Toward Highly Immersive Telepresence: A Novel Very-Large-Scale Kinesthetic Haptic Interface

Zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

von der KIT-Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation

von

Michael Klaus Fennel

Tag der mündlichen Prüfung:	12. Februar 2025
1. Referent:	Prof. Dr.-Ing. Uwe D. Hanebeck
2. Referentin:	Prof. Dr. Katherine J. Kuchenbecker

Acknowledgement

This thesis marks the culmination of almost five years of research and development at the Intelligent Sensor-Actuator-Systems (ISAS) laboratory, Institute for Anthropomatics and Robotics (IAR) at the Karlsruhe Institute of Technology (KIT). First and foremost, I would like to thank Prof. Uwe D. Hanebeck very much for his supervision and the opportunity to work at his lab. His generous financial and scientific support, combined with the freedom he granted, is the basis for the contributions of this thesis and many of the skills I have acquired over the past years. I am also sincerely grateful to Prof. Katherine J. Kuchenbecker for her willingness to take on the co-advisor role. Her thorough yet always positive feedback has been very helpful in refining this thesis to the highest possible quality.

During my time at ISAS, I have had the privilege of working with many great people, including Ajit, Antonio, Benjamin N., Benjamin S., Chris, Daniel, Dominik Pi., Dominik Pr., Eugen, Florian P., Florian R., Jana, Jiachen, Johannes, Kailai, Leon, Lukas D., Lukas M., Marcel, Marko, Markus, Susanne, Tim, and Ziyu. Scientific progress requires more than just scientists. Therefore, I am especially thankful to our technicians Achim, Alex, and Sascha, who were always ready to lend a helping hand, particularly during the intense construction phase of HapticGiant. Additionally, I would like to thank our secretaries Miss Gambichler and Pia for their administrative support. Special thanks go to Marcel, a fantastic office mate and travel buddy, and Markus, who kindly volunteered to proofread this document. I was fortunate to supervise and collaborate with several very talented students during my time at ISAS. In particular, I would like to acknowledge Giulio, Jakob, Lukas, Serge, and Stefan for their rock-solid contributions and dedication.

I would also like to thank the competence center ROBDEKON (robotic systems for decontamination in hazardous environments), which is funded by the German Federal Ministry of Education and Research (BMBF) under grant 13N14678, for giving me the opportunity to work on real-world applications. My sincere thanks go to Alexander, Angelika, Christian, Janko, Johannes, Philipp, Robert, Steffen, and all other project members for their support and collaboration. My gratitude extends to the team at Skydio, especially Greg, Philipp, and Teo, for an incredible experience in Silicon Valley. Moreover, I would like to thank Felix from the Institute of Product Engineering (IPEK) for our fruitful collaboration on the mechanical design of HapticGiant and Sensodrive GmbH for building an impressive piece of hardware.

Beyond academia, I am fortunate to have friends and companions who have supported me in countless ways. Special thanks to Jakob for producing outstanding videos, René for his impressive CAD skills, and Fabian for his unwavering encouragement.

Lastly, and most importantly, I want to express my deepest gratitude to my family and my partner, Miriam, for their invaluable and unconditional support – not only during the writing of this thesis, but also throughout the rest of my life. Words cannot fully express how grateful I am to have you all!

Karlsruhe, March 2025 Michael Fennel

Kurzfassung

An English version is provided on the next double page.

Das Konzept der immersiven Telepräsenz sorgt bereits seit Jahrzehnten für Faszination in verschiedenen Bereichen, darunter Wissenschaft, Ingenieurwesen und Kunst. Dank jüngster Entwicklungen im Bereich der sogenannten Head-Mounted Displays ist es inzwischen möglich, täuschend echt aussehende virtuelle Umgebungen zu schaffen. Allerdings stellt die Erzeugung von haptischem Feedback nach wie vor eine Herausforderung dar. Insbesondere kinästhetisch-haptisches Feedback, welches Kräfte und Drehmomente auf den Körper der benutzenden Person ausübt, ist durch die Fähigkeiten existierender haptischer Schnittstellen im Hinblick auf die Größe des Arbeitsraumes, die Anzahl und Ausnutzung der Freiheitsgrade sowie die Erzeugung von Kräften und Drehmomenten beschränkt.

Eine ideale kinästhetisch-haptische Schnittstelle, die dem Konzept der sogenannten Encountered-Type Haptic Displays folgt, ermöglicht es der nutzenden Person frei zu entscheiden, wann haptisches Feedback durch Berühren des Endeffektors der Schnittstelle erwünscht ist. Dadurch entfällt die Notwendigkeit einer dauerhaften mechanischen Verbindung, was zu einer Steigerung des erreichbaren Immersionsgrades führt. Darüber hinaus sollte die Menge der möglichen Interaktionen im Idealfall ausschließlich durch das dargestellte Objekt und nicht durch die Schnittstelle selbst beeinflusst werden. In Anlehnung an diese Vision präsentiert diese Dissertation HapticGiant als neuartige, bodengebundene kinästhetisch-haptische Schnittstelle, welche die Darstellung von digitalen Zwillingen mit einer sehr hohen haptischen Wirklichkeitstreue in einem sehr großen Arbeitsraum ermöglicht.

Im Hinblick darauf präsentiert der erste Teil der vorliegenden Arbeit ein neues, optimierungsbasiertes Designverfahren, das die kinematischen und dynamischen Eigenschaften des durchschnittlichen menschlichen Arms berücksichtigt, um das erreichbare Niveau der haptischen Transparenz eines Manipulators mit sechs aktuierten Freiheitsgraden und einem griffförmigen Endeffektor zu maximieren. Durch Kombination mit einer zweidimensionalen Vorpositioniereinheit resultiert ein Konzept, welches einen raumgroßen Arbeitsbereich sowie die natürliche Fortbewegung der nutzenden Person ermöglicht. Basierend darauf wird die Konstruktion eines voll funktionsfähigen Prototyps inklusive Sensorik, Aktorik und Einrichtungen der funktionalen Sicherheit beschrieben. Die resultierende Hardware-Plattform wird ergänzt durch das speziell für HapticGiant entwickelte Real-Time Control Framework (RTCF). Darüber hinaus wird eine eigens entwickelte, echtzeitfähige Simulation vorgestellt, die Effekte wie die Coulomb-Reibung und die endliche Steifigkeit der Gelenke inkludiert.

Aufbauend auf dieser Plattform beschäftigt sich der zweite Teil der Arbeit mit der haptischen Darstellung von digitalen Zwillingen. In diesem Zusammenhang wird eine neuartige Kraftregelung basierend auf hierarchischer Optimierung präsentiert. Diese ermöglicht das dynamisch konsistente Rendering von seriellen kinematischen Ketten bei intrischer Berücksichtigung von HapticGiants Hardwaregrenzen und der Gelenkgrenzen des digitalen Zwillings. Die Leistungsfähigkeit der Regelung wird mit einer Reihe von Experimenten belegt, welche die Bestimmung von Metriken des Haptify-Benchmarks umfassen. Die zugrundeliegenden Folgeregler auf Gelenkebene profitieren von möglichst genauer Kenntnis der Gelenkpositionen, -geschwindigkeiten und -beschleunigungen. Daher wird eine neue Methode zur

Fusion der Daten von Gelenkencodern und Inertialsensoren präsentiert, die weder auf Kalibrierung noch auf spezielle Sensorkonfigurationen angewiesen ist. Experimente bestätigen die Wirksamkeit der vorgeschlagenen Methode, solange die Messwerte von Gelenkencodern und Inertialsensoren konsistent sind. Die daraus entstehende Frage nach der optimalen Sensorkonfiguration wird durch eine neuartige, experimentell validierte Methode zur Bestimmung der optimalen Sensorkonfiguration beantwortet, welche auf dem Prinzip der Beobachtbarkeit in Verbindung mit der Berechnung des erwarteten Messbereichs von Inertialsensoren beruht.

Der dritte und letzte Teil der Arbeit erforscht wie eine dynamische Repositionierung des Endeffektors von HapticGiant bewerkstelligt werden kann, um eine unterbrechungsfreie Darstellung von mehreren digitalen Zwillingen zu realisieren. Hierfür wird der durch die nutzende Person anvisierte digitale Zwilling mithilfe einer neuartigen Intentionsschätzung auf Basis rekurrenter neuronaler Netze vorhergesagt. Die resultierende Genauigkeit ist vergleichbar mit dem Stand der Technik, ohne jedoch abhängig von vordefinierten Szenarien oder manuell ausgewählten Merkmalen zu sein. Für die Verarbeitung der geschätzten Intention wird eine neuartige Strategie zur Pfadplanung vorgeschlagen, welche als Erweiterung von roadmap-basierter Pfadplanung betrachtet werden kann und deutlich reduzierte Rechenzeiten bei ähnlicher Pfadqualität aufweist. In Kombination mit einer eigens entwickelten Methode zur Umwandlung von Pfaden in Trajektorien werden gleichmäßige und dennoch reaktive Bewegungen erzielt, die Hindernisse und wechselnden Intentionen mit einbeziehen.

Insgesamt ermöglichen die vorgestellten Technologien und Methoden die Errichtung von vielseitig einsetzbaren kinästhetisch-haptischen Schnittstellen, die Zugang zu einem bisher unerreichten Grad an Immersion in Telepräsenzanwendungen bieten. Darüber hinaus können die in dieser Arbeit gewonnenen Erkenntnisse zur Lösung von verschiedenen Herausforderungen in der Robotik herangezogen werden, darunter Design, Regelung und Zustandsschätzung von Manipulatoren.

Abstract

The concept of immersive telepresence has fascinated researchers, engineers, and artists for decades. With recent developments in head-mounted display technology, it is now possible to create deceptively real-looking virtual environments. However, generating realistic haptic feedback remains a challenge. In particular, kinesthetic haptic feedback, which exerts forces and torques on the user's body, is limited by the capabilities of existing haptic interfaces in terms of workspace size, dexterity, and force-torque output.

Following the concept of encountered-type haptic displays, the ideal kinesthetic haptic interface allows the user to initiate and release contact with the end effector of the interface whenever desired, eliminating the need for a permanent, and thus immersion-limiting, physical connection. In addition, the set of feasible user interactions should, in an ideal case, be constrained solely by the object being rendered and not by the interface itself. Guided by this vision, this thesis presents HapticGiant, a novel very-large-scale grounded kinesthetic haptic interface that enables the creation of digital twins with a very high degree of haptic realism.

The first part of this thesis proposes an optimization-driven design method that considers the kinematic and dynamic properties of the average human arm to maximize the achievable level of haptic transparency for a manipulator with six actuated degrees of freedom and a handle-shaped end effector. In combination with a two-dimensional prepositioning unit, the resulting design enables a room-sized workspace and natural user locomotion. Based on this concept, the construction of a fully functional prototype with details about instrumentation, actuation, and functional safety is presented. The resulting hardware platform is complemented with the introduction of Real-Time Control Framework (RTCF), which was specifically developed for HapticGiant. Moreover, a custom, real-time-capable simulation environment, featuring effects such as Coulomb friction and finite joint stiffness, is introduced.

With this infrastructure in place, the second part of this thesis deals with the haptic representation of digital twins. In this context, a novel closed-loop force controller based on hierarchical quadratic programming is proposed. This controller enables the rendering of serial kinematic chains in a dynamically consistent manner by intrinsically considering HapticGiant's hardware limitations as well as the joint limits of the digital twin. The effectiveness of the proposed control scheme is successfully demonstrated in a series of experiments that include the evaluation of metrics from the Haptify benchmarking method. The underlying low-level joint tracking controllers benefit from accurate joint position, velocity, and acceleration estimates. Therefore, this thesis proposes a novel method for fusing joint encoder measurements with data from inertial sensors without relying on calibration or specific sensor configurations. Experiments confirm that the proposed method is effective, provided that the readings from the encoders and inertial sensors are consistent. The emerging question of the optimal inertial sensor configuration is answered using a novel, experimentally validated sensor placement procedure that is based on the concept of observability and the computation of the expected measurement range of inertial sensors.

The third and final part of this thesis explores how HapticGiant's end effector can be dynamically repositioned to render multiple digital twins without disrupting the user experience. To achieve this goal, the user's targeted digital twin is predicted using a novel intention estimation method based on recurrent neural networks. Results show that this approach achieves prediction accuracies comparable to state-of-the-art methods, without relying on predefined scenarios or hand-crafted features. To process the intention estimation output, a novel path planning method based on an extension to roadmap-based path planning is proposed, facilitating superior planning times without sacrificing path quality. Combined with a custom path-to-trajectory conversion, this results in smooth and reactive motions that incorporate obstacles and changing user intentions.

Overall, the presented technologies and methods enable the creation of highly versatile kinesthetic haptic displays, paving the way for an unprecedented level of immersion in telepresence applications. Beyond the scope of HapticGiant, the insights gained in this thesis also provide valuable solutions to a number of challenges in robotics, including manipulator design, control, and state estimation.

Notation

General Conventions

$[i]$	Regular reference with number i
$[\mathrm{D}i]$	Reference to datasheet with number i
$[\mathrm{O}i]$	Reference to the author's publication with number i
$[\mathrm{S}i]$	Reference to supervised student thesis with number i
$\underline{\square}$	Vector
$\square$	Matrix
$\square^{\mathsf{T}}$	Transpose of matrix or vector
$\square^{a}$	Entity designated with a
$\square^{A}$	Resolving frame A
$\square_{BC}$	Reference frame B and object frame C
$\square_{C}$	Object frame C
$\tilde{\square}$	Measured quantity
$\hat{\square}$	Estimated quantity
$\square^{*}$	Optimal value
$\square_{[i]}$	i-th element of a vector
$\square_{[i,j]}$	Element in row i and column j of a matrix
$\square_{[i:j,:]}$	Rows i to j of a matrix
$\left\|\square\right\|_{p}$	p-norm of a vector
$[\square\times]$	Skew-symmetric matrix of a vector
$\square_{\mathrm{cmd}}$	Commanded setpoint
$\square_{\mathrm{comp}}$	Friction compensated setpoint
$\square_{\mathrm{dis}}$	Dissipative term
$\square_{\mathrm{dri}}$	Driving term
$\square_{\mathrm{lb}}$	Lower bound
$\square_{\mathrm{ref}}$	Reference value
$\square_{\mathrm{ub}}$	Upper bound
$\underline{0}_{n}$	n-dimensional zero vector
$\mathbf{0}_{n\times m}$	n-by-m zero matrix
$\underline{1}_{n}$	n-dimensional one vector
$\mathbf{I}_{n\times n}$	n-by-n identity matrix
$\mathcal{N}(\underline{\mu},\mathbf{\Sigma})$	Normal distribution with mean $\underline{\mu}$ and covariance $\mathbf{\Sigma}$

Variables

$\underline{a}$	Linear acceleration
$\underline{b}$	Sensor bias
$\mathbf{B}$	Diagonal load-side joint rotor inertia matrix
$\underline{c}$	Bias term in the equation of motion for serial kinematic chains
$\mathcal{C}$	Sensor configuration
C_m	Set of feasible sensor configurations
$\mathbf{C}$	Rotation matrix
$\underline{d}$	Gaze direction
$\mathbf{E}$	Matrix whose columns represent the principal axes of an ellipsoid
$\underline{f}$	Force
$\underline{g}$	Gravitational acceleration
$\boldsymbol{\mathcal{H}}$	Kinematic Hessian
J	Objective
$\boldsymbol{\mathcal{J}}_{\underline{o}}$	Kinematic Jacobian of output quantity $\underline{o}$
$\mathbf{K}$	Diagonal joint stiffness matrix
$\underline{l}$	Link lengths or acceleration bounds
$\underline{m}$	Torque
$\mathbf{M}$	Generalized inertia matrix
n	Number of joints or number of targets
n_a	Number of accelerometers
n_g	Number of gyroscopes
p	Probability
$\underline{p}$	Parameter vector or probability vector
$\underline{q}$	Generalized (load-side) joint positions
Q	Set of joint configurations for reaching an end effector pose
$\underline{r}$	Rotation axis
t	Time
$\underline{t}$	Translation axis
T	Set of reference trajectories
$\mathbf{T}$	Homogeneous transformation composed of rotation and translation
$\underline{v}$	Linear velocity or measurement noise
$\underline{w}$	Wrench composed of force and torque or process noise
W	Set of workspace poses
$\underline{x}$	Linear position or system state
$\underline{y}$	Measurement
α	Rotation angle
$\underline{\alpha}$	Angular acceleration
Δt	Sample time
$\underline{\theta}$	Motor-side joint positions
$\underline{\nu}$	Twist composed of linear and angular velocity
$\underline{\tau}$	Generalized joint torques
$\underline{\omega}$	Angular velocity
$\boldsymbol{\Omega}$	Skew-symmetric matrix of an angular velocity

Frames

A_i	Frame of accelerometer i
B	Base frame of haptic manipulator
C	Center of mass frame
E	End effector frame of digital twin
G	Shoulder frame
G_i	Frame of gyroscope i
H	End effector frame of haptic manipulator
L	Elbow frame of haptic manipulator
M	Palm frame
P	Base frame of prepositioning unit
R	Base frame of digital twin
S	Sensor frame
T_i	Frame of target i
U	User frame
W	World frame
β_i	Link start frame of link i
ϵ_i	Link end frame of link i

Other Designators

a	Kinematic chain of the digital twin
h	Human arm
m	Manipulator (revolute joints)
p	Prepositioning unit (prismatic joints)

Abbreviations

ABA	Articulated body algorithm
ADL	Activities of daily living
AR	Augmented reality
CAD	Computer-aided design
CPU	Central processing unit
CRBA	Composite rigid-body algorithm
DH	Denavit-Hartenberg
DOF	Degree of freedom
DT	Digital twin
EKF	Extended Kalman filter
ETHD	Encountered-type haptic display
FCL	Flexible Collision Library
GPU	Graphics processing unit
HAIR	Head-mounted AR intention recognition
HMD	Head-mounted display
HMM	Hidden Markov model
HQP	Hierarchical quadratic programming
IMU	Inertial measurement unit
KF	Kalman filter
LSTM	Long short-term memory
NN	Neural network
OROCOS	Open Robot Control Software
PD	Proportional-derivative
PDO	Process data object
PI	Proportional-integral
PID	Proportional-integral-derivative
PLC	Programmable logic controller
PMSM	Permanent-magnet synchronous motor
PoMD	Principle of maximum dissipation
PPU	Prepositioning unit
QP	Quadratic programming
RMS	Root mean square
RMSE	Root mean square error
RNEA	Recursive Newton-Euler algorithm
ROS	Robot Operating System
RRT	Rapidly-exploring random tree
RTCF	Real-Time Control Framework
S2KF	Smart sampling Kalman filter
SCARA	Selective compliance assembly robot arm
SEA	Series elastic actuator
SGE	System Generation Engineering
SNR	Signal-to-noise ratio
SVD	Singular value decomposition
TCP	Tool center point
TE	Target environment
UE	User environment
URDF	Unified Robot Description Format
VR	Virtual reality

Contents

CHAPTER 1

Introduction

Contents

> Nothing happens unless first we dream.
>
> CARL SANDBURG (1878 – 1967)

Highly immersive telepresence, i.e., the strong feeling "of presence within a physically *remote* or *simulated* site" [1], has been envisioned by humanity for more than four decades. Notable examples in literature include the *Metaverse* in *Snow Crash* (1992), the virtual world *OASIS* in *Ready Player One* (2012), the *Mirrorspace* in *Drone State* (2014), and the *Ludorama* in *Qube* (2020). This interest extends to film and television, including the digital world in *Tron* (1982), the *Holodeck* in the *Star Trek* series (1988), the remote controlled robots in *Surrogates* (2009), and the simulation in the recently released *The Peripheral* series (2022). All of these occurrences have in common that they create a convincing illusion of reality for the protagonists by providing haptic and visual sensations.

In recent years, major breakthroughs in the field of head-mounted displays (HMDs) have led to consumer-grade virtual reality (VR) and augmented reality (AR) hardware, such as *Oculus Rift* (2013), *HTC Vive* (2016), *Microsoft Hololens 2* (2019), *Magic Leap 2* (2022), *Meta Quest 3* (2023) and *Apple Vision Pro* (2024). With these devices, the creation of deceptively real-looking virtual environments is no longer science fiction, but technological reality. As a result, commercial interest in telepresence continues to grow. Notable examples include *Facebook*'s rebranding to *Meta* [2] and *Apple*'s announcement to create a new product category [3], both betting on the rise of VR and AR technology.

While the human eye is the only organ for visual perception, the sense of touch is distributed over the whole body and produces a wide variety of sensations that can be grouped into tactile and kinesthetic

Figure 1.1: HapticGiant with a user wearing an HMD.

sensations [4–6]. Tactile sensations are caused by cutaneous receptors that detect how an object affects our skin. For example, we sense mechanical deformation or temperature changes at our fingertips, from which we can infer the surface properties of the object being touched. In contrast, kinesthetic sensations arise from the perception of positions and forces exerted on muscles, tendons, and joints. This taxonomy is also commonly used to classify haptic devices.

Compared to the visual aspect, the technology for haptic feedback is still in a stage of intensive research, as there is no device yet that can cover all haptic sensations a human might experience. On the tactile side, progress has been made with the widespread adoption of low-dimensional vibrotactile feedback in consumer electronics, such as handhelds and displays [7]. Beyond that, more and more tactile technologies are transitioning from the research stage to the product stage [8, 9]. On the kinesthetic side, commercial devices have been around for several decades. Their applications include precision surgery, rehabilitation, training, computer-aided engineering, and nuclear material handling [6, 10, 11]. With recently developed hardware, kinesthetic feedback in the form of haptic gloves is also becoming affordable for consumers [12].

To achieve a high degree of immersion, the visual and haptic sensations must be coherent. With existing systems, this already works well on small scales [13, 14]. However, current solutions cannot maintain the illusion once the user interacts at room-scale. For example, imagine a user that intends to leave a virtual room by walking through an initially closed door. With the current HMD technology, a real-looking view of the door is rendered for the user. However, the user experience is disrupted as soon as the user tries to open the door if no haptic feedback is available, as this requires reaching for the door handle in the void or pressing a controller button. This situation can be improved by employing a haptic exoskeleton. In this case, a more natural interaction with the door handle is enabled, but the displayed forces are passed onto the user through the carrying system, creating a *self-tapping on shoulder* sensation. Aside from that, exoskeletons must be fastened and adjusted in a time-consuming procedure before use. A better alternative are grounded kinesthetic haptic interfaces, but these are usually designed to be held continuously, also causing an unnatural, non-immersive feeling. Furthermore, their workspace is usually too small to cover the user's full range of motion, let alone scenarios where the user is locomoting and can choose between different doors or other targets.

With this in mind, the overall goal of this thesis is to overcome the described limitations of existing kinesthetic haptic interfaces by presenting *HapticGiant* – a novel, very large-scale kinesthetic haptic

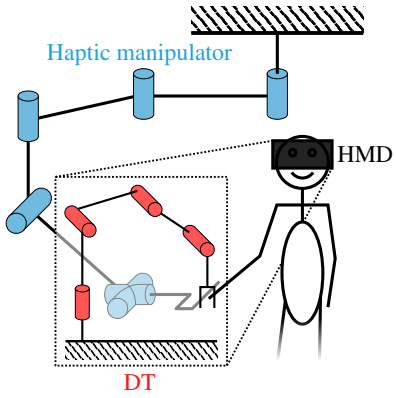


Figure 1.2: Illustration of the DT scheme. The visualization of the DT (red) on the HMD (black) matches the haptic representation provided by the haptic manipulator (blue).

interface. To achieve this goal, the prototype shown in Figure 1.1 is explained, suitable control concepts are proposed, and the application in immersive telepresence scenarios, which require the simultaneous rendering of multiple objects, is explored. With the resulting system, the user in the above example can interact with HapticGiant's end effector as if it was the handle of a physical door that can be *encountered* whenever desired. As a result, an unprecedented level of immersion can be achieved in a broad range of scenarios with one or multiple tangible objects.

In the remainder of this chapter, one of the application scenarios for HapticGiant is elaborated in Section 1.1. Afterwards, related work is presented in Section 1.2, followed by the contributions of this thesis in Section 1.3. Finally, the structure of the remaining document is outlined in Section 1.4.

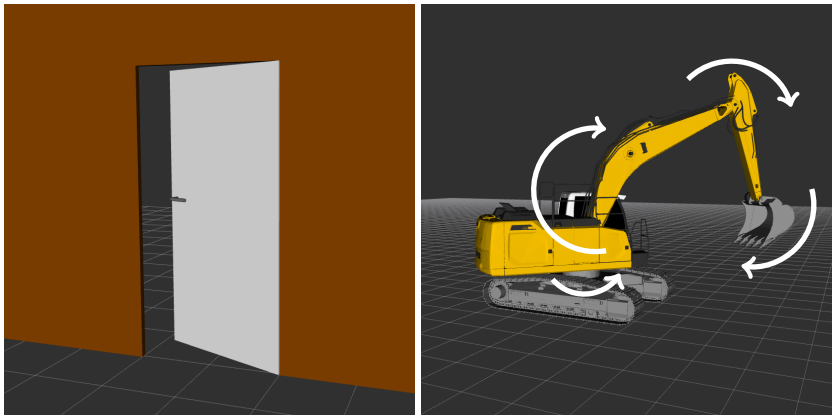
1.1 The Concept of Digital Twins

In this thesis, the feeling of telepresence is achieved using the concept of *digital twins (DTs)*, which is also known as "what you can see is what you can feel" [13]. Formally, this means that the haptic rendering must be aligned with the visual rendering on the HMD to achieve a consistent scene representation for the user. For simplicity, the objects that are rendered with HapticGiant in the context of this thesis are limited to serial kinematic chains. This motivates the illustration in Figure 1.2.

With this setup, the user is able to manipulate or interact with objects in a virtual or remote environment, hereinafter referred to as the target environment (TE), while physically being located in the so-called user environment (UE). For instance, the user can be led to believe that the door in Figure 1.3a is actually real because its DT looks and feels like a real door, including actions such as grasping the handle, opening, and traversing. Likewise, a downscaled model of an excavator, as shown in Figure 1.3b, can be presented to the user for simulation, training, or teleoperation purposes.

1.2 Related Work

In the following, related kinesthetic haptic interfaces are reviewed in Section 1.2.1. In addition, so-called *encountered-type haptic displays* are presented in Section 1.2.2, as HapticGiant also contributes to this category. It is important to note that this thesis deals with issues from different disciplines in order to achieve the intended goal. For this reason, further related work will be discussed in the respective chapters.



(a) Door. **(b)** Excavator.

Figure 1.3: Exemplary DTs. Taken from [O1].

1.2.1 Kinesthetic Haptic Interfaces

Hand-held and wearable kinesthetic haptic interfaces offer superior workspace properties, but they add significant weight that must be carried, and, as mentioned above, generate reaction forces that are felt at the attachment points of the device [15]. For this reason, only grounded kinesthetic haptic interfaces are considered in the following. To further reduce the number of devices, the interactive *Haptipedia* library [16] was used to find existing devices that have a large workspace. From these, all systems that can be considered outdated, i.e., older than 25 years, were filtered out. The remaining devices are shown in Figure 1.4 and supplemented by three additional devices from the literature, including the device in Figure 1.4d, which can be considered as one of the ancestors of HapticGiant.

The corresponding device properties in Table 1.1 reveal that all of the devices have at least one of the following drawbacks:

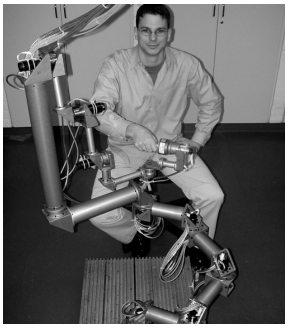
- The end effector has less than six actuated Cartesian degrees of freedom (DOF), which imposes substantial constraints on the choice of objects that can be rendered.
- The device has not been designed with the capabilities of the human arm in mind and, thus, limits the achievable range of motion during the interaction with DTs.
- The workspace is still small and does not support natural locomotion of the user as required for room-scale DTs, such as the door in Figure 1.3a.

On top of that, there is no consistent treatment of joint limits, Cartesian limits, and singularities in the force control scheme of the mentioned devices. As a result, only selected limits are intrinsically respected by the control scheme at best. This means that additional post-processing steps or mechanical stops are required to guarantee basic functional safety. If neither of these measures is implemented, the safe operation depends on the application or, in the worst case, on the user, which is not acceptable for a general-purpose haptic interface.

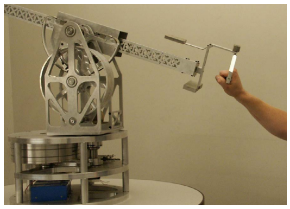
1.2.2 Encountered-Type Haptic Displays

The concept of DTs is closely related to that of *encountered-type haptic displays (ETHDs)*. According to the definition in [14], “an Encountered-Type Haptic Display is a device capable of placing a part of itself or in its entirety in an encountered location that allows the user to have the sensation of voluntarily eliciting haptic feedback”. To achieve this functionality, a so-called surface display [14], which is a rigid object in its simplest form, is dynamically positioned within the user’s workspace

(a) HapticMaster, 2003 [17].

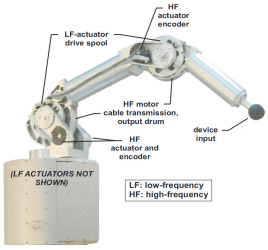


(b) VISHARD10, 2004 [18, 19].



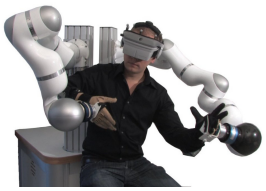
(c) Brake-actuated manipulator, 2006 [20, 21].

(d) HapticGiant's ancestor, 2006 [22, 23].

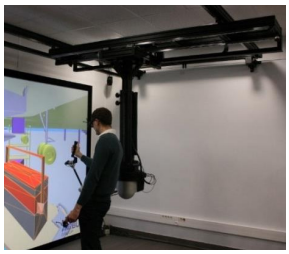


(e) Macro-mini-actuated manipulator, 2008 [24].

(f) Haption INCA 6D, 2009 [25].



(g) DLR bimanual haptic device, 2011 [26].



(h) Haption Scale1™, unknown [D1].

Figure 1.4: A selection of grounded kinesthetic haptic interfaces with a very large workspace.

Figure	Device	# of active DOF		# of passive DOF		Human-centric design	User locomotion possible
		Trans.	Rot.	Trans.	Rot.		
1.4a	HapticMaster [17]	3	0	0	0	✗	✗
1.4b	VISHARD10 [18, 19]	3	3	0	0	✗	✗
1.4c	Brake-actuated manipulator [20, 21]	3	0	0	3	✗	✗
1.4d	HapticGiant's ancestor [22, 23]	3/2	0	0	1	✗	✓
1.4e	Marco-mini-actuated manipulator [24]	3	0	0	0	✗	✗
1.4f	Haption INCA 6D [25]	3	3	0	0	✗	✗
1.4g	DLR bimanual haptic device [26]	3	3	0	0	✓	✗
1.4h	Haption Scale1™ [D1]	3	3	0	0	✗	(✓)
1.1	**HapticGiant**	**3**	**3**	**0**	**0**	✓	✓

Table 1.1: Selected properties of the devices from Figure 1.4. For comparison, the properties of HapticGiant are given in the last row.

to provide haptic feedback that mimics a tangible object, but only when necessary. Given the above definition, DTs can be considered as one of the components for developing ETHDs that are built upon kinesthetic haptic interfaces. In this case, the configuration of a particular DT that the user intends to interact with is reflected by the pose of the end effector of the haptic interface. The end effector itself represents the surface display, e.g., a handle in the previously introduced door example.

In the past, various ETHDs have been presented in the literature [14]. For the sake of brevity, only those covering room-scale workspaces will be reviewed below. Devices that meet this criterion can be divided into ungrounded and grounded devices. In the former category, flying devices, wearables, and mobile platforms can be distinguished.

For example, *HapticDrone* [27] is a system that utilizes an unmanned aerial vehicle to simulate the weight and stiffness of a virtual object in the direction of gravity. In this way, a superior workspace size is achieved, but the ability to display haptic feedback is limited to one dimension. A wearable ETHD is realized in *EncounteredLimbs* [28], where a shoulder-mounted manipulator with a plate at its end effector is used to render flat surfaces, such as walls, in front of the user. Again, this approach has a superior workspace size, but it suffers from the previously mentioned limitations of exoskeletons. A noteworthy example of an ETHD with a mobile platform is the *RoomShift* system [29], which uses a swarm of mobile robots to transport props, such as tables, chairs, and wall pieces. The same concept is adopted by *ZoomWalls* [30] to reposition movable wall elements at room-scale. Both systems share the disadvantages that the movement of the props is limited to the ground plane and that the objects cannot be manipulated by the user. To mitigate the former, a robotic manipulator can be inserted between the prop and the mobile platform. An example of this is the *CoboDeck* system [31], which can place props anywhere in 3D space. However, this system does not implement the ability to manipulate the props either, which is necessary to realize the type of DTs described in Section 1.1.

Unlike ungrounded ETHDs, most of the existing grounded ETHDs [14] that are qualified for rendering DTs have a workspace size that is insufficient for room-scale scenarios. A rare exception to this is *CoVR* [32], which consists of a solid column that is moved across a room by a gantry crane in order to provide haptic sensations such as pushing or leaning. Like Roomshift and ZoomWalls, the column cannot be moved in 3D space or manipulated by the user, making *CoVR* equally unsuitable for rendering DTs. Other noteworthy systems are the predecessors of HapticGiant [22, 33], but these have never been used as ETHDs. In addition, both systems have too few DOF to display arbitrary DTs.

1.3 Challenges and Contributions

This section outlines the main challenges and contributions of this thesis. The field of kinesthetic haptics is quite broad. For this reason, some interesting topics, such as teleoperation and low-level control, are explicitly excluded from the scope of this thesis.

1.3.1 Large-Scale Kinesthetic Haptic Interfaces

Challenge: As seen in Section 1.2.1, there is no kinesthetic haptic interface with a very large workspace and six actuated Cartesian DOF that is sufficient to render DTs in an ETHD at room-scale. On the one hand, this is due to the fact that the design of a particular kinesthetic haptic interface is usually based on the designer's experience rather than on a systematic approach, especially for dimensioning. As a side effect, kinesthetic haptic devices are usually not adapted to the unique properties of the human arm. On the other hand, most of the existing kinesthetic haptic interfaces have not been designed with a locomoting user in mind. Beyond that, existing haptic systems were created using very specialized tools rather than state-of-the-art robotic building blocks. Although this can be explained by the special

requirements, such as high sampling rates, it results in limited applicability of well-established tools for standard problems in robotics, ultimately reducing the accessibility of the technology to researchers and engineers.

Contribution: This thesis contributes a fully functional prototype of HapticGiant, which is a novel kinesthetic haptic interface with a total of eight DOF. In contrast to existing interfaces, HapticGiant has a room-scale workspace and still achieves a very high force-torque output. For the conception, a novel optimization-driven dimensioning method was developed. This method ensures that the resulting design matches the kinematic and dynamic properties of the human arm to achieve good haptic transparency for all feasible user interactions. As a result, HapticGiant enables Cartesian motion with six DOF within a walkable workspace and, thus, paves the way for innovative telepresence applications in engineering, entertainment, safety, and many other fields with an unprecedented level of immersion. The entire platform can be treated like any other robotic system, making it easily accessible to developers and external components. To achieve this, a novel real-time-capable control framework and a realistic, but also real-time capable simulation environment are presented as part of this thesis.

1.3.2 Haptic Rendering of Serial Kinematic Chains

Challenge: HapticGiant needs to generate appropriate forces and torques for the rendering of DTs. A variety of impedance- and admittance-based force control schemes are available for this purpose, but most of them are unable to directly render arbitrary serial kinematic chains, in particular when the kinematic chain to be rendered has joint limits. Furthermore, to the best of the author's knowledge, there is no method for rendering DTs that simultaneously accounts for all mechanical limitations of the haptic manipulator. Most control schemes benefit from a good estimation of the kinematic state, which consists of joint position, velocity, and acceleration. Traditionally, this state is calculated using numerical differentiation and smoothing, but this leads to noise amplification and/or time delays. There are some methods that incorporate data from inertial sensors to improve this situation; however, these rely on specific sensor setups or extensive calibration.

Contribution: This thesis presents a novel force control scheme based on hierarchical quadratic programming (HQP) for the rendering of arbitrary serial kinematic chains in DT scenarios. Compared to existing schemes, the limits of the rendered kinematic chain as well as the joint limits, singularities, and Cartesian limits of the haptic manipulator are jointly included as prioritized constraints in a single optimization problem. This enables HapticGiant to function as a highly immersive ETHD. To improve the kinematic state estimation, a novel sensor fusion method that combines measurements from encoders and inertial sensors is proposed. In contrast to existing approaches, this method does not depend on calibration or specific sensor configurations. Motivated by these simplifications, the state estimation is complemented by a novel optimization-driven method for placing inertial sensors along a kinematic chain, which also takes into account measurement ranges.

1.3.3 Rendering of Multiple Digital Twins

Challenge: When used as an ETHD, HapticGiant's workspace is large enough to accommodate multiple DTs at the same time. For a disruption-free and immersive user experience, the end effector of the interface must be correctly positioned before the user makes contact with any of the DTs. To realize this, the intention of the user must be inferred from the available sensor and scenario data. HapticGiant must then adjust its configuration in a timely but collision-free manner to reach an end effector pose that suits the intention of the user. Existing ETHDs also perform these steps, but very often in a primitive way, e.g., using a nearest neighbor method. Moreover, state-of-the-art approaches

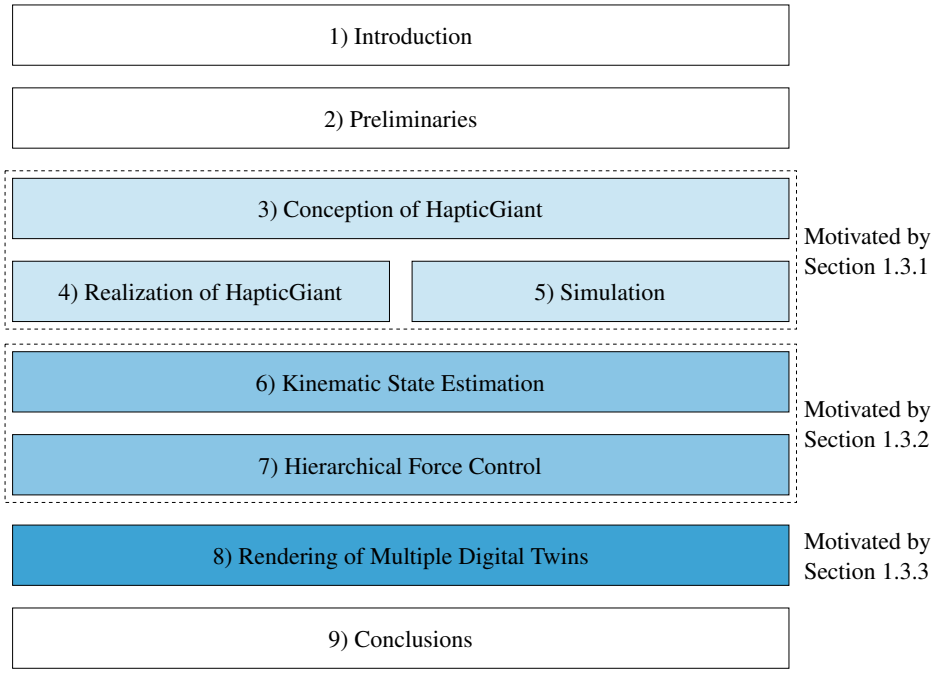


Figure 1.5: Graphical outline. Equally shaded boxes match the subsections of Section 1.3.

cannot be transferred to HapticGiant due to its higher system complexity compared to other ETHD platforms.

Contribution: To address this challenge, this thesis introduces a novel method for intention estimation. Unlike existing solutions, the proposed intention estimation algorithm can handle a variable number of DTs at mutable positions without relying on hand-crafted features. The estimated intentions are then processed in a tailored intention-based motion planning strategy in order to display multiple DTs. The basis for this is a novel path planning algorithm that separates the high-dimensional and slow path planning problem into two simpler problems to achieve responsive and collision-free behavior without significant loss of quality compared to state-of-the-art methods. Furthermore, suitable concepts for the path-to-trajectory conversion and the processing of the estimated intentions are presented to complete the motion planning strategy.

1.4 Outline

The structure of this thesis is illustrated in Figure 1.5. After this introductory chapter, Chapter 2 briefly explains the notation and key theoretical foundations. The remaining chapters are dedicated to the challenges and contributions introduced above, starting with those from Section 1.3.1.

Accordingly, the overall concept of HapticGiant, including a novel optimization-driven dimensioning procedure, is presented in depth in Chapter 3. Starting from there, Chapter 4 describes the realization of a fully functional prototype. For this purpose, important aspects of the hardware setup, the overall software architecture, and selected software modules are presented. Chapter 5 introduces a real-time-capable simulation environment to complete HapticGiant as a platform.

Subsequently, the contributions from Section 1.3.2, that enable and improve the haptic rendering of DTs, are presented in Chapter 6 and Chapter 7. In particular, Chapter 6 explores how inertial sensors can be used without calibration to improve kinematic state estimates. Furthermore, the question of

how to optimally place inertial sensors along a kinematic chain is answered. Chapter 7 presents HapticGiant's control architecture for rendering arbitrary serial kinematic chains, including a novel force controller based on HQP and a performance evaluation of the overall control system.

Chapter 8 explores the setting in which HapticGiant is used as an ETHD that is capable of rendering multiple DTs as discussed in Section 1.3.3. For this purpose, a novel intention estimation algorithm is presented and combined with a suitable motion planning strategy.

Finally, Chapter 9 summarizes the main findings of this thesis, discusses their implications, and suggests ideas for future research, especially in the context of HapticGiant and its applications.

CHAPTER 2

Preliminaries

Contents

> He who loves practice without theory is like the sailor who boards ship without a rudder and compass and never knows where he may cast.
>
> LEONARDO DA VINCI (1452 – 1519)

In this chapter, we first introduce the mathematical notation required for this thesis in Section 2.1. The kinematic foundations are then outlined in Section 2.2. In Section 2.3, the kinematic and dynamic modeling of articulated bodies is presented. Finally, the concept of HQP is introduced in Section 2.4.

This chapter contains results from the author's publications [O1–O3].

2.1 Mathematical Notation

In this thesis, vectors are printed underlined and matrices are printed in bold. Positions, linear velocities, and linear accelerations in Cartesian space are denoted with $\underline{x}$, $\underline{v}$, and $\underline{a}$, respectively. Orientations are either described using rotation matrices $\mathbf{C} \in SO(3)$ or the axis-angle notation, where the axis is $\underline{r}$ and the rotation angle is α. The symbols $\underline{\omega}$ and $\underline{\alpha}$ are used for the corresponding angular velocities and accelerations in Cartesian space. The pose of a frame, which consists of position and orientation, can be described using the homogenous transformation matrix $\mathbf{T}$. Generalized joint positions and efforts are denoted with $\underline{q}$ and $\underline{\tau}$, respectively. The Cartesian force $\underline{f}$ and torque $\underline{m}$ can be stacked as wrench $\underline{w}^\mathsf{T} = \begin{pmatrix}\underline{f}^\mathsf{T} & \underline{m}^\mathsf{T}\end{pmatrix} \in \mathbb{R}^6$. All quantities are given in SI units unless stated differently.

Following the notation from [34], a superscript in Cartesian quantities, such as positions, velocities, and accelerations, indicates the *resolving frame*. The subscripts denote the *reference frame* and the

Symbol	Entity	Chapter
h	Human arm	3
m	Manipulator (revolute joints)	3, 5, 7, 8
p	Prepositioning unit (prismatic joints)	4, 5, 7, 8
a	Kinematic chain of the DT	7

Table 2.1: Designators that are used in superscripts to distinguish non-Cartesian quantities.

object frame in their order of appearance. For example, $\underline{x}_{BC}^{A}$ quantifies the position of the object frame C relative to the reference frame B, given in the coordinates of the resolving frame A. For orientations, the matrix $\mathbf{C}_{B}^{A}$ describes the orientation of frame B with respect to frame A. In axis-angle notation, the rotation axis may be resolved in frame A, yielding the notation $\underline{r}_{AB}^{A} \in \left\{\underline{r} \in \mathbb{R}^3 | \, \|\underline{r}\| = 1\right\}$ and $\alpha_{AB} \in \mathbb{R}$. The subscript in wrenches, forces, and torques is interpreted as the point of attack, i.e., $\underline{w}_{B}^{A}$ is the wrench acting at frame B resolved in frame A. The frames used throughout this thesis are all right-handed and defined in the respective chapters to improve readability. Superscripts in non-Cartesian quantities are used to differentiate between the entities listed in Table 2.1. For example, $\underline{q}^{\mathrm{h}}$ is the configuration of the human arm. In the remainder of this document, functional arguments may be omitted for brevity if the context is clear.

To address individual parts of a matrix or vector, subscripts in square brackets are used with Matlab-style indexing. A colon is used to access ranges or a whole dimension if no limits are included. For example, $[1\!:\!3, :]$ refers to the first three rows and all columns of a matrix. Special matrices, such as the zero matrix $\mathbf{0}$ and the identity matrix $\mathbf{I}$, may come with a subscript containing their dimensions to improve comprehensibility. The same is true for the zero-vector $\underline{0}$ and the one-vector $\underline{1}$.

The cross-product can be written as $\underline{\omega} \times \underline{x} = [\underline{\omega}\times]\, \underline{x}$ using the skew-symmetric matrix

$$[\underline{\omega}\times] = \begin{pmatrix} 0 & -\underline{\omega}_{[3]} & \underline{\omega}_{[2]} \\ \underline{\omega}_{[3]} & 0 & -\underline{\omega}_{[1]} \\ -\underline{\omega}_{[2]} & \underline{\omega}_{[1]} & 0 \end{pmatrix}.$$

2.2 Kinematics

Following [34], several rules are used for kinematic calculations with the notation from the previous section. Rotation matrices can be chained and inverted using

$$\begin{aligned} \mathbf{C}_{C}^{A} &= \mathbf{C}_{B}^{A}\mathbf{C}_{C}^{B} \quad \text{and} \\ \mathbf{C}_{B}^{A} &= \mathbf{C}_{A}^{B\mathsf{T}} = \mathbf{C}_{A}^{B-1}, \end{aligned}$$

respectively. The rotation axis $\underline{r}_{AB}^{A}$ is an eigenvector of the matrix $\mathbf{C}_{B}^{A}$. Hence,

$$\underline{r}_{AB}^{A} = \underline{r}_{AB}^{B} = -\underline{r}_{BA}^{A} \tag{2.1}$$

holds. As explained in [35], the exponential map is used to convert the axis-angle representation into a rotation matrix. The inverse operation is achieved using the logarithmic map. To combine translation and rotation into a single symbol, the homogenous transformation matrix

$$\mathbf{T}_{B}^{A} = \begin{pmatrix} \mathbf{C}_{B}^{A} & \underline{x}_{AB}^{A} \\ \mathbf{0}_{1\times 3} & 1 \end{pmatrix}$$

can be used to transform point C relative to frame B into frame A using

$$\begin{pmatrix} \underline{x}_{AC}^{A} \\ 1 \end{pmatrix} = \mathbf{T}_{B}^{A} \begin{pmatrix} \underline{x}_{BC}^{B} \\ 1 \end{pmatrix} .$$

Similar to rotation matrices, the concatenation and inversion of homogenous transformation matrices is defined as

$$\begin{aligned} \mathbf{T}_{C}^{A} &= \mathbf{T}_{B}^{A}\mathbf{T}_{C}^{B} \quad \text{and} \\ \mathbf{T}_{B}^{A} &= \mathbf{T}_{A}^{B\,-1}. \end{aligned}$$

For translations and positions, the rules

$$\begin{aligned} \underline{x}_{BC}^{A} &= -\underline{x}_{CB}^{A} , \\ \underline{x}_{BD}^{A} &= \underline{x}_{BC}^{A} + \underline{x}_{CD}^{A} , \quad \text{and} \\ \underline{x}_{BC}^{D} &= \mathbf{C}_{A}^{D}\underline{x}_{BC}^{A} \end{aligned}$$

are valid. Analogously, these equations hold for angular velocities. To obtain quantities for the linear velocity, linear acceleration, and angular acceleration, the time derivative of the position or velocity must be calculated with identical resolving and reference frames, before the resolving frame is changed to the sensing frame. For example, the linear velocity of frame C relative to frame B, that is measured in frame A, is

$$\underline{v}_{BC}^{A} = \mathbf{C}_{B}^{A}\dot{\underline{x}}_{BC}^{B} .$$

However, $\dot{\underline{x}}_{BC}^{A} \neq \underline{v}_{BC}^{A}$ in general, because

$$\dot{\underline{x}}_{BC}^{A} = \frac{\partial \underline{x}_{BC}^{A}}{\partial t} = \frac{\partial \left(\mathbf{C}_{B}^{A}\underline{x}_{BC}^{B}\right)}{\partial t} = \dot{\mathbf{C}}_{B}^{A}\underline{x}_{BC}^{B} + \mathbf{C}_{B}^{A}\dot{\underline{x}}_{BC}^{B} = \dot{\mathbf{C}}_{B}^{A}\underline{x}_{BC}^{B} + \underline{v}_{BC}^{A} .$$

The time derivative of the rotation matrix in this expression is

$$\dot{\mathbf{C}}_{B}^{A} = \mathbf{\Omega}_{AB}^{A}\mathbf{C}_{B}^{A} = \mathbf{C}_{B}^{A}\mathbf{\Omega}_{AB}^{B} , \tag{2.2}$$

where $\mathbf{\Omega}_{AB}^{A} = \left[\underline{\omega}_{AB}^{A}\times\right]$. The resolving frame of the skew-symmetric matrix can be changed arbitrarily using

$$\mathbf{\Omega}_{AB}^{D} = \mathbf{C}_{C}^{D}\mathbf{\Omega}_{AB}^{C}\mathbf{C}_{D}^{C} . \tag{2.3}$$

This rule is also applicable to the skew-symmetric matrix of angular accelerations.

2.3 Articulated Bodies

When multiple rigid bodies are connected with joints, an articulated body is formed. In the following, the relevant kinematic and dynamic properties of such systems are described on the basis of [36, 37].

2.3.1 Kinematics

In this thesis, we assume that the articulated body with its root at frame W is composed as depicted in Figure 2.1. For simplicity, we consider only serial kinematic chains with $n \in \mathbb{N}^{+}$ joints of revolute or prismatic type. Each link has a *start frame* β_j coinciding with the output frame of the preceding joint. The *end frame* ϵ_j is located at the input side of the following joint or the end effector. Thus, each link is defined by the fixed rotation $\mathbf{C}_{\epsilon_j}^{\beta_j}$ and translation $\underline{x}_{\beta_j\epsilon_j}^{\beta_j}$. The joint axes are $\underline{r}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \in \left\{\underline{r} \in \mathbb{R}^3 \middle| \|\underline{r}\| = 1\right\}$

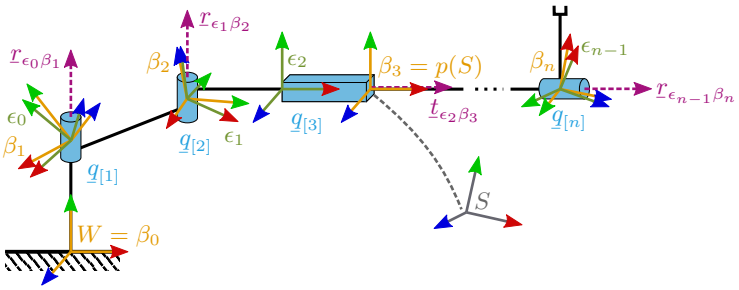


Figure 2.1: Structure of the model used for the kinematic calculations. Rigid links connect the link start frames β_j (orange) to the link end frames ϵ_j (olive). Joints (blue) with the axes $\underline{r}_{\epsilon_j\beta_j+1}$ or $\underline{t}_{\epsilon_j\beta_j+1}$ (purple) connect adjacent links. Adapted from [O2].

and $\underline{t}^{\epsilon_{j-1}}_{\epsilon_{j-1}\beta_j} \in \left\{\underline{t} \in \mathbb{R}^3 \middle| \|\underline{t}\| = 1\right\}$ for revolute and prismatic joints, respectively. Furthermore, the conventions $\underline{t}^{\epsilon_{j-1}}_{\epsilon_{j-1}\beta_j} = \underline{0}_3$ for revolute joints and $\underline{r}^{\epsilon_{j-1}}_{\epsilon_{j-1}\beta_j} = \underline{0}_3$ for prismatic joints are used to establish a unified mathematical representation. As a result, the joint coordinate $\underline{q}_{[j]}$ determines the rotation $\mathbf{C}^{\epsilon_{j-1}}_{\beta_j}$ and the translation $\underline{x}^{\epsilon_{j-1}}_{\epsilon_{j-1}\beta_j}$.

By exploiting the rules from Section 2.2 and $W = \beta_0$, the rotational component of the forward kinematics up to the second derivative can be described recursively using

$$\mathbf{C}^W_{\beta_j} = \mathbf{C}^W_{\beta_{j-1}} \mathbf{C}^{\beta_{j-1}}_{\epsilon_{j-1}} \mathbf{C}^{\epsilon_{j-1}}_{\beta_j}(\underline{q}_{[j]}) \,, \tag{2.4}$$

$$\underline{\omega}^W_{W\beta_j} = \underline{\omega}^W_{W\beta_{j-1}} + \mathbf{C}^W_{\beta_{j-1}} \mathbf{C}^{\beta_{j-1}}_{\epsilon_{j-1}} \underline{r}^{\epsilon_{j-1}}_{\epsilon_{j-1}\beta_j} \dot{\underline{q}}_{[j]} \,, \text{ and} \tag{2.5}$$

$$\underline{\alpha}^W_{W\beta_j} = \underline{\alpha}^W_{W\beta_{j-1}} + \mathbf{\Omega}^W_{W\beta_{j-1}} \mathbf{C}^W_{\beta_{j-1}} \mathbf{C}^{\beta_{j-1}}_{\epsilon_{j-1}} \underline{r}^{\epsilon_{j-1}}_{\epsilon_{j-1}\beta_j} \dot{\underline{q}}_{[j]} + \mathbf{C}^W_{\beta_{j-1}} \mathbf{C}^{\beta_{j-1}}_{\epsilon_{j-1}} \underline{r}^{\epsilon_{j-1}}_{\epsilon_{j-1}\beta_j} \ddot{\underline{q}}_{[j]} \,. \tag{2.6}$$

The translational equivalent is

$$\underline{x}^W_{W\beta_j} = \underline{x}^W_{W\beta_{j-1}} + \mathbf{C}^W_{\beta_{j-1}} \left(\underline{x}^{\beta_{j-1}}_{\beta_{j-1}\epsilon_{j-1}} + \mathbf{C}^{\beta_{j-1}}_{\epsilon_{j-1}} \underline{t}^{\epsilon_{j-1}}_{\epsilon_{j-1}\beta_j} \underline{q}_{[j]} \right) \,, \tag{2.7}$$

$$\begin{aligned} \underline{v}^W_{W\beta_j} &= \underline{v}^W_{W\beta_{j-1}} + \mathbf{\Omega}^W_{W\beta_{j-1}} \mathbf{C}^W_{\beta_{j-1}} \left(\underline{x}^{\beta_{j-1}}_{\beta_{j-1}\epsilon_{j-1}} + \mathbf{C}^{\beta_{j-1}}_{\epsilon_{j-1}} \underline{t}^{\epsilon_{j-1}}_{\epsilon_{j-1}\beta_j} \underline{q}_{[j]} \right) + \mathbf{C}^W_{\beta_{j-1}} \mathbf{C}^{\beta_{j-1}}_{\epsilon_{j-1}} \underline{t}^{\epsilon_{j-1}}_{\epsilon_{j-1}\beta_j} \dot{\underline{q}}_{[j]} \,, \text{ and} \\ \underline{a}^W_{W\beta_j} &= + \underline{a}^W_{W\beta_{j-1}} + \left(\left[\underline{\alpha}^W_{W\beta_{j-1}} \times \right] + \left(\mathbf{\Omega}^W_{W\beta_{j-1}} \right)^2 \right) \mathbf{C}^W_{\beta_{j-1}} \left(\underline{x}^{\beta_{j-1}}_{\beta_{j-1}\epsilon_{j-1}} + \mathbf{C}^{\beta_{j-1}}_{\epsilon_{j-1}} \underline{t}^{\epsilon_{j-1}}_{\epsilon_{j-1}\beta_j} \underline{q}_{[j]} \right) \\ &\quad + 2\, \mathbf{\Omega}^W_{W\beta_{j-1}} \mathbf{C}^W_{\beta_{j-1}} \mathbf{C}^{\beta_{j-1}}_{\epsilon_{j-1}} \underline{t}^{\epsilon_{j-1}}_{\epsilon_{j-1}\beta_j} \dot{\underline{q}}_{[j]} + \mathbf{C}^W_{\beta_{j-1}} \mathbf{C}^{\beta_{j-1}}_{\epsilon_{j-1}} \underline{t}^{\epsilon_{j-1}}_{\epsilon_{j-1}\beta_j} \ddot{\underline{q}}_{[j]} \,. \end{aligned} \tag{2.8}$$

To calculate the kinematic state and its derivatives of an arbitrary frame S with the parent link k, whose link start frame is $\beta_k = p(S)$, the first k iterations are calculated. Then, a last iteration with the link offset that matches the sensor pose, i.e, $\mathbf{C}^{\beta_k}_{\epsilon_k} = \mathbf{C}^{\beta_k}_S$ and $\underline{x}^{\beta_k}_{\beta_k\epsilon_k} = \underline{x}^{\beta_k}_{\beta_k S}$, is performed. In this iteration, the joint is treated as *identity* with $\mathbf{C}^{\epsilon_k}_{\beta_{k+1}} = \mathbf{I}$, $\underline{r}^{\epsilon_k}_{\epsilon_k\beta_{k+1}} = \underline{0}$ and $\underline{t}^{\epsilon_k}_{\epsilon_k\beta_{k+1}} = \underline{0}$, so that $\beta_{k+1} = \epsilon_k = S$.

