F}_{\mathrm{meas}}$ in a frame with orientation $\tilde{\boldsymbol{R}}_{\mathrm{f}}$, can be specified using constraints of the form of $\dot{\boldsymbol{e}}_i \rightarrow \tilde{\boldsymbol{R}}_{\mathrm{f}} k_{\mathrm{gain}} \boldsymbol{C}_a \boldsymbol{F}_{\mathrm{meas}}$, with a compliance matrix $\boldsymbol{C}_a$ and a control gain

k_{gain} [101]. For instance, constraint 8 performs an admittance control in the tool frame, which has orientation $\boldsymbol{R}(\boldsymbol{q})$.

3.4.2 Adaptive paths (RAMs)[3]

An adaptive path that changes its shape through the specification of two RAMs is generated using the method explained in chapter 2. The first RAM allows the adaptation of a nominal path to reach changing target positions. This path preserves the encoded shapes that are obtained through a Learning from Demonstration (LfD) method based on Probabilistic Principal Component Analysis (PPCA).

This LfD method pre-processes offline a small number of demonstrations (e.g. five) by re-parameterizing them with respect to a progress variable s, and then generates M basis functions represented in the columns of $\boldsymbol{W}(s)$ [101]. These basis functions encode the learned information and are used during execution to generate an adaptive path $\boldsymbol{f}_{\text{lm}}$ (lm = learned model) in $\mathbb{R}^3$:

$$\boldsymbol{f}_{\text{lm}}(s, \boldsymbol{\chi}_{\text{lv}}) = \boldsymbol{W}(s)\boldsymbol{\chi}_{\text{lv}} + \boldsymbol{b}(s), \tag{3.4}$$

where $\boldsymbol{b}(s)$ corresponds to the mean path of the demonstrations, and $\boldsymbol{\chi}_{\text{lv}} \in \mathbb{R}^M$ contains feature variables (i.e. which are included in $\boldsymbol{\chi}$) representing latent variables of the learned model whose time derivative is constrained with constraint 1 in Table 3.1. The normalized progress variable s, whose value is zero at the beginning and one at the end of the path (bounded with constraint 2), is included as a feature variable.

The second RAM defines a path $\boldsymbol{f}_{\text{p}}$ in $\mathbb{R}^3$ that can locally deviate from $\boldsymbol{f}_{\text{lm}}$ and adapt to different disturbances specified through SIMs. This adaptability is achieved by including $\boldsymbol{\chi}_{\text{path}} = \begin{bmatrix}\boldsymbol{\chi}_{\text{lv}}^T & \boldsymbol{\chi}_N^T & \boldsymbol{\chi}_B^T\end{bmatrix}^T$ as feature variables. Thus,

$$\boldsymbol{f}_{\text{p}}(s, \boldsymbol{\chi}_{\text{path}}) = \boldsymbol{f}_{\text{lm}} + \sum_{h=1}^{H} B_{s_h}(\chi_{N_h}\hat{\boldsymbol{N}}_f + \chi_{B_h}\hat{\boldsymbol{B}}_f), \tag{3.5}$$

where the bell-shaped functions $B_{s_h}(s)$ are centered at H instances $s = s_h$ along $\boldsymbol{f}_{\text{lm}}(s, \boldsymbol{\chi}_{\text{lv}})$. Their amplitudes are modulated by $\boldsymbol{\chi}_N = [\chi_{N_1}, ...\chi_{N_H}]^\intercal$ and $\boldsymbol{\chi}_B = [\chi_{B_1}, ...\chi_{B_H}]^\intercal$, respectively in directions $\hat{\boldsymbol{N}}_f(s_h)$ and $\hat{\boldsymbol{B}}_f(s_h)$

[3] This subsection summarizes the adaptive paths introduced in section 2.3.

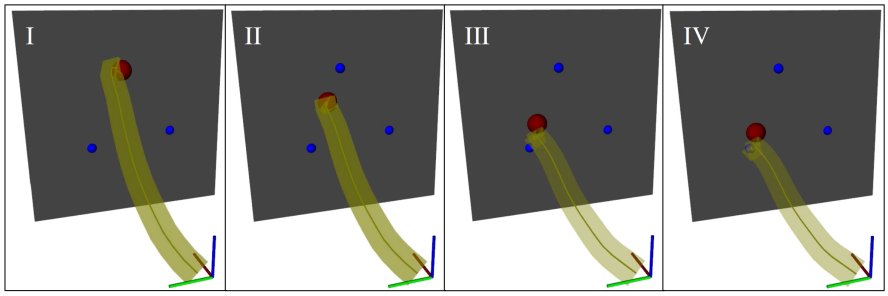


Figure 3.2. Snapshots of how the RVGF (translucent yellow) adapts its shape when continuously shifting $\boldsymbol{x}_{\text{hmi}}$ (red sphere). The blue spheres represent known objects and the opaque yellow path depicts $\boldsymbol{f}_{\text{p}}$.

perpendicular to the tangent of the nominal path at s_h. Constraint 3 gives these feature variables a preferred value, such that it deforms elastically due to disturbances (similar to [18]).

Not only human demonstrations, as in [101] and [102], but also artificial demonstrations (e.g. from modeled or planned paths) can be used, depending on the application at hand.

3.4.3 Attraction to intent-inferred targets (RAM)

The end position $\boldsymbol{f}_p|_{s=1}$ of path $\boldsymbol{f}_{\text{p}}(s, \boldsymbol{\chi}_{\text{path}})$ can be constrained to an intent-inferred position $\boldsymbol{x}_{\text{pred}}$ by means of constraint 9. This RAM grants the computer autonomy over this end position, while the SIM corresponding to constraint 4 gives autonomy to the user. With both BBs active, the autonomy is shared and determined by the weight ratio $\hat{w}_4/(\hat{w}_4+\hat{w}_9)$. By modulating these weights in function of the posterior probability $P(O_{\text{pred}}|\boldsymbol{x}_{\text{hmi}})$ of the user wanting to grasp the predicted object O_{pred} given an input $\boldsymbol{x}_{\text{hmi}}$, we obtain a behavior that assists the user by attracting the end position of the path towards predicted objects (see Fig. 3.2). If we do not consider effects caused by other BBs, the end position will be determined by the weighted average $\boldsymbol{f}_p|_{s=1} = (\hat{w}_4\boldsymbol{x}_{\text{hmi}}+\hat{w}_9\boldsymbol{x}_{\text{pred}})/(\hat{w}_4+\hat{w}_9)$. The modulation of the weights is explained in section 3.5 and the intent-estimation algorithm in section 3.6.

3.4.4 Automatic orientation path (RAM)

The orientation of the end effector $\boldsymbol{R}(\boldsymbol{q})$ can be constrained to follow a custom orientation path $\hat{\boldsymbol{R}}(s)$ by means of constraint 10. As an example,

Table 3.1. Specification of BBs that create the reactive behaviors used in the validation use-cases. Hard constraints are indicated by the omission of $\hat{w}_i$.

#	Expression e_i	Params	Constraints
			RAM, reactive nominal path (3.4)
1)	χ_{lv}	$\hat{w}_1$	$\dot{e}_1 \rightarrow \mathbf{0}$: only adapt shape when needed.
2)	s	$k_2 > 0$	$0 \leq e_2 \leq 1$: limits on the progress.
3)	$\left[\chi_N^\intercal,\ \chi_B^\intercal\right]^\intercal$	$\hat{w}_3$ $k_3 > 0$	$e_3 \rightarrow \mathbf{0}$: RAM, nominal path is preferred in (3.5).
4)	$f_p\vert_{s=1} - x_{\text{hmi}}$	$\hat{w}_4$ $k_4 > 0$	$e_4 \rightarrow \mathbf{0}$: SIM, user controlling $f_p\vert_{s=1}$.
5)	$[d_1, d_2, ...d_j]^\intercal$	$\hat{w}_5$ $k_5 > 0$	$e_{5i} \geq d_{\text{th}}, \forall i = 1, \dots j$: SIM, path avoiding obstacles.
6)	$\left[\tilde{d}_1, \tilde{d}_2, ...\tilde{d}_h\right]^\intercal$	$\hat{w}_6$ $k_6 > 0$	$e_{6i} \geq \tilde{d}_{\text{th}}, \forall i = 1, \dots h$: SIM, tool avoiding obstacles.
7)	$p(q)$	$\hat{w}_7$	$\dot{e}_7 \rightarrow v_{\text{hmi}}$: SIM, end effector velocity.
8)	$p(q)$	$\hat{w}_8$	$\dot{e}_8 \rightarrow R(q) k_{\text{gain}} C_a F_{\text{meas}}$: SIM, end effector force.
9)	$f_p\vert_{s=1} - x_{\text{pred}}$	$\hat{w}_9$ $k_9 > 0$	$e_9 \rightarrow \mathbf{0}$: RAM, attraction to intent-inferred targets.
10)	$\hat{n}_R$	$\hat{w}_{10}$ $k_{10} > 0$	$e_{10} \rightarrow \mathbf{0}$: RAM, automatic orientation path.
11)	$\left\|p(q) - f_p\right\|$	$\hat{w}_{11}$ $k_{11} > 0$	RAM, RVGF $e_{11} \leq r_{\text{tube}}(s)$: remain in tube.
12)	s	$\hat{w}_{12}$	$\dot{e}_{12} \rightarrow 0$: dampen s.
13)	s	$\hat{w}_{13}$	RAM, Auto. reactive approach motion $\dot{e}_{13} \rightarrow V_p(s)$: follow desired velocity along the path.
14)	$p(q) - f_p$	$\hat{w}_{14}$ $k_{14} > 0$	$e_{14} \rightarrow \mathbf{0}$: robot follows path.
15)	q	$k_{15} > 0$	RAM, joint limits $q_{\text{min}} \leq e_{15} \leq q_{\text{max}}$: position limits.
16)	q	$k_{16} > 0$	$\dot{q}_{\text{min}} \leq \dot{e}_{16} \leq \dot{q}_{\text{max}}$: velocity limits.

the following specifies an axis-angle interpolation between an initial orientation $\boldsymbol{R}_0$ and a final orientation $\boldsymbol{R}_1$. The axis-angle representation $(\hat{\boldsymbol{n}}, \hat{\theta})$ is determined by $[\hat{\boldsymbol{n}}\times]\hat{\theta} = \log(\boldsymbol{R}_0^{-1}\boldsymbol{R}_1)$, with $[\hat{\boldsymbol{n}}\times]$ a skew-symmetric matrix corresponding to a normalized vector $\hat{\boldsymbol{n}}$, and $\hat{\theta}$ a scalar angle. The orientation path can be computed as $\hat{\boldsymbol{R}}(s) = \boldsymbol{R}_0 \exp([\hat{\boldsymbol{n}}\times] f_{\hat{\theta}}(s))$ with $0 \leq s \leq 1$. A smooth evolution of the angle $f_{\hat{\theta}}$ that ensures zero derivatives at $s = 0$ and $s = 1$ can be given, for example, by the cubic polynomial $f_{\hat{\theta}}(s) = 3\hat{\theta}s^2 - 2\hat{\theta}s^3$.

We can apply this orientation path $\hat{\boldsymbol{R}}(s)$ as a constraint on the robot orientation $\boldsymbol{R}(\boldsymbol{q})$ using the task function $\hat{\boldsymbol{n}}_R \to 0$:

$$[\hat{\boldsymbol{n}}_R\times] = \log(\boldsymbol{R}(q)^{-1}\hat{\boldsymbol{R}}(s)). \tag{3.6}$$

Notice that if the end effector advances or recedes through the path $\boldsymbol{f}_{\text{p}}(s)$, the progress variable s gets updated and thus the orientation changes automatically.

3.4.5 Reactive Virtual Guidance Fixtures (RAM)

We propose the generation of a tube-shaped Reactive Virtual Guidance Fixture (RVGF) that adapts its shape due to disturbances and guides the user to the target. In this RAM, a tube with radius $r_{\text{tube}}(s)$ wraps around the nominal path $\boldsymbol{f}_{\text{p}}$ and attracts the position of the end effector $\boldsymbol{p}(\boldsymbol{q})$ towards its interior, where the user can freely move the end effector. This RAM consists of an inequality constraint $\left\|\boldsymbol{p}(\boldsymbol{q}) - \boldsymbol{f}_p\right\| \leq r_{\text{tube}}(s)$ (see constraint 11), and results in a non-linear impedance-like guidance when combined with e.g. a velocity or a force SIM. At the same time, this constraint ensures that the value of s is adapted whenever $\boldsymbol{p}(\boldsymbol{q})$ changes due to a disturbance. An additional constraint 12 dampens the motion of s along the path. Notice that the value of s will only be updated when soft constraint 11 is violated, and thus a backlash effect that depends on the magnitude of r_{tube} exists. When r_{tube} is large enough, this can cause an undesired lag on other task expressions that depend on s, such as $\hat{\boldsymbol{R}}(s)$[4].

Fig. 3.3 shows how this RAM, in combination with the previously defined BBs, guides a user through the RVGF, whose shape reactively adapts to

[4] In these cases, the following equality constraint can be added to eliminate the backlash: $(\boldsymbol{p}(\boldsymbol{q}) - \boldsymbol{f}_p(s)) \cdot \boldsymbol{x}_{\text{tan}}(s) = 0$, where the tangential vector of the path can be found with $\boldsymbol{x}_{\text{tan}} = \frac{d\boldsymbol{f}_{\text{p}}}{ds} / || \frac{d\boldsymbol{f}_{\text{p}}}{ds} ||$.

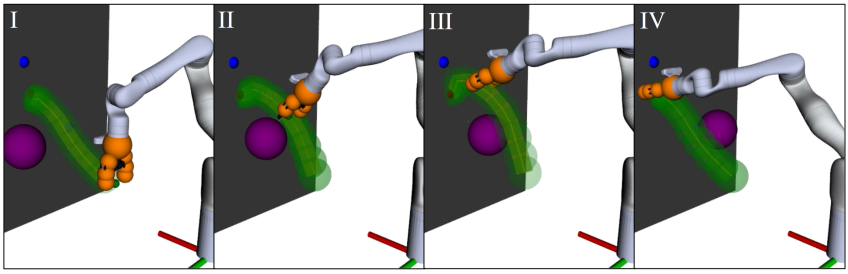


Figure 3.3. Snapshots of the RVGF adapting to a dynamic obstacle (purple sphere) in human-in-the-loop simulation. Protective spheres in the tool and in the RVGF are shown orange and green, respectively.

avoid a dynamic obstacle. For this example, $r_{\text{tube}}(s)$ was defined as a piecewise function that is constant throughout $0 \leq s \leq s_a$, then linearly decreases to zero at $s_a \leq s \leq s_b$, and keeps a zero-value near the target position throughout $s_b \leq s \leq 1$. In addition, Fig. 3.2 shows how the RVGF is attracted to the intent-inferred targets due to the behavior described in subsection 3.4.3.

3.4.6 Automatic Reactive Approach Motion (RAM)

At an arbitrary time, the user may want to grant more autonomy to the computer. For this purpose, two constraints enable the robot to move autonomously from an arbitrary starting position $\boldsymbol{p}_0$ to the target while reacting to disturbances caused by other BBs. Constraint 13 drives s from an arbitrary starting value s_0 towards 1 by imposing a velocity profile $V_p(s)$. Constraint 14 makes the end effector slowly approach $\boldsymbol{f}_{\text{p}}(s)$ (due to a low k_{14}).

Fig. 3.4 shows a diagram that depicts the obtained behavior. While not using it for specification or execution, for analysis purposes, we present a simplified dynamic model[5] that includes this RAM, plus constraints 7 and 12 that allow us to drive the dynamics to zero (with $\boldsymbol{v}_{\text{hmi}} = 0$ since the user is excluded from this analysis). We assume that the Jacobian of end effector does not loose row rank and we omit other constraints such as joint limits, so that the state of the simplified system is purely determined by feature variables $\boldsymbol{Q} = \boldsymbol{\chi} = [\boldsymbol{p}^\intercal \quad s]^\intercal$. By substituting the slack variables of

[5] A more detailed explanation on how to find such simplified dynamic model can be found in Appendix A.1.

the objective function O_b in (3.1) using (3.2) and (3.3), and explicitly solving ${}^{\partial O_b}/_{\partial \dot{\boldsymbol{Q}}} = \mathbf{0}$, we obtain:

$$\dot{\boldsymbol{p}} = \frac{w_{14}}{\mu + w_7 + w_{14}} \left(\frac{\partial \boldsymbol{f}_p}{\partial s} \dot{s} - k_{14} \left(\boldsymbol{p} - \boldsymbol{f}_p(s) \right) \right), \tag{3.7a}$$

$$\dot{s} = \frac{w_{14} \frac{\partial \boldsymbol{f}_p}{\partial s}^{\mathsf{T}} \left(\dot{\boldsymbol{p}} + k_{14} (\boldsymbol{p} - \boldsymbol{f}_p(s)) \right) + w_{13} V_p(s)}{(\mu + w_{14} \frac{\partial \boldsymbol{f}_p}{\partial s}^{\mathsf{T}} \frac{\partial \boldsymbol{f}_p}{\partial s} + w_{12} + w_{13})}. \tag{3.7b}$$

Equation (3.7a) shows that $\dot{\boldsymbol{p}}$ will reduce the error of $\boldsymbol{e}_{14}$ proportionally to the corresponding conflicting weights, and such behavior can be turned off by simply setting $w_{14} = 0$. The term $\dot{s}\, {}^{\partial \boldsymbol{f}_p}/_{\partial s} = \dot{\boldsymbol{f}}_p$ in (3.7a) is a feedforward term that substantially improves the tracking error of $\boldsymbol{e}_{14}$.

On the other hand, (3.7b) shows that if $\hat{w}_{14} = 0$, then $\dot{s} \approx {}^{w_{13}}/_{(\mu + w_{12} + w_{13})} V_p(s)$. This means that $\dot{s}$ will also track the velocity profile V_p proportionally to the corresponding conflicting weights. Nevertheless, when constraint 14 is on, $\dot{s}$ will adapt to disturbances over $\boldsymbol{p}$ based on ${}^{\partial \boldsymbol{f}_p}/_{\partial s} \cdot (\boldsymbol{p} - \boldsymbol{f}_p)$, which can be seen as the tracking error of $\boldsymbol{p}$ projected in the tangent direction of the path, and also based on the corresponding feedforward term ${}^{\partial \boldsymbol{f}_p}/_{\partial s} \cdot \dot{\boldsymbol{p}}$.

Fig. 3.4 shows an example of several numerical solutions of $\boldsymbol{p}$, obtained by solving (3.7) with different initial positions $\boldsymbol{p}_0$ and a fixed path. Since constraints 2 and 11 were omitted, s_0 was estimated offline by minimizing $\left\| \boldsymbol{p}_0 - \boldsymbol{f}_p(s) \right\|^2$ subject to $0 \leq s \leq 1$ (notice the equivalence). The velocity profile $V_p(s)$ was defined as a quasi-trapezoidal profile, whose details will be given in the next section.

3.5 Levels of Autonomy

Autonomy over different aspects of a task can be allocated (i.e. granted) to the user and/or to the robot/computer by the specification of BBs (e.g. user and/or robot control over the velocity of the end effector). Therefore, autonomy can be allocated to the user through a SIM that takes HMI inputs that have been previously mapped according to the desired interaction modality (see section 1.4.1). On the other hand, autonomy to the computer can be allocated through a RAM and/or a SIM, where the latter takes sensor inputs to autonomously interact according to measured changes in the

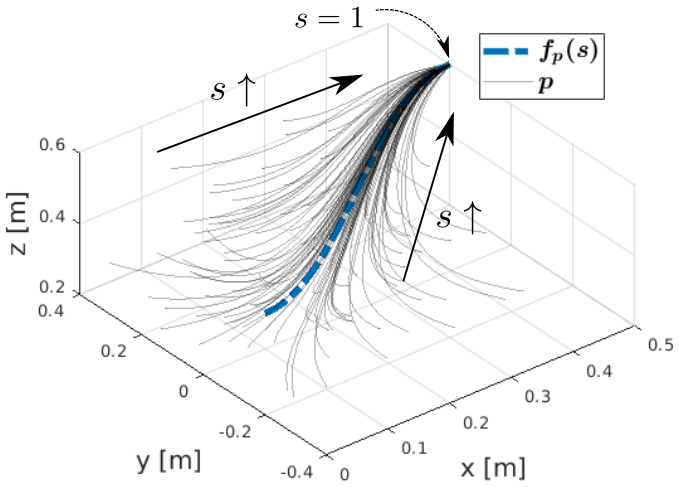


Figure 3.4. Multiple paths obtained by solving the simplified dynamics (3.7) with several initial values of $\boldsymbol{p}$, with $k_{14} = 0.7s^{-1}$ and a fixed path $\boldsymbol{f}_p(s)$, without the presence of disturbances over s and $\boldsymbol{p}$.

environment. When autonomy over the same aspect has been allocated for the user and the robot, it can be modulated online such that it can gradually shift between both. Such modulation action is known as arbitration in literature [66]. The allocated autonomy over one aspect of the task and the ability to modulate it (i.e. arbitration), is referred to as a *level of autonomy*.

This section introduces a systematic and modular approach for enabling arbitration and thus creating different levels of autonomy that are composable. In this approach, arbitration is performed by modulating the weights of the constraints that constitute the different BBs. For example, a level of autonomy over the endpoint of the path can be specified by: 1) allocating autonomy to the user and robot through the BBs related to constraints 4 and 9, respectively, and 2) by modulating the corresponding weights to achieve online arbitration. An assistive application can have multiple levels of autonomy, each arbitrating the autonomy over some aspects specified by certain BBs. Each level of autonomy can influence the weight of multiple BBs and multiple levels of autonomy can influence the same BB.

Each level of autonomy has an *arbitration parameter* G_k that indicates whether the user and/or the robot will have autonomy, such as the distance

to the center of the RVGF or the magnitude of an applied force. These parameters can also be directly controlled by the user discretely (e.g. with an HMI for events) or continuously (e.g. with a knob).

This arbitration architecture is modeled by splitting up each of the weights $\hat{w}_i$ in Table 3.1 into a number of *weight factors* $\hat{w}_{i_k}$ that are function of an arbitration parameter G_k:

$$\hat{w}_i = b_i + a_i \prod_{k \in \Gamma_i} \hat{w}_{i_k}(G_k), \tag{3.8}$$

where Γ_i is the set of subscripts of weight factors that influence the weight $\hat{w}_i$, a_i can be used to scale the product and b_i can be used to avoid driving $\hat{w}_i$ completely to zero. Notice that if any weight factor $\hat{w}_{i_k}$ is zero, the entire product becomes zero and the influence of the corresponding BBs will be negligible (if b_i is zero or a small number). In [102] some intuitive heuristics are specified for selecting the maximum and minimum values of $\hat{w}_i$. The weight factors are a piecewise-continuous function of G_k as defined in Fig. 3.5 and can be *ascending* or *descending*.

Two policies can be used to determine the values of G_k at each time instant: i) *time-policy*, where G_k starts to increase or decrease linearly in time from a fixed value whenever a triggering event occurs (e.g. through an HMI for events); and ii) *generic-policy*, where G_k takes the value of any task-related or user-related parameter or expression occurring in the specification of the BBs.

Table 3.2 presents an example of five different levels of autonomy, each arbitrating the autonomy between user and computer over different aspects of a reaching application. These levels of autonomy are explained as follows:

1. G_1 — Manual (desc.) versus automatic (asc.) target positions: The computer and the user share the autonomy over the end position of path $\boldsymbol{f}_p$, according to the definition of G_1. The arbitration is done automatically by the computer according to the posterior probability (similarly to [36]) explained in section 3.6. This causes $\boldsymbol{f}_p|_{s=1}$ to be attracted to intent-inferred targets, as illustrated in Fig. 3.2.

2. G_2 — Automatic motion (desc.) versus RVGF (asc.): The computer and the user share the autonomy over the position $\boldsymbol{p}(\boldsymbol{q})$ of the robot end effector, and its arbitration is manually performed according to G_2. Thus, the user can activate the automatic reactive approach motion BB by triggering events through an HMI. The force SIM ($\hat{w}_{8_2}$) was excluded from G_2 so that there is

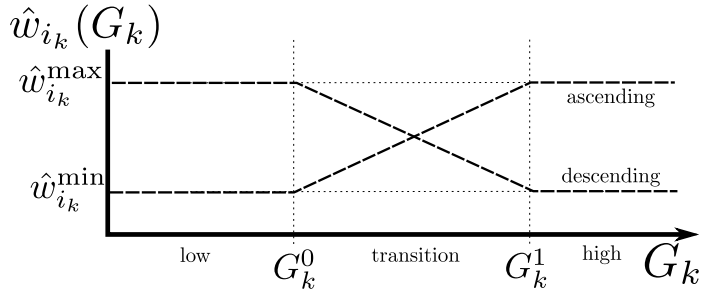


Figure 3.5. Definition of the ascending or descending piecewise-continuous function $\hat{w}_{i_k}(G_k)$ in terms of the parameters G_k^0, G_k^1, $\hat{w}_{i_k}^{\min}$, and $\hat{w}_{i_k}^{\max}$.

always reaction to forces. In addition, a quasi-trapezoidal velocity profile of $\dot{s}$ is achieved ($\hat{w}_{13_2}$). Linear (with respect to time) rise and fall sections of this velociy profile are obtained by triggering an event that switches the value of G_2. Also, by defining $V_p(s)$ as a piecewise function that is constant at $s \leq s_{\text{dwell}}$ and linear from $s_{\text{dwell}} < s \leq 1$, the velocity profile autonomously decreases linearly (with respect to s) towards zero when reaching the object (see Fig. 3.8).

3. G_3 — RVGF (desc.) versus full user control (asc.): Arbitration over $\boldsymbol{p}(\boldsymbol{q})$ is done based on the distance $G_3 = \left\|\boldsymbol{p}(\boldsymbol{q}) - \boldsymbol{f}_p(s)\right\|$. Thus, the user can escape from or come back to the RVGF by simply forcing the end effector away or driving it close to the RVGF.

4. G_4 — Idle (desc.) versus RVGF (asc.): The user has partial autonomy over $\boldsymbol{p}(\boldsymbol{q})$ by means of SIMs that incorporate HMIs for teleoperation or kinesthetic guidance. However, by choosing $G_4 = \|\boldsymbol{v}_{\text{hmi}}\| + \|k_{\text{gain}}\boldsymbol{C}_a\boldsymbol{F}_{\text{meas}}\|$, the impedance-like behavior related to the RVGF is deactivated automatically (i.e. non-linear impedance) when the user is not interacting with the robot. This avoids drift when the user is not interacting with the robot (even if $e_{11} \geq r_{\text{tube}}(s)$).

5. G_5 — Teleoperation (desc.) versus force interaction (asc.): The end effector position $\boldsymbol{p}(\boldsymbol{q})$ can be driven manually by an HMI for teleoperation. Nevertheless, when forces in the end effector are sensed, the corresponding autonomy can be automatically overtaken by the force SIM (to deal with human-robot physical interaction or unexpected collisions).

Table 3.2. Specification of five adaptable levels of autonomy and the involved weight factors. The descending and ascending weight factors $\hat{w}_{i_k}$ are displayed in separate columns. Subscript i indicates the corresponding constraint (see Table 3.1) and subscript k indicates the corresponding arbitration parameter. For all weights $\hat{w}_i$ whose factors $\hat{w}_{i_k}$ are not mentioned here: $a_i = 0$ and $b_i > 0$. For all the factors mentioned here: $\hat{w}_{i_k}^{\max} = 1$, $\hat{w}_{i_k}^{\min} = 0$ and $b_i = 0$, unless stated otherwise.

$\mathbf{G_k}$	**Desc.**	**Asc.**	**Description of the arbitration parameter**
G_1	$\{\hat{w}_{4_1}\}$	$\{\hat{w}_{9_1}\}$	Manual vs automatic target positions *Generic-policy*: $G_1 = P(O_{\text{pred}}\|\boldsymbol{x}_{\text{hmi}})$, $G_1^0 = P_{\min}$ and $G_1^1 = P_{\max}$.
G_2	$\{\hat{w}_{13_2}, \hat{w}_{14_2}\}$	$\{\hat{w}_{7_2}, \hat{w}_{11_2}, \hat{w}_{12_2}\}$	Automatic motion vs RVGF *Time-policy*: $b_{11} > 0$ is a small value in order to keep adapting s.
G_3	$\{\hat{w}_{11_3}, \hat{w}_{12_3}\}$	$\{\}$	RVGF vs full user control *Generic-policy*: $G_3 = \left\|\boldsymbol{p}(\boldsymbol{q}) - \boldsymbol{f}_p(s)\right\|$, where G_3^0 and G_3^1 are threshold distances.
G_4	$\{\}$	$\{\hat{w}_{11_4}, \hat{w}_{12_4}\}$	Idle vs RVGF *Generic-policy*: $G_4 = \|\boldsymbol{v}_{\text{hmi}}\| + \left\|k_{\text{gain}}\boldsymbol{C}_a\boldsymbol{F}_{\text{meas}}\right\|$, where $G_4^0 = G_4^1$ is a small velocity threshold.
G_5	$\{\hat{w}_{7_5}\}$	$\{\hat{w}_{8_5}\}$	Teleoperation vs force interaction *Generic-policy*: $G_5 = \|\boldsymbol{F}_{\text{meas}}\|$, where G_5^0 and G_5^1 are small force thresholds.

3.6 Intent Estimation

In this section, we propose a simple Bayesian algorithm that estimates the intention of the user online. In the context of this work, which is assisting the user to reach the target object, the intention of the user is determined by selecting the most likely object O_{pred} that the user intends to grasp with its corresponding probability $P(O_{\text{pred}}|\boldsymbol{x}_{\text{hmi}})$. The target position input $\boldsymbol{x}_{\text{hmi}}$ is given explicitly (e.g. with a touchscreen) or implicitly (e.g. with gaze estimation) by the user through HMIs for target indication. This approach is able to guide the user from the beginning of the execution, in contrast to approaches that estimate the intention based on user inputs coming from HMI inputs for teleoperation.

We model the posterior probability that object $O_k \in \boldsymbol{O}$ is intended, given the input $\boldsymbol{x}_{\text{hmi}}$, using Bayes' rule:

$$P(O_k|\boldsymbol{x}_{\text{hmi}}) = \frac{P(\boldsymbol{x}_{\text{hmi}}|O_k)P(O_k)}{P_r + \sum_{O_{\text{j}} \in \boldsymbol{O}} P(\boldsymbol{x}_{\text{hmi}}|O_{\text{j}})P(O_{\text{j}})}, \tag{3.9}$$

where the joint probability $P_r = P(\boldsymbol{x}_{\text{hmi}}|O_{\text{none}})P(O_{\text{none}})$ estimates when the user does not want to grasp any object (class O_{none}). The prior probabilities $P(O_{\text{j}})$ indicate known preferences of the user, or can all be assumed to be equal as done in section 3.7.

The conditional probabilities $P(\boldsymbol{x}_{\text{hmi}}|O_k) \sim \mathcal{N}(\boldsymbol{x}_k, \boldsymbol{\Sigma})$ are modelled as multivariate Gaussian distributions with covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{N_x \times N_x}$ and mean $\boldsymbol{x}_k \in \mathbb{R}^{N_x}$, where $N_x = 2$ if the objects are assumed to be in a plane or $N_x = 3$ otherwise. The position $\boldsymbol{x}_k$ of object O_k is measured by a vision system. The covariance matrix $\boldsymbol{\Sigma}$ can be estimated from a dataset or, as we did in this work, by setting it to $\boldsymbol{\Sigma} = \sigma \boldsymbol{I}$ and heuristically selecting a convenient σ.

Let $d_m(\boldsymbol{x}_{\text{hmi}}) = \sqrt{(\boldsymbol{x}_{\text{hmi}} - \boldsymbol{x}_{\text{pred}})^{\intercal}\, \Sigma^{-1}\, (\boldsymbol{x}_{\text{hmi}} - \boldsymbol{x}_{\text{pred}})}$ denote the Mahalanobis distance, where $\boldsymbol{x}_{\text{pred}}$ is the position of the object corresponding to the maximum $P(\boldsymbol{x}_{\text{hmi}}|O_k)$. Then, we define $P(\boldsymbol{x}_{\text{hmi}}|O_{\text{none}})$ as follows:

$$P(\boldsymbol{x}_{\text{hmi}}|O_{\text{none}}) = \frac{d_{\text{sat}} - d_{\text{min}}}{d_{\text{max}} - d_{\text{min}}}, \tag{3.10}$$

which has two effects:

(i) it saturates $d_m(\boldsymbol{x}_{\text{hmi}})$ through $d_{\text{sat}} = \max(d_{\text{min}}, \min(d_{\text{max}}, d_m(\boldsymbol{x}_{\text{hmi}})))$ with d_{min} and d_{max} parameters; and

(ii) it performs a normalization such that $0 \leq P(\boldsymbol{x}_{\text{hmi}}|O_{\text{none}}) \leq 1$.

Because d_m^2 follows a χ^2-distribution with N_x DOF, a confidence interval $[P_{\text{min}}, P_{\text{max}}]$ that determines d_{min} and d_{max} can be selected instead.

As a last step, we select the most likely object and its corresponding position by maximizing the posterior probability of this choice. Fig. 3.6 shows a bidimensional example with three arbitrary objects, two of which are sufficiently close to affect the autonomy regions of each another.

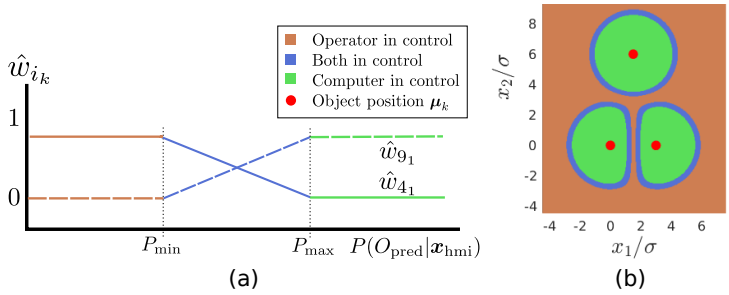


Figure 3.6. Example of regions depicting the levels of autonomy according to G_1, with $P_{\min} = 0.6$, $P_{\max} = 0.8$ and $P(O_{\text{none}}) = 0.5$. (a) Weight factors vs posterior probability, (b) Regions for three objects with $N_x = 2$ in dimensionless position coordinates $\boldsymbol{x}_{\text{hmi}}/\sigma$, where $\boldsymbol{x}_{\text{hmi}} = [x_1\ x_2]^\mathsf{T}$.

Further explanation of the intent estimation algorithm is given in Appendix A.2, including comparative plots that show the influence of each parameter.

3.7 Validation Use Cases

Because of the focus on adaptability and reconfigurability of our framework, we chose to demonstrate three widely differing use cases rather than focussing on a user study to show the suitability of a particular configuration to a user group with specific mobility limitations. The configurations shown in Table 3.3 were designed to demonstrate the customization capabilities of our framework to comply with various mobility limitations and scenarios. They were not designed to be the optimal for specific mobility limitations.

In all cases, an able-bodied user was chosen from the authors. An off-the-shelf vision system was used for determining positions $\boldsymbol{x}_k$ of taught objects O_k, and these were used as inputs for the intent-inference algorithm. Also, the force $\boldsymbol{F}_{\text{meas}}$ at the robot end effector was estimated by means of measured joint torques.

Fig. 3.7a shows snapshots of use case 3 and Fig. 3.8 shows relevant control signals that describe the obtained behavior. First, the user indicated the desired target with the fingertip, which was detected through the vision system. Then, from time instances *I* to *II*, the user pulled the position of the end effector $\boldsymbol{p}(\boldsymbol{q})$ away from the RVGF defined by path $\boldsymbol{f}_{\text{p}}$ in (3.5) by means of the HMI for kinesthetic guidance (RVGF versus full user control).

Table 3.3. Validation Use Cases. The use cases are numbered according to the use cases presented in the whole thesis, as explained in table 1.2

	Use Case 3	**Use Case 4**	**Use Case 5**
Target User	Able-bodied	Paraplegic	Tetraplegic
Robotic Platform	Franka Panda	Franka Panda	Kinova Gen 3
HMI for Teleoperation	None	SpaceMouse	None
HMI for kinesthetic guidance	Joint torque sensors	None	None
HMI for target indication	3D Camera: hand pointing	SpaceMouse	Eye tracker
HMI for events	Microphone: voice commands	Buttons	Eye tracker: blink detection
Tube Shape Source	Human demonstrations	Planned paths	Modeled paths

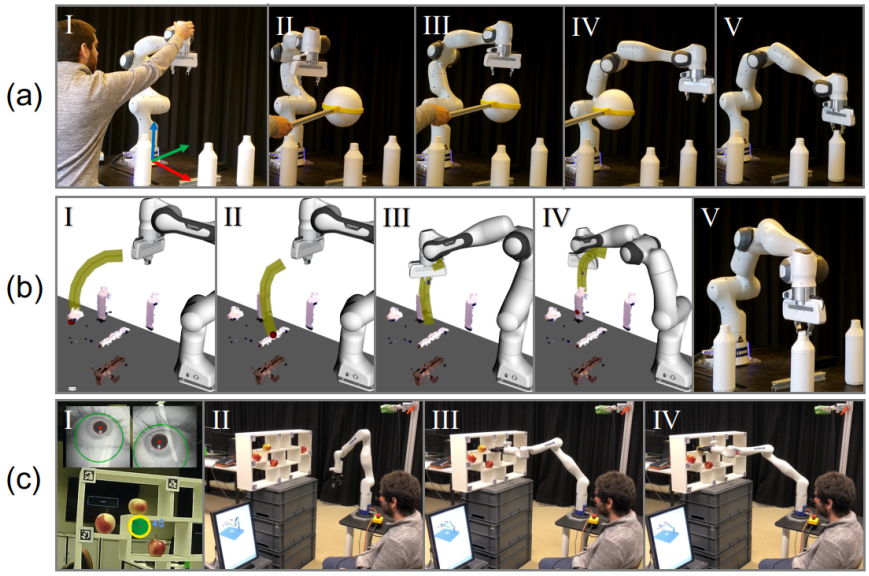


Figure 3.7. Snapshots of the use cases, whose target user and configuration is defined in Table 3.3. (a) Use case 3, (b) use case 4 and (c) use case 5.