Due to the recursion, the linear and angular velocities are linearly dependent on the joint velocities. For this reason, the notation

$$\begin{pmatrix} \underline{v}^W_{WS} \\ \underline{\omega}^W_{WS} \end{pmatrix} = \begin{pmatrix} \mathcal{J}_{\underline{v}^W_{WS}} \\ \mathcal{J}_{\underline{\omega}^W_{WS}} \end{pmatrix} \dot{\underline{q}} \tag{2.9}$$

with the Jacobians $\mathcal{J}_{\underline{v}^W_{WS}} \in \mathbb{R}^{3\times n}$ and $\mathcal{J}_{\underline{\omega}^W_{WS}} \in \mathbb{R}^{3\times n}$ can be used. Here, the subscripts indicate for which quantity the Jacobian is valid. For cases where the Jacobian is required for the combination of rotational and translational velocities, the short-hand notation $\underline{\nu}^{W\mathsf{T}}_{WS} = \begin{pmatrix} \underline{v}^{W\mathsf{T}}_{WS} & \underline{\omega}^{W\mathsf{T}}_{WS} \end{pmatrix}$ is available. The corresponding Jacobian is

$$\mathcal{J}_{\underline{\nu}^W_{WS}} = \begin{pmatrix} \mathcal{J}_{\underline{v}^W_{WS}} \\ \mathcal{J}_{\underline{\omega}^W_{WS}} \end{pmatrix} \in \mathbb{R}^{6\times n} \,. \tag{2.10}$$

The derivative of $\underline{\nu}_{WS}^{W}$ with respect to time yields the acceleration

$$\begin{pmatrix} \underline{a}_{WS}^{W} \\ \underline{\alpha}_{WS}^{W} \end{pmatrix} = \boldsymbol{\mathcal{J}}_{\underline{\nu}_{WS}^{W}} \ddot{\underline{q}} + \dot{\boldsymbol{\mathcal{J}}}_{\underline{\nu}_{WS}^{W}} \dot{\underline{q}} \,. \tag{2.11}$$

The second term in this expression is a quadratic function of $\dot{\underline{q}}$ because the Jacobian depends on $\underline{q}$. If only the translational part of the acceleration is considered, the elements of the second term can be rewritten as

$$\left(\dot{\boldsymbol{\mathcal{J}}}_{\underline{v}_{WS}^{W}} \dot{\underline{q}}\right)_{[i]} = \left(\frac{\partial \boldsymbol{\mathcal{J}}_{\underline{v}_{WS}^{W}}}{\partial t} \dot{\underline{q}}\right)_{[i]} = \frac{\partial^2 \underline{x}_{WS[i]}^{W}}{\partial \underline{q} \partial t} \dot{\underline{q}} = \dot{\underline{q}}^{\mathsf{T}} \left(\frac{\partial^2 \underline{x}_{WS[i]}^{W}}{\partial \underline{q} \partial t}\right)^{\mathsf{T}} = \dot{\underline{q}}^{\mathsf{T}} \frac{\partial}{\partial \underline{q}} \left(\frac{\partial \underline{x}_{WS[i]}^{W}}{\partial \underline{q}}\right)^{\mathsf{T}} \dot{\underline{q}} \tag{2.12}$$

by applying *Schwarz's* theorem [38] and the chain rule. The Hessian

$$\boldsymbol{\mathcal{H}}_i = \frac{\partial}{\partial \underline{q}} \left(\frac{\partial \underline{x}_{WS[i]}^{W}}{\partial \underline{q}}\right)^{\mathsf{T}} = \begin{pmatrix} \frac{\partial^2 \underline{x}_{WS[i]}^{W}}{\partial \underline{q}_{[1]} \partial \underline{q}_{[1]}} & \cdots & \frac{\partial^2 \underline{x}_{WS[i]}^{W}}{\partial \underline{q}_{[1]} \partial \underline{q}_{[n]}} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \underline{x}_{WS[i]}^{W}}{\partial \underline{q}_{[n]} \partial \underline{q}_{[1]}} & \cdots & \frac{\partial^2 \underline{x}_{WS[i]}^{W}}{\partial \underline{q}_{[n]} \partial \underline{q}_{[n]}} \end{pmatrix} \tag{2.13}$$

in this expression can be calculated using [39] or, as in this thesis, using Theorem A.7 from Appendix A. Beyond that, the first-order partial derivatives for Cartesian quantities with respect to $\underline{q}$, $\dot{\underline{q}}$, and $\ddot{\underline{q}}$ can be calculated recursively using Theorems A.2 to A.4.

2.3.2 Dynamics

So far, only the geometric properties of the articulated body have been considered. However, the links have individual weight distributions that affect the dynamic behavior of the articulated body, i.e., the response to forces and torques. In general, the equation of motion for a serial kinematic chain with the root frame W and n joints is

$$\mathbf{M}(\underline{q}) \ddot{\underline{q}} + \underline{c}(\underline{q}, \dot{\underline{q}}) = \underline{\tau} - \underline{\tau}_{\text{ext}} \,. \tag{2.14}$$

In this formula, $\mathbf{M}(\underline{q}) \in \mathbb{R}^{n \times n}$ is the positive definite joint-space inertia matrix, which depends on the joint position $\underline{q}$ and is calculated from the weight, the center of mass, and the inertia matrix of each link. The bias term $\underline{c}(\underline{q}, \dot{\underline{q}}) \in \mathbb{R}^n$ contains the generalized torque resulting from Coriolis, centrifugal, and gravitational effects and depends on the joint position $\underline{q}$ and velocity $\dot{\underline{q}}$. The joint actuation is represented by the generalized torque $\underline{\tau} \in \mathbb{R}^n$. An external load wrench $\underline{w}_S^W$ acting at the origin of frame S is transformed to the joint space using

$$\underline{\tau}_{\text{ext}} = \boldsymbol{\mathcal{J}}_{\underline{\nu}_{WS}^{W}}^{\mathsf{T}}(\underline{q}) \, \underline{w}_S^W \tag{2.15}$$

with the Jacobian from (2.10). Additional dissipative torques are subtracted from the right-hand side of the dynamic equation, while driving torques are added. To obtain the torque contribution of gravitational effects

$$\underline{\tau}_{\text{grav}} = \underline{c}(\underline{q}, \underline{0}_n) \tag{2.16}$$

is used. The composite rigid-body algorithm (CRBA) [40] with complexity $\mathcal{O}(n^2)$ can be used to calculate the inertia matrix

$$\mathbf{M}(\underline{q}) = \text{CRBA}(\underline{q}) \,.$$

Two major problems in robot dynamics can be described with (2.14). The first problem is the so-called *inverse dynamics* problem, where the kinematic joint state, including $\underline{q}$, $\dot{\underline{q}}$, and $\ddot{\underline{q}}$, is used to calculate

the driving joint torque $\underline{\tau}$. In practice, the recursive Newton-Euler algorithm (RNEA) [40] is usually utilized to solve this problem, so that

$$\underline{\tau} = \mathrm{RNEA}(\underline{q}, \dot{\underline{q}}, \ddot{\underline{q}}) \,.$$

This algorithm can be used as well to calculate $\underline{c}(\underline{q}, \dot{\underline{q}})$ by setting $\ddot{\underline{q}} = \underline{0}_n$. The second problem in the context of manipulator dynamics is the *forward dynamics* problem. Here, the joint acceleration $\ddot{\underline{q}}$ is calculated from the joint state $\underline{q}$, $\dot{\underline{q}}$, and the driving torque $\underline{\tau}$. In theory, this can be achieved by solving the linear system (2.14) for $\ddot{\underline{q}}$. In practice, the specialized articulated body algorithm (ABA) [40] is favored to efficiently obtain the desired acceleration

$$\ddot{\underline{q}} = \mathrm{ABA}(\underline{q}, \dot{\underline{q}}, \underline{\tau})$$

with complexity $\mathcal{O}(n)$.

2.4 Hierarchical Quadratic Programming

Hierarchical quadratic programming (HQP) is an extension of quadratic programming (QP), in which the problem formulation is split into linear equality and inequality tasks with different priorities, which are solved iteratively. For the explanation in this section, the mathematical formulation from [41] and [42], which also has been presented in the author's publication [O1], is borrowed and adapted to the specific needs of this thesis. In hierarchical quadratic problems, each of the $p_{\max} \in \mathbb{N}^+$ tasks is formulated as

$$T_p : \begin{cases} \mathbf{A}_p \underline{x} + \underline{w}_p = \underline{b}_p \\ \underline{f}_{\mathrm{lb},p} \leq \mathbf{C}_p \underline{x} + \underline{v}_p \leq \underline{f}_{\mathrm{ub},p} \end{cases} ,$$

where $\underline{x} \in \mathbb{R}^n$ is the decision variable and $p \in \mathbb{N}^+$, $p < p_{\max}$ is the task priority. Lower values of p indicate higher priorities. The equality constraint is defined using the matrix $\mathbf{A}_p \in \mathbb{R}^{n_{\mathrm{eq},p} \times n}$ and the offset vector $\underline{b}_p \in \mathbb{R}^{n_{\mathrm{eq},p}}$. Likewise, the inequality constraint is defined using the matrix $\mathbf{C}_p \in \mathbb{R}^{n_{\mathrm{iq},p} \times n}$ and the bounds $\underline{f}_{\mathrm{lb},p}, \underline{f}_{\mathrm{ub},p} \in \mathbb{R}^{n_{\mathrm{iq},p}}$. The slack variables $\underline{w}_p \in \mathbb{R}^{n_{\mathrm{eq},p}}$ and $\underline{v}_p \in \mathbb{R}^{n_{\mathrm{iq},p}}$ are optimized in combination with the decision variable $\underline{x}_p^*$ from lowest to highest priority in the sequence of optimization problems

$$\begin{aligned} (\underline{x}_p^*, \underline{w}_p^*, \underline{v}_p^*) = \underset{\underline{x}_p, \underline{w}_p, \underline{v}_p}{\mathrm{argmin}} & \frac{1}{2} \left\| \underline{w}_p \right\|_2^2 + \frac{1}{2} \left\| \underline{v}_p \right\|_2^2 \\ \text{subject to } & \mathbf{A}_1 \underline{x}_p + \underline{w}_1^* = \underline{b}_1 \,, \\ & \vdots \\ & \mathbf{A}_{p-1} \underline{x}_p + \underline{w}_{p-1}^* = \underline{b}_{p-1} \,, \\ & \mathbf{A}_p \underline{x}_p + \underline{w}_p = \underline{b}_p \,, \\ & \underline{f}_{\mathrm{lb},1} \leq \mathbf{C}_1 \underline{x}_p + \underline{v}_1^* \leq \underline{f}_{\mathrm{ub},1} \,, \\ & \vdots \\ & \underline{f}_{\mathrm{lb},p-1} \leq \mathbf{C}_{p-1} \underline{x}_p + \underline{v}_{p-1}^* \leq \underline{f}_{\mathrm{ub},p-1} \,, \\ & \underline{f}_{\mathrm{lb},p} \leq \mathbf{C}_p \underline{x}_p + \underline{v}_p \leq \underline{f}_{\mathrm{ub},p} \end{aligned} \tag{2.17}$$

with p from 1 to $p_{\max}$. Here, the asterisk marks already optimized variables. This problem states that the solution of the current priority must not cause an increase of the slack variables of all previous

tasks. Mathematically, this requires that the solution of the current priority must be in the null space of all equality constraints from the previous tasks with dimension $n_{\text{ns},p}$, motivating the substitution

$$\underline{x}_p = \underline{x}^*_{p-1} + \mathbf{Z}_p \underline{z}_p \tag{2.18}$$

with $\underline{x}^*_0 = \underline{0}_n$ and $\underline{z}_p \in \mathbb{R}^{n_{\text{ns},p}}$. According to [41], the basis of the null space in matrix notation

$$\mathbf{Z}_p = \operatorname{kern}\Big(\begin{pmatrix} \mathbf{A}_1^\mathsf{T} & \dots & \mathbf{A}_{p-1}^\mathsf{T} \end{pmatrix}^\mathsf{T} \Big) \in \mathbb{R}^{n \times n_{\text{ns},p}}$$

can be calculated computationally efficiently using the recursion

$$\mathbf{Z}_p = \mathbf{Z}_{p-1} \operatorname{kern}(\mathbf{A}_{p-1} \mathbf{Z}_{p-1})$$

with $\mathbf{Z}_1 = \mathbf{I}_{n \times n}$. With this information, (2.17) can be reformulated as

$$\begin{aligned} \left(\underline{z}^*_p, \underline{v}^*_p\right) = \underset{\underline{z}_p, \underline{v}_p}{\operatorname{argmin}} & \frac{1}{2} \left\| \mathbf{A}_p (\underline{x}^*_{p-1} + \mathbf{Z}_p \underline{z}_p) - \underline{b}_p \right\|_2^2 + \frac{1}{2} \left\| \underline{v}_p \right\|_2^2 \\ \text{subject to } & \underline{f}_{\text{lb},1} \leq \mathbf{C}_1 (\underline{x}^*_{p-1} + \mathbf{Z}_p \underline{z}_p) + \underline{v}^*_1 \leq \underline{f}_{\text{ub},1} \,, \\ & \vdots \\ & \underline{f}_{\text{lb},p-1} \leq \mathbf{C}_{p-1} (\underline{x}^*_{p-1} + \mathbf{Z}_p \underline{z}_p) + \underline{v}^*_{p-1} \leq \underline{f}_{\text{ub},p-1} \,, \\ & \underline{f}_{\text{lb},p} \leq \mathbf{C}_p (\underline{x}^*_{p-1} + \mathbf{Z}_p \underline{z}_p) + \underline{v}_p \leq \underline{f}_{\text{ub},p} \end{aligned} \tag{2.19}$$

by substituting $\underline{w}_p$ with the equality constraint of task p and $\underline{x}_p$ with (2.18). The optimal value of the decision variable for the current level is then calculated using

$$\underline{x}^*_p = \underline{x}^*_{p-1} + \mathbf{Z}_p \underline{z}^*_p \,,$$

analogous to (2.18). As a result, the remaining equality constraints can be omitted because they are always fulfilled. With the definitions

$$\begin{aligned} \underline{\zeta}_p &= \begin{pmatrix} \underline{z}_p \\ \underline{v}_p \end{pmatrix} \in \mathbb{R}^{n_{\text{ns},p} + n_{\text{iq},p}} \,, & \mathbf{H}_p &= \begin{pmatrix} \mathbf{Z}_p^\mathsf{T} \mathbf{A}_p^\mathsf{T} \mathbf{A}_p \mathbf{Z}_p & \mathbf{0}_{n_{\text{ns},p} \times n_{\text{iq},p}} \\ \mathbf{0}_{n_{\text{iq},p} \times n_{\text{ns},p}} & \mathbf{I}_{n_{\text{iq},p} \times n_{\text{iq},p}} \end{pmatrix} \,, \\ \underline{g}_p &= \begin{pmatrix} \mathbf{Z}_p^\mathsf{T} \mathbf{A}_p^\mathsf{T} \left(\mathbf{A}_p \underline{x}^*_{p-1} - \underline{b}_p \right) \\ \underline{0}_{n_{\text{iq},p}} \end{pmatrix} \,, & \tilde{\mathbf{C}}_p &= \begin{pmatrix} \mathbf{C}_1 \mathbf{Z}_p & \mathbf{0}_{n_{\text{iq},1} \times n_{\text{iq},p}} \\ \vdots & \vdots \\ \mathbf{C}_{p-1} \mathbf{Z}_p & \mathbf{0}_{n_{\text{iq},p-1} \times n_{\text{iq},p}} \\ \mathbf{C}_p \mathbf{Z}_p & \mathbf{I}_{n_{\text{iq},p} \times n_{\text{iq},p}} \end{pmatrix} \,, \end{aligned} \tag{2.20}$$

$$\tilde{\underline{f}}_{\text{lb},p} = \begin{pmatrix} \underline{f}_{\text{lb},1} - \underline{v}^*_1 - \mathbf{C}_1 \underline{x}^*_{p-1} \\ \vdots \\ \underline{f}_{\text{lb},p-1} - \underline{v}^*_{p-1} - \mathbf{C}_{p-1} \underline{x}^*_{p-1} \\ \underline{f}_{\text{lb},p} - \mathbf{C}_p \underline{x}^*_{p-1} \end{pmatrix} \,, \text{ and } \quad \tilde{\underline{f}}_{\text{ub},p} = \begin{pmatrix} \underline{f}_{\text{ub},1} - \underline{v}^*_1 - \mathbf{C}_1 \underline{x}^*_{p-1} \\ \vdots \\ \underline{f}_{\text{ub},p-1} - \underline{v}^*_{p-1} - \mathbf{C}_{p-1} \underline{x}^*_{p-1} \\ \underline{f}_{\text{ub},p} - \mathbf{C}_p \underline{x}^*_{p-1} \end{pmatrix} \,,$$

the optimization problem (2.19) can be written as the standard quadratic problem

$$\begin{aligned} \underline{\zeta}^*_p = \underset{\underline{\zeta}_p}{\operatorname{argmin}} & \frac{1}{2} \underline{\zeta}_p^\mathsf{T} \mathbf{H}_p \underline{\zeta}_p + \underline{g}_p^\mathsf{T} \underline{\zeta}_p \\ \text{subject to } & \tilde{\underline{f}}_{\text{lb},p} \leq \tilde{\mathbf{C}}_p \underline{\zeta}_p \leq \tilde{\underline{f}}_{\text{ub},p} \,, \end{aligned} \tag{2.21}$$

which is solved iteratively for each task, starting with $p = 1$. In practice, $\mathbf{H}_p$ is only positive semi-definite, which is an issue for some QP solvers [43, 44]. For this reason, small $\epsilon_{\text{reg}} > 0$ are added to

Algorithm 2.1 Optimization procedure for hierarchical quadratic problems.

Input: $p_{\max}$ hierarchical tasks $T_1, \dots T_{p_{\max}}$
Output: Optimal solution $\underline{x}^*_{p_{\max}}$

$\mathbf{Z} \leftarrow \mathbf{I}_{n \times n}$
if $\mathbf{C}_1$ is $[\,]$ **then** `// Optimized case of first iteration`
  $\underline{x}^*_1 \leftarrow$ particular solution of $\mathbf{A}_1 \underline{x}^*_1 = \underline{b}_1$
  $\underline{v}^*_1 \leftarrow [\,]$
else `// Unoptimized case of first iteration`
  Calculate $\mathbf{H}_1$, $\underline{g}_1$, $\tilde{\mathbf{C}}_1$, $\underline{\tilde{f}}_{\text{lb},1}$, $\underline{\tilde{f}}_{\text{ub},1}$ according to (2.20) with $p = 1$
  $\mathbf{H}_{1,[1:n,1:n]} \leftarrow \mathbf{H}_{1,[1:n,1:n]} + \epsilon_{\text{reg}} \mathbf{I}$ `// Regularization`
  $\begin{pmatrix} \underline{x}^*_1 \\ \underline{v}^*_1 \end{pmatrix} \leftarrow$ solution of (2.21) with $p = 1$ `// Solve quadratic program`
end if
for $p = 2$ to $p_{\max}$ **do** `// Remaining iterations`
  $\mathbf{Z} \leftarrow \mathbf{Z} \operatorname{kern}(\mathbf{A}_{p-1} \mathbf{Z})$ `// Null space update`
  if $\mathbf{Z}$ is $[\,]$ **then** `// Early stop when null space is empty`
    return $\underline{x}^*_{p-1}$
  end if
  Calculate $\mathbf{H}_p$, $\underline{g}_p$, $\tilde{\mathbf{C}}_p$, $\underline{\tilde{f}}_{\text{lb},p}$, $\underline{\tilde{f}}_{\text{ub},p}$ according to (2.20)
  $\mathbf{H}_{p,[1:n_{\text{ns},p},1:n_{\text{ns},p}]} \leftarrow \mathbf{H}_{1,[1:n_{\text{ns},p},1:n_{\text{ns},p}]} + \epsilon_{\text{reg}} \mathbf{I}_{n_{\text{ns},p} \times n_{\text{ns},p}}$ `// Regularization`
  $\underline{\tilde{f}}_{\text{lb},p} \leftarrow \underline{\tilde{f}}_{\text{lb},p} - \epsilon_{\text{lim}}(p-1)\underline{1}$, $\underline{\tilde{f}}_{\text{ub},p} \leftarrow \underline{\tilde{f}}_{\text{ub},p} + \epsilon_{\text{lim}}(p-1)\underline{1}$ `// Relaxation`
  $\begin{pmatrix} \underline{z}^*_p \\ \underline{v}^*_p \end{pmatrix} \leftarrow$ solution of (2.21) `// Solve quadratic program`
  $\underline{x}^*_p \leftarrow \underline{x}^*_{p-1} + \mathbf{Z} \underline{z}^*_p$
end for
return $\underline{x}^*_{p_{\max}}$

the diagonal of the top-left submatrix of $\mathbf{H}_p$. In addition, numerical inaccuracies can yield infeasible inequality constraints after several iterations. To mitigate this, the inequality bounds $\underline{\tilde{f}}_{\text{lb},p}$ and $\underline{\tilde{f}}_{\text{ub},p}$ can be relaxed by adding $\pm\epsilon_{\text{lim}}(p-1)$ with small $\epsilon_{\text{lim}} > 0$. The performance of the algorithm can be increased if the task with the highest priority has an empty inequality constraint. In this case, $\underline{x}^*_1$ is calculated directly as a particular solution of

$$\mathbf{A}_1 \underline{x}^*_1 = \underline{b}_1 \,.$$

The resulting procedure for solving hierarchical quadratic problems is summarized in Algorithm 2.1.

CHAPTER 3

Conception of HapticGiant

Contents

> By failing to prepare,
> you are preparing to fail.
>
> BENJAMIN FRANKLIN (1706 – 1790)

HapticGiant is designed to be a universal kinesthetic haptic interface, which implies that the list of requirements is long and diverse. As a result, developing a suitable system concept is a challenging task requiring many trade-offs and a lot of theoretical conception, especially before the first hardware iteration can be built. For this reason, this chapter is devoted to the conception of HapticGiant. To maintain a systematic approach, we first cover the requirements in Section 3.1. Then, a concept for HapticGiant's manipulator is presented in Section 3.2 and elaborated in Section 3.3 using a novel optimization-driven design approach. The discussion in Section 3.4 concludes this chapter.

This chapter is based on results presented in the author's publication [O4].

3.1 Design Goals

The properties of the ideal, universal kinesthetic haptic device can be split into the three categories *user experience*, *research*, and *applications*.

In terms of user experience, the system is expected to provide superior haptic transparency. This means that the end effector should cover as much as possible from the user's workspace with six

Cartesian DOF. Note, that this does not necessarily dictate a very large dexterous workspace, but rather a workspace that fits the user's properties. Haptic transparency also requires a sufficient range of achievable end effector velocities and accelerations. As for the workspace, *adequate* here means that the capabilities should be on par with or surpass those of the human user. For optimal dynamic behavior, the manipulator should exhibit high stiffness, low inertia, and actuators with a high mechanical bandwidth. Ideally, the system should be able to render TEs with very low inertia objects and free space. The rendering of stiff contacts is achieved through high force and torque output capabilities in combination with position and orientation sensing. Beyond the haptic quality, the user's safety during physical human-machine interaction is a matter of priority. Consequently, appropriate safety measures, including collision detection and avoidance, must be in place.

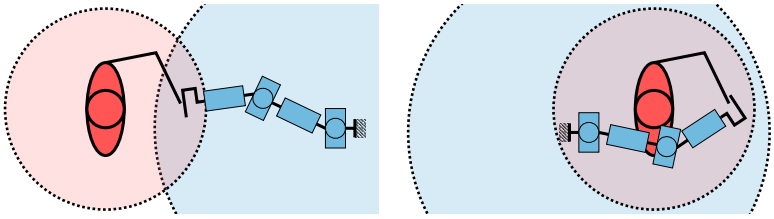
From a research perspective, HapticGiant is required to offer significant flexibility and the ability to adapt to evolving needs. The system's extensibility must be enabled by well-defined interfaces for communicating with other internal or external systems. A modular approach to hardware and software design is essential to facilitate the easy replacement and addition of components. In order to be future-proof, the system should possess plenty of computational power to run complex control algorithms. Additionally, the system should be designed for high repeatability and require comparatively low effort for software development to streamline the research and development workflow.

On the application side, HapticGiant must provide a high level of reconfigurability and reusability. For motion compression [45, 46], a very large workspace is required, and the user must be able to walk freely within this workspace while maintaining physical contact with the end effector. Furthermore, the user might walk in circles for an indefinite duration, requiring an infinite manipulator rotation around the yaw axis. In general, HapticGiant will be used in simulated and remote TEs, which require appropriate visualization tools, using either AR, VR, or a mixture of both. For the force feedback, suitable haptic rendering algorithms with appropriate interfaces for the description of what to render are necessary.

3.2 Manipulator Concept

As presented in Section 1.2.1, a variety of kinesthetic haptic devices featuring different kinematic structures have been designed. This indicates that the design space is very complex and that major design elements stem from existing concepts or are motivated by the learnings from previous device generations.

A notable example of this can be observed in the *VISHARD* family of kinesthetic haptic devices with three to ten DOF that is presented in [19]. In the first generation (VISHARD3), the authors probably intended to explore actuator technology and state-of-the-art force control concepts, but the workspace of the resulting device is rather small and the versatility with three translational end effector DOF is limited. Using similar mechanical concepts, a version with six DOF (VISHARD6) was developed in [47], which is meant as a universal haptic device with a larger workspace. The authors justify some of their design choices. For example, the first three joints are arranged in a planar selective compliance assembly robot arm (SCARA) configuration to avoid the need for gravity compensation. Joints 4 and 5 are used to adjust the pitch angle and the height of the end effector, and joint 6 is added to add the remaining rotational DOF to the end effector. Furthermore, some link lengths are chosen based on arguments regarding self-collisions or singularities. In [18], a highly redundant version with ten DOF (VISHARD10) is presented to improve the workspace size and to reduce the singularities visible to the user during operation. The high level of redundancy is justified by the idea of building a universal device that can be reconfigured depending on the workspace requirements of the actual



(a) Opposite configuration. **(b)** Enclosing configuration.

Figure 3.1: Top view of a commercial serial manipulator in opposite and enclosing configuration. The effectively usable workspace is either very limited in opposite configurations or collisions between the manipulator (blue) and the user (red) are likely to occur in enclosing configurations as the manipulator design is not optimized for human-robot interaction.

interaction task. The resulting topology is based on qualitative arguments and borrows some ideas from the previous designs, i.e., that the first joints are arranged in a planar configuration.

Another example of the conception of a haptic manipulator is given in [48], where the *ANYexo* exoskeleton for upper limb rehabilitation is presented, whose primary design goal is to achieve high coverage of the range of motion during activities of daily living (ADL). Based on performance reports in the literature, it was decided to reflect the movements of the sternoclavicular joint in the shoulder girdle using two actuated joints. The spherical movement of the glenohumeral joint is realized using three joints with perpendicular axes, also based on arguments from the literature. The orientation of these joints relative to the user is based on manipulability arguments. In a second iteration [49], ANYexo is expanded with a fully actuated wrist. This is achieved by adding three joints that correspond to forearm pronation/supination, hand flexion/extension, and radial/ulnar deviation. The exact alignment of these joints relative to the user and relative to each other is determined using the available installation space and anatomical arguments.

The concept of HapticGiant is no exception to this. The first realization of a large-scale kinesthetic haptic interface at the author's lab can be found in [50, 51], where a wheeled robot is combined with a table-top kinesthetic haptic interface. Over time, this idea evolved to the interface in [22, 23] with a SCARA manipulator attached to a 2D linear prepositioning unit (PPU). The end effector height is adjusted with a linear axis, resulting in a total of five joints and three translational DOF in Cartesian space. In [33], it was decided to remove the linear axis for height adjustment in favor of a new manipulator design with a parallel SCARA in a user-enclosing configuration. As a result, the position coverage of the human arm without PPU and the utilizable workspace size were increased. This comes at the cost of having only two active translational and one passive rotational DOF at the end effector, which considerably limits the range of potential applications.

With this knowledge and the design goals from the previous section in mind, the kinematic topology of HapticGiant was developed. In an early stage, standard industrial manipulators were considered to be used as haptic manipulator. With many reports of such setups in the literature [52–54], this seems to be a cost-effective and easy-to-implement solution. However, early experiments with a *Universal Robots UR16e* [D2] in the context of [O5] revealed that collaborative robots are unsuitable for this task. First of all, their kinematic properties are optimized for manufacturing tasks and thus not suitable for kinesthetic haptic feedback as illustrated in Figure 3.1. Second, the safety features cannot be deactivated beyond a certain level, which caused many involuntary stops during the experiments. Lastly, the robots do not provide any access to their inner control loops and the interfaces are limited. For example, the UR16e only accepts position and velocity setpoints at a maximal rate of 500 Hz.

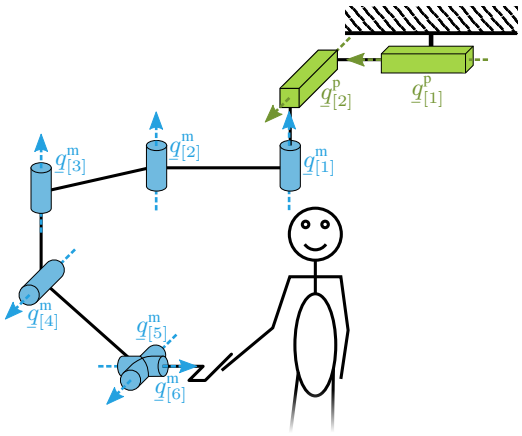


Figure 3.2: The proposed kinematic concept with a two DOF PPU and a six DOF manipulator.

For this reason, it was decided to design a custom manipulator for HapticGiant. Based on the experiences with the direct predecessors [33], a user-enclosing manipulator configuration was chosen to increase the workspace size and to reduce the risk of manipulator-user collisions. This is also in agreement with the findings in [55], where an enclosing configuration is preferred. Based on that, the proposed manipulator topology from Figure 3.2 was developed. Two prismatic joints forming a PPU, that is moving above the user's head, are used to create a very large workspace in which the user can walk freely. During force-torque rendering, the movement of the user's upper limb is reflected by a manipulator with six revolute joints, whose base is attached to the PPU in a way that guarantees sufficient head clearance. Similar to [47], the first three manipulator joints form a planar SCARA segment covering horizontal translations and rotations around the yaw axis. Due to the experiences with the prismatic height adjustment in [22], the height adjustment is realized by a fourth revolute joint with a horizontal rotation axis. The fifth joint enables changing the end effector inclination independent of the desired height. To cover all Cartesian DOF, a sixth and last joint is added. The axes of the last two joints intersect perpendicularly to simplify the inverse kinematics problem as explained in Section 4.1.4. One may also interpret the arrangement of joints 4 to 6 as a mirror image of the human forearm. Therefore, these joints may be referred to as elbow and wrist, respectively, in the following. The manipulator itself does not have redundancies. Singularities, that would be mitigated by redundancy, are not predominant as the only relevant singularity appears when the axis of manipulator joint 6 is aligned with joints 1 to 3. Other singularities, such as the alignment of the first three joints along a straight line, are outside of the intended operating conditions.

3.3 Optimization-Driven Dimensioning

Topology decisions can be based on qualitative arguments. However, the actual dimensioning process is far more difficult because there are many interdependent parameters, such as link lengths, joint sizes, and transmission ratios. Furthermore, the exact dimensioning of a manipulator depends on the desired task.

For this reason, quantitative methods to assess a particular manipulator design were developed in the past. A famous example is *Yoshikawa's manipulability index* [56], which corresponds to the product of the singular values of the end effector Jacobian. With the underlying singular value decomposition (SVD) of the Jacobian, the reachable set of velocities in Cartesian space can be visualized as a manipulability ellipsoid. Later, this concept was extended to the *dynamic manipulability* [57] to quantify the acceleration capabilities of a manipulator by taking into account the link inertia. The

analysis of the singularity values has also found its way into the design of kinesthetic haptic interfaces, where appropriate metrics are used to verify the velocity isotropy over the device's workspace [58].

The problem with the presented manipulability and isotropy indices is that they are purely local measures. In contrast to that, [59] formulates a metric to quantify the dynamic end effector capabilities using an operational space formulation, that also takes into account the loss of acceleration isotropy due to gravity. This measure is used in a global optimization scheme to maximize the isotropy of the manipulator acceleration. Although the formulated optimization problem is mathematically sound, the authors do not provide any information on how to solve it efficiently, especially in large joint configuration spaces. Similarly, [60] presents isotropy indices for the output acceleration and velocity of a manipulator, that are used in a global optimization problem and an accompanying optimization algorithm. Although the authors prove the effectiveness of their method by optimizing the design of a 2D haptic pen, it is not applicable for dimensioning the manipulator of HapticGiant, because the human arm is anisotropic in general. Besides velocity and acceleration capabilities, the reachability of poses of interest is an important factor when it comes to designing manipulators. In this context, a method to optimize the link lengths for reaching a set of uniformly distributed, pre-defined poses is described in [47].

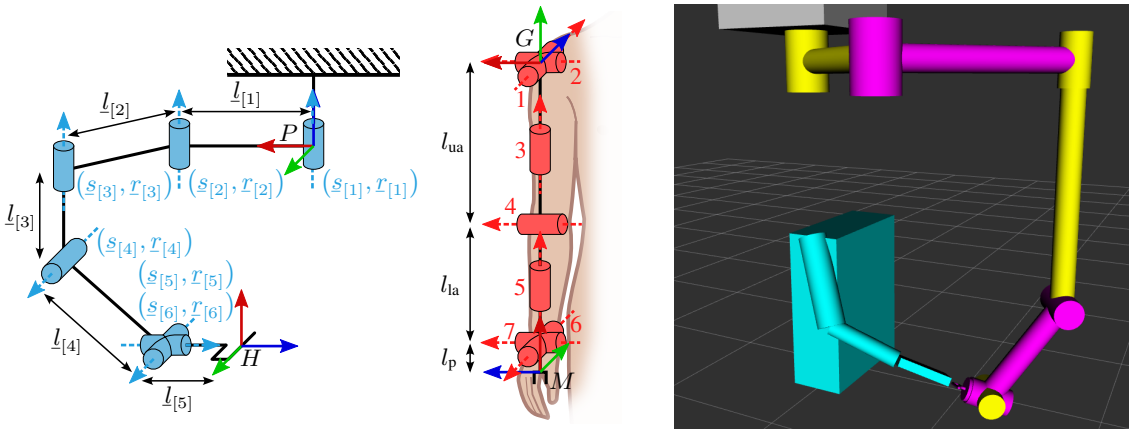
All methods mentioned so far do not optimize with respect to the properties of the human arm. In contrast to that, the authors of [61] state that the design of an exoskeleton should be optimized for the human arm and the tasks that the user intends to accomplish. Therefore, a detailed study with human subjects performing ADL was conducted. The resulting qualitative insights were used to improve an existing exoskeleton regarding its singularities. The study in [55] goes one step further by quantifying the workspace overlap of a human arm and a given manipulator to optimize the kinesthetic haptic device from [26]. To achieve this, a pose coverage metric for a given manipulator setup is calculated using kinematic models of the human arm and the manipulator. Velocity and acceleration capabilities are not taken into account. Furthermore, the authors do not provide information on how to apply their method to high-dimensional parameter spaces. An interesting contribution in this regard can be found in [62], where the leg parameters, including transmission ratios, spring parameters, and link lengths, for a walking robot are optimized. The presented metrics, such as energy consumption and peak torque, are formulated depending on a motion plan.

None of the existing methods is suitable to solve the given dimensioning problem when the goal is to achieve the highest level of compatibility with the human arm. To overcome this issue, a novel optimization-driven dimensioning approach for kinesthetic haptic interfaces is presented in the following. We commence with a presentation of the design space and a detailed analysis of the human arm. Subsequently, the optimization process with objective functions for pose, velocity, and acceleration coverage is explained, before the method's results for dimensioning HapticGiant's manipulator are evaluated.

3.3.1 Design Space

As illustrated in Figure 3.3a, the manipulator concept from Section 3.2 has many yet unknown parameters: Each joint is characterized by the joint size $\underline{s}_{[i]}$ selected from the set of mechanical dimensions S and the transmission ratio $\underline{r}_{[i]}$ selected from the set of available discrete transmissions R. The maximum output torque, velocity, and mass for each joint are determined by the look-up-tables

$$\begin{aligned} \underline{\tau}_{\max[i]} &= f_\tau(\underline{s}_{[i]}, \underline{r}_{[i]}) \,, \\ \underline{\dot{q}}^{\mathrm{m}}_{\max[i]} &= f_{\dot{q}}(\underline{s}_{[i]}, \underline{r}_{[i]}) \,, \quad \text{and} \\ \underline{m}^{\mathrm{m}}_{[i]} &= f_m(\underline{s}_{[i]}) \,, \end{aligned}$$



(a) Model of the manipulator. **(b)** Model of the human arm. **(c)** Interaction between both models.

Figure 3.3: Illustration of the models used for the manipulator design, including lengths, frames, and other properties. Adapted from [O4].

Joint	θ	d	a	α
1	$\underline{q}^{\text{h}}_{[1]} + 90°$	0	0	90°
2	$\underline{q}^{\text{h}}_{[2]} + 90°$	0	0	90°
3	$\underline{q}^{\text{h}}_{[3]}$	$-l_{\text{ua}}$	0	−90°
4	$\underline{q}^{\text{h}}_{[4]}$	0	0	90°
5	$\underline{q}^{\text{h}}_{[5]} - 90°$	$-l_{\text{la}}$	0	90°
6	$\underline{q}^{\text{h}}_{[6]} + 90°$	0	0	−90°
7	$\underline{q}^{\text{h}}_{[7]}$	0	$-l_{\text{p}}$	0

Table 3.1: Denavit-Hartenberg (DH) parameters of the human arm model. Taken from [O4].

respectively. In addition, the unknown link lengths in $\underline{l} \in \mathbb{R}^5$ must be selected. In summary, the unknown parameters can be summarized in the decision vector

$$\underline{p}^{\mathsf{T}} = \begin{pmatrix} \underline{l}^{\mathsf{T}} & \underline{s}^{\mathsf{T}} & \underline{r}^{\mathsf{T}} \end{pmatrix} \in \mathbb{R}^5 \times S^6 \times R^6 \,.$$

The goal now is to find an appropriate value of $\underline{p}$, that represents the optimal manipulator design with respect to the properties of the human arm in terms of pose, velocity, and acceleration coverage. Moreover, the resulting design should have sufficient output force and torque capabilities. For HapticGiant, no specific interaction tasks are considered. For this reason, we assume that the desired forces and torques are isotropic and have a fixed maximum magnitude. Obviously, these goals are contradictory. For example, long links are advantageous for a large workspace and large end effector velocities but lead to reduced acceleration capabilities due to increased inertia. Similarly, larger transmission ratios provide higher force-torque capabilities at the expense of the velocity capabilities. It is therefore important to find a reasonable compromise between the different objectives.

3.3.2 Analysis of the Human Arm

To obtain a manipulator design that is highly compatible with the human arm, the latter has to be studied first. In the following, this is done using the model from Figure 3.3b and Table 3.1, where the human arm is modeled as a serial kinematic chain with seven rotational DOF. In this model, the shoulder girdle of the human arm is fixed and the subject is assumed to be standing upright. The

	Joint Limits of $\underline{q}^{\mathrm{h}}_{[i]}$ in °								$\underline{\dot{q}}^{\mathrm{h}}_{\mathrm{max}[i]}$ in °/s	$\underline{\ddot{q}}^{\mathrm{h}}_{\mathrm{max}[i]}$ in °/s^2
	[66]		[65]		[67]		selected			
Joint	min	max	min	max	min	max	min	max	[65]	[65]
1	-30	180	-14	134	-50	180	-30	90	172	1312
2	-50	180	-18	62	-30	130	-50	90	137	950
3	-80	80	-72	55	-90	60	-80	80	141	1049
4	0	145	42	163	0	160	0	160	173	1266
5	-90	85	–*	–*	-85	80	-90	85	486	4344
6	-85	85	-40	55	-80	90	-85	85	233	2790
7	-45	15	–*	–*	-30	15	-45	15	204	2476

* Data was omitted due to implausible values.

Table 3.2: Joint limits of the human arm model regarding position, velocity, and acceleration. Taken from [O4].

spherical glenohumeral joint, which represents the predominant shoulder movements, is modeled as a series of three revolute joints within the shoulder and the upper arm following the discussion in [55]. Thus, the first three joints model shoulder abduction/adduction, shoulder flexion/extension, and shoulder internal/external rotation, respectively. The elbow flexion/extension is reproduced using a single revolute joint. For the nearly spherical wrist movements, a combination of three revolute joints is used analogous to the shoulder. The first joint in the forearm represents wrist pronation/supination, the second wrist flexion/extension, and the third radial/ulnar deviation. As sketched in Figure 3.3c, the manipulator tool center point (TCP) at frame H coincides with the center of the palm at frame M and the root joint of the manipulator at frame P is placed above the shoulder joint at frame G with sufficient head clearance, resulting in the constant offsets

$$\underline{x}^{P}_{PG} = \begin{pmatrix} 0 \\ 0 \\ -0.61 \end{pmatrix} \mathrm{m}\,, \quad \underline{x}^{M}_{MH} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \mathrm{m}\,, \quad \mathbf{C}^{P}_{G} = \begin{pmatrix} 0 & 0 & -1 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \quad \text{and} \quad \mathbf{C}^{M}_{H} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \end{pmatrix}.$$

Based on a median body height of $1.75\,\mathrm{m}$ for males[1] [63] and the anthropometric data reported in [64], the link lengths were determined to be $l_{\mathrm{ua}} = 0.326\,\mathrm{m}$, $l_{\mathrm{la}} = 0.256\,\mathrm{m}$, and $l_{\mathrm{p}} = 0.095\,\mathrm{m}$. The origin of the shoulder frame is then located $1.43\,\mathrm{m}$ above the ground. To complete the kinetostatic model, the joint limits in Table 3.2 were chosen based on values from the literature, keeping in mind that extreme movements behind the user's back and above the head are feasible, but rather irrelevant during ADL and the interaction with HapticGiant. Finally, the maximum joint velocities $\underline{\dot{q}}^{\mathrm{h}}_{\mathrm{max}}$ and accelerations $\underline{\ddot{q}}^{\mathrm{h}}_{\mathrm{max}}$ were assigned according to Table 3.2. The underlying data is from [65] and was captured during ADL. Although this means that the limits are not the transient maximum of what the human arm can achieve, it is a good representation of what to expect during normal interaction.

Workspace

With the given kinetostatic data, the workspace of the human arm can be analyzed in detail using Cartesian pose samples. This has the advantage that all regions and orientations are equally well represented in contrast to joint space sampling. For reproducibility, we propose a deterministic sampling strategy, where the translational part is discretized using a 3D lattice with a resolution of $0.1\,\mathrm{m}$. The resulting set of positions is called W_{tran} and has $14 \times 14 \times 7$ elements. To sample

[1] Male data was actively chosen over female data to make the design more challenging as females are statistically smaller, creating a less challenging design problem.

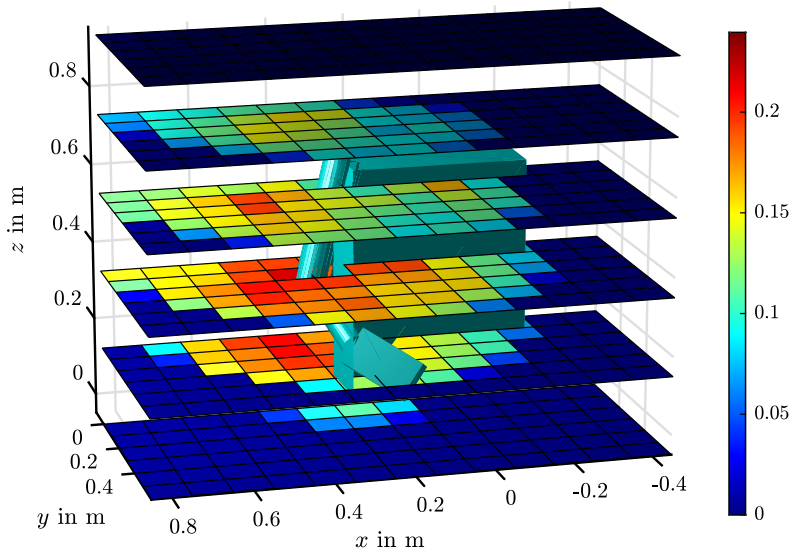


Figure 3.4: Visualization of the Cartesian workspace of the human arm. For the visualization, $\left|Q^{\mathrm{h}}(\mathbf{T}_M^G)\right|$ was summed over all orientations in W_{rot} for each position and divided by $|W_{\mathrm{rot}}|$. Taken from [O4].

the orientation, 505 deterministic samples from the hypercube in $\mathbb{R}^3$ are generated using a Fibonacci lattice [68, 69] in the first step. Then, the samples are transformed to nearly equidistant samples on the 3-hemisphere, which can be interpreted as quaternions in the desired set of orientations $W_{\mathrm{rot}} \subset SO(3)$, using the equiareal mapping from [70]. The cross-product $W = W_{\mathrm{tran}} \times W_{\mathrm{rot}} \subset SE(3)$ with 692 860 elements encompasses all poses that need to be checked for reachability.

In the next step, the inverse kinematics problem for the human arm is solved for each pose $\mathbf{T}_M^G \in W$. Due to the redundancy of the human arm, an infinite number of solutions may exist for a single configuration, which would multiply the computational effort. In the literature, however, it was discovered that the redundancy resolution of the human arm is predictable [71,72], reducing the relevant number of solutions to one or zero. In combination with the fact, that most inverse kinematics solvers cannot return a continuous set of solutions, it was decided to discretize joint 3, which corresponds to the human elbow, with 17 samples between its minimum and maximum joint angle to get a representative subset of the continuous solution set. The inverse kinematics problem is then solved for each of these samples and the solution with the best manipulability index according to [56] is selected. After this procedure, each pose in W is assigned an empty or one-element set $Q^{\mathrm{h}}(\mathbf{T}_M^G)$ describing the configuration to reach a given hand pose, if any is available.

Using this information, the visualization of the human arm workspace from Figure 3.4 can be generated. As one might expect from personal experience, the regions in front of the human body and on the side of the analyzed arm, between the chest and navel, are the most accessible. The coverage magnitude, which is always below 25 %, may look counterintuitive at first. Nevertheless, a quick self-experiment, where all hand orientations in $SO(3)$ are tried to be reached systematically while maintaining the position of the palm center, reveals that the produced data is indeed plausible.

Velocity and Acceleration Capabilities

To incorporate the end effector velocity capabilities into the analysis, the kinematic relation

$$\begin{pmatrix} \dot{\underline{x}}_{GM}^G \\ \underline{\omega}_{GM}^G \end{pmatrix} = \boldsymbol{\mathcal{J}}_{\underline{\nu}_{GM}^G}^{\mathrm{h}}(\underline{q}^{\mathrm{h}})\, \dot{\underline{q}}^{\mathrm{h}}$$

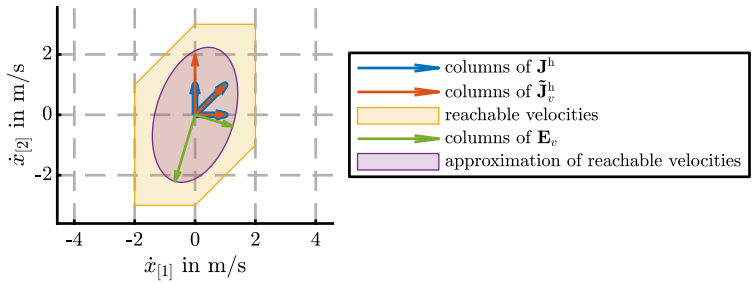


Figure 3.5: The velocity ellipsoid is a reasonable approximation for the set of actually reachable velocities. Adapted from [O4].

is used as specialization of (2.9). Based on a representative set of human joint velocities, a set of end effector velocities could be obtained for rating a given haptic manipulator design. However, this becomes computationally intractable with seven different joint velocities, when taking into account that we already sample along six Cartesian DOF. For this reason, approximating the set of feasible human end effector velocities at a given configuration $\underline{q}^{\mathrm{h}}$ seems natural. To achieve this, we first introduce the normalized joint velocities

$$\dot{\underline{\tilde{q}}}^{\mathrm{h}} = \operatorname{diag}\left(\dot{\underline{q}}^{\mathrm{h}}_{\max}\right)^{-1} \dot{\underline{q}}^{\mathrm{h}} \,. \tag{3.1}$$

The normalized end effector Jacobian and its SVD then become

$$\mathbf{J}_v^{\mathrm{h}} = \boldsymbol{\mathcal{J}}^{\mathrm{h}}_{\underline{\nu}^G_{GM}}\left(\underline{q}^{\mathrm{h}}\right) \operatorname{diag}\left(\dot{\underline{q}}^{\mathrm{h}}_{\max}\right) = \mathbf{U}_v \mathbf{S}_v \mathbf{V}_v^{\mathsf{T}} \,, \tag{3.2}$$

where $\mathbf{U}_v \in \mathbb{R}^{6\times 6}$ and $\mathbf{V}_v \in \mathbb{R}^{7\times 7}$ are orthonormal matrices and $\mathbf{S}_v \in \mathbb{R}^{6\times 7}$ is a diagonal matrix. In this case, the columns of

$$\mathbf{E}_v\left(\underline{q}^{\mathrm{h}}\right) = \mathbf{U}_v \mathbf{S}_{v[:,1:6]} \in \mathbb{R}^{6\times 6} \tag{3.3}$$

contain the principal axes of the end effector velocity ellipsoid, which approximates the set of feasible end effector velocities as illustrated in the following example.

Example 3.1:

Let us consider a redundant manipulator in the 2D Cartesian plane with the Jacobian

$$\boldsymbol{\mathcal{J}}^{\mathrm{h}}_{\underline{v}} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \mathrm{m/rad}$$

and the maximum joint velocities

$$\dot{\underline{q}}^{\mathrm{h}\mathsf{T}}_{\max} = \begin{pmatrix} 1 & 2 & 1 \end{pmatrix} \mathrm{rad/s} \,.$$

The resulting approximation of the reachable end effector velocities using the velocity ellipsoid is visualized in Figure 3.5. Analog to (3.3)*, the principal axes are the columns of*

$$\mathbf{E}_v = \mathbf{U}_v \mathbf{S}_{v[:,1:2]} = \begin{pmatrix} -0.6673 & 1.2469 \\ -2.2040 & -0.3775 \end{pmatrix} \mathrm{m/s} \,.$$

The acceleration capabilities are characterized similarly. With (2.11), the end effector acceleration is

$$\begin{pmatrix} \ddot{\underline{x}}^G_{GM} \\ \dot{\underline{\omega}}^G_{GM} \end{pmatrix} = \boldsymbol{\mathcal{J}}^{\mathrm{h}}_{\underline{\nu}^G_{GM}}\left(\underline{q}^{\mathrm{h}}\right) \ddot{\underline{q}}^{\mathrm{h}} + \dot{\boldsymbol{\mathcal{J}}}^{\mathrm{h}}_{\underline{\nu}^G_{GM}}\left(\underline{q}^{\mathrm{h}}, \dot{\underline{q}}^{\mathrm{h}}\right) \dot{\underline{q}}^{\mathrm{h}} \,.$$

For ADL, we assume that high absolute joint accelerations occur only when the absolute velocities are low. Accordingly, the second term can be omitted, yielding the approximation

$$\begin{pmatrix} \ddot{\underline{x}}_{GM}^{G} \\ \dot{\underline{\omega}}_{GM}^{G} \end{pmatrix} \approx \underline{\mathcal{J}}_{\underline{\nu}_{GM}^{G}}^{\mathrm{h}}\left(\underline{q}^{\mathrm{h}}\right) \ddot{\underline{q}}^{\mathrm{h}} \,.$$

Analog to (3.1) to (3.3)

$$\mathbf{J}_a^{\mathrm{h}} = \underline{\mathcal{J}}_{\underline{\nu}_{GM}^{G}}^{\mathrm{h}}\left(\underline{q}^{\mathrm{h}}\right) \operatorname{diag}\left(\ddot{\underline{q}}_{\max}^{\mathrm{h}}\right) = \mathbf{U}_a \mathbf{S}_a \mathbf{V}_a^{\mathsf{T}} \quad \text{and}$$
$$\mathbf{E}_a\left(\underline{q}^{\mathrm{h}}\right) = \mathbf{U}_a \mathbf{S}_{a[:,1:6]} \in \mathbb{R}^{6\times 6}$$

can be used to calculate the principal axes of the acceleration ellipsoid.

3.3.3 Optimization Problem

With the presented analysis, the goal of maximizing the compatibility with the human arm can be formulated as an optimization problem.

Objective Function

The scalar objective function

$$J(\underline{p}) = J_x(\underline{p}) + J_v(\underline{p}) + J_a(\underline{p}) + J_w(\underline{p}) \,, \tag{3.4}$$

depends on the previously defined parameter vector $\underline{p}$ and takes into account the *pose coverage* J_x, the *velocity coverage* J_v, the *acceleration coverage* J_a, and the *force-torque coverage* J_w. Without any specific information about the task, the four terms with identical ranges are weighted equally. In the following, the individual terms that shall be maximized during the optimization are explained.