At *II*, automatic motion is activated through a voice command after an obstacle is brought into the scene. From *II* the progress variable s advances with a quasi-trapezoidal velocity profile, making the orientation angle $f_{\hat{\theta}}(s)$ (see subsection 3.4.4) also advance. Meanwhile, $\boldsymbol{p}(\boldsymbol{q})$ slowly converges to $\boldsymbol{f}_{\mathrm{p}}$ following first order dynamics. At *III*, the minimum distance d_i from the vision-detected obstacle to the path reaches its threshold d_{th} (collision avoidance SIM). This causes path $\boldsymbol{f}_{\mathrm{p}}$ to adapt its shape to circumvent the obstacle. Then, at *IV*, the obstacle is retracted and $\boldsymbol{f}_{\mathrm{p}}$ returns elastically to match $\boldsymbol{f}_{\mathrm{lm}}$ (constraint 3) and grasps the intent-inferred object at *V*.

Fig. 3.7b shows snapshots of use case 4 and Fig. 3.9 shows relevant control signals that describe the obtained behavior. Two interaction modalities of the SpaceMouse, which were transitioned by pressing a button, were used to teleoperate the robot and to implicitly select the target object. The computer's autonomy over $\boldsymbol{f}_p|_{s=1}$ (indicated by $\hat{w}_{9_1}$) is continuously shifted between *I* to *II*, due to $\boldsymbol{x}_{\mathrm{hmi}}$ being driven by the user. The change of $\boldsymbol{x}_{\mathrm{hmi}}$ caused the shape of the RVGF to be adapted towards new targets by changing the latent variables $\boldsymbol{\chi}_{\mathrm{lv}}$ defined in (3.4). From *II* to *III*, the robot was teleoperated by the user who is assisted by the RVGF. At *III*, the user shifted $\boldsymbol{x}_{\mathrm{hmi}}$ which dragged away the RVGF from $\boldsymbol{p}(\boldsymbol{q})$, causing a shift towards full user control by means of the weight factors $\hat{w}_{11_3}$ and $\hat{w}_{12_3}$ (see G_3 signal explained in Table 3.2). The user freely teleoperated the robot until *IV*, where G_3 decreased to a low value, re-enabling the RVGF towards the grasp of the object at *V*.

Fig. 3.7c shows snapshots of use case 5, where a user is able to command the robot with a head-mounted eye-tracker [57]. In *I*, the user drives $\boldsymbol{x}_{\mathrm{hmi}}$ with the gaze (green circle), which is only updated when fixating on an area delimited by QR tags. The location and dimensions of that area were determined by the vision system, and the eye-tracker provided the gaze expressed in a local frame attached to the QR tags, allowing relocation of the user. After the user confirmed that the target was updated correctly through a screen, at *II* he activated the automatic motion with two consecutive forced blinks. At *III*, the robot reached the target location with an added offset, so that the user could correct if necessary. Finally, at *IV* the user commanded the grasp by a *long blink* gesture.

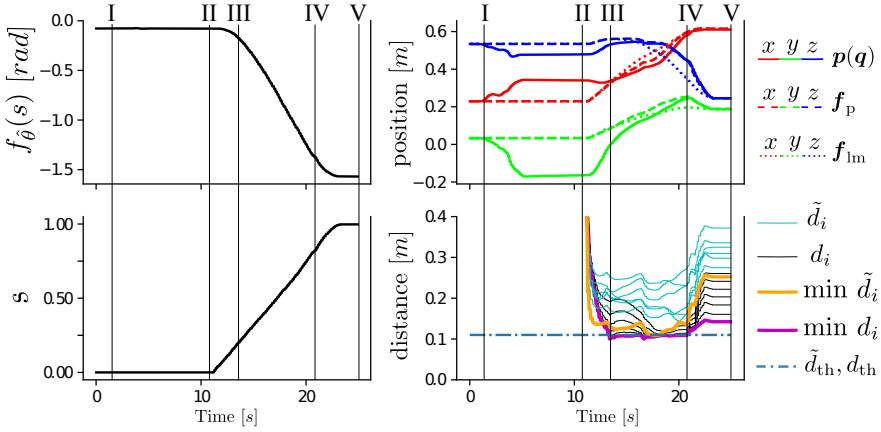


Figure 3.8. Control signals of an experiment performed for validation use case 3, where the time instances denoted with roman numbers correspond to the snapshots in Fig. 3.7a.

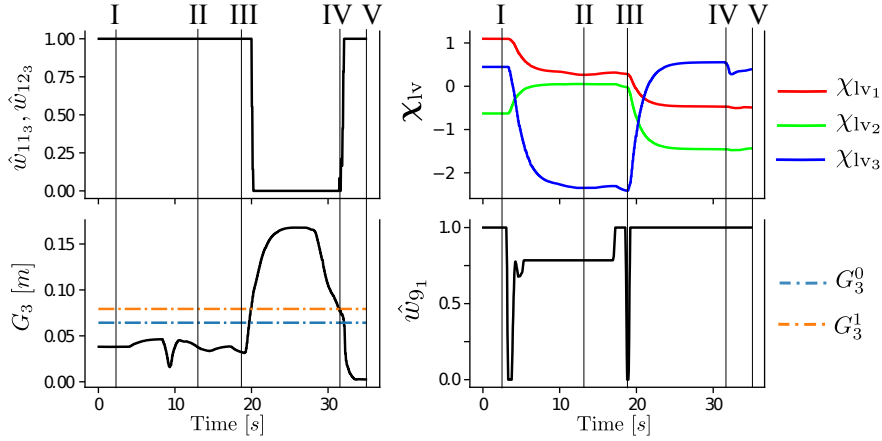


Figure 3.9. Control signals of an experiment performed for validation use case 4, where the time instances denoted with roman numbers correspond to the snapshots in Fig. 3.7b.

3.8 Discussion and Conclusion

This work provides a framework consisting of tools for defining adaptable reactive behaviors that allocate user or computer autonomy over robot-related or task-related variables. The modulation of the autonomy (i.e. arbitration) over each variable can be done systematically via two types of weight shifting, which enable online adaptation of the levels of autonomy. These modulation can be performed in function of relevant computer-controlled or user-controlled variables in a flexible and composable way.

During the validation use cases, the modularity and composability of the building blocks enabled the easy reconfiguration of the system to comply to the mobility limitations of able-bodied, paraplegic, and tetraplegic targeted users. This was achieved by separating *(i)* the sensor-interaction, specified using specific SIMs suited to different types of HMIs, from *(ii)* the specification of nominal actions and the deviation from these using the RAMs. The composition and modulation of these two types of building blocks results in the described levels of autonomy. This stands in contrast to the state of the art, where the resulting behavior that combines task and sensor-information is directly implemented in e.g. impedance-like behaviors, which makes it more difficult to reconfigure to new types of user input.

The used constraint-based approach also bears similarities with the autonomous dynamical systems described in [59] for human-robot interactive applications. For instance, in the case of the automatic reactive approach motion BB, the only explicit time dependency is contained within the weights, which are used to give the user the possibility of activating/deactivating its resulting behavior at arbitrary time instances. Once the activation has been made, the weights remain constant and the dynamical system becomes autonomous (i.e. time-invariant). This, in combination with other BBs such as the automatic orientation path, allows us to easily specify autonomous dynamical systems governing reactive behaviors in *SE(3)*, which are otherwise cumbersome to establish and to adapt to other applications.

The proposed intent-inference algorithm enables the use of HMIs for target indication with high uncertainty such as an eye-tracker. Its online inference capability leverages the reactiveness of our framework by allowing adaptation to changes of intention. The class O_{none} is explicitly included such that the user can automatically be granted with full autonomy over $\boldsymbol{f}_p|_{s=1}$ when, for example, the vision system fails to detect an object that

the user wants to reach. This would allow the user to still reach the goal at the expense of more effort.

The proposed modular specification of assistive tasks and arbitration of autonomy facilitates the potential combination with other intent estimation approaches based on HMI inputs for teleoperation, such as the ones proposed in [36], [55] and [75].

The reactiveness of our framework can also enable a safe teleoperation of the robot within an unstructured environment, adapting compliantly to dynamic obstacles and external forces. We are convinced that the offered reconfigurability could also be used for other tasks such as pouring, insertion, or completely different scenarios, such as teleoperation in hazardous environments or human-robot collaborative industrial applications. Future assistive devices could benefit from this work and impact the quality of life of people that require robotic assistance. Of course, this work needs to be followed up with a user study to evaluate the response and acceptance of subjects with limitations of their mobility. The offered reconfigurability and adaptability enables user-specific customization that is essential to obtain the acceptance of a larger user community.

Chapter 4

Rapid development of assistive applications through the use of reconfiguration: presenting two assistive scenarios

Abstract

This chapter summarizes the work performed in the context of two assistive scenarios. The applications related to both scenarios were developed by using the framework proposed in this thesis, and provide additional illustration of its capabilities. Both scenarios had specific challenges, including strict timing requirements for designing and commissioning the application, such that many of the features of the proposed framework could be exploited.

4.1 Introduction

The first assistive scenario, explained in section 4.2, took place in Gasthuisberg, at the University hospital UZ Leuven, in the laboratory for Neuro- and Psychophysiology, where prof. Peter Janssen and his team of researchers investigate which regions of the brain can provide useful information for robot control and how to decode this information from signals coming from invasive brain-machine interfaces (BMIs). A series of experiments is performed in which a trained non-human primate (henceforth referred to as the monkey) controls a robot arm using neural decodings. The success rates of the monkey reaching its goal are compared for the different proposed assistive strategies.

The second assistive scenario, explained in section 4.3, took place in Oberpfaffenhofen, Germany, at the Institute of Robotics and Mechatronics of the German Aerospace Center (DLR). Here the EDAN robotic research platform [103] was designed and manufactured. This robotic platform consists of an 8-DOF robot arm mounted on a 4-DOF wheelchair (one DOF is used to adjust the height of the wheelchair). Through the reconfiguration capabilities that our framework offers, a new assistive strategy was developed in a short time and tested using EDAN.

4.2 Assistive Scenario 1: assistance for controlling a robot arm with monkey users through brain-decoded signals

This section shows how the proposed framework can be used in a first assistive scenario using a brain-machine interface (BMI) to control a robot arm. Two use cases, each with its own interaction modality and a compatible assistive strategy, were developed and deployed.

4.2.1 Research context and challenges

The assistive scenario presented in this section took place in the laboratory of Neuro- and Psychophysiology, at the university hospital UZ Leuven in Gasthuisberg in the context of the interdisciplinary KU Leuven C1 project “Neurophysiological investigations of the human, and nonhuman

primate brain: from perception to action" (3M180301). In this project, a multidisciplinary group of researchers lead in the different fields by prof. Peter Janssen (neurophysiology), prof. Tom Theys (neurosurery), prof. Joris De Schutter (robotics) and Dr. Erwin Aertbeliën (robotics), worked together to investigate how brain-machine interfaces (BMIs) can be used to decode visual and motor information that is useful to control assistive robotic arms. The BMI-related aspects of the project, including implantation, decodings, and monkey-related experimental aspects were handled by our project partners.

The research performed in this assistive scenario had strict development and reconfiguration timing challenges, due to the following reasons:

1. Initially, the capabilities of a given implant were uncertain and were rapidly deteriorating over time. The set-up and application needed to be quickly adapted to the capabilities. On the robotics side, this allowed us to test out and demonstrate the reconfiguration ability of the framework under stringent deadlines. In more recent work, the capabilities of the implants became more stable and predictable.

2. the assistive strategy needed to work properly and be ready to be reconfigured as soon as the monkey was trained to perform the tasks as required in the use case.

In the project experiments were also planned to be conducted in human patients with tetraplegia and epilepsy. Such experiments had even more strict timings since the availability of patients is quite limited and a lot of reconfiguration needs to be performed in situ. Unfortunately, these were canceled because of the Covid-19 pandemic. Therefore, all experiments were done in monkeys and had the following limitations:

1. Training a monkey to perform a task is more time-consuming and can only be done for simple and repetitive tasks. Making modifications to the task and re-training the monkey is therefore more difficult than just verbally instructing the human.

2. No subjective qualitative data (e.g questionnaires) can result from the experiments with monkeys. These results would have been very relevant since, among other reasons, they can be interpreted to reconfigurations that would improve the acceptance and the performance of the user.

3. Compared to the monkey's own manual grasping motions, the robot set-up had significant velocity limitations. Therefore, the physical robot could not be used and a virtual set-up was used in the second use case.

Taking into account these challenges and limitations, we conducted experiments in two distinct use cases that significantly benefited from the proposed framework.

4.2.2 BMI-based robot control: an overview

A Brain-Machine interface (BMI), also known as Brain-Computer interface (BCI), is an HMI that is used to measure neural activity in certain regions of the brain, which is further translated (i.e. decoded) into interpretable data. Such BMIs can be non-invasive, meaning that signals are measured externally with a wearable device on the user's head, or invasive, meaning that they require a surgical implantation.

Several researchers have developed methods to decode signals from non-invasive BMIs based on electroencephalography (EEG) and then use these signals to control wheelchairs (e.g. [29]) and robotic arms (e.g. [64], [61]). Despite their promising results, it is well known that non-invasive EEG yield lower performance compared to invasive BMIs [104].

For this reason, several researchers have alternatively investigated how invasive BMIs can be used by people with severe motor impairments (e.g. people with tetraplegia) to control robotic devices. For example, in [48], researchers managed to decode three-dimensional linear velocities and one-dimensional grasping command and use them to control a robotic arm to perform tasks such as drinking from a bottle. Furthermore, in [26], the researchers decoded a seven-dimensional velocity vector from an invasive BMI that allow to control the position, the orientation, and the gripper aperture of a robot arm. In order to improve the performance, in [33], the researchers added vision-based shared autonomy so that the system assisted the user to reach a stable grasping pose.

In this section, we show how the framework proposed in this thesis can be used in order to control a robot arm from an invasive BMI. The invasive BMI was surgically implanted in a monkey[1] in the dorsal premotor cortex (PMd), which is a region of the brain involved on the motion planning and the replanning done when online changes of intention occur [80]. Two decoding

algorithms were developed by the team lead by prof. Janssen, each being able to decode the following information from neural activity:

- *target object*: the index of an object (i.e. 0, 1 or 2) indicating the desired object to be grasped (UC6).
- *desired velocities*: continuous two-dimensional desired end effector velocities (UC7).

Each type of decoded information is interpreted as a different interaction modality and sent to the controller that runs the assistive strategy, as shown in Fig. 4.1. When decoding desired velocities, these velocities were integrated so that desired positions were sent instead to the assistive strategy block. Each use case (UC6 and UC7) exploit a different decoded information as summarized in Table 4.1. The details of each use case are discussed later on in this section.

The technical and medical details about the BMI decoding are omitted from this thesis since they are currently confidential due to a future publication under preparation in collaboration with all the involved research partners.

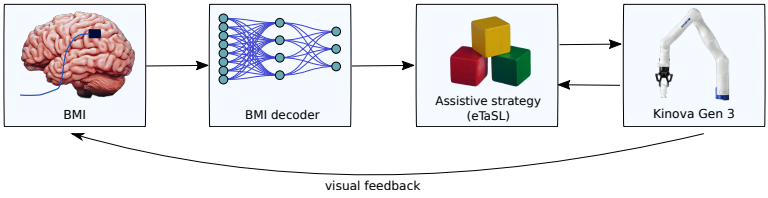


Figure 4.1. Control scheme used in the use cases related to the assistive scenario 1 with the brain-machine interface. A data-driven BMI decoder takes the measured neural activity in the dorsal premotor cortex and sends the decoded information to the eTaSL constraint-based assistive strategy block. The monkey observes the executed commands in the virtual/real robot and corrects if necessary, closing the loop via visual feedback.

4.2.3 Assistive strategies for BMI decodings

In this subsection the assistive strategies used for UC6 and UC7 are explained. Such assistive strategies were developed by re-using specific

[1] All BMI experiments conducted were strictly revised and approved by the Ethics committee of KU Leuven, and performed by experts at the university hospital. The author of this thesis developed the robot-related aspects of the project and was involved in the integration of the robot with the decoding experimental setups.

Table 4.1. Comparison between the two use cases (UC) related to the BMI decodings. The training phase refers to the process of data recording and training of the data-driven decoder algorithm, while the execution phase refers to the task execution phase with the robot using the trained decoder.

	UC6	**UC7**
Decoded information	Index of a target object (i.e. 0, 1 or 2) whose position $\boldsymbol{x}_{\text{target}}$ is known beforehand.	Desired linear velocity of the end effector in $\mathbb{R}^2$ (in a plane), which is then integrated to obtain desired position $\boldsymbol{x}_{\text{bmi}}$.
Robot arm	Real Kinova Gen 3.	Virtual (simulated) Kinova Gen 3.
Training phase	The monkey grasps and pulls the object whose LED light indicator blinks. Brain signals were recorded and labeled according to the grasped object. The monkey receives a reward (i.e. juice) if it grasped and pulled the indicated object (see Fig. 4.2.a).	The monkey passively observes a simulated robot executing pre-programmed trajectories towards different virtual objects on a screen.
Execution phase	The training phase procedure is repeated, but this time the monkey's hand is restricted. Instead, the monkey fixates its attention on the object indicated through the blinking target LED and, if the correct object was grasped by the robot arm (see Fig. 4.2.b), the monkey received a reward.	The procedure of the training phase is repeated, but this time the monkey has control over the virtual robot's end effector in a 2D plane. The monkey is rewarded if the end effector reached the indicated goal.
Assistive strategy	Autonomous motion towards the decoded object.	Tube-shaped RVGF (see section 3.4.5) generated towards locations that were known beforehand, since these were visually indicated to the monkey (more details in section 4.2.4).
Speed of motion	Trajectories with medium speed, constrained to conservative joint velocity limits and low control gains.	High allowed speed due to the increment of joint velocity limits and higher control gains. This was necessary since the monkey lost attention if the goal was not reached rapidly and, thus, a virtual robot was used instead of a real robot.
Visual feedback	The monkey observes the robot going to certain object and, via visual feedback, is able to notice mistakes. The monkey can change/correct the target object during the execution and the robot is able to adapt its trajectory online.	The monkey observes the robot continuously moving towards a target and, via visual feedback, closes the control loop to correct the generated BMI commands.

building blocks (BBs) described in chapter 3, demonstrating again the reconfigurability capabilities of the proposed framework.

Table 4.2 shows the BBs used for creating an assistive strategy to handle the target object BMI decodings in use case 6. The assistance provided

by this strategy is based on the automatic approach motion described in chapter 3, since the BBs used for this assistive strategy are the same as the ones used for use case 5. However, in this use case there are no adaptable levels of autonomy (i.e. all weights of the constraints have fixed values). Since the positions of all potential target objects are known beforehand, the known position associated to the decoded target object $\boldsymbol{x}_{\text{target}}$ is used to constrain the end position $\boldsymbol{f}_{\text{lm}}|_{s=1}$ of a learned trajectory $\boldsymbol{f}_{\text{lm}}$. Such trajectory $\boldsymbol{f}_{\text{lm}}$ is learned through PPCA and represented with (3.4). As explained in section 2.3, the latent variables $\boldsymbol{\chi}_{\text{lv}}$ are modeled in eTaSL as controlled feature variables, allowing adaptation of the trajectory due to online changes in intention.

$$\boldsymbol{f}_{\text{lm}}(s, \boldsymbol{\chi}_{\text{lv}}) = \boldsymbol{W}(s)\boldsymbol{\chi}_{\text{lv}} + \boldsymbol{b}(s) \quad (3.4)$$

On the other hand, Table 4.3 shows the BBs used for creating an assistive strategy to handle the desired velocity BMI decodings in use case 7. The assistance provided by this assistive strategy consist on the tube-shaped RVGF explained in chapter 3, but with no adaptable levels of autonomy (i.e. all weights of the constraints have fixed values). Instead of constraining the velocity of the end effector, as done in previous similar use cases, constraint 1 enforces the end effector to follow integrated desired positions $\boldsymbol{x}_{\text{bmi}}$ that come from the decoded desired velocities (as explained in subsection 4.2.2).

Table 4.2. Specification of building blocks (BBs) that are used to generate the automatic approach motion for use case UC6. Hard constraints are indicated by the omission of $\hat{w}_i$.

#	Expression e_i	Params	Constraints
1)	$\boldsymbol{f}_{\text{lm}}\|_{s=1} - \boldsymbol{x}_{\text{target}}$	$\hat{w}_1$ $k_1 > 0$	$\boldsymbol{e}_1 \to \boldsymbol{0}$: RAM, end point of trajectory towards target object.
2)	$\boldsymbol{\chi}_{\text{lv}}$	$\hat{w}_2$	RAM, reactive nominal path (3.4) $\dot{\boldsymbol{e}}_2 \to \boldsymbol{0}$: only adapt shape when needed.
3)	s	$k_3 > 0$	$0 \leq e_3 \leq 1$: limits on the progress variable s.
4)	$\hat{\boldsymbol{n}}_R$	$\hat{w}_4$ $k_4 > 0$	$\boldsymbol{e}_4 \to \boldsymbol{0}$: RAM, automatic orientation path.
5)	s	$\hat{w}_5$	RAM, Auto. reactive approach motion $\dot{e}_5 \to V_p(s)$: follow a desired velocity along the path.
6)	$\boldsymbol{p}(\boldsymbol{q}) - \boldsymbol{f}_{\text{lm}}(s)$	$\hat{w}_6$ $k_6 > 0$	$\boldsymbol{e}_6 \to \boldsymbol{0}$: position of the end effector $\boldsymbol{p}(\boldsymbol{q})$ follows path.
7)	$\boldsymbol{q}$	$k_7 > 0$	RAM, joint limits $\boldsymbol{q}_{\text{min}} \leq \boldsymbol{e}_7 \leq \boldsymbol{q}_{\text{max}}$: position limits.
8)	$\boldsymbol{q}$	$k_8 > 0$	$\dot{\boldsymbol{q}}_{\text{min}} \leq \dot{\boldsymbol{e}}_8 \leq \dot{\boldsymbol{q}}_{\text{max}}$: velocity limits.

Table 4.3. Specification of building blocks (BBs) that are used to generate the tube-shaped RVGF for use case UC7. Hard constraints are indicated by the omission of $\hat{w}_i$.

#	Expression e_i	Params	Constraints
1)	$\boldsymbol{p}(\boldsymbol{q})$	$\hat{w}_1$ $k_1 > 0$	$\boldsymbol{e}_1 \to \boldsymbol{x}_{\text{bmi}}$: SIM, end effector position control.
2)	$\boldsymbol{\chi}_{\text{lv}}$	$\hat{w}_2$	RAM, reactive nominal path (3.4) $\dot{\boldsymbol{e}}_2 \to \boldsymbol{0}$: only adapt shape when needed.
3)	s	$k_3 > 0$	$0 \leq e_3 \leq 1$: limits on the progress.
4)	$\hat{\boldsymbol{n}}_R$	$\hat{w}_4$ $k_4 > 0$	$\boldsymbol{e}_4 \to \boldsymbol{0}$: RAM, automatic orientation path.
5)	$\|\boldsymbol{p}(\boldsymbol{q}) - \boldsymbol{f}_{\text{lm}}\|$	$\hat{w}_5$ $k_5 > 0$	RAM, RVGF $e_5 \leq r_{\text{tube}}(s)$: remain in tube.
6)	s	$\hat{w}_6$	$\dot{e}_6 \to 0$: dampen s.
7)	$\boldsymbol{f}_{\text{lm}}\|_{s=1} - \boldsymbol{x}_{\text{target}}$	$\hat{w}_7$ $k_7 > 0$	$\boldsymbol{e}_7 \to \boldsymbol{0}$: RAM, trajectory towards target object.
8)	$\boldsymbol{q}$	$k_8 > 0$	RAM, joint limits $\boldsymbol{q}_{\text{min}} \leq \boldsymbol{e}_8 \leq \boldsymbol{q}_{\text{max}}$: position limits.
9)	$\boldsymbol{q}$	$k_9 > 0$	$\dot{\boldsymbol{q}}_{\text{min}} \leq \dot{\boldsymbol{e}}_9 \leq \dot{\boldsymbol{q}}_{\text{max}}$: velocity limits.

4.2.4 Implementation and results

The previously described assistive strategies, and their corresponding BMI decodings, were implemented and tested with a monkey as a substitute of a human user. Both use cases are described and compared in Table 4.1.

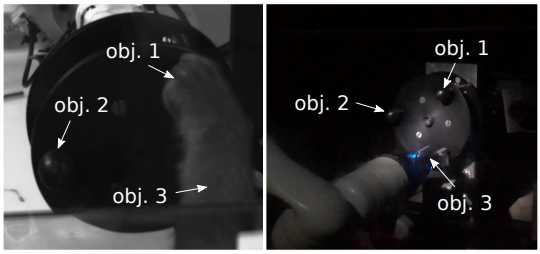


Figure 4.2. Photos of the setup used for decodings of the target object. The setup consists of three spherical objects (obj) which can be pulled (with a spring) by the monkey. a) Picture of the monkey pulling one of the spherical objects with its hand during the training phase of the data-driven decoder, and b) picture of the robot grasping and pulling the decoded object during the execution phase.

In both use cases, a *trial* refers to all the experimental steps involved in each use case for the monkey to complete the goal once. A *trial session* refers to the collection of trials performed in a single day without long interruptions.

Fig. 4.3 shows the snapshots of a single trial of use case 6, where a real Kinova Gen 3 was installed in a dark room where the monkey was only able to observe the illuminated robot and three spherical objects. At *I*, the only visible element is a green cue LED light, which indicates the monkey the start of the trial. At *II* a target object is indicated to the monkey through a blue target LED light and makes the monkey focus on this object. The robot approaches towards the target object in *III* and *IV*, and in *V* it finally reaches it and grasps it. Then the robot is sent towards the home position at *VI*, where it waits until the next trial begins. Several trials with pseudo-random target objects were made to collect data that is useful only for neurophysiological studies, and thus it is not reported in this thesis.

On the other hand, for use case 7, continuous two-dimensional velocities were decoded from the BMI and used to control the position of the end effector of the robot, with the assistive strategy specified in Table 4.3. For this use case, the automatic orientation path was kept constant. In addition, the radius of the tube-shaped RVGF follows a cubic polynomial that starts

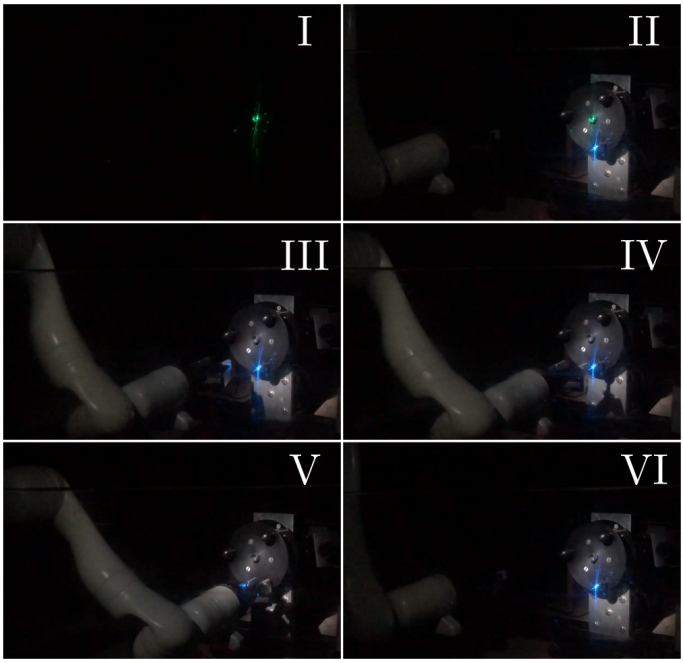


Figure 4.3. Snapshots of a trial of UC6, where the target object is decoded from the monkey's neural activity through the BMI, and subsequently the robot executes a trajectory to grasp it. The cue LED (green) indicates the monkey the start of the trial and the target LED (blue) indicates which object the monkey needs to focus on. The room is dark to help the monkey to stay focused during the trials.

with a value of 0.3 [m] when $s = 0$ and finishes with a value of zero when $s = 1$:

The shape of the nominal path $\boldsymbol{f}_p$ was learned with PPCA using straight-line demonstrations from the center towards the target. This was done under the assumption that the monkey wants to follow an optimal trajectory with minimum traveled distance.

The experimental setup consisted of the monkey observing at a screen, where a virtual robot is controlled through the BMI decodings. Each trial follows a conventional procedure known in neuroscience as center-out tasks [44] and consists of the following steps:

1. the end effector of the robot starts in the center of a virtual vertical plane while a yellow cue indicator blinks at this same position to make the monkey focus at this point (see Fig. 4.4).

2. the yellow cue indicator disappears and another orange target indicator that makes the monkey realize the desired target position appears.
3. the robot execution starts and the monkey is allowed to control the robot arm with the BMI decodings, while the monkey closes the control loop via visual feedback.
4. the trial stops and the monkey is rewarded with juice if the trial was successful.
5. the robot resets to its home position.

Each trial was stopped and classified as successful or unsuccessful when one of the following conditions was met:

- *successful:* the end effector of the robot reached a distance less than 6 cm from the target position and remained below this threshold for 500 milliseconds.
- *unsuccessful:* 7.5 seconds elapsed and no classification as successful occurred.

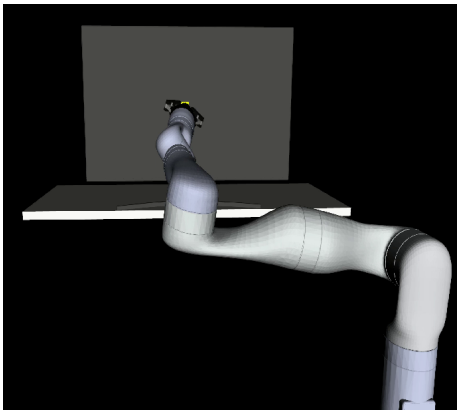


Figure 4.4. Snapshot of what the monkey observes through the screen at the beginning of each trial of use case 7, where the end effector of the robot starts resting in the center of a virtual vertical plane while a yellow cue indicator blinks at this same position.

Two groups of trials were carried out in order to assess the influence of the assistance provided by the tube-shaped RVGF on the success rate. The first

group, referred to as trials *without assistance*, were executed by setting $\hat{\omega}_5 = 0$, such that tube-shaped RVGF has no influence on the motion of the end effector. The second group, referred to as trials *with assistance*, were executed by setting $\hat{\omega}_5 = 5\hat{\omega}_1$ (only the relative weight between conflicting constraints 1 and 5 is relevant).

Fig. 4.5 and Fig. 4.6 show snapshots of three trials without assistance and with assistance, respectively. It can be observed that in the trials with assistance, the executed trajectories resemble more straight lines than in the trials without assistance, where in the latter the monkey seems to have more difficulty in reaching the goal. While it is an assumption that the monkey wants to follow straight lines (which are optimal trajectories with minimum traveled distance), it is clear that going towards the opposite direction of the target at any moment is never desirable to complete the goal. This is observed in a trial without assistance in Fig. 4.5 (target 3) between *I* and *II*.

Despite that the tube-shaped RVGF seems to enforce that the end effector remains in a certain region of the workspace, it is not the case. This is because, instead of constraining the velocity of the end effector as done in chapter 3, this use case relies on constraint 1 that enforces the end effector to follow a desired position $\boldsymbol{x}_{\text{bmi}}$. This desired position is obtained by integrating the decoded desired velocity. Thus, even if the desired velocity is bounded, the desired position $\boldsymbol{x}_{\text{bmi}}$ is not bounded. In [99] it is demonstrated that, when having two position constraints that are in conflict (such as constraint 1 and constraint 5), the resulting position is a weighted average. This makes the autonomy of the monkey greater since the assistance becomes less restrictive, enabling it to go far from the boundaries of the tube if desired, and the BMI decodings can be studied in a better way. Once again the reconfiguration capabilities of the framework were demonstrated for customization, taking into account the needs of the neurophysiologist researchers.

Several trials of the monkey controlling the virtual robot were performed, both with and without assistance. Table 4.4 shows the data collected during these trials and Fig. 4.7 shows corresponding comparative plots.

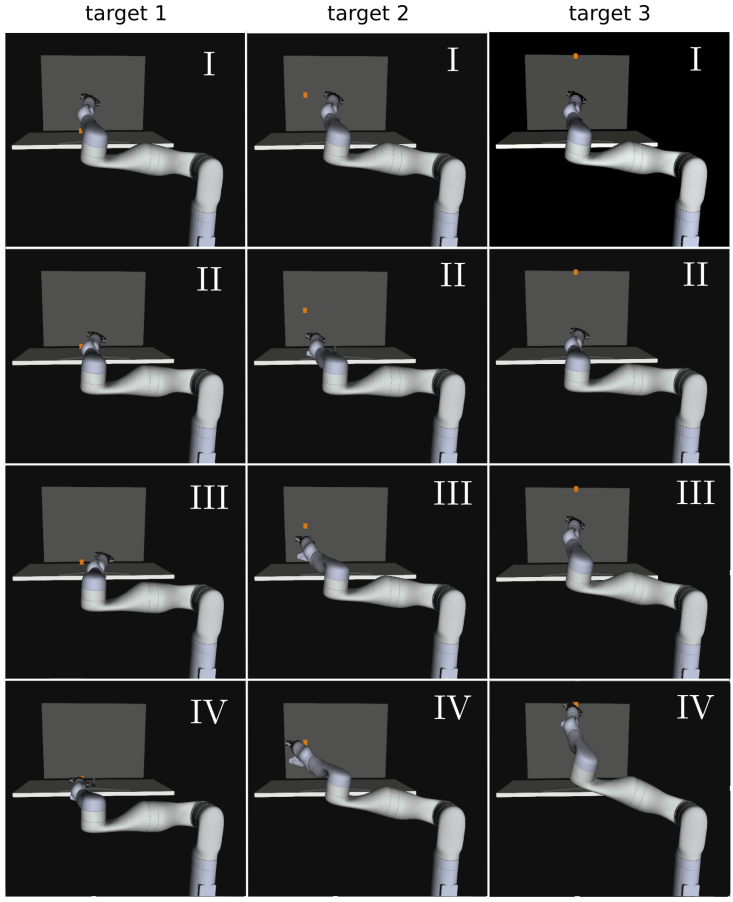


Figure 4.5. Snapshots of reaching three targets during three trials of UC7 without the assistance provided by the RVGF.

Table 4.4. Success rate and number of trials for the experiments done without assistance (WOA) and with assistance (WA). Not all trial sessions had the same number of trials since they needed to be interrupted due to the monkey's temper or lost of focus due to fatigue.

	1	**2**	**3**	**4**	**5**	**6**	**7**
success rate WOA	56.3%	39.3%	49.3%	68.3%	44%	54.7%	56.5%
number of trials WOA	105	86	112	105	93	101	108
success rate WA	91.2%	86.5%	92.6%	97.4%	73.3%	89.6%	97.3%
number of trials WA	120	105	108	110	82	108	107

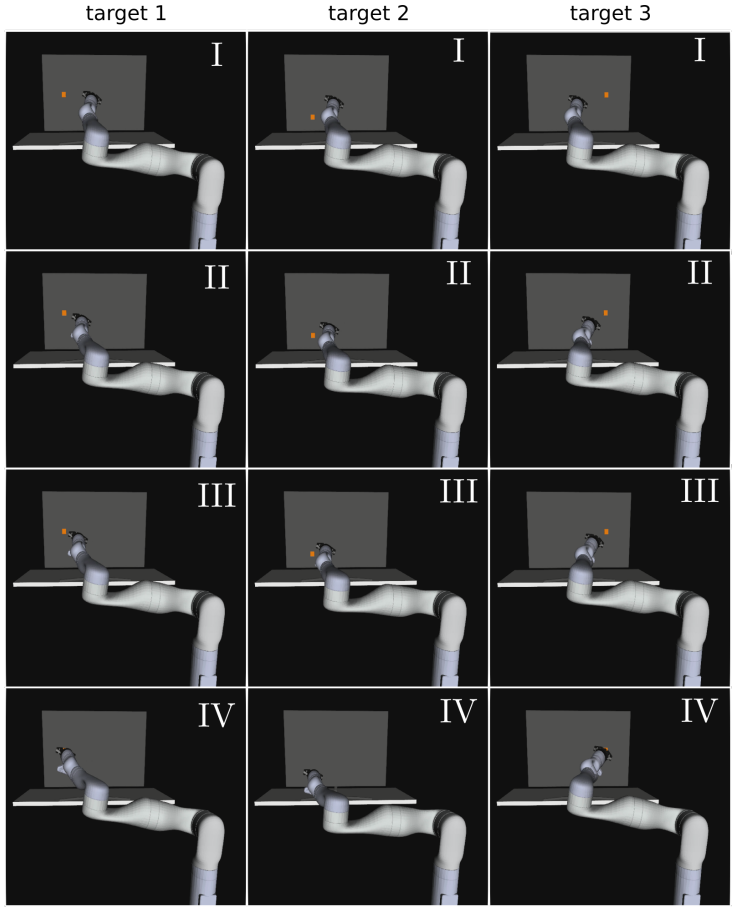


Figure 4.6. Snapshots of reaching three targets during three trials of UC7 with the assistance provided by the RVGF.

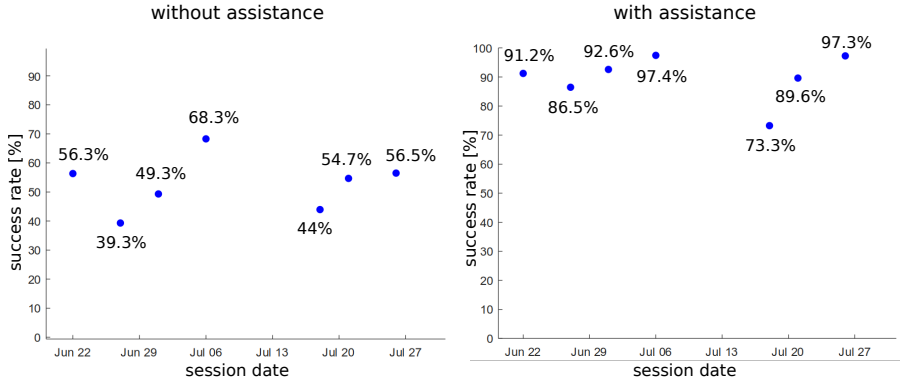


Figure 4.7. Plots comparing the success rate of the 7 trial sessions without and with assistance.

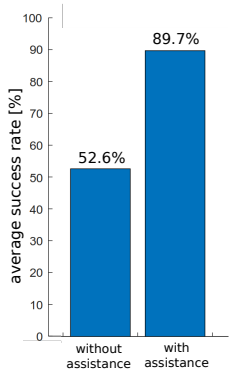


Figure 4.8. Average of the success rates of all trial sessions, were each session has the same weight regardless of the number of trials

4.2.5 Discussion and conclusions

For this assistive scenario we demonstrated the capabilities of our framework by developing and deploying two use cases for BMI applications in laboratory conditions (low TRL). Each use case has its own assistive strategy and interaction modality, and tackles the challenges involved in this assistive scenario. The used BMI was surgically implanted in a monkey, whose neural activity was used to decode relevant information for controlling the robot in each use case. Since the experiments related to BMI had strict development timings, the framework allowed us to rapidly reconfigure the corresponding assistive strategies based on the changing requirements.

During the first use case (UC6) the index of a target object (whose position was known beforehand) was decoded from the BMI. Subsequently, an autonomous motion, which could adapt online its nominal path to changes of intention, was generated and executed. Such nominal path was specified through demonstrations and generalized through the use of PPCA. We showcased the use case with a real Kinova Gen 3 robot grasping the object that was decoded from the monkey's neural activity.

The first use case allowed the team to understand better the BMI decodings and develop a second use case (UC7) that gave more autonomy to the user. In this second use case, continuous desired velocities were decoded and used to control a virtual robot arm in 2D. Quantitative results were obtained and compared when executing the task with and without the assistance

based on tube-shaped RVGF. The results clearly show that the assistance improves the success rate of the monkey (see Fig. 4.8), while still giving the monkey autonomy to reach the desired position or divert from it.

In this second use case, the tube-shaped RVGF was generated towards the target goal, which was known beforehand since it was indicated to the monkey. Nevertheless, in practice, the target object is not known but needs to be predicted. Such prediction can be done in different ways by exploiting parallel co-existing interaction modalities coming from the BMI. A first option is to decode the target object directly from the BMI (as done in the first use case), with the limitation that this only works in very structured environments (always with the same objects). A second (more general) approach is to also decode (if possible) target positions, while, in combination with the target estimation (section 3.6) and BBs used in use case 5 with the eye tracker, the end position of the tube could be attracted to such inferred targets based on the posterior probability. A third approach is estimating the end position of the trajectory based on the executed motion, assuming a motion model, as done in [30] (master thesis supervised by the author of this thesis). Exploring these options would yield to an assistance that is more applicable to activities of daily living.

The assistive strategies developed to suit both BMI decodings showed promising results with the use of the monkey, even with these limitations. Thus, we expect that these methods are intuitive (e.g. no complicated explanation of how to use it is needed) and therefore can be easily accepted by human end users. Experimental support for claiming this is still needed and is encouraged for future work.

Acknowledgements

The author of this thesis gratefully acknowledges the help and the contributions performed by prof. Peter Janssen, Sofie De Schrijver, Jesus Garcia and Celia Bougou for their help and their contributions and their collaboration during the BMI experiments and for providing relevant experimental results.

This work was supported by the KU Leuven C1 Project “Neurophysiological investigations of the human, and nonhuman primate brain: from perception to action" (3M180301)

4.3 Assistive Scenario 2: assistance for wheelchair-mounted robot arm EDAN

In Oberpfaffenhofen, Germany, the second assistive scenario was performed in collaboration with members of the *re-enabling Robotics Team* of DLR. This work was done in the context of a one-month research visit at the DLR, which was carried out by the author of this thesis. The DLR researchers provided the EDAN robotic research platform and the corresponding software packages (implemented in links-and-nodes [89]) and kindly contributed to supervise and help the author of this thesis. In addition, the contribution of the following aspects is shared between the DLR team members and the author of this thesis:

- the ideation of the proposed assistive strategy,
- the used intent estimation algorithm (shared between Samuel Bustamante from DLR and the author of this thesis) and its implementation (done by Samuel Bustamante),
- the experimental validation of the methods.

During the one-month research visit, the author of this thesis:

- made all the implementations related to the assistive strategies using the framework proposed in this thesis,
- developed the EDAN C++ driver that interfaces the framework with EDAN (i.e. a bridge between eTaSL and links-and-nodes),
- developed and implemented the control modes that characterize the different use cases.

Erwin Aertbeliën and Joris De Schutter remotely supervised the work and contributed to the formalization of the control modes.

4.3.1 Research context and challenges

The purpose of this section is to demonstrate once more the applicability of our framework to rapidly develop and deploy an assistive application in a

different assistive scenario. This time the focus is on the assistance during tasks that involve reaching objects that are placed in a grid-like pattern on a flat surface. Activities of daily living that involve these tasks are commonly found. Examples include: ringing a doorbell from a doorbell panel of a building, pressing buttons inside an elevator, or grabbing a beer from a crate. Fig. 4.9 shows two examples of tasks that were tackled in this work. Nevertheless, there is no methodological difference between both and the implementation of the task shown in Fig. 4.9.a is omitted from this text.

The described tasks were chosen because DLR researchers went to the homes of few end users (i.e. with real motor impairments) and identified that these users frequently encountered these type of tasks and required assistance to perform them. Based on observations and conversations with the users, the DLR researchers gathered some requirements to provide effective assistance to the users. Therefore, we used these gathered requirements as input to develop and deploy a full user-centered integrated solution using the framework proposed in this thesis. Since this thesis focuses on user-centered assistance, identifying requirements that are associated to real end users was extremely valuable. The development of the different components related to the design workflow proposed in this thesis, which starts by inquiring user requirements, is discussed throughout this section. Later in the conclusions section, the connection between such components and the proposed workflow is explicitly discussed.

One of the contributions of this section is to demonstrate that our framework is also applicable to wheelchair-mounted robots such as EDAN [103]. This particular robotic platform consists of an 8-DOF robot arm mounted on a 4-DOF wheelchair. While previous use cases focused on standalone robot arms, we explain how assistive strategies can be executed in wheelchair-mounted robots with minimal effort. Using a robotic device such as EDAN is much closer to fulfill end user requirements and demonstrates a higher TRL [50] than the previous use cases. Other wheelchair-mounted robots are commercially available and clinico-economic and social impact of these devices has been studied and assessed. For the commercial wheelchair-mounted robot arms iARM and JACO, such studies were performed in [85] and [68], respectively.

Thanks to the reconfiguration capabilities of the developed framework, all the results presented in this section could be obtained during a short research visit of one month.

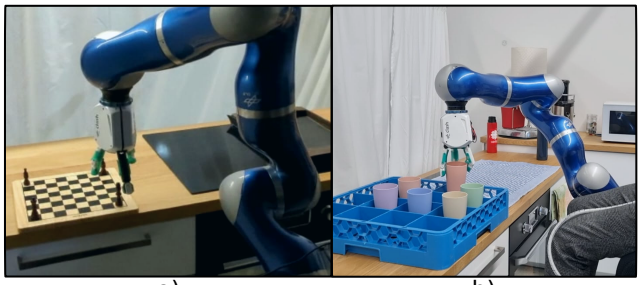


Figure 4.9. Two examples of tasks that are tackled within the assistive scenario 2 through the use of the EDAN platform. Both tasks involve the pick and place of objects which are placed with a grid-like pattern on a flat surface. a) pick and place operations during a chessboard game, and b) pick and place of cups located inside a grid-like storing crate.

4.3.2 Assistive strategy

In this assistive scenario, a new method based on virtual fixtures assists the users by guiding them towards target objects. These virtual fixtures are refered to as *GET-ME* virtual fixtures, which is an acronym of "Guiding with Escapable and Teleportable Mecanically Elastic" virtual fixtures. These virtual fixtures are showcased using a task that involves pick-and-place operations of objects arranged in a grid-like configuration (see Fig. 4.9).

The GET-ME virtual fixtures share several properties with the tube-shaped virtual fixtures described in chapter 3. The framework presented in chapter 3 is flexible enough to reconfigure the building blocks to another assistive strategy with novel features.

The following similarities between both types of virtual fixtures are highlighted:

- They both guide the user while controlling the robot to reach a certain object/location.
- Inside the volume of both virtual fixtures, the user is able to move the robot freely, while outside the volume a spring-like behavior (i.e. virtual impedance) acts to bring the end effector back towards the interior.
- They both can deal with changes of intention of the user as they are able to change the guidance towards new objects during execution.

- The both allow the user to escape from them, meaning that if the user pushes hard enough to get outside, they will deactivate the spring-like behavior. If the user approaches closely enough, then the spring-like behavior is re-activated.

The main differences between both types of fixtures are:

- While the virtual fixtures presented in chapter 3 are able to continuously adapt their shape (e.g. to guide towards changing targets), the GET-ME virtual fixtures maintain their shape constant.
- While the virtual fixtures continuously adapt following first order dynamics, the GET-ME virtual fixtures change position instantaneously (i.e. teleport) towards new intent-inferred target. This does not cause stability issues since they can only teleport after they have been deactivated (more details are given later).

Fig. 4.10 explains how a user can escape from the GET-ME virtual fixture that is attempting to provide guidance towards a first object and then gets trapped by another one that is generated to guide the user towards a second object. At instant *I* the end effector (blue sphere) starts inside the virtual fixture (depicted in yellow), where there is no impedance acting upon the end effector. After this instant, assume that the user starts commanding the robot to move to the right with a constant input velocity resulting in a velocity of the end effector that is illustrated by the black vector. At instant *II* the end effector reaches a spring-like impedance region (depicted in light red) that pulls the end effector towards the interior and therefore causes the end effector to continuously slow down. As soon as the end effector reaches the border of the impedance region at instant *III*, the impedance of the virtual fixture gets deactivated and the user is able to control the end effector freely, without any guidance. At this stage the virtual fixture is allowed to continuously teleport towards new targets that are selected by an online intent estimation algorithm. When the end effector is close enough, it gets trapped again by a virtual fixture at instant *IV*.

The shape of this virtual fixtures, like in the ones presented in chapter 3, is defined through a nominal path $\boldsymbol{f}_{\mathrm{p}}(s)$ and a radius $r_{\mathrm{tube}}(s)$ along that path, both parametrized with respect to the normalized progress variable s. Instead of learning $\boldsymbol{f}_{\mathrm{p}}(s)$ from demonstrations, in this use case it was sufficient to define it as a straight line. Thus, just the start position $\boldsymbol{f}_p|_{s=0}$ and the final position $\boldsymbol{f}_p|_{s=1}$ (selected as the inferred target position) need to be selected.

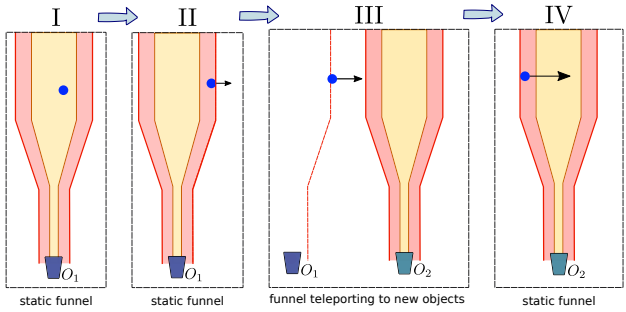


Figure 4.10. 2D scheme that illustrates how the user can interact with the GET-ME virtual fixture to switch from a first object (O_1) to a second object (O_2). The free-motion volume inside the GET-ME virtual fixture is illustrated in light yellow, the impedance region that guides the user is illustrated in light red and the end effector of the robot is represented with a blue sphere. The black vector represents the velocity (direction and magnitude) of the end effector at the different stages, given that the user is giving an input to command the end effector to move with a constant speed to the right.

$$\boldsymbol{f}_{\mathrm{p}}(s) = (\boldsymbol{f}_p|_{s=1} - \boldsymbol{f}_p|_{s=0})s + \boldsymbol{f}_p|_{s=0}, \tag{4.1}$$

On the other hand, we specify the radius as piece-wise linear functions as follows:

$$r_{\mathrm{tube}}(s) = \begin{cases} D_1/2 & \text{if } 0 \leq s \leq s_a \\ \frac{D_2 - D_1}{2s_b}(s - s_a) + \frac{D_1}{2} & \text{if } s_a \leq s \leq s_a + s_b \\ D_2/2 & \text{if } s_a + s_b \leq s \leq 1 \end{cases}, \tag{4.2}$$

where D_1 and D_2 are two diameters that need to be selected, and s_a and s_b are values of s that define the lengths of the sections of the GET-ME virtual fixtures. This geometric definition make the shape of the GET-ME virtual fixtures resemble to the one of a funnel (see Fig. 4.11).

Below, the assistive strategy of the funnel is described using the constraint-based framework described in chapter 3. First, the used building blocks are listed in (see Table 4.5). All building blocks were explained in chapter 3, and were used in combination with other BBs[2]to create different assistive

strategies.

Table 4.5. Specification of building blocks (BBs) that are used to generate GET-ME virtual fixtures. Hard constraints are indicated by the omission of $\hat{w}_i$.

#	Expression e_i	Params	Constraints
1)	$\boldsymbol{p}(\boldsymbol{q})$	$\hat{w}_1$	$\dot{\boldsymbol{e}}_1 \rightarrow \boldsymbol{v}_{\text{hmi}}$: SIM, end effector velocity.
2)	s	$k_2 > 0$	$0 \leq e_2 \leq 1$: RAM, limits on the progress.
3)	$\hat{\boldsymbol{n}}_R$	$\hat{w}_3$ $k_3 > 0$	$\boldsymbol{e}_3 \rightarrow \boldsymbol{0}$: RAM, automatic orientation path.
4)	$\left\|\boldsymbol{p}(\boldsymbol{q}) - \boldsymbol{f}_p\right\|$	$\hat{w}_4$ $k_4 > 0$	RAM, RVGF $e_4 \leq r_{\text{tube}}(s)$: remain in tube.
5)	s	$\hat{w}_5$	$\dot{e}_5 \rightarrow 0$: dampen s.
6)	$\boldsymbol{q}$	$k_6 > 0$	RAM, joint limits $\boldsymbol{q}_{\min} \leq \boldsymbol{e}_6 \leq \boldsymbol{q}_{\max}$: position limits.
7)	$\boldsymbol{q}$	$k_7 > 0$	$\dot{\boldsymbol{q}}_{\min} \leq \dot{\boldsymbol{e}}_7 \leq \dot{\boldsymbol{q}}_{\max}$: velocity limits.

In order to use constraint 1, the 6-DOF signals coming from the joystick transducer of the SpaceMouse go through a mapping such that it only keeps the linear terms (i.e. $\boldsymbol{v}_{\text{hmi}} \in \mathbb{R}^3$). This is detailed using the transducer configuration shown in listing 4.1 and is consistent with specification of constraint 1 (SIM).

Listing 4.1. Transducer configuration used for the SpaceMouse joystick transducer during the EDAN use cases (UC8, UC9 and UC10).

```
{
    "name": "joystick",
    "parent HMI": "SpaceMouse",
    "information flow": H → M,
    "DOFs": 6,
    "interaction modalities":
    {{
        "identifier": "Linear teleoperation",
        "cause": "Linear Interaction with joystick",
        "effect": "Linear motion of end effector",
```

[2]Notice that, despite the virtual fixtures moving towards intent-inferred targets, the RAM "attraction to intent-inferred targets" is not included as done in chapter 3, because this adds first-order dynamics. Instead, we want the virtual fixtures to teleport towards the intent-inferred locations.

```
            "parameters": {...},
            "mapping":
            {
                "Input interpretation": "Desired twist of the
                    tcp expressed in a frame attached to the
                    joystick",
                "Input units": [m/s,m/s,m/s,rad/s,rad/s,rad/s],
                "Output interpretation": "Desired linear
                    velocity of the tcp expressed in a frame
                    attached to the world",
                "Output units": [m/s,m/s,m/s],
                "function": "linear map ℝ^6 → ℝ^3",
                "periodicity": 0.01,
            }
        }}
    }
```

In addition, the two levels of autonomy shown in Table 4.6 are specified in order to have certain adaptive behaviors. The first one, as explained in chapter 3, allows the user to escape from the tube (i.e. deactivate the impedance of virtual fixtures) when the end effector goes far away enough from it. The second one, linearly increases the "apparent stiffness[3]" of the virtual fixture when the progress variable increases (i.e. when the end effector is closer to the target) by increasing weight $\hat{w}_{4_2}$. The effect of both levels of autonomy are illustrated in Fig. 4.11.

Fig. 4.11 also shows different regions which arise from the aforementioned definition of BBs and levels of autonomy. These regions are explained as follows, from inside to outside:

- *free motion region*: this region is in the inner part of the GET-ME virtual fixture grants free motion to the user when $e_4 \leq r_{\text{tube}}(s)$, thanks to the inequality constraint 4.
- *impedance region*: this region attracts the end effector with a specified impedance towards the free motion region when the inequality constraint 4 is violated. The impedance is specified through gain k_4 and weight $\hat{w}_{4_2}$.
- *escaping region*: this region still attracts the end effector, but decreases the weights of the RVGF RAM ($\hat{w}_{4_1}$ and $\hat{w}_{5_1}$) linearly in

the outward radial direction, thanks to the level of autonomy related to G_1.

- *escaped region*: in this region the impedance given by constraint 4 is negligible and it is said that the user (or more precisely, the robot) has escaped the virtual fixture, thanks to the level of autonomy related to G_1.

Table 4.6. Specification of levels of autonomy and the involved weight factors for the GET-ME virtual fixtures. The descending and ascending weight factors $\hat{w}_{i_k}$ are displayed in separate columns. Subscript i indicates the corresponding constraint (see Table 3.1) and subscript k indicates the corresponding arbitration parameter. For all weights $\hat{w}_i$ whose factors $\hat{w}_{i_k}$ are not mentioned here: $a_i = 0$ and $b_i > 0$. For all the factors mentioned here: $\hat{w}_{i_k}^{\max} = 1$, $\hat{w}_{i_k}^{\min} = 0$ and $b_i = 0$, unless stated otherwise.

$\mathbf{G_k}$	**Desc.**	**Asc.**	**Description of the arbitration parameter**
G_1	$\{\hat{w}_{4_1}, \hat{w}_{5_1}\}$	$\{\}$	Escaping: RVGF vs full user control *Generic-policy*: $G_1 = \left\|\boldsymbol{p}(\boldsymbol{q}) - \boldsymbol{f}_p(s)\right\|$, where $G_1^0 = r_{\text{tube}}(s) + d_{\text{th}}^0$ and $G_1^1 = r_{\text{tube}}(s) + d_{\text{th}}^0 + d_{\text{th}}^1$ define the escaping regions. b1 > 0 is a small value in order to keep adapting s.
G_2	$\{\}$	$\{\hat{w}_{4_2}\}$	apparent stiffness: softer vs stronger RVGF *Generic-policy*: $G_2 = s$, where $G_2^0 = 0$, $G_2^1 = 1$, $\hat{w}_{i_k}^{\max} = 1$ and $\hat{w}_{i_k}^{\min} > 0$.

4.3.3 Intent estimation

This subsection introduces a simple algorithm that predicts the next target object (whose position is $\boldsymbol{x}_{\text{pred}}$) and determines the target position of the virtual fixture ($\boldsymbol{f}_p|_{s=1} = \boldsymbol{x}_{\text{pred}}$). As explained by Fig. 4.10, this position is only updated when the user escapes from the virtual fixture and also only if $\|\boldsymbol{v}_{\text{hmi}}\| > 0$. After entering the escaped region, the virtual fixture is able to teleport towards the predicted target while continuously updating whenever there is a change of intention. For predicting such intention, the user input velocity $\boldsymbol{v}_{\text{hmi}}$ and the position of the object relative to the tcp $\boldsymbol{p}^{\text{tcp},O_k}$ are used as intention indicators. An object is considered to be more likely to be the target object if the *distance* between the position of the object and

[3]The term "apparent stiffness" is used because the user feels like the tube is stiffer when it is increased. The higher it is, the more the user needs to push the SpaceMouse (which has a spring) to go outside the virtual fixture, and therefore it is more difficult to escape.

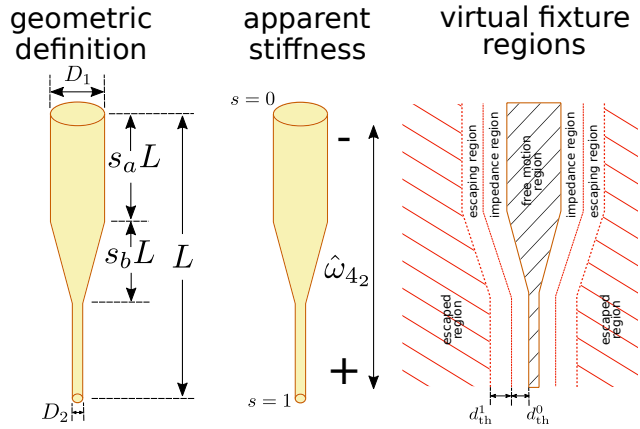


Figure 4.11. Illustration of some properties of the GET-ME virtual fixtures. The left part of the figure shows the variables involved in the geometric definition. The middle part of the figure shows how the apparent stiffness varies with respect to the progress variable s. The right part of the figure shows a sectional view with relevant regions involved in the virtual fixtures. In this figure distinction is made between the impedance region and the escaping region (in contrast to Fig.4.10 where both are shown in light red).

the tcp and the *angle* between $\boldsymbol{p}^{\text{tcp},O_k}$ and $\boldsymbol{v}_{\text{hmi}}$ are both smaller. Thus, both criteria are used to create the following loss function:

$$f_{\text{loss}} = w_{\text{distance}} \underbrace{\left\| \boldsymbol{p}^{\text{tcp},O_k} \right\|}_{distance} + w_{\text{angle}} \underbrace{\arccos\left(\frac{\boldsymbol{v}_{\text{hmi}}^T \boldsymbol{p}^{\text{tcp},O_k}}{\|\boldsymbol{v}_{\text{hmi}}\| \, \|\boldsymbol{p}^{\text{tcp},O_k}\|} \right)}_{angle}, \tag{4.3}$$

where $\boldsymbol{p}^{\text{tcp},O_k} = \boldsymbol{x}_k - \boldsymbol{p}(\boldsymbol{q})$ is the position of object O_k relative to the tcp and w_{distance} and w_{angle} are parameters from 0 to 1 that weight both criteria. The higher that w_{angle} is compared to w_{distance}, the more important the alignment criterion becomes over the closeness criterion. The value of the loss function (4.3) is calculated for all objects $O_k \in \boldsymbol{O}_{\text{set}}$, which are retrieved by EDAN's vision system that takes into account a world representation[103], and the object with the minimum loss is then selected as the predicted target object. Objects that have a negative alignment criterion (i.e.the term inside the $\arccos$ on (4.3)) are not taken into account. Fig. 4.12 illustrates two examples that show why both criteria in the loss function play an important role.

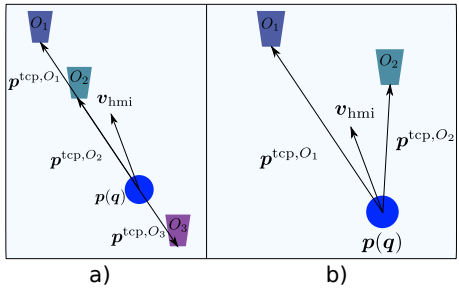


Figure 4.12. Diagram that illustrate, through two examples, the loss function criteria used to predict the most likely object. a) shows an example where both objects have the same alignment with $\boldsymbol{v}_{\text{hmi}}$; it is clear that O_2 should be selected (in case the intended object is O_1 the user can then escape from the generated virtual fixture to target O_2). b) shows an example where O_1 is more aligned with $\boldsymbol{v}_{\text{hmi}}$ but is further away than O_2, and thus the prediction depends on the selection of w_{distance} and w_{angle}. O_3 is not chosen since the alignment term is negative and, thus, it is not taken into account.

4.3.4 Control modes

So far, in previous use cases, we have only used a single control mode in order to execute the various assistive strategies proposed. This control mode consists in sending joint velocity setpoints to the robot (which are generated by eTaSL) and then get joint positions feedback to close the loop. Nevertheless, there are more control modes that are relevant to produce different behaviors.

Some control modes are enabled or restricted by the robot manufacturer. They are enabled by: *(1)* the type of commands that are allowed to be sent to the robot (e.g. torque commands, joint position commands, cartesian position commands, etc), *(2)* the type of low level controller that executes them (e.g. position control, impedance control, etc), and *(3)* the types of feedback that can be obtained from the robot (e.g. measured torques, measured currents, measured joint positions, cartesian poses, etc). The system developer[4] then needs to combine some of these features to create different control modes. A collection of such control modes should be included in a robot driver as alternative ways to control the robot. The application developer then can select the most suitable control mode for the specific application. In this subsection, we briefly discuss alternative control modes that we tested on EDAN. We showcase these alternatives as

exploratory work performed during the research visit at DLR.

EDAN is an advanced robot that can run different controllers at different layers. At a low level, EDAN runs a whole-body impedance controller [31] which coordinates the movements of the robot arm and the wheelchair, allowing the user to simultaneously control the combined DOFs with one lower-dimensional HMI (e.g. SpaceMouse). This strategy combines a velocity-based controller for the wheelchair and a torque-based Cartesian impedance controller [10] for the robot arm, which makes the arm behave compliantly like a spring for safe physical interaction. This approach also allows changing the null space configuration of the robot arm through force interaction. These low-level controllers run at 1 kHz and receive Cartesian robot poses in order to command the robot.

Since the generation of such pose set-points is usually slower than 1 kHz, EDAN uses a Cartesian space pose interpolator [107] that eliminates set-point discontinuities. The control scheme is summarized in Fig. 4.13, where all the described lower level impedance controllers and the pose interpolator combined run in the “EDAN” block and the integration step is done in the “robot driver” block.

In the context of this research visit, we were restricted to control only the Cartesian pose of the end effector instead of the joint positions/velocities, as done in previous use cases. We could choose between purely torque-based Cartesian impedance control (for only controlling the robot arm) or whole-body impedance control (for both arm and wheelchair). Thus, the explored control modes are limited according to these restrictions.

Using Cartesian pose commands that are executed by the offered impedance controllers have the following advantages:

- The null-space control feature of EDAN remains enabled, such that the configuration of the robot can safely change within the nullspace due to physical interaction.
- the kinematic modeling of the robot is not required to execute the assistive strategies.
- The whole-body impedance controller can be exploited without much effort, where the assistance remains the same as for a standalone robot arm, as shown in the use cases.

[4]The role of the different stakeholders was explained in subsection 1.3.1 and is repeated at the conclusions of this section.

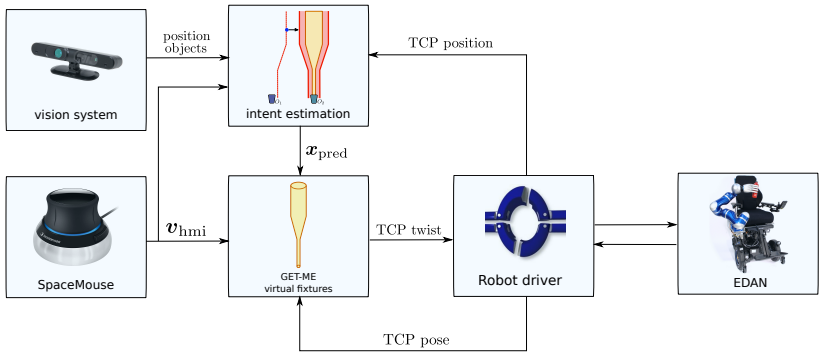


Figure 4.13. Control scheme used to control EDAN. Cartesian impedance whole-body control runs at the EDAN block, while the GET-ME virtual fixtures are generated by a velocity-resolved eTaSL controller.

- Safety measures provided by EDAN's controller, such as a safety manifold to avoid collision with the user, remain active and can be exploited.

On the other hand, this also has the following disadvantages:

- Singularities given by the rotation representation (e.g. euler angles) are introduced.
- Robot joint limit constraints cannot be taken into account.
- Constraints in different frames than the end effector of the robot (e.g. a frame at the wrist) cannot be constrained.

To control the robot in Cartesian space with eTaSL, instead of using the kinematic model of EDAN, we used a virtual kinematic chain composed of three orthogonally placed prismatic joints and a spherical joint (i.e. three revolute joints), each with corresponding position and velocity limits for safety (e.g. not hitting the table). Thus, the joint vector $\boldsymbol{q} \in \mathbb{R}^6$ contains three position coordinates and three orientation coordinates (in roll-pitch-yaw angles[5]) and, thus, controlling the joints is equivalent as directly controlling the Cartesian pose.

With such kinematic model in mind, eTaSL generates desired velocities $\dot{\boldsymbol{q}}_{\text{des}}$ at 100 Hz. Nevertheless, EDAN is a position-controlled robot that receives

[5]This virtual kinematic chain, due to the euler angle representation, introduces singularities.

Cartesian pose setpoints $\boldsymbol{q}_{\text{set}}$. To translate the velocity setpoints into $\boldsymbol{q}_{\text{set}}$, a common approach is to integrate the velocity setpoints over time. This assumes that the tracking error of the robot is negligible, and therefore a simple integration (open-loop) is used, which ignores the position feedback.

Alternatively, we propose the following method that takes into account position feedback (closed-loop), which does not make such assumptions and additionally exploits impedance control at a lower level. This occurs because physical interactions (e.g. collisions) affect the position feedback according to the specified impedance in the controller, and this is taken into account when updating the setpoints. Our method updates the pose setpoints $\boldsymbol{q}_{\text{set}}$ as follows:

$$\boldsymbol{q}_{\text{set}} = \boldsymbol{q}_{\text{feedback}} + T_l \dot{\boldsymbol{q}}_{\text{des}}, \tag{4.4}$$

where $\boldsymbol{q}_{\text{feedback}}$ is the position feedback that comes from EDAN and T_l is referred to as the *lead time*[6]. The job of the lead time is to compensate for round-trip time delay of the eTaSL-EDAN control loop and the slow response introduced by the pose interpolator of EDAN. EDAN enforces the use of such interpolator to smooth the execution of setpoints produced at a lower rate, but drastically reduces the bandwidth of EDAN's position control. This pose interpolator could not be bypassed for practical reasons, and therefore it was compensated by selecting the lead time as $T_l = T_d + \tau_{\text{robot}}$, where T_d is the time delay and τ_{robot} is the time constant of the robot (i.e. EDAN) approximating its behavior as a first order system.

Proof:

First, the robot is modeled in the Laplace domain as a first order plant with time delay as follows:

$$\mathcal{L}\{\boldsymbol{q}_{\text{feedback}}\} = \frac{1}{\tau_{\text{robot}}\varsigma + 1} e^{-T_d\varsigma} \mathcal{L}\{\boldsymbol{q}_{\text{set}}\} \tag{4.5}$$

where ς is the Laplace variable, τ_{robot} is the time constant of the first order plant (mainly influenced by the pose interpolator), and T_d is the time delay. Combining (4.5) with (4.4) in the Laplace domain:

[6] In control systems theory, the lead time is found in a lead compensator, where it is known to cause a phase lead angle in the frequency domain [43]. The intuitive meaning is that, if the reference signal is changing with time, the controller anticipates by looking ahead in time to compensate a delay in the response of the controlled plant.

$$\mathcal{L}\{\boldsymbol{q}_{\text{feedback}}\} = \frac{1}{\tau_{\text{robot}}\varsigma + 1} e^{-T_d\varsigma}(\mathcal{L}\{\boldsymbol{q}_{\text{feedback}}\} + T_l\varsigma\mathcal{L}\{\boldsymbol{q}_{\text{des}}\}) \tag{4.6}$$

By taking a first order approximation of the exponential function as $e^{-T_d\varsigma} = 1 - \varsigma T_d$ and replacing in (4.6):

$$\mathcal{L}\{\boldsymbol{q}_{\text{feedback}}\} = \frac{1}{\tau_{\text{robot}}\varsigma + 1}(1 - \varsigma T_d)(\mathcal{L}\{\boldsymbol{q}_{\text{feedback}}\} + T_l\varsigma\mathcal{L}\{\boldsymbol{q}_{\text{des}}\}) \tag{4.7}$$

After factorizing the terms $\mathcal{L}\{\boldsymbol{q}_{\text{feedback}}\}$:

$$\frac{(\tau_{\text{robot}}\varsigma + 1 - (1 - \varsigma T_d))}{\tau_{\text{robot}}\varsigma + 1}\mathcal{L}\{\boldsymbol{q}_{\text{feedback}}\} = \frac{T_l\varsigma\mathcal{L}\{\boldsymbol{q}_{\text{des}}\}}{\tau_{\text{robot}}\varsigma + 1} \tag{4.8}$$

By simplifying terms, the following expression is obtained:

$$(\tau_{\text{robot}} + T_d)\varsigma\mathcal{L}\{\boldsymbol{q}_{\text{feedback}}\} = T_l\varsigma\mathcal{L}\{\boldsymbol{q}_{\text{des}}\} \tag{4.9}$$

By applying the inverse Laplace transform to return to the time domain:

$$(\tau_{\text{robot}} + T_d)\dot{\boldsymbol{q}}_{\text{feedback}} = T_l\dot{\boldsymbol{q}}_{\text{des}} \tag{4.10}$$

Therefore, T_l is selected as $T_d + \tau_{\text{robot}}$ since it is desired that $\dot{\boldsymbol{q}}_{\text{feedback}} = \dot{\boldsymbol{q}}_{\text{des}}$.

Equation (4.4) is exploited to illustrate three different use cases, each arising from a specific control mode as shown in Table 4.7. Each control mode is achieved through a combination of the type of the pose feedback $\boldsymbol{q}_{\text{feedback}}$, the reference frame in which the terms of (4.4) are expressed, and the low level controller chosen. The effects and implications that each control mode has on each use case is summarized in Table 4.7 and further discussed in the next subsection.

4.3.5 Implementation and results

The implementation of the methods was done in the EDAN robot at the Institute of Robotics and Mechatronics of the German Aerospace Center

Table 4.7. Control modes used in the three use cases related to EDAN. The terms of (4.4) are expressed in the specified reference frame. More combinations can be explored and have alternative effects, but we limited our use cases to these.

Use Case	**Type of q_{feedback}**	**Reference frame**	**Low level controller**	**Effect**
UC8	pose calculated by the pose interpolator.	attached to the wheelchair.	Cartesian impedance controller.	The joint deviations caused by physical interactions due to the impedance controller do not affect the position feedback. The arm and the wheelchair are controlled separately.
UC9	pose calculated through forward kinematics using joint angles measured by EDAN's encoders.	attached to the wheelchair.	Cartesian impedance controller.	The joint deviations caused by physical interactions due to the impedance controller do affect the position feedback. Thus, the robot arm can be hand guided. The arm and the wheelchair are controlled separately.
UC10	pose calculated by the pose interpolator.	attached to the world.	whole-body impedance controller.	The joint deviations caused by physical interactions due to the impedance controller do not affect the position feedback. The arm and the wheelchair are controlled simultaneously according to the whole body controller.

(DLR). The daily living task being showcased is a pick-and-place task with cups in a mockup kitchen. Three use cases were executed using the assistance explained before in this chapter, each with variations in the control mode, as summarized in Table 4.7.

For all use cases, the parameter selection was made in situ based on the specific requirements of the objects and the scenario (a mockup kitchen). For example, the diameters of the tube and the apparent stiffness parameters were tuned such that the user cannot escape from the tube when there was risk of collision with the walls of the crate. In addition, the intent estimation was done over objects $O_k \in \boldsymbol{O}_{\text{set}}$, where $\boldsymbol{O}_{\text{set}}$ is a set containing all possible target objects. In all three use cases, the possible target objects were always considered to be the spaces of the grid-like cup storing crate plus the cups that are outside the crate.

Fig. 4.14 shows snapshots of UC8 being executed on EDAN. At instant *I*, the robot has a grasped cup and the user is commanding the robot towards one of the empty spaces in the grid. While moving, the intent estimation algorithm is running and makes the virtual fixture teleport to the predicted locations. At *II* the end effector gets trapped by a GET-ME virtual fixture and the user approaches towards placing the object while being guided.

Then, at *III* the gripper automatically releases the cup when the progress variable s is higher than a threshold. Subsequently, the user then wants to grasp another cup. To do so, at *IV*, the user decides to move the end effector upwards since it is easier to escape from the GET-ME virtual fixtures due to the variable weight $\hat{w}_{4_2}$ (see Fig. 4.11). Then, at *V*, the user approaches towards another cup where the virtual fixture is waiting to provide guidance. Finally, at *VI*, the user easily manages to reach and grasp the cup thanks to the provided assistance.