1. For the pose coverage, the total number of palm poses that can be reached by the manipulator end effector is calculated by checking whether there is an inverse kinematics solution for each palm pose $\mathbf{T}_M^G \in W$, that is actually reachable by the human arm, i.e., $\left|Q^{\mathrm{h}}(\mathbf{T}_M^G)\right| > 0$. Mathematically, we can represent this using the nested sum

$$n_{\mathrm{reach}}^{\mathrm{m}}(\underline{p}) = \sum_{\mathbf{T}_M^G \in W} \sum_{\underline{q}^{\mathrm{h}} \in Q^{\mathrm{h}}(\mathbf{T}_M^G)} \left|Q^{\mathrm{m}}(\mathbf{T}_H^P, \underline{p})\right| \,.$$

   Here, $\mathbf{T}_H^P = \mathbf{T}_G^P \mathbf{T}_M^G \mathbf{T}_H^M$ is the desired Cartesian manipulator end effector pose with the fixed offsets $\mathbf{T}_G^P$ and $\mathbf{T}_M^G$. The empty or one-element set $Q^{\mathrm{m}}(\mathbf{T}_H^P, \underline{p})$ contains the inverse kinematics solutions for the manipulator as a function of the end effector pose and the design parameters. With the total number of reachable human hand poses in the analyzed workspace

$$n_{\mathrm{reach}}^{\mathrm{h}}(\underline{p}) = \sum_{\mathbf{T}_M^G \in W} \left|Q^{\mathrm{h}}(\mathbf{T}_M^G)\right| \,,$$

   the pose coverage is defined as

$$J_x(\underline{p}) = \frac{n_{\mathrm{reach}}^{\mathrm{m}}(\underline{p})}{n_{\mathrm{reach}}^{\mathrm{h}}(\underline{p})} \in [0,1] \,. \tag{3.5}$$

2. Starting with the human arm configuration $\underline{q}^{\mathrm{h}} \in Q^{\mathrm{h}}(\mathbf{T}_M^G)$ and the corresponding inverse kinematics solution for the manipulator $\underline{q}^{\mathrm{m}} \in Q^{\mathrm{m}}(\mathbf{T}_H^P, \underline{p})$, the principal axes of the human arm can be transformed into manipulator joint space using

$$\mathbf{A}_v\left(\underline{q}^{\mathrm{m}}, \underline{q}^{\mathrm{h}}, \underline{p}\right) = \left(\boldsymbol{\mathcal{J}}_{\nu_H^P}^{\mathrm{m}}(\underline{q}^{\mathrm{m}}, \underline{p})\right)^{-1} \begin{pmatrix} \mathbf{C}_G^P & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_G^P \end{pmatrix} \mathbf{E}_v(\underline{q}^{\mathrm{h}})\,. \tag{3.6}$$

In the resulting matrix, each column corresponds to the joint velocities required for reaching the respective principal axis of the velocity ellipsoid from (3.3). Using

$$\eta_v\big(\underline{q}^{\mathrm{m}}, \underline{q}^{\mathrm{h}}, \underline{p}\big) = \min\left(1, \min_{i,j}\left(\frac{\dot{\underline{q}}^{\mathrm{m}}_{\mathrm{max}[i]}}{|\mathbf{A}_{v[i,j]}\big(\underline{q}^{\mathrm{m}}, \underline{q}^{\mathrm{h}}, \underline{p}\big)\,|}\right)\right)\,, \tag{3.7}$$

this is converted into a dimensionless scalar value describing the worst-case coverage of the examined velocity ellipsoid for a single configuration. By summing over the entire analyzed workspace, the global velocity coverage

$$J_v(\underline{p}) = \frac{1}{n^{\mathrm{m}}_{\mathrm{reach}}(\underline{p})} \sum_{\mathbf{T}_M^G \in W} \sum_{\underline{q}^{\mathrm{h}} \in Q^{\mathrm{h}}(\mathbf{T}_M^G)} \sum_{\underline{q}^{\mathrm{m}} \in Q^{\mathrm{m}}(\mathbf{T}_H^P, \underline{p})} \eta_v\big(\underline{q}^{\mathrm{m}}, \underline{q}^{\mathrm{h}}, \underline{p}\big) \in [0,1] \tag{3.8}$$

is obtained. In contrast to (3.5), this expression is normalized with $n^{\mathrm{m}}_{\mathrm{reach}}$ instead of $n^{\mathrm{h}}_{\mathrm{reach}}$ to avoid penalizing unreachable hand poses twice.

3. The dynamic properties must be included in the manipulator model for rating the acceleration capabilities. For this reason, the motors are included as point masses depending on their size and simple geometries are used to model the inertia of the end effector and the links depending on the decision variable. With the resulting joint space inertia matrix $\mathbf{M}^{\mathrm{m}}(\underline{q}^{\mathrm{m}}, \underline{p})$, the principal axes of the acceleration ellipsoid can be transformed into corresponding manipulator joint torques using

$$\mathbf{A}_a\left(\underline{q}^{\mathrm{m}}, \underline{q}^{\mathrm{h}}, \underline{p}\right) = \mathbf{M}^{\mathrm{m}}(\underline{q}^{\mathrm{m}}, \underline{p}) \left(\boldsymbol{\mathcal{J}}_{\nu_H^P}^{\mathrm{m}}(\underline{q}^{\mathrm{m}}, \underline{p})\right)^{-1} \begin{pmatrix} \mathbf{C}_G^P & \mathbf{0} \\ \mathbf{0} & \mathbf{C}_G^P \end{pmatrix} \mathbf{E}_a(\underline{q}^{\mathrm{h}})$$

analogous to (3.6). In this formulation, the manipulator is assumed to have zero velocity. This approximation is reasonable, as the highest isotropic accelerations occur only at low velocities. Next, the local acceleration coverage

$$\eta_a\big(\underline{q}^{\mathrm{m}}, \underline{q}^{\mathrm{h}}, \underline{p}\big) = \min\left(1, \min_{i,j}\left(\frac{\underline{\tau}_{\mathrm{max}[i]}}{|\mathbf{A}_{a[i,j]}\big(\underline{q}^{\mathrm{m}}, \underline{q}^{\mathrm{h}}, \underline{p}\big)\,| + |\underline{\tau}^{\mathrm{m}}_{\mathrm{grav}[i]}(\underline{q}^{\mathrm{m}})|}\right)\right) \tag{3.9}$$

is defined. This equation is similar to (3.7) but takes into account the joint torque $\underline{\tau}^{\mathrm{m}}_{\mathrm{grav}}$ that is necessary to compensate for gravity at a given configuration. Integrating this over the entire workspace analogous to (3.8) eventually yields the global acceleration coverage

$$J_a(\underline{p}) = \frac{1}{n^{\mathrm{m}}_{\mathrm{reach}}(\underline{p})} \sum_{\mathbf{T}_M^G \in W} \sum_{\underline{q}^{\mathrm{h}} \in Q^{\mathrm{h}}(\mathbf{T}_M^G)} \sum_{\underline{q}^{\mathrm{m}} \in Q^{\mathrm{m}}(\mathbf{T}_H^P, \underline{p})} \eta_a\big(\underline{q}^{\mathrm{m}}, \underline{q}^{\mathrm{h}}, \underline{p}\big) \in [0,1]\,. \tag{3.10}$$

4. The last part of the objective function quantifies how well the manipulator can produce isotropic forces and torques with the maximum magnitude f_{max} and m_{max}, respectively. Based on the matrix of principal Cartesian wrenches

$$\mathbf{E}_w = \begin{pmatrix} f_{\mathrm{max}}\mathbf{I} & \mathbf{0} \\ \mathbf{0} & m_{\mathrm{max}}\mathbf{I} \end{pmatrix} \in \mathbb{R}^{6\times 6}\,,$$

the corresponding principal joint torque vectors can be calculated using

$$\mathbf{A}_w(\underline{q}^{\mathrm{m}}, \underline{q}^{\mathrm{h}}, \underline{p}) = \left(\boldsymbol{\mathcal{J}}^{\mathrm{m}}_{\underline{\nu}^P_H}(\underline{q}^{\mathrm{m}}, \underline{p})\right)^{\mathsf{T}} \mathbf{E}_w \,.$$

Analog to (3.9) and (3.10), the local and global force-torque coverage are then

$$\eta_w(\underline{q}^{\mathrm{m}}, \underline{q}^{\mathrm{h}}, \underline{p}) = \min\left(1, \min_{i,j}\left(\frac{\underline{\tau}_{\max[i]}}{|\mathbf{A}_{w[i,j]}(\underline{q}^{\mathrm{m}}, \underline{q}^{\mathrm{h}}, \underline{p})| + |\underline{\tau}^{\mathrm{m}}_{\mathrm{grav}[i]}(\underline{q}^{\mathrm{m}})|}\right)\right) \quad \text{and}$$

$$J_w(\underline{p}) = \frac{1}{n^{\mathrm{m}}_{\mathrm{reach}}(\underline{p})} \sum_{\mathbf{T}^G_M \in W} \sum_{\underline{q}^{\mathrm{h}} \in Q^{\mathrm{h}}(\mathbf{T}^G_M)} \sum_{\underline{q}^{\mathrm{m}} \in Q^{\mathrm{m}}(\mathbf{T}^P_H, \underline{p})} \eta_w(\underline{q}^{\mathrm{m}}, \underline{q}^{\mathrm{h}}, \underline{p}) \in [0, 1] \,.$$

Constraints

For the given problem, the sizes of the first two joints are fixed to the largest size, and the link lengths are constrained to $\underline{l}_{\mathrm{lb}} \leq \underline{l} \leq \underline{l}_{\mathrm{ub}}$ in order to obtain mechanically realizable solutions. Furthermore, the available motor sizes and transmission ratios are encoded into suitable integer constraints for $\underline{s}$ and $\underline{r}$. Lastly, a constraint for monotonically decreasing joint sizes from the base to the end effector, i.e., no joint may be larger than its predecessor in the kinematic chain, is added for sufficient mechanical stiffness of the resulting manipulator concept.

Optimization

The objective function (3.4) with the given constraints represents a non-linear, non-convex mixed-integer problem. Furthermore, gradient information is not readily available as standard implementations for calculating the Jacobian or the inertia matrix do not provide derivatives with respect to underlying design parameters. For this reason, a genetic algorithm with support for mixed-integer programs [73] is chosen. Besides having reasonable convergence properties for the given problem, genetic algorithms can be parallelized on a massive scale, which is beneficial for the large search space in combination with the complex objective function.

Implementation

With the goal of building a prototype in mind, the parameters of the optimization problem were chosen to be as realistic and close to the target hardware as possible. For this reason, a preliminary search was performed, which identified the *Sensodrive Sensojoint* [D3] series elastic actuators (SEAs) as a suitable joint series. The resulting look-up tables for maximum velocity, output torque, and joint mass depending on the joint parameters can be found in Table 3.3. In the above-defined mass model, the assumed motor weight is the nominal motor weight plus $20\,\%$ for the mounting brackets and cables. The links are assumed to be hollow tubes with an outer diameter of $100\,\mathrm{mm}$, a wall thickness of $2\,\mathrm{mm}$, and a weight of $4\,\mathrm{kg}$ per meter. The end effector is modeled with a weight of $0.5\,\mathrm{kg}$ that is evenly distributed across a cylinder with a length of $50\,\mathrm{mm}$ and a diameter of $100\,\mathrm{mm}$. The link length limits are set to $\underline{l}^{\mathsf{T}}_{\mathrm{lb}} = \begin{pmatrix}0.3 & 0.3 & 0.3 & 0.3 & 0.12\end{pmatrix}\mathrm{m}$ and $\underline{l}^{\mathsf{T}}_{\mathrm{ub}} = \begin{pmatrix}1.0 & 1.0 & 1.5 & 1.0 & 0.3\end{pmatrix}\mathrm{m}$. Furthermore, we assume that the design shall deliver isotropic forces and torques with the magnitudes $f_{\max} = 50\,\mathrm{N}$ and $\tau_{\max} = 10\,\mathrm{Nm}$, respectively.

The inverse kinematics solution for the human arm analysis is calculated very efficiently using a closed-form solution that was generated with *ikfast* [74]. The solution to the inverse kinematics problem of the manipulator is hand-crafted as explained in Section 4.1.4. All other kinematic and dynamic calculations, such as calculating the Jacobian, are implemented using the *Matlab Robotics System Toolbox* [75]. For the optimization, the genetic algorithm provided by the *Matlab Global Optimization*

	Maximum torque $\underline{\tau}_{\max[i]}$ in Nm						Maximum output velocity $\underline{\dot{q}}^{\mathrm{m}}_{\max[i]}$ in rpm						
	Transmission						Transmission						Mass
Size	1:31	1:51	1:81	1:101	1:121	1:161	1:31	1:51	1:81	1:101	1:121	1:161	$\underline{m}^{\mathrm{m}}_{[i]}$ in kg
Small (S)	9	18	23	28	–	–	220	134	84	67	–	–	1.3
Medium (M)	–	62	96	107	113	120	–	75	47	38	31	24	2.0
Large (L)	100	166	208	208	208	208	95	57	36	29	24	18	4.9

Table 3.3: Joint specifications for different motor sizes and transmissions. Data taken from [D3].

Cluster	# of runs	J	J_x	J_v	J_a	J_w	$\underline{l}^{\mathsf{T}}$ in cm	Joints
1	10	3.24	0.81	0.88	0.55	1.00	(50.2 64.3 79.9 37.5 12.0)	(L51 L51 M51 M51 S51 S51)
2	14	3.25	0.72	0.78	0.75	1.00	(50.8 59.0 75.0 32.3 12.0)	(L81 L51 M51 M51 S51 S51)
3	**7**	**3.24**	**0.88**	**0.88**	**0.48**	**1.00**	**(52.9 66.8 75.2 42.3 12.0)**	**(L51 L51 M51 M51 S51 S51)**

Table 3.4: Clusters of optimal solutions within the results of 31 randomly initialized optimizer runs. The optimal joint configuration $(\underline{r}, \underline{s})$ is encoded in the last column from base to end effector. For example, *M51* indicates a medium-sized joint with a 1:51 transmission. The selected parameter vector is printed in bold. Adapted from [O4].

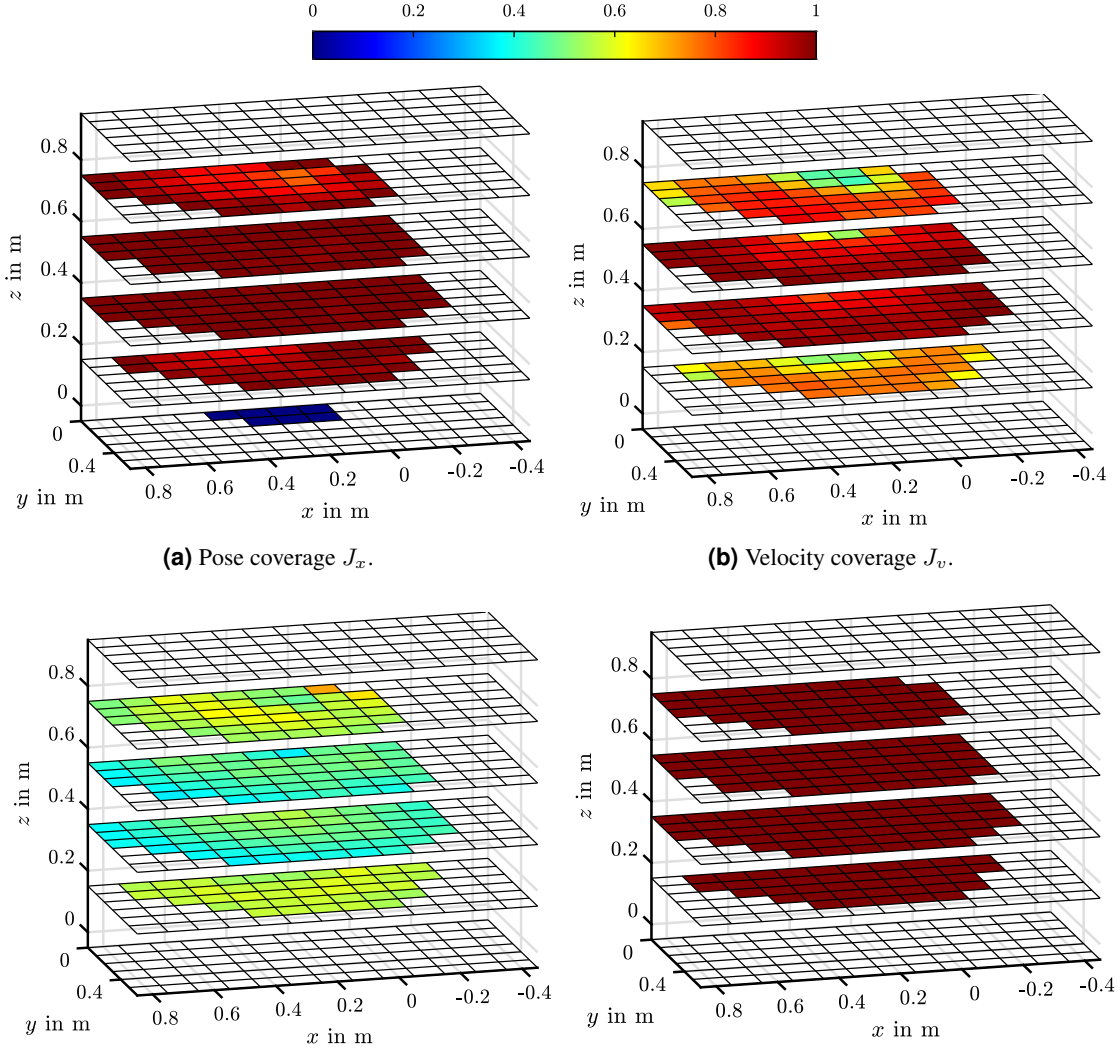
Toolbox [76] is leveraged in combination with the *Matlab Parallel Computing Toolbox* [77] for nearly linear speed-up on a population of size 200.

3.3.4 Results

The presented problem was optimized on an *Intel Xeon Gold 6230* central processing unit (CPU) with 40 threads. Within 31 runs, which took between ten and 29 hours to complete, the genetic algorithm reliably converged to solutions with objective values J between 3.236 and 3.245 representing different local minima. The results were found to be insensitive to the resolution of W_{rot} and the hyperparameters of the genetic algorithm. Thus, denser Cartesian pose samples do not lead to different results. A closer look at the optimization results using k-means clustering and the average silhouette score [78] reveals that the optimized parameters form three different clusters, of which each represents a different local minimum with different contributions to the individual cost terms. Table 3.4 lists the resulting clusters with motor configurations and centroids for the link lengths $\underline{l}$. The average Euclidean distance between the individual link length and the corresponding cluster centroid link length is $6.5\,\mathrm{mm}$. Compared to the average distance between the cluster centroid link length and the overall link length centroid, which is $58.0\,\mathrm{mm}$, this confirms that the clusters found are valid.

Theoretically, there are several nearly equally optimal solutions according to the unweighted objective function from (3.4). In practice, however, these solutions will have different characteristics, as seen in Table 3.4. Therefore, it is necessary to select one of these *optimal trade-offs* for the realization in a prototype. For HapticGiant, the cluster with the highest pose coverage J_x was selected, resulting in a total link length of $2.49\,\mathrm{m}$ and a manipulator weight of $30.1\,\mathrm{kg}$. The reason for this decision is that an unreachable pose is felt by the user under all circumstances. In contrast to that, velocities and accelerations that are not fully covered will reduce the achievable haptic transparency only when the excitation is high enough.

Figure 3.6 visualizes the individual coverage terms for the selected design. The pose coverage in Figure 3.6a shows that all poses in front of the user are reachable. Only hand poses that are located above the chest or below the navel are less accessible due to the choice of the height-adjusting link



(a) Pose coverage J_x.

(b) Velocity coverage J_v.

(c) Acceleration coverage J_a.

(d) Force-torque coverage J_w.

Figure 3.6: Spatial visualization of the individual coverage values. Zero means no coverage, one means perfect coverage. Transparent cells indicate that the human arm cannot reach the corresponding position. The human operator is placed identically to Figure 3.4, but not drawn here for clarity. Taken from [O4].

with length $\underline{l}_{[4]}$. The velocity coverage in Figure 3.6b exhibits similar characteristics. Here, the reduced coverage for very high and low poses is due to the reduced manipulability that occurs when joint 4 is close to its singular configuration, i.e., $\underline{q}^{\mathrm{m}}_{[4]} = 0°$ or $\underline{q}^{\mathrm{m}}_{[4]} = 180°$. In addition, there is a degradation for poses close to the body due to the singularity that appears when $\underline{q}^{\mathrm{m}}_{[2]}$ approaches $180°$. Compared to the pose and velocity coverage, the acceleration coverage in Figure 3.6c shows inverted behavior. This is caused by the gravity compensating torque for joint 4, which is $21\,\mathrm{Nm}$ when $\underline{q}^{\mathrm{m}}_{[4]} = 90°$ and therefore consumes a significant amount of the available joint torque. For angles far away from $90°$, the effect of gravity is reduced, which results in a higher acceleration coverage. The force-torque coverage in Figure 3.6d indicates that the selected design will be able to produce the specified forces and torques for all human hand poses that can be reached by the manipulator.

3.3.5 Validation

The optimized and selected manipulator design can be validated using real motion data from ADL and an interactive demonstrator.

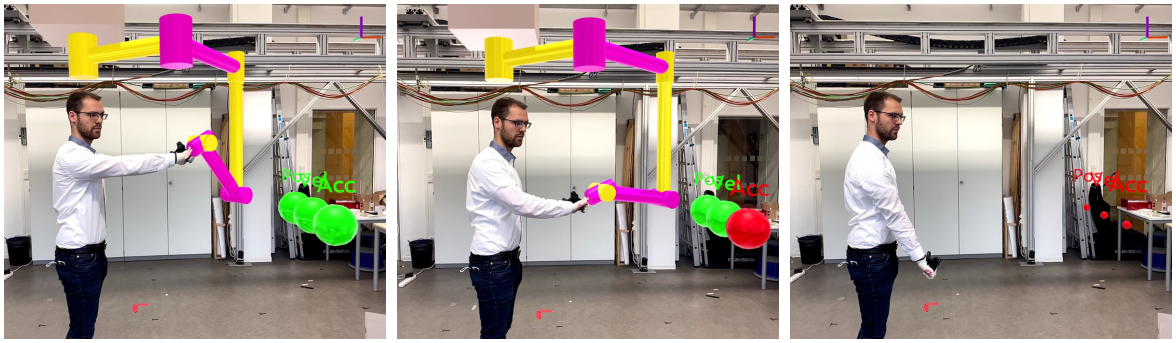
Design Performance in Activities of Daily Living

The model of the human arm in Section 3.3.2 is entirely derived from the literature and limited to the average parameters of the human arm. For this reason, checking the compatibility of the optimized design with real-world data is a meaningful validation approach. To realize this, the dataset in [79] is used. It contains nine hours of skeleton tracking data collected from five males and eight females, aged between 19 and 42 years, while performing ADL such as cooking, doing the laundry, and cleaning. In the first step, the pose of the subject's right hand relative to the right shoulder is extracted from the data to isolate the movement of the human arm from other body movements. Using an acausal, delay-free version of a fourth-order Butterworth filter with a cutoff frequency of $5\,\mathrm{Hz}$, the hand pose information is smoothed before being numerically differentiated to obtain Cartesian velocity and acceleration estimates. If possible, this information is then converted into matching manipulator joint angles, velocities, and torques using the kinematic and dynamic models from above. Due to the lack of subject height information and ground plane data in the dataset, the shoulder frame G is placed according to $\underline{x}_{PG}^{P}$ in this step, as specified in Section 3.3.2. The resulting joint velocities and torques are then compared to the motor ratings to determine whether the given hand velocity and acceleration are covered by the manipulator design.

In the analyzed dataset, a suitable manipulator configuration was found for $94.0\,\%$ of the hand poses across all subjects and activities. Unreachable poses were mainly caused by the previously discussed limited height adjustment capabilities. With the selected design, $93.7\,\%$ of the estimated hand velocity samples and $90.1\,\%$ of the estimated hand acceleration samples were fully reachable. These numbers also include samples for which no inverse kinematics solution was found, meaning that the velocity and acceleration coverage cannot exceed the $94.0\,\%$ pose coverage. The standard deviations for these pose, velocity, and acceleration metrics are $4.15\,\%$, $4.15\,\%$, and $4.41\,\%$ across all subjects, respectively.

Interactive Demonstrator

An interactive virtual demonstrator was built to facilitate online testing and validation of the manipulator topology and the optimized parameters. For this purpose, the subject's hand pose is captured using a *HTC Vive* tracker [D4] while standing at a fixed position. The manipulator base frame P is located at a height of $2.04\,\mathrm{m}$, matching the model from Section 3.3.2, and aligned with the horizontal position of the user's shoulder. Based on the pose data, the hand velocity and acceleration are estimated using numerical differentiation and suitable low-pass filters. Similar to the previous section, the hand state is then converted into joint angles, velocities, and torques and compared to the specified maximum joint velocities and torques. From this data, a binary flag for the momentary pose coverage and two scalar values ranging from zero to one for the momentary velocity and acceleration coverage are calculated. This information is then presented to the experimenter in real time using an AR/VR display. Figure 3.7 shows an example of the demonstrator in action. During the experiments, no deviations from the results presented in the previous section were found. Furthermore, the understanding of the kinematic and dynamic properties of the manipulator was deepened.



(a) Full coverage. **(b)** Reduced acceleration coverage. **(c)** No coverage.

Figure 3.7: AR view of the interactive virtual demonstrator. The size of the three spheres represents the momentary pose, velocity, and acceleration coverage, from back to front.

3.4 Discussion

In this chapter, the high-level requirements of HapticGiant were listed and used to develop a system concept. Based on a literature review and the lessons learned from previous designs, a new kinematic topology, consisting of a two-axis linear PPU and a six-axis manipulator, was created. The manipulator has many design parameters that must be carefully selected. For this reason, a novel optimization-driven dimensioning approach was developed that is based on a detailed kinematic model of the human arm in order to maximize the manipulator's coverage of the human arm in terms of workspace, velocities, and accelerations. Sufficient force-torque capabilities are also ensured. The optimization procedure yields several design candidates, from which the one with the highest pose coverage was selected based on the argument of haptic transparency.

As presented in Section 3.3.4, the coverage of the selected optimal design regarding poses and velocities can be considered very good, while the acceleration coverage has some potential for improvement. However, this trade-off was a deliberate decision, as pose coverage is more important than perfect transparency. In Section 3.3.5, the results of the optimization were validated using real data from ADL, confirming that the human arm analysis from Section 3.3.2 is a valid generalization of the capabilities of the human arm. Furthermore, the low inter-subject variance of the coverage values indicates that the design is robust to variations in the user's physique. Interactive experiments with a virtual demonstrator confirmed the expected coverages and the overall feasibility of the manipulator design.

Despite the logical and numerical justification, the presented concept and the methodology behind it are not without limitations. The human arm model used throughout the optimization is derived from highly processed data. To obtain a better representation of the human arm capabilities, the focus could be shifted to the incorporation of raw motion data. With the presented sample-based representation of the human workspace, this should be possible. Furthermore, the current manipulator model is limited because effects such as rotor inertia, joint losses, and material stiffness are not considered in the optimization. Another limitation that may appear, when the design method is applied to other problems, is that the presented optimization algorithm can handle only non-redundant, serial manipulators.

Nevertheless, the results of this chapter seem promising enough to build a real prototype of HapticGiant. This process, with aspects going far beyond the hardware design, will be covered in the remainder of this thesis.

CHAPTER 4

Realization of HapticGiant

Contents

> Scientists study the world as it is,
> engineers create the world that has never been.
>
> THEODORE VON KÁRMÁN (1881 – 1963)

In the previous section, a concept for a manipulator was presented, but without manufacturing information it will remain purely theoretical. To enable large-scale kinesthetic haptic feedback, the manipulator must be able to move within its intended workspace using a gantry crane-like PPU, which was not yet covered. Furthermore, sensing capabilities and a control framework must be added to obtain a functional system. To fill these gaps, this chapter provides an overview of the hardware and software components that were specifically designed during the realization of HapticGiant.

Having the requirements from Section 3.1 in mind, the system architecture in Figure 4.1 was designed. The hardware interface takes care of the communication with the sensors and actuators in the physical plant. The sensor information is then processed in the state estimation to obtain the kinematic state of the system (Chapter 6) and the user wrench at the end effector (Section 4.3.1). This information is then used by the feedback control system, which is split into a PPU controller (Section 4.3.2) and

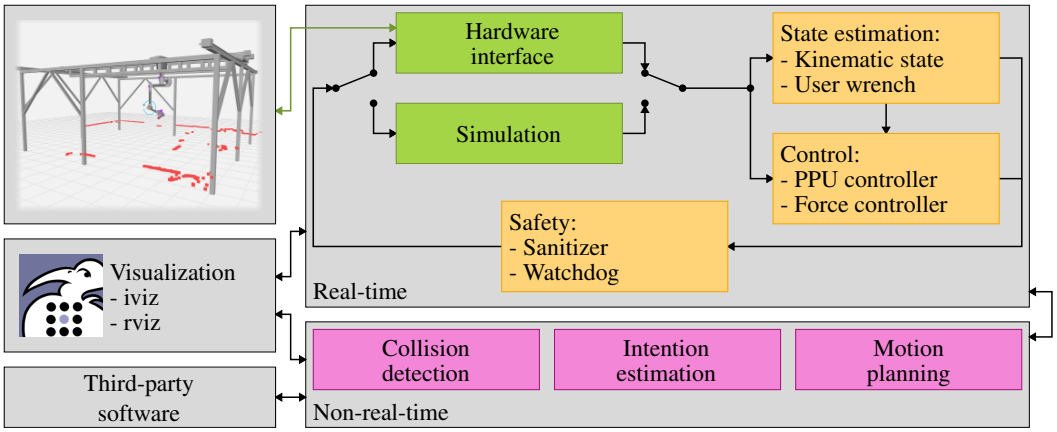


Figure 4.1: Overview of HapticGiant's system architecture. Hardware-related components are drawn in green, real-time software components in orange, and non-real-time software components in magenta. Adapted from [O1].

a force controller (Chapter 7). The resulting setpoints are forwarded to the safety-related components (Section 4.1.3), including a sanitizer and a watchdog module, before they are commanded to the actuators. For testing purposes, the hardware interface can be swapped with a simulation (Chapter 5). The components up to this point are required to be real-time capable. However, there are several non-real-time components, such as a module for collision detection (Section 4.1.3), an intention estimation algorithm (Section 8.1), and a motion planning system (Section 8.2). To communicate with visualization software (Section 4.3.3) and third-party applications, appropriate interfaces are defined.

In the following, we start with the description of the hardware platform as a central point of HapticGiant in Section 4.1. Because of the inevitable human-machine interaction, the safety concept is treated separately in Section 4.1.3. Then, the custom Real-Time Control Framework (RTCF) is introduced in Section 4.2 as the backbone to implement the presented architecture. Some of the software components, such as the force-torque estimation and the simulation, are explained in the remainder of this chapter. The other components are covered in subsequent chapters as their scope is too large for a comprehensive treatment in this chapter.

This chapter is based on results presented in the author's publications [O1, O6].

4.1 Hardware

In the following, some key aspects of the hardware including sensors, actuators, the construction, the safety concept, and the solution of the inverse kinematics problem are explained. This section is not meant as detailed technical documentation, as this would exceed the scope of this thesis. Instead, the focus is on the challenges and design features that make HapticGiant a unique and versatile piece of hardware.

4.1.1 Instrumentation and Actuation

The linear PPU with its two DOF is belt-driven by a total of four *SEW Eurodrive* permanent-magnet synchronous motors (PMSMs) with integrated mechanical brakes, powered by matching *Movidrive MDX61B* [D5] inverters. This means that both axes are driven redundantly. The manipulator is driven by six *Sensojoint Sensodrive* [D3] SEAs, with two *7005*, two *5005*, and two *3008* models connected

in series. As with the PPU motors, the manipulator joints come with integrated, safety-certified mechanical brakes. The specific joint models are taken directly from the optimized manipulator design in Section 3.3.4, except that the transmission ratio of the last two joints was changed from 1:51 to 1:81 due to limited part availability.

The overall system is equipped with a variety of sensors. The support frame of the PPU carries a safety-certified *Sick TiM781S* [D6] 2D laserscanner monitoring the floor. This facilitates tracking the user position as well as preventing unintended intrusions of the security zone. All four PPU motors have integrated resolvers enabling position measurements. Each PPU axis is equipped with two pairs of limit switches: the inner pair is responsible for normal operation, and the outer pair detects abnormal PPU positions. As the PPU axes are driven redundantly without a synchronization shaft, the synchronization of the driving motors must be ensured separately. Therefore, position measurements are obtained with two *Elgo EMAX* [D7] absolute encoders per axis. Each manipulator joint is equipped with a load-side and a motor-side absolute encoder. Furthermore, the joint torque can be determined using a motor current sensor and a load-side torque sensor. The user forces and torques at the end effector are captured using a *Botasys SensONE* [D8] six-axis force-torque sensor. Dead man switches in the end effector handle and the remote of the safety operator complete the user input. Beyond that, HapticGiant is equipped with several inertial measurement units (IMUs) for the kinematic state estimation. A *Beckhoff EP3752-0000* [D9] with two redundant three-axis accelerometers is attached to the trolley of the PPU. Next, a *Gable Systems SE1-HSD* [D10] with a three-axis gyroscope and accelerometer is mounted at manipulator joint 5. The last IMU, an *Invensense MPU-9250* [D11] is integrated into the force-torque sensor. Further details about the IMU setup can be found in Section 6.2.

The *EtherCAT* [80] technology is used as a low-latency and high-bandwidth fieldbus system to connect all sensors and actuators, which are not safety-critical, with reduced cabling effort. This, in combination with the consistent use of off-the-shelf components, drastically reduces the need for custom embedded device development, which was a major drawback of the previous system. Furthermore, proven industrial components and standards improve the reliability and extensibility of the overall system. All PPU and manipulator joints have at least one absolute encoder, enabling an immediate system initialization without the need for referencing.

4.1.2 Construction

As the manipulator of HapticGiant's predecessors [22, 33] has only two active DOF, it was quickly decided to discard the old manipulator. This decision was also backed by severe issues with functional safety in combination with sensor and interface limitations. For this reason, only the PPU with its frame-like supporting structure and its drivetrain as seen in Figure 4.2a were kept for HapticGiant. Based on the results from Chapter 3, a whole new manipulator with the Denavit-Hartenberg (DH) parameters from Table 4.1 was constructed and manufactured in the context of [S1]. To achieve this, the general requirements from Section 3.1 were reformulated into a set of more manipulator-specific goals and requirements:

- The resulting manipulator shall provide maximal haptic transparency and dexterity for a potentially walking human user. This means that the kinematic design from Section 3.3.4 needs to be respected and that the first manipulator joint needs an infinite range of motion. The desired user interface is a handle-like structure. Additionally, users with a height of up to $1.95\,\mathrm{m}$ shall be able to interact with the system, which means that appropriate clearance is required.
- The stiffness of the manipulator has to be high enough to display forces and torques within the desired range. At the same time, the manipulator shall produce high absolute velocities and accelerations, which demands a lightweight design. For reproducibility, the design must be free of backlash and unambiguous assembly must be guaranteed, e.g., by means of fits.

Joint	θ	d	a	α
PPU 1	0	$\underline{q}^{\mathrm{p}}_{[1]}$	0	$90°$
PPU 2	$90°$	$\underline{q}^{\mathrm{p}}_{[2]}$	0	$90°$
Manipulator 1	$\underline{q}^{\mathrm{m}}_{[1]}$	0	$0.53\,\mathrm{m}$	0
Manipulator 2	$\underline{q}^{\mathrm{m}}_{[2]}$	0	$0.67\,\mathrm{m}$	0
Manipulator 3	$\underline{q}^{\mathrm{m}}_{[3]} + 180°$	$-0.75\,\mathrm{m}$	0	$90°$
Manipulator 4	$\underline{q}^{\mathrm{m}}_{[4]} - 90°$	0	$0.42\,\mathrm{m}$	0
Manipulator 5	$\underline{q}^{\mathrm{m}}_{[5]} + 90°$	0	0	$90°$
Manipulator 6	$\underline{q}^{\mathrm{m}}_{[6]}$	$0.12\,\mathrm{m}$	0	0

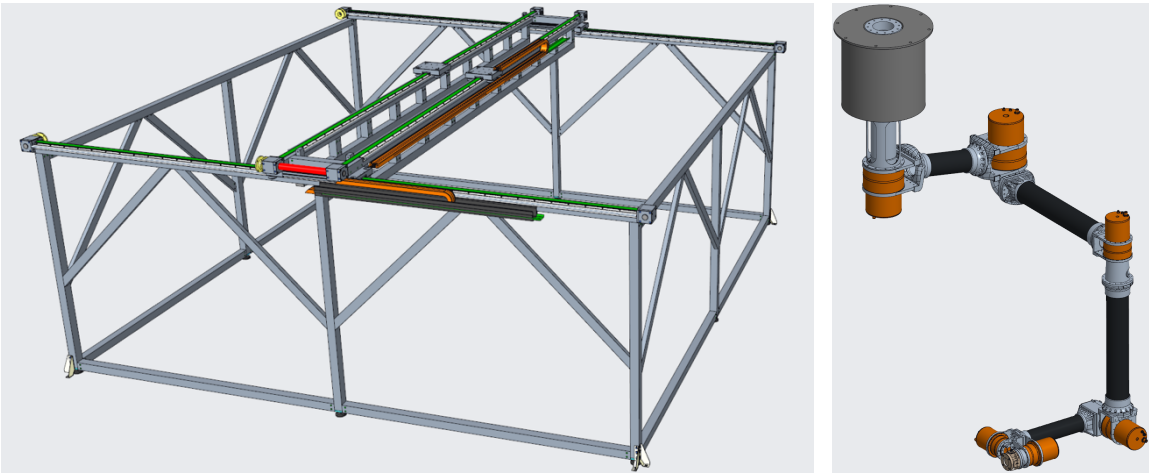
Table 4.1: Full DH parameters of HapticGiant. Taken from [O1].

- The manipulator must be well-integrated with other subsystems, including the above-mentioned sensors and actuators, the existing PPU, and the wiring.
- Even though HapticGiant is meant as a prototype, the manipulator design should use some recurring building blocks to provide some degree of modularity. Moreover, mechanical and electrical interfaces shall be included to integrate additional components in the future.
- The design must be manufacturable with common tools and materials.
- Lastly, user safety is a concern and must be regarded in the design, for instance by minimizing pinch points or sharp edges.

These requirements are complex, partially contradict each other as previously discussed in Section 3.3, and do not come with a clear solution. To still derive a suitable solution, the method of System Generation Engineering (SGE) [81] was applied. SGE is a method in which the development of new systems and their subsystems is based on a so-called reference system by applying three different types of variations. This reference system is created from subsystems of existing systems already covering some of the desired functionality. In this case, the reference system is composed of the light-weight manipulators at the *German Center for Aerospace* [82, 83], the humanoid robot *ARMAR* [84], the collaborative robots from *Universal Robots* [D2], the *AnyExo* exoskeleton [48], and the carbon fiber reinforced mountain bike frames from *Atherton bikes* [85].

Based on the reference system, different solution concepts for subsystems were investigated. For HapticGiant, the connection between the actuators is particularly critical, as high stiffness and low mass are required, while the mechanical interface of the joints is given. After eliminating poorly realizable solutions, two concepts for the link elements were left. The first solution consists of aluminum-links in tube-shaped or lattice-like structures. The second solution uses carbon fiber reinforced tubes, whose ends are glued into an outer and inner aluminum sleeve. As no experience with gluing carbon fiber and aluminum was available at the author's lab, an experimental setup was built to successfully verify the feasibility and durability of the second solution. The integration of a through-bore slip ring between the PPU and joint 1 was explored with the same methodology. In the chosen concept, the rotor of joint 1 is rigidly connected to the PPU, while the rotor of the slip ring is attached to the housing of the joint. Following the remaining requirements, a *B-Command rotarX* [D12] slip ring with an inner diameter of $140\,\mathrm{mm}$ and signals for EtherCAT, Gigabit Ethernet, power supply, and discrete signals was selected.

In the last step, detailed components were designed using computer-aided design (CAD) tools. To meet the clearance requirements, the link length between joints 3 and 4 was slightly adjusted. As a result, manipulator joints 1 through 3 were moved vertically, but without changing the size and shape of the manipulator workspace as experienced by the user. Even though the joints are placed in various sizes and angles relative to each other, the requirement for modularity was met by creating standardized



(a) The PPU with the supporting frame structure. The manipulator is attached to the trolley of the gantry crane-like structure.

(b) The manipulator with six joints, slip ring and force-torque sensor.

Figure 4.2: Representation of HapticGiant in CAD.

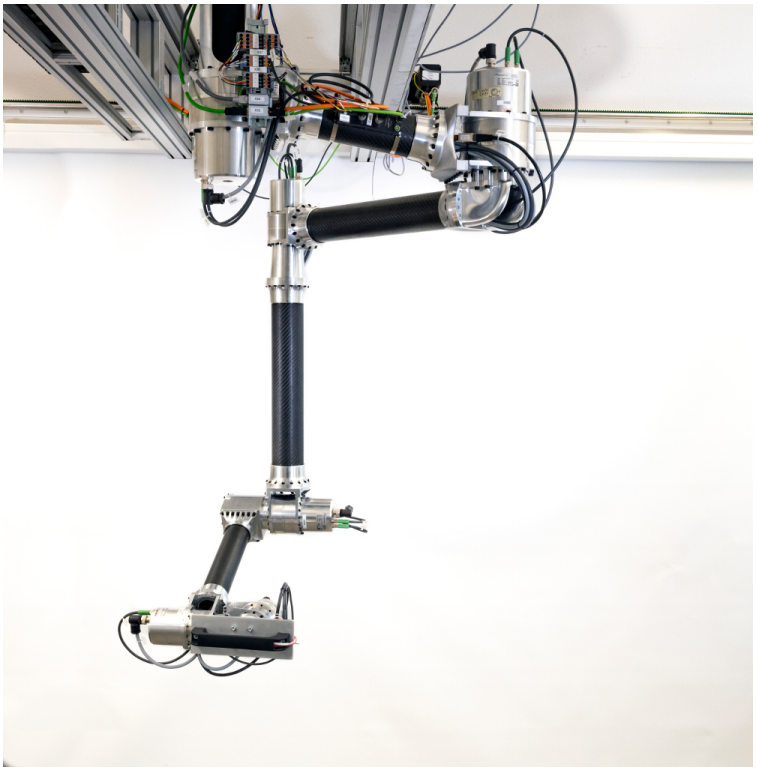
pairs of fittings holding the carbon fiber reinforced link tubes. Topological optimization was applied to improve the stiffness-to-weight ratio of the connecting pieces at joints 1 and 2.

The final manipulator design can be seen in Figure 4.2b. It has a moving weight of $36.8\,\mathrm{kg}$, excluding the rotor of the slip ring, and a total link length of $2.5\,\mathrm{m}$ between the first actuator and the end effector. To complete the construction of HapticGiant, a new switching cabinet was planned and installed. It contains the inverters, a safety programmable logic controller (PLC), and other electrical components. To avoid overvoltage during dynamic braking of the manipulator, two *Nanotec BC72-50* brake choppers [D13] are installed on the trolley of the PPU. Some impressions of the real system can be found in Figure 4.3.

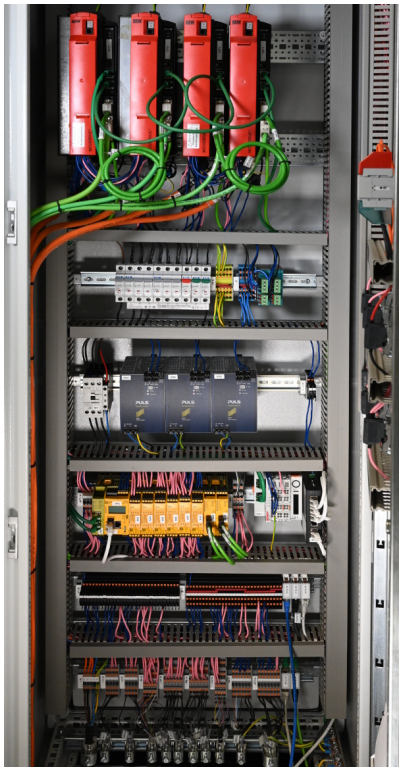
4.1.3 Safety Concept

Human-machine interaction is inherently required for the generation of artificial kinesthetic haptic feedback and should therefore be considered in the safety concept of kinesthetic haptic interfaces. Nevertheless, many interfaces, including those from Section 1.2.1, do not pay much attention to safety, as the physical workspace and the output capabilities of the devices regarding force, torque, and velocity are rather limited. In contrast to that, the user is moving *within* the workspace of HapticGiant and the output capabilities of the kinematic chain, in particular the PPU, are big enough to seriously endanger humans. In combination with the safety concept of the previous system, which is proven to be fault-intolerant, this leads to the necessity of a comprehensive safety concept for HapticGiant. To achieve this, the four-layer concept from Figure 4.4, which is described in the following, was developed. In general, the goal of this concept is not to create an error-free system, but rather an accident-free system. This means the safe stop of the system must be guaranteed under all circumstances.

In the lowest layer, a fully redundant *Pilz PNOZ m B1* safety PLC [D14] is connected via two-channel signaling to all safety-relevant sensors, including the laser scanner as well as two dead man's switches for the user and the safety operator. The mechanical brakes of the actuators are either redundant in the case of the PPU, or safety-certified with a redundant *safe torque off* signal [D3] in the case of the manipulator. The brakes of the Sensojoints are not rated as operating brakes. For this reason, the manipulator is braked cooperatively by commanding suitable setpoints. However, if this fails, the



(a) The manipulator with its six joints and carbon fiber reinforced links. One of the topologically optimized parts can be observed at the beginning of the link connecting joints 2 and 3.



(b) The switching cabinet with PPU inverters (red), safety PLC (yellow), DC power supply (dark blue), and various other components.

Figure 4.3: Impressions of the assembled and operational hardware.

mechanical brakes are still triggered through the safety PLC. By default, the actuators are enabled only when both dead man's switches are pressed. However, this can be bypassed in two stages. In the first stage, the dead man's switch of the user is overridden, but then the laser scanner must signal a clear workspace. In the second stage, the laser scanner is also overridden. In this case, the risk for the user is increased, which is signaled by a flashing red light and a delay before the system is engaged. To avoid the desynchronization of the redundant PPU motors during repeated mechanical braking, stopping the PPU is realized using the *safe stop 2* functionality [86] on *PNOZ m EF 2MM* motion monitoring modules [D14]. As part of this, the hold position is maintained by an inverter-integrated position controller and deviations from the intended behavior are detected by the safety PLC. Additionally, the safety PLC monitors the emergency stop buttons and the limit switches.

The next layer is located in the EtherCAT slaves, where communication watchdogs ensure that the periodic EtherCAT data exchange is alive. If this is not the case, the slaves fall back to a safe state. Additionally, EtherCAT process data object (PDO) watchdogs are implemented to ensure that the internal communication in the slave devices works as expected.

In the software layer, the safety mechanisms from Figure 4.1 are implemented. First, the sensor data is checked in a watchdog for potential violations of hardware specifications, such as joint limits, velocity limits, and the synchronization of the redundant PPU drives. Second, the hardware setpoints from the controllers are sanitized. For example, a positive torque setpoint for a joint already being at its upper limit will be zeroed. The third dedicated software component for safety is a predictive collision detection algorithm. Using *Flexible Collision Library (FCL)* [87], a simplified model of the

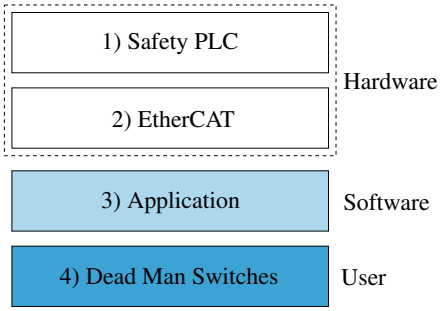


Figure 4.4: HapticGiant's layered safety concept.

manipulator geometry is deployed for this purpose. The expected configuration q_{pred} is derived from the current joint state with configuration q and velocity $\dot{q}$ by calculating the stopping position after a braking maneuver with maximal constant deceleration $\ddot{q}_{\max}$. Solving the corresponding equations of motion yields

$$q_{\text{pred}} = q + \operatorname{sign}(\dot{q}) \frac{\dot{q}^2}{2\,\ddot{q}_{\max}} .$$

Furthermore, all other software modules not dedicated to safety, such as the state estimator and the force controller, can trigger a full system stop in case they detect implausibilities in their inner workings, e.g., when limit violations or non-finite values are detected.

The final layer of safety is the user and the always-required safety operator. The dead man's switch of the latter is always *AND*ed with the actuator clearance of the system. As soon as it is released, the system is put into a safe state. All mechanical brakes are triggered immediately when any of the emergency stop buttons is pressed or a failure is detected.

4.1.4 Inverse Kinematics

The kinematic structure of HapticGiant and additional information needed for solving the inverse kinematics problem is illustrated in Figure 4.5. The corresponding DH parameters can be found in Table 4.1. Due to the kinematic structure of the manipulator, the solution of the inverse kinematics problem can be calculated in closed form:

1. Without loss of generality, we assume that the desired end effector position is given relative to the PPU position, i.e., the tuple $\left(\underline{x}^P_{PH}, \mathbf{C}^P_H\right)$ is known. Using

   $$\mathbf{C}^P_H = \mathbf{C}_z(\alpha)\mathbf{C}_y(\beta)\mathbf{C}_z(\gamma)\mathbf{C}_x(\pi)\mathbf{C}_z(\pi) ,$$

   the desired end effector orientation can be expressed using the Euler angles α, β, and γ. Here, $\mathbf{C}_a(\phi)$ denotes the matrix representation of a rotation around the coordinate axis a by ϕ. The resulting angles correspond to

   $$\begin{aligned} \alpha &= \underline{q}^{\text{m}}_{[1]} + \underline{q}^{\text{m}}_{[2]} + \underline{q}^{\text{m}}_{[3]} , \\ \beta &= \underline{q}^{\text{m}}_{[4]} + \underline{q}^{\text{m}}_{[5]} , \quad \text{and} \\ \gamma &= -\underline{q}^{\text{m}}_{[6]} , \end{aligned} \tag{4.1}$$

   from which $\underline{q}^{\text{m}}_{[6]}$ is directly determined.

2. The joint angles $\underline{q}^{\text{m}}_{[4]}$ and $\underline{q}^{\text{m}}_{[5]}$ can be calculated using the height of the vertical end effector position

   $$\underline{x}^P_{PH[3]} = -\underline{l}_{[3]} - \underline{l}_{[5]} \cos(\beta) - \underline{l}_{[4]} \cos(\underline{q}^{\text{m}}_{[4]}) .$$

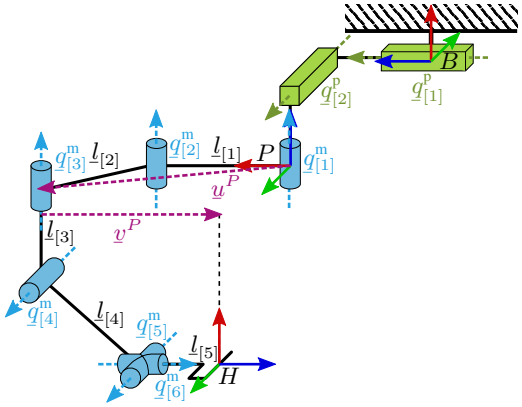


Figure 4.5: Illustration of the kinematic structure for solving the inverse kinematics problem.

Using the above definition of β, this can be solved for the joint angles

$$\underline{q}^{\mathrm{m}}_{[4]} = \arccos\left(\frac{\underline{x}^{P}_{PH[3]} + \underline{l}_{[3]} + \underline{l}_{[5]}\cos(\beta)}{-\underline{l}_{[4]}}\right) \quad \text{and}$$
$$\underline{q}^{\mathrm{m}}_{[5]} = \beta - \underline{q}^{\mathrm{m}}_{[4]}\,.$$

3. The end effector position can be interpreted as the result of chaining an ordinary planar SCARA manipulator with the remaining manipulator joints. In this case, the position displacement of the SCARA manipulator is

$$\underline{u}^{P} = \begin{pmatrix} \underline{l}_{[1]}\cos(\underline{q}^{\mathrm{m}}_{[1]}) + \underline{l}_{[2]}\cos(\underline{q}^{\mathrm{m}}_{[1]} + \underline{q}^{\mathrm{m}}_{[2]}) \\ \underline{l}_{[1]}\sin(\underline{q}^{\mathrm{m}}_{[1]}) + \underline{l}_{[2]}\sin(\underline{q}^{\mathrm{m}}_{[1]} + \underline{q}^{\mathrm{m}}_{[2]}) \\ 0 \end{pmatrix}. \tag{4.2}$$

The contribution of the remaining joints to the horizontal displacement is

$$\underline{v}^{P} = -\left(\underline{l}_{[5]}\sin(\beta) + \underline{l}_{[4]}\sin(\underline{q}^{\mathrm{m}}_{[4]})\right)\begin{pmatrix}\cos(\alpha)\\ \sin(\alpha)\\ 0\end{pmatrix},$$

yielding

$$\underline{u}^{P} = \begin{pmatrix}\underline{x}^{P}_{PH[1:2]}\\ 0\end{pmatrix} - \underline{v}^{P}\,.$$

With the standard solution [36] to (4.2), the SCARA joint angles are

$$\underline{q}^{\mathrm{m}}_{[1]} = \operatorname{arctan2}(\underline{u}^{P}_{[2]}, \underline{u}^{P}_{[1]}) - \arccos\left(\frac{d^2 + \underline{l}^2_{[1]} - \underline{l}^2_{[2]}}{2\underline{l}_{[1]}d}\right) \quad \text{and}$$
$$\underline{q}^{\mathrm{m}}_{[2]} = \pi - \arccos\left(\frac{\underline{l}^2_{[1]} + \underline{l}^2_{[2]} - d^2}{2\underline{l}_{[1]}\underline{l}_{[2]}}\right)$$

with reach

$$d = \left\|\underline{u}^{P}_{[1:2]}\right\|_2\,.$$

The last unknown manipulator joint angle is then calculated using

$$\underline{q}^{\mathrm{m}}_{[3]} = \alpha - \underline{q}^{\mathrm{m}}_{[1]} - \underline{q}^{\mathrm{m}}_{[2]}\,.$$

Note, that even though only one solution is stated here, up to seven additional solutions, originating from branches in the Euler angle conversion and the SCARA inverse kinematics solution, exist. However, these, as well as the handling of singularities, are not of practical relevance with the given joint limits and hence not detailed here.

In practice, the end effector pose relative to the base $\left(\underline{x}_{BH}^{B}, \mathbf{C}_{H}^{B}\right)$ will be given as input to the inverse kinematics calculation. For this reason, the transformation

$$\begin{aligned} \underline{x}_{PH}^{P} &= \mathbf{C}_{P}^{B\mathsf{T}} \left(\underline{x}_{BH}^{B} - \underline{x}_{BP}^{B}\right) \quad \text{and} \\ \mathbf{C}_{H}^{P} &= \mathbf{C}_{P}^{B\mathsf{T}} \mathbf{C}_{H}^{B} \end{aligned} \tag{4.3}$$

must be applied in preparation. Here, $\mathbf{C}_{P}^{B}$ is a fixed rotation offset and $\underline{x}_{BP}^{B\mathsf{T}} = \begin{pmatrix} 0 & -\underline{q}_{[2]}^{\mathrm{p}} & \underline{q}_{[1]}^{\mathrm{p}} \end{pmatrix}$ incorporates the PPU configuration $\underline{q}^{\mathrm{p}}$, which can be interpreted as a solution parameter. If arbitrary end effector poses are given, $\underline{q}^{\mathrm{p}}$ may not be known a priori. In this case, the calculations for $\underline{q}_{[4]}^{\mathrm{m}}$ to $\underline{q}_{[6]}^{\mathrm{m}}$ remain unchanged. The values of $\underline{q}_{[2]}^{\mathrm{m}}$ and $\underline{q}_{[3]}^{\mathrm{m}}$ are set to feasible values, e.g., $\pi/2$ and $-\arctan(\underline{l}_{[1]}/\underline{l}_{[2]})$, respectively, which directly yields $\underline{q}_{[1]}^{\mathrm{m}}$ according to (4.1). The now fully known manipulator configuration $\underline{q}^{\mathrm{m}}$ is then plugged into the manipulator forward kinematics, yielding $\underline{x}_{PH}^{P}$. Rearranging (4.3) for $\underline{x}_{BP}^{B\mathsf{T}}$ eventually yields $\underline{q}^{\mathrm{p}}$.

4.2 Real-Time Control Framework

The controllers of HapticGiant's predecessor were running on custom embedded devices [22], which required a lot of specific knowledge and had limited hardware resources. Even worse, standard libraries, e.g., for kinematics and linear algebra, could not be used, and the only available communication interface was a low-bandwidth serial connection. *Robot Operating System (ROS)* [88] on the other end of the spectrum is a widely used robotics framework, which is shipped with many standard libraries and useful tools to encourage clear interfaces and reusability. Nevertheless, ROS was not made for high-performance control applications with strict timing requirements or fully synchronous operation, as it was designed for general-purpose Linux computers.

Even though general-purpose computers are not primarily designed to be real-time safe, impressive real-time performance was demonstrated by projects such as *Xenomai* [89] and the *Preempt-RT* Linux kernel patch [90]. While it is comparatively easy to implement simple stand-alone applications with these tools, there are many pitfalls when using them for a complex project. The ROS community has tried to address the lack of features for real-time control with the *ros_control* package [91] and the development of ROS 2 [92], but the first is missing the necessary modularity for complex control tasks and the latter was still in early development when the design decisions for HapticGiant had to be made. In addition, the real-time features of ROS 2 are still behind the level of convenience that traditional ROS code offers: To this day, the runtime composition of real-time nodes is not supported in ROS 2 and the process configuration for real-time properties has to be set up manually [93]. Other less well-known projects [94–96] also try to bridge the gap between ROS and real-time requirements. While these perform reasonably well in their respective use cases, they suffer from limited modularity and missing compatibility with standard ROS workflows.