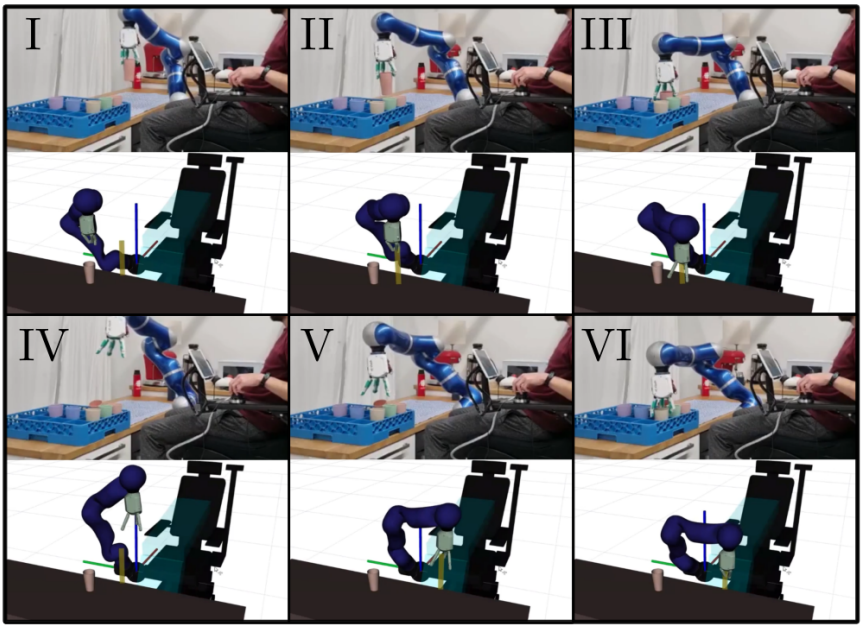


Figure 4.14. Snapshots of UC8. A pick-and-place task with cups is being executed using EDAN with assistance given by GET-ME virtual fixtures. For each time instance (denoted by roman numbers) a real view and a virtual view is shown. The real view shows the task being executed for picking and placing real cups, while the virtual view allows to see the location of the GET-ME virtual fixture. The GET-ME virtual fixture is depicted as a straight yellow tube despite it actually has the shape of a funnel. The cup in the virtual view is static and should be ignored.

Fig. 4.15 shows snapshots of UC9 being executed on EDAN. This use case is running the same assistive strategy based on the GET-ME virtual fixtures. As observed in Table 4.5, there are no admittance (i.e. force) constraints. Nevertheless, a (mockup) caregiver is able to move the robot with the hand. This is a consequence of the control mode used, particularly the selection of the position $\boldsymbol{q}_{\text{feedback}}$, which takes into account the joint deviation of the

robot caused by the hand thanks to the Cartesian impedance controller. Since such controller allows joint deviations, eTaSL updates the positions with the feedback and does not generate any velocity when the user is not moving the SpaceMouse and when the end effector is in the escaped region (as shown in *I*) or in the free motion region. Since the assistance is running, at *II*, the end effector gets trapped by the GET-ME virtual fixture on top of the cup. At *III* conflicting constraints start acting (i.e. the walls of the virtual fixture) and it can be observed that the mockup caregiver pushes the robot and is able to feel the walls of the virtual fixture. This is done because, when the robot is pushed towards the impedance region, eTaSL generates opposing velocities that are then integrated with (4.4) and sent to EDAN.

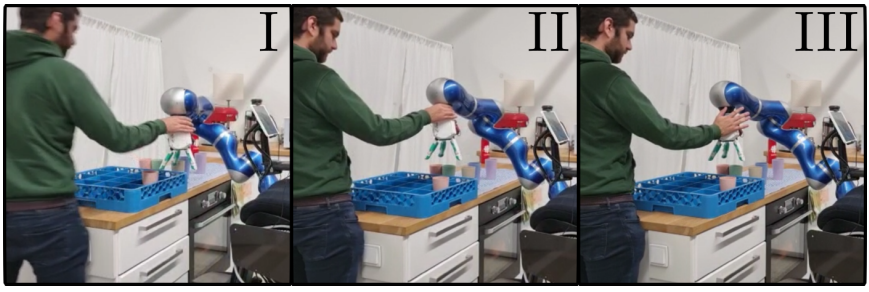


Figure 4.15. Snapshots of UC9, where a mockup caregiver is hand guiding the robot. Despite the assistive strategy being the same as in the two other use cases (i.e. same building blocks and levels of autonomy, without admittance constraints), hand guiding is enabled with GET-ME virtual fixtures active.

Finally, the last use case within this assistive scenario is explained. While the task and assistive strategy remains the same, now the user has the possibility of controlling the arm and the wheelchair simultaneously and seamlessly with the same SpaceMouse. This is a consequence of two modified aspects in the control mode (as mentioned in Table 4.5): *(i)* the low-level impedance whole-body controller of EDAN is chosen, and *(ii)* the terms of (4.4) (including $\boldsymbol{q}_{\text{feedback}}$) are expressed in a frame attached to the world, which remains still when the wheelchair moves. Since the kinematic model used in eTaSL is in Cartesian space, the kinematics of the wheelchair do not need to be modeled. Thus, no modification (besides modifying the position limits of the end effector) was needed from the eTaSL side. In other words, the same assistive strategy can be executed in the three use cases without any effort, by exploiting the different low-level controllers and capabilities provided by the manufacturer.

Fig. 4.16 shows snapshots of UC10 being executed on EDAN. At *I*, the user

starts commanding EDAN using the SpaceMouse to go towards the cups. Before *II* only the robot arm was moved due to this command, but at this moment the arm almost reached its workspace limits. Therefore, the whole-body controller activates the wheelchair so that, together with the arm, they both move simultaneously according to the user inputs while expanding the workspace. At *III* the end effector is just above the cup and, for this reason, the user switches the desired direction to reach the cup, facilitating the control of the user. The whole-body controller automatically stops moving the wheelchair and uses only the arm to move downwards towards the cup. At *IV* the user is able to grasp the desired cup. Throughout the sequence the whole-body controller only uses the wheelchair to expand the workspace of the robot, and leaves the precise manipulation to the robot arm.

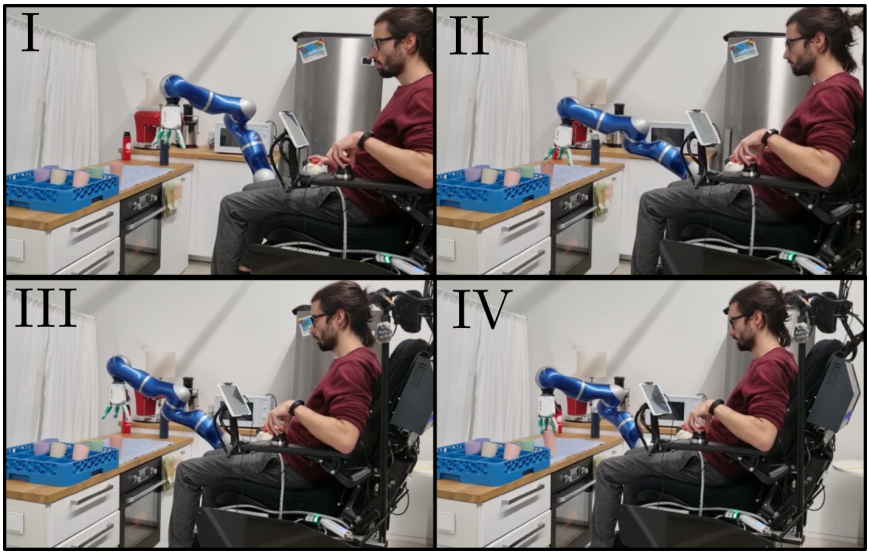


Figure 4.16. Snapshots of UC10, where a user controlling EDAN (both arm and wheelchair) through the SpaceMouse, with the impedance whole-body controller active, while being assisted by the GET-ME virtual fixtures.

4.3.6 Discussion and conclusions

For this assistive scenario we demonstrated the capabilities of our framework by rapidly developing and deploying a novel assistive strategy to reach objects. This was done in one month, including integration with the whole body controller of DLR, deployment and gathering results from

the experiments. Approximately 3/4 of the time was used to develop the C++ robot driver (shown in Fig. 4.13) that interfaces our framework with EDAN, and the remaining time was used to develop and deploy the assistive strategy and explore the different control modes. This short developing and deploying time of the assistive strategies was enabled by the reconfiguration capabilities of our framework, which allowed us to reuse building blocks, levels of autonomy, and HMI interaction modalities, that were previously developed for other applications.

The novel assistive strategy introduced in this chapter, referred to as the GET-ME virtual fixture, is able to effectively and seamlessly guide the user towards different objects that are detected by vision. Through the combination of escaping regions and intent estimation, the user can be assisted to reach desired objects even with such a simple intent estimation that is prone to make wrong predictions. This occurs because, in case of a wrong prediction, or if the user changes the intention, the user is still able to correct by escaping from the virtual fixture and, subsequently, the virtual fixture teleports towards the newly predicted object.

As an additional contribution, a new way of integrating our constraint-based controller with a complex lower-level controller was developed (i.e. (4.4)). This approach allows the combination of higher and lower level controllers in such a way that all the features of both controllers can be exploited. Different control modes can be used with this approach, each having particular features (as shown in the use cases). While the use cases provide relevant insights about the importance of carefully choosing these control modes, the work performed around this topic was merely explorative and requires more in-depth study and formalization in the context of user-centered robotics.

Our approach also provides a "poor man's" (passive) haptic feedback. Several approaches in literature (e.g. [76]) rely on haptic force feedback so that the users feel the virtual fixtures and the forces applied by the robot. Such approaches provide better performance when the manipulation task is challenging and requires precision. However, such HMIs that are capable of providing such haptic feedback are bulky, expensive, and usually not suitable for users with motor impairments. During the experiments the user was able to feel the presence of such virtual fixtures due to the combined effects of having a spring in the joystick of the SpaceMouse, and that the end effector slows down in the impedance regions when colliding with the virtual fixture walls. Since the user notices through visual feedback that the end effector slows down, the user naturally presses the SpaceMouse

harder to escape from the virtual fixture while feeling a proportional force. So, even if there is no active and high-performance haptic feedback, there is a passive haptic-visual feedback that makes the user aware of the presence of the virtual fixtures.

For the intent estimation, ideally the set of possible target objects O_{set} should be updated by a reasoner agent that considers additional information about the task and affordances of the objects. For example, if the cup is grasped, then O_{set} is the set of empty spaces, and if there is no grasped cup, O_{set} should be the set of graspable objects. This could be implemented with a PDDL-based approach such as the one presented in [22]. However, this is out of the scope of this thesis and it is also suggested as future work.

This assistive scenario was used as an example of the complete workflow proposed in Fig. 4.17. Throughout this section, we described the different tasks mentioned in Fig. 4.17, related to the assigned roles of the multiple stakeholders[7] (SD=system developer, AD=application developer, SI=solution integrator). The connection between this assistive scenario and the design workflow is made explicit as follows:

1. by combining existing **Assistive Building Blocks** (SD), a new **assistive strategy** (AD), referred to as the GET-ME virtual fixtures, was rapidly developed.
2. an **HMI driver** (SD), which interfaced the SpaceMouse (connected to EDAN) to our framework, was developed; and the selection of a simple **interaction modality** (AD) was performed (i.e. listing 4.1).
3. a **robot driver** (SD), which interfaced EDAN to our framework, was developed, while different **control modes** (AD), which characterized the difference between the three use cases, were explored.

These components were combined to develop and deploy a **user-centered integrated solution** (SI) during the research visit at DLR. Along the whole process, the initially gathered requirements from users with real motor impairments were taken into account. Nevertheless, the system was never tested by these end users due to time limitations, and hence, the customization/personalization process was only performed for DLR researchers who acted as mockup users. Such testing process with end users is encouraged as future work.

[7] The role of all stakeholders was assumed by the author of this thesis and the DLR collaborators.

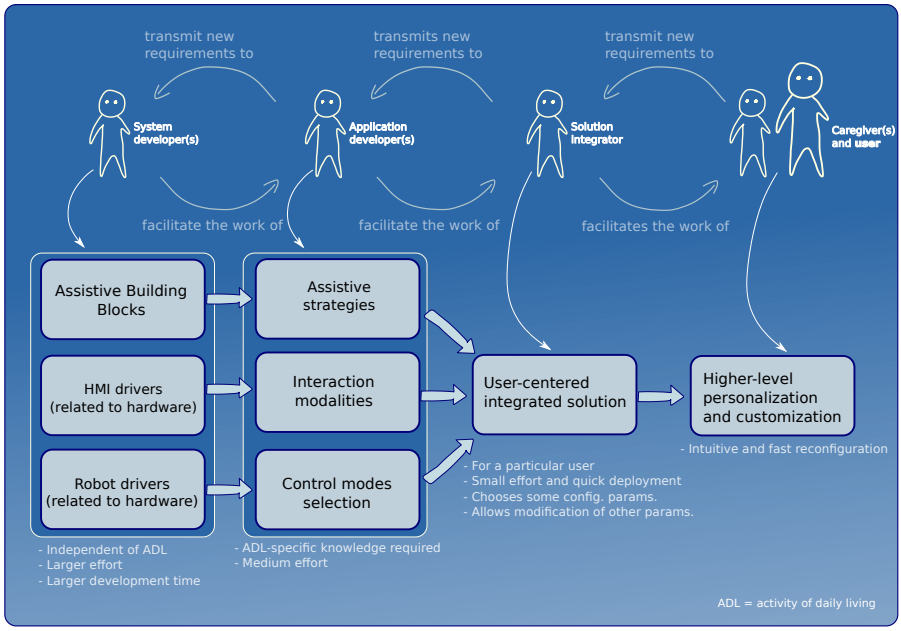


Figure 4.17. Scheme showing a proposed design workflow describing the roles of the main stakeholders involved in the software development and deployment of a user-centered assistive robotic device. This diagram was duplicated from Fig. 1.3 for convenience of the reader.

Acknowledgements

The author of this thesis gratefully acknowledges the help and the valuable contributions of Samuel Bustamante, Gabriel Quere, Jörn Vogel, João Silvério, and other members of the EDAN team at the Institute of Robotics and Mechatronics of the German Aerospace Center (DLR) during this research visit.

The author of this thesis also gratefully acknowledges the financial support provided by Proud_SBO Flanders Make project during the research visit at DLR.

4.4 General conclusions

In this chapter, the proposed framework (*contribution 1*) was exploited to develop multiple use cases in the context of two assistive scenarios. A

first assistive scenario made use of brain-machine interface decodings to control a robot arm. In a second assistive scenario, signals coming from a SpaceMouse were used to control EDAN, a wheelchair-mounted robot arm, in combination with a new assistive strategy. Both assistive scenarios had specific challenges that were tackled through the use of the proposed framework, and emulated realistic and challenging activities of daily living (*contribution 2*). Extensive experimental validation was performed for both assistive scenarios (*contribution 3*).

Particularly, both assistive scenarios had strict development timings that required fast development and deployment capabilities. This was achieved by combining BBs, that were developed in previous use cases, into new assistive strategies. Hence, the reconfiguration capabilities of the framework and its advantages were demonstrated.

Chapter 5

RAMASA: Robotic Assisted Manipulation of Articulated objects with force-based Shared Autonomy

Abstract

In this work we present a shared-autonomy method to assist users during activities of daily living that involve the manipulation of articulated objects, such as doors or drawers, by means of a teleoperated robot controlled with a 6-DOF joystick. By providing assistance in a convenient task frame, we attempt to reduce the workload and improve the performance of users that rely on robotic systems to interact with such objects during activities of daily living. The method relies on active compliance (i.e. force control) to provide useful assistance and, when available, exploits geometric information gathered through interactive perception to increase performance. By exploiting the properties of a velocity-resolved admittance controller, the user can control the velocity and the exerted force of the robot, depending on the directions of contact with the object, by means of the joystick. We validated the applicability and effectivity of our method by performing a user study with able-bodied users.

5.1 Introduction

On a daily basis, people perform several tasks that involve interaction with objects composed by a set of bodies connected through joints/articulations (such as doors, drawers, laptops, jars, etc), henceforth referred to as *articulated objects* (AOs). Interacting with these objects is especially difficult for users with motor impairments due to their force and mobility limitations. This can be alleviated by the use of robotic assistive devices that are teleoperated through Human-Machine Interfaces such as joysticks.

Commercial robotic assistive devices often rely on a joystick with two degrees of freedom (DOF) for teleoperation, which allows a user to control two DOF at a time and, in combination with mode switching, results in the control of the robot's 6-DOF motion. Other approaches map the signals coming from a 6-DOF joystick to directly control the end effector velocity of the robot. These teleoperation strategies require a lot of effort from the users, especially for tasks that involve interaction with AOs. This effort can be reduced by means of shared-autonomy strategies that can provide further assistance to the users.

Thus, as our main contribution, we propose a shared-autonomy strategy that can assist users during teleoperation of AOs with *unknown* geometric properties within *unstructured* environments. The mental model of the user about how to interact with the system does not need to change along the whole task execution, avoiding mode confusion [86]. This makes our assistive method transparent and intuitive for the user, which are two key characteristics of shared autonomy systems [6], as demonstrated through a user study.

In our approach we adopt ideas from the task-frame formalism [21], which exploits force-motion duality in order to specify force/velocity constraints expressed in a convenient frame referred to as the *task frame*. For making a good selection of the task frame, however, geometric properties of the AO must be known beforehand. Our method tackles this by leveraging online *interactive perception* to determine (discretely) which type of joint the robot is interacting with and to estimate (continuously) the pose of the joint axis. The method continuously adapts the task frame to exploit the geometric information gathered by means of interaction.

Subsequently, our method provides the user with further assistance when interaction with an AO is detected. This is done by modulating the user input in certain axes of the adaptive task frame. In this way, the user commands

the robot to interact with an AO, initially with full autonomy, while after enough information is gathered more autonomy is gradually given to the computer to achieve assistance in a seamless fashion.

We focus our attention on the interaction with prismatic and revolute joints, one at a time, since most of the objects commonly found can be modeled as kinematic chains composed only by both. This allows the proposed interactive perception method to work without information from vision systems, avoiding cumbersome (re-)calibration procedures. Nevertheless, our framework could also be applicable to higher DOF joints, such as cylindrical, spherical, etc, as discussed in section 5.6.

This chapter is organized as follows. Section 5.2 summarizes the related work. The notation and the primitives used for the mathematical formulation is presented in section 5.3. Subsequently, section 5.4 proposes the force-based shared autonomy method that provides assistance during interaction with articulated objects. This approach is then validated in a user study in section 5.5. Finally, we discuss the capabilities and limitations of our method and conclude in section 5.6. The words “articulation" and “joint" will be used interchangeably throughout the chapter.

5.2 Related Work

Several works tackle *perception*, *control*, and the combination of both in order to allow robot manipulators to interact with AOs within the autonomous domain. Starting with perception, some methods such as DART [90] can track previously known (e.g. modeled with CAD) AOs in real-time from depth cameras. However, this has the limitation that all objects need to be modeled beforehand (e.g. with CAD software) and therefore unknown objects cannot be dealt with. To address this, some works extract higher-level information from AOs without the need of modeling the specific object beforehand.

A first group of methods tries to predict the kinematics of novel observed objects based on datasets taken from other AOs. For example, in [5], the authors use synthetic data of AOs to train a neural network that is able to estimate off-line a kinematic model for objects observed through a depth camera. Similarly, in [37] a neural network computes from a depth camera a vector field that represents the directions in which different points of the object are allowed to move due to the kinematic constraints of the

articulation. In both, [5] and [37], the information is used to generate and execute a motion plan with the robot.

A second group of methods use interactive perception [15] as a different paradigm to exploit measurements coming from interaction with the AO in order to estimate its kinematic properties. For example, in [81] and [65], visual human demonstrations showing interaction are used for the robot to learn how to interact with a specific object. On the other hand, other approaches use online interactive perception instead. For instance, in [97], the kinematic structure and the corresponding properties (e.g. location of joints) are inferred and estimated, respectively, from pose observations of QR-like tags (e.g. coming from depth cameras) with a Bayesian approach. For the same purpose, in [70] coupled estimators use multimodal sensor information (such as depth cameras, force-torque sensors, etc) without the need of the QR tags.

Given that current perception methods have associated estimation errors and/or make wrong classifications, some works such as [21] proposed smart force controllers that deal with such uncertainties. Also, in [77], an impedance controller is used to open a door autonomously with a wheeled mobile manipulator, without explicitly planning a trajectory. In [54] a robot with elastic series actuators with low mechanical impedance and a hook-like gripper (i.e. no firm grasps) is used in combination with *equilibrium point control* (EPC) to interact with AOs. This approach is extended in [96] with an improved estimation technique that uses previously learned articulation models as priors. Furthermore, in [56], the authors used an adaptive velocity-resolved force controller in order to update online the estimates of the AO while autonomously interacting with it.

All previously mentioned works focus on autonomous robots, while only few examples of approaches that focus on assisting the users while interacting with AOs. For example, the *shared control templates* (SCT) in [83] can be used to constrain the robot motion (e.g. for opening doors or drawers) to assist users that control the system through a 3-DOF joystick or an electromyography interface. However, in [83], the kinematic structure and properties for the motion constraints are associated to a known object which is detected by vision. This limitation is addressed in [82], where the SCT are combined with Learning from Demonstrations (LfD) to learn constraints that can, for instance, assist the users for opening a drawer. In contrast, in [71], the authors do not rely on LfD but infer geometric constraints online, directly from the observed trajectory, and use these constraints to assist the user in following simple trajectories in a simulated environment. The

authors mention that such constraints are difficult to identify as the method is not specific for certain tasks, no prior knowledge is incorporated, and no force constraints are taken into account. Thus, it is unclear how well it would generalize for AOs.

To the best of the authors' knowledge, so far no methods have been proposed that are able to assist users while interacting with unknown AOs (i.e. objects not previously modeled or taught with LfD).

5.3 Representation of Geometrical Primitives and Transformations

This section explains the notation of the geometrical primitives used throughout the text and their corresponding transformations[1]. A position vector ${}_{\mathrm{r}}\boldsymbol{p}^{\mathrm{a,b}} \in \mathbb{R}^3$ represents the origin of a frame {b} with respect to the origin of a frame {a}, expressed in a given frame {r}. A rotation matrix ${}^{\mathrm{r}_1}_{\mathrm{r}_2}\mathbf{R} \in \mathbb{R}^{3\times3}$ expresses a relative orientation of frame $\{\mathrm{r}_1\}$ with respect to frame $\{\mathrm{r}_2\}$. A twist ${}_{r}\mathbf{t}^{p} = [{}_{r}\boldsymbol{\omega}^{T}\ {}_{r}\boldsymbol{v}^{pT}]^{T} \in \mathbb{R}^6$ of a moving body is a vector containing, in that order, angular velocity ${}_{r}\boldsymbol{\omega}$ of the moving body and linear velocity ${}_{r}\boldsymbol{v}^{p}$ of the origin of a frame {p} attached to the moving body, both expressed in a given frame {r}. A wrench ${}_{r}\mathbf{w}^{p} = [{}_{r}\boldsymbol{F}^{T}\ {}_{r}\boldsymbol{T}^{pT}]^{T} \in \mathbb{R}^6$ applied to a body is a vector containing, in that order, applied force ${}_{r}\boldsymbol{F}$ and applied moment (or "torque") ${}_{r}\boldsymbol{T}^{p}$ in the origin of a frame {p} on the body, both expressed in a given frame {r}. The reference frame {r} of both, twists and wrenches, can be transformed by the pre-multiplication of the following matrix:

$$ {}^{\mathrm{r}_1}_{\mathrm{r}_2}\mathbf{P} = \begin{bmatrix} {}^{\mathrm{r}_1}_{\mathrm{r}_2}\boldsymbol{R} & \mathbf{0}_3 \\ \mathbf{0}_3 & {}^{\mathrm{r}_1}_{\mathrm{r}_2}\boldsymbol{R} \end{bmatrix}, \tag{5.1} $$

which performs a rotation operation from $\{\mathrm{r}_1\}$ to $\{\mathrm{r}_2\}$. In addition, the reference frame {r} as well as the reference frame {p} of both, twists and wrenches, can be transformed (by a rotation and a translation, respectively) by the pre-multiplication of the following matrix:

[1] The notation of all geometric primitives has been chosen for convenience in terms of frames. For example, for the position vector, a and b could represent points instead of frames since the orientation of the frame is not needed.

$$ {}^{\mathrm{r}_1}_{\mathrm{r}_2}\mathbf{S} = \begin{bmatrix} {}^{\mathrm{r}_1}_{\mathrm{r}_2}\boldsymbol{R} & \mathbf{0}_3 \\ \left[{}_{\mathrm{r}_2}\boldsymbol{p}^{\mathrm{r}_2,\mathrm{r}_1}\right]_\times {}^{\mathrm{r}_1}_{\mathrm{r}_2}\boldsymbol{R} & {}^{\mathrm{r}_1}_{\mathrm{r}_2}\boldsymbol{R} \end{bmatrix}, \tag{5.2} $$

where $\left[\,\cdot\,\right]_\times$ represents the skew-symmetric matrix of a vector that in the case of (5.2) is the desired translation. Transformations done through matrix ${}^{\mathrm{r}_1}_{\mathrm{r}_2}\mathbf{S}$ will be referred to as *screw transformations*, which are also known in literature as adjoint transformations for $se(3)$. Subscripts and superscripts may be omitted for compactness when the context is clear.

A useful matrix that inverts the order of the linear part and angular part of a screw, and that satisfies $\tilde{\Delta}^{-1} = \tilde{\Delta}$, is composed as follows:

$$ \tilde{\Delta} = \begin{bmatrix} \mathbf{0}_3 & \boldsymbol{I}_3 \\ \boldsymbol{I}_3 & \mathbf{0}_3 \end{bmatrix}, \tag{5.3} $$

5.4 Force-based shared autonomy

This section proposes a force-based shared autonomy method that is divided in three modules:

1. a *velocity-resolved control module* that computes a twist setpoint that constrains the robot to follow a given assisted twist while maintaining joint limits and controlling zero forces and moments to safely maintain a compliant interaction;

2. an *online interactive perception module* that, based on a sliding-window of tcp twists gathered through online interaction with the articulated object, classifies the interaction type (i.e. revolute, prismatic or free space) and estimates on the fly an adaptive task frame accordingly; and

3. an *automatic assistance module* that computes the assisted twist (used by the first module) by transforming the joystick input to the adaptive task frame and downscaling it in the directions that produce motion that is not compatible with the kinematics of the interaction type. This module only downscales the incompatible directions without completely removing them, such that the user can still deviate from the identified interaction type, but requires more

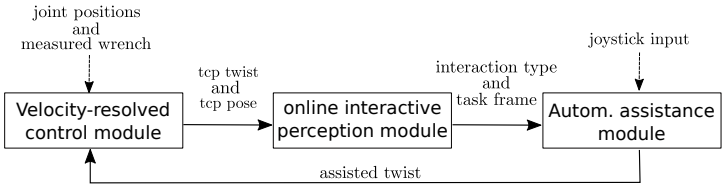


Figure 5.1. Scheme that shows how the three modules of the proposed force-based shared autonomy method are interconnected.

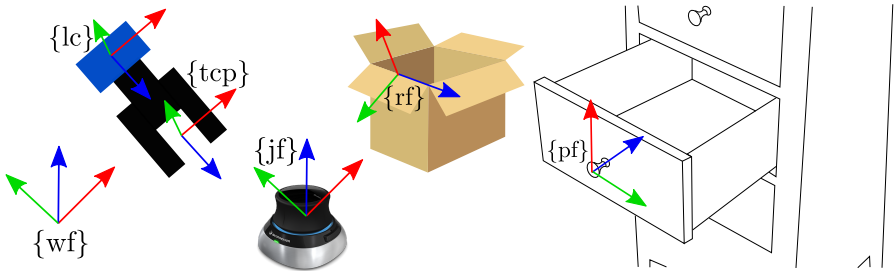


Figure 5.2. Frames (indicated by $\{\cdot\}$) that are relevant for the robot control: wf=world frame, lc=load cell, tcp=tool center point, jf=joystick frame, rf= revolute frame and pf= prismatic frame. RGB color convention is used to denote $\boldsymbol{x}$ $\boldsymbol{y}$ $\boldsymbol{z}$ axes, respectively. By convention, the $\boldsymbol{z}$ axes of prismatic and revolute frames are aligned with the corresponding joint axes.

effort to do so, i.e. the autonomy between the user and the robot controller is shared.

Fig. 5.1 shows how the different modules are interconnected and Fig. 5.2 illustrates all the frames used by the three modules. The 6-DOF joystick of a SpaceMouse (also shown in Fig. 5.2) is used as a human-machine interface (HMI) to acquire the joystick input.

The task execution is said to be without assistance when the interaction type is fixed to be *free space* and consequently the adaptive task frame is kept at the {tcp}. Otherwise, it is said to be with assistance.

The different modules are explained in the remaining part of this section.

5.4.1 Velocity-resolved control module

input: joint positions $\boldsymbol{q}$, measured wrench ${}_{\text{lc}}\mathbf{w}_{\text{m}}{}^{\text{lc}}$ expressed at the load cell frame {lc} (see Fig. 5.2) exerted by the robot on the environment and

assisted twist ${}_{\text{tcp}}\mathbf{t}_{\text{a}}{}^{\text{tcp}}$.

output: instantaneous tcp twists ${}_{\text{tcp}}\mathbf{t}^{\text{tcp}}$ and tcp pose.

In order to allow the user to control the tcp of the robot, while remaining compliant when interacting with objects, this module uses the following velocity-resolved (admittance) wrench controller to obtain a desired twist ${}_{\text{tcp}}\mathbf{t}_{\text{d}}{}^{\text{tcp}}$:

$$ {}_{\text{tcp}}\mathbf{t}_{\text{d}}{}^{\text{tcp}} = {}_{\text{tcp}}\mathbf{t}_{\text{a}}{}^{\text{tcp}} - k_{\text{w}}\,\tilde{\Delta}\,\boldsymbol{C}_{\text{w}}\;{}_{\text{tcp}}\mathbf{w}_{\text{m}}{}^{\text{tcp}} \tag{5.4} $$

where the left term ${}_{\text{tcp}}\mathbf{t}_{\text{a}}{}^{\text{tcp}}$ is the assisted twist coming from the automatic assistance module and the right term is a feedback control law composed of a proportional control gain k_{w} in $[1/s]$, a diagonal compliance matrix $\boldsymbol{C}_{\text{w}} \in \mathbb{R}^{6\times 6}$ that is expressed in the tcp, and wrench ${}_{\text{tcp}}\mathbf{w}_{\text{m}}{}^{\text{tcp}}$. The wrench ${}_{\text{tcp}}\mathbf{w}_{\text{m}}{}^{\text{tcp}}$ is obtained from the measured wrench ${}_{\text{lc}}\mathbf{w}_{\text{m}}{}^{\text{lc}}$ after compensating for the weight of the attached gripper tool and applying the screw transformation ${}^{\text{lc}}_{\text{tcp}}\boldsymbol{S}$ to transform it to the {tcp} frame. $\tilde{\Delta}$ ensures that linear velocities are mapped to forces and rotational velocities to moments. Equation (5.4) can be seen as an admittance control law that follows zero forces/moments and has a feedforward term ${}_{\text{tcp}}\mathbf{t}_{\text{a}}{}^{\text{tcp}}$ that, with assistance, should not cause large forces in the directions of contact since it is compatible with the estimated physical joint constraints. The only forces that ${}_{\text{tcp}}\mathbf{t}_{\text{a}}{}^{\text{tcp}}$ will cause are given by errors associated with the estimation done (see subsection 5.4.2) and user-induced reaction forces that are deliberately kept for making the user remain in (partial) control. These forces are compensated by the right (feedback) term of (5.4).

On the other hand, without assistance, no component of ${}_{\text{tcp}}\mathbf{t}_{\text{a}}{}^{\text{tcp}}$ is downscaled and thus the user is able to freely control the tcp twist in the directions of free space and the exerted wrench at the tcp in the directions of contact. This occurs because when in contact, (5.4) reaches an equilibrium point at ${}_{\text{tcp}}\mathbf{w}_{\text{m}}{}^{\text{tcp}} = (1/k_{\text{w}})\,\boldsymbol{C}_{\text{w}}{}^{-1}\,\tilde{\Delta}\;{}_{\text{tcp}}\mathbf{t}_{\text{a}}{}^{\text{tcp}}$, where the linear/rotational velocities of ${}_{\text{tcp}}\mathbf{t}_{\text{a}}{}^{\text{tcp}}$ are proportional to the forces/moments applied by the user to the joystick (i.e. the larger the force/moment that the user exerts on the joystick, the larger the force/moment that the tcp is going to exert in the direction of contact).

The wrench control (5.4) compensates for measured moments/torques in the tcp frame even if the automatic assistance module (see subsection 5.4.3) performs the assistance in the task frame (i.e. as force-controlled actions

are specified in the autonomous domain [21]). In this way, the stability of the feedback controller[2] is not affected by the dynamics of the task frame estimation.

In order to also take into account additional constraints such as joint position (5.5b) and velocity (5.5c) limits, the desired twist ${}_{\text{tcp}}\mathbf{t}_{\text{d}}{}^{\text{tcp}}$ from (5.4) is applied using the following quadratic optimization problem that at each time instant solves the inverse velocity kinematics (i.e. *velocity resolved*):

$$\underset{\dot{\boldsymbol{q}},\boldsymbol{\varepsilon}}{\text{minimize}} \quad \mu\dot{\boldsymbol{q}}^{\mathsf{T}}\dot{\boldsymbol{q}} + \boldsymbol{\varepsilon}^{\mathsf{T}}\boldsymbol{\varepsilon}, \tag{5.5a}$$

$$\text{subject to} \quad \boldsymbol{q}_{\min} \leq \boldsymbol{q} \leq \boldsymbol{q}_{\max}, \tag{5.5b}$$

$$\dot{\boldsymbol{q}}_{\min} \leq \dot{\boldsymbol{q}} \leq \dot{\boldsymbol{q}}_{\max}, \tag{5.5c}$$

$${}^{\text{wf}}_{\text{tcp}}\boldsymbol{P}\,{}_{\text{wf}}\mathbf{J}^{\text{tcp}}\dot{\boldsymbol{q}} = {}_{\text{tcp}}\mathbf{t}_{\text{d}}{}^{\text{tcp}} + \boldsymbol{\varepsilon}, \tag{5.5d}$$

where the desired twist ${}_{\text{tcp}}\mathbf{t}_{\text{d}}{}^{\text{tcp}}$ computed with (5.4) and included as a constraint. Vector $\boldsymbol{q}$ contains the joint angles, ${}_{\text{wf}}\mathbf{J}^{\text{tcp}}$ is the geometric Jacobian computed at the {tcp} and expressed in {wf}, $\boldsymbol{\varepsilon}$ is a slack variable that relaxes the constraint in case of a conflict (e.g. joint limit), and μ is a regularization factor. The desired joint velocities $\dot{\boldsymbol{q}}$ are obtained by solving (5.5) numerically and sent to the robot (see Fig. 5.3).

The above formulation is only a simplification of the eTaSL formulation (i.e. optimization problem (2.11)), were no feature variables are used and only few constraints are specified.

Both outputs of this module, which are used by the online interactive perception module, are computed as follows. The tcp pose is computed through forward kinematics and the tcp twist is computed as ${}_{\text{tcp}}\mathbf{t}^{\text{tcp}} = {}^{\text{wf}}_{\text{tcp}}\boldsymbol{P}\,{}_{\text{wf}}\mathbf{J}^{\text{tcp}}\dot{\boldsymbol{q}}$, where ${}_{\text{wf}}\mathbf{J}^{\text{tcp}}$ is the Jacobian matrix at the tcp expressed in the

[2] During experiments, we found that when (5.4) was expressed in the task frame, sometimes instabilities were observed. We believe that this can happen because 1) the influence of the dynamics of the adaptive task frame and 2) because the compliance matrix can be determined in tcp frame, but it is difficult to model its behavior in a frame that is always adapting. In addition, we observed that when performing the control in the task frame, instead of the tcp frame, more deformations of the lid of the cardboard box (object used in the user study) appeared because the force/torque interactions at the tcp were not compensated well by the wrench controller. However, the study of the interaction with flexible elements is out of the scope of this thesis.

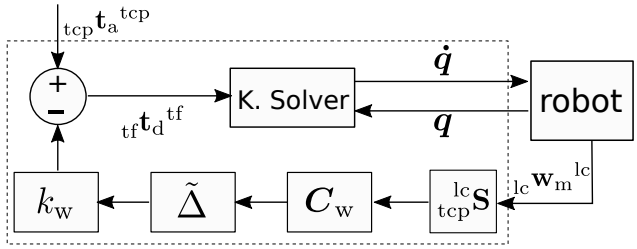


Figure 5.3. Block diagram that summarizes the velocity-resolved control module (blocks inside the dashed rectangle) that is composed of (5.4) and (5.5). The frame {lc} is located at the load cell as shown in Fig. 5.2. The block “K. Solver" stands for kinematic solver, which refers to the numerical solution of (5.5).

world frame {wf}. This assumes that the tracking error of the joint velocity setpoints due to the robot’s low level controller are negligible and offers the following two advantages: *(i)* avoids estimation artifacts caused by noisy numerical derivatives of encoder sensor data, and *(ii)* includes information about the measured forces in the estimation due to the compliant interaction enabled by the wrench control.

This approach for formulating the task in terms of constraints is a straightforward application of the more general framework eTaSL [8]. Thus, more constraints could be included in (5.5) to further assist the user as explained in [52]. More details about the formulation of eTaSL[3] and the numerical solution of the quadratic optimization problem are given in [8].

5.4.2 Online interactive perception module

input: instantaneous tcp twist ${}_{\text{tcp}}\mathbf{t}^{\text{tcp}}$ and tcp pose.

output: adaptive task frame {tf} and interaction type (revolute, prismatic or free space).

This module uses online interactive perception to compute the interaction type and an online task frame {tf} that is determined and smoothly adapted. The proposed approach only uses motion data to compute the interaction type and does not need vision. Force and moment data are used indirectly through their influence on motion.