As a promising alternative to ROS, *Open Robot Control Software (OROCOS)* [97] provides a large number of real-time safe and modular features that can be linked with existing ROS environments using the *rtt_ros_integration* package [98]. In practice, however, OROCOS does not seem to be adopted very often despite its powerful design. The author suspects that this is caused by the rather complex concepts in combination with missing built-in standards for visualization. For this reason, this section presents *Real-Time Control Framework (RTCF)* [O6] as a novel tool to seamlessly build high-performance

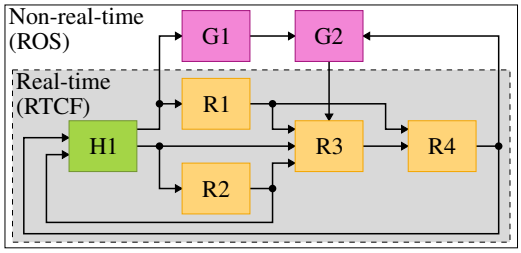


Figure 4.6: Exemplary composition of RTCF components and ROS nodes representing a typical real-world robotic system with hardware interface (green), cascaded controllers (orange), and high-level systems (magenta). Adapted from [O1].

control applications under real-time constraints. To achieve this, OROCOS is combined with ROS using rtt_ros_integration to maximize usability and convenience at the same time.

4.2.1 Design

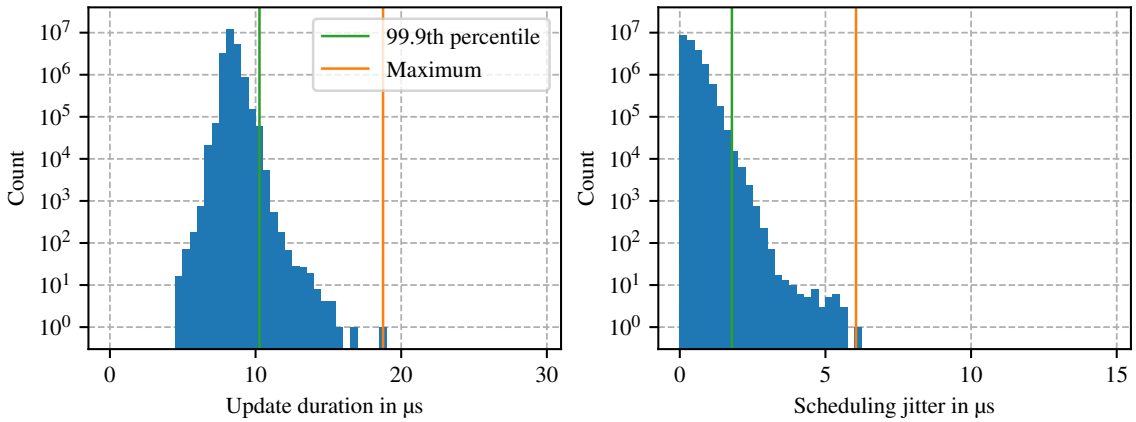
The following requirements have to be met by a suitable software framework in the context of HapticGiant, but also in many other robotic projects with real-time control:

- The framework must facilitate *modularity* through reusable components with well-defined interfaces, that can be recomposed according to the needs of the application.
- *Interoperability* with existing robotic frameworks, especially ROS, is crucial to minimize the development effort and to boost the utilization of readily available tools, e.g., for visualization, parameter handling, and logging.
- Fast, low-overhead, real-time safe execution is required to achieve satisfactory *performance*. The desired control frequencies are in the kHz-range, requiring a jitter below $10\,\mu s$. Real-time safe here means that soft real-time guarantees are required.
- The framework should be *easy to use* to enable quick results, especially for developers coming from the ROS ecosystem.

RTCF, which is written in C++, aims to fulfill these requirements by choosing suitable OROCOS functions and making them ROS-compatible. The most important building block of the framework is the so-called *component*. Its input and output interface is defined using ROS messages. Therefore, it is directly compatible with the publisher-subscriber concept used for ROS topics, making the components conceptually very similar to ROS nodes. This is also reflected in the C++ code from Listing B.2.

Like ROS nodes, a set of components can be connected to form a graph as illustrated in Figure 4.6. In contrast to ROS, where nodes are executed in concurrent processes, the components in RTCF are executed sequentially in a single thread to ensure deterministic behavior and to avoid synchronization issues. The determination of the order is done by means of an automated dependency resolution. To achieve this, each instance of an RTCF component maintains a list of predecessors and successors based on its connections to other components. With this information, Kahn's algorithm [99] is deployed, which results in the desired topological order. Possibly occurring loops, e.g., due to a hardware interface component with sensors and actuators, are avoided by excluding specific connections from Kahn's algorithm or by marking a component that shall be executed first.

To maximize interoperability, RTCF uses rtt_ros_integration to transparently bridge between the real-time capable components and the non-real-time capable nodes in both directions. Topics can be white- and blacklisted to minimize the bandwidth consumption of fast-running inner control loops according to the application's needs. To launch a complex controller architecture with components



(a) Histogram of the update duration.

(b) Histogram of the scheduling jitter.

Figure 4.7: Performance benchmark of RTCF. Taken from [O6].

specified at runtime, ROS launch files can be used in RTCF instead of the rather unfamiliar OROCOS scripting language. For this purpose two ROS nodes are used: The *rt_runner* node, whose concept is comparable to the *nodelet_manger* [100], is responsible for executing and managing all components within a single thread. The *rt_launcher* node facilitates the convenient loading and unloading of components. Listing B.1 shows an exemplary launch file. Static parameters in RTCF can be used in the same way as in ROS because each component has access to a ROS node handle. In addition, a real-time safe wrapper for *dynamic_reconfigure* [101] is provided to enable features such as online controller tuning. Real-time safe logging to the ROS console is also supported through wrapping the OROCOS logging system in ROS-like logging macros. This way, ROS tools such as *rqt_console* and *rqt_logger_level* stay fully functional and logs are automatically transported over the network if necessary. Last but not least, RTCF is compatible with third-party simulation tools by handling the *use_sim_time* parameter.

4.2.2 Evaluation

The performance of RTCF was evaluated using the architecture from Figure 4.6 running at 2 kHz. To measure the actual framework performance instead of the payload performance, all components perform a simple sum operation on their scalar inputs. The topology of the examples is chosen to resemble a complex real-world control architecture. For this reason, three topics are bridged between the real-time and the non-real-time domain using the whitelist feature. The loop closure that comes from the hardware interface H1 is resolved by marking the component to be executed first.

Timing measurements were performed on an off-the-shelf desktop computer equipped with an *Intel Core i5-8600* CPU (kernel version 5.4.44 with Preempt-RT). The system was artificially loaded using the *stress-ng* tool [102] with the options `--cpu 48 --io 48`, resulting in full processor load. The scheduling jitter, i.e., the deviation from the desired invocation time, and the update duration for each RTCF iteration were recorded for three hours with RTCF's integrated profiling tools. As depicted in Figure 4.7a, the update duration has a maximum at 18.8 µs and a 99.9th percentile at 10.3 µs. This indicates that RTCF indeed has low overhead and consistent delays, even when some topics are bridged to the non-real-time domain. According to Figure 4.7b, the maximum and the 99.9th percentile of the scheduling jitter are 6.1 µs and 1.8 µs, respectively. Hence, the combination of OROCOS and Preempt-RT works as intended. The data also suggests that control frequencies far beyond 2 kHz are possible without modifications.

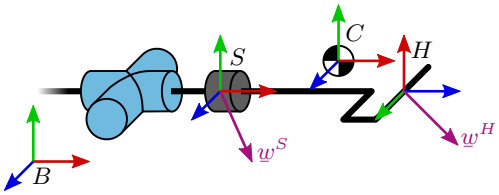


Figure 4.8: Illustration of the frames used for the end effector force-torque estimation with a force-torque sensor (gray). The handle's center of mass is located at frame C.

4.3 Selected Components

This chapter presents a few selected components that are important for the operation of HapticGiant, but whose contributions are too lightweight for individual chapters.

4.3.1 Force-Torque Estimation

The desired end effector wrench cannot be measured directly due to the size and the mechanical interface of the selected force-torque sensor. For this reason, the wrench $\underline{w}_S^{S\mathsf{T}} = (\underline{f}_S^{S\mathsf{T}}\ \underline{m}_S^{S\mathsf{T}})$ at the force-torque sensor and the kinematic state of HapticGiant are used to obtain an estimate of the actual end effector wrench $\underline{w}_H^{B\mathsf{T}} = (\underline{f}_H^{B\mathsf{T}}\ \underline{m}_H^{B\mathsf{T}})$. To achieve this, effects such as the lever arm, the inertia of the handle, and gravity must be taken into account. Furthermore, the measured force-torque values

$$\tilde{\underline{f}}_S^S = \underline{f}_S^S + \underline{b}_f^S \quad \text{and} \tag{4.4}$$

$$\tilde{\underline{m}}_S^S = \underline{m}_S^S + \underline{b}_m^S \tag{4.5}$$

will exhibit the biases $\underline{b}_f^S$ and $\underline{b}_m^S$, which must be compensated for unbiased estimation results.

In the following, the coordinate systems as depicted in Figure 4.8 are used. The measurements of the force-torque sensor are resolved in frame S. The end effector and the sensor are connected using a link with inertia matrix $\mathbf{I}^S$ in sensor coordinates and mass m, whose center of mass is located at frame C. The end effector frame H and the base frame B remain defined as before.

For the force, we start with the time-dependent center of mass position in base coordinates

$$\underline{x}_{BC}^B = \underline{x}_{BS}^B + \mathbf{C}_S^B \underline{x}_{SC}^S \,,$$

where $\underline{x}_{SC}^S$ is the constant center of mass position in sensor coordinates. Differentiating this twice and changing the resolving frame following the rules from Section 2.2 yields

$$\ddot{\underline{x}}_{BC}^S = \ddot{\underline{x}}_{BS}^S + \left(\left(\mathbf{\Omega}_{BS}^S \right)^2 + \dot{\mathbf{\Omega}}_{BS}^S \right) \underline{x}_{SC}^S \,,$$

which can be inserted into the force balance equation

$$\underline{f}_H^S = \underline{f}_S^S + m \left(\underline{g}^S - \ddot{\underline{x}}_{BC}^S \right) \tag{4.6}$$

to obtain the desired end effector force $\underline{f}_H^B = \mathbf{C}_S^B \underline{f}_H^S$ in base coordinates. Here, $\underline{g}^S = \mathbf{C}_B^S \underline{g}^B$ is the gravitational acceleration. For the torque, Euler's equation of rigid body rotation [103]

$$\mathbf{I}^S \dot{\omega}_{BS}^S + \mathbf{\Omega}_{BS}^S \mathbf{I}^S \omega_{BS}^S = m_{\text{total}}^S \tag{4.7}$$

is combined with the total accelerating torque

$$\underline{m}_{\text{total}}^S = \underline{m}_S^S - \underline{m}_H^S - \underline{x}_{SC}^S \times \underline{f}_S^S + \left(\underline{x}_{SC}^S - \underline{x}_{SH}^S \right) \times \underline{f}_H^S \,. \tag{4.8}$$

Combining (4.7) and (4.8) yields

$$\underline{m}_H^S = \underline{m}_S^S - \underline{x}_{SC}^S \times \underline{f}_S^S + \left(\underline{x}_{SC}^S - \underline{x}_{SH}^S\right) \times \underline{f}_H^S - \mathbf{I}^S \dot{\underline{\omega}}_{BS}^S - \mathbf{\Omega}_{BS}^S \mathbf{I}^S \underline{\omega}_{BS}^S \,, \tag{4.9}$$

which can be directly transformed to base coordinates using $\underline{m}_H^B = \mathbf{C}_S^B \underline{m}_H^S$. For the implementation, the kinematic quantities $\ddot{\underline{x}}_{BS}^S$, $\dot{\underline{\omega}}_{BS}^S$, $\underline{\omega}_{BS}^S$, and $\mathbf{C}_B^S$ are calculated using the forward kinematics equations and the estimated joint positions, velocities, and accelerations. If only joint positions are available or low end effector speeds and accelerations are to be expected, the velocity and acceleration quantities in (4.6) and (4.9) can be set to zero to get an approximation that compensates stationary effects.

To calibrate the biases under zero-load conditions while all movement is halted, we set the actual wrench to zero, i.e., $\underline{f}_H^S = 0$ and $\underline{m}_H^S = 0$, and sample a force-torque measurement. By inserting (4.4) and (4.5) into (4.6) and (4.9), we obtain

$$\begin{aligned} \underline{b}_f^S &= \tilde{\underline{f}}_S^S + m\underline{g}^S \quad \text{and} \\ \underline{b}_m^S &= \tilde{\underline{m}}_S^S + \underline{x}_{SC}^S \times m\underline{g}^S \end{aligned}$$

after trivial rearrangements. If these are calculated from a reasonable number of samples, the measured force-torque values can be corrected by subtracting the biases before applying (4.6) and (4.9) to obtain an unbiased end effector force-torque estimate.

4.3.2 Control of the Prepositioning Unit

As explained in Section 4.1.1, each PPU axis is driven by two redundant motors that are coupled through the mechanical structure but not through synchronization shafts. The practical experience with HapticGiant's predecessor revealed that this design can cause skewing of the traverse of the gantry-like crane, which in turn results in mechanical stress and reduced repeatability, especially after a brake-actuated emergency stop is triggered. Although this effect is less significant for the trolley, it is likely to occur there as well. Mathematically, this means that the position measurements from side A and B of each PPU axis, $\underline{q}^{\mathrm{p,A}}$ and $\underline{q}^{\mathrm{p,B}}$, start to differ significantly. While the root cause of this effect cannot be eliminated without seriously affecting the PPU construction and its supporting frame, it can be mitigated by overlaying a simple synchronization controller with the proportional gain $\underline{k}_{\mathrm{sync}}^{\mathrm{p}}$ on the commanded velocity $\dot{\underline{q}}_{\mathrm{cmd}}^{\mathrm{p}}$. Evenly distributing the controller output over the redundancy results in the individual motor commands

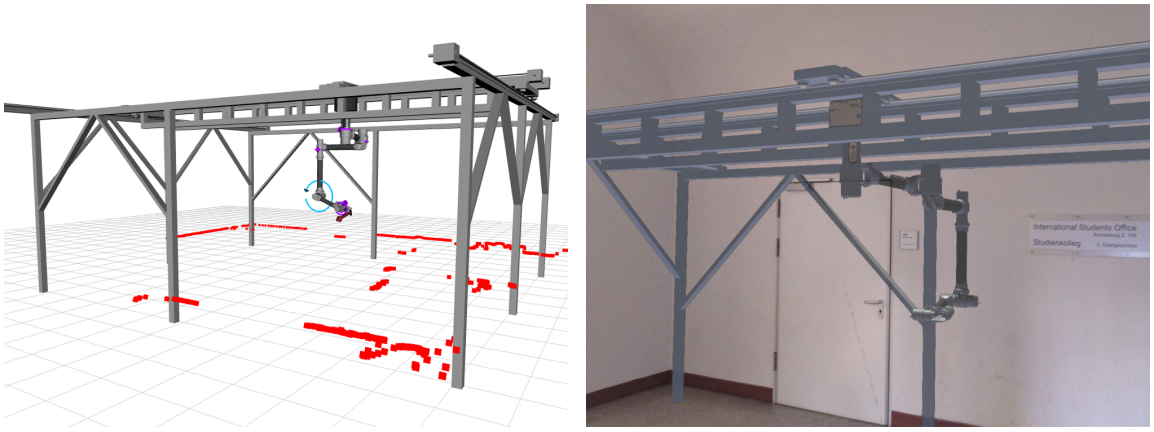
$$\begin{aligned} \dot{\underline{q}}_{\mathrm{cmd}}^{\mathrm{p,A}} &= \dot{\underline{q}}_{\mathrm{cmd}}^{\mathrm{p}} - \frac{1}{2} \operatorname{diag}(\underline{k}_{\mathrm{sync}}^{\mathrm{p}}) \left(\underline{q}^{\mathrm{p,A}} - \underline{q}^{\mathrm{p,B}}\right) \quad \text{and} \\ \dot{\underline{q}}_{\mathrm{cmd}}^{\mathrm{p,B}} &= \dot{\underline{q}}_{\mathrm{cmd}}^{\mathrm{p}} + \frac{1}{2} \operatorname{diag}(\underline{k}_{\mathrm{sync}}^{\mathrm{p}}) \left(\underline{q}^{\mathrm{p,A}} - \underline{q}^{\mathrm{p,B}}\right) \,, \end{aligned}$$

which are sent to either side of the PPU axes.

4.3.3 Visualization

The visualization of HapticGiant is essential for the simulation and the supervision of the system. Additionally, there are some situations in VR, where the user should be aware of the actual system state while interacting with HapticGiant. Visualization tools are also necessary to display the visual aspects of any TE and their DTs. Game engines, such as *Unity* [104] or *Unreal Engine* [105], provide very powerful and sometimes even photorealistic rendering capabilities; however, they require a lot of dedicated development effort.

For this reason, it was decided to build the visualization of HapticGiant upon the ROS tool *rviz* [88]. As depicted in Figure 4.9a, rviz can visualize 3D information, such as point cloud data, joint torques,



(a) In rviz on a Linux computer.

(b) In iviz on a Microsoft Hololens 2.

Figure 4.9: Visualization of HapticGiant.

wrenches, and the kinematic robot state. For the latter, the Unified Robot Description Format (URDF) file format [106] is used to specify a robot model, whose information was derived from CAD data. The visualization on head-mounted displays and mobile devices is achieved with *iviz* [107], which can be considered as a Unity-based substitute for rviz. This way, HapticGiant can be visualized on a variety of AR/VR platforms, such as *HTC Vive*, *Apple iPad*, or *Microsoft Hololens*. An example of the visualization using iviz is shown in Figure 4.9b.

4.4 Discussion

Although the novelty of this chapter is limited from a purely scientific point of view, the design of HapticGiant is unique in many ways and, therefore, the platform as a whole can be considered as a novelty. In contrast to other kinesthetic haptic interfaces, HapticGiant's hardware is optimized for delivering force and torque feedback within the full range of motion of the human arm, while the user can walk freely in a room-sized environment. The increased risk resulting from the permanent whole-body human-machine interaction is reflected in a sophisticated safety concept.

With a total weight of $36.8\,\mathrm{kg}$, the manipulator design is overweight compared to the expected $30.1\,\mathrm{kg}$ from Section 3.3.4, causing a slight degradation of the actual dynamic coverage. Despite that, the presented hardware still seems to be a very good realization of the design, especially when considering that it is the first and only iteration at the time of this writing. For future prototype iterations, the primary cause of the overweight should be analyzed and incorporated into the design process.

To the best of the author's knowledge, RTCF is the first software framework enabling high-performance control on general-purpose computers with the given degree of ROS interoperability and modularity. While it will probably not become a standard tool in robotics due to the advent of stable ROS 2 versions, it is still a valuable demonstration of how seamless real-time control could look in future ROS versions.

Together with the visualization and other basic components, the presented system is a cornerstone for many of the practical research results in the following chapters. Putting the hardware and software to the test will also reveal the system's weaknesses and potential for improvement.

CHAPTER 5

Simulation

Contents

> ... in real life mistakes are likely to be irrevocable. Computer simulation, however, makes it economically practical to make mistakes on purpose.
>
> JOHN H. MCLEOD (1911 – 2005)

Simulation tools are essential for the efficient development of robotic systems as algorithms can be tested before they are deployed on the real system, reducing the risk of malfunctions and minimizing development costs. Furthermore, a suitable simulation can increase the availability of the system during maintenance or even when the system is still under construction. Another interesting aspect is that a simulation can be used to create a portable virtual prototype, which is especially appealing for HapticGiant due to its size-induced immobility.

For a successful simulation, all relevant effects must be included in the simulation model, while maintaining reasonable speed and accuracy. For HapticGiant with its eight DOF, this is reflected in the following requirements:

- The resulting simulation environment should act as a drop-in replacement for the real system. This means that especially the interfaces must be identical.
- The joints in HapticGiant's manipulator exhibit properties of series elastic actuators (SEAs) due to their mechanical design. As a result, the joint flexibility must be considered in the simulation to achieve realistic transient behavior.

- The PPU of the real system does not have a physically derived dynamic model and is commanded using velocity setpoints. For this reason, the PPU must be modeled as a velocity-controlled black box affecting the manipulator movement.
- The simulation of non-linear friction effects, especially the Coulomb friction originating from the joint transmissions, is important to enable the use of the same low-level controllers in the simulation as in the real system.
- The simulation must be able to run in combination with the rest of the real-time feedback control loop. This means that the cycle time of the simulation must be way below $1\,\mathrm{ms}$ to facilitate an update frequency of $1\,\mathrm{kHz}$ for the overall system, including controllers.
- A virtual prototype cannot directly create user interaction forces and torques. Hence, a suitable input alternative must be provided.
- Collision detection between body parts is not mandatory, as HapticGiant is not supposed to collide with anything during normal operation.

Several tools are available for the simulation of robotic systems. In academia, the combination of *Matlab Simulink* [108] with *Matlab Simscape Multiody* [109] and *Matlab Robotics System Toolbox* [75] is a popular closed-source choice. While these tools are very good for quickly creating standalone simulation models, the model formulation is tedious for large systems, and the resulting simulations are not directly compatible with ROS. Furthermore, experiments in the context of [S2] revealed that the resulting simulation is far from being real-time capable. Robots with ties to the ROS ecosystem are often simulated using the open-source tool *Gazebo* [110], which features excellent integration with ROS but no real-time capabilities. Gazebo uses Cartesian states to represent the kinematic state of robot links, which is proven to lead to stability issues in practice [111]. Moreover, Gazebo does not support flexible joints or hybrid dynamics [40] as needed to fulfill the above requirements. With its transition to an open-source license, *MuJoCo* [112] has recently gained traction in the robotics community. Nevertheless, it is not capable of simulating hybrid dynamics either.

In general, none of the mentioned simulators is capable of meeting all the requirements for HapticGiant's simulation. Aside from that, the simulation of HapticGiant requires only a small subset of the features available in full-blown simulation suites. For this reason, it was decided to build a custom simulation tool tailored to the above requirements. In the following, Section 5.1 will first introduce the system dynamics, followed by information about the sensor simulation in Section 5.2. The parameterization and the modeling of the user behavior are briefly presented in Section 5.3 and Section 5.4, respectively, before the simulation is evaluated based on examples in Section 5.5. The discussion in Section 5.6 concludes this chapter.

5.1 System Dynamics

In the following, the dynamic model of flexible joints is explained. The simulation is then extended with the inclusion of the PPU and Coulomb friction effects.

5.1.1 Flexible Joints

The joints in HapticGiant's manipulator are SEAs. Following [113] and as depicted in Figure 5.1, these can be modeled using a motor with the rotor angle θ_i^{rot} and rotor inertia I_i^{rot}. A transmission with gear ratio $r_i > 1$ reduces the rotor angle to the motor-side angle

$$\underline{\theta}_{[i]} = \frac{1}{r_i}\theta_i^{\mathrm{rot}}$$

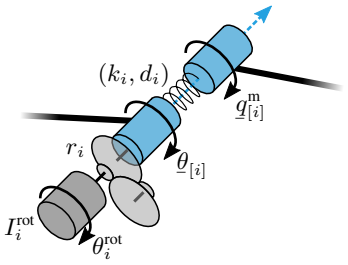


Figure 5.1: Model of a joint with an SEA. Adapted from [S3].

with the effective motor inertia

$$I_i = r_i^2 I_i^{\mathrm{rot}} .$$

The corresponding motor-side torque after the transmission is denoted as $\underline{\tau}_{[i]}$ and considered as input variable. The load-side angle $\underline{q}^{\mathrm{m}}_{[i]}$ is coupled through a spring with stiffness k_i and damping d_i. For a kinematic chain with multiple joints, the coupling torque $\underline{\tau}_{\mathrm{cpl}}$ is defined as

$$\underline{\tau}_{\mathrm{cpl}} = \mathbf{D}(\dot{\underline{\theta}} - \dot{\underline{q}}^{\mathrm{m}}) + \mathbf{K}(\underline{\theta} - \underline{q}^{\mathrm{m}}) ,$$

where $\mathbf{D}$ and $\mathbf{K}$ are diagonal matrices with the damping d_i and stiffness k_i of each joint, respectively. Under the assumption that $\underline{\theta}_{[i]} \ll \theta_i^{\mathrm{rot}}$, the motor- and load-side angles obey

$$\begin{aligned} \mathbf{B}\ddot{\underline{\theta}} &= \underline{\tau}^{\theta}_{\mathrm{dri}} \quad \text{and} \\ \mathbf{M}^{\mathrm{m}}\ddot{\underline{q}}^{\mathrm{m}} + \underline{c}^{\mathrm{m}} &= \underline{\tau}^{\mathrm{m}}_{\mathrm{dri}} \end{aligned} \tag{5.1}$$

with the driving torques

$$\begin{aligned} \underline{\tau}^{\theta}_{\mathrm{dri}} &= \underline{\tau} - \underline{\tau}_{\mathrm{cpl}} - \mathbf{D}^{\theta}\dot{\underline{\theta}} \quad \text{and} \\ \underline{\tau}^{\mathrm{m}}_{\mathrm{dri}} &= \underline{\tau}_{\mathrm{cpl}} - \mathbf{D}^{\mathrm{m}}\dot{\underline{q}} - \boldsymbol{\mathcal{J}}^{\mathrm{mT}}_{\underline{\nu}^P_H}\underline{w}^P_H . \end{aligned}$$

Here, the diagonal matrices $\mathbf{D}^{\theta}$ and $\mathbf{D}^{\mathrm{m}}$ contain the motor- and load-side viscous friction coefficients, respectively. The diagonal matrix $\mathbf{B}$ holds the effective motor inertias I_i. The remaining quantities are as in Section 2.3.

5.1.2 Hybrid Dynamics

The PPU determines the position of the manipulator base by tracking velocity setpoint $\dot{\underline{q}}^{\mathrm{p}}_{\mathrm{cmd}}$. Due to the belt-driven design of the PPU and the inverter-integrated velocity control loops, no physical model can be derived. However, the tracking behavior of the velocity setpoint can be approximated by a linear time-invariant system with the transfer function

$$\frac{\mathcal{L}\{\dot{\underline{q}}^{\mathrm{p}}_{[i]}\}(s)}{\mathcal{L}\{\dot{\underline{q}}^{\mathrm{p}}_{\mathrm{cmd}[i]}\}(s)} = \frac{\underline{a}_{0[i]}}{\underline{b}_{0[i]} + \underline{b}_{1[i]}s + \cdots + \underline{b}_{p[i]}s^p} ,$$

where $i \in \{1, 2\}$. The system order p and the coefficient vectors $\underline{a}_0, \underline{b}_0, \ldots, \underline{b}_p$ can be identified using the real system. The corresponding differential equation is

$$\frac{\partial^{p+1}\underline{q}^{\mathrm{p}}}{\partial t^{p+1}} = \left(\mathrm{diag}\,(\underline{b}_p)\right)^{-1} \left(\mathrm{diag}\,(\underline{a}_0)\,\dot{\underline{q}}^{\mathrm{p}}_{\mathrm{cmd}[i]} - \sum_{j=0}^{p-1} \mathrm{diag}\,(\underline{b}_j)\,\frac{\partial^{j+1}\underline{q}^{\mathrm{p}}}{\partial t^{j+1}} \right) . \tag{5.2}$$

This model can be combined with the concept of hybrid dynamics from [40], where the robot joints are divided into *forward-dynamics* and *inverse-dynamics joints*. For HapticGiant, the dynamic model of the PPU and the manipulator can be jointly written as

$$\begin{pmatrix} \mathbf{M}^{\mathrm{p}} & \mathbf{M}^{\mathrm{pm}} \\ \mathbf{M}^{\mathrm{mp}} & \mathbf{M}^{\mathrm{m}} \end{pmatrix} \begin{pmatrix} \ddot{\underline{q}}^{\mathrm{p}} \\ \ddot{\underline{q}}^{\mathrm{m}} \end{pmatrix} + \begin{pmatrix} \underline{c}^{\mathrm{p}} \\ \underline{c}^{\mathrm{m}} \end{pmatrix} = \begin{pmatrix} \underline{\tau}^{\mathrm{p}} \\ \underline{\tau}^{\mathrm{m}}_{\mathrm{dri}} \end{pmatrix} . \tag{5.3}$$

The forward-dynamics joints in this case are the manipulator joints with known driving torque $\underline{\tau}^{\mathrm{m}}_{\mathrm{dri}}$ but unknown acceleration $\ddot{\underline{q}}^{\mathrm{m}}$. For the inverse-dynamics joints, the acceleration $\ddot{\underline{q}}^{\mathrm{p}}$ is known. To obtain the desired manipulator acceleration

$$\begin{pmatrix} \underline{c}'^{\mathrm{p}} \\ \underline{c}'^{\mathrm{m}} \end{pmatrix} = \mathrm{RNEA}\left(\begin{pmatrix} \underline{q}^{\mathrm{p}} \\ \underline{q}^{\mathrm{m}} \end{pmatrix}, \begin{pmatrix} \dot{\underline{q}}^{\mathrm{p}} \\ \dot{\underline{q}}^{\mathrm{m}} \end{pmatrix}, \begin{pmatrix} \ddot{\underline{q}}^{\mathrm{p}} \\ \underline{0} \end{pmatrix} \right)$$

is calculated. According to [40], this simplifies (5.3) to

$$\mathbf{M}^{\mathrm{m}} \ddot{\underline{q}}^{\mathrm{m}} + \underline{c}'^{\mathrm{m}} = \underline{\tau}^{\mathrm{m}}_{\mathrm{dri}} .$$

The manipulator acceleration becomes

$$\ddot{\underline{q}}^{\mathrm{m}} = \mathbf{M}^{\mathrm{m}-1} \left(\underline{\tau}^{\mathrm{m}}_{\mathrm{dri}} - \underline{c}'^{\mathrm{m}} \right) , \tag{5.4}$$

while the motor-side acceleration $\ddot{\underline{\theta}}$ from (5.1) remains unaffected. With the information so far, the state vector

$$\underline{\psi}^{\mathsf{T}} = \begin{pmatrix} \frac{\partial^p \underline{q}^{\mathrm{p}\mathsf{T}}}{\partial t^p} & \cdots & \dot{\underline{q}}^{\mathrm{p}\mathsf{T}} & \underline{q}^{\mathrm{p}\mathsf{T}} & \underline{q}^{\mathrm{m}\mathsf{T}} & \dot{\underline{q}}^{\mathrm{m}\mathsf{T}} & \underline{\theta}^{\mathsf{T}} & \dot{\underline{\theta}}^{\mathsf{T}} \end{pmatrix}$$

and its derivative $\dot{\underline{\psi}}$ are fully defined by combining the information from (5.1), (5.2), and (5.4). To simulate the system over time, an explicit fourth-order Runge-Kutte integrator [114] with a fixed step size Δt and, hence, constant calculation time is used.

5.1.3 Coulomb Friction

Coulomb friction is present in almost every mechanical system and significantly contributes to the dynamic behavior of HapticGiant, too. According to [115], the friction force τ_{friction}, that needs to be added to the driving force τ before integration to include this effect for a single joint, is modeled as

$$\tau_{\mathrm{friction}} = \begin{cases} -\tau, & \text{if } \dot{q} = 0 \wedge |\tau| \leq \mu_{\mathrm{c}} \\ -\mu_{\mathrm{c}} \operatorname{sign}(\tau), & \text{if } \dot{q} = 0 \wedge |\tau| > \mu_{\mathrm{c}} \\ -\mu_{\mathrm{c}} \operatorname{sign}(\dot{q}), & \text{otherwise} \end{cases}$$

with the Coulomb friction coefficient μ_{c}. Although this model is conceptually simple, it poses major problems for numerical integration schemes due to the discontinuity at $\dot{q} = 0$ and its dependence on the driving force τ. In practice, this means that a moving mass under Coulomb friction in 1D space will never come to a full stop with a fixed step integrator, as the velocity will keep oscillating around zero. This becomes even more complicated when several joints are coupled through a non-diagonal inertia matrix because the Coulomb friction forces of one joint might cause the *breakaway* of another joint with zero velocity.

In literature, a variety of solutions to this problem have been proposed [115–119]. In the context of HapticGiant, the question of how to model the Coulomb friction was also examined in [S4], which led to the conclusion that the so-called *principle of maximum dissipation (PoMD)* from [116] is a

suitable approach for including Coulomb friction effects in the simulation. The principle states that the Coulomb friction minimizes the kinetic energy

$$E_{\text{kin}} = \frac{1}{2}\left(\underline{\dot{q}} + \Delta\underline{\dot{q}}\right)^{\mathsf{T}} \mathbf{M}\left(\underline{\dot{q}} + \Delta\underline{\dot{q}}\right)$$

by producing a suitable velocity change $\Delta\underline{\dot{q}}$ within Δt. Using the generalized impulse $\underline{p} = \mathbf{M}\Delta\underline{\dot{q}}$ and the Coulomb friction coefficients $\underline{\mu}_{\text{c}}$, the resulting friction effects are encoded in the inequality constraint

$$-\Delta t\underline{\mu}_{\text{c}} \leq \underline{p} \leq +\Delta t\underline{\mu}_{\text{c}}\,.$$

This can be reformulated as minimization problem

$$\begin{aligned} \underline{p}^* = \underset{\underline{p}}{\operatorname{argmin}} \frac{1}{2}\underline{p}^{\mathsf{T}}\mathbf{M}^{-1}\underline{p} + \underline{\dot{q}}^{\mathsf{T}}\underline{p}\,, \\ \text{subject to} \quad -\Delta t\underline{\mu}_{\text{c}} \leq \underline{p} \leq +\Delta t\underline{\mu}_{\text{c}}\,, \end{aligned} \tag{5.5}$$

which can be solved using quadratic programming (QP) under real-time constraints. In the context of HapticGiant, both, the motor- and the load-side of the manipulator joints are expected to experience Coulomb friction with the coefficients $\underline{\mu}_{\text{c}}^{\theta}$ and $\underline{\mu}_{\text{c}}^{\text{m}}$, respectively. In the above problem, the variables are then

$$\Delta\underline{\dot{q}} = \begin{pmatrix} \Delta\underline{\dot{q}}^{\text{m}} \\ \Delta\underline{\dot{\theta}} \end{pmatrix} = \mathbf{M}^{-1}\underline{p}^*\,, \quad \underline{\dot{q}} = \begin{pmatrix} \underline{\dot{q}}^{\text{m}} \\ \underline{\dot{\theta}} \end{pmatrix}\,, \quad \mathbf{M} = \begin{pmatrix} \mathbf{M}^{\text{m}} & \mathbf{0} \\ \mathbf{0} & \mathbf{B} \end{pmatrix}\,, \quad \text{and} \quad \underline{\mu}_{\text{c}} = \begin{pmatrix} \underline{\mu}_{\text{c}}^{\text{m}} \\ \underline{\mu}_{\text{c}}^{\theta} \end{pmatrix}. \tag{5.6}$$

If a first-order explicit Runge-Kutta integrator was used, the velocity change $\Delta\underline{\dot{q}}$ could be applied after each iteration. However, this would lead to a drift in the position over time as the position is not corrected. As an alternative, it seems reasonable to divide the velocity change $\Delta\underline{\dot{q}}$ by Δt to obtain an acceleration change, that is applied before the integration takes place. As before, this would result in a position drift: With $\underline{\dot{q}} = \underline{0}$, the minimization problem (5.5) reaches its global minimum at $\underline{p} = \underline{0}$, which translates to $\Delta\underline{\dot{q}} = \underline{0}$, and, thus, causes no acceleration change. As a result, a driving torque smaller than the Coulomb friction would still cause non-zero effective acceleration, ultimately leading to undesired position and velocity changes.

To resolve this issue, Algorithm 5.1 was developed. The key idea of this algorithm is to predict the system velocity without Coulomb friction before calculating the velocity change according to the PoMD. Based on that, the acceleration is corrected and predictions for velocity $\underline{\dot{q}}^{+\mathsf{T}} = \begin{pmatrix} \underline{\dot{q}}^{\text{m}+\mathsf{T}} & \underline{\dot{\theta}}^{+\mathsf{T}} \end{pmatrix}$ and position $\underline{q}^{+\mathsf{T}} = \begin{pmatrix} \underline{q}^{\text{m}+\mathsf{T}} & \underline{\theta}^{+\mathsf{T}} \end{pmatrix}$ are obtained. The predicted values are subsequently used to approximate the true gradient for the Runge-Kutta method using

$$\underline{\dot{\psi}}^{\mathsf{T}} = \begin{pmatrix} \frac{\partial^{p+1}\underline{q}^{\text{p}\mathsf{T}}}{\partial t^{p+1}} & \cdots & \underline{\dot{q}}^{\text{p}\mathsf{T}} & \frac{1}{\Delta t}(\underline{q}^{\text{m}+} - \underline{q}^{\text{m}})^{\mathsf{T}} & \frac{1}{\Delta t}(\underline{\dot{q}}^{\text{m}+} - \underline{\dot{q}}^{\text{m}})^{\mathsf{T}} & \frac{1}{\Delta t}(\underline{\theta}^{+} - \underline{\theta})^{\mathsf{T}} & \frac{1}{\Delta t}(\underline{\dot{\theta}}^{+} - \underline{\dot{\theta}})^{\mathsf{T}} \end{pmatrix}\,,$$

which concludes the mathematics behind the simulation of the system dynamics.

5.2 Sensors

Based on the simulated system state $\underline{\psi}$, the sensors listed in Table 5.1 were integrated into the simulation. The sensor model of the gyroscope G_i, which is depicted in Figure 5.2, contains a constant bias $\underline{b}_{G_i,\text{const}}$, a bias depending on temperature T with coefficient $\underline{c}_{G_i}$, and the white Gaussian noise $\underline{v}_{G_i} \sim \mathcal{N}\left(\underline{0},\, \sigma_{G_i}^2\mathbf{I}\right)$ with standard deviation σ_{G_i}. The axis scaling and cross-axis sensitivity are summarized in the matrix $\mathbf{S}_{G_i}$. A simple ADC model with resolution Δ_{G_i} and saturation $\omega_{G_i,\text{max}}$

Algorithm 5.1 Application of the PoMD in the context of a Runge-Kutta integrator.

Input: Kinematic state $\underline{q} = \begin{pmatrix} \underline{q}^{\mathrm{m}} \\ \underline{\theta} \end{pmatrix}$ and $\dot{\underline{q}} = \begin{pmatrix} \dot{\underline{q}}^{\mathrm{m}} \\ \dot{\underline{\theta}} \end{pmatrix}$

Output: Current acceleration $\ddot{\underline{q}} = \begin{pmatrix} \ddot{\underline{q}}^{\mathrm{m}} \\ \ddot{\underline{\theta}} \end{pmatrix}$ and predicted state $\underline{q}^{+} = \begin{pmatrix} \underline{q}^{\mathrm{m}+} \\ \underline{\theta}^{+} \end{pmatrix}$, $\dot{\underline{q}}^{+} = \begin{pmatrix} \dot{\underline{q}}^{\mathrm{m}+} \\ \dot{\underline{\theta}}^{+} \end{pmatrix}$

1: Calculate $\ddot{\underline{q}}_{\mathrm{nc}} = \begin{pmatrix} \ddot{\underline{q}}^{\mathrm{m}}_{\mathrm{nc}} \\ \ddot{\underline{\theta}}_{\mathrm{nc}} \end{pmatrix}$ using (5.1) and (5.4) `// Use predicted velocity`
2: $\dot{\underline{q}}^{+}_{\mathrm{nc}} \leftarrow \dot{\underline{q}} + \Delta t \ddot{\underline{q}}_{\mathrm{nc}}$ `// Predict velocity as if Coulomb friction did not exist`
3: Calculate $\Delta\dot{\underline{q}}$ using (5.5) and (5.6) with $\dot{\underline{q}} = \dot{\underline{q}}^{+}_{\mathrm{nc}}$
4: $\ddot{\underline{q}} \leftarrow \ddot{\underline{q}}_{\mathrm{nc}} + \frac{1}{\Delta t}\Delta\dot{\underline{q}}$ `// Adjust acceleration`
5: $\dot{\underline{q}}^{+} \leftarrow \dot{\underline{q}}^{+}_{\mathrm{nc}} + \Delta\dot{\underline{q}}$ `// Adjust predicted velocity`
6: $\underline{q}^{+} \leftarrow \underline{q} + \Delta t \dot{\underline{q}} + \frac{1}{2}\Delta t^2 \ddot{\underline{q}}$ `// Predict position`
7: **return** $\underline{q}^{+}, \dot{\underline{q}}^{+}, \ddot{\underline{q}}$

Sensor	Effects
Accelerometer	Noise, axis scaling, cross-axis sensitivity, bias, bias drift, saturation, quantization
Gyroscope	Noise, axis scaling, cross-axis sensitivity, bias, bias drift, saturation, quantization
Encoder	Noise, quantization, 1st-order low-pass for velocity measurement
Force-torque sensor	Noise, deviations regarding true handle mass, lever arm, and center of mass
Joint strain gauge	Noise, bias
Limit switch	Hysteresis

Table 5.1: Overview of the simulated sensors and included effects.

completes the setup delivering the measured angular rate $\tilde{\underline{\omega}}^{G_i}_{BG_i}$. The accelerometer model is analog. In both cases, the respective groundtruth values are calculated using the forward kinematics and gravity information. The encoder models take into account a zero-mean white Gaussian measurement noise and quantization steps, that incorporate the transmission ratio and the encoder resolution. Optionally, a pseudo velocity measurement is provided through numerical differentiation and a first-order low-pass filter. The force-torque sensor data is generated using the model from Section 4.3.1, but with slightly deviating values for handle inertia, lever arm, and center of mass compared to the end effector wrench estimation. Zero-mean white Gaussian noise is then added to the groundtruth data to obtain the simulated measurements. For the simulation of the joint torque sensors, the corresponding elements of the spring torque

$$\underline{\tau}_{\mathrm{spring}} = \mathbf{K}(\underline{\theta} - \underline{q}^{\mathrm{m}})$$

are distorted with a constant bias and zero-mean white Gaussian noise. The limit switches are modeled with a simple position hysteresis to avoid oscillations of the sensor output.

5.3 Parameterization

The simulation depends on a high number of parameters, which were determined using measurements, datasheets, or existing models. In particular, the inertia properties of the manipulator were calculated from the CAD model by assigning the correct material properties to the individual parts. Unmodeled parts, such as screws, were included by adjusting the weight of the individual link parts to match the total weight of the components after assembly. To account for the masses of the cables, the cables were weighed and their mass was distributed along the manipulator according to the cable routes. Together with the maximum joint ratings from the datasheets, the resulting mass, center of mass, and inertia matrix for each link were then added to the URDF file introduced in Section 4.3.3. This URDF file

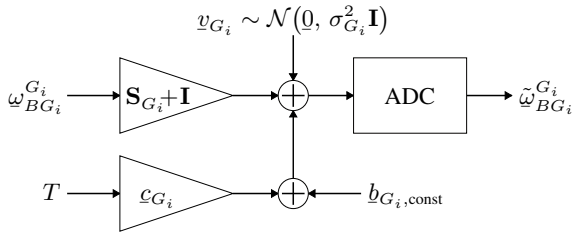


Figure 5.2: The measurement model of the gyroscope sensor at frame G_i. Taken from [O2].

also contains the forward kinematics information. Parameters such as the transmission ratio, the rotor inertia, the joint spring constant, and the joint spring damping were provided by Sensodrive. The remaining friction parameters, including Coulomb friction, were determined with the procedure from Section 7.2.2. The behavior of the PPU was identified by recording step responses and fitting a system of order $p = 2$ for each axis.

The information for the gyroscope and accelerometer sensors was mostly taken from datasheets, whereas resolution and range specifications were used directly. Values that are specified with typical values, such as axis-scaling, cross-axis sensitivity, bias, and bias temperature coefficient, were drawn from a corresponding uniform random distribution. The standard deviation of the noise was derived from datasheets and verified by measurements. The encoder resolution and its noise standard deviation were determined from datasheets and measurements, respectively. If the encoder provides a velocity estimate, the filter cutoff frequency was obtained from the configuration of the encoder. Measurements were used to identify the standard deviation of the measurement noise for each axis of the force-torque sensor. The deviations of the center of mass and the lever arm were drawn from a symmetric random uniform distribution with a maximum of $1\,\mathrm{cm}$ and $0.5\,\mathrm{cm}$, respectively. The mass error was set to $50\,\mathrm{g}$. Lastly, the noise intensity and the bias of the individual manipulator joint strain gauges were determined from the standard deviation and the mean of measured data. The remaining simulation parameters are not explained here, as their values are not critical for the fidelity of the simulation.

5.4 User Behavior

With the presented simulation, it is already possible to simulate the effect of arbitrary end effector wrenches. However, the forces and torques are independent of the actual system state, which is not realistic. In fact, the human user forms a crucial feedback loop with their musculoskeletal system that converts pose displacements of the hand into end effector forces and torques, as illustrated in Figure 5.3. Therefore, a linear omnidirectional spring-damper model is proposed as an option to simulate the exerted user wrench.

In this model, the end effector force, i.e., the translational part of the wrench,

$$\underline{f}^B_H = -k_{\text{user,tran}} \left(\underline{x}^B_{BU} - \underline{x}^B_{BH}\right) + d_{\text{user,tran}} \dot{\underline{x}}^B_{BH}$$

is modeled using a Cartesian spring with stiffness $k_{\text{user,tran}}$ between a virtual user reference frame U and the end effector frame H. The damping $d_{\text{user,tran}}$ acts as a dissipative element. The sign convention, which may seem counterintuitive, is chosen to be compatible with Section 5.1. To calculate the rotational part of the wrench, the orientation deviation $\mathbf{C}^H_U = \mathbf{C}^{B\mathsf{T}}_H \mathbf{C}^B_U$ is converted to an axis-angle notation, yielding the end effector torque

$$\underline{m}^B_H = -k_{\text{user,rot}} \left(\mathbf{C}^B_H \underline{r}^H_{HU} \alpha_{HU}\right) + d_{\text{user,rot}} \underline{\omega}^B_{BH}$$

with the spring constant $k_{\text{user,rot}}$ and the damping parameter $d_{\text{user,rot}}$ analogous to the translational part.

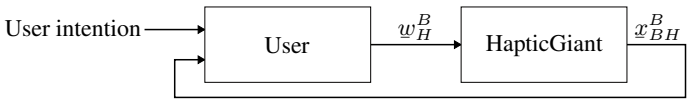


Figure 5.3: The user interaction forms a secondary feedback loop. This behavior is especially important for the design of HapticGiant's control system.

With this concept, the reference pose U can be set to arbitrary values. Besides preprogrammed reference trajectories or fixed values, the reference pose can be updated at runtime through external inputs, such as a keyboard or a gamepad. As an alternative, the reference pose can be determined from the hand pose of a user wearing a *Microsoft Hololens 2* [D15]. If the finger poses relative to the palm are evaluated with a suitable classifier for grasp detection, the state of the dead man's switch at the manipulator end effector can be emulated as well. As demonstrated in [S4], the resulting system can be used as a full-scale and highly immersive virtual prototype of HapticGiant. Evidently, the user is not receiving any real force feedback in this setup. However, the visually observed pose difference still provides some feedback about the expected haptic sensation to the user. Hence, interactive experiments regarding the haptic transparency or stability of the system can still be conducted without having access to the real system.

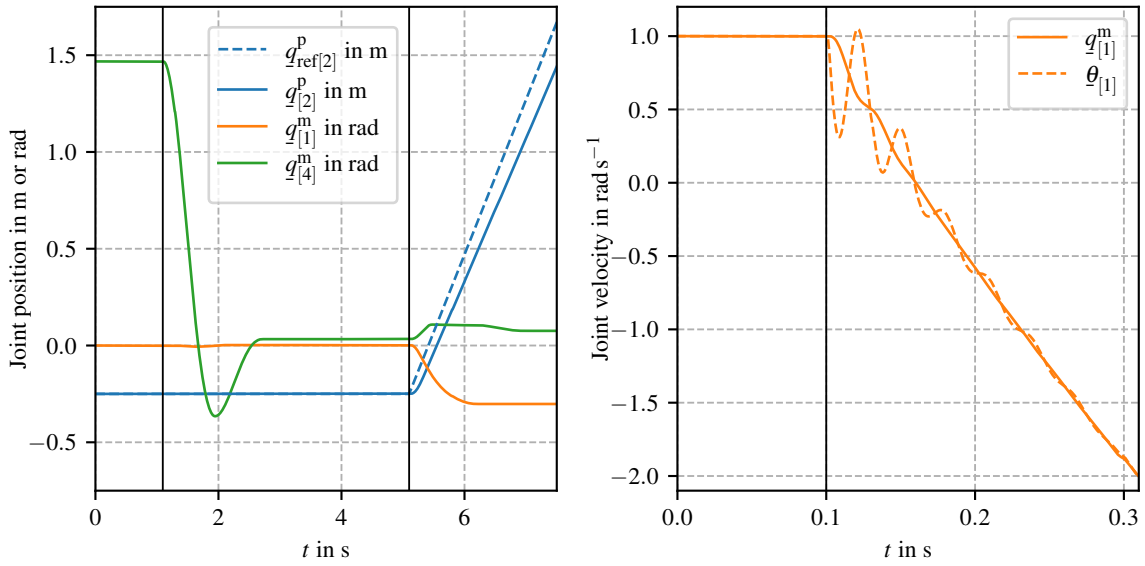
5.5 Evaluation

The simulation was implemented in C++ and RTCF with the help of *pinocchio* [120] for kinematic and dynamic calculations. To prove the effectiveness of the proposed simulation, two example scenarios were analyzed on a laptop with an *Intel Core i7-9750H* CPU.

In the first scenario shown in Figure 5.4a, the system starts with the manipulator set to the configuration $\underline{q}^{\mathrm{m}} = \begin{pmatrix} 0 & \pi/2 & 0 & \pi/2 & 0 & -0.8 \end{pmatrix}^{\mathsf{T}} \mathrm{rad}$, i.e., the link between the manipulator elbow and wrist is aligned horizontally. All setpoints are zero. At $t = 1.1\,\mathrm{s}$, the dead man's switch is pressed in simulation, causing the elbow joint (joint 4) to rotate towards the ground. The joint comes to rest after less than $2\,\mathrm{s}$ at an angle that is different from the stable equilibrium without Coulomb friction, which is at $\underline{q}^{\mathrm{m}}_{[4]} = 0\,\mathrm{rad}$. At $t = 5.1\,\mathrm{s}$, the y-axis of the PPU is commanded to move with $0.8\,\mathrm{m/s}$. Due to the inclusion of the hybrid dynamics in the simulation, manipulator joint 1 begins to rotate during the acceleration phase of the PPU. Likewise, the elbow joint changes its configuration. When the PPU has reached constant velocity, the elbow joint remains in a different equilibrium than before the PPU acceleration, also due to Coulomb friction effects.

In the second scenario, manipulator joint 1 is initially commanded with a constant torque of $-20\,\mathrm{Nm}$. At $t = 0.1\,\mathrm{s}$, the commanded torque is changed to $166\,\mathrm{Nm}$. The resulting joint velocity regarding motor- and load-side encoders is shown in Figure 5.4b. As expected with SEAs, the load-side velocity lags behind the motor-side velocity, which exhibits damped oscillations.

On average, the simulation took $64.8\,\mu\mathrm{s}$ and maximally $115\,\mu\mathrm{s}$ to complete a simulation step of $1\,\mathrm{ms}$, including all computations for the system dynamics and sensors as well as the handling of input and output data. This proves that the simulation can run under real-time constraints with a fraction of the actual cycle time, meaning that there is plenty of time left for other components. The calculation of the gradient $\dot{\underline{\psi}}$ took $3.3\,\mu\mathrm{s}$ on average, whereas $2.2\,\mu\mathrm{s}$ where spent on solving the quadratic program (5.5). To get the total time spent for the gradient calculation in one integration step, the stated computation times must be multiplied by five because the gradient is calculated four times for the Runge-Kutta integrator and once more to get the system acceleration for the new state.



(a) Coulomb friction and hybrid dynamics. The highlighted times correspond to the moment of the brake release and the step of the velocity command.

(b) Oscillations during reversal of the torque setpoint. The highlighted time corresponds to the moment of torque reversal.

Figure 5.4: Dynamic simulation behavior in two example scenarios.

5.6 Discussion

As demonstrated in the previous section, the presented simulation is very fast. In fact, it is faster than the actual hardware interface, which already spends 109.4 µs on average for communicating with the EtherCAT devices. Many aspects of HapticGiant are modeled in the simulation. In particular, the joint elasticity and non-linear Coulomb friction effects, combined with a velocity-controlled PPU, create a unique solution regarding the computational speed and the number of features. Although this solution is tailored to HapticGiant, the presented combination of methods may be useful for other robotic systems, especially mobile manipulators, when a general-purpose simulation framework is unsuitable.

In the current state, the simulation is already satisfactory for developing and testing new features for HapticGiant. In practice, almost all features from Chapter 7 were developed and tested using the proposed simulation before they were deployed on the real system. In many cases, only the parameters related to the human-machine interaction had to be fine-tuned during deployment on the target hardware, which confirms the benefits of the proposed simulation.

Nevertheless, there is still much room for improvement. Neither the Stribeck effect [121] nor load-dependent friction effects are considered in the current state of the simulation. Furthermore, the system dynamics are assumed to be fully deterministic, and the real system has many resonances beyond the ones coming from the SEAs, as demonstrated in Section 6.1.4. On the input side, extra interaction wrenches to explore the handling of user-manipulator collisions can be added, although this requires additional research regarding suitable user models. Moreover, the sensor models can be improved, especially by incorporating additional error terms and colored non-Gaussian sensor noise. These improvements would also require more sophisticated methods for system identification and potentially stochastic system models. With that many options, it is also important to study which effects are significant for a good match between simulation and reality to avoid overfitting.

CHAPTER
6

Kinematic State Estimation

Contents

> If you cannot measure it
> you cannot control it.
>
> JOHN GREBE (1900 – 1984)

At some point, a joint position tracking controller is required for any kinesthetic haptic interface under admittance control. Basic controllers, such as proportional-integral-derivative (PID) controllers, only require measurements of the joint position, but suffer from poor performance. More sophisticated robot joint controllers [122–126], safety features [127], or identification methods [128] require additional information about the kinematic state of the manipulator. For these, the load-side joint velocity and acceleration are of particular interest to achieve better tracking behavior, especially in the context of SEAs, where the load-side joint angles tend to oscillate due to the spring-damper nature of the underlying system. This is also the case for HapticGiant, which requires accurate and real-time data for load-side position $\underline{q}$, velocity $\dot{\underline{q}}$, and acceleration $\ddot{\underline{q}}$ to achieve optimal control performance.

In theory, the joint position data from the encoders can be numerically differentiated to obtain the desired data. In practice, however, the differentiation results in noise amplification and delayed velocity estimates. The noise can be mitigated using an appropriate low-pass filter, but this comes at the cost of an additional phase delay, especially at higher frequencies, which will negatively affect the stability, tracking behavior, and disturbance rejection of downstream controllers. While the resulting delay is usually still tolerable for controllers with velocity feedback, the second numerical differentiation, which

is used to obtain an acceleration estimate, will result in unacceptable signal properties. To still obtain accurate and timely kinematic state estimates, additional sensors are required. One promising option for this is to attach IMUs with gyroscopes and accelerometers to the manipulator. Motivated by the low cost and small form factors, this idea has also attracted interest from the research community [129].

In [130], the authors propose a method to estimate the velocity of a 1D motion by combining low-frequency information from position measurements and high-frequency information from accelerometer measurements. For the data fusion, a complementary filter-like approach is compared with a velocity observer. In both cases, appropriate strategies for estimating the scale factor and the bias of the accelerometer are included. Although the method is experimentally proven and an extension to 3D motions is presented, no acceleration estimates are obtained despite the presence of accelerometers in the sensor setup. In [131], a similar approach is studied to improve the end effector position estimate of an industrial robot with a triaxial gyroscope and a triaxial accelerometer at its end effector. The end effector position and orientation are estimated using a complementary filter or a Kalman filter (KF). In both cases, an improvement of the estimated tool position was observed in practical experiments, especially in regimes with high acceleration changes. Nevertheless, the approach is only useful for Cartesian control concepts, since no information about the individual joints is obtained.

An interesting approach moving away from Cartesian estimates can be found in [132], where load-side joint positions are estimated from motor-side encoders and a triaxial accelerometer at the end effector. However, this approach is not applicable to HapticGiant because the load-side joint positions are measured directly by encoders. Furthermore, the exact parameters of the dynamic model as well as the sensor calibration must be known beforehand.

The methods presented so far require the inertial sensor to be placed at the end effector, which is questionable for arbitrary kinematic topologies. As an alternative, [133] adds one triaxial gyroscope to each link of the manipulator to estimate the angular rate of the individual links. In combination with one additional triaxial accelerometer and three uniaxial accelerometers per link, the joint accelerations can be additionally estimated. Compared to the number of scalars that are estimated with this setup, the number of sensors is quite high. In addition, prismatic joints are not considered, and intrinsic sensor calibration regarding scale factors and accelerometer biases is required. A similar approach is proposed in [134], where each link is equipped with a triaxial gyroscope and a triaxial accelerometer. The measured angular velocities and linear accelerations of two adjacent links are used to obtain raw joint velocity and acceleration data, which is then fused with encoder measurements in two KFs per joint. As before, this method works only with revolute joints and requires off-line calibration. The same limitations appear in [135], where a setup with one triaxial accelerometer per link is used to estimate the kinematic joint state up to acceleration level. In [136], this approach is extended with one triaxial gyroscope per link. In both cases, no raw joint velocity or acceleration is computed prior to the fusion with the encoder data. Instead, all data is processed simultaneously in an extended Kalman filter (EKF) per joint.

Up to this point, all methods require accurate sensor calibrations regarding the sensor pose and the scale-factor errors. The approach in [126] avoids this problem by using a two-stage filtering approach within the control system of an exoskeleton. In the first stage, the accelerometer measurements from each link are used in combination with the differential kinematics to compute a biased joint acceleration estimate. In parallel, the encoder measurements are numerically differentiated twice and low-pass filtered to obtain an unbiased but delayed joint acceleration estimate. The biased joint acceleration estimate is then delayed by a time, which matches the delay of the unbiased estimate, and subtracted from the joint acceleration from encoder measurements to obtain an estimate of the bias. In the second stage, the low frequency content of the bias is extracted using a low-pass filter and used to correct the biased joint acceleration estimate, which is smoothed with another low-pass filter. This approach

can also handle prismatic joints and is robust to calibration errors, but it depends on one triaxial accelerometer per joint. Moreover, gyroscope measurements cannot be incorporated, and velocity estimates are still purely encoder-based.

To overcome the limitations of the presented state-of-the-art methods, a novel kinematic state estimation method based on an EKF is proposed in the first half of this chapter. Due to its unified approach, this method can handle arbitrary sensor configurations of gyroscopes and accelerometers on serial manipulators with prismatic or revolute joints without placement constraints. Furthermore, the systematic inclusion of measurement errors eliminates the need for accurate intrinsic or extrinsic sensor calibration. Various experiments will also demonstrate that the proposed method outperforms comparable methods in terms of estimation accuracy, while still being real-time capable with execution frequencies above $1\,\mathrm{kHz}$.

Having an algorithm that can fuse data from an arbitrary inertial sensor configuration raises the question of where to ideally place the inertial sensors. From a naïve perspective, as many sensors as possible may seem to be the best choice. In reality, however, cost, mounting space, bandwidth, and computational constraints prohibit such a trivial answer. In the specific case of kinematic state estimation with inertial sensors, the problem of optimal sensor placement encompasses attributes such as the sensor count, the estimation accuracy, and the measurement range utilization.

The problem of optimal sensor placement arises in various engineering fields [137, 138]. In the context of this chapter, [139] contains a study on the placement of triaxial accelerometers to obtain the most accurate joint angle estimates for a SCARA manipulator without encoders. In this study, the estimation error of different discrete sensor placements is quantified using a Monte Carlo simulation and the results are validated experimentally. A very similar approach is pursued in [140], where optimal IMU poses for an IMU-based motion capture system are determined based on real data using the root mean square error (RMSE) metric for different sensor configurations.

The need to run a simulation or an experiment poses the risk of missing scenarios in which the sensor configuration does not perform as expected. Furthermore, running a full-blown estimator for each sensor configuration can be very time-consuming for typically large configuration spaces, and it causes the results to be dependent on the estimation method. A more sophisticated approach is presented in [141], where the authors exploit the fact that the state covariance matrix of a Bayesian estimator can be independent of the actual measurements. This facilitates the formulation of an objective function depending on the sensor positions, which is subsequently minimized. While this method cures the above-mentioned issues, it cannot be applied to our kinematic state estimation problem because the posterior covariance of the EKF depends on the linearization point and thus indirectly on the measurements. Moreover, the method does not take into account additional sensor properties such as the measurement range. However, this is crucial because a saturated inertial sensor is practically useless for the proposed state estimation algorithm.

Another, general approach to the problem of optimal sensor placement originates from control theory, where observability is defined as the ability to determine the state of the system from its output, which can be interpreted as sensor measurements [142]. As with the kinematic state estimation based on IMU data, the state variables may not be directly measurable, making the concept of observability a promising approach for rating different sensor configurations and placements. This idea was also picked up in [143] to determine optimal sensor configurations for non-linear systems using the *observability Gramian*. Unfortunately, the computation of the observability Gramian of non-linear system requires a nominal trajectory. In the context of the kinematic state estimation, this is an issue as no prior knowledge about the trajectories may be available. The same method with the same problem appears in [144], where an optimal sensor set is selected for a variable impedance actuated robot. In addition

to the missing nominal trajectory, both methods do not consider the measurement range of the inertial sensors during the placement.

To the best of the author's knowledge, all existing methods for optimal sensor placement have some limitations that prevent their direct application to the sensor placement problem for kinematic state estimation. Therefore, a novel sensor placement method based on the observability Gramian is proposed in the second half of this chapter. In particular, this method deals with the lack of a reference trajectory as well as the finite measurement range of the inertial sensors. As a result, the best sensor configurations with a predefined number of sensors can be quickly evaluated regarding the quality of velocity and acceleration estimates without the need for simulation or real experiments.

As mentioned above, this chapter is divided into two parts. Section 6.1 presents a novel, inertial sensor-backed method for the kinematic state estimation of robotic manipulators. Section 6.2 then answers the question of which inertial sensor configurations are best suited for the proposed estimator. In both parts, the general problem is covered before the proposed solution is applied to HapticGiant.

This chapter is based on results presented in the author's publications [O2, O3].

6.1 Calibration-Free IMU-Based Kinematic State Estimation

This section presents a novel, IMU-based and calibration-free kinematic state estimation method. First, the problem is stated more precisely in Section 6.1.1. Then, the estimator concept is introduced in Section 6.1.2. The evaluation is split into a simulative study and experiments on real hardware in Section 6.1.3 and Section 6.1.4, respectively.

6.1.1 Problem Formulation

In the following, we assume a serial kinematic chain consisting of n revolute or prismatic joints with known forward kinematics. Its kinematic quantities are modeled using the rules from Section 2.3.1. Each joint is equipped with an encoder, which is positioned at the load-side when SEAs are in use. For the state estimation, $n_\mathrm{g} \geq 0$ triaxial gyroscopes $G = \{G_1, \ldots, G_{n_\mathrm{g}}\}$ and $n_\mathrm{a} \geq 0$ triaxial accelerometers $A = \{A_1, \ldots, A_{n_\mathrm{a}}\}$ are arbitrarily distributed across the kinematic chain. Each inertial sensor $S \in G \cup A$ is rigidly coupled to its parent link with link start frame $p(S)$. The mounting offset $\underline{x}_{p(S)S}^{p(S)}$ and $\mathbf{C}_S^{p(S)}$ is assumed to be known with tolerances typical for the stage after manufacturing and prior to extrinsic sensor calibration. Except for these mounting poses and general sensor characteristics, such as noise level, no parameters are determined by the calibration of individual sensors. In particular, the biases, scale-factor errors, cross-axis sensitivity, and the exact mounting pose of the inertial sensors are assumed to be unknown. For the state estimation, the measured data of the gyroscopes and accelerometers is denoted with $\tilde{\underline{\omega}}_{WG_1}^{G_1}, \ldots, \tilde{\underline{\omega}}_{WG_{n_\mathrm{g}}}^{G_{n_\mathrm{g}}} \in \mathbb{R}^3$ and $\tilde{\underline{a}}_{WA_1}^{A_1}, \ldots, \tilde{\underline{a}}_{WA_{n_\mathrm{a}}}^{A_{n_\mathrm{a}}} \in \mathbb{R}^3$, respectively, with W being a common reference frame at the base of the manipulator. The raw measurements of all encoders are collected in $\tilde{\underline{q}} \in \mathbb{R}^n$. The goal now is to estimate the kinematic state composed of joint position $\underline{q}$, velocity $\dot{\underline{q}}$, and acceleration $\ddot{\underline{q}}$ as accurately as possible using all available data from the encoders, gyroscopes, and accelerometers. Ideally, the resulting state estimate can be computed without delay and under real-time constraints.

6.1.2 Estimator Design

According to the literature presented at the beginning of this chapter, KF-based estimators are a promising option for the task of inertial sensor-based state estimation. When arbitrary sensor configurations

are required, the measurement functions are non-linear in general. For this reason, the EKF is a suitable choice due to its simplicity and computational efficiency. Furthermore, arbitrary sensor configurations prohibit the use of decoupled estimators per joint as found in [135, 136]. With the requirement of a calibration-free solution, the measurement error must be handled in some way within the filter. As a result, the time-discrete system model

$$\underline{x}_{k+1} = \begin{pmatrix} \mathbf{I} & \Delta t\mathbf{I} & \frac{1}{2}\Delta t^2\mathbf{I} & \frac{1}{6}\Delta t^3\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \Delta t\mathbf{I} & \frac{1}{2}\Delta t^2\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \Delta t\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{pmatrix} \underline{x}_k + \underline{w}_k$$

with the state

$$\underline{x}^\mathsf{T} = \begin{pmatrix} \underline{q}^\mathsf{T} & \dot{\underline{q}}^\mathsf{T} & \ddot{\underline{q}}^\mathsf{T} & \dddot{\underline{q}}^\mathsf{T} & \underline{b}_\mathrm{g}^\mathsf{T} & \underline{b}_\mathrm{a}^\mathsf{T} \end{pmatrix} \in \mathbb{R}^{1\times(4n+3(n_\mathrm{g}+n_\mathrm{a}))}$$

and the sampling time Δt is proposed. This model combines a constant jerk model with a constant bias model. It is deliberately formulated without inputs, as these would require either a dynamic model of the manipulator, directly measurable joint accelerations, or knowledge of the reference trajectory [134]. Although this means that not all available information may be considered for the kinematic state estimation, it guarantees that the resulting estimator can be paired with any manipulator without dictating special requirements. The stacked biases

$$\underline{b}_\mathrm{g}^\mathsf{T} = \begin{pmatrix} \underline{b}_{G_1}^\mathsf{T} & \cdots & \underline{b}_{G_{n_\mathrm{g}}}^\mathsf{T} \end{pmatrix} \in \mathbb{R}^{1\times 3n_\mathrm{g}} \quad \text{and}$$
$$\underline{b}_\mathrm{a}^\mathsf{T} = \begin{pmatrix} \underline{b}_{A_1}^\mathsf{T} & \cdots & \underline{b}_{A_{n_\mathrm{a}}}^\mathsf{T} \end{pmatrix} \in \mathbb{R}^{1\times 3n_\mathrm{a}}$$

contain the bias value for each of the triaxial gyroscopes and accelerometers. In an ideal world, these values are identical to the constant sensor bias. In the proposed estimator formulation, $\underline{b}_\mathrm{g}$ and $\underline{b}_\mathrm{a}$ serve as *momentary biases* reflecting the current sum of all calibration errors, similar to the estimator concept in [126]. This eliminates the need for an accurate calibration of the inertial sensors. The zero-mean white Gaussian process noise $\underline{w}_k \in \mathbb{R}^{4n+3(n_\mathrm{g}+n_\mathrm{a})}$ with the covariance

$$\mathrm{Cov}(\underline{w}_k) = \mathrm{diag}\begin{pmatrix} \underline{0}_n^\mathsf{T} & \underline{0}_n^\mathsf{T} & \underline{0}_n^\mathsf{T} & \sigma_{\dddot{q}}^2\underline{1}_n^\mathsf{T} & \sigma_{b_{G_1}}^2\underline{1}_3^\mathsf{T} & \cdots & \sigma_{b_{G_{n_\mathrm{g}}}}^2\underline{1}_3^\mathsf{T} & \sigma_{b_{A_1}}^2\underline{1}_3^\mathsf{T} & \cdots & \sigma_{b_{A_{n_\mathrm{a}}}}^2\underline{1}_3^\mathsf{T} \end{pmatrix}$$

is used to enable the necessary changes of the joint jerk $\dddot{\underline{q}}$ as well as the bias changes. The individual standard deviations σ are tuning parameters.

The measurement model for the proposed EKF-based filter stacks the measurements from encoders and inertial sensors. This way, arbitrary inertial sensor setups are feasible, which is an advantage over the presented state of the art. Furthermore, dynamic switching of sensors at runtime is possible, e.g., to disable a faulty or saturated sensor. Formally, the resulting measurement vector is

$$\tilde{\underline{y}} = \begin{pmatrix} \tilde{\underline{q}} \\ \tilde{\underline{\omega}}_{WG_1}^{G_1} \\ \vdots \\ \tilde{\underline{\omega}}_{WG_{n_\mathrm{g}}}^{G_{n_\mathrm{g}}} \\ \tilde{\underline{a}}_{WA_1}^{A_1} \\ \vdots \\ \tilde{\underline{a}}_{WA_{n_\mathrm{a}}}^{A_{n_\mathrm{a}}} \end{pmatrix} = \begin{pmatrix} \underline{q} \\ \underline{\omega}_{WG_1}^{G_1}(\underline{q}, \dot{\underline{q}}) + \underline{b}_{G_1} \\ \vdots \\ \underline{\omega}_{WG_{n_\mathrm{g}}}^{G_{n_\mathrm{g}}}(\underline{q}, \dot{\underline{q}}) + \underline{b}_{G_{n_\mathrm{g}}} \\ \underline{a}_{WA_1}^{A_1}(\underline{q}, \dot{\underline{q}}, \ddot{\underline{q}}) + \underline{b}_{A_1} \\ \vdots \\ \underline{a}_{WA_{n_\mathrm{a}}}^{A_{n_\mathrm{a}}}(\underline{q}, \dot{\underline{q}}, \ddot{\underline{q}}) + \underline{b}_{A_{n_\mathrm{a}}} \end{pmatrix} + \underline{v}\,. \tag{6.1}$$

Sensor	Parameter	Value
Encoders	Noise standard deviation	$4 \cdot 10^{-4}$ m or rad
	Resolution	$1.2 \cdot 10^{-5}$ m or rad
IMUs	Gyroscope noise standard deviation	$0.32\,°/\mathrm{s} = 5.6 \cdot 10^{-3}\,\mathrm{rad/s}$
	Accelerometer noise standard deviation	$9.5 \cdot 10^{-3}\,\mathrm{m/s^2}$
	Position calibration error	Drawn uniformly from $[-2, 2]$ mm
	Orientation calibration error	Drawn uniformly from $[-2, 2]$ °

Table 6.1: Sensor parameters used for the evaluation in simulation.

Here, $\underline{v} \in \mathbb{R}^{n+3(n_\mathrm{g}+n_\mathrm{a})}$ is the zero-mean white Gaussian measurement noise, whose covariance matrix is diagonal with entries corresponding to the measurement noise covariance of the individual sensor channels. The measured angular rates $\tilde{\underline{\omega}}_{WG_i}^{G_i}$ are composed of the groundtruth value $\underline{\omega}_{WG_i}^{G_i}$ originating from the forward kinematics, the momentary bias $\underline{b}_{G_i}$, and the corresponding part of the measurement noise. The earth rotation is assumed to be negligible for the considered application. The measured linear accelerations are modeled analogously, except that the effect of gravity is included in $\underline{a}_{WA_i}^{A_i}$.

To use this measurement model in an EKF, the Jacobian $\partial\tilde{\underline{y}}/\partial\underline{x}$ is required. The partial derivatives with respect to quantities that appear linear or not at all in (6.1) are trivial to compute, e.g., $\partial\tilde{\underline{y}}/\partial\underline{b}_\mathrm{g}$ and $\partial\tilde{\underline{q}}/\partial\underline{q}$. The partial derivatives for the gyroscope and accelerometer measurements with respect to the desired kinematic state $\underline{q}$, $\dot{\underline{q}}$, and $\ddot{\underline{q}}$ are more elaborate due to the nonlinearities of the kinematic model. Using numerical derivatives for this purpose is theoretically possible, but computationally expensive because the resulting sub-Jacobian has $3(n_\mathrm{g}+n_\mathrm{a}) \times 3n$ entries, requiring many calculations of the forward kinematics. For this reason, the partial derivatives of the groundtruth angular rates and linear accelerations are computed analytically using the recursive expressions in appendix A with complexity $\mathcal{O}(n^2)$. To calculate these values for arbitrary sensor frames, the approach from Section 2.3.1 is used. As a result, the partial derivatives of $\underline{\omega}_{WG_i}^{G_i}$ are computed recursively using Theorem A.2. To include the effect of gravity on the accelerometers, $\underline{a}_{W\beta_0}^{W}$ is set to the gravity vector $\underline{g}^W$ before calculating the necessary link accelerations in the recursion (2.8). The partial derivatives of $\underline{a}_{WG_i}^{G_i}$ are then computed analogously using Theorems A.3 and A.4. A less accessible alternative to this approach can be found in [145], where the desired partial derivatives appear as a by-product in spatial coordinate notation.

For experiments, the proposed estimator, including the scheme for the efficient computation of the measurement Jacobian, was implemented in C++. Wherever possible, *pinocchio* [120] is used for kinematic calculations. The information about the sensor placement and the robot geometry is automatically extracted from a URDF file, making the implementation robot-agnostic. For the backend, a custom allocation-free version of the standard EKF was implemented using *Eigen* [146]. The initial state of the EKF is determined with low covariance using the joint encoder measurements at standstill. The biases are zero-initialized with high covariance.

6.1.3 Evaluation in Simulation

For the first part of the evaluation, the sensor models from Section 5.2 are used. The parameters of the IMUs are determined based on the datasheet of an *Invensense MPU-9520* [D11] with a triaxial gyroscope and a triaxial accelerometer as described in Section 5.3, with the exceptions listed in Table 6.1. The encoder parameters are chosen to match the sensing quality of a *Sensojoint Sensodrive 7005* [D3] joint and are also listed in Table 6.1. In the following, the manipulator model from Figure 6.1 with two prismatic and six revolute joints is used. The corresponding DH parameters are listed in Table 6.2.

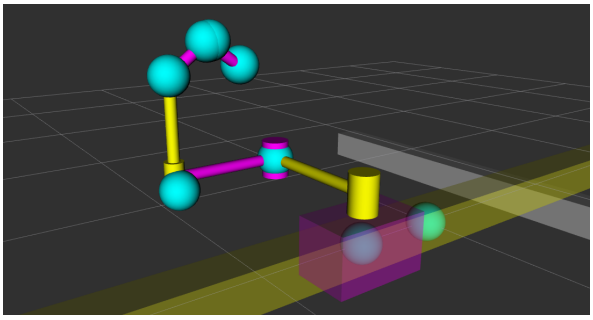


Figure 6.1: The manipulator used for the evaluation in simulation. The IMUs are drawn as blue spheres. Note, that IMU 6 and 7 are very close together due to the universal joint that is formed by the last two joints. Taken from [O2].

Joint i	θ	d	a	α
1	0	$\underline{q}_{[1]}$	0	90°
2	90°	$\underline{q}_{[2]}$	0	90°
3	$\underline{q}_{[3]}$	0	0.7 m	0
4	$\underline{q}_{[4]} + 90°$	0	0.6 m	180°
5	$\underline{q}_{[5]}$	−0.5 m	0	90°
6	$\underline{q}_{[6]}$	0	−0.25 m	0
7	$\underline{q}_{[7]}$	0	0	90°
8	$\underline{q}_{[8]}$	0.25 m	0	0

Table 6.2: DH parameters of the manipulator used for the evaluation in simulation. Taken from [O2].

Although the model is not equivalent to HapticGiant, it has a reasonable kinematic similarity, which can be verified by comparing Table 6.2 with Table 4.1. An IMU is placed at the end of each link, adding a total of eight IMUs to the manipulator. For experiments with fewer sensors, see Section 6.2.3.

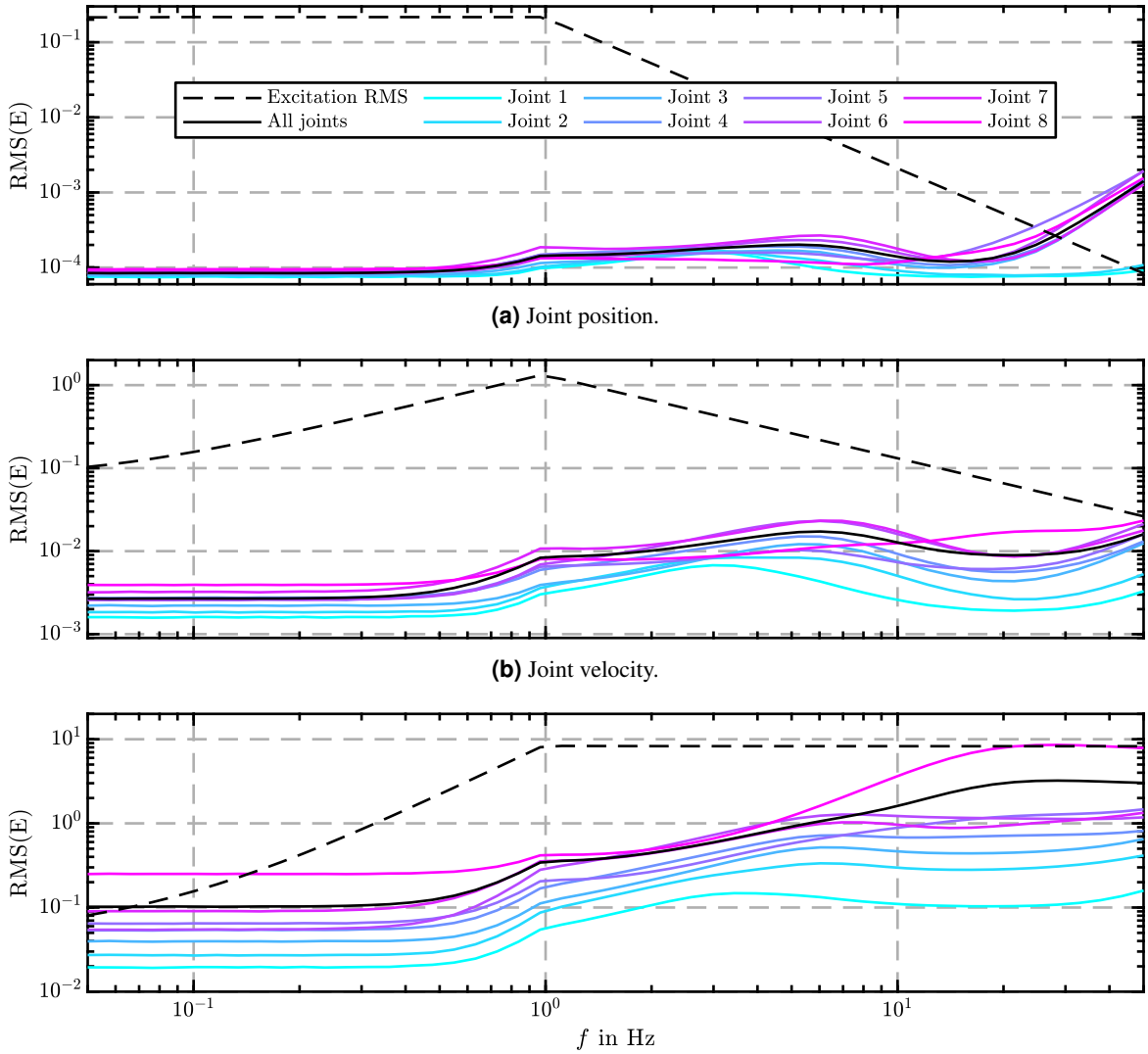
The kinematic excitation with a sampling rate of 1 kHz is realized using joint-wise sinusoidal signals with varying frequencies. To obtain feasible state trajectories, the excitation is windowed with a cosine window of length 10 s. The resulting acceleration amplitude is set to 20 rad/s or m/s as long as the maximum position does not exceed 30° or 0.52 m. If the excitation exceeds these limits, the acceleration amplitude is adjusted accordingly. This matches the experience from real manipulation tasks where high-frequency movements are usually paired with low position amplitudes and vice versa. To consider different situations, 30 initial configurations are drawn uniformly from $[-0.52, 0.52]$ m or rad and paired with initial phase offsets drawn uniformly from $[0, 2\pi]$. The temperature is simulated as a sinusoidal signal with an amplitude of 5 K and a period duration of 10 s.

Estimator Performance

For the evaluation, the measurement noise covariance matrix $\mathrm{Cov}(\underline{v})$ was set according to the values in Table 6.1. The process noise covariance was set to

$$\mathrm{Cov}(\underline{w}_k) = \mathrm{diag}\left(\underline{0}_8^\mathsf{T} \quad \underline{0}_8^\mathsf{T} \quad \underline{0}_8^\mathsf{T} \quad 12.5^2\underline{1}_{24}^\mathsf{T} \quad 0.001^2\underline{1}_{24}^\mathsf{T} \quad 0.01^2\underline{1}_{24}^\mathsf{T}\right)$$

after manual tuning. The large joint jerk noise is necessary to track transient acceleration changes. As motivated above, the bias drift is much higher than the specified values, enabling the filter to track the momentary bias instead of the idealized intrinsic sensor bias. In addition, the noise for the



(a) Joint position.

(b) Joint velocity.

(c) Joint acceleration.

Figure 6.2: RMSE data of the proposed kinematic state estimation depending on frequency regarding position, velocity, and acceleration. The solid black line represents the RMSE over all joints. Taken from [O2].

accelerometer biases is higher than for the gyroscope biases, because acceleration measurements are more sensitive to misalignment due to the ever-present measurement of the gravitational acceleration.

Figure 6.2 shows the resulting RMSEs over the excitation frequency for all joints. We can see that the proposed estimator determines the joint positions with an RMSE that is lower than the standard deviation of the raw encoder measurements from Table 6.1 for frequencies up to 20 Hz. For higher frequencies, the proposed estimator performs worse than pure encoder measurements and the error becomes larger than the excitation. However, the absolute errors are still very small. According to Figure 6.2b, the RMSE of the velocity estimates is more than a decade below the root mean square (RMS) value of the excitation for frequencies up to 5 Hz, indicating a functional joint velocity estimation. As with the position estimates, the RMSE is below the noise level of the gyroscopes for excitations in the frequency range below 0.5 Hz. Similarly, the RMSE of the acceleration estimates in Figure 6.2c is about a decade below the RMS value of the excitation for frequencies between 0.5 Hz and 5 Hz. At lower frequencies, the RMSE is greater than the excitation. Although this looks

problematic at first glance, it is not a real problem because the excitations under consideration are very low due to the position limits of the excitation. A more serious issue is the limited estimation quality of the last joint, especially at higher frequencies. However, this is not a limitation of the estimator itself, but a problem of the sensor setup, in which the only accelerometer affected by the last joint is mounted in direct proximity to its rotation axis. In this case, the resulting short lever arm makes the accelerometer insensitive to the joint acceleration of interest.

For the given eight-axis manipulator, each estimator update took $560\,\mu s$ on average to complete on an *Intel Core i7-9750H* CPU. From this time, $6\,\mu s$ were spent on average for the computation of the Jacobian. Therefore, the estimator not only delivers accurate and meaningful results, but also real-time compatible computation times, enabling update frequencies greater than $1\,kHz$ with typical manipulator designs.

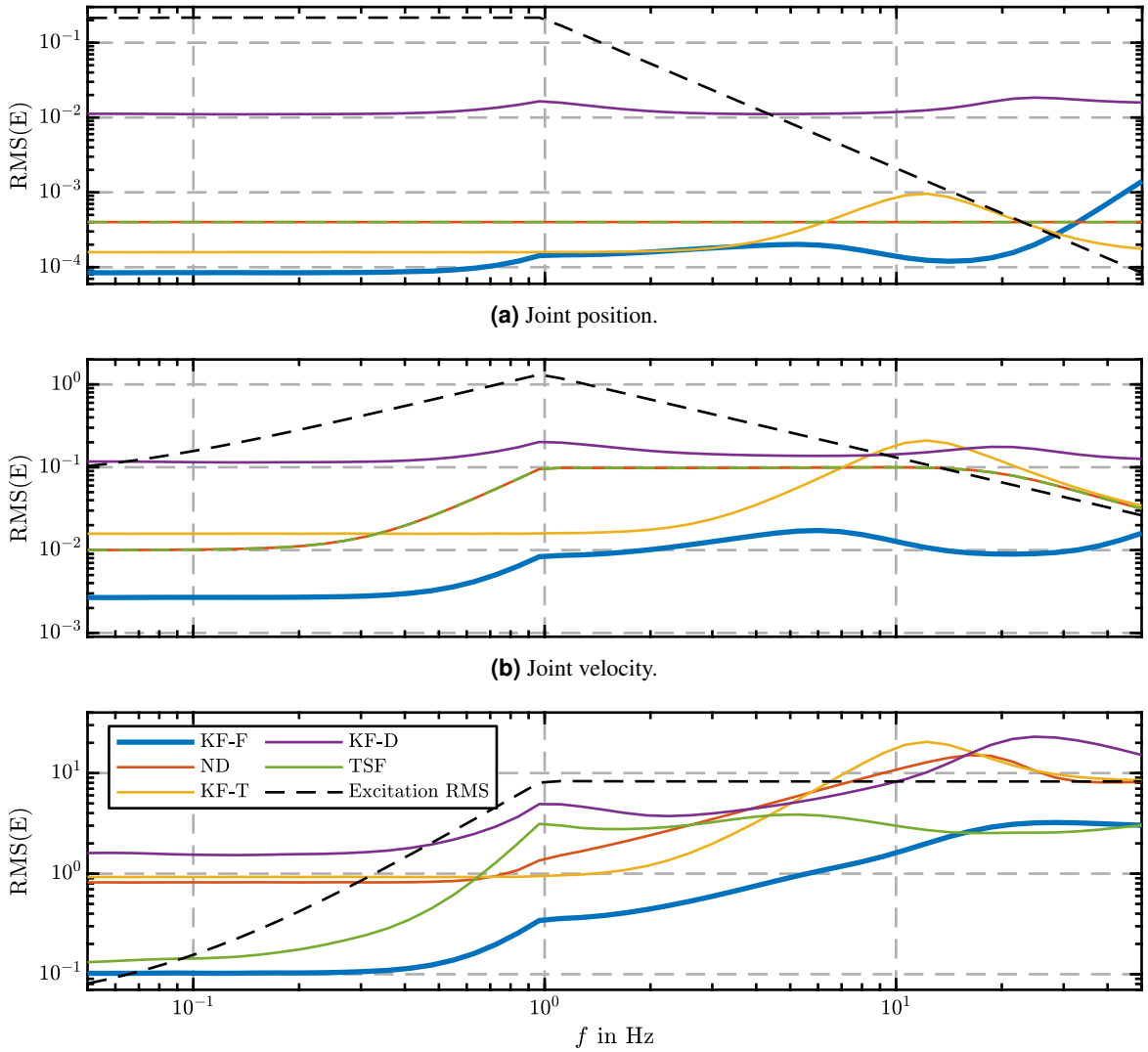
To study the effect of different modeling errors, the estimator was tested in simulations with a gradually increasing number of simulated sensor effects. This validated the estimator's ability to handle constant and temperature-dependent biases. Test results with perfect sensor calibrations imply that a good calibration of the inertial sensors, especially with respect to scale-factor errors, cross-axis sensitivity, and sensor placement, is beneficial for the estimator's performance, but not necessary in general. Thus, the estimator can be considered calibration-free. Furthermore, the estimator appears to be robust against increased noise levels without parameter adjustments [O2].

Comparison with Other Estimators

Next, the proposed estimator with its **f**ull **K**alman **f**ilter (*KF-F*) is compared with four competing concepts for kinematic state estimation. The competitors, whose parameterization is explained in detail in [O2], are:

- *ND*: This estimator calculates the first- and second-order **n**umerical **d**erivatives from encoder data and applies low-pass filters to obtain less noisy velocity and acceleration estimates. Data from inertial sensors is not considered.
- *KF-T*: In this estimator concept, the joint velocity and acceleration estimates are obtained from encoder measurements using a **t**rivial **K**alman **f**ilter with a constant jerk model. As with ND, no inertial sensor data is processed.
- *KF-D*: As presented in the introduction, this estimator with **d**ecoupled **K**alman **f**ilters implements the concept from [135, 136] to include data from gyroscopes and accelerometers.
- *TSF*: The last estimator in the comparison is the previously introduced **t**wo-**s**tage **f**iltering approach from [126]. By design, only data from accelerometers is fused with the encoder data.

Based on the evaluation in Figure 6.3, we can see that the proposed estimator (KF-F) outperforms all competitors in the majority of the tested frequency range. Especially for the acceleration and velocity estimation, KF-F shows superior properties compared to methods that do not take into account inertial sensors (ND and KF-T). Interestingly, KF-D performs worse than all other methods, despite having more data available. This effect is caused by neglecting the IMU biases and calibration errors in the estimation algorithm. The comparison between TSF and KF-F shows that the proposed KF-F is superior to TSF in terms of position and velocity accuracy. This is due to the fact that TSF relies solely on encoder measurements and numerical derivatives for estimating position and velocity. With regard to the acceleration accuracy, KF-F also outperforms TSF by having lower RMSE values up to $10\,Hz$. A closer analysis of the simulation data revealed that this observation also holds on a per-joint basis. The largest gains are achieved for the last joint, which, as discussed earlier, has a limited observability regarding the accelerometer measurements. The average computation time for KF-F is $560\,\mu s$. In comparison, ND, KF-T, KF-D, and TSF require $< 1\,\mu s$, $24\,\mu s$, $113\,\mu s$, and $6\,\mu s$, respectively, indicating that the improved accuracy of KF-F comes at the cost of increased computation times.



(a) Joint position.

(b) Joint velocity.

(c) Joint acceleration.

Figure 6.3: Comparison of the proposed kinematic state estimation with competing approaches. Taken from [O2].

6.1.4 Evaluation on Real Hardware

The evaluation on real hardware is based on experiments with a SCARA manipulator and HapticGiant.

SCARA Manipulator

For this part, the planar SCARA manipulator from Figure 6.4 is actuated with an update frequency of $1\,\mathrm{kHz}$. Both links have a length of $0.25\,\mathrm{m}$ and are driven by two *Sensodrive Sensojoint 3008* motors [D3]. An *Invensense MPU-9520* IMU [D11] is mounted on the end effector. The covariance matrices were manually tuned to

$$\mathrm{Cov}(\underline{w}_k) = \mathrm{diag}\left(\underline{0}_6^\mathsf{T} \quad 12.5^2\underline{1}_2^\mathsf{T} \quad 0.005^2\underline{1}_3^\mathsf{T} \quad 0.05^2\underline{1}_3^\mathsf{T}\right) \quad \text{and}$$
$$\mathrm{Cov}(\underline{v}) = \mathrm{diag}\left((6\cdot 10^{-5})^2\underline{1}_2^\mathsf{T} \quad 0.055^2\underline{1}_3^\mathsf{T} \quad 0.188^2\underline{1}_3^\mathsf{T}\right)$$

to match the environmental conditions. For the experiments, the end effector is moved back and forth along a straight line. At $t = 25\,\mathrm{s}$, the mechanical brakes of the joints are triggered.

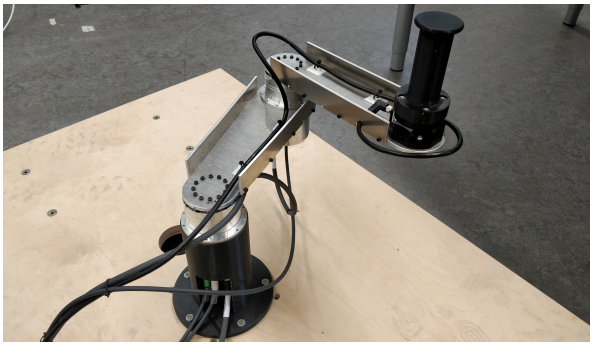


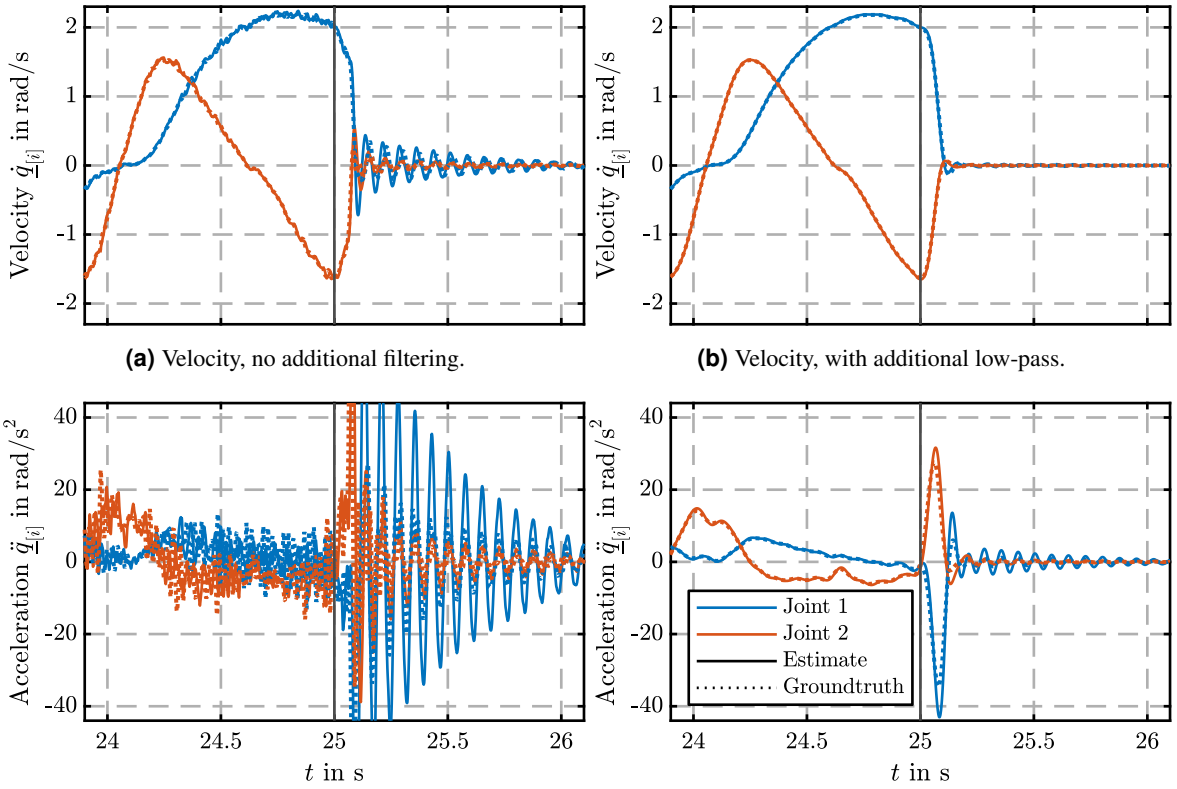
Figure 6.4: The planar SCARA manipulator used for the state estimation experiments on real hardware. Taken from [O2].

The resulting joint velocity and acceleration estimates are shown in Figures 6.5a and 6.5c. For comparison, the encoder data is also plotted as groundtruth after numerical differentiation and smoothing with an acausal, delay-free Butterworth filter of order eight with a cutoff frequency of $50\,\mathrm{Hz}$. Unfortunately, the match between estimated and groundtruth data is poor, especially for the acceleration. For this reason, the actual accelerometer measurements were compared with the expected accelerometer measurements during the braking phase. To obtain these, the encoder data was numerically differentiated and smoothed as described above. The resulting joint velocity and acceleration estimates are then fed into the differential forward kinematics to predict the sensor readings. Figure 6.6 shows the resulting Cartesian accelerations for the second sensor axis. Clearly, there is a significant mismatch between the expected and actual measurements that cannot be fully attributed to scale-factor errors, cross-axis sensitivity, or small errors in the kinematic model. The smoothing of the numerical derivatives is also not the cause, since the attenuation of the Butterworth filter for the relevant frequencies is less than $1\,\mathrm{dB}$. As a consequence, there must be additional errors, which may originate from the encoders, the IMU, or the limited stiffness of the links in the test setup. To still be able to evaluate the performance of the proposed estimator without a dedicated motion capture setup, the bandwidth is limited to $10\,\mathrm{Hz}$ using an acausal, delay-free Butterworth filter of order eight, resulting in Figures 6.5b and 6.5d.

From these figures, we can conclude that the proposed estimator is indeed able to estimate the joint velocities and accelerations, even during abrupt acceleration changes and fast oscillation processes, as seen after $t = 25\,\mathrm{s}$. In the presented example, the RMS value of the groundtruth signal is $1.253\,\mathrm{rad/s}$ and $5.727\,\mathrm{rad/s^2}$ for velocity and acceleration, respectively. The corresponding estimator yields an RMSE of $0.005\,\mathrm{rad/s}$ and $0.430\,\mathrm{rad/s^2}$, respectively.

HapticGiant

Unfortunately, all attempts to integrate the proposed estimator into HapticGiant failed. As suggested by the evaluation with the SCARA robot, one might suspect that the particularly long links of HapticGiant's manipulator have insufficient stiffness compared to the joint stiffness. For this reason a test was performed with the y-axis of the PPU, where a triaxial accelerometer is mounted axis-parallel to the trolley, in close proximity to the linear encoder measuring the y-coordinate of the PPU. See Section 4.1.1 for details about the setup. The trolley is very stiff, so the predicted acceleration should match the measured acceleration. In Figure 6.7a, the expected acceleration is compared to the measured acceleration analogous to Figure 6.6. Setting the cutoff frequency to $300\,\mathrm{Hz}$ ensures that the relevant frequencies are not attenuated. Interestingly, the measured acceleration is much smaller than expected, which is the inverse behavior of the SCARA manipulator. Therefore, the author suspects that there are additional effects within the encoders and inertial sensors, such as internal resonances or undocumented low-pass filters in the internal signal processing chains, that significantly violate the



(a) Velocity, no additional filtering.

(b) Velocity, with additional low-pass.

(c) Acceleration, no additional filtering.

(d) Acceleration, with additional low-pass.

Figure 6.5: Comparison of the joint state estimates for velocity and acceleration with groundtruth data from acausal filtering of encoder measurements. The highlighted time corresponds to the moment of brake actuation.

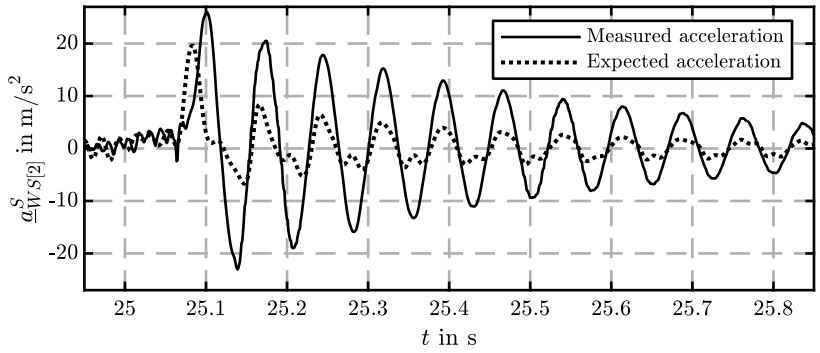
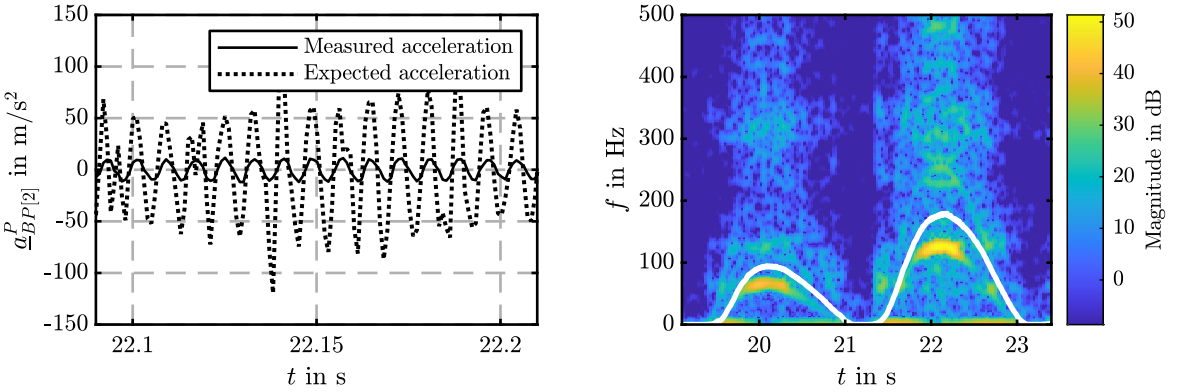


Figure 6.6: Discrepancies between measured and expected acceleration data.



(a) Discrepancies between measured and expected acceleration data.

(b) Short-time Fourier transform of the measured acceleration $\underline{a}^P_{BP[2]}$. The ordinate of the white line is proportional to $|\underline{q}^{\mathrm{p}}_{[2]}|$.

Figure 6.7: Analysis of the PPU acceleration data in y-direction for a single-axis motion.

model assumptions of the proposed estimator. The spectrogram in Figure 6.7b shows that filtering out the observed oscillations, which originate from structural vibrations, is not a viable workaround as the vibrations are velocity dependent and extend into the frequency bands of interest.

6.2 Optimal Placement of Inertial Sensors

The proposed estimator can fuse data from arbitrary inertial sensor configurations, raising the question of which sensor configurations are actually useful and which are a waste of hardware and computational resources. In the following, this question is formalized in Section 6.2.1. Then, a novel observability-based placement procedure is presented in Section 6.2.2 and evaluated on two different manipulators, including HapticGiant, in Section 6.2.3.

6.2.1 Problem Formulation

The proposed sensor placement procedure assumes a serial kinematic chain as in Section 6.1.1. Each joint is equipped with a load-side absolute encoder with known measurement noise standard deviation. This guarantees that all joint velocities and accelerations are fundamentally observable. To improve the accuracy of the joint velocity and acceleration estimates produced by the presented estimator, or any other compatible estimator, additional triaxial inertial sensors can be placed across the kinematic chain. Each inertial sensor $\mathcal{S}$, which can be either a gyroscope $\mathcal{G}$ or an accelerometer $\mathcal{A}$, is described by the tuple

$$\mathcal{S} = \left(\mathcal{I},\, \sigma,\, y_{\max},\, l,\, \mathbf{T}_S^{\beta_l}\right) . \tag{6.2}$$

In this tuple, $\mathcal{I} \in \{\mathcal{G}, \mathcal{A}\}$ specifies the type of the sensor with measurement noise standard deviation σ, measurement range $\pm y_{\max}$, and mounting offset $\mathbf{T}_S^{\beta_l}$ at link $l \leq n$. In the following, a set of $k \geq 0$ inertial sensors is denoted as sensor configuration

$$\mathcal{C} = \{\mathcal{S}_1, \ldots, \mathcal{S}_k\} .$$

With this notation, the goal of finding the optimal sensor configuration $\mathcal{C}^*$ can be written as

$$\mathcal{C}^* = \underset{\mathcal{C} \in C_m}{\operatorname{argmin}}\, J(\mathcal{C}) \ , \tag{6.3}$$

where C_m is the set of all feasible sensor configurations with up to m sensors. The objective function J quantifies the achievable estimation quality with respect to joint position, velocity, and acceleration and is defined successively in the next section. Furthermore, the maximally expected inertial sensor axis readings $\omega^S_{WS,\max}$ and $a^S_{WS,\max}$ for a given sensor placement $(l,\ \mathbf{T}_S^{\beta_l})$ must be calculated as a function of the maximum absolute joint velocities $\underline{\dot{q}}_{\max}$ and accelerations $\underline{\ddot{q}}_{\max}$ in order to select sensor models with suitable measurement ranges $\pm y_{\max}$ and to exclude infeasible sensor configurations from C_m.

6.2.2 Observability-Based Placement Procedure

To solve the stated problems, we first derive a novel method for computing the configuration-dependent maximal axis readings of inertial sensors. Then, an observability measure is presented and applied to select optimal sensor configurations.

Gyroscope and Accelerometer Measurement Ranges

For the derivation of the maximal inertial sensor readings in configuration $\underline{q}$, we assume that the set of feasible joint velocities is an axis-parallel ellipsoid. In mathematical notation, this corresponds to

$$\left\{\underline{\dot{q}} = \mathbf{D}_{\dot{q}}\underline{\tilde{\dot{q}}} \,|\, \underline{\tilde{\dot{q}}} \in \mathbb{R}^n, \left\|\underline{\tilde{\dot{q}}}\right\|_2 \leq 1\right\} \quad \text{with} \quad \mathbf{D}_{\dot{q}} = \operatorname{diag}\left(\underline{\dot{q}}_{\max}\right). \tag{6.4}$$

Although this is an approximation, it is a reasonable assumption as not all joints reach peak velocity simultaneously in typical applications. Alternatively, the ellipsoid can be expanded to include all joint velocities of interest. Following the same scheme, the set of feasible joint accelerations is defined using the scaling matrix $\mathbf{D}_{\ddot{q}} = \operatorname{diag}\left(\underline{\ddot{q}}_{\max}\right)$.

Based on (2.9) and (6.4), the gyroscope readings on axis i at frame S relative to the base link reference frame W are

$$\underline{\omega}^S_{WS[i]} = \left(\mathbf{C}^S_W \boldsymbol{\mathcal{J}}_{\underline{\omega}^W_{WS}} \mathbf{D}_{\dot{q}}\right)_{[i,:]} \underline{\tilde{\dot{q}}}$$

with the corresponding maximal absolute value

$$\omega^S_{WS,\max[i]} = \max_{\underline{\dot{q}},\underline{\ddot{q}}} \left|\underline{\omega}^S_{WS[i]}\right| = \left\|\left(\mathbf{C}^S_W \boldsymbol{\mathcal{J}}_{\underline{\omega}^W_{WS}} \mathbf{D}_{\dot{q}}\right)_{[i,:]}\right\|_2 \geq \left\|\left(\mathbf{C}^S_W \boldsymbol{\mathcal{J}}_{\underline{\omega}^W_{WS}} \mathbf{D}_{\dot{q}}\right)_{[i,:]}\right\|_2 \left\|\underline{\tilde{\dot{q}}}\right\|_2 \geq \left\|\underline{\omega}^S_{WS[i]}\right\|_2, \tag{6.5}$$

which follows from $\left\|\underline{\tilde{\dot{q}}}\right\|_2 \leq 1$ and the submultiplicativity of the spectral norm [147]. Note, that this value is a tight upper bound. Combining this over all three axes yields the overall maximal gyroscope reading at frame S, which is

$$\omega^S_{WS,\max} = \left\|\underline{\omega}^S_{WS,\max}\right\|_\infty . \tag{6.6}$$

The derivation of the maximal readings for an accelerometer at frame S in configuration $\underline{q}$ is a bit more involved. We start with the linear part of the acceleration according to (2.11). After adding the effect of gravity $\underline{g}^W$ and changing the resolving frame from W to S, we obtain the element-wise sensed acceleration

$$\underline{a}^S_{WS[i]} = \underbrace{\left(\mathbf{C}^S_W \boldsymbol{\mathcal{J}}_{\underline{v}^W_{WS}}\right)_{[i,:]} \underline{\ddot{q}}}_{=\underline{\beta}_{[i]}} + \underbrace{\underline{\dot{q}}^\mathsf{T} \mathbf{H}_i \underline{\dot{q}}}_{=\underline{\gamma}_{[i]}} + \underbrace{\mathbf{C}^S_{W[i,:]} \underline{g}^W}_{=\underline{\delta}_{[i]}} \quad \text{with} \quad \mathbf{H}_i = \sum_{j=1}^{3} \mathbf{C}^S_{W[i,j]} \boldsymbol{\mathcal{H}}_j$$

by applying (2.12) and (2.13). In this equation, the value of the $\underline{\delta}$ term with respect to $\underline{\dot{q}}$ and $\underline{\ddot{q}}$ is fixed as it only depends on $\underline{q}$. The element-wise maximum absolute value of the $\underline{\beta}$ term is calculated using

$$\underline{\beta}_{\max[i]} = \max_{\underline{\dot{q}},\underline{\ddot{q}}} \left|\underline{\beta}_{[i]}\right| = \left\|\left(\mathbf{C}^S_W \boldsymbol{\mathcal{J}}_{\underline{v}^W_{WS}} \mathbf{D}_{\ddot{q}}\right)_{[i,:]}\right\|_2$$

with the same derivation as in (6.5). Using the scaling matrix $\mathbf{D}_{\dot{q}}$ from (6.4), the elements of the quadratic term $\underline{\gamma}$ can be rewritten as

$$\underline{\gamma}_{[i]} = \underline{\dot{q}}^{\mathsf{T}} \mathbf{H}_i \underline{\dot{q}} = \underline{\dot{\tilde{q}}}^{\mathsf{T}} \tilde{\mathbf{H}}_i \underline{\dot{\tilde{q}}} \quad \text{with} \quad \tilde{\mathbf{H}}_i = \mathbf{D}_{\dot{q}}^{\mathsf{T}} \mathbf{H}_i \mathbf{D}_{\dot{q}} \,.$$

The matrix $\tilde{\mathbf{H}}_i$ is symmetric and therefore has real eigenvalues, which can be ordered as $\lambda_1, \ldots, \lambda_n$. Thus, the *Rayleigh* theorem [147] can be applied, yielding the inequality

$$\lambda_1 \underline{\dot{\tilde{q}}}^{\mathsf{T}} \underline{\dot{\tilde{q}}} \leq \underline{\gamma}_{[i]} \leq \lambda_n \underline{\dot{\tilde{q}}}^{\mathsf{T}} \underline{\dot{\tilde{q}}} \,.$$

Remembering that $0 \leq \|\underline{\dot{\tilde{q}}}\|_2^2 = \underline{\dot{\tilde{q}}}^{\mathsf{T}} \underline{\dot{\tilde{q}}} \leq 1$, a case distinction can be made according to the definiteness of $\tilde{\mathbf{H}}_i$. As a result, the element-wise upper and lower bounds of the $\underline{\gamma}$ term are

$$\underline{\gamma}_{\min[i]} = \begin{cases} 0\,, & \text{if } \tilde{\mathbf{H}}_i \text{ is positive definite} \\ \lambda_1\,, & \text{otherwise} \end{cases} \quad \text{and} \quad \underline{\gamma}_{\max[i]} = \begin{cases} 0\,, & \text{if } \tilde{\mathbf{H}}_i \text{ is negative definite} \\ \lambda_n\,, & \text{otherwise} \end{cases} .$$

These bounds are also tight, because the feasible values of $\underline{\dot{\tilde{q}}}$ include the eigenvectors of $\tilde{\mathbf{H}}_i$ and $\underline{0}$. Superimposing the derived bounds and the fixed value for $\underline{\delta}$, while taking into account the symmetry of $\underline{\beta}$, yields the axis-wise maximal accelerometer reading

$$\underline{a}_{WS,\max[i]}^{S} = \max_{\underline{\dot{q}},\underline{\ddot{q}}} \left| \underline{a}_{WS[i]}^{S} \right| = \underline{\beta}_{\max[i]} + \max\left(|\underline{\gamma}_{\min[i]} + \underline{\delta}_{[i]}|, |\underline{\gamma}_{\max[i]} + \underline{\delta}_{[i]}| \right) ,$$

which is combined to the overall maximal accelerometer reading

$$a_{WS,\max}^{S} = \left\| \underline{a}_{WS,\max}^{S} \right\|_{\infty} , \tag{6.7}$$

analogous to equation (6.6).