[3]Open source software, templates, tutorials and ready-to-use examples are available at: `https://etasl.pages.gitlab.kuleuven.be/`

In [21], the task frame is defined and different options for selecting this task frame are explained, which affect the performance of the controller for interacting with the joint of an AO. A good task frame should decouple the required control actions along each of its axes (i.e. force-controlled and velocity-controlled directions) to facilitate the task specification. Furthermore, a task frame attached to the moving body (*condition 1*), with its origin located at the joint axis (*condition 2*), and with one axis aligned with the joint axis (*condition 3*), enables the highest performance. For revolute and prismatic joints, if a task frame that satisfies these conditions is available, it is easy to specify interaction with the object by rotating and translating, respectively, with respect to the joint-aligned axis. This solution results in a better performance compared to generating those motions purely based on force control, which has a limited bandwidth.

This module exploits the task frame approach of [21] but, instead of designing a specific location of the task frame, the task frame is determined and adapted online based on a proposed method. For that purpose, this module estimates in parallel two frames, referred to as a revolute frame {rf} and a prismatic frame {pf} (see Fig. 5.2). If enough information is available, it performs an online adaptation of the task frame towards one of these frames, depending on the interaction type. Otherwise, the interaction type is assumed to be free space and the task frame will adapt towards the tcp frame.

In this work, we considered objects that are not necessarily fixed to the world, but instead are lightweight and easy to move (willingly or unwillingly), which explains the importance of condition 1. For this reason, and since no data coming from vision is used, the estimation done by the module assumes a firm grip between the gripper and the AO.

All summation terms encountered in this section are done over the elements of a set $\mathbf{\Omega}$ containing instantaneous twists ${}_{\text{tcp}}\mathbf{t}_i^{\text{tcp}} = [\boldsymbol{\omega}_i^T \ \boldsymbol{v}_i^{\text{tcp}^T}]^T$ and thus this information will be omitted in all equations for the sake of compactness. In order to update the estimate online, the set $\mathbf{\Omega}$ is treated as a sliding window with a fixed window size $|\mathbf{\Omega}|$ (i.e. cardinality), whose oldest element is replaced by the newest (i.e. input ${}_{\text{tcp}}\mathbf{t}^{\text{tcp}}$ of the module) at each update step.

Revolute frame {rf} estimation

In this work, a frame that satisfies conditions 1, 2 and 3, for a revolute joint, is referred to as a *revolute frame* {rf}. A revolute frame, which is expressed with respect the tcp, is composed of an orientation ${}^{\text{rf}}_{\text{tcp}}\mathbf{R}_t$ and an origin ${}_{\text{tcp}}\boldsymbol{p}_t^{\text{tcp,rf}}$ for the current time t. By convention, axis $\boldsymbol{z}_{\text{rf}}$ will maintain alignment with the joint axis to satisfy condition 3.

In order to determine the **orientation** of {rf}, the unit vector $\hat{\boldsymbol{n}}_t^{\text{rev}}$ that represents the revolute axis is estimated. This is done by calculating the *Average Screw Axis* introduced in [12]. In contrast to [12], in this work the estimates are updated online with a sliding-window approach. The approach is further improved by introducing a regularization term based on the prior belief for the revolute axis:

$$\begin{aligned}\hat{\boldsymbol{n}}_t^{\text{rev}} =& \underset{s.t.\|\hat{\boldsymbol{n}}\|=1}{\text{argmax}}\ \varepsilon_n^{\text{rev}}(\hat{\boldsymbol{n}}^T\hat{\boldsymbol{n}}_{\text{pr}}^{\text{rev}})^2 + \frac{1}{|\boldsymbol{\Omega}|}\sum(\boldsymbol{\omega}_i^T\hat{\boldsymbol{n}})^2 \\ =& \underset{s.t.\|\hat{\boldsymbol{n}}\|=1}{\text{argmax}}\ \hat{\boldsymbol{n}}^T \underbrace{\left(\varepsilon_n^{\text{rev}}\hat{\boldsymbol{n}}_{\text{pr}}^{\text{rev}}\hat{\boldsymbol{n}}_{\text{pr}}^{\text{rev}T} + \frac{1}{|\boldsymbol{\Omega}|}\sum\boldsymbol{\omega}_i\boldsymbol{\omega}_i^T\right)}_{\boldsymbol{C}_\omega}\hat{\boldsymbol{n}},\end{aligned} \tag{5.6}$$

The first term of (5.6) makes the estimate tend to a prior belief $\hat{\boldsymbol{n}}_{\text{pr}}^{\text{rev}}$ (pr = prior) of the revolute axis, and is weighted with a small regularization factor $\varepsilon_n^{\text{rev}}$ related to the direction $\hat{n}$ of a revolute axis. The second term of (5.6) maximizes the weighted average of the alignment (i.e. inner product) between $\boldsymbol{\omega}_i$ and $\hat{\boldsymbol{n}}$, with the weight being the magnitude of $\boldsymbol{\omega}_i$. This weighted average causes small magnitudes of $\boldsymbol{\omega}_i$ to have less influence on the estimate and, in combination with the regularization, avoids the singularity when all $\|\boldsymbol{\omega}_i\| = 0$. This form of optimization problems can be solved efficiently by performing a *singular value decomposition* $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^T$ (svd) of the symmetric[4] matrix $\boldsymbol{C}_\omega$ and selecting $\hat{\boldsymbol{n}}_t^{\text{rev}}$ as the first column of $\boldsymbol{U}$ (i.e. the singular vector corresponding to the highest singular value) [45].

The previous only determines the direction of the revolute axis. The full orientation ${}^{\text{rf}}_{\text{tcp}}\mathbf{R}_t = [\boldsymbol{x}_{\text{rf}}\,\boldsymbol{y}_{\text{rf}}\,\boldsymbol{z}_{\text{rf}}]$ of {rf} is determined such that condition 3 is satisfied by selecting the $\boldsymbol{z}_{\text{rf}}$ axis as the rotation axis $\hat{\boldsymbol{n}}_t^{\text{rev}}$. The two other

[4] Since $\boldsymbol{C}_\omega$ is symmetric, singular value decomposition $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ is the same as eigendecomposition $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^T$ since $\boldsymbol{U} = \boldsymbol{V}$ and $\boldsymbol{U}^T = \boldsymbol{U}^{-1}$.

axes can be selected arbitrarily, but we select them such that: *(i)* they align as much as possible with axes of the tcp frame (this does not affect the control performance but helps the developer interpreting the resulting motion and wrenches), and *(ii)* singularities happening due to alignments between different axes of {tcp} and {rf} are avoided. This is done by following a Gram-Schmidt orthonormalization process, where the original basis is selected to maintain alignment with the tcp, whose orientation with respect to the {tcp} frame is the identity $\boldsymbol{I}_3 = [\hat{\boldsymbol{i}}\,\hat{\boldsymbol{j}}\,\hat{\boldsymbol{k}}]$. This is done as follows:

when $\hat{\boldsymbol{n}}_t^{\text{rev}}$ is mostly aligned with $\hat{\boldsymbol{i}}$ (i.e. $\hat{\boldsymbol{i}}^T\hat{\boldsymbol{n}}_t^{\text{rev}} > \hat{\boldsymbol{j}}^T\hat{\boldsymbol{n}}_t^{\text{rev}}$ and $\hat{\boldsymbol{i}}^T\hat{\boldsymbol{n}}_t^{\text{rev}} > \hat{\boldsymbol{k}}^T\hat{\boldsymbol{n}}_t^{\text{rev}}$), we select $\boldsymbol{y}_{\text{rf}}$ as the orthogonal complement $(\boldsymbol{I}_3 - \hat{\boldsymbol{n}}_t^{\text{rev}}\hat{\boldsymbol{n}}_t^{\text{rev}T})\hat{\boldsymbol{j}}$ after normalization. Because $\hat{\boldsymbol{n}}_t^{\text{rev}}$ is moslty aligned with $\hat{\boldsymbol{i}}$, and $\hat{\boldsymbol{j}}$ is perpendicular to $\hat{\boldsymbol{i}}$, the normalized selection is well conditioned. Similarly, when $\hat{\boldsymbol{n}}_t^{\text{rev}}$ is mostly aligned with $\hat{\boldsymbol{j}}$ or $\hat{\boldsymbol{k}}$, we select $\boldsymbol{y}_{\text{rf}}$ to be $(\boldsymbol{I}_3 - \hat{\boldsymbol{n}}_t^{\text{rev}}\hat{\boldsymbol{n}}_t^{\text{rev}T})\hat{\boldsymbol{i}}$ after normalization. The remaining axis is selected as $\boldsymbol{x}_{\text{rf}} = \boldsymbol{y}_{\text{rf}} \times \boldsymbol{z}_{\text{rf}}$.

In order to determine the **origin** of {rf}, a kinematic model of a perfect revolute joint is assumed such that all rotations made by the {tcp} should be around a fixed axis $\hat{\boldsymbol{n}}_t^{\text{rev}}$, and thus linear velocities $\boldsymbol{v}_i^{\text{tcp}}$ should be the cross product between the angular velocities projected at the rotation axis (i.e. $(\boldsymbol{\omega}_i^T\hat{\boldsymbol{n}}_t^{\text{rev}})\hat{\boldsymbol{n}}_t^{\text{rev}}$) and the estimated origin ${}_{\text{tcp}}\boldsymbol{p}^{\text{tcp,rf}}$. Since a firm grip of the object is assumed, we express ${}_{\text{tcp}}\boldsymbol{p}^{\text{tcp,rf}}$ with respect to the {tcp}, in which a constant value is expected. To estimate ${}_{\text{tcp}}\boldsymbol{p}^{\text{tcp,rf}}$, we propose the following regularized least square problem that incorporates this kinematic model:

$$
\begin{aligned}
{}_{\text{tcp}}\boldsymbol{p}_t^{\text{tcp,rf}} = \underset{\boldsymbol{p}}{\text{argmin}}\ \ & \frac{1}{|\boldsymbol{\Omega}|}\sum \left\| \boldsymbol{\omega}_i^T\hat{\boldsymbol{n}}_t^{\text{rev}}[\hat{\boldsymbol{n}}_t^{\text{rev}}]_\times\boldsymbol{p} + \boldsymbol{v}_i^{\text{tcp}} \right\|^2 \\
& + \varepsilon_p^{\text{rev}} \left\| \boldsymbol{p} - \boldsymbol{p}_{\text{pr}}^{\text{rev}} \right\|^2,
\end{aligned}
\tag{5.7}
$$

where $\boldsymbol{p}_{\text{pr}}^{\text{rev}}$ is a prior location for the origin and $\varepsilon_p^{\text{rev}}$ the corresponding regularization factor. This optimization problem looks for the origin ${}_{\text{tcp}}\boldsymbol{p}^{\text{tcp,rf}}$ that best explains the velocities at the tcp. The following closed-form solution can be found by setting the derivative of the objective function in (5.7) to zero:

$$
{}_{\text{tcp}}\boldsymbol{p}_t^{\text{tcp,rf}} = \left(\varepsilon_p \boldsymbol{I}_3 - \frac{1}{|\boldsymbol{\Omega}|}\hat{\boldsymbol{n}}_t^T \left(\sum \boldsymbol{\omega}_i \boldsymbol{\omega}_i^T\right) \hat{\boldsymbol{n}}_t [\hat{\boldsymbol{n}}_t]_\times [\hat{\boldsymbol{n}}_t]_\times\right)^{-1} \left(\varepsilon_p \boldsymbol{p}_{\text{pr}}^{\text{rev}} + \frac{1}{|\boldsymbol{\Omega}|}[\hat{\boldsymbol{n}}_t]_\times \left(\sum \boldsymbol{v}_i^{\text{tcp}} \boldsymbol{\omega}_i^T\right) \hat{\boldsymbol{n}}_t\right), \quad (5.8)
$$

where the superscript *rev* was omitted from the terms $\varepsilon_p^{\text{rev}}$ and $\hat{\boldsymbol{n}}_t^{\text{rev}}$ for compactness. Noting that the second term in the matrix inverse in (5.8) is positive *semi*-definite, the matrix inverse is well conditioned due to the addition of the positive definite regularization term.

Prismatic frame {pf} estimation

In this work, a frame that satisfies conditions 1 and 3 for a prismatic joint is referred to as a *prismatic frame* {pf} (see fig. 5.2). Condition 2 is not relevant in the case of a prismatic joint, since there is not a clear axis location. Instead, we assume that the human grasps the AO in a convenient and stable point (e.g. the gripper maintains a firm grip of the handle of a drawer). Thus, we can select the **origin** of {pf} as the origin of {tcp}, ${}_{\text{tcp}}\boldsymbol{p}^{\text{tcp,pf}} := {}_{\text{tcp}}\boldsymbol{p}^{\text{tcp,tcp}} = \boldsymbol{0}$.

On the other hand, the **orientation** of {pf} (${}^{\text{pf}}_{\text{tcp}}\mathbf{R} = [\boldsymbol{x}_{\text{pf}}\, \boldsymbol{y}_{\text{pf}}\, \boldsymbol{z}_{\text{pf}}]$) can be computed by repeating a similar procedure as the one proposed for the orientation of {rf}, but using the translational velocities $\boldsymbol{v}_i^{\text{tcp}}$ instead of $\boldsymbol{\omega}_i$, such that an optimization problem in terms of $\boldsymbol{C}_{\text{v}}$ instead of $\boldsymbol{C}_\omega$ is obtained:

$$
\underset{s.t.\|\hat{\boldsymbol{n}}\|=1}{\text{argmax}}\ \hat{\boldsymbol{n}}^T \underbrace{\left(\varepsilon_n^{\text{prism}} \hat{\boldsymbol{n}}_{\text{pr}}^{\text{prism}} \hat{\boldsymbol{n}}_{\text{pr}}^{\text{prism}\,T} + \frac{1}{|\boldsymbol{\Omega}|}\sum \boldsymbol{v}_i^{\text{tcp}} \boldsymbol{v}_i^{\text{tcp}\,T}\right)}_{\boldsymbol{C}_v} \hat{\boldsymbol{n}}, \quad (5.9)
$$

where the first term makes the estimate tend to a prior belief $\hat{\boldsymbol{n}}_{\text{pr}}^{\text{prism}}$ (pr = prior) of the prismatic axis, and is weighted with a small regularization factor $\varepsilon_n^{\text{prism}}$. The prismatic axis $\hat{\boldsymbol{n}}_t^{\text{prism}}$ is found as the first singular vector of matrix $\boldsymbol{C}_{\text{v}}$ (i.e. the solution of (5.9)) and represents the most aligned unit vector with each $\boldsymbol{v}_i^{\text{tcp}}$. The geometric interpretation of (5.9) is equivalent to the one of (5.6). Axis $\boldsymbol{z}_{\text{pf}}$ is selected as $\hat{\boldsymbol{n}}_t^{\text{prism}}$, and axes $\boldsymbol{x}_{\text{pf}}$ and $\boldsymbol{y}_{\text{pf}}$ are selected in the same way as done for {rf}.

Task Frame $\{\mathbf{tf}\}$ Adaptation

In the remaining part of this section, we explain how to compute the task frame {tf} such that it smoothly adapts to {rf}, {pf} or {tcp}, respectively, when the interaction type is classified as *revolute*, *prismatic*, or *free space*. The {tf} adaptation is divided into three steps: *estimation*, *classification* and *smoothing*. All the summation terms encountered in the estimation and classification steps can be efficiently updated by subtracting the corresponding oldest term and adding the newest, without the need of recomputing the whole summation. Thus, increasing the window size does not affect the time complexity, but the required memory does increase (the size of the circular buffer $\boldsymbol{\Omega}$ increases).

Step 1 - Estimation: For the estimation step, the previously explained algorithms to estimate {rf} and {pf} are executed in parallel, but the estimates are only updated when the tcp traveled a specified distance **or** a specified angle since the previous updates (i.e. the estimates update even if there is only pure translation or pure rotation). The distance is calculated as the norm of the difference between the current position of the tcp (module input) and the position where the last estimate was performed. The angle is calculated using the axis-angle representation between the current tcp orientation (module input) and the tcp orientation where the last estimate was performed. The estimates will not be updated if the traveled distance or angle is too small because, when the robot is commanded not to move or to move slowly, there is not enough information to reliably determine the axis of the revolute or prismatic joint.

Step 2 - Classification: This work uses error metrics obtained from the estimation step to classify whether the robot is interacting with a revolute joint, a prismatic joint or (otherwise) moving in free space:

Metrics for revolute: there are two error metrics that can be used to classify the interaction type as revolute, related to the orientation and the origin estimates of the revolute frame. The first error metric related to orientation represents the angular variance (with units $[rad]$) of the revolute axis, and it is calculated as follows:

$$\theta_{\text{rev}} = \arctan\left(\sqrt{\frac{\lambda_{\omega_2} + \lambda_{\omega_3}}{\lambda_{\omega_1}}}\right), \tag{5.10}$$

where λ_{ω_1}, λ_{ω_2} and λ_{ω_3} are the singular values of matrix $\boldsymbol{C}_\omega$ in decreasing

order and can be interpreted as the magnitudes of the axes of the corresponding covariance ellipsoid with units $[\frac{rad^2}{s^2}]$ (as explained in [12]). The metric θ_{rev} goes from zero $[rad]$ when all $\boldsymbol{\omega}_i \in \boldsymbol{\Omega}$ are perfectly parallel (e.g. in an ideal revolute joint or, in general, motion in a plane) to a maximum value of $\pi/2$ $[rad]$.

The second error metric related to the origin uses the coefficient of determination R^2 of the kinematic model fitted with (5.7) (without the regularization term) and it is calculated with the following equation[5]:

$$R^2 = 1 - \frac{\sum \left\| \boldsymbol{\omega}_i^T \hat{\boldsymbol{n}}_t^{\text{rev}} [\hat{\boldsymbol{n}}_t^{\text{rev}}]_\times \, {}_{\text{tcp}}\boldsymbol{p}_t^{\text{tcp,rf}} + \boldsymbol{v}_i^{\text{tcp}} \right\|^2}{\sum \left\| \bar{\boldsymbol{v}} - \boldsymbol{v}_i \right\|^2}, \tag{5.11}$$

where the numerator and denominator are known as the *sum of squares of residuals* and *total sum of squares*, respectively, and $\bar{\boldsymbol{v}}$ is the average linear velocity of the window. The metric R^2 is dimensionless and goes from one when the revolute kinematic model fits perfectly the data, to minus infinity when all velocities $\boldsymbol{v}_i$ are similar (e.g. in a prismatic joint) or when the model does not fit the data.

Metrics for prismatic: only one metric is used to classify the interaction type as prismatic, since only the orientation of the prismatic frame is estimated. This metric has the same geometrical interpretation as θ_{rev}, which also represents the angular variance (in radians) of the prismatic axis and is calculated as follows:

$$\theta_{\text{prism}} = \arctan\left(\sqrt{\frac{\lambda_{v_2} + \lambda_{v_3}}{\lambda_{v_1}}} \right), \tag{5.12}$$

where λ_{v_1}, λ_{v_2} and λ_{v_3} are the singular values of matrix $\boldsymbol{C}_v$, with units $[\frac{m^2}{s^2}]$, in decreasing order.

With the error metrics θ_{rev}, R^2 and θ_{prism}, the classification can be done based on the corresponding thresholds $\theta_{\text{rev}}^{\text{th}}$, R_{th}^2, and $\theta_{\text{prism}}^{\text{th}}$ as follows:

[5] The sum of squares of residuals of R^2 can be efficiently computed by expanding the terms $\sum \left\| \boldsymbol{\omega}_i^T \hat{\boldsymbol{n}}_t [\hat{\boldsymbol{n}}_t]_\times \hat{\boldsymbol{p}} + \boldsymbol{v}_i \right\|^2 = -(\hat{\boldsymbol{n}}_t^T \left(\sum \boldsymbol{\omega}_i \boldsymbol{\omega}_i^T \right) \hat{\boldsymbol{n}}_t) \hat{\boldsymbol{p}}^T [\hat{\boldsymbol{n}}_t]_\times [\hat{\boldsymbol{n}}_t]_\times \hat{\boldsymbol{p}} + 2\hat{\boldsymbol{n}}_t^T \left(\sum \boldsymbol{\omega}_i \boldsymbol{v}_i^T \right) [\hat{\boldsymbol{n}}_t]_\times \hat{\boldsymbol{p}} + \sum \boldsymbol{v}_i^T \boldsymbol{v}_i$ and $\sum \left\| \bar{\boldsymbol{v}} - \boldsymbol{v}_i \right\|^2 = |\boldsymbol{\Omega}| \bar{\boldsymbol{v}}^T \bar{\boldsymbol{v}} - 2\bar{\boldsymbol{v}}^T \sum \boldsymbol{v}_i + \sum \boldsymbol{v}_i^T \boldsymbol{v}_i$, where the superscript *rev* and *tcp* were omitted for compactness. Notice that all summations can be easily updated by subtracting the corresponding oldest term and adding the newest, without the need to recompute the summation for the whole window. The same can also be done for the denominator.

$$\text{interaction type} := \begin{cases} \text{revolute}, & \text{if } (R^2 > R_{\text{th}}^2) \text{ and } (\theta_{\text{rev}} < \theta_{\text{rev}}^{\text{th}}) \\ \text{prismatic}, & \text{if } \theta_{\text{prism}} < \theta_{\text{prism}}^{\text{th}} \\ \text{free space}, & \text{otherwise} \end{cases}$$

Step 3 - Smoothing:

In this step the task frame is smoothly adapted accordingly to the previous classification towards a target frame with position ${}_{\text{tcp}}\boldsymbol{p}_{\text{target}}^{\text{tcp,tf}}$ and with orientation ${}_{\text{tcp}}^{\text{tf}}\mathbf{R}_{\text{target}}$. If the interaction type is classified as revolute or prismatic, then the target frame will be selected as the revolute frame or the prismatic frame, respectively, which were calculated in *step 1*. Otherwise, the target frame is selected as the {tcp}. The smooth adaptation of {tf} towards the target frame is done using the following first order discrete dynamics (at time t), avoiding discontinuities caused by the classification:

$$\boldsymbol{p}_t = \boldsymbol{p}_{t-1} + k_{pos}(\boldsymbol{p}_{\text{target}} - \boldsymbol{p}_{t-1})\Delta t, \tag{5.13}$$

$$\mathbf{R}_t = \mathbf{R}_{t-1} \exp\left(k_{rot}[\hat{\boldsymbol{n}}_e]_\times \theta_e \Delta t\right), \tag{5.14}$$

where the sub/superscripts of ${}_{\text{tcp}}\boldsymbol{p}_t^{\text{tcp,tf}}$ and ${}_{\text{tcp}}^{\text{tf}}\mathbf{R}_t$ are omitted for clarity, k_{pos} and k_{rot} (with s^{-1} as units) are proportional control gains for smoothly adapting the position and the orientation, respectively, of {tf}. The normalized axis $\hat{\boldsymbol{n}}_e$ and angle θ_e correspond to the axis-angle representation of the orientation error between the target task frame and the task frame itself:

$$[\hat{\boldsymbol{n}}_e]_\times \theta_e = \log\left({}_{\text{tcp}}^{\text{tf}}\mathbf{R}_{\text{t-1}}^{-1}\ {}_{\text{tcp}}^{\text{tf}}\mathbf{R}_{\text{target}}\right). \tag{5.15}$$

The task frame will be used in the next section for assisting the user during tasks that involve the interaction with AOs. The assistance is therefore affected by estimation errors, e.g. caused by disturbances related to deformations of flexible AO, but these are compensated by the wrench controller.

5.4.3 Automatic assistance module

input: adaptive task frame {tf} and interaction type (revolute, prismatic or free space), both provided by the online interactive perception module,and joystick input.

output: assisted twist ${}_{\text{tcp}}\mathbf{t}_{\text{a}}{}^{\text{tcp}}$.

The automatic assistance module provides assistance to the user when the interaction type is either revolute or prismatic, and automatically deactivates the assistance when the interaction type is free space. This assistance is generated by following three steps: **interpretation step**, **downscaling step** and **twist generation step**. In the interpretation step, the joystick input is interpreted as a wrench that is in a normalized and dimensionless spherical space. Then, in the downscaling step, the interpreted wrench is transformed to the adaptive task frame (determined in the previous module) and downscaled in certain axes of the task frame that, otherwise, would produce reaction forces/torques or motion that is not compatible with the kinematics associated to the interaction type. Subsequently, in the twist generation step, this downscaled and dimensionless wrench is re-interpreted as a twist, then transformed to the tcp, and finally scaled so that a physically meaningful assisted twist (module output) is generated. Each of the steps is explained in the remaining part of subsection.

Interpretation step: the module starts by interpreting the joystick inputs as proportional desired forces/torques that the user wants to exert to the end effector (by exerting forces/torques to the joystick), as done in [106]. These force/torque inputs are scaled to a dimensionless space, such that the force term $\boldsymbol{F}_{\text{joy}} \in \mathbb{R}^3$ corresponding to translation input and the torque term $\boldsymbol{T}_{\text{joy}} \in \mathbb{R}^3$ corresponding to rotation input, have maximum norms of one (i.e. bounded by unitary-radius spheres). The maximum values $\|\boldsymbol{F}_{\text{joy}}\| = 1$ or $\|\boldsymbol{T}_{\text{joy}}\| = 1$ occur when the user translates or rotates, respectively, the joystick to the corresponding maximum physical limits of the joystick and decrease proportionally to zero when the user does not exert force/torque. On the other hand, the direction of $\boldsymbol{F}_{\text{joy}}$ and $\boldsymbol{T}_{\text{joy}}$ can be interpreted as the desired proportion of force/torque in each of the three axes of the joystick frame. These terms are organized in a wrench ${}_{\text{jf}}\mathbf{w}_{\text{joy}}{}^{\text{tcp}} = [\boldsymbol{F}_{\text{joy}}\ \boldsymbol{T}_{\text{joy}}]^T$ and are interpreted as acting upon the tcp frame {tcp} and expressed in the joystick frame {jf}.

Downscaling step: to provide assistance, first the ${}_{\text{jf}}\mathbf{w}_{\text{joy}}{}^{\text{tcp}}$ is transformed

to a wrench expressed in the task frame {tf} and acting upon the origin of {tf} as follows:

$$ {}_{\text{tf}}\mathbf{w}_{\text{joy}}{}^{\text{tf}} = {}^{\text{tcp}}_{\text{tf}}\tilde{\mathbf{S}}\ {}^{\text{jf}}_{\text{tcp}}\mathbf{P}\ {}_{\text{jf}}\mathbf{w}_{\text{joy}}{}^{\text{tcp}}, \tag{5.16} $$

where ${}^{\text{tcp}}_{\text{tf}}\tilde{\mathbf{S}}$ is a modified screw transformation that scales by a factor of $\beta[1/m]$ the torque that the force of the wrench causes due to the lower-left term of the screw transformation[6]:

$$ {}^{\text{tcp}}_{\text{tf}}\tilde{\mathbf{S}} = \begin{bmatrix} {}^{\text{tcp}}_{\text{tf}}\boldsymbol{R} & \mathbf{0}_3 \\ \beta \left[{}_{\text{tf}}\boldsymbol{p}^{\text{tf,tcp}} \right]_{\times} {}^{\text{tcp}}_{\text{tf}}\boldsymbol{R} & {}^{\text{tcp}}_{\text{tf}}\boldsymbol{R} \end{bmatrix}. \tag{5.17} $$

Because of such screw transformation, the linear terms $\boldsymbol{F}_{\text{joy}}$ will not only cause a translation but also a rotation with respect to the task frame in case that $\left\| {}_{\text{tf}}\boldsymbol{p}^{\text{tf,tcp}} \right\| > 0$. The factor β specifies how fast the revolute joint could be rotated by force joystick inputs $\boldsymbol{F}_{\text{joy}}$. In prismatic joints moments are not relevant, and therefore this factor is only plays a role when the interaction type is revolute.

Since ${}_{\text{tf}}\mathbf{w}_{\text{joy}}{}^{\text{tf}}$ is expressed at the task frame, we know that:

- If the interaction type is prismatic, only the force along the $\boldsymbol{z}_{\text{pf}}$ axis contributes to a proper interaction.
- If the interaction type is revolute, only the moment along the $\boldsymbol{z}_{\text{rf}}$ axis contributes to a proper interaction.

In order to assist users while keeping them in (partial) control in all directions, the module downscales (instead of completely remove) the force/moment components of ${}_{\text{tf}}\mathbf{w}_{\text{joy}}{}^{\text{tf}}$ that do not contribute to the correct interaction of the AO as follows:

[6] β can be interpreted as the ratio between the maximum physical limits of $\boldsymbol{T}_{\text{joy}}$ and $\boldsymbol{F}_{\text{joy}}$ mentioned in the previous paragraphs. In the user study, these maximum physical limits were not determined experimentally in terms of numerical values, i.e. the compliance matrix of the joystick was not identified experimentally. However, typical load cells in robotic applications have a ratio between moment and force ranges in the order of 10 $[1/m]$, and therefore this value was selected in the user study.

$$
\begin{aligned}
{}_{\text{tf}}\mathbf{w}_{\text{D}}{}^{\text{tf}} &= \boldsymbol{M}_{\text{sel}}\,{}_{\text{tf}}\mathbf{w}_{\text{joy}}{}^{\text{tf}}, \\
\boldsymbol{M}_{\text{sel}} &= diag([\alpha_t^{\perp}, \alpha_t^{\perp}, \alpha_t^{\text{prism}}, \alpha_t^{\perp}, \alpha_t^{\perp}, \alpha_t^{\text{rev}}]),
\end{aligned}
\tag{5.18}
$$

where D = downscaled and the selection matrix $\boldsymbol{M}_{\text{sel}} \in \mathbb{R}^{6\times 6}$ is a non-negative diagonal matrix with the selection parameters $\boldsymbol{\alpha}_t = \begin{bmatrix}\alpha_t^{\perp} & \alpha_t^{\text{prism}} & \alpha_t^{\text{rev}}\end{bmatrix}^T \leq [1\ 1\ 1]^T$ in its diagonal that depend on the interaction type at the current time t ($\perp$= orthogonal). The target values $\boldsymbol{\alpha}_{\text{target}} = \begin{bmatrix}\alpha_{\text{target}}^{\perp} & \alpha_{\text{target}}^{\text{prism}} & \alpha_{\text{target}}^{\text{rev}}\end{bmatrix}^T$ to which $\boldsymbol{\alpha}_t$ will adapt depending on the interaction type are shown in table 5.1, where α_{assist} is a parameter that indicates how much assistance the system should give (e.g. $\alpha_{\text{assist}} = 0.9$ means 90% assistance that downscales by 0.1). This smooth adaptation is performed with the following first order dynamics:

$$
\boldsymbol{\alpha}_t = \boldsymbol{\alpha}_{t-1} + k_{\alpha}(\boldsymbol{\alpha}_{\text{target}} - \boldsymbol{\alpha}_{t-1})\Delta t, \tag{5.19}
$$

where k_{α} is the corresponding proportional gain in $[1/s]$.

Table 5.1. Chosen selection parameters that compose the selection matrix for each interaction type, where $0 < \alpha_{\text{assist}} < 1$.

Interaction type	$\alpha_{\text{target}}^{\perp}$	$\alpha_{\text{target}}^{\text{rev}}$	$\alpha_{\text{target}}^{\text{prism}}$
revolute	$1-\alpha_{\text{assist}}$	1	$1-\alpha_{\text{assist}}$
prismatic	$1-\alpha_{\text{assist}}$	$1-\alpha_{\text{assist}}$	1
free space	1	1	1

Twist generation step: the velocity-resolved admittance control in (5.4) expects a twist ${}_{\text{tcp}}\mathbf{t}_{\text{a}}{}^{\text{tcp}}$ (a=assisted) expressed in the tcp. To translate the downscaled wrench to a twist, we assume a compliance matrix that is scaled such that the non-downscaled translational velocity limit at the tcp and the non-downscaled rotational velocity ω_{limit} never exceed their corresponding limits v_{limit} and ω_{limit} (achieved through scale factor γ_{limit}), while remaining proportional to the joystick input (achieved through scale factor γ_{joy}). The twist is generated with this compliance matrix in the following way:

$$
{}_{\text{tcp}}\mathbf{t}_{\text{a}}{}^{\text{tcp}} = {}_{\text{tcp}}^{\text{tf}}\mathbf{S}\left(\gamma_{\text{joy}}\ \gamma_{\text{limit}}\ \tilde{\Delta}\right)\ {}_{\text{tf}}\mathbf{w}_{\text{D}}{}^{\text{tf}}, \tag{5.20}
$$

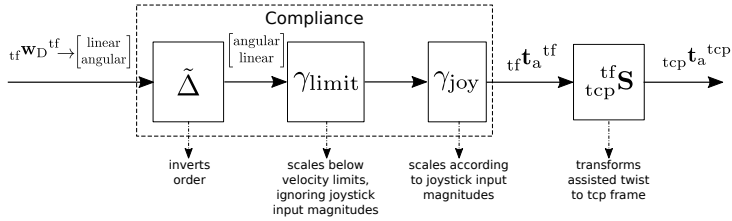


Figure 5.4. Block diagram illustrating the twist generation step of the automatic assistance module.

where the matrix inside the parenthesis can be interpreted as the aforementioned compliance that maps the wrench to a twist expressed in the task frame. This compliance matrix is composed by the two dimensionless scale factors γ_{limit} and γ_{joy}, and matrix $\tilde{\Delta}$, which inverts the order of the linear and angular terms of the wrench. The upper-left identity matrix of $\tilde{\Delta}$, which is determined with (5.3), has units of $[\frac{rad}{s}]$ and the lower-right identity matrix has units of [m/s]. The process is illustrated in Fig. 5.4.5.4

To compute the factor γ_{limit}, the twist at the tcp when the selection matrix is not used is computed as follows:

$$ {}_{\text{tcp}}\mathbf{t}_{\text{ND}}{}^{\text{tcp}} = \left[\boldsymbol{\omega}_{\text{ND}}{}^{T}\ \boldsymbol{v}_{\text{ND}}{}^{T}\right]^{T} = {}^{\text{tf}}_{\text{tcp}}\mathbf{S}\,\tilde{\Delta}\ {}_{\text{tf}}\mathbf{w}_{\text{joy}}{}^{\text{tf}}, \tag{5.21} $$

where $\boldsymbol{\omega}_{\text{ND}}$ and $\boldsymbol{v}_{\text{ND}}$ are respectively the rotational and translational velocities of the joystick expressed in the tcp, with the *tcp* indices dropped for compactness. $\boldsymbol{\omega}_{\text{ND}}$ and $\boldsymbol{v}_{\text{ND}}$ are non-downscaled (ND) since (5.18) is not applied. Factor γ_{limit} then gets a value of ${}^{v_{\text{limit}}}/_{\|\boldsymbol{v}_{\text{ND}}\|}$ or ${}^{\omega_{\text{limit}}}/_{\|\boldsymbol{\omega}_{\text{ND}}\|}$ (the smallest one is chosen), to ensure that both angular and linear velocities are below their specified limit parameters, which are denoted as ω_{limit} and v_{limit}, respectively. It is calculated as follows to avoid dividing by zero when the user does not move the joystick:

$$ \gamma_{\text{limit}}^{-1} = \max\left(\frac{\|\boldsymbol{v}_{\text{ND}}\|}{v_{\text{limit}}}, \frac{\|\boldsymbol{\omega}_{\text{ND}}\|}{\omega_{\text{limit}}}, \delta\right), \tag{5.22} $$

where δ is a very small number[7] that is only used to avoid dividing by zero when calculating γ_{limit} from (5.22).

[7]The exact value of δ is not relevant because only when ${}_{\text{tf}}\mathbf{w}_{\text{D}}{}^{\text{tf}}$ in (5.20) is a zero wrench, γ_{limit} will have a value of $1/\delta$ that will cancel out.

The second factor γ_{joy} proportionally scales the twist to the most dominant (normalized and dimensionless) joystick input: $\boldsymbol{F}_{\text{joy}}$ or $\boldsymbol{T}_{\text{joy}}$. Factor γ_{joy} allows the whole range of the joystick to be used, exploiting the full joystick resolution, and is calculated as follows:

$$\gamma_{\text{joy}} = \max\left(\left\|\boldsymbol{F}_{\text{joy}}\right\|, \left\|\boldsymbol{T}_{\text{joy}}\right\|\right). \tag{5.23}$$

The assisted twist ${}_{\text{tcp}}\mathbf{t}_{\text{a}}{}^{\text{tcp}}$ is executed by the velocity-resolved control module as explained before.

5.5 User Study

We performed a user study involving sixteen able-bodied participants (eleven male, five female), aged 21-44 (mean=30.2, standard deviation= 6.2). The procedure was approved by the Social and Societal Ethics Committee of KU Leuven after a GDPR assessment and ethical review. The study followed a within-subjects design, with only a binary factor (i.e. with or without assistance), and lasted up to one hour.

1) Tasks and apparatus: Each participant was instructed to complete two tasks (see Fig. 5.5), each with and without assistance, resulting in four trials, by teleoperating a robot arm with a SpaceMouse. The first task (referred to as "pump task") consisted in inflating a balloon using a tilted bicycle pump (i.e. prismatic joint), where the task was completed when the participant achieved 5 inflating cycles. One cycle consisted of going from retracted to extended and back to retracted. Two yellow stroke delimiters (approximately 30 cm apart from each other) were placed on the rod to indicate when the pump reached the extended and retracted states. The second task (referred to as "box task") consisted in manipulating the lid of a lightweight cardboard box and performing 5 cycles. One cycle consisted of taking the lid from a horizontal state (0 rad) to a vertical state ($\pi/2$ rad) and again back to horizontal. Two angle delimiters were used to indicate when the lid reached the horizontal and vertical states. The goal for both tasks was performing the cycles as fast as possible by leaving the base of the objects as steady as possible with respect to the table. The participants were asked to give more priority to not moving the box over completing the task in a shorter time.

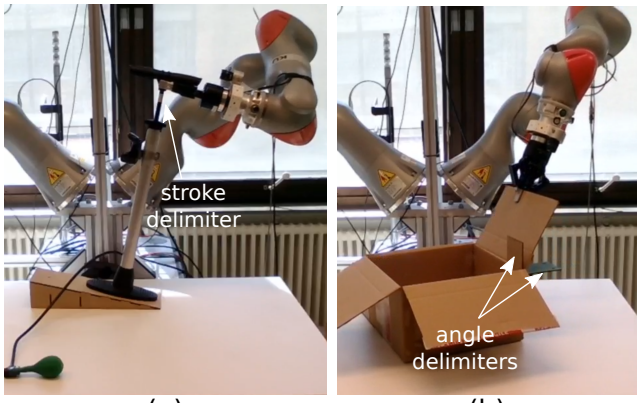


Figure 5.5. Tasks performed by the participants during the user study, (a) *pump task* with a bicycle pump (i.e. prismatic joint) that is tilted 13 degrees with respect to the table and has a balloon attached that is inflated in the process and, (b) *box task* with a lightweight empty cardboard box and a lid that rotates (i.e. revolute joint).

Participants were seated and observing the robot scene by means of a screen. The robot scene was captured by a full HD webcam, as shown in Fig. 5.6. The task was performed with a commercial KUKA LBR iiwa robot with a JR3 force/torque sensor mounted in the end effector. The robot was configured to follow joint velocities sent at 200 Hz.

The two variations of the trials (i.e. with assistance and without assistance) were executed with $\alpha_{\text{assist}} = 0.9$ and $\alpha_{\text{assist}} = 0$, respectively. While the task frame during the trials with assistance was adaptive and computed as explained in subsection 5.4.2, the task frame during the trials without assistance was kept fixed at the {tcp} frame. Apart from these two differences, both tasks were executed in the same way and with the same parameters; including the same wrench controller, as explained in subsection 5.4.1.

A button in the SpaceMouse was configured to open or close the gripper. A second button, referred to as the reset button, was configured to reset the estimation (i.e. erasing the data in the sliding window and starting from scratch with the estimation). This was done for two purposes: *(i)* to deal with bad assistance caused by misclassifications; and *(ii)* to register when the participant experienced a confusion when controling the robot (for both, with and without assistance).

The prior joint axes used in the estimation were arbitrarily chosen as

$\hat{\boldsymbol{n}}_{\text{pr}}^{\text{prism}} = \hat{\boldsymbol{k}}$ and $\hat{\boldsymbol{n}}_{\text{pr}}^{\text{rev}} = \hat{\boldsymbol{i}}$ to coincide with the Z and X axis of the tcp, respectively. The prior for the origin of {rf} was set to a zero vector so that it coincides with the origin of {tcp} (i.e. $\boldsymbol{p}_{\text{pr}}^{\text{rev}} := {}_{\text{tcp}}\boldsymbol{p}^{\text{tcp,tcp}} = \boldsymbol{0}$).

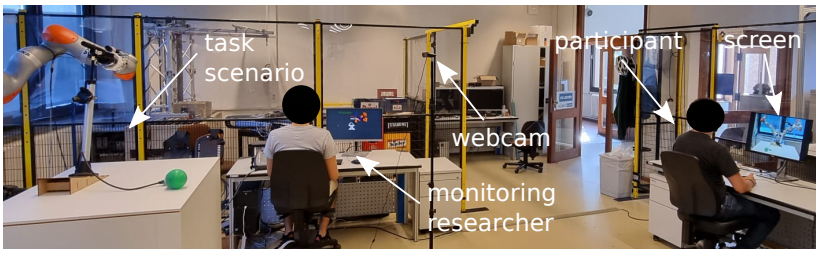


Figure 5.6. Photo showing the setup arrangement in the laboratory for the user study. The participants were seated and teleoperating the robot with the SpaceMouse by looking at a screen (which was showing the video streamed by the webcam). The robot was located behind participants, who were not allowed to turn and look at it directly.

2) Procedure: Following an informed consent, participants were shown an interactive eight-minute-long video tutorial where they learned how to control the robot in free space with the SpaceMouse (including how to use the two buttons). After practicing while watching the video, the participants were asked to perform both tasks with and without assistance. At the beginning of each task, the participants watched a one or two-minute video tutorial which explained how to perform the task. Subsequently, they were instructed to practice by performing three and five cycles for the box and the pump tasks, respectively. After the practice, the participants proceeded to execute the task with both variations and do a subjective evaluation right after finishing each variation. The task begins with the gripper in place to grasp the AO and the user begins the task by closing the gripper. Right after they executed both variations, they were asked to indicate their preferred method and argue their choice.

To reduce order effects, half of the participants first performed the pump task, while the other half first performed the box task, with varying order of executions with assistance and without assistance. This results in eight possible permutations that were secretly assigned to the sixteen participants. None of the participant was informed about the differences between the different trial executions. The starting position of the objects was approximately the same for all trials

3) Hypotheses: The study consisted on getting several qualitative and quantitative metrics for two tasks, each tested with and without assistance.

A lower metric signifies an improvement in execution of the task. The following null hypotheses were evaluated for each metric:

- $\boldsymbol{H_0^1}$: Assistance does not result in a lower metric for the task of the pump
- $\boldsymbol{H_0^2}$: Assistance does not result in a lower metric for the task of the box
- $\boldsymbol{H_0^3}$: Assistance does not result in a lower metric for both tasks combined (without distinction).

4) Qualitative metrics and results: After completing each variation of the task, participants filled out the NASA Task Load Index (TLX) questionnaire [47], which consists of two parts. In the first part, the participants gave a score (with 0-100 range and 5-point step) to six workload-related subscales (mental demand, physical demand, temporal demand, performance, effort and frustration) and the average, referred to as the TLX, is computed. In the second part, the participants compare all the subscales pairwise based on their perceived importance and a weight is obtained (see Fig. 5.7). This weight is then used to compute a weighted average of the scores, referred to as the weighted TLX, which takes into account this perceived importance. The individual scores for the subscales, the TLX and the weighted TLX are all used as qualitative metrics for the study.

The data related to all qualitative metrics can be visualized in Fig. 5.8. It can be observed that, overall, the scores tend to be lower (i.e. better) for the tasks executed with assistance. To confirm this, a one-tailed paired-sample t-test was used in order to accept the alternative hypotheses (i.e. reject the null hypotheses) for each of the metrics. The results of this analysis can be seen in table 5.2, where for most of the metrics the test confirms that participants perceive improvement when performing the task with assistance.

Finally, a pie chart indicating the preferred methods of the participant can be observed in Fig. 5.9. It is clearly seen that most of the participants preferred the method with the assistance. Participant 4 and participant 8 indicated that there was no preference on the method, for the pump and the box tasks, respectively, since they could not perceive any major difference. No participants indicated a preference for the pump task without assistance and only one participant (participant 5) indicated a preference for the box task without assistance.

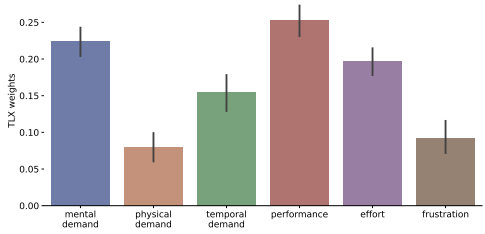


Figure 5.7. Average of the NASA TLX weights determined by the second part of the questionnaire, indicating the perceived importance of each of the subscales. The error bars represent a 95% confidence interval.

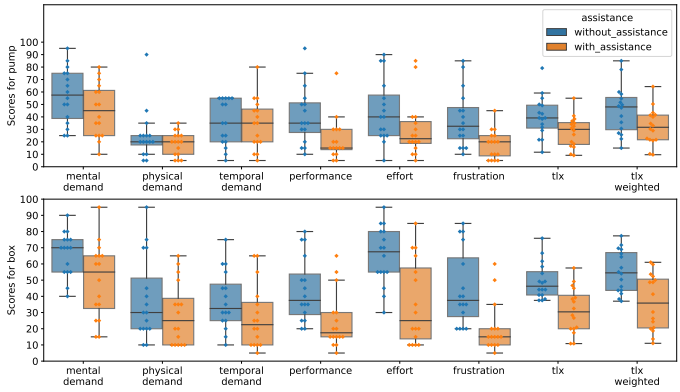


Figure 5.8. Data overlaid on a Boxplot for each qualitative metric, comparing the two variations of the tasks (with and without assistance). The six subscales of the questionnaire are shown individually and also the (raw) TLX and the weighted TLX are shown.

5) Quantitative metrics and results: The following three performance metrics were computed: *(i) task duration*, measured from the moment in which the participant pressed the button to close the gripper to grasp the object until the moment in which the participant pressed the button to open the gripper after completing the five cycles; *(ii) traveled distance*, calculated as the sum of all consecutive translational displacements of the base of the manipulated object; and *(iii) traveled angle*, calculated as the sum of all angles (in axis-angle representation) between all consecutive rotational displacements of the base of the manipulated object. The last two metrics were calculated based on post-processed pose signals coming from

Table 5.2. One-tailed paired-sample t-test analysis for the qualitative metrics based on the NASA-TLX questionnaire. The p-value (p) and the t-test statistic (stat) are both shown. The "Both" column was calculated by combining the data of both tasks (i.e. not discriminating between tasks). The tests that rejected the null hypotheses with a significance level of 5% (i.e. p<0.05), and therefore indicate an improvement, are indicated in bold.

Metric	Pump (H_0^1)	Box (H_0^2)	Both (H_0^3)
mental demand	**p=0.002, stat=3.33**	**p=0.002, stat=3.45**	**p=0.000, stat=4.78**
physical demand	p=0.057, stat=1.68	**p=0.043, stat=1.83**	**p=0.009, stat=2.51**
temporal demand	p=0.404, stat=0.25	**p=0.035, stat=1.95**	p=0.051, stat=1.69
performance	**p=0.000, stat=5.31**	**p=0.000, stat=4.58**	**p=0.000, stat=7.05**
effort	**p=0.003, stat=3.28**	**p=0.000, stat=5.52**	**p=0.000, stat=6.01**
frustration	**p=0.000, stat=4.39**	**p=0.000, stat=4.26**	**p=0.000, stat=6.14**
tlx	**p=0.000, stat=4.58**	**p=0.000, stat=5.11**	**p=0.000, stat=6.75**
tlx weighted	**p=0.000, stat=5.08**	**p=0.000, stat=4.24**	**p=0.000, stat=6.21**

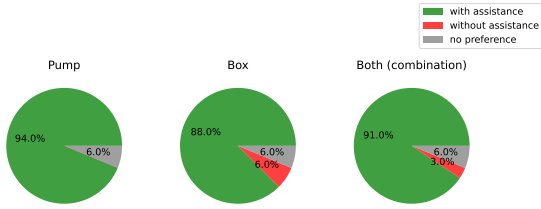


Figure 5.9. Pie chart showing percentages of the preferences of the participants for both tasks. The pie chart combining both answers (i.e. not discriminating between tasks) is also shown. 6% represents only one participant.

HTC-Vive trackers that were attached to the base of the object and calibrated to report the location and orientation of the corresponding articulation.

All quantitative metrics can be visualized and compared in Fig. 5.10. It can be observed that, overall, the scores tend to be lower (i.e. better) for the tasks executed with assistance. In addition, the difference between the metrics with assistance and without assistance is shown in Fig. 5.11 for the pump task and Fig. 5.12 for the box task. It can be observed that in all the trials (with one small exception for participant 3), the participants performed better (green bars) in terms of the traveled distance and traveled angle metrics. On the other hand, it can be observed that for most of

the trials the participants performed better in terms of task duration with assistance and only in few cases they performed worse (red bars).

To confirm the improvement related to the metrics, a one-tailed paired-sample t-test was used in order to accept the alternative hypotheses (i.e. reject the null hypotheses) for each of the quantitative metrics. The results of this analysis can be seen in table 5.3, where for **all** metrics the test confirms that the participants achieve a performance improvement when performing the task with assistance.

As an additional quantitative metric, the number of times that the participants pressed the reset button during the trials was recorded. The participants were instructed to press the reset button only when they felt that the robot was not responding properly to their commands. Only in three out of the sixty-four trials the reset button was pressed. However, only two of those occurred when the task was being performed with assistance. Participant 2 pressed only one time the button during the pump with assistance, because a misclassification where a revolute was detected occurred. This happened because the participant was tilting a lot the pump with respect to its based, causing the algorithm to activate its assistance for revolute joint. Participant 4 pressed three times the button during the box task without assistance, probably because he/she felt confused (without assistance the participant was in full control and the button had no effect). Finally, participant 8 pressed one time during the box task with assistance. There was no misclassification when this happened and, according to the video recording of the task, the participant seemed confused on how to do the task for a moment.

Table 5.3. One-tailed paired-sample t-test analysis for the quantitative metrics. The traveled distance and angle from participant 4 was not taken into account for the test of the box task since the tracker was lost in the task without assistance. The tests that reject the null hypothesis and thus accept the alternative hypothesis, with a significance of 5% (i.e. $p<0.05$), are indicated in bold.

Metric	Pump	Box	Both
task durations	**p=0.030, stat=2.03**	**p=0.003, stat=3.14**	**p=0.000, stat=3.66**
traveled distances	**p=0.001, stat=3.99**	**p=0.000, stat=6.15**	**p=0.000, stat=7.04**
traveled angles	**p=0.000, stat=4.47**	**p=0.000, stat=7.66**	**p=0.000, stat=7.42**

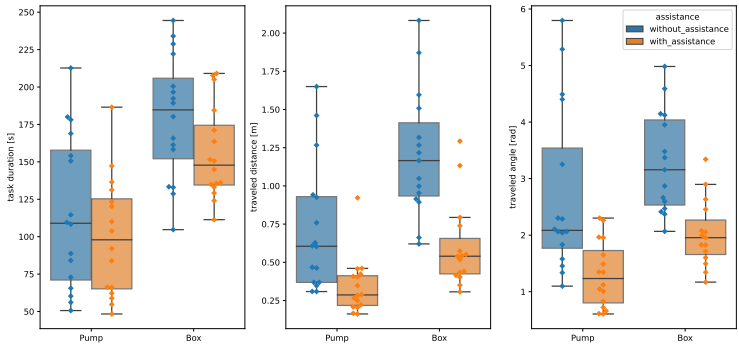


Figure 5.10. Data overlaid on a Boxplot for each quantitative metric, comparing the two variations of the tasks (with and without assistance).

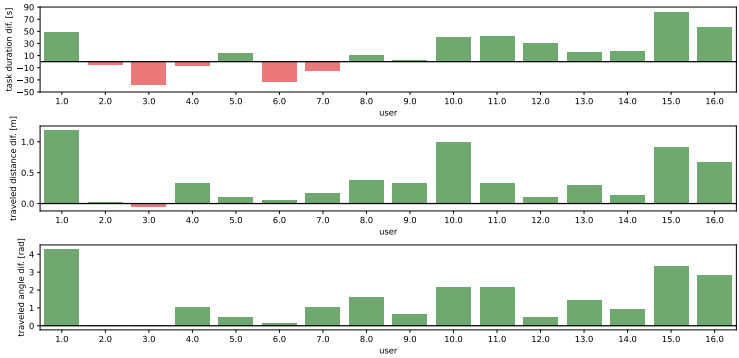


Figure 5.11. Difference between the quantitative metrics between executions without and with assistance for the **pump** task for each participant. Positive values (depicted in green) can be interpreted as improvements observed in the execution with assistance.

5.6 Discussion and Conclusion

In this work, we introduced a shared-autonomy method for assisting users during tasks that involve manipulating articulated objects with unknown geometric properties in unstructured environments. A user study on able-bodied participants, performed with a pump (for prismatic interaction) and a box (for revolute interaction), supports that our method offers a

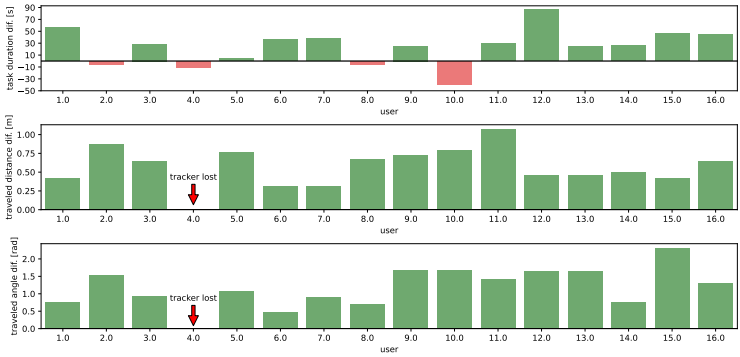


Figure 5.12. Difference between the quantitative metrics between executions without and with assistance for the **box** task for each participant. Positive values (depicted in green) can be interpreted as improvements observed in the execution with assistance.

significant improvement in terms of quantitative and qualitative metrics. Even if our work was motivated to help people with motor impairments, it can also be reconfigured to other contexts (e.g. teleoperation in hazardous environments) with other human-machine interfaces that adjust better to the task and user requirements (e.g. with haptic feedback).

The online interactive perception (*OIP*) module and the velocity-resolved control module (that includes a wrench controller) can be seen as two mutualistic modules that improve each others performance. In other words, the OIP improves with the motion corrections done by compliant behavior of the wrench controller; and the wrench controller achieves less zero-wrench tracking error if the provided motion is consistent with the kinematics of the AO thanks to the OIP. This synergy between gathered information through OIP and control performance influences the speed with which the user can complete the task. Thus, there is no difference between the beginning of the task with assistance (where there is no information available yet) and the complete task without assistance. In these situations the users need more effort and time to perform the task and, in the case of the assistance, this is alleviated automatically once enough information is gathered.

Even if we focused our attention on prismatic and revolute joints, most of the higher dimensional joints (e.g. spherical joint) and commonly found AOs can be modeled as kinematic chains composed by both. Thus, in some cases it would still be possible to obtain assistance through the same methods by

manipulating one axis or joint at a time. For example, for opening the tray in the back of an airplane seat (prismatic + revolute), the user can sequentially rotate the table (revolute) and then bring the table closer (prismatic). When the user wants to interact with another axis or joint, it would still be possible to correct by pressing the reset button or by fighting the assistance and let the task frame to automatically adapt (if $\alpha_{\text{assist}} < 1$). However, this was not tested and it is out of the scope of this work.

The same correction mechanics can be applied if a misclassification occurs. However, the only problematic misclassification that would cause user confusion is when the interaction type is classified as a revolute instead of a prismatic. If the opposite occurs (classified as prismatic instead of revolute), the given assistance would still contribute to achieve a better performance by maintaining a tangential velocity for the articulation and the wrench controller would cause the gripper to re-orient automatically (with the trade-off of having to move slower).

The assistance, including the choice of the task frame, does not influence the interpretation (i.e. mental model) of the user regarding the joystick usage (if there are no misclassifications), making it transparent and intuitive for the users. This is confirmed by the user study since the participants did not receive any additional explanation about the assistance, and they were not aware of which method they were using during the trials. Even without this knowledge, most of the participants preferred doing the task with assistance. In addition they were able to perform better in all quantitative metrics and perceived a workload improvement in terms the qualitative metrics. Such claim is also confirmed in the number of times that the participants pressed the reset button during the tasks when they felt that the robot was not responding correctly to their commands.

Even if it is not strictly needed, in order to achieve a higher technology readiness level and bringing the method closer to real users as future work, it is best to inform the users the moment in which they start being assisted and how they are assisted (e.g. by informing them that a revolute or a prismatic joint was detected). The user then would be able to understand better the situation (avoiding confusion in case of misclassifications) and to accept or reject the suggested assistance. In addition, allowing the user to customize the assistance by setting α_{assist} would certainly improve the overall user experience. In our opinion these modifications would help to improve the situation awareness [38], which can be enabled due to the explainability of our method.

As additional future work, a better classifier (e.g. data driven) would be beneficial for avoiding misclassifications instead of relying on thresholds that do not generalize so well to different objects and situations. The used threshold-based classifier was merely used as a proof of concept and it can be replaced by a more robust technique.

Chapter 6

Conclusions and Limitations

This chapter summarizes the contributions of the thesis, discusses its limitations and proposes future work to address these limitations.

Contribution 1 – reconfigurable and adaptable assistive framework

Chapter 3 presents an assistive framework that is:

1. *reconfigurable,* by allowing offline customization for different users through the combination of different types of HMIs, HMI transducer interaction modalities, robotic platforms, assistive strategies, levels of autonomy, and physically-interpretable parameters; and
2. *adaptable,* by enabling a systematic way of transitioning online between levels of autonomy over decoupled DOF.

Two types of building blocks (BBs) allow the specification of interaction with sensors and HMIs (SIMs) and the specification of modeled reactive behaviors (RAMs) in a decoupled way. Therefore, existing BBs can be (re-)combined to create novel assistive strategies, reducing the required development and deployment time and effort. These assistive strategies can easily be reconfigured by adding, removing or exchanging BBs. For

example, adding/exchanging the teleoperation HMI for one that is preferred by a particular user can be done by adding/exchanging the corresponding SIM, if the transducer of the new HMI has a different output interpretation. If the interaction modalities of both HMI transducers have the same output interpretation (e.g. an end-effector-mounted load cell and a SpaceMouse controlling the velocity in the tcp frame) the same SIM can be used.

The two key features of the framework (reconfigurability and adaptability) are extensively demonstrated by proposing several assistive behaviors for activities of daily living (contribution 2) and validating them in heterogeneous configurations (contribution 3).

Contribution 2 – assistive behaviors for activities of daily living

This thesis proposes several assistive behaviors that are encoded in BBs and are then (re-)configured to create different user-centered assistive strategies that assist users for activities of daily living. System developers and application developers are free to create and combine, respectively, many BBs by following different assumptions that make them suitable for particular contexts. However, in this thesis, the focus is on activities of daily living that are assumed to occur in unstructured environments (i.e. only little information about the objects and the environment is known beforehand) so that, in the future, they can be applied outside laboratory conditions. Therefore, in the obtained library of BBs, most of the required information to provide assistance is either measured or estimated through the use of multiple sensors and HMIs, and used to reactively deal with changing environments and user intentions.

The framework is not limited to particular tasks, and therefore a wide variety of assistive strategies could be created from these BBs to assist users during many activities of daily living. Nevertheless, this thesis focuses on two types of tasks:

1. reaching or picking objects, where different assistive behaviors that are suitable for different targeted users and other conditions are showcased from UC1 to UC10 (see Table 1.2), and
2. interacting with articulated objects with unknown geometric properties as demonstrated in UC11 (with the proposed RAMASA assistive

strategy that is explained in chapter 5). Here, the sensors and motion are used to extract more information about the environment, i.e. the location and direction of the articulated joint axis.

The proposed assistive strategy for interacting with articulated objects was validated in a user study, where not only quantitative improvements related to performance-related metrics were demonstrated, but also qualitative (i.e. subjective) improvements related to workload metrics and preferences were demonstrated.

Contribution 3 – experimental validation of heterogeneous configurations

From chapter 2 to chapter 5, the capabilities of the framework in terms of reconfigurability, adaptability and reactiveness was extensively demonstrated by means of eleven use cases (see Table 1.2). These use cases were defined based on variations in terms of: environment description (e.g. objects placed on a table with dynamic obstacles being present), targeted users (e.g. tetraplegic), test users (i.e. able-bodied humans or monkeys), robotic devices (i.e. one out of the five tested robotic platforms), HMI(s) to interact with the system (e.g. BMI, eye tracker, etc), sensors to measure the environment (e.g. load cell, cameras, etc), autonomy (e.g. fully autonomous, adaptable levels of autonomy, etc), control modes (e.g. with or without whole-body impedance control) and activities of daily living (i.e. reaching objects and manipulating articulated objects).

This clearly shows that the framework enables the possibility to rapidly develop and deploy user-centered assistive solutions based on previously developed assistive building blocks. As a result, the framework facilitates the development and deployment of assistive applications, thereby reducing the required effort and time. This claim is emphasized in chapter 4 by presenting several use cases in the context of two assistive scenarios that were developed and deployed under strict timing requirements.

Limitations and future work

In this section several limitations and corresponding proposals on how to address these limitations in future work are discussed.

Limitation – feedback to users: The focus of this thesis regarding reconfiguration of HMIs was mainly on $H \rightarrow M$ transducers to command the robot at a high-level (e.g. voice commands, pressing buttons, etc) or a low-level (e.g. velocity control with a joystick or with a BMI). For this reason, throughout the presented use cases, only simple $M \rightarrow H$ transducers were used in order to provide the user with minimum required information. Nevertheless, as explained in [6], providing task-related and robot-related information to the user, in an intuitive and convenient way, is very important to improve situation awareness [38].
Future work: Hence, for increasing the TRL of assistive devices, more suitable $M \rightarrow H$ interfaces (e.g. augmented reality headset as done in [109]) alongside with more information provided to the user should also be explored and integrated to the proposed assistive strategies. For instance, the tablet incorporated in EDAN [103] not only enables the user to provide high-level commands to the robot ($H \rightarrow M$) but also enables the robot to provide information about the status of the assistive system ($M \rightarrow H$). The way in which users receive the information (e.g. visually through a tablet or AR headset, auditorily through speech, etc) should also be easily personalizable.

Limitation – haptic feedback: Taking into account the previous limitation, we acknowledge the fact that haptic interfaces provide force feedback (through $M \rightarrow H$ transducers) during teleoperation tasks and exhibit performance advantages. During the use cases HMIs with active haptic feedback were not used, since most of them are bulky, expensive, and are often not suitable for people with motor impairments. In UC3, however, haptic feedback was provided to the user who could feel the presence of the virtual fixtures, since the HMI interaction modality was based on kinesthetic guidance (not during teleoperation).
Future work: As an alternative to commonly used haptic devices, some interfaces such as the foldaway haptic joystick [87] show promising features that make them potential candidates to be used for these applications. Therefore, implementation of such haptic devices and related user studies are proposed as future work. In addition, commercially available haptic devices such as the Virtuose Haption, can be explored in other domains (e.g. teleoperation in hazardous environments) where the developed framework

and assistive strategies can still be applied. Alternatively, a second robot arm can be used as an HMIs with force haptic feedback, as done in the rehabilitation application presented in [28]. The integration with such HMIs that support force haptic feedback should be straightforward with the current BBs library, since all the necessary BBs are already available.

Limitation – prior information about the task: In this thesis prior information about the type of task at hand (e.g. reaching an object or manipulating an articulated object) is assumed to be known. In a real scenario, robotic devices need to be aware about the type of task that the user wants to perform, which frequently changes, in order to provide an appropriate assistance.
Future work: There are several ways in which the user could provide this information to the robotic device through an HMI with a $H \rightarrow M$ transducer. For example, through a touchscreen (e.g. tablet) the user could select the task out of a menu or, alternatively, the user could use voice commands to indicate it. Even if this solution addresses the problem in a simple way, work must be done such that the use of the HMI is intuitive and easy to use. Another possibility is that the robotic device can perform a symbolic inference of the desired task and either take the decision for the user, or make suggestions to the user who then would be able to confirm or reject the suggested assistance with a simpler HMI interaction type. The CATs approach [22] provides interesting insights of how to do this though the use of a symbolic planner specified through Planning Domain Definition Language (PDDL).

Limitation – intent estimation: The intent estimation scheme proposed in chapter 3 requires the use of two co-existing HMI interaction modalities, one for indicating approximately the target position and other for teleoperating the robot. The target indication can be done by the user implicitly (e.g. in UC5 with the eye tracker) or explicitly (e.g. in UC4 with the SpaceMouse). When the user indicates it explicitly, the user needs to spend some time and effort to indicate an approximate target position. Furthermore, in the BMI experiments related to UC7, the target position was not indicated by the user (i.e. the monkey), but instead the target position was indicated to the monkey with LED lights.
Future work: Intent estimation can be done based on the motion provided by the teleoperation HMI (e.g. SpaceMouse, BMI, etc), instead of requiring a separate HMI input. However, assumptions about the motion model that the user performs are needed and a suitable motion primitive/model (e.g. from the learning-from-demonstrations community) must be selected. This

is convenient because the user would be able to start moving (initially without assistance), and when enough information is gathered, assistance could be gradually granted to the user based on a predicted target position. This assistance could be achieved, for example, with the RVGFs proposed in chapter 3. Preliminary work related to this topic was already done in the master thesis [30] with a PPCA motion model, which was mentored by the author of this thesis.

Limitation – reconfiguration of control modes: Different control modes were used within the proposed use cases, especially on the ones presented with EDAN in section 4.3. While these use cases provide relevant insights about the importance of carefully choosing these control modes, the work performed around this topic was merely exploratory.
Future work: More careful studies and formalization in the context of user-centered robotics is suggested to exploit different control modes to achieve better performance and comply better to the user's needs and preferences.

Limitation – able-bodied users: In all the presented use cases, including UC11 where a user study was performed, able-bodied users (human or non-human primates) tested the different assistive strategies. The BMI experiments related to UC6 and UC7 were planned to be conducted with human patients with tetraplegia and epilepsy, but had to be canceled due to the Covid-19 pandemic. Thus, only non-human primate users were used during these experiments. For the remaining use cases, human able-bodied users were used due to all the practical and ethical difficulties related to finding people with motor impairments and performing experiments with them.
Future work: The suggestions for this limitation are twofold: 1) user studies with users with motor impairments should be done to demonstrate the suitability of the different assistive strategies developed in this thesis and to study the benefits that the reconfiguration (enabled by our framework) brings to them, and 2) more assistive strategies should be developed based on requirements expressed by users with motor impairments (e.g. as done in UC8-UC10 with EDAN). This requires more experience with the assistive devices and how users handle challenging problems using the provided assistive strategies. Competions such as CYBATHLON [53] can help gain experience, but typically require a wheelchair-mounted robot such as EDAN.

Limitation – allowing customization to users: In this work the reconfiguration capabilities of our framework were demonstrated by presenting several use cases that were reconfigured by mimicking the role of a solution integrator, but the reconfiguration that should be enabled

to caregivers and users was not demonstrated. Even if our framework facilitates such reconfiguration (e.g. by using only explainable models with physically-interpretable parameters and using composable SIMs), users were not asked to manually do *parameter reconfiguration* or *HMI reconfiguration* based on their preferences.
Future work: Regarding the *parameter reconfiguration*, some parameters should be exposed to the caregivers/users to be able to achieve customization and improve the experience of the assistance. Only a limited selection of these parameters should be exposed to the user. User studies should be done to understand which parameters are desirable for the user to change and which are easy to interpret by users with no technical background. Carefully choosing a selection of parameters to be exposed to the user, based on such user studies, are crucial for increasing the usability of personalizable assistive robots. In addition, intuitive mechanisms (e.g. user-friendly graphical user interfaces) should be designed such that users can comprehend the parameters and change them according to their needs and preferences. The exposed parameters should be presented in a hierarchical way, since some parameters are general for the robotic device, some are specific to the HMIs, and some are even more specific to each assistive strategy and task. Only parameters that are easy to understand should be exposed, and the values for all the more complex parameters should be selected by stakeholders with more technical expertise. On the other hand, for the *HMI reconfiguration*, users should be able to easily pair different types of HMIs and control the robotic device based on their preferences. Enabling the users to do this requires a higher TRL level but the concepts in this thesis, especially the division of the BBs into SIMs and RAMS, go already a long way towards the right direction.

Limitation – instantaneous control: Our framework is based on instantaneous (i.e. reactive) control that is subject to local minima and has performance limitations. In RAMo (chapter 2) and the RVGFs (chapter 3) a workaround of this limitation was done for collision avoidance by considering different instances of the tool along the path (i.e. looking into the future), so that the shape of the trajectory adapts before the robot encounters the obstacle. However, this solution is “custom-made" and is not generalizable to other assistive strategies.
Future work: An extension of the framework for including controllers that look into the near future within a finite time-horizon (e.g. with model predictive control) would be useful for the specification of some assistive strategies such that the system can give assistance proactively instead of just reactively. Such task specifications could be done by using approaches

such as the TASHO approach [88].

Limitation – time and effort reduction: We showed that the time and effort required when developing and deploying assistive devices could be reduced using the framework. This is expected to result in an improvement in the affordability of personalized assistive devices. However, it is out of the scope of the thesis to quantify the time and effort reduction and the corresponding impact on affordability.
Future work: A microeconomical study on how the time and effort reduction can improve the affordability of personalized assistive devices is suggested. Such study would be valuable for investors to take decisions.

Limitation – HMI configuration templates: The formalization given to HMI transducer configuration through the configuration template (meta-model), presented in subsection 1.4.1, is preliminary and needs further investigation. This configuration template was only used for explanation purposes within this thesis and no implementation that exploits the structure of the template was made.
Future work: Further work on the use of the proposed HMI configuration template is recommended. Firstly, a literature review to see how well the configuration template adjusts to different HMI transducers should be done. Secondly, the template should be formalized as JSON Schema to exactly describe the format of the JSON transducer configurations. This offers a more formal structure, documentation and validation. Thirdly, a library that reads JSON transducer configurations and deploys the HMI transducer with the proper configuration can be developed. This would be the final step towards highly reconfigurable assistive behaviors with true plug-and-play HMIs.

Appendix A

Appendix

A.1 Dynamics and Stability of eTaSL

In this appendix we explain how to analyze the dynamics of the resulting behaviors that are specified using the building blocks described in chapter 3. Few simplified examples of assistive strategies proposed in this thesis are given.

Solving the eTaSL optimization problem when there are only equality constraints (i.e. no inequality constraints), is equivalent of solving the following unconstrained problem (assume unitary weights for the joints and feature variables):

$$\min_{\dot{\boldsymbol{Q}}} \; O(\dot{\boldsymbol{Q}}) = \min_{\dot{\boldsymbol{Q}}} \; \dot{\boldsymbol{Q}}^{\mathsf{T}} \mu \boldsymbol{I} \dot{\boldsymbol{Q}} + \sum_{i=1}^{n} \boldsymbol{\varepsilon}_i^{\mathsf{T}} w_i \boldsymbol{I} \boldsymbol{\varepsilon}_i, \tag{A.1}$$

where $\dot{\boldsymbol{Q}} = [\dot{\boldsymbol{q}}^{\mathsf{T}} \quad \dot{\boldsymbol{\chi}}^{\mathsf{T}}]^{\mathsf{T}}$ and $w_i = \mu + \hat{w}_i$. The slack variables $\boldsymbol{\varepsilon}_i$ are found with eq. (A.2), where $\boldsymbol{J}_i$ are Jacobian matrices resulting from differentiating the expressions $\boldsymbol{e}_i$ with respect to $\boldsymbol{Q}$. Here we consider only for position constraints for compactness, but the same can be done also for velocity constraints (recall eq. (3.3)). For position constraints, the slack variables can be replaced by the following expression:

$$\boldsymbol{\varepsilon}_i = \boldsymbol{J}_i\dot{\boldsymbol{Q}} + k_i(\boldsymbol{e}_i - \boldsymbol{e}_{s_i}) + \frac{\partial \boldsymbol{e}_i}{\partial t}. \tag{A.2}$$

In order to obtain a dynamic model of the task (i.e. an ordinary differential equation), we solve the optimization problem in eq. (A.1) analytically by differentiating finding the solution when it is equal to zero:

$$\frac{\partial O(\dot{\boldsymbol{Q}})}{\partial \dot{\boldsymbol{Q}}} = 2\mu\boldsymbol{I}\dot{\boldsymbol{Q}} + \sum_{i=1}^{n} 2w_i \left[\frac{\partial \boldsymbol{\varepsilon}_i}{\partial \dot{\boldsymbol{Q}}}\right]^{\intercal} \boldsymbol{\varepsilon}_i = 0. \tag{A.3}$$

By combining eq. (A.2) with eq. (A.3), we obtain:

$$\mu\boldsymbol{I}\dot{\boldsymbol{Q}} + \sum_{i=1}^{n} w_i \boldsymbol{J}_i^{\intercal}\left(\boldsymbol{J}_i\dot{\boldsymbol{Q}} + k_i(\boldsymbol{e}_i - \boldsymbol{e}_{s_i}) + \frac{\partial \boldsymbol{e}_i}{\partial t}\right) = 0. \tag{A.4}$$

Reorganizing the terms, we obtain:

$$(\mu\boldsymbol{I} + \sum_{i=1}^{n} w_i \boldsymbol{J}_i^{\intercal}\boldsymbol{J}_i)\dot{\boldsymbol{Q}} = -\sum_{i=1}^{n} w_i \boldsymbol{J}_i^{\intercal}\left(k_i(\boldsymbol{e}_i - \boldsymbol{e}_{s_i}) + \frac{\partial \boldsymbol{e}_i}{\partial t}\right) \tag{A.5}$$

By taking the inverse of the non-singular square matrix (notice the regularization in the diagonals) to isolate $\dot{\boldsymbol{Q}}$, we obtain a generic differential equation for the dynamics of the task:

$$\dot{\boldsymbol{Q}} = -\left(\mu\boldsymbol{I}_Q + \sum_{i=1}^{n} w_i \boldsymbol{J}_i^{\intercal}\boldsymbol{J}_i\right)^{-1} \sum_{i=1}^{n} w_i \boldsymbol{J}_i^{\intercal}\left(k_i(\boldsymbol{e}_i - \boldsymbol{e}_{s_i}) + \frac{\partial \boldsymbol{e}_i}{\partial t}\right) \tag{A.6}$$

Alternatively, the summations can be replaced by an equivalent matrix representation for compactness:

$$\begin{aligned} \dot{\boldsymbol{Q}} &= -\boldsymbol{J}_{DL}^{+}\left(\boldsymbol{K}(\boldsymbol{e} - \boldsymbol{e}_s) + \frac{\partial \boldsymbol{e}}{\partial t}\right), \\ \boldsymbol{J}_{DL}^{+} &= \left(\mu\boldsymbol{I} + \boldsymbol{J}^{\intercal}\boldsymbol{W}_{\varepsilon}\boldsymbol{J}\right)^{-1}\boldsymbol{J}^{\intercal}\boldsymbol{W}_{\varepsilon}, \end{aligned} \tag{A.7}$$

where $\boldsymbol{e}$, $\boldsymbol{e}_s$ and $\boldsymbol{J}$ are vertical concatenations of their homologous variables

(with subscript i), $\boldsymbol{K}$ contains the corresponding k_i in its diagonal, and $\boldsymbol{J}_{DL}^{+}$ is a weighted pseudoinverse. The DL subscript is used to indicate its similarity with the pseudoinverse of the Damped Least-squares Inverse Kinematics formulation [25].