Quantitative Observability Measure

The classical concept of observability from control theory [142] is not useful here because it does not quantify *how well* a system is observable. As proposed in [143, 144], this problem can be circumvented by using the *observability Gramian* [148]. To apply this method, we define the kinematic state

$$\underline{x}^{\mathsf{T}} = \begin{pmatrix} \underline{q}^{\mathsf{T}} & \underline{\dot{q}}^{\mathsf{T}} & \underline{\ddot{q}}^{\mathsf{T}} \end{pmatrix} \in \mathbb{R}^{1 \times 3n} \,,$$

whose dynamic behavior is described by the deterministic constant acceleration model

$$\underline{\dot{x}} = \mathbf{A}\underline{x} = \begin{pmatrix} \mathbf{0}_{n\times n} & \mathbf{I}_{n\times n} & \mathbf{0}_{n\times n} \\ \mathbf{0}_{n\times n} & \mathbf{0}_{n\times n} & \mathbf{I}_{n\times n} \\ \mathbf{0}_{n\times n} & \mathbf{0}_{n\times n} & \mathbf{0}_{n\times n} \end{pmatrix} \underline{x} \,. \tag{6.8}$$

As we will see later, this is a reasonable approximation for the real system if short time horizons are considered. The corresponding measurement model depends on the sensor configuration $\mathcal{C}$ and is defined as

$$\underline{\tilde{y}}(\underline{x}, \mathcal{C}) = \underline{y}(\underline{x}, \mathcal{C}) + \underline{v}(\mathcal{C}) = \begin{pmatrix} \underline{q} \\ \underline{\omega}_{WG_1}^{G_1}(\underline{q}, \dot{\underline{q}}) \\ \vdots \\ \underline{\omega}_{WG_{n_\mathrm{g}}}^{G_{n_\mathrm{g}}}(\underline{q}, \dot{\underline{q}}) \\ \underline{a}_{WA_1}^{A_1}(\underline{q}, \dot{\underline{q}}, \ddot{\underline{q}}) \\ \vdots \\ \underline{a}_{WA_{n_\mathrm{a}}}^{A_{n_\mathrm{a}}}(\underline{q}, \dot{\underline{q}}, \ddot{\underline{q}}) \end{pmatrix} + \underline{v}(\mathcal{C}) \in \mathbb{R}^{n+3(n_\mathrm{g}+n_\mathrm{a})} \tag{6.9}$$

with the same quantities as in the state estimation measurement model (6.1). As in Section 6.1.2, the sensor noise covariance matrix $\mathbf{R}_\mathcal{C} = \mathrm{Cov}(\underline{v}(\mathcal{C}))$ is populated with the individual sensor noise variances on the diagonal.

If this system model is time-discretized with sample time Δt, the observability Gramian $\mathbf{G}$ for non-linear time-discrete systems [148] with time horizon M can be calculated using

$$\mathbf{G}(\mathcal{C}, \mathcal{T}) = \sum_{k=0}^{M} (\mathbf{\Phi}^k)^\mathsf{T} \mathbf{J}^\mathsf{T}(\hat{\underline{x}}_k, \mathcal{C})\, \mathbf{R}_\mathcal{C}^{-1} \mathbf{J}(\hat{\underline{x}}_k, \mathcal{C})\, \mathbf{\Phi}^k\,. \tag{6.10}$$

In this expression, $\mathbf{\Phi} = \exp(\Delta t \mathbf{A})$ is the constant state transition matrix and $\mathbf{J}(\underline{x}, \mathcal{C}) = {}^{\partial \underline{\tilde{y}}(\underline{x}, \mathcal{C})}/_{\partial \underline{x}}$ is the Jacobian used to linearize the measurement function, whose entries can be calculated as in Section 6.1.2 using Theorems A.2 to A.4. The reference joint trajectory $\mathcal{T}$ provides the linearization points $\hat{\underline{x}}_k, k = 0, \ldots, M$. Given the measurements $\underline{\tilde{y}}(\underline{x}_0, \mathcal{C}), \ldots, \underline{\tilde{y}}(\underline{x}_M, \mathcal{C})$, the model defined by (6.8) and (6.9) can be used with these linearization points to formulate the linear least squares problem to estimate the initial state $\underline{x}_0^*$. In this case, the covariance of the estimate $\mathrm{Cov}(\underline{x}_0^*) = \mathbf{G}^{-1}(\mathcal{C}, \mathcal{T})$ is equal to the inverse of the observability Gramian [143, 149]. Due to the presence of an encoder at each joint, the system is always fully observable for $M \geq 2$ and therefore $\mathbf{G}$ is always invertible, meaning that $\mathbf{G}$ can be used to quantify the *estimation uncertainty* for any sensor configuration.

However, the observability Gramian or its inverse cannot be plugged directly into (6.3) because the objective function J is scalar. To overcome this, the diagonal elements of $\mathbf{G}^{-1}$, which represent the expected mean square errors due to its interpretability as a covariance matrix, can be considered. With this knowledge,

- the velocity observability $h_{\dot{q}}(\mathcal{C}, \mathcal{T}) = \mathrm{tr}\big(\mathbf{G}^{-1}(\mathcal{C}, \mathcal{T})_{[n+1:2n, n+1:2n]}\big)$ and
- the acceleration observability $h_{\ddot{q}}(\mathcal{C}, \mathcal{T}) = \mathrm{tr}\big(\mathbf{G}^{-1}(\mathcal{C}, \mathcal{T})_{[2n+1:, 2n+1:]}\big)$

can be defined as scalar observability measures. Theoretically, the position observability can be formulated as another objective function, but this does not gain meaningful insights because the joint position is already known accurately from the encoders. To evaluate the observability Gramian, we also need the reference trajectory $\mathcal{T}$. Without specifying a particular trajectory, we assume that the objective function J averages over a set of equally weighted trajectories T. Thus, the objective to be minimized becomes

$$J(\mathcal{C}, h) = \frac{1}{|T|} \sum_{\mathcal{T} \in T} h(\mathcal{C}, \mathcal{T}) \tag{6.11}$$

and depends on the choice of the observability measure $h \in \{h_{\dot{q}}, h_{\ddot{q}}\}$.

To actually use this function for the optimal placement of sensors, some assumptions and simplifications have to be made:

1. In the original definition, the sum in the observability Gramian (6.10) comes with an infinite time horizon $M \to \infty$. Besides being computationally infeasible, high values for M definitely violate the constant acceleration assumption in (6.8). On the other hand, three encoder samples are sufficient to obtain an estimate of $\underline{x}_0^*$ without additional sensors. Therefore, $M = 2$ is selected. With this choice, the objective function (6.11) describes the achievable mean square error of the joint velocities or accelerations, when three samples are used to reconstruct $\underline{x}_0^*$.
2. The set of reference trajectories T must be defined, even for short time horizons. If certain reference trajectories are available, they can be used for T. Usually, this information is not available. In this case, the individual reference trajectory $\mathcal{T}$ with its samples $\hat{\underline{x}}_k$ can be set to a constant value $\hat{\underline{x}}_0$, which is a reasonable approximation in combination with the short time horizon. As a consequence, (6.11) collapses to a sum over kinematic states that can be derived

Algorithm 6.1 The proposed sensor placement procedure.

Input: Maximal number of sensors m
Output: Optimal sensor configuration $\mathcal{C}^*$

1: Identify a set of suitable mounting poses for the inertial sensors.
2: Select a set of reference trajectories T or a set of typical joint configurations that can be used as constant reference trajectories.
3: Calculate the expected maximal gyroscope and accelerometer readings, $\omega^S_{WS,\max}$ and $a^S_{WS,\max}$, for each mounting pose over all relevant joint configurations according to (6.6) and (6.7).
4: Based on the maximal readings, select a suitable gyroscope and accelerometer model for each mounting pose.
5: Generate the set of feasible sensor configurations C_m.
6: Solve (6.3) with the objective function (6.11) to obtain $\mathcal{C}^*$ by performing an exhaustive search over all elements in C_m.
7: **return** $\mathcal{C}^*$

either from situations of interest or by sampling the combined joint position-velocity-acceleration space with dimension $3n$. For manipulators with a larger number of joints, the latter requires a very large number of samples, which may become computationally intractable. A further simplification to address this issue is the limitation of the sampling to the joint position space of dimension n, or to use a set of configurations of interest, while setting the joint velocities and accelerations to zero. Although this simplification is relatively crude, it still yields meaningful results, as we will see in the next section.

3. The placement problem (6.3) is a non-linear mixed-integer problem due to the assignment of each sensor to one of the links and the inversion of the observability Gramian $\mathbf{G}$, for which finding a global optimum is hard. However, the problem can be greatly simplified by assuming that all inertial sensors can only be placed at certain discrete positions on the robot, which is actually a realistic assumption for real robots with a limited number of installation options. Mathematically this means, that the sensor placement $(l, \mathbf{T}_S^{\beta_l})$ in (6.2) is constrained to p predefined locations that may contain either a gyroscope, an accelerometer, or both. The number of feasible sensor configurations with k inertial sensors is then $\binom{2p}{k}$. This results in

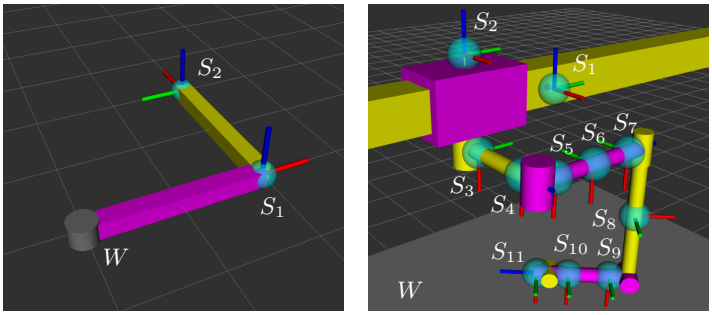
$$|C_m| = \sum_{k=0}^{\min(2p,m)} \binom{2p}{k} \tag{6.12}$$

feasible configurations with up to m sensors. For a typical robot with six joints ($n = 6$), three possible sensor locations per link ($p = 3n = 18$), and up to twelve inertial sensors ($m = 12$), the resulting number of configurations is $2.24 \cdot 10^9$ and therefore still computationally tractable. This means, that the optimization problem can be transformed into an exhaustive, combinatorial search over all feasible sensor configurations.

Now, all the components are in place to determine the measurement range of the inertial sensors and to rate a given sensor configuration in terms of joint velocity and acceleration estimation quality. Algorithm 6.1 summarizes the necessary steps of the proposed sensor placement procedure.

6.2.3 Application and Evaluation

The proposed sensor placement method is applied to a SCARA manipulator to illustrate and validate the methodology. Subsequently, the inertial sensor placement for HapticGiant is optimized.



(a) SCARA manipulator. **(b)** HapticGiant.

Figure 6.8: Manipulators used to evaluate the proposed sensor placement procedure. The feasible inertial sensor poses are marked with blue spheres. Taken from [O3].

SCARA Manipulator

For this part of the evaluation, we will cover the steps of Algorithm 6.1 according to the line numbers:

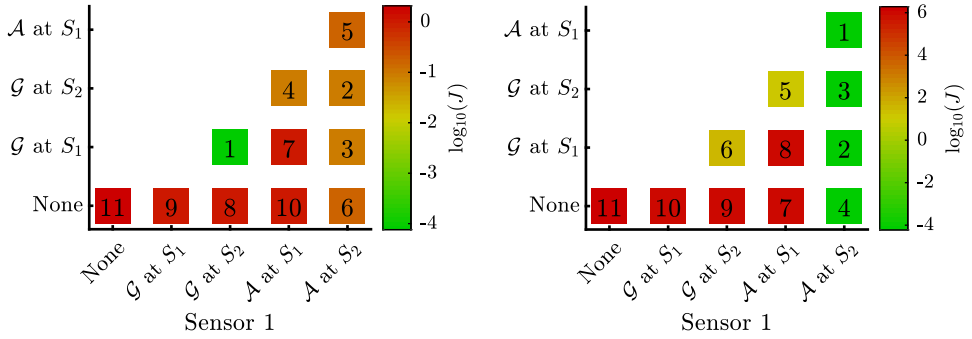
1. We consider the planar SCARA manipulator from Figure 6.8a with two joints and two mounting poses for the inertial sensors. Both links have a length of $1\,\mathrm{m}$.
2. For comprehensibility, the configuration shown in Figure 6.8a represents the only, constant reference trajectory in T.
3. Based on the maximal joint velocities and accelerations, which are $\dot{\underline{q}}_{\max}^{\mathsf{T}} = \begin{pmatrix} 180 & 180 \end{pmatrix} ^\circ/\mathrm{s}$ and $\ddot{\underline{q}}_{\max}^{\mathsf{T}} = \begin{pmatrix} 720 & 720 \end{pmatrix} ^\circ/\mathrm{s}^2$, the maximal gyroscope and accelerometer readings from the proposed method are $\omega_{WS_1,\max}^{S_1} = 180\,^\circ/\mathrm{s}$, $\omega_{WS_2,\max}^{S_2} = 255\,^\circ/\mathrm{s}$, $a_{WS_1,\max}^{S_1} = 12.6\,\mathrm{m/s^2}$, and $a_{WS_2,\max}^{S_2} = 32.3\,\mathrm{m/s^2}$. This can be manually verified using the measurement equations

$$\underline{\omega}_{WS_1}^{S_1} = \begin{pmatrix} 0 \\ 0 \\ \dot{\underline{q}}_{[1]} \end{pmatrix}, \qquad \underline{a}_{WS_1}^{S_1} = \begin{pmatrix} -\dot{\underline{q}}_{[1]}^2\ \mathrm{m} \\ \ddot{\underline{q}}_{[1]}\ \mathrm{m} \\ 9.81\,\mathrm{m/s^2} \end{pmatrix},$$

$$\underline{\omega}_{WS_2}^{S_2} = \begin{pmatrix} 0 \\ 0 \\ \dot{\underline{q}}_{[1]} + \dot{\underline{q}}_{[2]} \end{pmatrix}, \text{ and } \qquad \underline{a}_{WS_2}^{S_2} = \begin{pmatrix} (\ddot{\underline{q}}_{[1]} - (\dot{\underline{q}}_{[1]} + \dot{\underline{q}}_{[2]})^2)\ \mathrm{m} \\ (\dot{\underline{q}}_{[1]}^2 + \ddot{\underline{q}}_{[1]} + \ddot{\underline{q}}_{[2]})\ \mathrm{m} \\ 9.81\,\mathrm{m/s^2} \end{pmatrix}.$$

   Note, that the definition of the feasible joint velocity and acceleration ellipsoid, as in (6.4), must be taken into account to obtain correct results.
4. With these values, we can see that the gyroscope and the accelerometer of an *Invensense MPU-9250* [D11] are suitable sensors. Their noise parameters are extracted from the datasheet.
5. Each of the two mounting poses can accommodate no sensor, a gyroscope, an accelerometer, or both. A maximum of two sensors can be used overall. According to (6.12), this results in $|C_2| = 11$.
6. The resulting velocity and acceleration observability scores are visualized in Figure 6.9.
7. If $h = h_{\ddot{q}}$, the best configuration is the one with an accelerometer at S_1 and S_2.

A closer look at Figure 6.9a reveals that the dual gyroscope configuration yields the highest velocity observability score. This result is expected because the two joints with aligned rotation axes require measurements at both links to separate the joint velocities in the measured angular velocities. Just adding accelerometers to the instrumentation does not significantly improve the velocity observability score. This is due to the fact that the velocity cannot be determined from integrating the acceleration data alone when the initial velocity is unknown, as it is the case for short time horizons. Furthermore, the proposed zero-velocity linearization removes the influence of the joint velocity from the



(a) Velocity observability $h = h_{\dot{q}}(\mathcal{C}, \mathcal{T})$. **(b)** Acceleration observability $h = h_{\ddot{q}}(\mathcal{C}, \mathcal{T})$.

Figure 6.9: Observability scores for all sensor configurations with up to two sensors on the SCARA manipulator. The numbers in the colored squares rank the sensor configurations based on their observability scores. Adapted from [O3].

accelerometer signals. As seen in Figure 6.9b, the acceleration observability score is high when an accelerometer is placed at the end effector. This results from the fact that the accelerations of the individual joints produce Cartesian accelerations with different directions at the end effector for the given SCARA configuration. Interestingly, adding more sensors does not significantly improve the acceleration observability. Without the accelerometer at the end effector, configurations with two gyroscopes or with a gyroscope at the end effector and an accelerometer at the first link are the best choices. However, these perform much worse because of the required numerical differentiation.

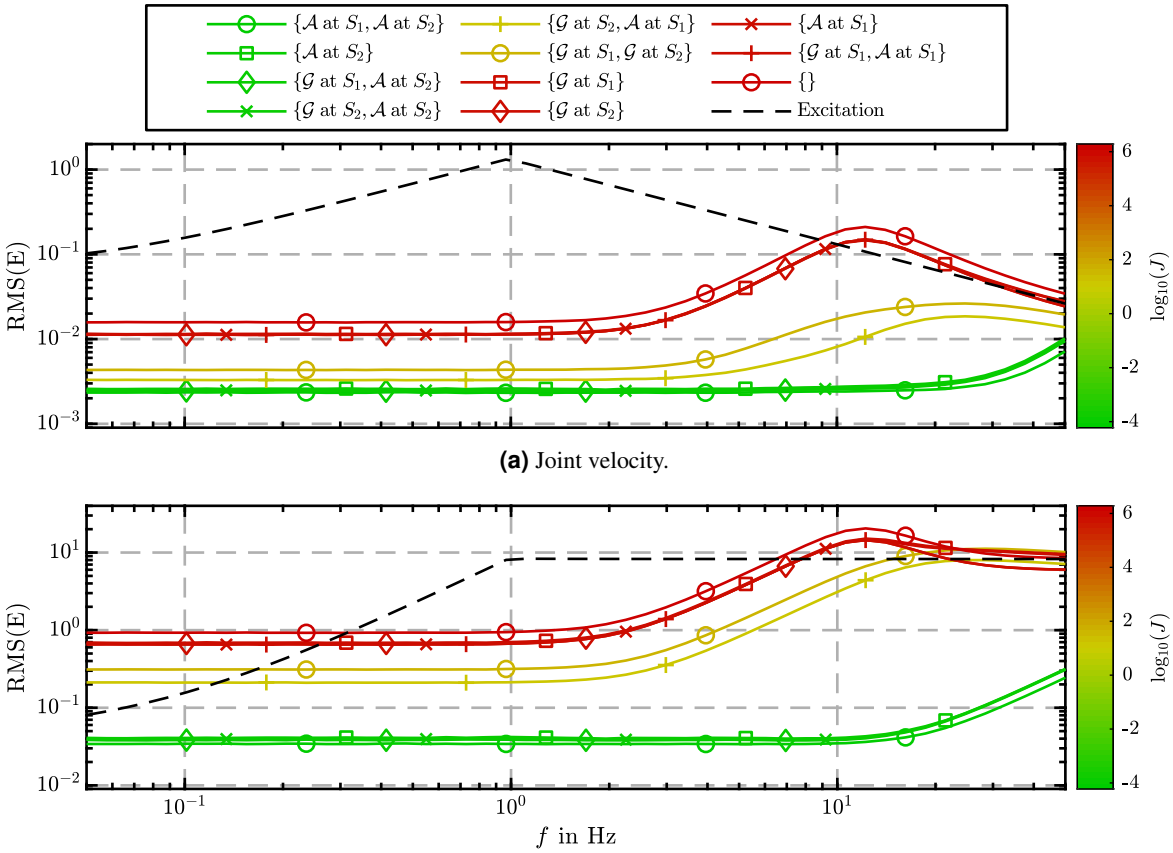
Next, the results are validated by running the proposed kinematic state estimator from Section 6.1 with the examined sensor configurations. For this purpose, the same setup as in Section 6.1.3 is used, except that placement errors, scale-factor errors, and cross-axis sensitivity are disabled to avoid perturbing the results. The resulting RMSEs of the estimated joint velocities and accelerations are depicted in Figure 6.10. A detailed comparison between the ranking in Figure 6.9b and the estimation error over frequency indeed proves that there is a strong correlation between the *acceleration observability* $h_{\ddot{q}}$ and the achievable estimator performance. Interestingly, the acceleration observability also correlates with the velocity estimation error. Although this is unexpected, it can be explained by the memory of the state estimator's underlying KF, which enables the integration of estimated accelerations to obtain updated velocity estimates. Beyond that, these results show that the approximations made in Section 6.2.2 do not limit the ability of the proposed method to predict the estimation quality of a given sensor configuration when the acceleration observability metric is used.

HapticGiant

HapticGiant with the kinematic structure from Figure 3.2 and the potential sensor locations from Figure 6.8b is too complex for an efficient manual sensor placement. For this reason, the proposed placement procedure is applied to determine a justified and economically efficient sensor configuration. To cover all operational states that occur during haptic rendering, the configuration from Figure 6.8b is augmented with 323 additional configurations. These include variations of the manipulator joint angle $\underline{q}^{\mathrm{m}}_{[1]}$ between 0° and 135° in 45° steps. The angles $\underline{q}^{\mathrm{m}}_{[2]}$ to $\underline{q}^{\mathrm{m}}_{[5]}$ are varied between −22.5° and 22.5° with 22.5° increments. The remaining joints are position invariant and therefore fixed. The joint limits

$$\underline{\dot{q}}^{\mathrm{pT}}_{\mathrm{max}} = \begin{pmatrix} 1.5 & 1.5 \end{pmatrix} \mathrm{m/s}\,, \qquad \underline{\dot{q}}^{\mathrm{mT}}_{\mathrm{max}} = \begin{pmatrix} 342 & 342 & 450 & 450 & 804 & 804 \end{pmatrix} {}^{\circ}/\mathrm{s}\,,$$
$$\underline{\ddot{q}}^{\mathrm{pT}}_{\mathrm{max}} = \begin{pmatrix} 6.0 & 6.0 \end{pmatrix} \mathrm{m/s^2}\,, \text{ and} \qquad \underline{\ddot{q}}^{\mathrm{mT}}_{\mathrm{max}} = \begin{pmatrix} 720 & 720 & 720 & 720 & 720 & 720 \end{pmatrix} {}^{\circ}/\mathrm{s^2}$$

are taken from the conception in Section 3.3 or determined from practical experience with the joints, if not specified in the datasheets.



(a) Joint velocity.

(b) Joint acceleration.

Figure 6.10: Estimation errors for different sensor configurations on the SCARA manipulator. The colors encode the observability measure J with $h = h_{\ddot{q}}$ for each configuration. Taken from [O3].

The resulting maximal inertial sensor readings are listed in Table 6.3 and proof that IMUs with measurement ranges of $\pm 1000\,^\circ/\mathrm{s}$ and $\pm 8 \cdot 9.81\,\mathrm{m/s^2}$ are sufficient. Based on this, the *Botasys SensONE* [D8] force-torque sensor with an integrated *Invensense MPU-9250* IMU [D11], the *Gable Systems SE1-HSD* IMU [D10] with an *XSENS MTi 1-series* [D16], and the *Beckhoff EP3752-0000* accelerometer [D9] with two *STM LIS3DH* [D17] can be identified as suitable sensor choices. All sensors have similar noise characteristics. For this reason, the standard deviations of the *Invensense MPU-9250* IMU are used for the following evaluation of the observability scores.

In total, $110\,056$ configurations were analyzed using the acceleration observability metric. Table 6.4 lists the best configuration for a given number of sensors. From this, we can see that adding three additional inertial sensors causes a substantial drop of the objective value, indicating significantly improved estimates. Adding a fourth sensor further improves the quality of the sensor configuration. However, this trend ends as more sensors are added. Even a sensor configuration with a gyroscope/accelerometer pair at the end of each accessible link, resulting in a total of 14 sensors, does not visibly outperform the configuration with four sensors. Again, these results can be verified using Figure 6.11, where the predicted estimation accuracies from the acceleration observability $h_{\ddot{q}}$ are reflected by the actual, simulated RMSEs for the joint velocities and accelerations.

With the knowledge gained, the best sensor configuration with four sensors according to $J(\mathcal{C}, h_{\ddot{q}})$ was selected for implementation on HapticGiant. Thus, the trolley of the PPU (S_2) is equipped

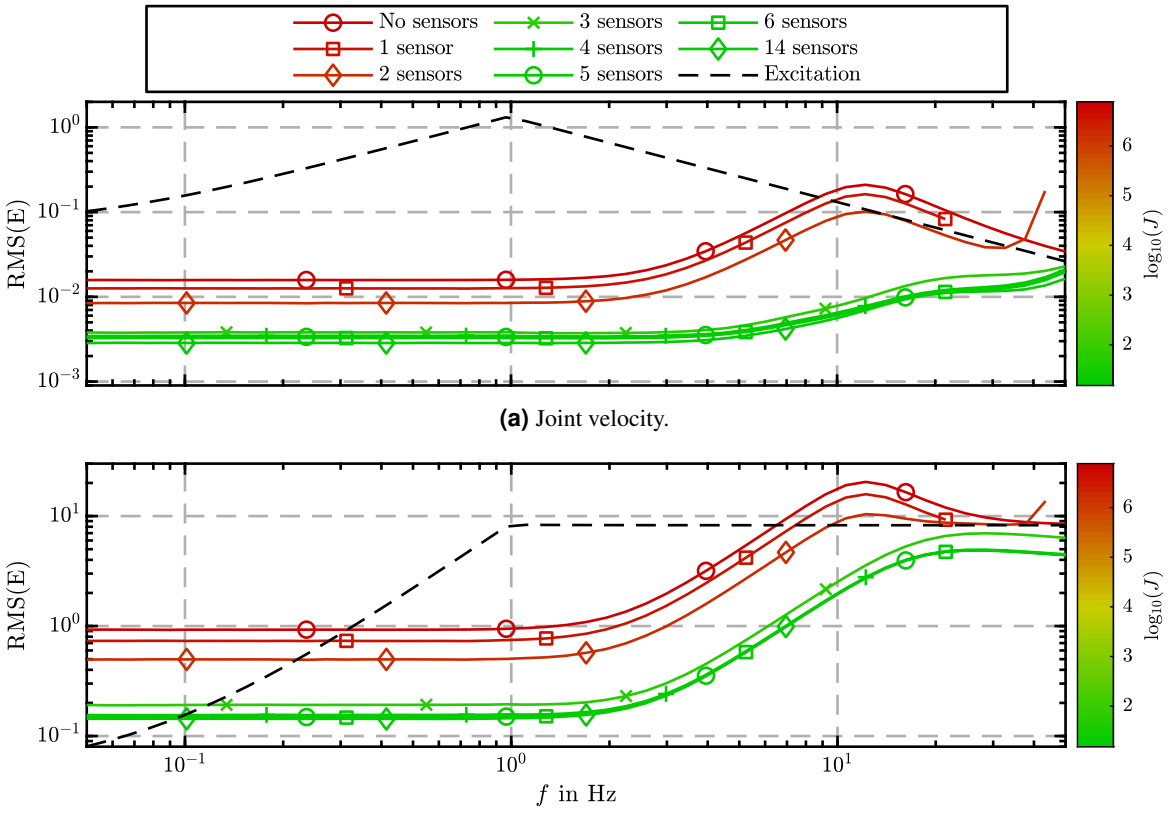
At frame	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}	S_{11}
$\omega^{S_i}_{WS_i,\max}$ in °/s	0	0	342	342	484	484	484	661	661	661	930
$a^{S_i}_{WS_i,\max}$ in m/s^2	9.8	9.8	9.8	21.3	25.1	36.5	53.0	59.1	53.7	53.5	65.2

Table 6.3: Maximal inertial sensor readings for the studied mounting poses on HapticGiant. Adapted from [O3] with HapticGiant's specification.

Number of sensors	Sensor configuration $\mathcal{C}$	$\log_{10} J$
0	$\{\}$	6.89
1	$\{\mathcal{A}$ at $S_{11}\}$	6.66
2	$\{\mathcal{G}$ at $S_{11}, \mathcal{A}$ at $S_{10}\}$	6.28
3	$\{\mathcal{G}$ at $S_{11}, \mathcal{A}$ at $S_2, S_{10}\}$	1.69
4	$\{\mathcal{G}$ at $S_{11}, \mathcal{A}$ at $S_2, S_{10}, S_{11}\}$	1.18
5	$\{\mathcal{G}$ at $S_{11}, \mathcal{A}$ at $S_2, S_9, S_{10}, S_{11}\}$	1.18
6	$\{\mathcal{G}$ at $S_{11}, \mathcal{A}$ at $S_2, S_8, S_9, S_{10}, S_{11}\}$	1.18
14*	$\{\mathcal{G}$ at $S_1, S_2, S_4, S_7, S_8, S_{10}, S_{11}, \mathcal{A}$ at $S_1, S_2, S_4, S_7, S_8, S_{10}, S_{11}\}$	1.18

* Unoptimized configuration with 14 sensors.

Table 6.4: Best sensor configurations and their observability measures with $h = h_{\ddot{q}}$, depending on the number of inertial sensors. Adapted from [O3] with HapticGiant's specification.



(a) Joint velocity.

(b) Joint acceleration.

Figure 6.11: Estimation errors of the examined sensor configurations for HapticGiant. The color coding corresponds to scores from Table 6.4. Adapted from [O3] with HapticGiant's specification.

with a Beckhoff EP3752-0000 accelerometer. For the end effector (S_{11}), the gyroscope and the accelerometer in the Botasys SensONE force-torque sensor are utilized. Due to mechanical constraints, the acceleration measurement at the wrist (S_{10}) is obtained using a Gable Systems SE1-HSD IMU, which contains an excess gyroscope. This setup corresponds exactly to the instrumentation described in Section 4.1.1.

6.3 Discussion

In this chapter, a novel, inertial sensor-based approach for the kinematic state estimation of serial manipulators under real-time constraints was presented. In contrast to state-of-the-art algorithms, the proposed estimator does not rely on specific sensor configurations or on accurate sensor calibrations. To achieve this, a generalized kinematic model with states for the sensor biases and calibration errors is combined with an EKF. The evaluation of the proposed estimator in Section 6.1.3 demonstrates superior performance in simulation compared to competing methods and emphasizes the importance of inertial sensors for the purpose of kinematic state estimation. It also highlights that a basic sensor calibration can improve the estimation accuracy, although the proposed algorithm can handle uncalibrated sensors.

The absence of sensor configuration constraints raises the question of where sensors should be placed to obtain the most accurate state estimates with as few sensors as possible. This question is answered by providing a novel, observability-based sensor placement procedure that also takes into account the expected maximal sensor readings and does not require any knowledge about the actual estimator. Experiments show that the predicted accuracies of the resulting sensor configurations are in good agreement with the achievable estimator accuracies. As expected, configurations with many sensors are not always advantageous for achieving lower estimation errors. Furthermore, the results in simple cases are consistent with the design engineer's intuition, which demands that the movement of different joint axes ideally causes orthogonal sensor excitations and that no integration or differentiation of the sensor signals is required. This makes the proposed placement procedure a justified tool for both simple and complex manipulators.

Regarding HapticGiant, the placement procedure was successfully applied to determine suitable inertial sensors and their placement. Unfortunately, the implementation of the proposed estimator did not succeed due to a model mismatch as explained in Section 6.1.4. Despite this setback, the experiments with the SCARA manipulator indicate that the proposed estimator has the potential to improve not only kinesthetic haptic interfaces, but also any kind of robotic manipulator. Nevertheless, more effort needs to be put into the analysis of the root cause beforehand.

Beyond the mentioned model mismatch, there are more topics for further research. Currently, all calibration errors are collected in a single bias term. This could be replaced by an online calibration method with more parameters. Furthermore, computational optimizations that facilitate the use of more powerful estimator backends, such as the smart sampling Kalman filter (S2KF) [150], seem promising. Regarding the sensor placement, the approximations from Section 6.2.2 and their impact on the placement results should be further investigated, especially the choice of the reference trajectories. Currently, all joint configurations are considered at once. However, finding the worst and best joint configurations for a given sensor setup could also provide useful insights for design engineers.

Although the goal of creating a high-end kinematic state estimator for HapticGiant was ultimately missed in this chapter, system development was not halted. Instead, it was decided to use less powerful control concepts, which are functional with velocity feedback from numerical differentiation, for the low-level joint control. This decision is supported by the fact that there are still many open research questions when it comes to the high-level part of admittance control for a non-idealized kinesthetic haptic interface, as we will see in the next chapter.

CHAPTER 7

Hierarchical Force Control

Contents

> The senses, being the explorers of the world, open the way to knowledge.
>
> MARIA MONTESSORI (1870 – 1952)

As motivated in Chapter 1, the goal of this thesis is to create a digital twin (DT) of a serial kinematic chain. To achieve this goal, the mechanical properties of the kinematic chain of interest must be rendered convincingly. This means that the haptic representation of the DT must behave *dynamically plausibly*, especially regarding inertia, joint limits, and friction. Plausibly here means that a behavior matching the expectations of the human user is sufficient. Exact replicas are not necessary and sometimes undesirable or even impossible due to the real physical weight of the rendered kinematic chain or unknown data. For example, an ordinary construction machine is very heavy, and manufacturers do not provide comprehensive information about the exact mass distribution. In addition, an appropriate force control algorithm must take into account the inherent constraints of the haptic interface, including kinematic limitations and safety aspects required for reliable and smooth operation.

The question of how to control kinesthetic haptic devices dates back to the invention of the first electromechanical teleoperation devices in the 1950s [151]. Since then, a vast number of force control schemes have been proposed for industrial manipulators and haptic devices alike [152–154]. As a result, the terminology used by different authors is inconsistent, and a complete overview

would exceed the scope of this thesis. However, there is a common, less intimidating taxonomy that distinguishes between impedance and admittance control schemes [19, 155, 156]. In impedance control, the environment or object to be rendered is described as a mechanical impedance, whose motion, such as deformation, results in a force. This force is then transformed into joint torque commands, optionally with additional terms for disturbance rejection. In admittance control, the environment is described by a mechanical admittance model that is the inverse of the aforementioned impedance. This means that a measured force is converted into a virtual reference motion, which is then tracked by the manipulator of the haptic interface using appropriate (low-level) motion controllers.

Typically, haptic rendering is used to display forces that occur during the interaction with surfaces of 3D volumes [157]. In this case, the impedance is represented by virtual springs generating a vector field. A popular library that implements this type of rendering and that comes with support for various surface rendering methods, such as the god-object method [158] or the virtual proxy method [159], is *CHAI3D* [160]. Despite its popularity, CHAI3D cannot be used for the presented goal for several reasons. First, CHAI3D relies on a scene description using geometric primitives, such as triangles, spheres, and planes. This Cartesian 3D representation is not compatible with serial kinematic chains with an arbitrary number of DOF. Second, its rendering algorithms are designed for impedance control schemes that rely on the penetration of constraints to produce force feedback, regardless of whether the formulation is in Cartesian or joint space. Due to the potentially long lever arms in serial kinematic chains, a penetration in joint space can result in significant, undesired end effector displacements. Lastly, HapticGiant's low backdriveability and high friction, originating from the joints with strain wave gearing and the belt-driven PPU, make the application of impedance control schemes challenging [19]. These observations also extend to similar libraries [161]. Consequently, a different approach based on admittance control is needed to render kinematic chains. Choosing an admittance control scheme over an impedance control scheme has the positive side effect that the resulting controller is more portable, because it is not tied to low-inertia, low-friction kinesthetic haptic interfaces [162].

The desired admittance control scheme must be capable of simultaneously handling the joint limits of the DT and HapticGiant. When no joint limit is active, the end effector motion must still be constrained to the DOF, the workspace size, and the dynamic properties of the DT. Conceptually, this corresponds to the idea of virtual fixtures that were first introduced in [163] and that can be used for guidance [164] or to avoid forbidden regions. As an example of the latter, [165] proposes three methods to constrain the range of motion of a teleoperated haptic paddle. However, these methods cannot be applied beyond teleoperation.

The Cartesian workspace of a manipulator with n DOF can be interpreted as an n-dimensional hypersurface. With this idea, the method in [166] can be applied to render the DT as parametric surface. Although this approach seems feasible, it cannot be applied to manipulators with more than two DOF or kinematic redundancies. Furthermore, the method does not consider joint limits, and it allows deviations from the surface due to its impedance-like nature. A similar idea utilizing a virtual fixture for spatial motion constraints under admittance control is proposed in [167]. Based on the measured end effector forces and the desired constraints, the velocity output of an admittance is modified to guide along lines and surfaces. A closed-loop compensation ensures that the deviation from the constraints is minimal. However, this idea is also limited to two DOF and does not consider joint limits.

Moving away from virtual fixtures, the mechanism simulation found in [162] is close to the subgoal of creating dynamically consistent behavior. For this simulation, the dynamic model of the DT manipulator, following (2.14), is propagated in time by solving the forward dynamics problem driven by an external end effector wrench. The resulting output trajectory corresponds to the behavior of a non-linear admittance and is then used as a setpoint for a motion-controlled industrial manipulator. An

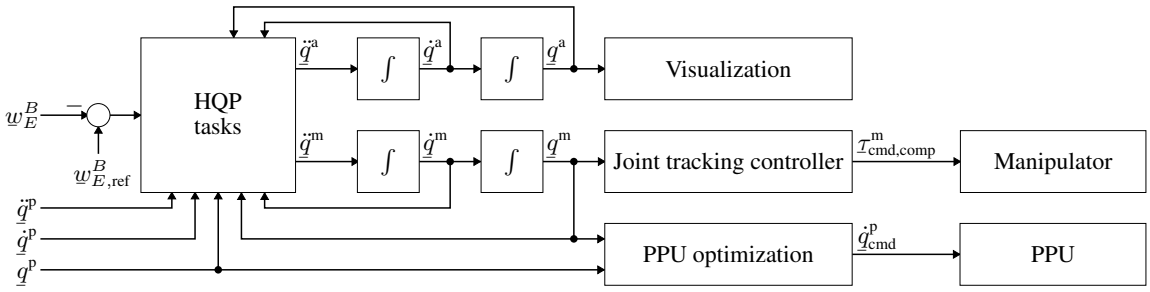


Figure 7.1: Overview of HapticGiant's force control architecture for the rendering of DTs. For simplicity, the integrators are drawn without the clamping described in Section 7.1.2. The frames are defined in Figure 7.3. Adapted from [O1].

extension of this can be found in [O5], where correcting joint torques and generalized impulses, based on the theory of rigid body contacts [40], are used to render infinitely stiff joint limits. However, the rendering of the joint limits is only feasible for the DT. As a result, the motion of the haptic interface is unconstrained, which either poses a safety risk or causes dynamically inconsistent DT behavior, when constraints of the haptic interface are encountered.

In general, a complex kinesthetic haptic device leads to many, sometimes conflicting objectives and constraints. This problem also arises in the context of humanoid [42, 168] or quadruped robots [41], where hierarchical quadratic programming (HQP), as presented in Section 2.4, is used to solve the underlying control problem in an elegant yet effective way. To apply HQP, the objectives are formulated as prioritized hierarchical tasks with equality and inequality constraints. In this way, mission-critical tasks, such as machine safety, will be always respected, while other less important tasks, such as trajectory tracking, are fulfilled only if possible. A good application example in the area of haptics is the *ANYexo* exoskeleton [48, 126], where HQP is used under real-time constraints to create a controller for haptic rehabilitation. According to the authors, the formulation respects joint and task space limits, presumably by applying the formulation from [41]. However, few details are provided, and the rendering of arbitrary non-linear admittances in a dynamically consistent manner is not addressed.

Nevertheless, the idea of HQP is very promising to solve the complex force control problem that arises in the context of HapticGiant and the rendering of DTs. Therefore, this chapter presents a novel, hierarchical force control scheme with the block diagram from Figure 7.1. By using HQP, the task-based formulation in Section 7.1 can handle joint limits of the DT, joint limits of the haptic interface, Cartesian workspace limits, and singularities, while rendering a dynamically consistent version of the DT. For the practical implementation of this admittance control scheme, joint tracking controllers are required. These are briefly explained in Section 7.2. Finally, Section 7.3 presents a thorough qualitative and quantitative evaluation of the proposed force control scheme based on real-world experiments with HapticGiant.

This chapter is based on results presented in the author's publications [O1, O5].

7.1 Hierarchical Controller Formulation

In the following, a novel force control scheme for the haptic rendering of arbitrary serial kinematic chains is presented. In preparation for this, the handling of joint limits is explained in Section 7.1.1. Subsequently, the control tasks are formulated in Section 7.1.2. The integration of HapticGiant's PPU is presented in Section 7.1.3.

7.1.1 Handling of Joint Limits

There are several methods for enforcing the limits of robot joints. In particular, the following theorems, extracted from [169] and [41], respectively, are used for the formulation of the control scheme in the next section.

Theorem 7.1:
Let q and $\dot{q}$ be the position and velocity of a joint with the acceleration setpoint $\ddot{q}$, which is updated with the sampling time Δt and has perfect acceleration tracking. The joint limits regarding position, velocity, and acceleration are defined as $q_{\min} \leq q \leq q_{\max}$, $-\dot{q}_{\max} \leq \dot{q} \leq +\dot{q}_{\max}$, and $-\ddot{q}_{\max} \leq \ddot{q} \leq +\ddot{q}_{\max}$, respectively. In this case, clamping $\ddot{q}$ to the intersection of the intervals

$$\left.\begin{aligned}
[\ddot{q}_{\text{lb},1}, \ddot{q}_{\text{ub},1}] &= \text{accBoundsFromPosLimits}(q, \dot{q})\,, \\
[\ddot{q}_{\text{lb},2}, \ddot{q}_{\text{ub},2}] &= {}^{1}\!/_{\Delta t}[(-\dot{q}_{\max} - \dot{q}), (\dot{q}_{\max} - \dot{q})]\,, \\
[\ddot{q}_{\text{lb},3}, \ddot{q}_{\text{ub},3}] &= \text{accBoundsFromViability}(q, \dot{q})\,, \text{and} \\
[\ddot{q}_{\text{lb},4}, \ddot{q}_{\text{ub},4}] &= [-\ddot{q}_{\max}, \ddot{q}_{\max}]
\end{aligned}\right\}\begin{matrix}\text{A}\\ \\ \\ \\ \end{matrix}\Bigg\}\text{B}$$

will cause the joint state to stay within the limits at all times for viable initial states. The functions accBoundsFromPosLimits *and* accBoundsFromViability *as well as the attribute* viable *are defined in [169].*

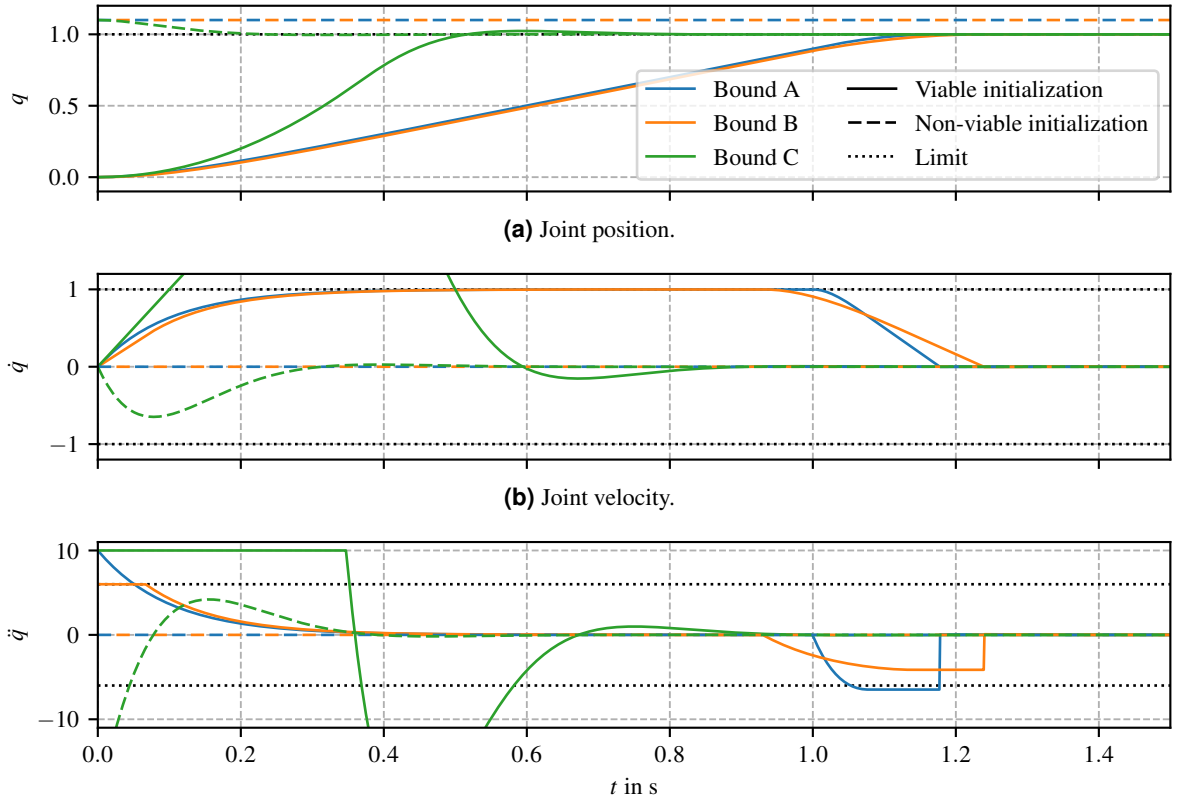
Theorem 7.2:
Let q, $\dot{q}$, $\ddot{q}$, and Δt be as in the previous theorem. Then, clamping $\ddot{q}$ to the interval

$$[\ddot{q}_{\text{lb},5},\ \ddot{q}_{\text{ub},5}] = \frac{2}{\Delta t^2}[q_{\min} - q - \Delta t\dot{q},\ q_{\max} - q - \Delta t\dot{q}]$$

will cause q to converge to a stationary value that satisfies $q_{\min} \leq q \leq q_{\max}$.

In a reduced variant of Theorem 7.1, the acceleration setpoint is clamped to the intersection of the first two intervals, which are marked with A. When this is applied to a manipulator with multiple joints, the resulting vector notation for the lower and upper bound of $\underline{\ddot{q}}$ is $\underline{l}_{\text{lb,A}}(\underline{q}, \underline{\dot{q}})$ and $\underline{l}_{\text{ub,A}}(\underline{q}, \underline{\dot{q}})$, respectively. As illustrated in Figure 7.2 (solid blue lines), these bounds cause the adherence to joint position and velocity limits, while acceleration limits are ignored. If Theorem 7.1 is used in its default form, which is marked with B, the corresponding acceleration bounds $\underline{l}_{\text{lb,B}}(\underline{q}, \underline{\dot{q}})$ and $\underline{l}_{\text{ub,B}}(\underline{q}, \underline{\dot{q}})$ result in compliance with the joint acceleration limits on top of the position and velocity limits, which is also visible in Figure 7.2 (solid orange lines).

Despite these perfect-looking results, there is a major limitation for practical implementations, as Theorem 7.1 requires the initial state to be *viable* [169], which cannot be guaranteed due to numerical inaccuracies or linearization errors. A workaround for this is to find a viable state close to the actual state and use this as the initial state. However, this will not drive the state back to a region within the joint limits as seen in Figure 7.2 (dashed blue and orange lines). An alternative solution to this is the application of Theorem 7.2 with the vector notation $\underline{l}_{\text{lb,C}}(\underline{q}, \underline{\dot{q}})$ and $\underline{l}_{\text{ub,C}}(\underline{q}, \underline{\dot{q}})$ for the acceleration bounds. As demonstrated in Figure 7.2 (green lines), the resulting acceleration setpoint is also able to drive the joint position back from non-viable states at the cost of some overshooting and unlimited velocities and accelerations. For this reason, this method needs to be paired with additional position margins and other methods for velocity and acceleration limits, when these are of importance.



(a) Joint position.

(b) Joint velocity.

(c) Joint acceleration.

Figure 7.2: Visualization of three different acceleration clamping strategies for joint limit avoidance with $\Delta t = 0.1\,\mathrm{s}$. In all cases, the unclamped acceleration setpoint is set to 10.

7.1.2 Tasks

As explained at the beginning of this chapter, the force controller must enable the rendering of arbitrary serial kinematic chains with n DOF in a dynamically consistent way. Thus, the motion of the DT shall obey the dynamic model

$$\mathbf{M}^{\mathrm{a}}\ddot{\underline{q}}^{\mathrm{a}} + \underline{c}^{\mathrm{a}}(\underline{q}^{\mathrm{a}}, \dot{\underline{q}}^{\mathrm{a}}) = \boldsymbol{\mathcal{J}}_{\nu_E^B}^{\mathrm{a}\mathsf{T}}(\underline{q}^{\mathrm{a}})(\underline{w}_{E,\mathrm{ref}}^B - \underline{w}_E^B) + \underline{\tau}_{\mathrm{dri}}^{\mathrm{a}} - \underline{\tau}_{\mathrm{dis}}^{\mathrm{a}}\,, \tag{7.1}$$

with the quantities defined in Section 2.3.2. This model, which can be interpreted as a generalized non-linear admittance in the absence of all other constraints, is driven by the difference between the desired end effector wrench $\underline{w}_{E,\mathrm{ref}}^B$ and the actual end effector wrench $\underline{w}_E^B$. Additionally, the torques $\underline{\tau}_{\mathrm{dri}}^{\mathrm{a}}$ and $\underline{\tau}_{\mathrm{dis}}^{\mathrm{a}}$ act as driving and dissipating torques in the joint space. The relevant frames are defined in Figure 7.3.

Based on this, we can now define the tasks for the HQP problem:

1. The kinematic coupling between the end effectors of the haptic manipulator and the DT is achieved using

$$\begin{pmatrix} \ddot{\underline{x}}_{BH}^B - \ddot{\underline{x}}_{BE}^B \\ \dot{\underline{\omega}}_{BH}^B - \dot{\underline{\omega}}_{BE}^B \end{pmatrix} = \begin{pmatrix} k_{\mathrm{P,tran}}(\underline{x}_{BE}^B - \underline{x}_{BH}^B) \\ k_{\mathrm{P,rot}}\frac{1}{2}(\mathbf{C}_E^B + \mathbf{C}_H^B)\underline{r}_{HE}^H \alpha_E^H \end{pmatrix} + \begin{pmatrix} k_{\mathrm{D,tran}}(\dot{\underline{x}}_{BE}^B - \dot{\underline{x}}_{BH}^B) \\ k_{\mathrm{D,rot}}(\dot{\underline{\omega}}_{BE}^B - \dot{\underline{\omega}}_{BH}^B) \end{pmatrix} = \underline{s} \tag{7.2}$$

   by calculating the desired acceleration difference $\underline{s}$ based on a proportional-derivative (PD) pose controller with the gains $k_{\mathrm{P,tran}}$, $k_{\mathrm{P,rot}}$, $k_{\mathrm{D,tran}}$, and $k_{\mathrm{D,rot}}$. The orientation error $\mathbf{C}_E^H$ is approximated in tangent space using its axis-angle representation. Under ideal conditions, the

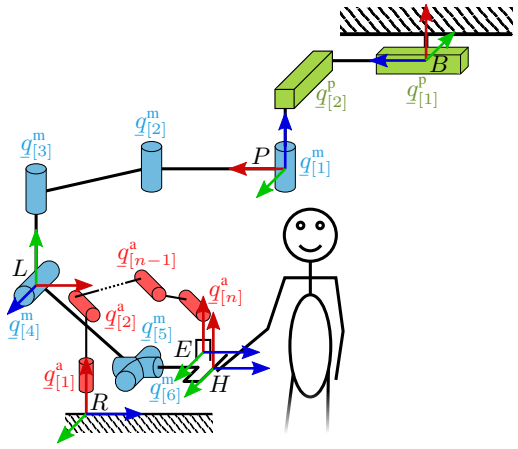


Figure 7.3: The kinematic chain of the DT (red) is rendered using HapticGiant, which is composed of PPU (green) and manipulator (blue). The force controller is supposed to keep the frames H and E coincident. Adapted from [O1].

gains can be set to zero, but linearization and other errors require the use of non-zero gains. Using (2.11), (7.2) can be written as the linear equality task

$$\begin{pmatrix} \boldsymbol{\mathcal{J}}^{\mathrm{m}}_{\underline{\nu}^B_{BH}} & -\boldsymbol{\mathcal{J}}^{\mathrm{a}}_{\underline{\nu}^B_{BE}} \end{pmatrix} \begin{pmatrix} \dot{\underline{q}}^{\mathrm{m}} \\ \dot{\underline{q}}^{\mathrm{a}} \end{pmatrix} = \underline{s} - \boldsymbol{\mathcal{J}}^{\mathrm{p}}_{\underline{\nu}^B_{BH}} \dot{\underline{q}}^{\mathrm{p}} - \begin{pmatrix} \dot{\boldsymbol{\mathcal{J}}}^{\mathrm{p}}_{\underline{\nu}^B_{BH}} & \dot{\boldsymbol{\mathcal{J}}}^{\mathrm{m}}_{\underline{\nu}^B_{BH}} \end{pmatrix} \begin{pmatrix} \underline{q}^{\mathrm{p}} \\ \underline{q}^{\mathrm{m}} \end{pmatrix} + \dot{\boldsymbol{\mathcal{J}}}^{\mathrm{a}}_{\underline{\nu}^B_{BE}} \underline{q}^{\mathrm{a}}$$

with the decision variables $\ddot{\underline{q}}^{\mathrm{m}}$ and $\ddot{\underline{q}}^{\mathrm{a}}$.

2. Based on the reduced variant of Theorem 7.1, the safety-relevant manipulator joint position and velocity limits are enforced by the inequality task

$$\underline{l}^{\mathrm{m}}_{\mathrm{lb,A}}(\underline{q}^{\mathrm{m}}, \dot{\underline{q}}^{\mathrm{m}}) \leq \ddot{\underline{q}}^{\mathrm{m}} \leq \underline{l}^{\mathrm{m}}_{\mathrm{ub,A}}(\underline{q}^{\mathrm{m}}, \dot{\underline{q}}^{\mathrm{m}}) \,.$$

3. As explained in Section 3.2, the manipulator features a singularity when the angle of the virtual *sum joint* $q^{\mathrm{sing}} = \underline{q}^{\mathrm{m}}_{[4]} + \underline{q}^{\mathrm{m}}_{[5]}$ approaches 0 or π. To prevent this from happening, the feasible range of q^{sing} is restricted to the interval $[q^{\mathrm{sing}}_{\mathrm{lb}}, q^{\mathrm{sing}}_{\mathrm{ub}}] \subset (0, \pi)$. Thus, the inequality task

$$\underline{l}^{\mathrm{sing}}_{\mathrm{lb,C}}(q^{\mathrm{sing}}, \dot{q}^{\mathrm{sing}}) \leq \left(\ddot{q}^{\mathrm{sing}}\right) = \left(\ddot{\underline{q}}^{\mathrm{m}}_{[4]} + \ddot{\underline{q}}^{\mathrm{m}}_{[5]}\right) \leq \underline{l}^{\mathrm{sing}}_{\mathrm{ub,C}}(q^{\mathrm{sing}}, \dot{q}^{\mathrm{sing}})$$

based on Theorem 7.2 can be used for singularity avoidance if the margins are sufficiently large.

4. Cartesian limits must be introduced to avoid collisions between the supporting frame and the manipulator of HapticGiant without overconstraining the PPU's range of motion. In this case, the position limit $\underline{x}^B_{BH,\mathrm{lb}} \leq \underline{x}^B_{BH} \leq \underline{x}^B_{BH,\mathrm{ub}}$ for three virtual Cartesian joints is of interest. Thus, the inequality

$$\underline{l}^{\mathrm{ee}}_{\mathrm{lb,C}}(\underline{x}^B_{BH}, \dot{\underline{x}}^B_{BH}) \leq \ddot{\underline{x}}^B_{BH} \leq \underline{l}^{\mathrm{ee}}_{\mathrm{ub,C}}(\underline{x}^B_{BH}, \dot{\underline{x}}^B_{BH}) \tag{7.3}$$

can be formulated based on Theorem 7.2. This has the positive side effect that small limit violations due to linearization are automatically compensated. Substituting $\ddot{\underline{x}}^B_{BH}$ in (7.3) with (2.11) eventually results in the inequality task

$$\underline{l}^{\mathrm{ee}}_{\mathrm{lb,C}} - \begin{pmatrix} \dot{\boldsymbol{\mathcal{J}}}^{\mathrm{p}}_{\underline{v}^B_{BH}} & \dot{\boldsymbol{\mathcal{J}}}^{\mathrm{m}}_{\underline{v}^B_{BH}} \end{pmatrix} \begin{pmatrix} \underline{q}^{\mathrm{p}} \\ \underline{q}^{\mathrm{m}} \end{pmatrix} - \boldsymbol{\mathcal{J}}^{\mathrm{p}}_{\underline{v}^B_{BH}} \dot{\underline{q}}^{\mathrm{p}} \leq \boldsymbol{\mathcal{J}}^{\mathrm{m}}_{\underline{v}^B_{BH}} \ddot{\underline{q}}^{\mathrm{m}} \leq \underline{l}^{\mathrm{ee}}_{\mathrm{ub,C}} - \begin{pmatrix} \dot{\boldsymbol{\mathcal{J}}}^{\mathrm{p}}_{\underline{v}^B_{BH}} & \dot{\boldsymbol{\mathcal{J}}}^{\mathrm{m}}_{\underline{v}^B_{BH}} \end{pmatrix} \begin{pmatrix} \underline{q}^{\mathrm{p}} \\ \underline{q}^{\mathrm{m}} \end{pmatrix} - \boldsymbol{\mathcal{J}}^{\mathrm{p}}_{\underline{v}^B_{BH}} \dot{\underline{q}}^{\mathrm{p}} \,.$$

5. In addition to the end effector, the manipulator elbow at frame L can cause collisions. For this reason, the manipulator elbow position $\underline{x}^B_{BL}$ is constrained analogously to the previous task.