As mentioned, this formulation was only done for position constraints and not for velocity constraints in the sake of compactness. However, we can still use this formulation for velocity constraints with a simple trick, without the need of explicitly reformulating. Let us take eq. A.2 and make the following modifications in case of desiring a velocity constraint:

1. make $k_i = 0$, so that the error in position does not matter
2. replace $\boldsymbol{e}_i$ by a new expression $\boldsymbol{e}_i - \boldsymbol{v}_i t$, such that the new expression derivative is $\frac{\partial \boldsymbol{e}_i}{\partial t} = \boldsymbol{v}_i$

Notice that if we do this, we obtain the same expression as in eq. (3.3).

In the remaining part of this section, we will give two simple examples that show the usefulness of analyzing the system dynamics. Explanation on how impedance behaviors result from the combination of force/position constraints (i.e. force as a velocity constraint with admittance control) can be found in [16] for the iTaSC framework. In addition, a more specific analysis for eTaSL of particular combinations of pairs of conflicting constraints (i.e. position/position, velocity/velocity, position/velocity and position/sensor-based) can be found in section 2.4. of [99].

A.1.1 Stability example

In this example we analyze the dynamics and stability of the resulting controller for the robot to track a trajectory. We simplify the problem by assuming that the trajectory is inside the singularity-free workspace of the robot and that the robot is able to move sufficiently fast. In other words, we take out the robot from the problem just to analyze the behavior. Thus:

$$\dot{\boldsymbol{Q}} = \dot{\boldsymbol{P}}(t), \tag{A.8}$$

where $\boldsymbol{P} \in R^3$ is a vector of feature variables representing the position of the end effector. The task specification, for the sake of the example consist on two conflicting constraints. The first constraint, eq. (A.9), is defined only

for explanation purposes and constraints the end effector to follow a zero velocity. The second constraint, eq. (A.10), constraints the position of the end effector to follow a trajectory denoted by $\boldsymbol{F}_t(t)$.

$$\boldsymbol{e_1} = \boldsymbol{P} \quad \text{with} \quad w_1 = \mu + \hat{w}_1 \ \text{ and } \ \boldsymbol{v}_1 = \boldsymbol{0} \tag{A.9}$$

$$\boldsymbol{e_2} = \boldsymbol{P} - \boldsymbol{F}_t(t) \quad \text{with} \quad w_2 = \mu + \hat{w}_2 \ \text{ and } \ k_2 > 0 \tag{A.10}$$

It can be observed that the jacobians of both expressions with respect to $\boldsymbol{Q}$ are 3×3 identity matrices I_3. Thus, the system dynamics reduce to:

$$\dot{\boldsymbol{Q}} = \dot{\boldsymbol{P}} = \frac{w_2}{(\mu + w_1 + w_2)} \left(-k_2 \boldsymbol{P} + k_2 \boldsymbol{F}_t(t) + \frac{\partial \boldsymbol{F}_t(t)}{\partial t} \right) \tag{A.11}$$

Notice that whenever $w_2 = 0$ the $\boldsymbol{P}$ remains still (constraint 2 is completely switched off). In addition, notice that if

$$\lim_{t \to \infty} \frac{\partial \boldsymbol{F}_t(t)}{\partial t} = 0 \tag{A.12}$$

then the system has the following equilibrium point:

$$P = \boldsymbol{F}_t(t)$$

If condition ((A.12)) holds, then:

$$\dot{\boldsymbol{P}} = \frac{w_2}{(\mu + w_1 + w_2)} \left(-k_2 \boldsymbol{P} + k_2 \boldsymbol{F}_t(t) \right) = \boldsymbol{A}\boldsymbol{P} + \boldsymbol{B}\boldsymbol{F}_t(t)$$

where $\boldsymbol{A} = -\frac{w_2 k_2}{(\mu + w_1 + w_2)} \boldsymbol{I}$ and $\boldsymbol{B} = \frac{w_2 k_2}{(\mu + w_1 + w_2)}$. Thus, the system is stable (matrix $\boldsymbol{A}$ is negative definite).

A.1.2 Automatic Reactive Approach Motion Analysis

In this example we analyze the behavior of a RAM discussed in subsection 3.4.6, referred to as Automatic Reactive Approach Motion. In order to analyze its behavior, we simplify the problem by assuming that the trajectory

is assumed inside the singularity-free workspace of the robot and that the robot is able to move sufficiently fast. In other words, we take out the robot from the problem just to analyze the behavior. Here we declare the path progress variable s as a feature variable and, thus, our robot state vector is the following:

$$\boldsymbol{Q} = [\boldsymbol{P}^\intercal \;\; s]^\intercal,$$

where $\boldsymbol{P}$ is again the position of the end effector. In order to analyze this RAM, we will specify four constraints. Constraints (A.14) and (A.16) correspond to the constraints of the aforementioned RAM. In addition, constraints (A.13) and (A.15) make the velocity of the end effector and the path progress variable, respectively, to be zero.

$$\boldsymbol{e_1} = \boldsymbol{P} \quad \text{with} \quad w_1 = \mu + \hat{w}_1 \;\text{ and }\; \boldsymbol{v}_1 = 0 \tag{A.13}$$

$$\boldsymbol{e_2} = \boldsymbol{P} - \boldsymbol{F}_t(s) \quad \text{with} \quad w_2 = \mu + \hat{w}_2 \;\text{ and }\; k_2 > 0 \tag{A.14}$$

$$e_3 = s \quad \text{with} \quad w_3 = \mu + \hat{w}_3 \;\text{ and }\; v_3 = 0 \tag{A.15}$$

$$e_4 = s \quad \text{with} \quad w_4 = \mu + \hat{w}_4 \;\text{ and }\; v_4 = V_p(s) \tag{A.16}$$

The RAM constraints make the end effector move autonomously, while the additional constraints make the end effector to stay still. By modulating the weights of such conflicting constraints, one can modulate the Level of Autonomy.

The Jacobian matrices for each expression are the following:

$$\begin{gathered}\boldsymbol{J}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = [\boldsymbol{I}_3 \;\; \boldsymbol{0}], \quad \boldsymbol{J}_2 = \begin{bmatrix} \boldsymbol{I}_3 & -\dfrac{\partial \boldsymbol{F}_t}{\partial s} \end{bmatrix}, \\ \boldsymbol{J}_3 = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{J}_4 = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}.\end{gathered} \tag{A.17}$$

By applying eq. (A.6):

$$\dot{\boldsymbol{Q}} = -\left(\mu \boldsymbol{I} + \sum_{i=1}^{4} w_i \boldsymbol{J}_i^{\intercal} \boldsymbol{J}_i\right)^{-1} \left(w_2 k_2 \boldsymbol{J}_2^T (\boldsymbol{P} - \boldsymbol{F}_t(s)) - w_4 \boldsymbol{J}_4^T V_p(s)\right) \quad \text{(A.18)}$$

By separating the 4×4 system into two systems (one of 3×4 and the other of 1×4), and reorganizing, we get:

$$\dot{\boldsymbol{P}} = \frac{w_2}{\mu + w_1 + w2} \left(\frac{\partial \boldsymbol{F}_t}{\partial s} \dot{s} - k_2 \left(P - \boldsymbol{F}_t(s)\right)\right)$$

$$(\mu + w_2 \frac{\partial \boldsymbol{F}_t}{\partial s}^{\intercal} \frac{\partial \boldsymbol{F}_t}{\partial s} + w_3 + w_4)\dot{s} = w_2 \left(\frac{\partial \boldsymbol{F}_t}{\partial s}^{\intercal} \dot{\boldsymbol{P}} + k_2 \frac{\partial \boldsymbol{F}_t}{\partial s}^{\intercal} (\boldsymbol{P} - \boldsymbol{F}_t(s))\right) + w_4 V_p(s) \quad \text{(A.19)}$$

A simulation of the system was performed in Matlab by using the ODE45 ordinary differential equation solver in order to generate some offline simulation plots. Fig. A.1 and Fig. A.2 show the solution of the system with and without the feedforward term $\frac{\partial \boldsymbol{F}_t}{\partial s}$, respectively, with the same starting position. It can be observed that the feedforward term contributes to reduce the tracking error of the path following.

On the other hand, Fig. 3.4 shows the path obtained by solving the system multiple times, with different initial positions. The initial value of s, given an initial position $\boldsymbol{P}_0$, was solved by means of the following least square problem:

$$\begin{aligned} s_0 = \min_{s} \quad & (\boldsymbol{P}_0 - \boldsymbol{F}_t(s))^{\intercal} (\boldsymbol{P}_0 - \boldsymbol{F}_t(s)) . \\ \text{s.t.} \quad & 0 \leq s \leq 1 \end{aligned} \quad \text{(A.20)}$$

Nevertheless, we do not do this when executing online due to the inclusion of additional constraints that enables the continuous estimation of s, as it is explained in section 3.4.6.

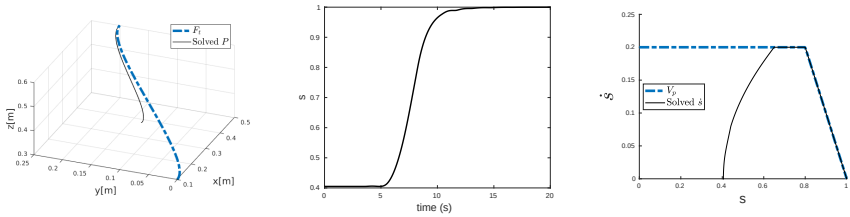


Figure A.1. Simulation solution with feedforward.

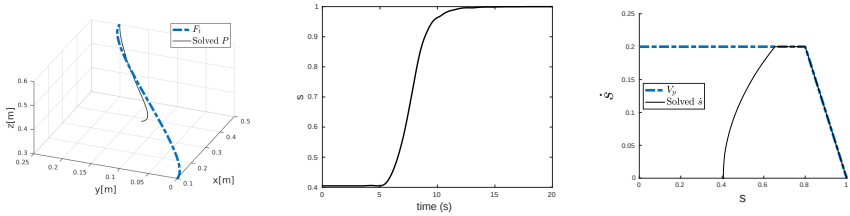


Figure A.2. Simulation solution without feedforward.

For this simulation, a time-policy level of autonomy was defined as shown in Table. A.1. A triggering event was sent at $t = 5s$ and the transition was defined to take $2.5s$. The gain k_2 was set to 0.7 s^{-1}.

Notice from Fig. A.1 and Fig. A.2 that a quasi-trapezoidal motion profile for $\dot{s}$ is achieved. The rising section, which is linear in the time domain, is achieved by defining the type of weight shifting to have a time-policy. On the other hand, the falling section, which is linear in the s domain, is achieved by the definition of $V_p(s)$ (blue curve). The simulation was repeated for multiple starting positions and Fig. 3.4 was generated.

Table A.1. Specification of five levels of autonomy and the involved weight factors. The descending and ascending weight factors $\hat{w}_{i_k}$ are displayed in separate columns. Subscript i indicates the corresponding constraint and subscript k indicates the corresponding arbitration parameter (in this simple case, we only have one).

$\mathbf{G_k}$	**Desc.**	**Asc.**	**Description of the arbitration parameter**
G_1	$\{\hat{w}_{1_1}, \hat{w}_{3_1}\}$	$\{\hat{w}_{2_1}, \hat{w}_{4_1}\}$	Automatic Reac. Appr. motion vs IDLE *Time-policy*.

A.2 Online intent estimation algorithm for assisted reaching tasks

In this section we give more details about the intent estimation algorithm by explaining the influence of the different parameters. Let's recall that the intent estimation algorithm used in section 3.6 is the Bayesian model given by Eq. (3.9):

$$P(O_k|\boldsymbol{x}_{\text{hmi}}) = \frac{P(\boldsymbol{x}_{\text{hmi}}|O_k)P(O_k)}{P_r + \sum_{O_j \in \boldsymbol{O}} P(\boldsymbol{x}_{\text{hmi}}|O_j)P(O_j)}, \quad (3.9)$$

where the joint probability $P_r = P(\boldsymbol{x}_{\text{hmi}}|O_{\text{none}})P(O_{\text{none}})$ estimates when the user does not want to grasp any object (rejection class O_{none}). The term $P(O_{\text{none}})$ is the prior probability of the user not wanting to grasp any object and $P(\boldsymbol{x}_{\text{hmi}}|O_{\text{none}})$ is modeled with eq. (3.10).

$$P(\boldsymbol{x}_{\text{hmi}}|O_{\text{none}}) = \frac{d_{\text{sat}} - d_{\text{min}}}{d_{\text{max}} - d_{\text{min}}}, \quad (3.10)$$

A.2.1 Intent estimation plots

In this section, four examples that vary four parameters of the intent estimation algorithm are compared with a baseline example. We show several 2D and 3D plots of important aspects of the intent estimation to show the influence of such parameters. All of these examples consider **three** objects in the scene, placed in the X-Y plane. We limit our explanation for the 2D intent estimation because in the 3D case we are not able to plot and understand graphically the influence. The values used for the parameters in the baseline example and in the four varying examples are shown in Table. A.2. As mentioned in chapter 3, since the mahalanobis

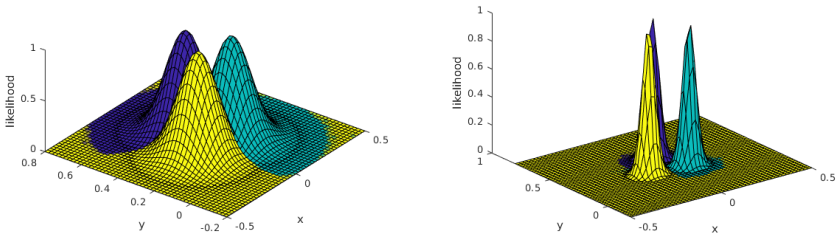
distance follows a χ^2-distribution 2 DOFs (in this 2D case), a confidence interval $[P_{\text{min}}, P_{\text{max}}]$ can be selected instead of d_{min} and d_{max}. Parameters P_{min} and P_{max} are probabilities that are more interpretable and easier to define. We can use the following equation to calculate d_{min} and d_{max} from P_{min} and P_{max}, respectively:

$$d_{\text{min}} = \sqrt{-2\log(1 - P_{\text{min}})}, \quad d_{\text{max}} = \sqrt{-2\log(1 - P_{\text{max}})}. \tag{A.21}$$

Table A.2. Values used for the parameters of the different example (ex.) variations to show the influence their influence in the intent estimation algorithm. Because d_{rn}^2 follows a χ^2-distribution with N_x DOF, a confidence interval $[P_{\text{min}}, P_{\text{max}}]$ that determines d_{min} and d_{max} is selected instead.

Parameter	**Baseline**	**Ex. 1**	**Ex. 2**	**Ex. 3**	**Ex. 4**
variance	0.008	0.008	0.008	0.008	**0.001**
$P(O_{\text{none}})$	0.5	0.5	0.5	**0.9**	0.5
P_{max}	0.75	**0.9**	0.75	0.75	0.75
P_{min}	0.5	0.5	**0.6**	0.5	0.5

Fig. A.3 shows the influence of such parameters in the likelihood $P(\boldsymbol{x}_{\text{hmi}}|O_{\text{k}})$. The mean of each likelihood is located at the position of each object k. It can be observed that the only parameter that influences this likelihood is the variance of the multivariate Gaussian distribution.



(a) (b)

Figure A.3. 3D plots to compare the influence on the intent estimation parameters on the likelihood. (a) Likelihood for baseline, Ex. 1, Ex. 2 and Ex. 3, and (b) Likelihood for Ex. 4.

Fig. A.4 shows the influence of such parameters in the posterior probabilities of all objects calculated with (3.9) by setting $P(O_{\text{none}})$ to zero (i.e. there is

no rejection class considered). It can be observed that the only parameter that influences this likelihood is the variance of the multivariate Gaussian distribution.

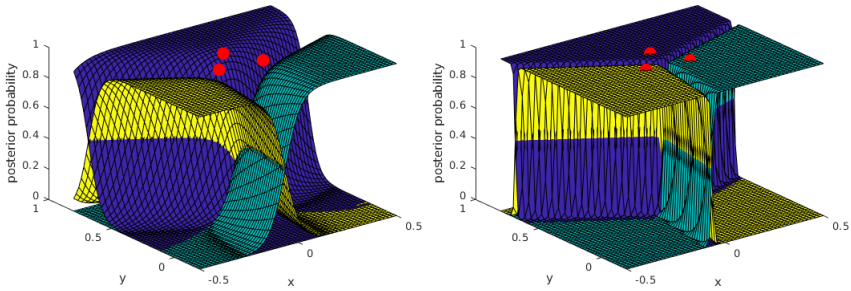


Figure A.4. 3D plots to compare the influence on the intent estimation parameters on the posterior probability obtained without the inclusion of the rejection class O_{none}. (a) baseline, Ex. 1, Ex. 2, and Ex. 3. (b) Ex. 4.

Fig. A.5 shows the influence of such parameters on the probability $P(\boldsymbol{x}_{\text{hmi}}|O_{\text{none}})$ calculated with (3.10).

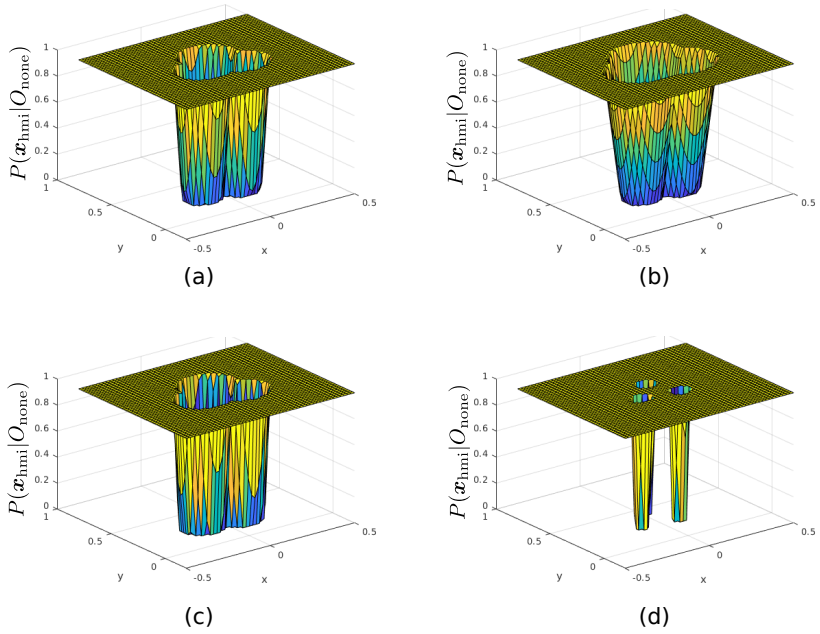


Figure A.5. 3D plots to compare the influence on the intent estimation parameters on the probability $P(\boldsymbol{x}_{\text{hmi}}|O_{\text{none}})$, which is calculated by saturating and normalizing the Mahalanobis distance. (a) baseline and Ex. 3, (b) Ex. 1, (c) Ex. 2, and (d) Ex. 4.

Fig. A.6 shows the influence of such parameters in the posterior probabilities of all objects calculated with (3.9). In contrast to Fig. A.4, here the rejection class is used to calculate the posterior probability, with the values of $P(O_{\text{none}})$ indicated in Table A.2.

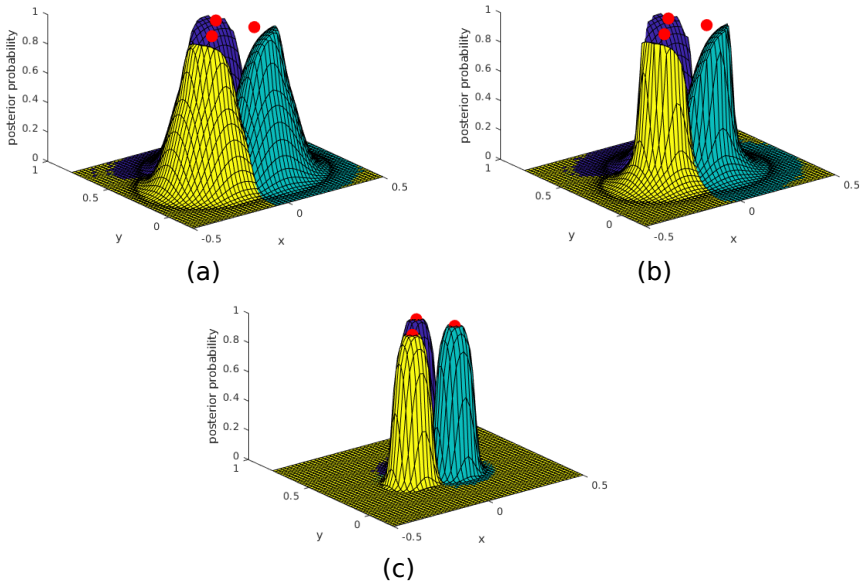


Figure A.6. 3D plots to compare the influence on the intent estimation parameters on the posterior probability. (a) baseline, Ex. 1 and Ex. 2, (b) Ex. 3, and (c) Ex. 4.

A.2.2 Combination of online intent estimation and shared autonomy for reaching tasks

In this section, plots resulting from the combination of the intent estimation and the constraints related to Attraction to intent-inferred targets (RAM), explained in subsection 3.4.3 are presented. The plots are also based on the values of the parameters presented in Table A.2. The relevant BBs are presented again in Table A.3.

Fig. A.7 and Fig. A.8 show the influence of such parameters in the shared autonomy regions obtained through the combination of the intent estimation and the aforementioned shared autonomy, without rejection region and with rejection region, respectively. When both, user and computer, are in control, a weighted average results from the specification of the conflicting constraints, whose weights are modulated according to the posterior probability $P(O_{\text{pred}}|\boldsymbol{x}_{\text{hmi}})$ that is chosen as arbitration parameter (see Fig. 3.5).

Table A.3. Specification of BBs that create the reactive behaviors related to the intent estimation. The BBs presented in this table are a repetition for the convenience of the reader and are originally presented in Table 3.1. The numbers of the BBs are maintained.

#	Expression e_i	Params	Constraints
4)	$\boldsymbol{f}_p\|_{s=1} - \boldsymbol{x}_{\text{hmi}}$	$\hat{w}_4$ $k_4 > 0$	$\boldsymbol{e}_4 \rightarrow \mathbf{0}$: SIM, user controlling $\boldsymbol{f}_p\|_{s=1}$.
5)	$\boldsymbol{f}_p\|_{s=1} - \boldsymbol{x}_{\text{pred}}$	$\hat{w}_9$ $k_9 > 0$	$\boldsymbol{e}_9 \rightarrow \mathbf{0}$: RAM, attraction to intent-inferred targets.

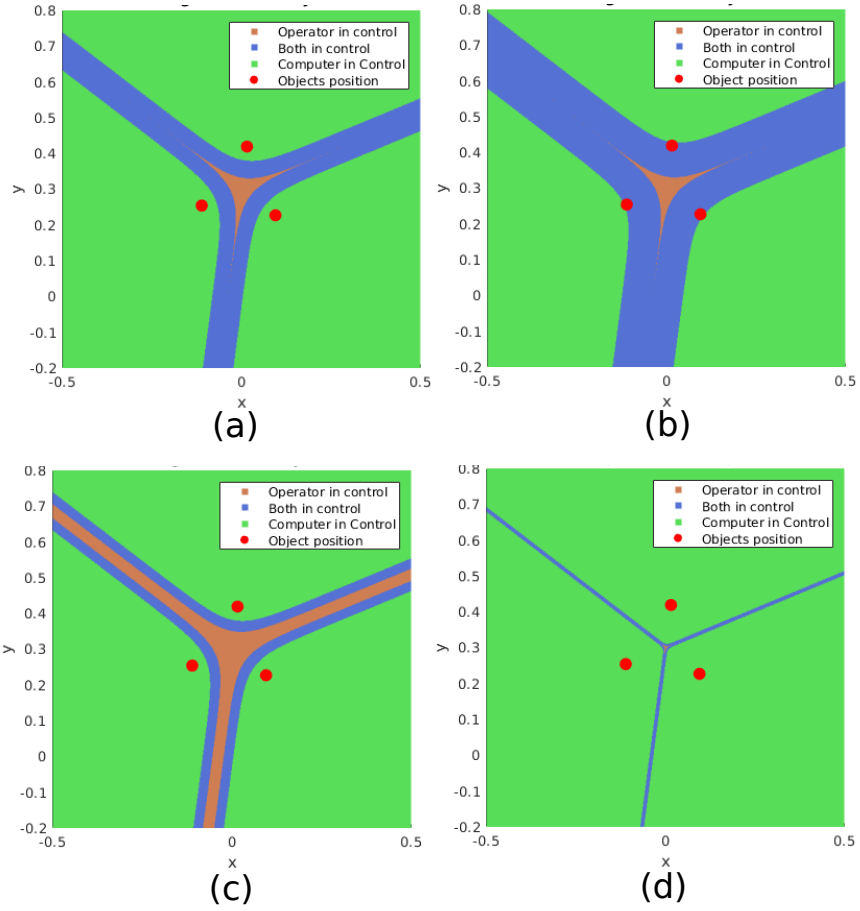


Figure A.7. 2D plots to compare the influence on the intent estimation parameters on the shared autonomy regions obtained without the inclusion the rejection class O_{none}. (a) baseline, (b) Ex. 1, (c) Ex. 2, and (d) Ex. 4. The plot for Ex. 3 is not displayed since $P(O_{\text{none}})$ is not taken into account in these plots and therefore has no influence.

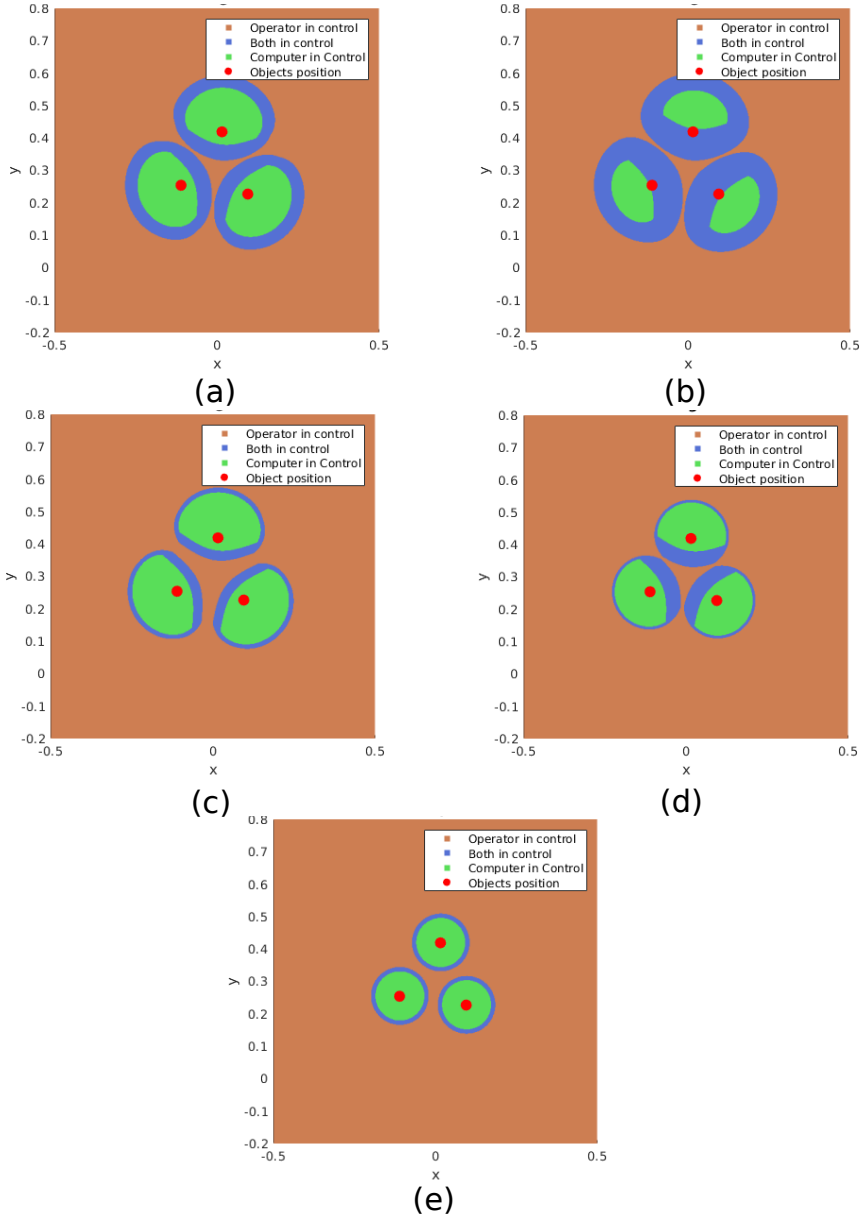


Figure A.8. 2D plots to compare the influence on the intent estimation parameters on the shared autonomy regions. (a) baseline, (b) Ex. 1, (c) Ex. 2, (d) Ex. 3, and (e) Ex. 4.

A.3 HMI configurations

This appendix explains several HMIs used in this thesis. For each HMI we give a general explanation, describe the available transducers, and provide few interaction modalities that can be used with those transducers. The interaction modalities are described using the transducer configuration template (meta-model) proposed in section 1.4.1.

A.3.1 SpaceMouse

This HMI, referred to as a SpaceMouse or as a 6-DOF joystick (see fig. A.9), is composed of the following three sets of transducers:

- *joystick*: a $H \rightarrow M$ transducer with six DOFs that converts displacements performed by the hand of the user to continuous signals proportional to such displacement. This displacement is bounded by the physical limits in which the joystick can move. Three DOFs correspond to linear displacements and three DOFs correspond to angular displacements. In addition, the joystick has a mechanical impedance (i.e. spring) in all six DOFs. When not touched, the spring takes the joystick to an equilibrium point where the measured displacements are zero.
- *buttons*: an $H \rightarrow M$ transducer with two binary DOFs that indicate when or when not the two buttons are pressed by the user.
- *LED light*: an $M \rightarrow H$ transducer that produces a blue light which can be seen by the user.

An example of a possible configuration of the joystick transducer is shown in listing A.1. In this configuration two interaction modalities are used to control, respectively, two aspects of the task: the linear motion and the angular motion of the end effector. This is a typical approach taken in the state-of-the-art, referred to as control mode switching. The human can trigger the transitions between different interaction modalities by, for example, pressing a button of the SpaceMouse or by having an additional HMI (e.g. voice commands or a tablet). Therefore, the configuration or other transducer(s) and a corresponding finite state machine definition are also needed.

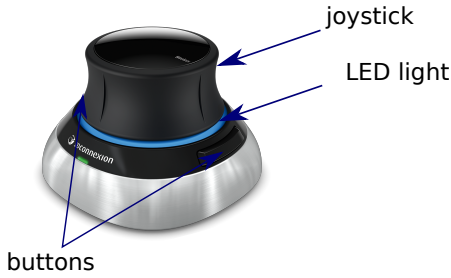


Figure A.9. Picture of the used SpaceMouse with indication where the different transducers are located.

Listing A.1. Example of a transducer configuration of the SpaceMouse joystick for a typical example of mode switching between linear and angular teleoperation of the robot's end effector.

```
{
    "name": "joystick",
    "parent HMI": "SpaceMouse",
    "information flow": H → M,
    "DOFs": 6,
    "interaction modalities":
    {{
        "identifier": "Linear teleoperation",
        "cause": "Linear Interaction with joystick",
        "effect": "Linear motion of end effector",
        "parameters": {...},
        "mapping":
        {
            "Input interpretation": "Desired twist of the
               tcp expressed in a frame attached to the
               joystick",
            "Input units": [m/s,m/s,m/s,rad/s,rad/s,rad/s],
            "Output interpretation": "Desired linear
               velocity of the tcp expressed in the tcp
               frame",
            "Output units": [m/s,m/s,m/s],
            "function": "linear map ℝ^6 → ℝ^3",
            "periodicity": 0.05,
        }
    }
    {
```

```
            "identifier": "Angular teleoperation",
            "cause": "Angular Interaction with joystick",
            "effect": "Angular motion of the end effector",
            "parameters": {...},
            "mapping":
            {
                "Input interpretation": "Desired twist of the
                    tcp expressed in a frame attached to the
                    joystick",
                "Input units": [m/s,m/s,m/s,rad/s,rad/s,rad/s],
                "Output interpretation": "Desired angular
                    velocity of the tcp expressed in the tcp
                    frame",
                "Output units": [rad/s,rad/s,rad/s],
                "function": "linear map ℝ^6 → ℝ^3",
                "periodicity": 0.05,
            }
        }}
    }
```

Alternatively, the SpaceMouse can be used also for controlling the linear and angular velocities of the end effector, simultaneously. In parallel, it can be used to indicate the intended (i.e. desired) target object as shown in listing A.2. The switch between both interaction modalities can be done, for example by pressing the button (other transducer of the SpaceMouse). The transducer configuration for the buttons is omitted from this text. This transducer configuration and the switching interaction modality approach using the button was used in UC4 in chapter 3.

Listing A.2. Transducer configuration of the SpaceMouse joystick for teleoperation and target indication modalities

```
    {
        "name": "joystick",
        "parent HMI": "SpaceMouse",
        "information flow": H → M,
        "DOFs": 6,
        "interaction modalities":
        {{
            "identifier": "Teleoperation",
            "cause": "Linear and angular Interaction with
                joystick",
```

```
            "effect": "Movement of end effector of the robot",
            "mapping":
            {
                "Input interpretation": "Desired twist of the
                    tcp expressed in a frame attached to the
                    joystick",
                "Input units": [m/s,m/s,m/s,rad/s,rad/s,rad/s]
                "Output interpretation": "Desired twist of the
                    tcp expressed in the world frame",
                "Output units": [m/s,m/s,m/s,rad/s,rad/s,rad/s]
                "function": "linear map ℝ^6 → ℝ^6"
            }
        }
        {
            "identifier": "Target Indication",
            "cause": "Linear Interaction with joystick",
            "effect": "Movement of the target end position",
            "mapping":
            {
                "Input interpretation": "Desired linear
                    velocity of the target expressed in the
                    joystick frame",
                "Input units": [m/s,m/s,m/s]
                "Output interpretation": "Desired linear
                    velocity of the target expressed in the
                    world frame",
                "Output units": [m/s,m/s,m/s]
                "function": "linear map ℝ^6 → ℝ^3"
            }
        }}
    }
```

A.3.2 Eye tracker

An eye tracker is a device that takes measurements of one or both eye pupils (e.g. through $H \rightarrow M$ camera transducers) and uses this information to estimate gaze output information such as gazing position, pupil dilatation, fixations, blinks, etc. This HMI can be exploited by spinal cord injured people (e.g. tetraplegic) to indicate intentions, implicitly or explicitly, about

the task. For example, blink gestures could be used to start/stop a robot execution or open/close a gripper. In addition, gaze could be used to indicate objects of interest that the robot should interact with. Furthermore, it could also be used to navigate through a graphical menu in a screen and take decisions about the assistive robotic system. Hence, an eye tracker has many possible modalities, but we only considered use cases with few modalities, as explained further in this section.

In this thesis we use the Pupil Labs Core eye tracker [57], which has built-in open source software that, among others, performs the necessary mappings to extract the aforementioned gaze information. This eye tracker consists of three camera transducers as shown in Fig. A.10. Two of those cameras point each towards one eye, and the remaining points towards the scene from the point of view of the person.

Images from all cameras must be combined for some estimating some of the gaze outputs and thus they will be grouped into the same transducer configuration. The following three modalities are used within this thesis:

1. *blink gestures*: The eye tracker API provides a blink detector that can be used to detect when eye blinks (natural or forced) occur. Natural blinks (blinks that last less than a time threshold) are discarded and only the remaining ones are used to handcraft blink gestures that can be used by the user to indicate intentions about desired tasks to be performed by the robotic system. In this thesis we use single blink, double blinks, and long blinks as blink gestures, which are classified according to selected time thresholds.

2. *fixations*: The eye tracker API provides a fixation detector based on an online dispersion-based algorithm. A fixation occurs when the user's gaze remains in a certain bounded region for a certain duration. We interpret points that are fixated by the user while the task is being executed as potential points of interest. Later on, in chapter 3, these points are associated to objects to be grasped through an intent estimation algorithm.

3. *gaze estimation*: The eye tracker API provides off-the-shelf 2D and 3D estimators for the gaze positions of the user. For this thesis we use a 2D gaze estimator that is always expressed in a local 2D coordinate system attached to a planar object in the scene [57]. This planar object is indicated with QR tags which are detected by the world camera. By an external vision system, which is static with respect to the robot and

calibrated, we determine the position of the planar object and we use it to transform this 2D position in the local frame to a 3D position in the robot's world frame. In this way, the user wearing the eye tracker can still move without the need for re-calibrating. More advanced techniques based on SLAM can also be used, which do not require the use of QR tags.