6. Next, the joint limits of the DT are considered by adding the inequality task

$$\underline{l}^{\mathrm{a}}_{\mathrm{lb,A}}(\underline{q}^{\mathrm{a}}, \dot{\underline{q}}^{\mathrm{a}}) \leq \ddot{\underline{q}}^{\mathrm{a}} \leq \underline{l}^{\mathrm{a}}_{\mathrm{ub,A}}(\underline{q}^{\mathrm{a}}, \dot{\underline{q}}^{\mathrm{a}}) \tag{7.4}$$

analogous to item 2. A violation of the joint limits of the DT is not critical from a safety perspective, hence the lower priority compared to the other limits.

7. To complement item 2 with the less critical manipulator joint acceleration limits, Theorem 7.1 is applied in its full form, yielding the inequality task

$$\underline{l}^{\mathrm{m}}_{\mathrm{lb,B}}(\underline{q}^{\mathrm{m}}, \dot{\underline{q}}^{\mathrm{m}}) \leq \ \ddot{\underline{q}}^{\mathrm{m}} \leq \underline{l}^{\mathrm{m}}_{\mathrm{ub,B}}(\underline{q}^{\mathrm{m}}, \dot{\underline{q}}^{\mathrm{m}})\,.$$

8. Similarly, the inequality task for the DT joint acceleration limits complements item 6 by requiring

$$\underline{l}^{\mathrm{a}}_{\mathrm{lb,B}}(\underline{q}^{\mathrm{a}}, \dot{\underline{q}}^{\mathrm{a}}) \leq \ \ddot{\underline{q}}^{\mathrm{a}} \leq \underline{l}^{\mathrm{a}}_{\mathrm{ub,B}}(\underline{q}^{\mathrm{a}}, \dot{\underline{q}}^{\mathrm{a}})\,. \tag{7.5}$$

9. The controller is still insensitive to the wrench exerted by the user. We change this by adding the desired admittance behavior (7.1) in the equality task

$$\ddot{\underline{q}}^{\mathrm{a}} - (\mathbf{M}^{\mathrm{a}})^{-1}\,\underline{\tau}_{\mathrm{con}} = (\mathbf{M}^{\mathrm{a}})^{-1}\left(\boldsymbol{\mathcal{J}}^{\mathrm{a}\mathsf{T}}_{\underline{\nu}^B_E}(\underline{w}^B_{E,\mathrm{ref}} - \underline{w}^B_E) + \underline{\tau}^{\mathrm{a}}_{\mathrm{dri}} - \underline{\tau}^{\mathrm{a}}_{\mathrm{dis}} - \underline{c}^{\mathrm{a}}\right)\,.$$

The equality is intentionally formulated in the joint position space as this results in geometrically interpretable slack variables, which was found to be advantageous for poorly conditioned $\mathbf{M}^{\mathrm{a}}$. Without the auxiliary variable $\underline{\tau}_{\mathrm{con}}$, the resulting haptic rendering would already respect the kinematic behavior of the DT at all times. However, the HQP solver will find a solution with minimal joint acceleration slack when limits are active, which does not necessarily correspond to a dynamically consistent behavior. In reality, a joint that reaches its upper limit will decelerate due to a counteracting constraint torque, which in turn may affect the other joints of the manipulator in a dynamically consistent way. We try to mimic this behavior by collecting the miscellaneous constraint torques in

$$\underline{\tau}_{\mathrm{con}} = \underline{\tau}^{\mathrm{a}}_{\mathrm{con}}(\underline{\alpha}^{\mathrm{a}}_{\mathrm{con}}) + \underline{\tau}^{\mathrm{m}}_{\mathrm{con}}(\underline{\alpha}^{\mathrm{m}}_{\mathrm{con}}) + \underline{\tau}^{\mathrm{sing}}_{\mathrm{con}}(\alpha^{\mathrm{sing}}_{\mathrm{con}}) + \underline{\tau}^{\mathrm{ee}}_{\mathrm{con}}(\underline{\alpha}^{\mathrm{ee}}_{\mathrm{con}}) + \underline{\tau}^{\mathrm{elb}}_{\mathrm{con}}(\underline{\alpha}^{\mathrm{elb}}_{\mathrm{con}}) \in \mathbb{R}^n\,,$$

as a function of the auxiliary decision variables $\underline{\alpha}^{\mathrm{a}}_{\mathrm{con}} \in \mathbb{R}^n$, $\underline{\alpha}^{\mathrm{m}}_{\mathrm{con}} \in \mathbb{R}^6$, $\alpha^{\mathrm{sing}}_{\mathrm{con}} \in \mathbb{R}$, $\underline{\alpha}^{\mathrm{ee}}_{\mathrm{con}} \in \mathbb{R}^3$, and $\underline{\alpha}^{\mathrm{elb}}_{\mathrm{con}} \in \mathbb{R}^3$. The definition of the individual terms depends on the type of constraint:

(a) The counter-torque $\underline{\alpha}^{\mathrm{a}}_{\mathrm{con}}$ acting on the individual DT joints to enforce their limits is already given in the DT joint space, i.e.,

$$\underline{\tau}^{\mathrm{a}}_{\mathrm{con}}(\underline{\alpha}^{\mathrm{a}}_{\mathrm{con}}) = \underline{\alpha}^{\mathrm{a}}_{\mathrm{con}}\,.$$

(b) Likewise, the manipulator joint limits are maintained by the counter torque $\underline{\alpha}^{\mathrm{m}}_{\mathrm{con}}$ in the manipulator joint space. According to (2.15), the transformation of this torque into the DT joint space via the Cartesian space is

$$\underline{\tau}^{\mathrm{m}}_{\mathrm{con}}(\underline{\alpha}^{\mathrm{m}}_{\mathrm{con}}) = \boldsymbol{\mathcal{J}}^{\mathrm{a}\mathsf{T}}_{\underline{\nu}^B_{BE}}\left(\boldsymbol{\mathcal{J}}^{\mathrm{m}\mathsf{T}}_{\underline{\nu}^B_{BH}}\right)^{-1}\underline{\alpha}^{\mathrm{m}}_{\mathrm{con}}\,. \tag{7.6}$$

(c) The singularity avoidance can be realized using a Cartesian torque with the variable magnitude $\alpha^{\mathrm{sing}}_{\mathrm{con}}$. The torque vector is parallel to the rotation axes of manipulator joints 4 and 5, which corresponds to $\underline{C}^B_{L[:,3]}$. The resulting DT joint torque is then

$$\underline{\tau}^{\mathrm{sing}}_{\mathrm{con}}(\alpha^{\mathrm{sing}}_{\mathrm{con}}) = \boldsymbol{\mathcal{J}}^{\mathrm{a}\mathsf{T}}_{\underline{\omega}^B_{BE}}\underline{C}^B_{L[:,3]}\alpha^{\mathrm{sing}}_{\mathrm{con}}\,.$$

(d) Similarly, the Cartesian end effector constraint is realized using the end effector constraint force $\underline{\alpha}^{\mathrm{ee}}_{\mathrm{con}}$. Transforming this force into the DT joint space using (2.15) yields

$$\underline{\tau}^{\mathrm{ee}}_{\mathrm{con}}(\underline{\alpha}^{\mathrm{ee}}_{\mathrm{con}}) = \boldsymbol{\mathcal{J}}^{\mathrm{a}\mathsf{T}}_{\underline{\nu}^B_{BE}}\underline{\alpha}^{\mathrm{ee}}_{\mathrm{con}}\,.$$

Priority	Type	Task
1	EQ	Kinematic coupling
2	IQ	Manipulator joint angle and velocity limits
3	IQ	Singularity avoidance
4	IQ	Cartesian end effector limits
5	IQ	Cartesian elbow limits
6	IQ	DT (non-linear admittance) joint angle and velocity limits
7	IQ	Manipulator joint acceleration limits
8	IQ	DT (non-linear admittance) joint acceleration limits
9	EQ	DT (non-linear admittance) dynamics
10	IQ	Generalized constraint torques range
11	EQ	Generalized constraint torques minimization

Table 7.1: Summary of the equality (EQ) and inequality (IQ) tasks. Adapted from [O1].

(e) Finally, the Cartesian elbow constraint is realized using

$$\underline{\tau}_{\text{con}}^{\text{elb}}(\underline{\alpha}_{\text{con}}^{\text{elb}}) = \boldsymbol{\mathcal{J}}_{\underline{\nu}_{BE}^{B}}^{\text{aT}} \left(\boldsymbol{\mathcal{J}}_{\underline{\nu}_{BH}^{B}}^{\text{mT}} \right)^{-1} \boldsymbol{\mathcal{J}}_{\underline{\nu}_{BL}^{B}}^{\text{mT}} \underline{\alpha}_{\text{con}}^{\text{elb}} .$$

To obtain this equation, the elbow force $\underline{\alpha}_{\text{con}}^{\text{elb}}$ is transformed to the manipulator joint space. Then, (7.6) is utilized for the transformation into the DT joint space.

10. Without additional tasks, the auxiliary decision variables can take any value at any time. This does not match the physical reality, where the constraint torques and forces can only counteract a motion *when* the constraint is about to be hit. Mathematically, this can be formulated as

$$\begin{cases} -\infty \leq \alpha \leq 0, & \text{if upper, but no lower limit active} \\ 0 \leq \alpha \leq \infty, & \text{if lower, but no upper limit active} \\ 0 \leq \alpha \leq 0, & \text{if no limit active} \end{cases} , \tag{7.7}$$

for each scalar α in the quantities $\underline{\alpha}_{\text{con}}^{\text{a}}$, $\underline{\alpha}_{\text{con}}^{\text{m}}$, $\alpha_{\text{con}}^{\text{sing}}$, $\underline{\alpha}_{\text{con}}^{\text{ee}}$, and $\underline{\alpha}_{\text{con}}^{\text{elb}}$. Note, that the last case is formulated as an inequality constraint to achieve a consistent representation of all cases. The decision which limits are active is made by taking the optimized values for $\underline{\ddot{q}}^{\text{m}}$ and $\underline{\ddot{q}}^{\text{a}}$ from the previous HQP iteration and inserting them into the inequality tasks, similar to an active-set strategy [170]. For instance, if an element of $\underline{\ddot{q}}^{\text{a}}$ exceeds $\underline{l}_{\text{ub,A}}^{\text{a}}(\underline{q}^{\text{a}}, \underline{\dot{q}}^{\text{a}})$ in (7.4), the upper limit is considered active and the corresponding entry of $\underline{\alpha}_{\text{con}}^{\text{a}}$ can take any value from $(-\infty, 0]$.

11. The last task ensures that all auxiliary decision variables are minimized by setting them equal to zero, in case the problem is still underconstrained.

A summary of the resulting tasks in HapticGiant's HQP formulation can be found in Table 7.1. In the above formulation, the limits of the manipulator and the DT are split into separate tasks. Furthermore, the joint acceleration limits are separated from the joint position and velocity limits. The reason for this is explained in the following examples.

Example 7.1:

Consider a manipulator consisting of a single prismatic joint with high acceleration limits $\ddot{q}_{\text{max}}^{\text{m}}$. The DT is composed of a single prismatic joint, that is parallel to the manipulator joint, with low acceleration limits $\ddot{q}_{\text{max}}^{\text{a}} < \ddot{q}_{\text{max}}^{\text{m}}$. The position limits of the manipulator are within the position limits of the DT and the kinematic coupling is perfect, i.e., $\underline{\ddot{q}}^{\text{m}} = \underline{\ddot{q}}^{\text{a}}$. When the manipulator starts to approach its upper position limit, the valid acceleration interval $[l_{\text{lb,B}}^{\text{m}}, l_{\text{ub,B}}^{\text{m}}]$ based on Theorem 7.1 shifts towards negative accelerations. At the same time, the feasible DT acceleration interval $[l_{\text{lb,B}}^{\text{a}}, l_{\text{ub,B}}^{\text{a}}] = [-\ddot{q}_{\text{max}}^{\text{a}}, +\ddot{q}_{\text{max}}^{\text{a}}]$ is

only affected by the DT acceleration limits, if the parameters are chosen appropriately. Initially, there is still an overlap between the feasible acceleration intervals.

However, as the manipulator continues to approach its limit, the feasible manipulator acceleration interval shrinks towards the lower manipulator acceleration limit, eventually resulting in an empty intersection with the DT joint acceleration interval. If both intervals are stacked in a single inequality task, the HQP solver will find a solution with slack. This in turn will cause a safety-critical violation of the manipulator joint position limit. This can be fixed by prioritizing the manipulator joint limits over the DT joint limits.

Example 7.2:

As in Example 7.1, we consider a manipulator and a DT, each with one prismatic joint. The joints are parallel and the kinematic coupling is perfect. This time, the DT joint acceleration limit $\ddot{q}^{\mathrm{a}}_{\mathrm{max}}$ *is greater than the manipulator joint acceleration limit* $\ddot{q}^{\mathrm{a}}_{\mathrm{max}}$*, i.e.,* $\ddot{q}^{\mathrm{a}}_{\mathrm{max}} > \ddot{q}^{\mathrm{m}}_{\mathrm{max}}$*. The manipulator joint limits have priority over the DT joint limits in the HQP formulation. When the manipulator joint is far from its position and velocity limits, the valid acceleration interval after applying Theorem 7.1 in its full version is* $[l^{\mathrm{m}}_{\mathrm{lb,B}}, l^{\mathrm{m}}_{\mathrm{ub,B}}] = [-\ddot{q}^{\mathrm{m}}_{\mathrm{max}}, +\ddot{q}^{\mathrm{m}}_{\mathrm{max}}]$*.*

If the DT now reaches a position limit, the interval $[l^{\mathrm{a}}_{\mathrm{lb,B}}, l^{\mathrm{a}}_{\mathrm{ub,B}}]$ *may become completely disjoint from* $[l^{\mathrm{m}}_{\mathrm{lb,B}}, l^{\mathrm{m}}_{\mathrm{ub,B}}]$ *due to* $\ddot{q}^{\mathrm{a}}_{\mathrm{max}} > \ddot{q}^{\mathrm{m}}_{\mathrm{max}}$*. In this case, the HQP solver will return a solution that will eventually cause a violation of the DT joint position limit. To avoid this, the parts of Theorem 7.1, that depend on the joint acceleration limit, must be reduced in priority to maintain the joint position and velocity limits for both the manipulator and the DT. This results in the presented separation into two tasks. Consequently, acceleration limits may sometimes be violated in favor of maintaining the more important position and velocity limits.*

After each HQP iteration, the optimized joint accelerations $\underline{\ddot{q}}^{\mathrm{m}}$, $\underline{\ddot{q}}^{\mathrm{a}}$ are integrated over time to obtain the joint velocities $\underline{\dot{q}}^{\mathrm{m}}$, $\underline{\dot{q}}^{\mathrm{a}}$ and positions $\underline{q}^{\mathrm{m}}$, $\underline{q}^{\mathrm{a}}$. Numerical inaccuracies from the optimization and integration process are corrected by clamping the joint positions $\underline{q}^{\mathrm{m}}$, $\underline{q}^{\mathrm{a}}$ to their respective limits. Additionally, infeasible joint velocities at the position limits, such as a positive joint velocity of a joint at its upper limit, are zeroed.

7.1.3 Integration of the Prepositioning Unit

The PPU state $\underline{q}^{\mathrm{p}}$ and its derivatives $\underline{\dot{q}}^{\mathrm{p}}$, $\underline{\ddot{q}}^{\mathrm{p}}$ are not part of the decision variables in the HQP formulation. This is due to the fact that HapticGiant's PPU with the actuation as presented in Section 4.1.1 provides very limited access to its internal low-level velocity controllers. Therefore, the actual PPU state is considered as a disturbance for the HQP formulation, which facilitates an independent shaping of the PPU behavior.

Without loss of generality, this thesis assumes that an optimal PPU position $\underline{q}^{\mathrm{p}}_{\mathrm{ref}}$ satisfies two properties: First, the planar SCARA manipulator, that is formed by manipulator joints 1 and 2, has maximum manipulability. For this reason, we desire $\underline{q}^{\mathrm{m}}_{[2]} = 90°$. Second, the ground projection of the line between the frames L (manipulator elbow) and H (manipulator end effector) should be parallel to the line between manipulator joints 1 and 3. Based on Table 4.1, this translates to $\underline{q}^{\mathrm{m}}_{[3]} = -\arctan(0.53\,\mathrm{m}/0.67\,\mathrm{m})$. With this information and the current end effector position $\underline{x}^{B}_{BH}$, all elements of the optimized manipulator configuration $\underline{q}^{\mathrm{m}}_{\mathrm{aux}}$ can be determined. Thus,

$$\underline{q}^{\mathrm{p}}_{\mathrm{ref}} = \underline{q}^{\mathrm{p}} + \underline{x}^{B}_{BH[1:2]} - \underline{x}^{B}_{BH,\mathrm{aux}[1:2]}\,,$$

where $\underline{x}^{B}_{BH,\text{aux}}$ is the manipulator forward kinematics solution with configuration $\underline{q}^{\text{m}}_{\text{aux}}$. Subsequently, the desired position $\underline{q}^{\text{p}}_{\text{ref}}$ is converted to a PPU setpoint using the control law

$$\underline{\dot{q}}^{\text{p}}_{\text{cmd}} = \min(\max(k_{\text{PPU}}(\underline{q}^{\text{p}}_{\text{ref}} - \underline{q}^{\text{p}}), \underline{\dot{q}}^{\text{p}}_{\text{lb}}), \underline{\dot{q}}^{\text{p}}_{\text{ub}})$$

with the proportional gain k_{PPU}. The velocity limits

$$\underline{\dot{q}}^{\text{p}}_{\text{lb}} = \frac{1}{\Delta t}(\underline{q}^{\text{p}}_{\text{lb}} - \underline{q}^{\text{p}}) \quad \text{and} \quad \underline{\dot{q}}^{\text{p}}_{\text{ub}} = \frac{1}{\Delta t}(\underline{q}^{\text{p}}_{\text{ub}} - \underline{q}^{\text{p}})$$

are based on a constant velocity prediction over the time horizon Δt, similar to Theorem 7.2, and used to smoothly enforce the PPU position limits $\underline{q}^{\text{p}}_{\text{lb}}$ and $\underline{q}^{\text{p}}_{\text{ub}}$.

7.2 Low-Level Control

Like any other admittance controller, the proposed hierarchical formulation produces a manipulator trajectory in joint space with position, velocity, and acceleration samples. These need to be tracked as accurately and quickly as possible by a low-level controller. Without active limits and good synchronization between the end effectors of the manipulator and the DT, the HQP formulation has no position, velocity, or acceleration feedback loops and therefore behaves like a real physical spring-mass-damper system, which is always passive. Hence, an ideal tracking controller would result in a stable system with perfect admittance characteristics.

In reality, however, the tracking controller will exhibit phase delays and varying closed-loop gains. At the same time, the human operator acts as a mechanical impedance that converts the difference between the desired and the actual hand pose into forces and torques. As previously illustrated in Figure 5.3, this results in a secondary feedback loop, with potentially high gain due to high muscle tension, which can lead to instabilities, despite the utilization of a stable trajectory tracking controller. Worse, it can be shown that any realizable linear admittance control scheme can exhibit contact instabilities in sufficiently stiff environments, if the rendered admittance is smaller than the passive device admittance [171]. This means, that the stability limits of the overall system are highly dependent on the tracking controllers and that there is no linear controller that is stable under all circumstances. To overcome this issue, several approaches have been presented in the literature, that rely on passivity-based controllers [172, 173] or on parameter adjustments when oscillations are detected [174, 175]. In the context of HapticGiant, the manipulator elasticity, originating from the SEAs as well as the finite link stiffness, results in visible mechanical oscillations under manual system excitation with applied joint brakes. This means that the system has additional complex poles, which pose an additional difficulty for the controller synthesis.

These two issues alone make it clear that designing a fast and stable joint tracking controller for HapticGiant is a challenge in itself. Although decades worth of control research can be used for this purpose, the resulting controller synthesis would be far beyond the scope of this thesis and may still not contribute significantly to the state of the art. For this reason, HapticGiant's operation is limited to comparatively simple joint tracking controllers, which are briefly explained in the next section. Furthermore, a simple friction compensation scheme is presented in Section 7.2.2.

7.2.1 Joint Tracking Controllers

The PPU setpoint $\underline{\dot{q}}^{\text{p}}_{\text{cmd}}$ is handled as described in Section 4.3.2. The tracking of the desired manipulator state with $\underline{q}^{\text{m}}$ and $\underline{\dot{q}}^{\text{m}}$ is done in joint space using the cascaded controller from Figure 7.4. The outer loop is a proportional position controller that outputs a velocity setpoint, which is extended by the velocity

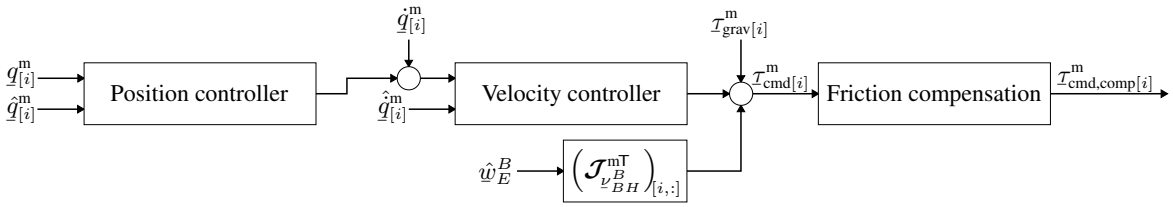


Figure 7.4: Block diagram of the joint tracking controller for a single joint.

feed-forward term $\dot{\underline{q}}^{\mathrm{m}}$. For manipulator joints 3 through 6, the inner loop is realized using a structure equivalent to a proportional-integral (PI) velocity controller. In an attempt to reduce oscillations, the inner loop of the first two joints implements a state feedback controller based on [176]. The required kinematic state estimates $\hat{\underline{q}}^{\mathrm{m}}$ and $\hat{\dot{\underline{q}}}^{\mathrm{m}}$ are determined from measurements in combination with numerical differentiation and low-pass filtering, since the kinematic state estimation from Section 6.1 could not be integrated into HapticGiant as explained in Section 6.1.4. The torque output of the velocity controller is extended with a feed-forward term for gravity compensation $\underline{\tau}^{\mathrm{m}}_{\mathrm{grav}}$, which is obtained using (2.16) and the model data from Section 5.3. Another feed-forward term for the estimated user wrench $\hat{\underline{w}}^{B}_{E}$ is added after transformation into the joint space to increase the achievable mechanical stiffness. Lastly, the friction compensation from the following section is applied.

7.2.2 Friction Compensation

As summarized in [19], robotic joints with strain wave gearing exhibit a variety of friction phenomena. For simplicity, the joint friction model in this section is limited to the constant Coulomb friction τ_{c}, the linear viscous friction with coefficient d, the linear load-dependent friction with coefficient μ, and the torque gain calibration k_τ of the permanent-magnet synchronous motor (PMSM). The corresponding mathematical model

$$\begin{aligned} \tau_{\mathrm{cmd}} &= k_\tau \left((\tau_{\mathrm{c}} + d|\dot{q}| + \mu|\tau|) \operatorname{sign}(\dot{q}) + \tau + I\ddot{q} \right) \\ &= \begin{pmatrix} \operatorname{sign}(\dot{q}) & \dot{q} & \operatorname{sign}(\dot{q})|\tau| & \tau + I\ddot{q} \end{pmatrix} \begin{pmatrix} k_\tau \tau_{\mathrm{c}} & k_\tau d & k_\tau \mu & k_\tau \end{pmatrix}^{\mathsf{T}} , \end{aligned} \tag{7.8}$$

relates the commanded torque τ_{cmd} to the measured load torque τ and the kinematic quantities $\dot{q}$ and $\ddot{q}$. With the effective rotor inertia I, this extends the model presented in [176]. Multiple observations can be stacked due to the matrix notation on the right-hand side of (7.8) to obtain a least-squares estimate of the parameter vector $\begin{pmatrix} k_\tau \tau_{\mathrm{c}} & k_\tau d & k_\tau \mu & k_\tau \end{pmatrix}$, from which the desired parameters can be extracted. This requires that all parameters are identifiable from the excitation signal. Furthermore, the method for identifying the friction parameters of HapticGiant must be able to respect the joint limits and to work without disassembling the manipulator. For this reason, a square wave joint velocity reference with modulated amplitude, that is tracked by a simple velocity controller, is used to obtain data for the identification. In addition to the velocity, τ is modulated by driving and braking the joint under identification by hand to distinguish between τ_{c} and μ for joints with little or no gravitational load.

Based on the identified parameters, the modified torque command

$$\underline{\tau}^{\mathrm{m}}_{\mathrm{cmd,comp}[i]} = k_\tau \left(\underline{\tau}^{\mathrm{m}}_{\mathrm{cmd}[i]} + \tanh\left(\frac{\dot{\underline{q}}^{\mathrm{m}}_{[i]}}{\epsilon} \right) \left(\tau_{\mathrm{c}} + \mu |\mathrm{RNEA}(\underline{q}^{\mathrm{m}}, \dot{\underline{q}}^{\mathrm{m}}, \ddot{\underline{q}}^{\mathrm{m}})_{[i]}| \right) + d\dot{\underline{q}}^{\mathrm{m}}_{[i]} \right)$$

for joint i of HapticGiant's manipulator is calculated from the cascade output $\underline{\tau}^{\mathrm{m}}_{\mathrm{cmd}}$. Here, the sign function is replaced by the $\tanh$ function with the tuning parameter $\epsilon > 0$ to avoid chattering around zero velocity. As a result, the proposed friction compensation considerably improves the subjectively experienced haptic rendering quality, especially for the highly loaded elbow joint. This is also

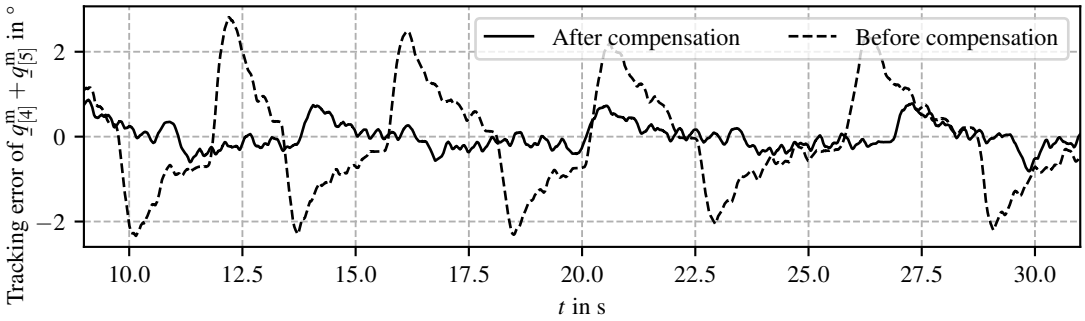


Figure 7.5: The friction compensation reduces the tracking error of joints 4 and 5 to about one third of the original value.

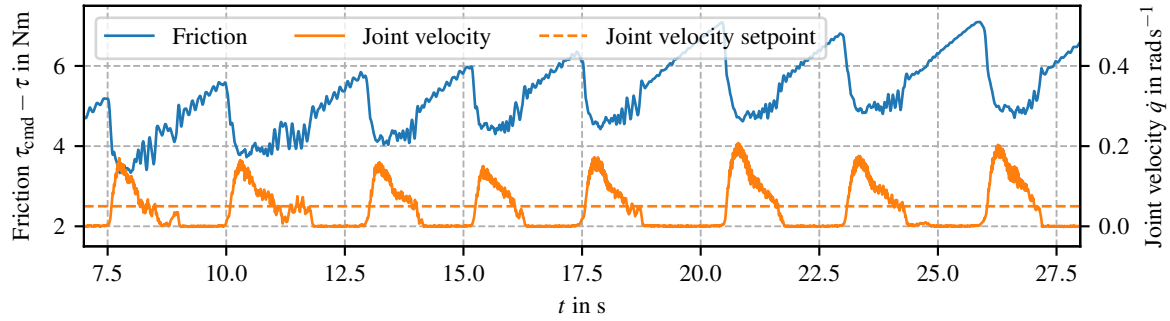


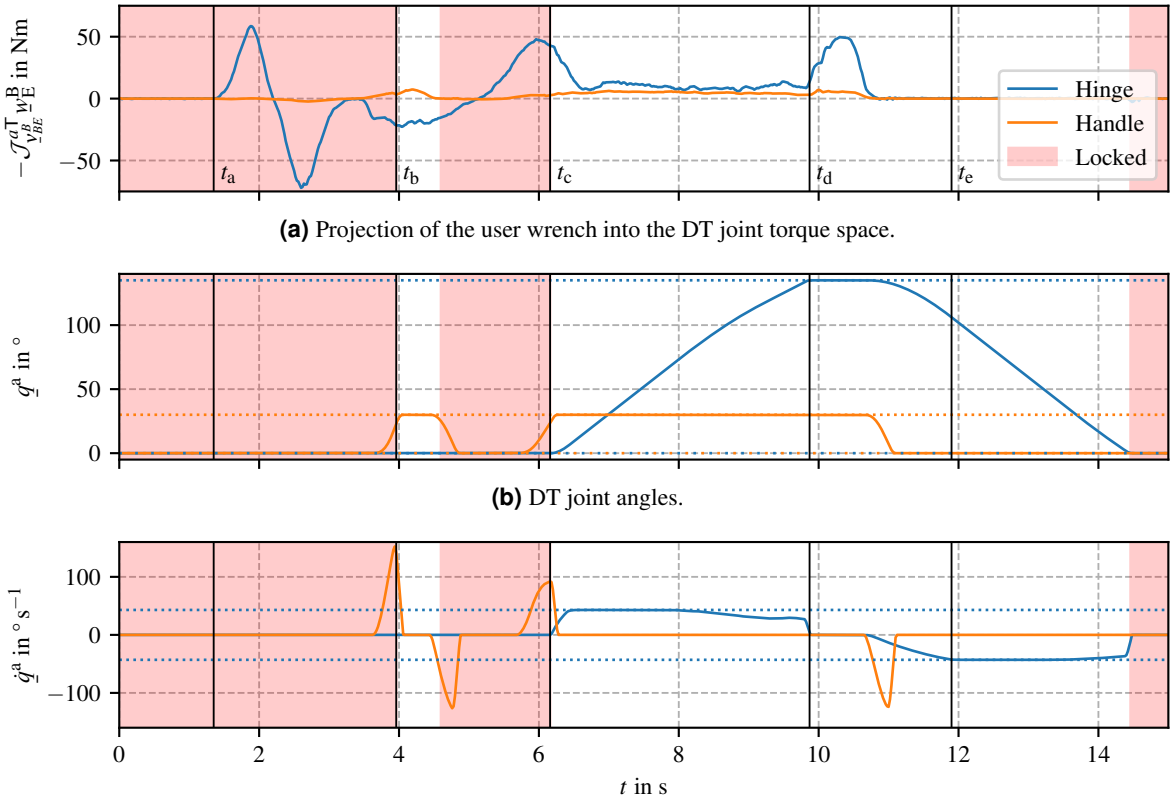
Figure 7.6: Example of the stick-slip behavior of manipulator joint 4 (elbow). As a result, the velocity controller is unable to track the velocity setpoint of $0.05\,\mathrm{rad/s}$.

confirmed by the data in Figure 7.5, which originates from an experiment where the end effector is controlled to stay in a fixed orientation while its height is adjusted. In addition, faster step responses with less overshooting were experimentally confirmed.

During the identification process of the friction parameters, it was found that there are significant residual friction effects that cannot be explained by the model (7.8). In particular, the joints exhibit a strong stick-slip behavior at low speeds as shown in Figure 7.6, which is attributed to the Stribeck effect [177]. Furthermore, the friction was observed to be temperature- and position-dependent in combination with variations between identically constructed joints. These findings indicate that more effort needs to be put into identifying the friction parameters in order to maximize the performance of HapticGiant. Alternatively, the current manipulator actuators could be replaced with models that exhibit less friction.

7.3 Evaluation

For the evaluation, the proposed control scheme was implemented in C++ in combination with *pinocchio* [120] for kinematic and dynamic calculations. The HQP solver was implemented as described in Section 2.4 with *QuadProg++* [178] as the backend for solving the quadratic programs. *PROX-QP* [43] and *qpOASES* [179] are available as alternative backends, but preliminary tests showed that QuadProg++ is the fastest solver for the given problem. All experiments were run with a desired update rate of 1 kHz on a standard desktop computer with an *Intel Core i5-8600* CPU running the control framework from Section 4.2. In the following, a qualitative and a quantitative evaluation will be covered in Section 7.3.1 and Section 7.3.2, respectively. In addition, Section 7.3.3 discusses the observed system limitations.



(a) Projection of the user wrench into the DT joint torque space.

(b) DT joint angles.

(c) DT joint velocities.

Figure 7.7: Experimental data that was captured while rendering a door with HapticGiant. Limits are marked with dotted lines. Taken from [O1].

7.3.1 Qualitative Evaluation

The qualitative evaluation is divided into two experiments, which focus on the validation of the controller features and the rendering of DTs, respectively.

Door

In the first experiment, HapticGiant is used to render the door from Figure 1.3a in an encountered-type haptics scenario. For this purpose, the door with hinge and handle is modeled as a serial kinematic change with two DOF. The door is considered unlocked when the handle is pushed down far enough to retract the latch or when the opening angle exceeds a small threshold. To model this behavior in the HQP formulation, the inequality constraints (7.4) and (7.5) in the tasks for the DT joint limits are modified, so that the upper and lower position limits for the first joint are set to the same value when the door is locked. To achieve a dynamically consistent rendering of the locked door, the case

$$-\infty \leq \alpha \leq \infty, \quad \text{if locked}$$

is added to (7.7) for $\underline{\alpha}^{\mathrm{a}}_{\mathrm{con}[1]}$ to allow constraint torques in both directions. The driving torque $\underline{\tau}^{\mathrm{a}}_{\mathrm{dri}}$ in (7.1) is used to reconstruct the spring in the handle and the self-closing behavior of the door.

The resulting experimental data is presented in Figure 7.7. The following observations can be made at the highlighted time points:

t_a) The user tries to open the door by pushing and pulling without pushing down the handle. Thus, the door remains shut.

t_b) When the handle is pushed down to its limit, the door is unlocked. However, it will not open until the user pushes in the opening direction of the door.
t_c) Pushing down the handle while pushing in the opening direction causes the door to pop open.
t_d) The door does not open further than the upper position limit of the hinge. Releasing the handle causes the handle to return to its initial position.
t_e) After the door is fully released, it slams shut while also respecting the velocity limits.

Isotropic Free-Space Admittance

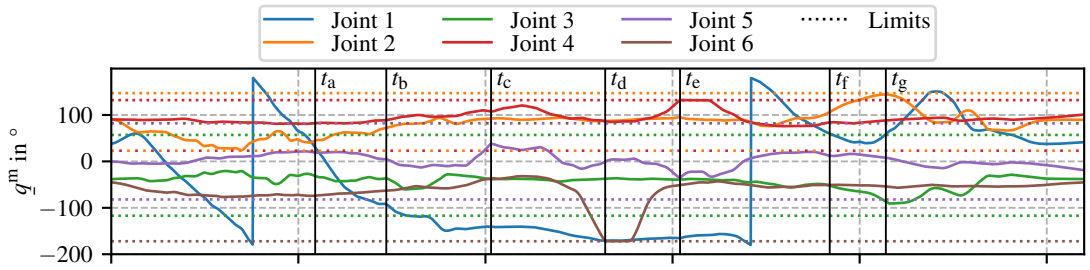
For the second experiment, the admittance model (7.1) was set up to render an isotropic free-space admittance by choosing an appropriate kinematic chain with six DOF. This means that a user can perform arbitrary wrist motions while walking freely in the workspace of HapticGiant, as long as none of the formulated limits are active. When one of the limits is triggered, the motion of the admittance is constrained in dynamically consistent manner. This behavior can also be observed in Figure 7.8. For a better understanding, the following explanations are paired with the highlighted times of interest.

t_a) Until t_a, the user walks in circular motions without any constraints. As shown in Figure 7.8a, the controller correctly handles motions that go beyond a full rotation of manipulator joint 1.
t_b) Figure 7.8d reveals that the manipulator elbow reaches its Cartesian limit in the x-direction. As a result, the end effector loses one rotational DOF, which causes a torque increase visible in Figure 7.8c. Shortly thereafter, the end effector also reaches its Cartesian limit, causing a surge of the user perceived force. Subsequently, the user retreats from both limits.
t_c) As seen in Figure 7.8b, the singularity avoidance task becomes active.
t_d) The lower position limit of manipulator joint 6 is reached and maintained in Figure 7.8a, resulting in the loss of a rotational end effector DOF. The low forces in Figure 7.8c prove that the translation of the end effector is not constrained at all.
t_e) Similarly, the upper position limit of manipulator joint 4, which is responsible for adjusting the end effector height, only restricts the end effector translation.
t_f) In Figure 7.8d, the PPU reaches its lower position limit in the y-direction. The end effector motion is not constrained at all. Instead, the angle of manipulator joint 2 is increased above $90°$ to compensate for the lack of PPU motion.
t_g) Manipulator joint 2 reaches its upper position limits, while the user continues to move into the Cartesian limit in the negative y-direction. As a consequence, the end effector motion is now constrained to the remaining manipulator DOF, and manipulator joint 1 rotates rapidly to facilitate the desired end effector motion as best as possible. This motion is stopped when the end effector and the elbow arrive at their Cartesian limits.

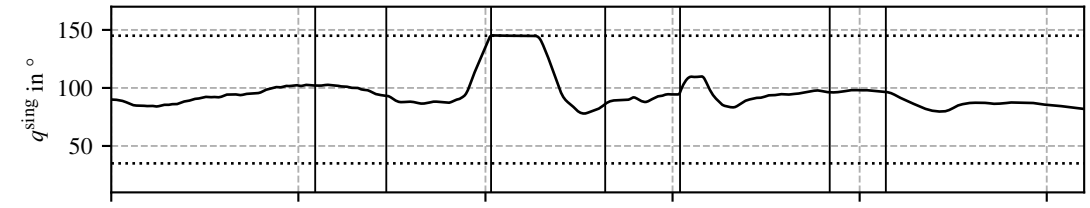
7.3.2 Quantitative Evaluation

Quantitative evaluation of kinesthetic haptic devices is difficult because there are many different experiments and quantities to choose from [16,180]. Recently, the *Haptify* benchmarking method [181] has been proposed as an attempt to standardize the performance evaluation of grounded force-feedback devices. Although Haptify is primarily designed for table-top devices under impedance control with external instrumentation, most of its methods can be adapted to HapticGiant. The resulting quantitative metrics are presented below and supplemented with additional metrics specific to admittance-type force-feedback devices.

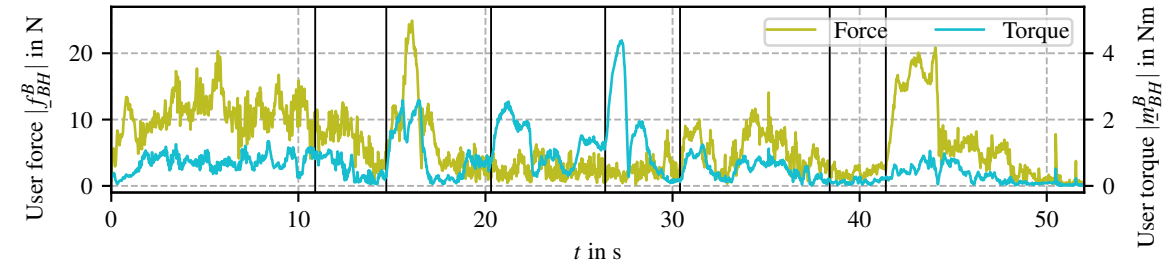
Note, that the low-level joint tracking controllers have a large impact on the performance of the device. This means that the following results are only valid for the current system state with the rather primitive joint tracking controllers. Therefore, quantitative statements about hardware-induced upper bounds of the device performance cannot be made without further investigation.



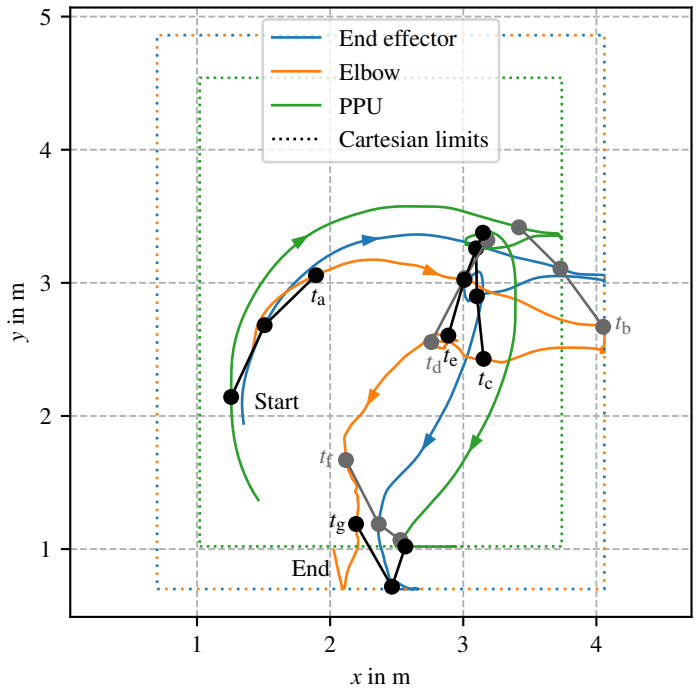
(a) Manipulator joint angles.



(b) Singularity avoidance measure.



(c) User wrench at the end effector.



(d) 2D projection of the trajectories relevant to the Cartesian limits. For the sake of clarity, the data in this plot is limited to the times between $t = 10\,\mathrm{s}$ and $t = 45\,\mathrm{s}$.

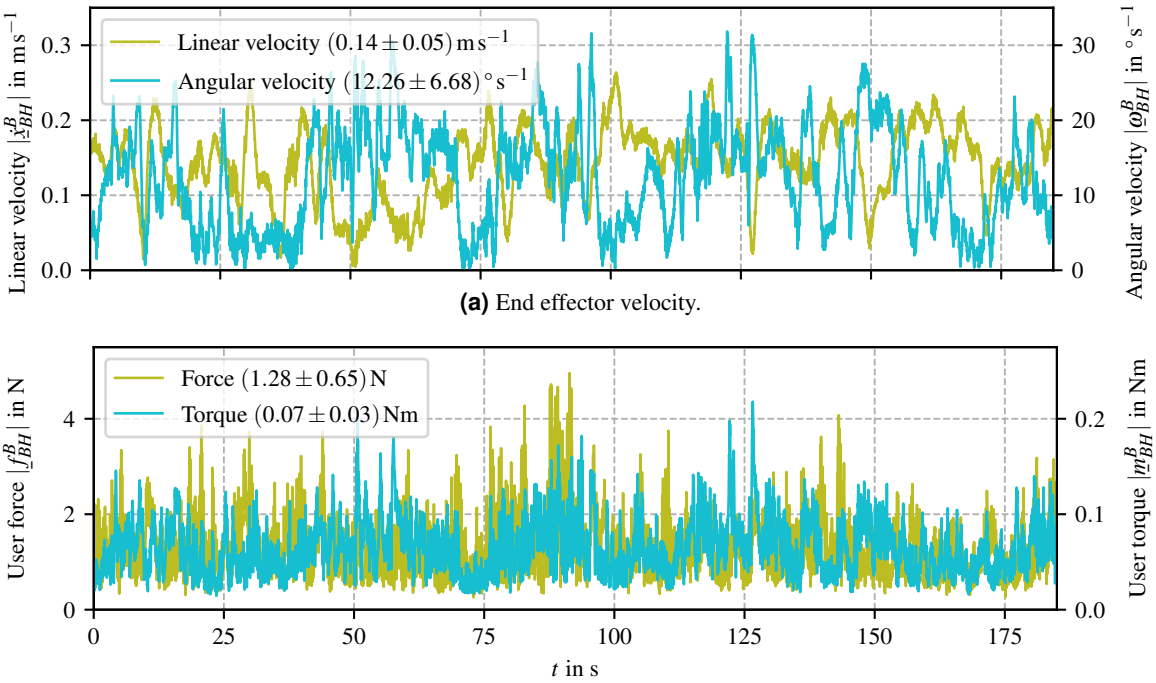
Figure 7.8: Experimental data that was captured while rendering an isotropic free-space admittance with 6 DOF. Limits are marked with dotted lines. Adapted from [O1].

Haptify

The Haptify benchmarking method uses its own instrumentation setup to assess the device performance independent of the device capabilities. For HapticGiant, this instrumentation is already built in or can be substituted without significantly affecting the interpretability of the results. Specifically, the end effector wrenches and accelerations are measured with the dedicated force-torque sensor and the IMU described in Section 4.1.1. For cost reasons, HapticGiant is not be equipped with a motion capture system. However, the load-side joint angles can be used to determine the end effector pose with an accuracy of a few millimeters as long as the link compliance is negligible. Fortunately, this is the case for all Haptify experiments with small end effector wrenches. The stiffness rendering experiment is also not substantially affected by this limitation, as the link stiffness is much higher than the rendered spring stiffness. The ground forces are not measured due to the immense size of HapticGiant.

All of the following experiments were conducted with the same controller parameters. The translational part of the admittance was tuned to a mass of $12\,\mathrm{kg}$ and a damping of $3\,\mathrm{N\,s/m}$. The rotational inertia and damping were set to $0.1\,\mathrm{kg\,m^2}$ and $0.2\,\mathrm{Nm\,s/rad}$, respectively. Following the Haptify procedure, which is explained in depth in [181], we obtain the following metrics:

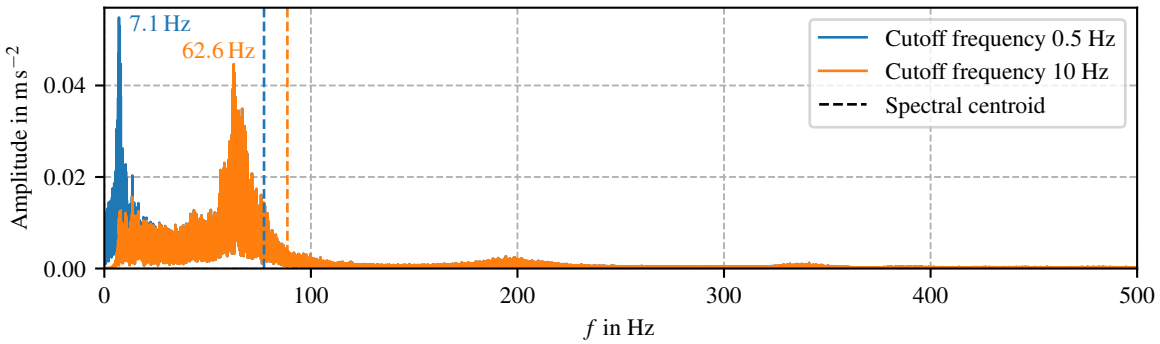
- *Workspace shape*: With all safety margins in place, the end effector of HapticGiant can be moved horizontally over a width of $3.36\,\mathrm{m}$ and a depth of $4.16\,\mathrm{m}$, independent of the end effector height. The height adjustment, taking into account the singularity avoidance and the limits of manipulator joints 4 and 5, spans a height of $0.87\,\mathrm{m}$. Thus, the translational workspace of HapticGiant is a cuboid with a volume of $12.2\,\mathrm{m^3}$. Note, that this data was calculated without measurements, as the forward kinematics and the control internals of the system are all known.
- *Global free-space forces*: For this metric, HapticGiant is configured to render an isotropic free-space admittance by configuring a suitable DT with six DOF as in Section 7.3.1. During the data acquisition, the device is moved with low end effector velocities. In contrast to the original procedure, HapticGiant is not turned off for this experiment in order to keep the admittance-based force controller active. Figure 7.9a shows the recorded linear and angular velocities with the maximum magnitudes $0.27\,\mathrm{m/s}$ and $32\,^\circ\mathrm{/s}$, respectively. The corresponding forces and torques are visualized in Figure 7.9b. Based on this data, the mean free-space force and torque are $1.28\,\mathrm{N}$ and $0.07\,\mathrm{Nm}$ with peaks at $5.31\,\mathrm{N}$ and $0.30\,\mathrm{Nm}$, respectively.
- *Global free-space vibrations*: The global free-space vibrations are evaluated based on the data from the previous experiment. For this purpose, the end effector accelerometer data is high-pass filtered and reduced to a 1D signal as described in [182]. Figure 7.10 shows the resulting amplitude spectrum and time-domain signal. The RMS acceleration and the spectral centroid are $0.69\,\mathrm{m/s^2}$ and $88.6\,\mathrm{Hz}$, respectively. During the analysis, it was found that the default high-pass cutoff frequency of $10\,\mathrm{Hz}$ [181] removes a significant portion of the vibrations that occurred during the experiment. For this reason, the evaluation was repeated with a cutoff frequency of $0.5\,\mathrm{Hz}$, yielding an RMS acceleration of $0.81\,\mathrm{m/s^2}$ and a spectral centroid at $77.3\,\mathrm{Hz}$.
- *Local dynamic forces and torques*: The analysis of the local dynamic forces and torques using non-linear models is omitted for two reasons. First, HapticGiant will not behave identically for the same end effector pose due to its kinematic redundancy. Second, local models require a base pose as an operating point. The choice of such a point for HapticGiant is unclear due to its size and the actuation of the rotational end effector DOF.
- *Frictionless surface rendering*: For this experiment, HapticGiant is configured to render a horizontal, frictionless surface with a stiffness of $750\,\mathrm{N/m}$. This is achieved by constructing a DT with a serial kinematic chain with three orthogonal prismatic joints, one of which resembles a spring. With this setup, the data in Figure 7.11 was recorded for circular end effector motions with a maximum velocity of $0.25\,\mathrm{m/s}$. The resulting average planar force magnitude is $1.96\,\mathrm{N}$, which is in rough agreement with the *global free-space forces* experiment.



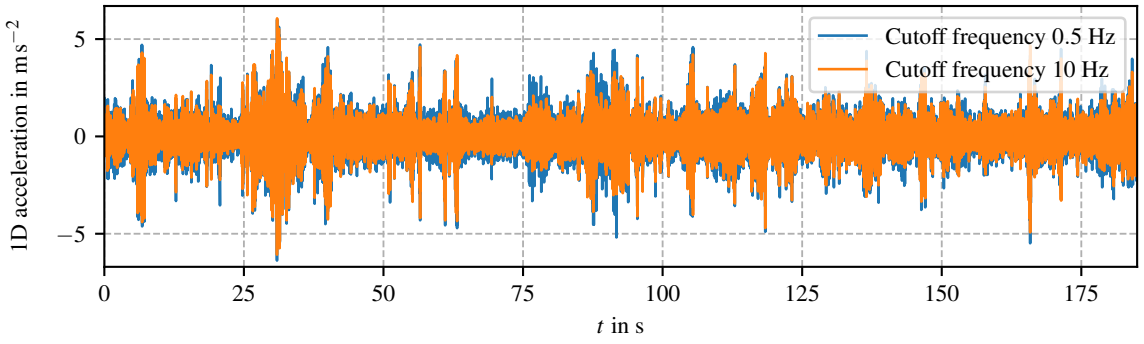
(a) End effector velocity.

(b) End effector force and torque. For better visibility, the data was smoothed with a second-order low-pass filter with a cutoff frequency of 20 Hz.

Figure 7.9: Data captured during the global free-space forces experiment. The ±-notation in the legends indicates mean and standard deviation.

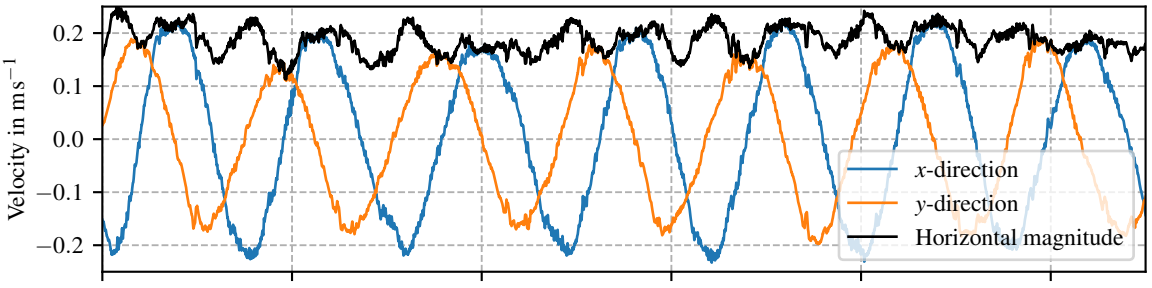


(a) Amplitude spectrum. The peaks are labeled with their frequencies. This reveals that the peak in the spectrum with the reduced cutoff frequency is not caused by the user motion.

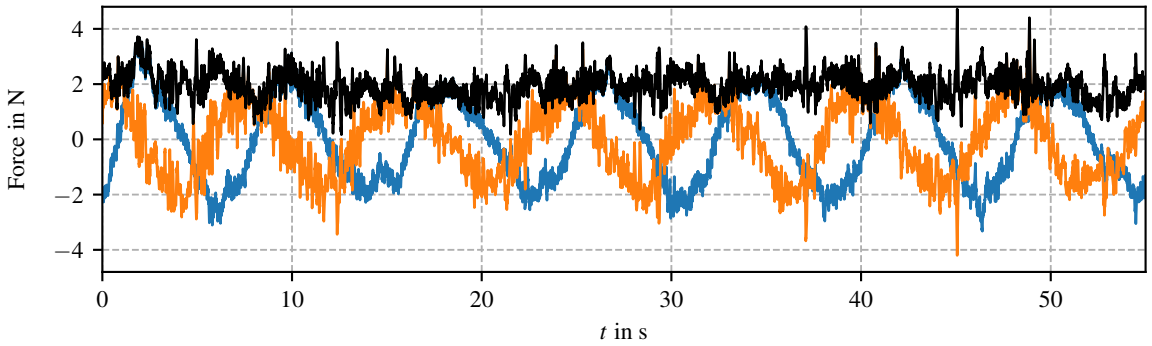


(b) Time-domain signal.

Figure 7.10: Analysis of the end effector vibrations based on accelerometer measurements during the global free-space experiment.



(a) Planar end effector velocity.



(b) Planar end effector force.

Figure 7.11: Data captured during the frictionless surface rendering experiment.

- *Stiffness rendering*: The setup of the previous experiment is reused for the final Haptify experiment, where the stiffness rendering of HapticGiant is tested by repeatedly pushing the end effector into the surface. To analyze the stiffness, all samples belonging to the compression phase were extracted and plotted in Figure 7.12. From this data, we can derive a guaranteed peak force of $76.7\,\mathrm{N}$, whose value is mainly determined by the peak torque of manipulator joint 4 (elbow). Other directions, where gravity compensation is not necessary, will have even higher peak forces. The measured stiffness based on the least-squares method is $818\,\mathrm{N/m}$, which is $9\,\%$ above the expected stiffness. In the Haptify benchmarking method, this translates to a stiffness rendering accuracy of $109\,\%$. Further analysis of the measurement data revealed that the reason for the higher-than-expected stiffness can be found in the tracking errors of the low-level joint controllers. The effect of the finite structural stiffness was not analyzed due to the lack of groundtruth data and a structural manipulator stiffness much higher than $750\,\mathrm{N/m}$.

Other Metrics

Frequency response: The desired end effector wrench $\underline{w}^B_{E,\mathrm{ref}}$ was set to zero in all of the previous experiments as the admittance model was used to generate forces and torques. For this reason, the transfer function between $\underline{f}^B_{E,\mathrm{ref}}$ and $\underline{f}^B_E$ was evaluated in a separate experiment, in which HapticGiant's end effector was attached to two antiparallel, equally preloaded springs with a total stiffness of $140\,\mathrm{N/m}$. The DT was configured to be a single prismatic joint with its axis parallel to the springs. Its end effector has a mass of $12\,\mathrm{kg}$ and a damping of $20\,\mathrm{N\,s/m}$. With this setup, the end effector forces were measured for sinusoidal force setpoints with an amplitude of $5\,\mathrm{N}$ and varying frequencies.