Listing A.3. Transducer configuration of the eye tracker camera used within this thesis. The input is the pixel image space whose dimensions are not relevant for the discussion.

```
{
    "name": "eye tracker cameras",
    "parent HMI": "eye tracker",
    "information flow": H → M,
    "DOFs": NR,
    "interaction modalities":
    {{
        "identifier": "blink gestures",
        "cause": "The user performs forced blinks",
        "effect": "A discrete set of blink gestures are
            detected",
        "parameters": {...},
        "mapping":
        {
            "Input interpretation": "Camera pixels",
            "Input units": NR,
            "Output interpretation": "Labeled blink
                gestures",
            "Output units": [],
            "function": "machine learning algorithm from
                continuous image space to discrete space",
            "periodicity": 0,
        }
    }
    {
        "identifier": "fixations",
        "cause": "The user fixates at a point of interest",
        "effect": "A fixation event is triggered",
        "parameters": {...},
        "mapping":
        {
            "Input interpretation": "Camera pixels",
```

```
                "Input units": NR,
                "Output interpretation": "Fixation event",
                "Output units": [],
                "function": "dispersion-based detector from
                    continuous image space to a discrete
                    fixation event space",
                "periodicity": 0,
            }
        }
        {
            "identifier": "gaze estimation",
            "cause": "The user observes the scene",
            "effect": "The observed position is estimated and
                interpreted as a task-related point of interest
                ",
            "parameters": {...},
            "mapping":
            {
                "Input interpretation": "Camera pixels",
                "Input units": NR,
                "Output interpretation": "3D position
                    coordinate in the robot's world frame",
                "Output units": [m,m,m],
                "function": "converts from continuous image
                    space to ℝ³",
                "periodicity": 0.05,
            }
        }}
    }
```

A.3.3 3D camera for hand pointing detection

Users that have mobility in their hands and arms can use their hands to indicate objects of interest and/or desired actions. It is natural for humans to point at objects, for example, to indicate other people to bring them to them. In addition, hand gestures can be used to communicate (e.g. thumb up universally indicates that it is OK). These pointing gestures can also be used for communicating with assistive robotic devices.

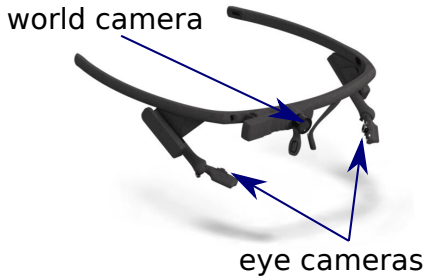


Figure A.10. Picture of the used eye tracker with indication where the different transducers are located.

Several hand tracking devices and methods are available that enable tracking positions of different points of the hand (palm and fingers) and to perform some hand gestures. Nevertheless, for practical reasons, in this work we exploited the capabilities of the vision system pickit 3D (with a stereo camera as $H \rightarrow M$ transducer) to track the hand while performing a single simple hand gesture. We use this off-the-shelf vision system because it was already implemented in our setup as a sensor to detect target objects. However, this system is not designed for that purpose and does not provide robust results.

Pickit 3D allows to teach object point clouds of the object of interest and then their software is able to recognize the object and report, among others, its pose. Therefore, we took five point clouds (from different angles) of the hand of the user while pointing (hand gesture) towards an object of interest, and we selected the desired reported pose to be at the fingertip. In this way we were able to get 3D coordinates of the position of the fingertip only when the user's hand made a pointing hand gesture. The mapping is done aperiodically because pickit takes a snapshot of the scene only when it is triggered (see listing A.4). We performed this snapshot through a voice command with the transducer of another HMI.

Listing A.4. Transducer configuration of the stereo camera transducer used for hand pointing. The input is the pixel image space whose dimensions are not relevant for the discussion.

```
1    {
2        "name": "stereo camera",
3        "parent HMI": "Pickit vision system",
4        "information flow": H → M,
5        "DOFs": NR,
6        "interaction modalities":
```

```
        {{
            "identifier": "hand pointing detection",
            "cause": "The user points at a point of interest",
            "effect": "The position of the fingertip is
                detected and reported as a point of interest",
            "parameters": {...},
            "mapping":
            {
                "Input interpretation": "RGB-D camera pixels",
                "Input units": NR,
                "Output interpretation": "position of the
                    fingertip obtained while pointing",
                "Output units": [m,m,m],
                "function": "proprietary method that maps the
                    image space to ℝ³",
                "periodicity": 0,
            }
        }}
    }
```

A.3.4 Kinesthetic HMI

Users with high mobility capabilities, such as able-bodied users, are able to physically interact with the robot with their hands. We refer to the modality of dragging the robot with the hands as kinesthetic guiding (also known as hand guiding or admittance mode). This can be done either in joint space (with measured joint torques) or in Cartesian space (with end effector measured force/torques). In this thesis we do this kinesthetic guiding in Cartesian space by measuring the forces and moments exerted by the human to the end effector with a Kinesthetic HMI. The Kinesthetic HMI is attached to the end effector of the robot and has a 6-DOF load cell as $H \rightarrow M$ transducer.

Listing A.5. Transducer configuration of the load cell transducer for kinesthetic guidance.

```
    {
        "name": "load cell",
        "parent HMI": "Kinesthetic HMI",
        "information flow": H → M,
        "DOFs": 6,
```

```
        "interaction modalities":
        {{
            "identifier": "kinesthetic guidance",
            "cause": "The user guides the end effector by
                physical interaction",
            "effect": "The end effector moves proportionally in
                the exerted force/torque directions",
            "parameters": {...},
            "mapping":
            {
                "Input interpretation": "Exerted forces and
                    torques at the load cell and expressed in
                    the load cell frame",
                "Input units": [N,N,N,Nm,Nm,Nm],
                "Output interpretation": "Exerted forces and
                    torques at the TCP and expressed in the TCP
                     frame",
                "Output units": [N,N,N,Nm,Nm,Nm],
                "function": "linear map ℝ^6 → ℝ^6",
                "periodicity": 0.005,
            }
        }}
    }
```

A.3.5 BMI or BCI

The explanation on the BMI used in this thesis is given in section 4.2. The transducer configuration of the used BMI, during UC6 and US7 is given in listing A.6, where some the details of the decodings cannot be not disclosed (ND).

Listing A.6. Transducer configuration of the microelectrode array transducers for the BMI use cases.

```
    {
        "name": "microelectrode array",
        "parent HMI": "BMI",
        "information flow": H → M,
        "DOFs": ND,
        "interaction modalities":
```

```
        {{
            "identifier": "teleoperation",
            "cause": "The user thinks about moving the robot
                and generates neural activity",
            "effect": "The end effector of the robot moves",
            "parameters": {...},
            "mapping":
            {
                "Input interpretation": "Electrode spikes",
                "Input units": ND,
                "Output interpretation": "Desired position in a
                    plane (2D) of the end effector, expressed
                    in the world frame",
                "Output units": [m,m],
                "function": "decoding from the microelectrode
                    array space to desired velocities and then
                    integrating to desired positions in ℝ²",
                "periodicity": ND,
            }
        }
        {
            "identifier": "target object",
            "cause": "The user thinks about reaching certain
                object",
            "effect": "The robot automatically grasps that
                object",
            "parameters": {...},
            "mapping":
            {
                "Input interpretation": "Electrode spikes",
                "Input units": ND,
                "Output interpretation": "Index of the desired
                    object Oᵢ",
                "Output units": [],
                "function": "decoding from the microelectrode
                    array space to a discrete set {O_1...O_N}",
                "periodicity": ND,
            }
        }}
    }
```

A.3.6 Sound HMI (Polycom system)

The sound HMI used in this work is composed of the following transducers:

- a microphone ($H \rightarrow M$) used for taking voice commands from the user.
- a speaker ($M \rightarrow H$) used to inform the user through sounds and spoken messages (using text-to-speech software) about the status of the system.

For this work a Polycom telephone was used as sound HMI. For some tasks the user could indicate desired actions (e.g. start motion, close gripper, etc) through voice commands, and in return a spoken message was then played through the speakers for the user to confirm that the command was correctly understood and that the correct action was now being executed. The transducer configuration is omitted.

A.3.7 Screen monitor

The screen monitor of a computer, with a screen as an $M \rightarrow H$ transducer, was used in some tasks to inform the user about certain aspects of the task. This HMI is highly customizable as many options for Graphical User Interface programming exist. In this work we did not exploit the capabilities of this HMI, but we rather used it to provide a limited amount of information when it was required (e.g. to give information about inferred target objects). The transducer configuration is omitted.

Bibliography

[1] Spinal cord injury. *World Health Organization* (2013). https://www.who.int/news-room/fact-sheets/detail/spinal-cord-injury (visited: 14-02-2023).

[2] Priority assistive products list. *World Health Organization* (2016). https://apps.who.int/iris/bitstream/handle/10665/207694/WHO_EMP_PHI_2016.01_eng.pdf (visited: 14-02-2023).

[3] Ageing and health. *World Health Organization* (Oct 2022). https://www.who.int/news-room/fact-sheets/detail/ageing-and-health (visited: 14-02-2023).

[4] Aarno, D., Ekvall, S., and Kragic, D. Adaptive virtual fixtures for machine-assisted teleoperation tasks. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation* (2005), pp. 1139–1144.

[5] Abbatematteo, B., Tellex, S., and Konidaris, G. D. Learning to generalize kinematic models to novel objects. In *Conference on Robot Learning* (2019).

[6] Abbink, D. A., Carlson, T., Mulder, M., de Winter, J. C. F., Aminravan, F., Gibo, T. L., and Boer, E. R. A topology of shared control systems—finding common ground in diversity. *IEEE Transactions on Human-Machine Systems 48*, 5 (2018), 509–525.

[7] Aertbeliën, E. The etasl task specification language, 2016. Last visited April 2023.

[8] Aertbeliën, E., and De Schutter, J. ETaSL/eTC: A constraint-based task specification language and robot controller using expression

graphs. In *IEEE Int. Conf. on Intel. Robots and Systems* (2014), pp. 1540–1546.

[9] Aertbeliën, E., and De Schutter, J. Learning a predictive model of human gait for the control of a lower-limb exoskeleton. In *Proc. IEEE RAS/EMBS Int. Conf. Biomedical Robotics and Biomechatronics* (2014), pp. 520–525.

[10] Albu-Schäffer, A., Ott, C., and Hirzinger, G. A unified passivity-based control framework for position, torque and impedance control of flexible joint robots. *The International Journal of Robotics Research 26*, 1 (2007), 23–39.

[11] Ambler, A., and Popplestone, R. Inferring the positions of bodies from specified spatial relationships. *Artificial Intelligence 6*, 2 (jun 1975), 157–174.

[12] Ancillao, A., Vochten, M., Verduyn, A., Schutter, J. D., and Aertbeliën, E. An optimal method for calculating an average screw axis for a joint, with improved sensitivity to noise and providing an analysis of the dispersion of the instantaneous axes. *PLOS ONE 17*, 10 (10 2022), 1–22.

[13] Argall, B. D. Autonomy in rehabilitation robotics: An intersection. *Annual Review of Control, Robotics, and Autonomous Systems 1*, 1 (2018), 441–463.

[14] Berenson, D., Srinivasa, S., and Kuffner, J. Task Space Regions: A framework for pose-constrained manipulation planning. *International Journal of Robotics Research* (2011).

[15] Bohg, J., Hausman, K., Sankaran, B., Brock, O., Kragic, D., Schaal, S., and Sukhatme, G. S. Interactive perception: Leveraging action in perception and perception in action. *IEEE Transactions on Robotics 33*, 6 (2017), 1273–1291.

[16] Borghesan, G., and De Schutter, J. Constraint-based specification of hybrid position-impedance-force tasks. In *2014 IEEE International Conference on Robotics and Automation (ICRA)* (2014), pp. 2290–2296.

[17] Borghesan, G., Scioni, E., Kheddar, A., and Bruyninckx, H. Introducing geometric constraint expressions into robot constrained motion specification and control. *IEEE Robot. and Autom. Lett. 1*, 2 (July 2016), 1140–1147.

[18] Brock, O., and Khatib, O. Elastic strips: A framework for motion generation in human environments. *The International Journal of Robotics Research 21*, 12 (2002), 1031–1052.

[19] Bruyninckx, H. Robot kinematics and dynamics. Tech. rep., KU Leuven, Leuven, Belgium, 2010.

[20] Bruyninckx, H. Building blocks for complicated and situational aware robotic and cyber-physical systems. Tech. rep., KU Leuven, Leuven, Belgium, 2023. Manuscript version 892476 updated on February 26th 2023. Available at `https://robmosys.pages.gitlab.kuleuven.be/composable-and-explainable-systems-of-systems.pdf`.

[21] Bruyninckx, H., and De Schutter, J. Specification of force-controlled actions in the "task frame formalism" - a synthesis. *IEEE Transactions on Robotics and Automation 12*, 4 (1996), 581–589.

[22] Bustamante, S., Quere, G., Leidner, D., Vogel, J., and Stulp, F. Cats: Task planning for shared control of assistive robots with variable autonomy. In *2022 International Conference on Robotics and Automation (ICRA)* (2022), pp. 3775–3782.

[23] Canal, G., Torras, C., and Alenyà, G. Are preferences useful for better assistance? a physically assistive robotics user study. *J. Hum.-Robot Interact. 10*, 4 (sep 2021).

[24] Cheng, C.-A., Mukadam, M., Issac, J., Birchfield, S., Fox, D., Boots, B., and Ratliff, N. RMPflow: A Computational Graph for Automatic Motion Policy Generation. 2020.

[25] Chiaverini, S., Siciliano, B., and Egeland, O. Review of the damped least-squares inverse kinematics with experiments on an industrial robot manipulator. *IEEE Transactions on Control Systems Technology 2*, 2 (1994), 123–134.

[26] Collinger, J. L., Wodlinger, B., Downey, J. E., Wang, W., Tyler-Kabara, E. C., Weber, D. J., McMorland, A. J., Velliste, M., Boninger, M. L., and Schwartz, A. B. High-performance neuroprosthetic control by an individual with tetraplegia. *The Lancet 381*, 9866 (2013), 557–564.

[27] De Schutter, J., De Laet, T., Rutgeerts, J., Decré, W., Smits, R., Aertbeliën, E., Claes, K., and Bruyninckx, H. Constraint-based task specification and estimation for sensor-based robot systems in the presence of geometric uncertainty. *The International Journal of Robotics Research 26*, 5 (2007), 433–455.

[28] Decré, W., De Schutter, J., and Bruyninckx, H. Optimization-based robot programming with application to human-robot interaction., 2011.

[29] Demeester, E., Huntemann, A., del R. Millan, J., and Van Brussel, H. Bayesian plan recognition for brain-computer interfaces. In *2009 IEEE International Conference on Robotics and Automation* (2009), pp. 653–658.

[30] Deroo, B., Aertbeliën, E., and De Schutter, J. Predicting the user's intention during shared control teleoperation, 2020.

[31] Dietrich, A., Wimbock, T., Albu-Schäffer, A., and Hirzinger, G. Reactive whole-body control: Dynamic mobile manipulation using a large number of actuated degrees of freedom. *IEEE Robotics and Automation Magazine 19*, 2 (2012), 20–33.

[32] Dimeas, F., Fotiadis, F., Papageorgiou, D., Sidiropoulos, A., and Doulgeri, Z. Towards progressive automation of repetitive tasks through physical human-robot interaction. In *Human Friendly Robotics* (Cham, 2019), F. Ficuciello, F. Ruggiero, and A. Finzi, Eds., Springer International Publishing, pp. 151–163.

[33] Downey, J. E., Weiss, J. M., Muelling, K., Venkatraman, A., Valois, J.-S., Hebert, M., Bagnell, J. A., Schwartz, A. B., and Collinger, J. L. Blending of brain-machine interface and vision-guided autonomous robotics improves neuroprosthetic arm performance during grasping. *Journal of NeuroEngineering and Rehabilitation 13*, 1 (Mar 2016), 28.

[34] Dragan, A., Lee, K., and Srinivasa, S. Teleoperation with intelligent and customizable interfaces. *Journal of Human-Robot Interaction 2*, 2 (June 2013), 33 – 57.

[35] Dragan, A. D., Ratliff, N. D., and Srinivasa, S. S. Manipulation planning with goal sets using constrained trajectory optimization. In *Proc. IEEE Int. Conf. Robotics and Automation* (2011).

[36] Dragan, A. D., and Srinivasa, S. S. A policy-blending formalism for shared control. *The International Journal of Robotics Research 32*, 7 (2013), 790–805.

[37] Eisner, B., Zhang, H., and Held, D. Flowbot3d: Learning 3d articulation flow to manipulate articulated objects. In *Robotics: Science and Systems (RSS)* (2022).

[38] Endsley, M. R. Towards a new paradigm for automation: Designing for situation awareness. *IFAC Proceedings Volumes 28*, 15 (1995), 365–370. 6th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design and Evaluation of Man-Machine Systems 1995, Cambridge, MA, USA, 27-29 June 1995.

[39] Endsley, M. R. From here to autonomy: Lessons learned from human–automation research. *Human Factors 59*, 1 (2017), 5–27. PMID: 28146676.

[40] Ewerton, M., Arenz, O., and Peters, J. Assisted teleoperation in changing environments with a mixture of virtual guides. *Advanced Robotics 34*, 18 (2020), 1157–1170.

[41] Ewerton, M., Maeda, G., Koert, D., Kolev, Z., Takahashi, M., and Peters, J. Reinforcement Learning of Trajectory Distributions: Applications in Assisted Teleoperation and Motion Planning. In *Proc. IEEE/RSJ Int. Conf. Int. Robots and Systems* (Macau, 2019), pp. 4294–4300.

[42] Fontana, G. Here's what the EU population could look like by 2100, Sept. 2022. Available at `https://www.weforum.org/agenda/2022/09/visualizing-the-european-union-s-aging-population-by-2100`.

[43] Franklin, G. F., Powell, D. J., and Emami-Naeini, A. *Feedback Control of Dynamic Systems*, 3th ed. Prentice Hall PTR, USA, 2001.

[44] Georgopoulos, A. P., Kalaska, J. F., Caminiti, R., and Massey, J. T. On the relations between the direction of two-dimensional arm movements and cell discharge in primate motor cortex. *J Neurosci 2*, 11 (Nov. 1982), 1527–1537.

[45] Golub, G. H., and van Loan, C. F. *Matrix Computations*, fourth ed. JHU Press, 2013.

[46] Gopinath, D., Jain, S., and Argall, B. D. Human-in-the-loop optimization of shared autonomy in assistive robotics. *IEEE Robotics and Automation Letters 2*, 1 (2017), 247–254.

[47] Hart, S. G., and Staveland, L. E. Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Human Mental Workload*, P. A. Hancock and N. Meshkati, Eds., vol. 52 of *Advances in Psychology*. North-Holland, 1988, pp. 139–183.

[48] Hochberg, L. R., Bacher, D., Jarosiewicz, B., Masse, N. Y., Simeral, J. D., Vogel, J., Haddadin, S., Liu, J., Cash, S. S., van der Smagt, P., and Donoghue, J. P. Reach and grasp by people with tetraplegia using a neurally controlled robotic arm. *Nature 485*, 7398 (May 2012), 372–375.

[49] Holloway, C., and Dawes, H. Disrupting the world of disability: The next generation of assistive technologies and rehabilitation practices. *Healthcare Technology Letters 3*, 4 (2016), 254–256.

[50] HORIZON 2020 - WORK PROGRAMME 2014–2015. Technology readiness levels (trl), 2014-2015. Extract from Part 19 - Commission Decision C(2014)4995, available at `https://ec.europa.eu/research/participants/data/ref/h2020/wp/2014_2015/annexes/h2020-wp1415-annex-g-trl_en.pdf`.

[51] Ijspeert, A. J., Nakanishi, J., Hoffmann, H., Pastor, P., and Schaal, S. Dynamical movement primitives: Learning attractor models for motor behaviors. *Neural Computation 25*, 2 (2013), 328–373.

[52] Iregui, S., De Schutter, J., and Aertbeliën, E. Reconfigurable constraint-based reactive framework for assistive robotics with adaptable levels of autonomy. *IEEE Robotics and Automation Letters 6*, 4 (2021), 7397–7405.

[53] Jaeger, L., Baptista, R. d. S., Basla, C., Capsi-Morales, P., Kim, Y. K., Nakajima, S., Piazza, C., Sommerhalder, M., Tonin, L., Valle, G., Riener, R., and Sigrist, R. How the cybathlon competition has advanced assistive technologies. *Annual Review of Control, Robotics, and Autonomous Systems 6*, 1 (2023), 447–476.

[54] Jain, A., and Kemp, C. C. Pulling open novel doors and drawers with equilibrium point control. In *2009 9th IEEE-RAS International Conference on Humanoid Robots* (2009), pp. 498–505.

[55] Jain, S., and Argall, B. Probabilistic human intent recognition for shared autonomy in assistive robotics. *J. Hum.-Robot Interact. 9*, 1 (Dec. 2019).

[56] Karayiannidis, Y., Smith, C., Barrientos, F. E. V., Ögren, P., and Kragic, D. An adaptive control approach for opening doors and drawers under uncertainties. *IEEE Transactions on Robotics 32*, 1 (2016), 161–175.

[57] Kassner, M., Patera, W., and Bulling, A. Pupil: An open source platform for pervasive eye tracking and mobile gaze-based interaction. In *Adjunct Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (New York, NY, USA, 2014), UbiComp '14 Adjunct, ACM, pp. 1151–1160.

[58] Khansari-Zadeh, S. M., and Billard, A. Learning stable nonlinear dynamical systems with Gaussian mixture models. *IEEE Trans. Robot.* (2011).

[59] Khoramshahi, M., and Billard, A. A dynamical system approach for detection and reaction to human guidance in physical human–robot interaction. *Autonomous Robots 44*, 8 (Nov 2020), 1411–1429.

[60] Kim, D., Hazlett-Knudsen, R., Culver-Godfrey, H., Rucks, G., Cunningham, T., Portee, D., Bricout, J., Wang, Z., and Behal, A. How autonomy impacts performance and satisfaction: Results from a study with spinal cord injured subjects using an assistive robot. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans 42*, 1 (2012), 2–14.

[61] Kim, Y. J., Nam, H. S., Lee, W. H., Seo, H. G., Leigh, J.-H., Oh, B.-M., Bang, M. S., and Kim, S. Vision-aided brain–machine interface training system for robotic arm control and clinical application on two patients with cervical spinal cord injury. *BioMedical Engineering OnLine 18*, 1 (Feb 2019), 14.

[62] Kirk-Wade, E. Uk disability statistics: Prevalence and life experiences. *UK Parliament - House of Commons Library* (2022).

[63] Krug, R., Stoyanov, T., Tincani, V., Andreasson, H., Mosberger, R., Fantoni, G., and Lilienthal, A. J. The next step in robot commissioning: Autonomous picking and palletizing. *IEEE Robot. and Autom. Lett. 1*, 1 (Jan 2016), 546–553.

[64] Kuhner, D., Fiederer, L., Aldinger, J., Burget, F., Völker, M., Schirrmeister, R., Do, C., Boedecker, J., Nebel, B., Ball, T., and Burgard, W. A service assistant combining autonomous robotics, flexible goal formulation, and deep-learning-based brain–computer interfacing. *Robotics and Autonomous Systems 116* (2019), 98–113.

[65] Liu, Y., Zha, F., Sun, L., Li, J., Li, M., and Wang, X. Learning articulated constraints from a one-shot demonstration for robot manipulation planning. *IEEE Access 7* (2019), 172584–172596.

[66] Losey, D. P., McDonald, C. G., Battaglia, E., and O'Malley, M. K. A Review of Intent Detection, Arbitration, and Communication Aspects of Shared Control for Physical Human–Robot Interaction. *Applied Mechanics Reviews 70*, 1 (02 2018). 010804.

[67] Losey, D. P., Srinivasan, K., Mandlekar, A., Garg, A., and Sadigh, D. Controlling assistive robots with learned latent actions. In *2020 IEEE International Conference on Robotics and Automation (ICRA)* (2020), pp. 378–384.

[68] Maheu, V., Archambault, P. S., Frappier, J., and Routhier, F. Evaluation of the jaco robotic arm: Clinico-economic study for powered wheelchair users with upper-extremity disabilities. In *2011 IEEE International Conference on Rehabilitation Robotics* (2011), pp. 1–5.

[69] Mansard, N., Stasse, O., Evrard, P., and Kheddar, A. A versatile Generalized Inverted Kinematics implementation for collaborative working humanoid robots: The Stack Of Tasks. In *International Conference on Advanced Robotics ICAR* (2009).

[70] Martín-Martín, R., and Brock, O. Coupled recursive estimation for online interactive perception of articulated objects. *The International Journal of Robotics Research 0*, 0 (2019), 1–37.

[71] Mehr, N., Horowitz, R., and Dragan, A. D. Inferring and assisting with constraints in shared autonomy. In *2016 IEEE 55th Conference on Decision and Control (CDC)* (2016), pp. 6689–6696.

[72] Miller, A. T., Knoop, S., Christensen, H. I., and Allen, P. K. Automatic grasp planning using shape primitives. In *Proc. IEEE Int. Conf. Robotics and Automation* (Sep. 2003), vol. 2, pp. 1824–1829.

[73] Mittendorfer, P., Yoshida, E., and Cheng, G. Realizing whole-body tactile interactions with a self-organizing, multi-modal artificial skin on a humanoid robot. *Advanced Robotics 29*, 1 (2015), 51–67.

[74] M.S., B. P., and Ph.D., H. Z. Predictors of assistive technology abandonment. *Assistive Technology 5*, 1 (1993), 36–45.

[75] Muelling, K., Venkatraman, A., Valois, J.-S., Downey, J. E., Weiss, J., Javdani, S., Hebert, M., Schwartz, A. B., Collinger, J. L., and Bagnell, J. A. Autonomy infused teleoperation with application to brain computer interface controlled manipulation. *Autonomous Robots 41*, 6 (Aug 2017), 1401–1422.

[76] Mühlbauer, M., Steinmetz, F., Stulp, F., Hulin, T., and Albu-Schäffer, A. Multi-phase multi-modal haptic teleoperation. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2022), pp. 7734–7741.

[77] Ott, C., Bäuml, B., Borst, C., and Hirzinger, G. Autonomous opening of a door with a mobile manipulator: A case study. *IFAC Proceedings Volumes 40*, 15 (2007), 349–354. 6th IFAC Symposium on Intelligent Autonomous Vehicles.

[78] Pairet, È., Ardón, P., Mistry, M., and Petillot, Y. Learning Generalisable Coupling Terms for Obstacle Avoidance via Low-dimensional Geometric Descriptors. In *Proc. IEEE/RSJ Int. Conf. Int. Robots and Systems* (Macau, 2019).

[79] Pane, Y. *Composing Concurrent Reactive Robot Skills: Integration of Task Specification and Control with Symbolic Planning and CAD Information*. PhD thesis, KU Leuven, Leuven, Belgium, Mar. 2022.

[80] Pastor-Bernier, A., Tremblay, E., and Cisek, P. Dorsal premotor cortex is involved in switching motor plans. *Frontiers in Neuroengineering 5* (Apr. 2012), 5.

[81] Pillai, S., Walter, M., and Teller, S. Learning articulated motions from visual demonstration. In *Robotics: Science and Systems X* (2014).

[82] Quere, G., Bustamante, S., Hagengruber, A., Vogel, J., Steinmetz, F., and Stulp, F. Learning and interactive design of shared control templates. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2021), pp. 1887–1894.

[83] Quere, G., Hagengruber, A., Iskandar, M., Bustamante, S., Leidner, D., Stulp, F., and Vogel, J. Shared control templates for assistive robotics. In *2020 IEEE International Conference on Robotics and Automation (ICRA)* (2020), pp. 1956–1962.

[84] Rodriguez, A., Basaez, L., and Celaya, E. A Relational Positioning Methodology for Robot Task Specification and Execution. *IEEE Trans. Robot. 24*, 3 (jun 2008), 600–611.

[85] Romer, G., Stuyt, H., and Peters, A. Cost-savings and economic benefits due to the assistive robotic manipulator (arm). In *9th International Conference on Rehabilitation Robotics, 2005. ICORR 2005.* (2005), pp. 201–204.

[86] Rushby, J. Using model checking to help discover mode confusions and other automation surprises. *RELIABILITY ENGINEERING & SYSTEM SAFETY 75*, 2 (FEB 2002), 167–177.

[87] Salerno, M., Paik, J., and Mintchev, S. FOLDAWAY DroneSense, a controller for haptic information encoding for drone pilots. In *1st International Workshop on Human-Drone Interaction* (Glasgow, United Kingdom, May 2019), Ecole Nationale de l'Aviation Civile [ENAC].

[88] Sathya, A. S., Astudillo, A., Gillis, J., Decré, W., Pipeleers, G., and Swevers, J. Tasho: A python toolbox for rapid prototyping and deployment of optimal control problem-based complex robot motion skills. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2022), pp. 9700–9707.

[89] Schmidt, F., and Robert, B. How we deal with software complexity in robotics: "links and nodes" and the "robotkernel". Tech. rep., DLR, Oberpfaffenhofen, Germany, 2014. talk abstract.

[90] Schmidt, T., Newcombe, R., and Fox, D. Dart: dense articulated real-time tracking with consumer depth cameras. *Autonomous Robots 39*, 3 (Oct 2015), 239–258.

[91] Selvaggio, M., Cognetti, M., Nikolaidis, S., Ivaldi, S., and Siciliano, B. Autonomy in physical human-robot interaction: A brief survey. *IEEE Robotics and Automation Letters 6*, 4 (2021), 7989–7996.

[92] Selvaggio, M., Fontanelli, G. A., Ficuciello, F., Villani, L., and Siciliano, B. Passive virtual fixtures adaptation in minimally invasive robotic surgery. *IEEE Robotics and Automation Letters 3*, 4 (2018), 3129–3136.

[93] Somani, N., Rickert, M., Gaschler, A., Cai, C., Perzylo, A., and Knoll, A. Task level robot programming using prioritized non-linear inequality constraints. In *Proc. IEEE/RSJ Int. Conf. Int. Robots and Systems* (oct 2016), IEEE, pp. 430–437.

[94] Somani, N., Rickert, M., and Knoll, A. An Exact Solver for Geometric Constraints With Inequalities. *IEEE Robot. and Autom. Lett. 2*, 2 (apr 2017), 1148–1155.

[95] Stoyanov, T., Krug, R., Kiselev, A., Sun, D., and Loutfi, A. Assisted telemanipulation: A stack-of-tasks approach to remote manipulator

control. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2018), pp. 1–9.

[96] Sturm, J., Jain, A., Stachniss, C., Kemp, C. C., and Burgard, W. Operating articulated objects based on experience. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems* (2010), pp. 2739–2744.

[97] Sturm, J., Stachniss, C., and Burgard, W. A probabilistic framework for learning kinematic models of articulated objects. *J. Artif. Intell. Res. (JAIR) 41* (05 2011), 477–526.

[98] Tipping, M. E., and Bishop, C. M. Probabilistic principal component analysis. *J. R. Stat. Soc. Ser. B Stat. Methodol. 61*, 3 (1999), 611–622.

[99] Vergara, C. *Combining modeled and learned information to rapidly deploy human-robot collaborative tasks*. PhD thesis, KU Leuven, Leuven, Belgium, Dec. 2020.

[100] Vergara, C. A., Borghesan, G., Aertbeliën, E., and De Schutter, J. Incorporating artificial skin signals in the constraint–based reactive control of human—robot collaborative manipulation tasks. *Industrial Robot 46*, 3 (2019), 360–368.

[101] Vergara, C. A., De Schutter, J., and Aertbeliën, E. Combining imitation learning with constraint-based task specification and control. *IEEE Robot. and Autom. Lett. 4*, 2 (April 2019), 1892–1899.

[102] Vergara, C. A., Iregui, S., De Schutter, J., and Aertbeliën, E. Generating reactive approach motions towards allowable manifolds using generalized trajectories from demonstrations. In *IEEE Int. Conf. on Intel. Robots and Systems* (2020), pp. 9697–9704.

[103] Vogel, J., Hagengruber, A., Iskandar, M., Quere, G., Leipscher, U., Bustamante, S., Dietrich, A., Höppner, H., Leidner, D., and Albu-Schäffer, A. Edan: An emg-controlled daily assistant to help people with physical disabilities. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2020), pp. 4183–4190.

[104] Waldert, S. Invasive vs. Non-Invasive neuronal signals for Brain-Machine interfaces: Will one prevail? *Frontiers in Neuroscience 10* (June 2016), 295.

[105] Wang, M., Kogkas, A. A., Darzi, A., and Mylonas, G. P. Free-view, 3d gaze-guided, assistive robotic system for activities of daily living. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2018), pp. 2355–2361.

[106] Wang, Q., and De Schutter, J. Towards real-time robot programming by human demonstration for 6d force controlled actions. In *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)* (1998), vol. 3, pp. 2256–2261 vol.3.

[107] Weitschat, R., Dietrich, A., and Vogel, J. Online motion generation for mirroring human arm motion. In *2016 IEEE International Conference on Robotics and Automation (ICRA)* (2016), pp. 4245–4250.

[108] Yang, G.-Z., Cambias, J., Cleary, K., Daimler, E., Drake, J., Dupont, P. E., Hata, N., Kazanzides, P., Martel, S., Patel, R. V., Santos, V. J., and Taylor, R. H. Medical robotics—regulatory, ethical, and legal considerations for increasing levels of autonomy. *Science Robotics 2*, 4 (2017), eaam8638.

[109] Zolotas, M., and Demiris, Y. Towards explainable shared control using augmented reality. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2019), pp. 3020–3026.

Curriculum Vitae

Santiago Iregui Rincón
E-mail: *santiago.iregui@kuleuven.be*
Place and date of birth: Bogotá, Colombia, 18 November 1993
Nationality: Colombian

Santiago received his mechanical engineering degree from Universidad de los Andes, Bogota, Colombia, in 2016. In 2018, he received his master's degree in mechanical engineering from the same university, where he worked half-time as a teaching assistant. Between 2018-2023, he worked as a PhD researcher in the Robotics Research Team at KU Leuven, under the supervision of prof. Joris De Schutter and Dr. Erwin Aertbeliën. His research focused on developing personalizable robotic assistive strategies to assist people with motor impairments during activities of daily living. During his PhD, he also participated in few side projects, including the rapid development of an emergency production line of mouth-masks in collaboration with the University of Antwerp, due to the mouth-mask deficiency in Belgium during covid times. He was also an active member of team Chorrobot, who won the first place of the KUKA Innovation award in 2021. In 2022, he did a one-month research stay at the German Aerospace Center (DLR), where he collaborated with researchers from the Re-enabling Robotics Team to develop assistive strategies that were tested in their assistive robot EDAN. Santiago will now continue as a postdoctoral researcher in the Robotics Research Team at KU Leuven.

List of publications

Journal papers (peer reviewed)

- **Iregui, S.**, De Schutter, J., and Aertbeliën, E. Reconfigurable constraint-based reactive framework for assistive robotics with adaptable levels of autonomy. IEEE Robotics and Automation Letters 6, 4 (2021), 7397– 7405.

Conference papers (peer reviewed)

- **Iregui, S.**, Rodriguez, C.F. (2020). Study of a Class of Variable Geometry Truss Manipulators. In: Multibody Mechatronic Systems: vol. 94, (85-93). Presented at the International Symposium on Multibody Systems and Mechatronics, Córdoba, Argentina. ISBN: 978-3-030-60372-4. doi: 10.1007/978-3-030-60372-4_10. Obtained the **silver best research paper award**.
- Vergara, C. A., **Iregui, S.**, De Schutter, J., and Aertbeliën, E. Generating reactive approach motions towards allowable manifolds using generalized trajectories from demonstrations. In IEEE Int. Conf. on Intel. Robots and Systems (2020), pp. 9697–9704.

Workshop papers at international conferences

- **Iregui, S.**, De Schutter, J., Aertbeliën, E. (2019). Constraint-Based Shared Control for BMI Applications. Presented at the IROS2019

Workshop - LEGACY DISRUPTORS IN APPLIED TELEROBOTICS: IMPROVING THE MACHINE, THE INTERFACE, AND THE HUMAN, Macau, China, 04 Nov 2019-08 Nov 2019.

- **Iregui, S.**, Vergara, C., De Schutter, J., Aertbeliën, E. (2020). Generating Reactive Virtual Guidance Fixtures for Assisted Telemanipulation Tasks. Presented at the ICRA 2020 Workshop on Shared Autonomy: Learning and Control, Paris, 04 Jun 2020-04 Jun 2020

- Vergara, C., Cuellar, J., Ulloa, F., **Iregui, S.**, Vanroye, L., Viljoen, R., Pane, Y., De Schutter, J., Aertbeliën, E., Decré, W. (2021). Learning Robust Dual-Arm Coordination in the Manipulation of Elastic Elements from Intuitive Human Demonstrations. (1-1). Presented at the ICRA2021 Workshop on Bimanual Manipulation: Learning, Planing, and Control, Xi'an, China, 30 May 2021-05 Jun 2021. Based on the work performed for the KUKA Innovation award 2021 were **our team won the first place**.

- **Iregui, S.**, De Schutter, J., Aertbeliën, E. (2022). Adaptable Assistive Robotic Manipulation of Articulated Objects with Force-Based Shared Autonomy. Presented at the IEEE ICRA 2022 Workshop - Shared Autonomy in Physical Human-Robot Interaction: Adaptability and Trust, Philadelphia, 23 May 2022-27 May 2022.

Other workshop papers

- **Iregui, S.**, De Schutter, J., Aertbeliën, E. (2019). Constraint-Based Robot Control for BMI Applications. (159-159). Presented at the 38th Benelux Meeting on Systems and Control, Lommel, Belgium, 19 Mar 2019–21 Mar 2019.

- **Iregui, S.**, Aertbeliën, E., De Schutter, J. (2023). Assistive shared control for interacting with articulated objects. Presented at the 42th Benelux Meeting on Systems and Control, Elspeet, The Netherlands, 21 Mar 2023–23 Mar 2023.



FACULTY OF ENGINEERING SCIENCE
DEPARTMENT OF MECHANICAL ENGINEERING
ROBOTICS RESEARCH GROUP
Celestijnenlaan 300 box 2420
3001 Leuven, Belgium

Supervisors:
Em. prof. dr. ir. Joris De Schutter
Dr. ir. Erwin Aertbeliën
(Co-Supervisor)
Prof. dr. ir. Herman Bruyninckx
(Co-Supervisor)