The resulting frequency response is depicted in Figure 7.13 and shows good agreement with the expected transfer function of the corresponding spring-mass-damper system up to frequencies of $3\,\mathrm{Hz}$. At higher frequencies, the signal-to-noise ratio (SNR) of the end effector force measurements crosses $0\,\mathrm{dB}$, which impairs the amplitude and phase estimation of the measured force. This is also the reason for

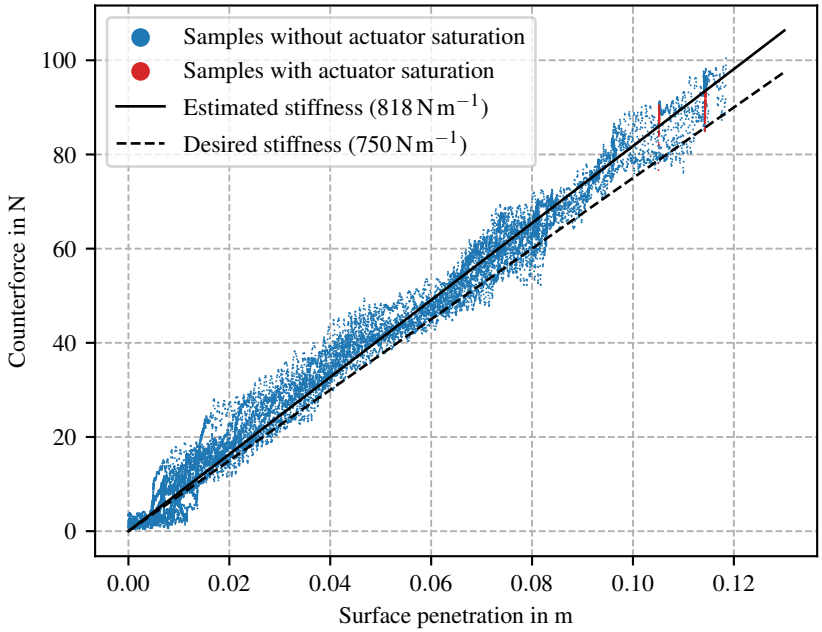


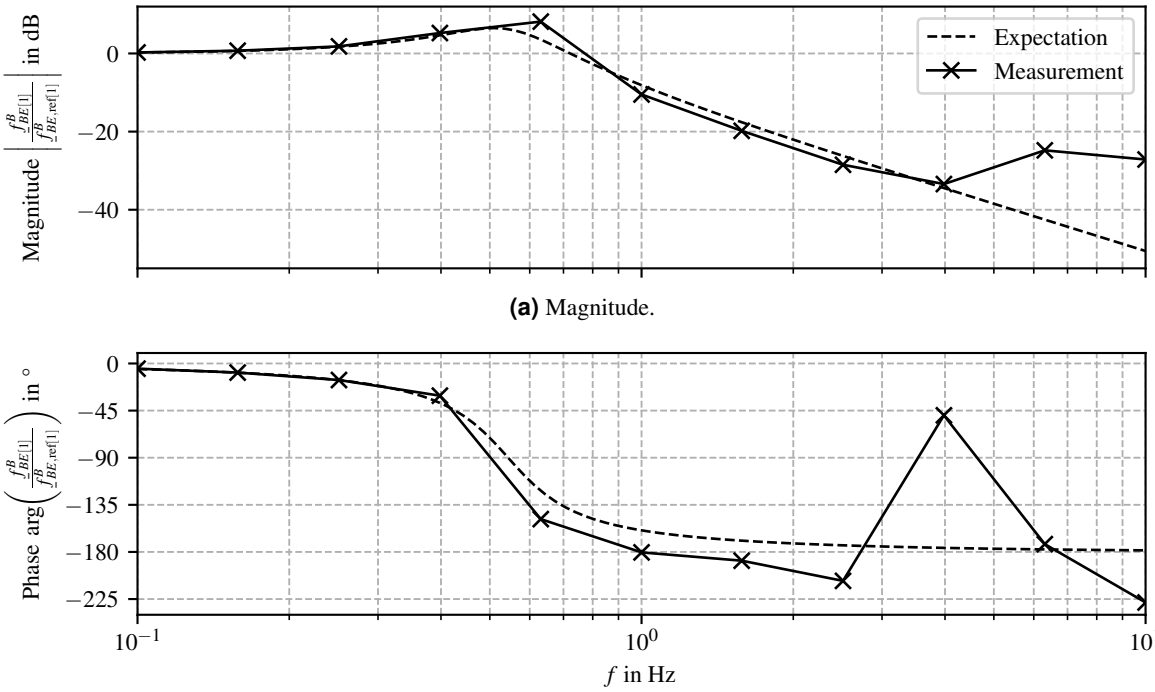
Figure 7.12: Experimental analysis of HapticGiant's stiffness rendering capability. The samples with actuator saturation are overlapping with those without saturation due to the joint torque required for accelerating the inertia of the manipulator.

the sudden phase deviation at 4 Hz. Thus, HapticGiant has a force tracking bandwidth of at least 3 Hz with its current low-level controllers.

Admittance parameter boundaries: Until now, the choice of the admittance parameters seemed arbitrary. However, real admittance controllers cannot be tuned to any inertia or damping due to stability constraints [33, 171]. To examine this behavior for HapticGiant, the admittance from the Haptify *global free-space forces* experiment is utilized with the translational mass set to a low value. Starting with a high translational damping value, the damping is gradually reduced until a user moving the end effector at will can destabilize the system. This way, the lowest stabilizing damping is identified for the given mass. The mass parameter is then increased and the search for the lowest stabilizing damping is repeated. The same procedure is applied to the rotational admittance parameters. Note, that the exact numerical results of this experiment depend on the user. Nevertheless, the general behavior is expected to be reproducible.

The results of this experiment are shown in Figure 7.14. At first glance, the stability boundaries for the translational and the rotational component look similar. In both cases, lower inertia requires higher damping, and the damping can be omitted entirely above a certain inertia. However, the behavior differs between rotation and translation for low inertia parameters. For the rotational admittance, the inertia can be set to very small values, if the damping is high enough. In contrast to that, the controller was not able to render translational inertia values below 2 kg without becoming unstable, regardless of the damping parameter. Furthermore, there is a sharp drop in the required damping in the translational case for masses between 4.5 kg and 5.9 kg. The suspected cause for this is the feed-forward term of the end effector wrench from Section 7.2.1, which seems to amplify translational end effector oscillations originating from the limited manipulator stiffness and the large link lengths. This effect is not present in the rotational case because the link lengths have less effect on the rotational oscillations.

Computation time: For a DT with six joints, the average computation time for the control scheme including low-level controllers was 529 µs with a maximum of 707 µs. On average, 92.6 % of this time



(a) Magnitude.

(b) Phase.

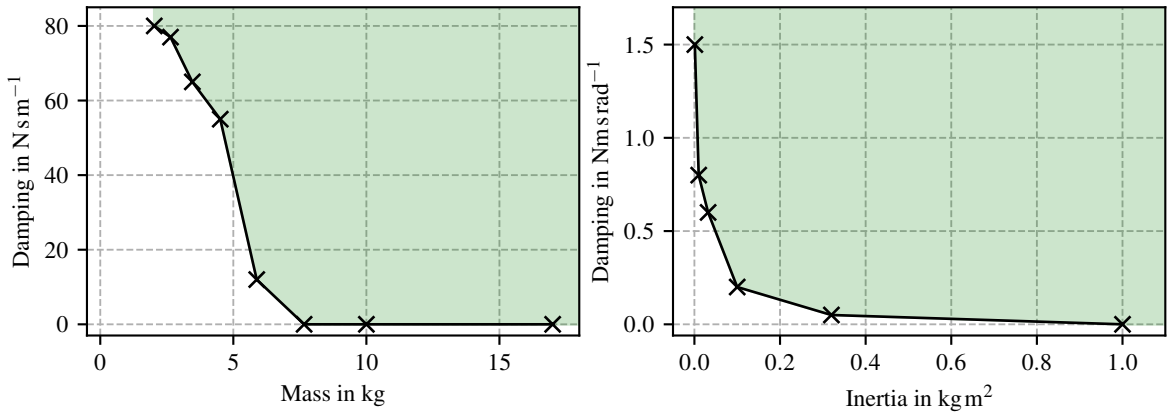
Figure 7.13: Measured closed-loop frequency response of the force setpoint in comparison with the ideal frequency response.

was spent for solving the HQP problem. As a result, the presented control scheme achieves the desired update frequency of 1 kHz, while leaving enough time for other modules with real-time constraints.

7.3.3 Known Limitations

Several limitations were discovered during the implementation and the evaluation of the controller:

- The kinematic models of the haptic manipulator and the DT are both linearized using their Jacobian. As a consequence, perfect end effector synchronization is never achieved in reality. Furthermore, a single linearization is used throughout the time horizon Δt in the case of the Cartesian limits, which requires Δt to be small enough. With the linearized kinematic coupling as the highest priority task, it is also possible to obtain solutions, where there is a small amount of slack in the task for the DT joint position and velocity limits. The potentially resulting violation of the DT position and velocity limits is fixed by the existing clamping step after the integration of the joint accelerations. In this case, the kinematic coupling task will reestablish the synchronization over time.
- The limit switching strategy from task 10 in Section 7.1.2 was observed to cause alternating limit activation in situations, where more than one limit is active at the same time. As shown in Figure 7.15, this can lead to oscillations in the joint accelerations signals of the DT. Although this could theoretically cause manipulator joint chatter, it was found to be irrelevant in practice due to the low-pass behavior of the manipulator and its joint tracking controllers.
- As mentioned above, the end-to-end force control and haptic rendering performance is heavily dependent on the low-level controllers. In the current state, this leads to non-ideal behavior when the user desires fine adjustments of the end effector position. To realize such an adjustment, the user applies a small force. The resulting force causes the position reference of the manipulator to



(a) Translational component.

(b) Rotational component.

Figure 7.14: Stability boundaries for the admittance parameters. The shaded areas represent stable parameter combinations.

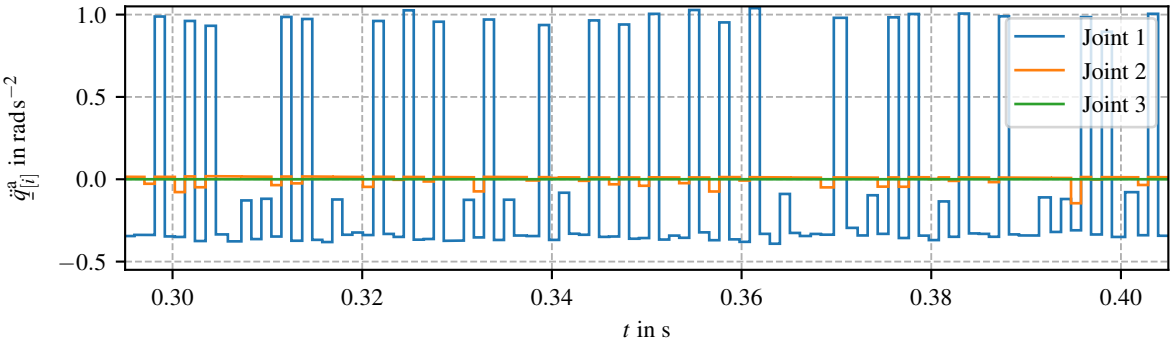


Figure 7.15: Joint acceleration $\ddot{q}^{\mathrm{a}}$ of a planar manipulator with three revolute joints, when all joints are driven into their upper position limits. All limits are respected despite the oscillations.

change as expected, but the joints stick at their initial positions due to incompletely compensated friction. To still achieve the desired adjustment, the user applies more force, which causes a position overshoot as soon as the manipulator joints break loose.

- There is no dynamic model of the manipulator in the current HQP formulation. This can result in manipulator reference trajectories that are impossible to track without violating the joint torque limits.

7.4 Discussion

This chapter introduced a novel force control scheme for rendering the kinematic and dynamic properties of articulated bodies with HapticGiant. Thereby, DTs for various objects, including doors, excavators, buttons, and free rigid bodies, can be created. The proposed force controller is formulated as a set of hierarchical tasks in the framework of HQP. As a result, manipulator joints limits, DT joint limits, Cartesian limits, and singularities are considered intrinsically and jointly in the control law, which is a novelty compared to state-of-the-art force control schemes. The proposed control scheme is also able to handle the PPU of HapticGiant by optimizing its position based on suitable criteria, which allows for a very large workspace. Beyond that, a concise overview of the low-level control concept, including a simple friction compensation, was given. Although the presented controller is tailored to

Metric	HapticGiant	Touch [D18]	Touch X [D18]
Workspace volume in cm^3	$\mathbf{12.2 \cdot 10^6}$	$11.8 \cdot 10^3$	$10.4 \cdot 10^3$
Mean free-space forces in N	1.28	0.98	$\mathbf{0.87}$
Free-space vibration RMS magnitude in $\mathrm{m/s^2}$	0.81	0.07	$\mathbf{0.04}$
Free-space vibration spectral centroid in Hz	77.3	123.0	$\mathbf{162.8}$
Frictionless surface rendering forces in N	1.96	0.53	$\mathbf{0.38}$
Stiffness rendering accuracy in $\%$	$\mathbf{109}$	47	80
Stiffness peak force in N	$\mathbf{76.7}$	2.7	5.7

Table 7.2: The Haptify metrics for HapticGiant and two commercial table-top devices. The values in the two columns on the right are taken from [181]. Bold text highlights the best value for each metric.

the specific requirements of HapticGiant, it can be easily adapted to other kinesthetic haptic devices, if their manipulator Jacobian is regular. Moreover, the interpretation of the PPU as a known disturbance enables the application on mobile platforms, even on those that are uncontrollable, e.g., a backpack or a vessel.

The proposed force control scheme was implemented on real hardware. Performance measurements show that the control scheme is real-time capable and can achieve update rates of more than $1\,\mathrm{kHz}$. The qualitative evaluation in Section 7.3.1 demonstrates that the task-based formulation of the control problem works as expected and that DTs can be rendered dynamically consistent. In fact, the dynamically consistent rendering works so well that test users sometimes did not notice when they reached some of the manipulator joint limits, thinking that the DT constrained the motion. The intrinsic handling of limits has the additional advantage that users cannot endanger themselves or HapticGiant with careless force and torque inputs. Therefore, almost anyone can use the system with minimal instruction. In Section 7.3.2, the Haptify benchmarking method was used as a quantitative evaluation tool to obtain comparable metrics. The results, which also include data for two commercial table-top devices, are summarized in Table 7.2. From this data, we can see that HapticGiant has a workspace whose mean edge length is about ten times the length of typical table-top devices. At the same time, HapticGiant's mean free-space force is not significantly higher. The comparison also shows that HapticGiant can render stiff surfaces with a higher accuracy and ten times higher peak forces than the commercial devices at the cost of higher planar forces. On the downside, HapticGiant has tenfold increased vibrations with a lower spectral centroid, which limits the rendering of fine textures and surfaces. Further experiments studied the ability to render force setpoints and the space of admittance parameters leading to stable behavior.

In future work, the low-level controllers should be revised with special attention to friction effects and the finite manipulator stiffness to further improve the haptic rendering performance of HapticGiant. At the level of the HQP formulation, new tasks can be added. For example, a compliance task to mitigate the severity of collisions between the user and the manipulator is a promising safety feature. As mentioned above, extra tasks for the dynamic model of the manipulator could be added to avoid infeasible manipulator trajectories. Furthermore, the HQP formulation should be generalized to support redundant haptic manipulators with non-regular Jacobians. If required, the limit switching strategy should be revised to remove alternating limit activations. On the application side, the existing force and torque setpoints should be used as a starting point for linking HapticGiant with existing haptic rendering software, such as CHAI3D [160]. In this way, a wide range of existing applications for kinesthetic haptic interfaces can be made accessible to HapticGiant. Last but not least, new applications, which take advantage of the unique capabilities of HapticGiant, should be developed. An example of such an application is given in the next chapter.

CHAPTER
8

Rendering of Multiple Digital Twins

Contents

> The science of today is the technology of tomorrow.
>
> EDWARD TELLER (1908 – 2003)

In the previous chapter, we have seen how HapticGiant's manipulator can be used to render arbitrary serial kinematic chains or admittances. According to Section 1.2.2, this facilitates the rendering of DTs in an encountered-type haptic display (ETHD) setup. As an example, the rendering of a single door at a fixed position has been presented. While this demonstration is already quite immersive, it does not fully exploit the large workspace of HapticGiant, which can be made even larger in the target environment (TE) by applying motion compression [45, 46]. Thus, the idea of seamlessly displaying multiple objects, hereinafter called *targets*, with HapticGiant as a single kinesthetic haptic interface is obvious. The resulting scenario, where HapticGiant's manipulator is relocated while the user is not interacting with it, is sketched in Figure 8.1.

The question now is how to accomplish this. For a maximum degree of user immersion, this thesis proposes the following two-step solution:

1. When the user is not interacting with an object, predict the next object the user is likely going to approach. This requires an intention estimation algorithm that processes the user data, such as head pose and gaze direction, along with the available target positions.

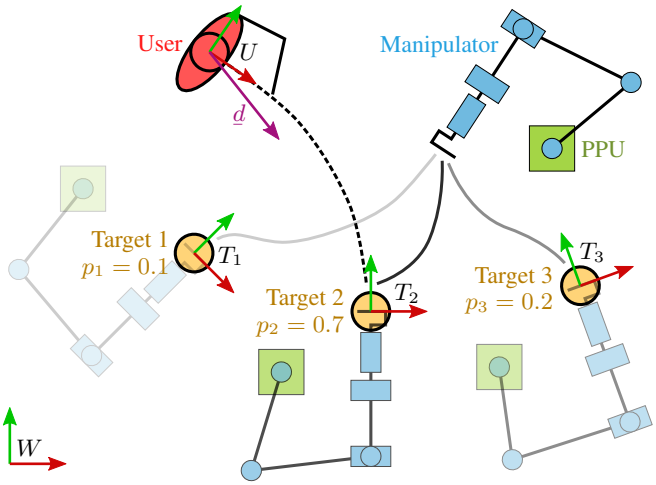


Figure 8.1: Top view of an example scenario where the user can choose between different DTs. In this example, the second target must be correctly predicted as the most probable user intention and HapticGiant's end effector must be repositioned without collisions. Optimally, the repositioning process is finished before the user reaches the desired target.

2. Find a suitable PPU and manipulator motion to a pose in agreement with the intention estimation output from the previous step, while taking into account that the workspace is shared with the user, who can have a change of mind at any time. This means that collisions must be actively avoided and that motion planning must be done online.

In general, the research community is familiar with the questions raised by these steps. However, as outlined below, current solutions either do not fully address the specific requirements for the rendering of multiple DTs or have limitations that could be addressed with further research. Motivated by this, the goal of this chapter is to provide a preview on future work with HapticGiant. This also means that the proposed methods should not be considered as completed research, but rather as a proof-of-concept for the rendering of multiple DTs.

Estimating a user's intention is a research topic that has been studied in various fields [183], with problems ranging from the detection of the low-level intention in 1D manipulation [184] to the prediction of high-level tasks [185]. The problem of multi-class intention estimation also occurred in HapticGiant's predecessor, where the prediction of the user's next waypoint based on walking direction and gaze was realized using a Bayesian estimator [186]. To achieve this, the intention is encoded in a hidden Markov model (HMM), whose observations are hand-crafted ego-centric features. The necessary conditional probability density function of the feature vector is calculated with a Gaussian mixture model based on experimentally recorded training data. While this approach could be already used for the given problem, it depends on manual feature engineering, which may lead to a loss of accuracy. Furthermore, the estimator cannot handle a user pursuing an unknown waypoint.

These limitations are also present in the conceptually similar Bayesian estimator in [187], which was published more than a decade later. The *head-mounted AR intention recognition (HAIR)* method also deploys an HMM to estimate the user's intention in the context of collaborative robots [188, 189]. Instead of constructing or learning probability densities, the measured data, consisting of head pose, gaze direction, and object positions, is directly converted into a likelihood vector based on hand-crafted rules. This way, the detection of irrational user behavior and unknown targets is enabled.

In the last decade, long short-term memory (LSTM) neural networks (NNs) have gained increased popularity for estimating the next object a user is going to interact with [190–192]. The advantage of

these methods is that they automatically learn features from measurements, potentially allowing for higher prediction accuracy. However, to the best of the author's knowledge, all NN-based methods depend on a fixed number of targets with immutable positions, which is a major limitation for the use case described above. Furthermore, none of the mentioned methods is able to detect whether a user is pursuing an unknown target.

In an attempt to combine the advantages of the Bayesian and the NN-based methods, the first part of this chapter, Section 8.1, presents a novel intention estimation algorithm based on recurrent NNs. The resulting estimator can handle a flexible number of targets, whose positions do not need to be known a priori, and does not depend on manual feature engineering or extensive parameter tuning. Furthermore, a user with an unknown target can be detected optionally.

The topic of motion planning can be divided into path planning, where a sequence of waypoints is obtained without time information, and trajectory planning, where waypoints are determined as a function of time. As summarized in [193–195], both subproblems have been studied extensively and come with a variety of solutions. In particular, rapidly-exploring random trees (RRTs) represent a well-understood method for finding paths in high-dimensional configuration spaces, which can also take costs into account with the extension to RRT*. As an alternative to RRT*, the A* algorithm can be used for path planning on so-called roadmaps, which are undirected graphs spanning the configuration space. Despite their simplicity and popularity, both methods cannot be used for motion planning in the context of HapticGiant due to their high computational cost for high-dimensional systems.

A subcategory of motion planning algorithms is specialized in mobile platforms with attached manipulators. HapticGiant can be considered as such a system, allowing the application of the optimized methods from [196]. However, most of these methods cannot handle dynamic obstacles during global planning or are too slow, rendering them unusable for the given problem with a moving user. A noteworthy exception in terms of computational speed can be found in [197], where an approach accelerated by graphics processing units (GPUs) is used for online motion planning of mobile manipulators. Despite its performance advantage, this approach is also unable to handle dynamic obstacles on a global scale as required for the given problem.

In addition to fast global planning with dynamic obstacles, a suitable motion planning algorithm must be able to handle the estimated and potentially fuzzy human intention. In the literature [198,199], this is partially covered by taking into account the knowledge of the human intention to avoid human-machine collisions. However, as far as the author is aware, no method has been published where the output of an intention estimation algorithm is directly used to determine the goal configuration of a planning algorithm.

Based on this concise description of the shortcomings, the second part of this chapter, Section 8.2, presents a novel motion planning algorithm for HapticGiant that separates the full path planning problem into two simpler problems for the manipulator and the PPU, respectively. This facilitates the inclusion of volatile intention estimation outputs as well as the handling of dynamic obstacles in online scenarios. For the integration with HapticGiant, the resulting system is complemented with methods for path-to-trajectory conversion and intention processing.

This chapter is based on results presented in the author's publication [O7].

8.1 Intention Estimation with Recurrent Neural Networks

After formalizing the problem statement in Section 8.1.1, the network architecture for the desired intention estimation is presented in Section 8.1.2, followed by information about the training process in Section 8.1.3. The intention estimation is concluded with an evaluation in Section 8.1.4.

8.1.1 Problem Statement

In the following, we assume the scenario illustrated in Figure 8.1. In this scenario, a user with head frame U can move relative to the fixed reference frame W without restrictions while carrying a HMD. The HMD captures the user's head pose $\mathbf{T}_U^W$ and gaze direction $\underline{d}^U$ with $\left\|\underline{d}^U\right\|_2 = 1$. The scene contains $n < N$ distinct, point-like targets with the positions $\underline{x}_{WT_i}^W$. At most, $N \in \mathbb{N}^+$ targets are present at the same time. In this setup, the goal is to estimate the discrete probability distribution that describes which object the user will interact with next. Formally, this distribution is described by

$$\underline{p}^\mathsf{T} = \begin{pmatrix} p_1 & \dots & p_n \end{pmatrix} \in \mathbb{R}^{1\times n} \quad \text{with} \quad \sum_{i=1}^{n} p_i = 1\,,$$

where p_i describes the probability that target i is the target the user is going to approach next. Optionally, the probability distribution $\underline{p}$ is augmented with the probability $p_\varnothing$. The value of $p_\varnothing$ describes the probability that the user is not pursuing any of the known targets. Hereinafter, this case is referred to as *no known target*. In general, $\underline{p}$ must be updated in real time. This translates to a required update rate of about 10 Hz with the intended downstream application in mind.

8.1.2 Network Architecture

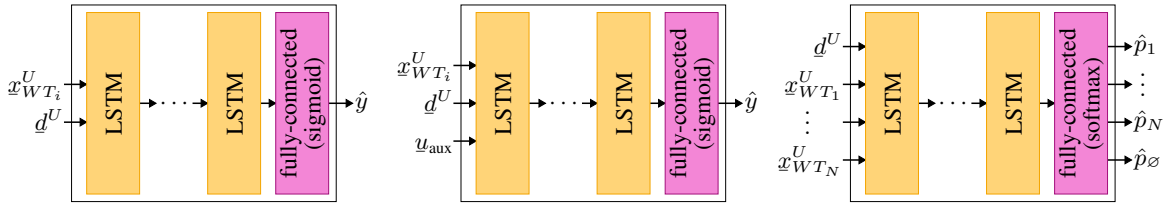
For training and inference, the user and target data is transformed into ego-centric coordinates. As a result, the situation is fully described by the gaze direction $\underline{d}^U$ and the resulting target positions $\underline{x}_{WT_i}^U$. This has the advantage that the user position can be omitted from the NN input and that the target positions can be adjusted without retraining. Motivated by [183, 190–192], the following architectures all have in common that they represent recurrent NNs based on LSTM layers, which process one sample at a time.

Three different network architectures are proposed and analyzed in the following:

1. *Binary classifier*: The first architecture, which is depicted in Figure 8.2a, is a binary classifier trained to predict whether the user is about to approach a particular target or not. This idea is motivated by [200], where the authors showed that a set of binary classifiers can replace a single multi-class classifier. During training, a single classifier with the output $\hat{y}$ is trained without distinguishing between different targets. With this architecture, $\hat{y} \approx 1$ indicates that the classifier is confident that the considered object is approached. For the inference, n instances of this binary classifier, each responsible for a different target i, are stacked with their sigmoid activation function of the last layer removed. The resulting outputs are then combined into the desired probability vector $\hat{\underline{p}}$ using a softmax layer.
2. *Extended binary classifier*: The main disadvantage of the binary classifier is that it is unaware of the surroundings of its target. For instance, two targets that are arranged in a row from the user's point of view are considered fully isolated, even though an estimator could exploit the relative spatial information to adjust the predicted probabilities. To address this, the second architecture extends the binary classifier for target i with the auxiliary input

   $$\underline{u}_{\text{aux}}^\mathsf{T} = \begin{pmatrix} \underline{x}_{WT_1}^{U\mathsf{T}} & \dots & \underline{x}_{WT_{i-1}}^{U\mathsf{T}} & \underline{x}_{WT_{1+1}}^{U\mathsf{T}} & \dots & \underline{x}_{WT_N}^{U\mathsf{T}} \end{pmatrix}$$

   as illustrated in Figure 8.2b. If $n < N$, i.e., the maximum number of targets is not reached in the current scenario, the unused elements of $\underline{u}_{\text{aux}}$ are set to a reserved value, that does not occur in the natural input domain.
3. *Multi-class classifier*: Both variants of the binary classifier cannot detect if the user has *no known target*. For this reason, a multi-class classifier is proposed as the third network architecture.



(a) Binary classifier i. **(b)** Extended binary classifier i. **(c)** Multi-class classifier.

Figure 8.2: Network architectures for the proposed intention estimation. For inference, both binary classifiers are instantiated n times and combined with a softmax function. The multi-class classifier can be used without modifications. Adapted from [O7].

As shown in Figure 8.2c, this classifier processes all target positions and the gaze direction in ego-centric coordinates to obtain the prediction $\underline{\hat{p}}^\mathsf{T} = \begin{pmatrix}\hat{p}_1 & \dots & \hat{p}_n & \hat{p}_\varnothing\end{pmatrix}$ for the true probability vector $\underline{p}$, including the probability for *no known target*. The multi-class classifier has a fixed number of inputs and is instantiated only once for inference. If there are less than N targets in the scenario, the corresponding inputs are set to reserved values, analogous to the extended binary classifier, and the corresponding outputs are ignored. As a variant, the multi-class classifier can be trained without the *no known target* output, so that $\underline{\hat{p}}^\mathsf{T} = \begin{pmatrix}\hat{p}_1 & \dots & \hat{p}_n\end{pmatrix}$. In the following, this variant is called *multi-class classifier without no known target*.

8.1.3 Training

At first glance, datasets with gaze and head pose data, such as [201, 202], seem to be suitable for training and evaluating the proposed network architectures. However, these datasets are typically created with a focus on eye movement and do not capture interactions with the environment. The *OpenNEEDS* [203] dataset is an exception to this, as it also contains information about interactive objects. Nevertheless, early experiments in the context of [S5] led to the conclusion that the recorded interactions are too chaotic to be automatically assigned to intentions for specific targets.

For this reason, two custom datasets with approaches to known targets were created by recording head pose and gaze direction with a sample rate of 60 Hz in an experimental study, where the participant's intention to approach a specific target was controlled by verbal or visual commands. The first dataset, hereinafter referred to as the *learning dataset*, contains randomly placed, virtual targets, which were displayed on a *Microsoft Hololens2* HMD [D15] in AR mode. During the recording, participants were instructed to keep approaching the sequentially highlighted targets. To compensate for the HMD's narrow field of view, participants were also provided with text-based directional cues on the HMD. To overcome the need for text-based cues and to get more genuine user behavior, a second dataset, referred to as the *evaluation dataset*, was created with fixed, real targets. In this case, the participants were instructed to approach a specific target using voice commands. Both datasets were published in [O8].

The learning dataset was partitioned, so that 70 % of the approaches are used for training, 20 % for validation, and 10 % for testing. To avoid negative side effects of the initial target search, which can be observed at the beginning of many approaches in the learning dataset, the first 0.5 s of each approach are discarded from the training data. The training was performed using a modified cross-entropy loss, where the training data samples are weighted, so that each target approach contributes equally to the total loss, independent of the duration of the approach. This idea is also reflected in the accuracy

measure

$$A = \frac{1}{N} \sum_{j=1}^{N} \frac{1}{D_j} \sum_{k=1}^{D_j} \delta \left(\underset{i}{\operatorname{argmax}} \left(\underline{\hat{p}}(j,k)_{[i]} \right), \underset{i}{\operatorname{argmax}} \left(\underline{p}(j,k)_{[i]} \right) \right) , \tag{8.1}$$

which sums over the number of approaches N with the individual approach durations D_j. In this expression, $\delta(\cdot,\cdot)$ is the Kronecker delta function and (j,k) are function arguments referencing to the probability vectors at time step k of approach j. During training, dropout is used as a regularization technique in the LSTM layers. Furthermore, the hyperparameters of each architecture were optimized using a random search, that samples from a grid of feasible values for the number of LSTM layers, the number of hidden states, and the dropout probability.

The training data assigns a single intention to each target approach. To achieve compatibility with the binary classifier, all approaches with the coordinates of the intended target are used as positive examples. The negative examples are generated by using the same set of approaches with the coordinates of the other targets. For the binary extended classifier, this data is augmented by randomly setting some of the elements in $\underline{u}_{\text{aux}}$ to the reserved value in order to learn the handling of different n. Furthermore, the order of the elements in $\underline{u}_{\text{aux}}$ is permuted. The multi-class classifier requires three data augmentation steps: First, the order of the targets is permuted to ensure that the network is invariant to integer target identifiers. Second, some input coordinates are randomly set to the reserved value to handle different n. Finally, data for the *no known target* class must be generated. For this purpose, the coordinates of one target are entirely deleted from the training dataset. Approaches to this target are then labeled with *no known target*. For the given training data with five coexisting targets, this results in $N = 4$. The last augmentation step is omitted if the multi-class classifier is trained without the *no known target* output. Detailed information about the training procedure can be found in [O7] and [S5].

8.1.4 Evaluation

The proposed NN architectures were implemented using *TensorFlow* [204]. For comparison, the existing Bayesian classifier from [186] was reimplemented using *scikit-learn* [205] with a Gaussian mixture with 50 components. Although the number of components is relatively high, no overfitting was observed. The second competitor in the following evaluation is the HAIR variant from [188]. As the original code is not available, a custom implementation with the improvements described in [S5] was created. In the following, all experiments were conducted with $N = 5$, except for the multi-class classifier with *no known target*, where $N = 4$ was used due to the required approaches with an unknown target.

Quantitative Evaluation

To compare the proposed NN architectures for intention estimation with each other and competing approaches, the accuracy measure (8.1) is evaluated on the test part of the learning dataset and the full evaluation dataset. Furthermore, the average remaining time of an approach, in which the target with the highest output probability is stable and corresponds to the intended target, is calculated. Hereinafter, this quantity is referred to as T_{cor}. With this definition, higher values for T_{cor} indicate less volatile intention estimates. Lastly, the average runtime of a single inference iteration is measured on an *Intel Core i7-11800H* CPU without GPU acceleration.

The resulting metrics are summarized in Table 8.1. From the data, we can see that HAIR is clearly inferior. All methods have in common that the accuracy on the learning dataset is lower than on the evaluation dataset. This effect can be explained by the reduced complexity and missing search phases in the evaluation dataset with fixed instead of random targets. The binary classifier and the extended binary classifier outperform the Bayesian classifier in terms of accuracy and T_{cor} on the learning

Estimator	Learning dataset (test part)		Evaluation dataset		Iteration time in ms	Parameters in NN
	Accuracy in %	T_{cor} in s	Accuracy in %	T_{cor} in s		
HAIR	48	2.08	68	2.49	**0.17**	–
Bayesian	61	2.28	**78**	2.62	0.45	–
Binary	**66**	**2.67**	76	2.48	1.24	27 253
Extended binary	64	2.58	**78**	**2.73**	1.12	87 361
Multi-class w/o *no known target*	59	2.39	75	2.60	0.66	21 573
Multi-class with *no known target*	58	2.16	72	2.34	0.66	20 805

Table 8.1: Performance of the proposed intention estimators compared to existing methods. Bold text highlights the best value for each metric. Adapted from [O7].

dataset. Regarding T_{cor}, this statement also holds for the extended binary classifier on the evaluation dataset, while its accuracy is on par with the Bayesian classifier. In contrast to that, the binary classifier is slightly worse than the Bayesian classifier on the evaluation dataset.

Both variants of the multi-class classifier perform worse than the Bayesian classifier with respect to the accuracy metric, independent of the considered dataset. However, the multi-class classifier without *no known target* yields a potentially higher value for T_{cor}, indicating a less volatile intention estimation compared to the Bayesian classifier. This behavior is advantageous for downstream applications. Furthermore, the data reveals that the multi-class classifier architecture can handle the *no known target* case, when a slight loss of accuracy is tolerable.

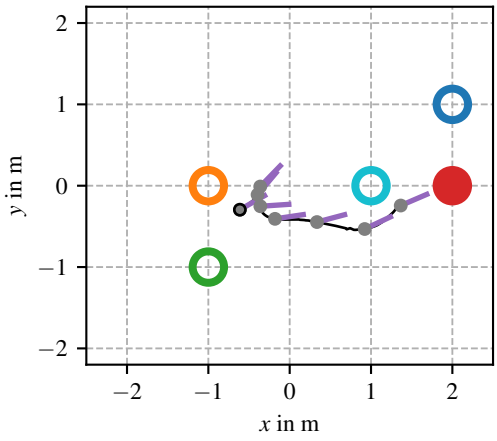
The average iteration times in Table 8.1 show that the proposed NN-based intention estimators are slower than the existing algorithms. Nevertheless, they can still be used without restrictions under real-time constraints, as the achievable update rate is far beyond the typical output rates of HMDs. Among the NN-based approaches, the binary and the extended binary classifier are the slowest.

Qualitative Evaluation

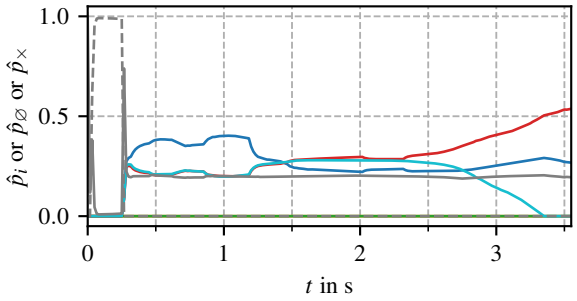
To get a better understanding of the estimator output, the example approach from Figure 8.3 is analyzed. In the presented scenario, it takes about $1.5\,\text{s}$ for the human subject to localize the commanded target in red, which is reflected in the limited translational movement at the beginning. After that, the subject starts approaching the red target, while avoiding a collision with the cyan target.

The resulting probability estimates in Figure 8.3b indicate that HAIR does not provide a clear prediction, although the situation is very clear at the end of the approach. In contrast to that, the Bayesian classifier in Figure 8.3c provides very definite predictions. However, they are quite volatile, especially between $1.3\,\text{s}$ and $2.5\,\text{s}$. This behavior, which results from the properties of the learned conditional probability densities [S5], is neither in line with the understanding of a human observer nor desirable for downstream systems.

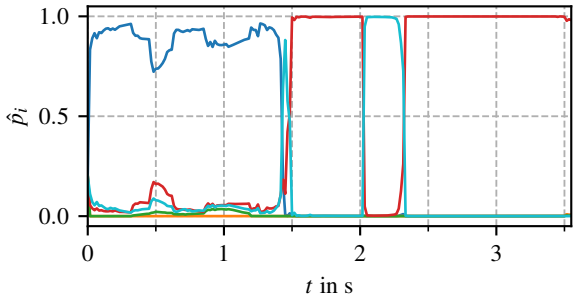
A better behavior is observed for the proposed NN-based intention estimators, whose variants are depicted in Figures 8.3d to 8.3f. In all cases, the red target is correctly predicted as the most probable target after $1.6\,\text{s}$. In detail, the binary classifier in Figure 8.3d favors the red target very early, which is beneficial for the accuracy metric, but premature from a human perspective as gaze and movement do not indicate the red target at the beginning of the approach. The extended binary classifier in Figure 8.3e does not show premature behavior, but the prediction is more erratic during the search phase. Compared to that, the multi-class classifier in Figure 8.3f provides a more plausible prediction, which is in line with the situational understanding of a human observer.



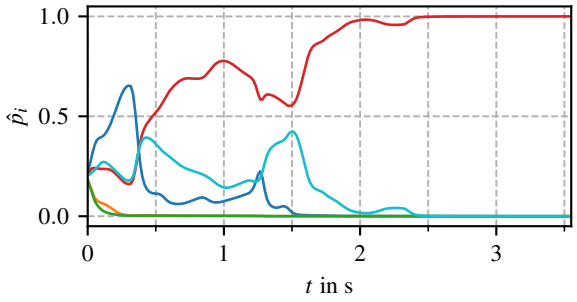
(a) 2D projection of the scenario, in which the human subject intends to interact with the target at the red circle. The gray dots along the user trajectory (black) are spaced 0.5 s apart. The purple bars indicate the gaze direction.



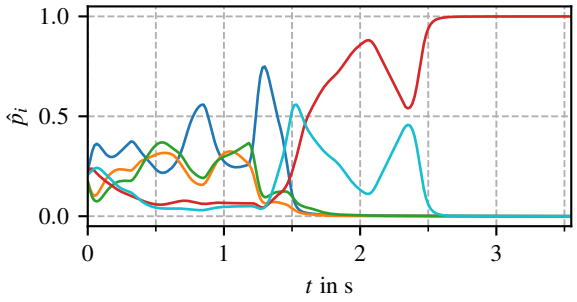
(b) HAIR. The solid gray line represents the probability $\hat{p}_\varnothing$ for *no known target*. The probability for *irrational user behavior* $\hat{p}_\times$ is drawn with a dashed gray line.



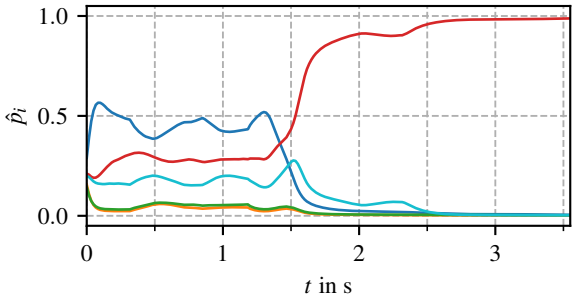
(c) Bayesian estimator.



(d) Binary classifier.

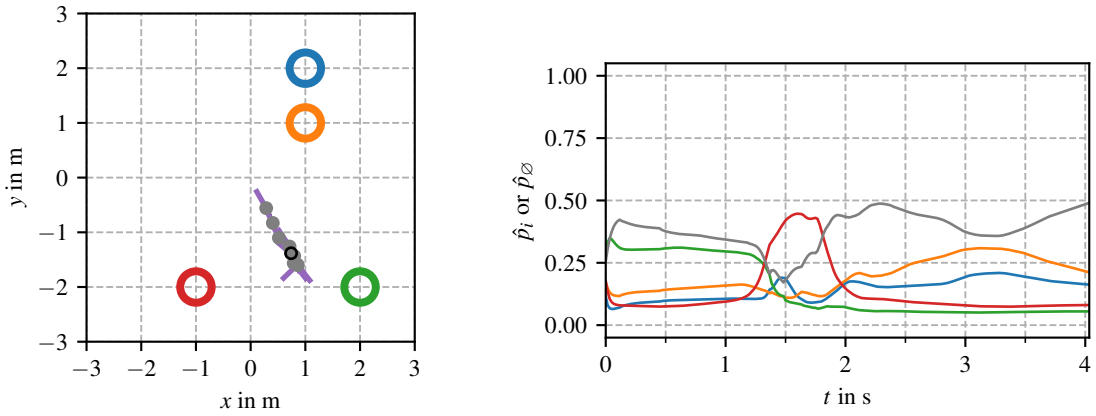


(e) Extended binary classifier.



(f) Multi-class classifier without the *no known target* output.

Figure 8.3: Intention estimates during an exemplary approach in an environment with five targets. The colors of the targets match across all subfigures. Adapted from [O7].



(a) 2D projection. The subject's target is unknown to the estimator and thus not drawn.

(b) Output probabilities of the multi-class classifier. The gray line corresponds to *no known target*.

Figure 8.4: Example scenario in which the subject does not approach any of the known targets. The symbols and colors are the same as in Figure 8.3. Adapted from [O7].

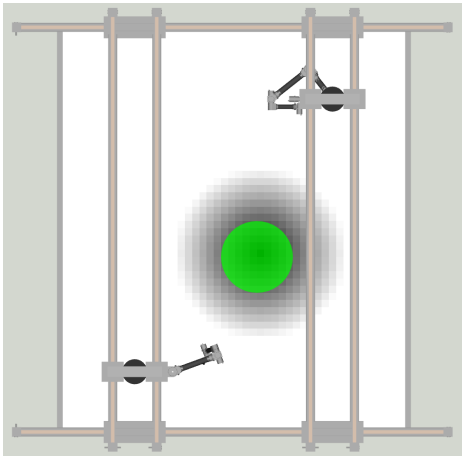
Figure 8.4 can be used to illustrate the behavior of the multi-class classifier with the *no known target* output. In the studied example, the subject pursues a target that is unknown to the estimator. At the beginning, the subject faces the green target, which is reflected in the high probabilities for the green target and *no known target*. Then, the subject turns clockwise, causing a temporary probability increase for the red target. After that, the subject walks in a direction that does not correspond to any of the known targets, causing the *no known target* probability $\hat{p}_\varnothing$ to become dominant. Despite that, the probabilities for the orange and blue targets remain increased, as the subject may still adjust the plan in favor of one of these targets. Therefore, the *no known target* output behaves in a completely plausible manner from the perspective of a human observer.

8.2 Intention-Based Motion Planning

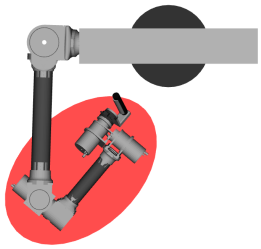
HapticGiant must react to the estimated user intention. This can be realized using the novel intention-based motion planning pipeline from [S6], which was supervised as part of this thesis. The pipeline consists of three parts: First, a specialized path planning approach is presented in Section 8.2.1. Second, the planned path is converted into a trajectory in Section 8.2.2. Finally, Section 8.2.3 deals with the question of how the output of the intention estimation can be transformed into a suitable input for the path planning. The presentation is completed with a preliminary evaluation in Section 8.2.4.

8.2.1 Path Planning

The goal of the path planning is to find a path in HapticGiant's eight-dimensional configuration space, that connects the current configuration $\underline{q}_0^\mathsf{T} = (\underline{q}_0^{\mathrm{pT}} \quad \underline{q}_0^{\mathrm{mT}})$ to a desired configuration $\underline{q}_{\mathrm{ref}}^\mathsf{T} = (\underline{q}_{\mathrm{ref}}^{\mathrm{pT}} \quad \underline{q}_{\mathrm{ref}}^{\mathrm{mT}})$, while avoiding joint limits, self-collisions and collisions with the user. As a side constraint, the path should be as short as possible and try to avoid regions in the workspace that are likely to be occupied by the user in the near future. In addition, the path planning must be able to react to environmental changes in real time, as the user's position and intention are constantly changing. This setup is illustrated in Figure 8.5a. The current user pose is defined by a green cylinder, which must not be penetrated by the manipulator. Regions in the 2D projection of the Cartesian workspace that shall be penalized, such as the area around the current user position, are determined using a planar costmap.



(a) Visualization of the initial (bottom left) and goal configuration (top right) with the user at the position of the green cylinder. The shaded area around the user marks costmap regions with increased costs.



(b) Approximation of the manipulator geometry for collision checking. The resulting collision ellipsoid is drawn in red.

Figure 8.5: 2D projections of HapticGiant to illustrate the models and simplifications used to solve the path planning problem. Taken from [S6].

Solving the resulting problem is non-trivial due to the high dimensionality and the complex constraints. Therefore, additional simplifications and assumptions are required:

- Self-collisions are impossible due to the joint limits and the design of HapticGiant.
- For the remaining collisions, the manipulator geometry is reduced to the elements after manipulator joint 3, as these are the only parts that can collide with the user by design. This geometry is projected onto the ground plane and approximated by an enclosing collision ellipsoid as illustrated in Figure 8.5b.
- With this simplification, a user-manipulator collision can be detected by checking whether the footprint of the user cylinder intersects with the collision ellipsoid. A collision causes the flag $c_{\text{col}} \in \{0, 1\}$ to be set.
- The configuration cost c_{cfg} is obtained by integrating the costmap over the area of the manipulator's collision ellipsoid for a given configuration.
- The resulting paths should be short in terms of their expected travel time. For this reason, the cost c_{dist} is defined as the weighted Euclidean distance between two samples in the eight-dimensional configuration space. The weights are the inverse of the maximum joint velocities to normalize the distance with respect to the velocity capabilities of the different joints.
- The total cost of a path segment is $c_\Sigma = w_{\text{col}}\, c_{\text{col}} + w_{\text{cfg}}\, c_{\text{cfg}} + w_{\text{dist}}\, c_{\text{dist}}$ with the tunable weights w_{cfg}, w_{dist}, and w_{col}.

Despite these simplifications, common path planning algorithms, such as RRT* or A*, proved to be too slow for the high-dimensional planning problem. Because of this, a new method with four steps, that are tailored to HapticGiant and the rendering of DTs, is proposed:

1. Split the eight-dimensional configuration space into a PPU configuration space with dimension two and a manipulator configuration space with dimension six.
2. Plan a manipulator motion that is free of self-collisions by considering only the desired start and end configuration, $\underline{q}_0^{\text{m}}$ and $\underline{q}_{\text{ref}}^{\text{m}}$, of the manipulator. In general, any planning method can be used for this purpose. For HapticGiant, it is sufficient to use linear interpolation with s samples due to the manipulator design without self-collisions.

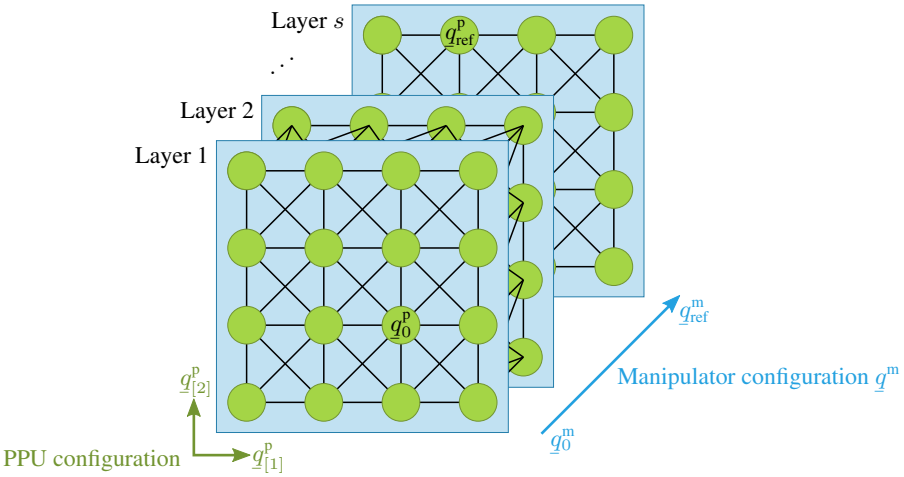


Figure 8.6: Illustration of the roadmap used for the proposed path planning method. The green nodes represent a PPU configuration, the blue layers a manipulator configuration.

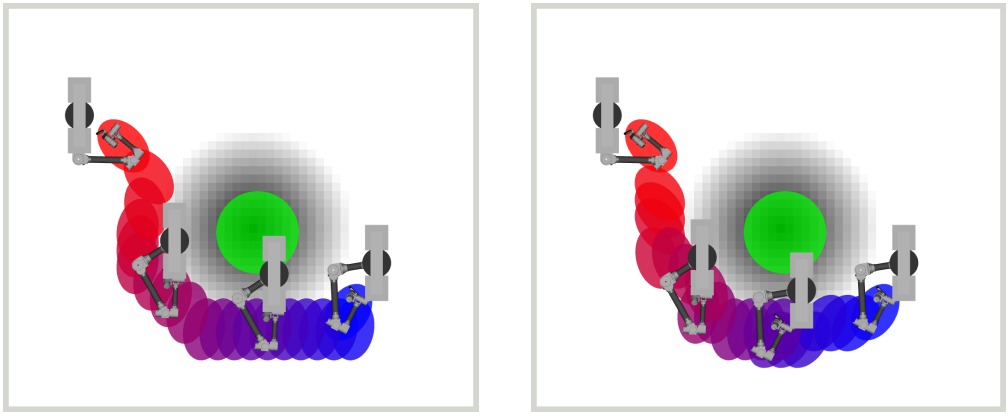
3. Plan a path for the PPU between q^{p}_0 and $q^{\mathrm{p}}_{\mathrm{ref}}$. To do this, create a 2D roadmap for the PPU configuration space and stack s of these roadmaps as shown in Figure 8.6. Each layer in this stack corresponds to a configuration in the manipulator path. The layers are then connected with edges that connect adjacent PPU configurations in adjacent layers. The resulting 3D roadmap can be fed into a lazy A* planning algorithm. As a result, the manipulator movement from item 2 can be continued, paused, or even reversed while the PPU is in motion. Likewise, the PPU can be paused while the manipulator is moving.
4. Extract the path for all eight DOF by looking up the corresponding manipulator configurations for each node in the output of the A* algorithm.

Evaluating the 3D roadmap becomes computationally expensive as s becomes large. If this is the case, the dimension of the roadmap can be reduced to two by assigning an individual manipulator configuration to each PPU configuration. This can be achieved using a heuristic that linearly maps the distance between the current and the desired PPU configuration to a configuration in the manipulator path. For instance, a PPU configuration, whose distance to the desired PPU configuration is half of the distance between the initial and the desired PPU configuration, is assigned the manipulator configuration that is halfway between the initial and the desired manipulator configuration.

The outcome of the proposed planning approach and its variant with the heuristic are visualized in Figure 8.7. Both approaches successfully find a collision-free path. Moreover, the algorithm provides an appropriate trade-off between avoiding regions with high costs and minimizing the path length. A study that goes beyond a single example can be found below in Section 8.2.4.

8.2.2 Path Tracking

The output of any path planning method is a sequence of PPU and manipulator configurations without time information. In order to track the desired path with HapticGiant, suitable trajectory commands must be generated from the path information. An additional challenge in the given context is that the resulting trajectory should utilize the full joint velocity range to achieve short repositioning times. The resulting problem can be solved using optimization methods. However, the computational effort of these methods is too high for real-time applications. For this reason, this chapter proposes a simple, yet effective method for generating the desired trajectory commands from the path information.



(a) Without heuristic. The manipulator configuration can be kept in a fixed configuration for several consecutive samples.

(b) With heuristic. The manipulator configuration is rigidly coupled to the PPU motion.

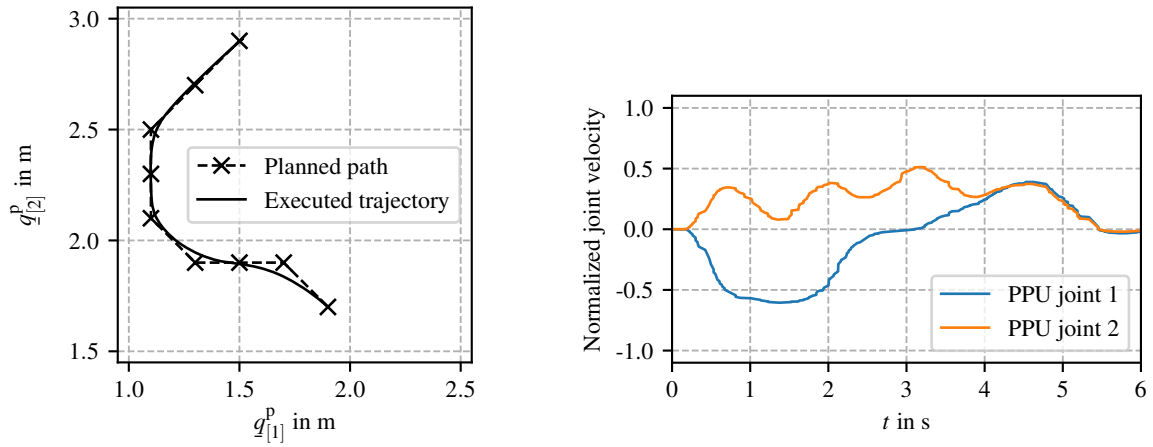
Figure 8.7: Exemplary results of the proposed path planning method. The initial and final configuration correspond to the blue and red collision ellipsoids, respectively. For the sake of clarity, some of the intermediate configurations are not drawn. Taken from [S6].

In preparation, an auxiliary configuration $\underline{q}_{\text{aux}}$ with distance D from the start configuration of the path is calculated via linear interpolation between the path samples. The distance metric for D, whose value can also be interpreted as a lookahead distance, is the same weighted Euclidean distance as in Section 8.2.1. Then, *ruckig* [206] is used to determine a time-optimal trajectory between the current manipulator state, including non-zero joint velocities and accelerations, and an auxiliary joint state with $\underline{q} = \underline{q}_{\text{aux}}$, $\dot{\underline{q}} = \underline{0}$, and $\ddot{\underline{q}} = \underline{0}$. Due to the properties of ruckig, the resulting synchronous point-to-point trajectory is time-optimal while respecting joint velocity, acceleration, and jerk limits. If the trajectory was executed from the current state all the way to the auxiliary state, the manipulator would come to a full stop at the auxiliary state, causing a discontinuous path tracking motion. To avoid this, the described process is repeated with updated values for the initial state and $\underline{q}_{\text{aux}}$ shortly after the previously generated trajectory is commanded to HapticGiant. This way, the final, static configurations of the commanded trajectories are continuously postponed, unless the end of the reference path is reached. Note, that this method is also robust to the frequently expected changes of the reference path.

An exemplary resulting trajectory is shown in Figure 8.8. Because of the lookahead, the resulting trajectory in Figure 8.8a does not track the reference path exactly. Instead, smoothing is observed. This is actually desired behavior, as discontinuous direction changes at non-zero velocities would require infinite joint accelerations. The resulting joint velocities in Figure 8.8b show that the resulting motion is continuous.

8.2.3 Processing of the Intention Estimation Output

The presented methods for path planning and path tracking require that the goal configuration $\underline{q}_{\text{ref}}$ is known. If the currently active DT is deterministic, $\underline{q}_{\text{ref}}$ can be obtained by simply converting the desired target pose $\mathbf{T}_{T_i}^{W}$ into the joint configuration $\underline{q}_{T_i}^{\mathsf{T}} = (\underline{q}_{T_i}^{\text{pT}} \quad \underline{q}_{T_i}^{\text{mT}})$ using the inverse kinematics solution from Section 4.1.4 and the assumptions from Section 7.1.3. The considered scenario with the exemplary situation from Figure 8.1 is different because the DT, that will be active next, is not known deterministically. Therefore, $\underline{q}_{\text{ref}}$ depends on the user's intention and must be derived from the n available target configurations $\underline{q}_{T_1}, \ldots, \underline{q}_{T_n}$ and the intention estimation output $\hat{\underline{p}} \in \mathbb{R}^n$. Without claiming completeness, two strategies are proposed for this task:



(a) Cartesian view of the desired and the resulting motion.

(b) Utilization of the joint velocities.

Figure 8.8: Example of the proposed path tracking method with $D = 0.3$. For simplicity, only the PPU motion is considered here.

1. Use the target with the highest probability as goal configuration, i.e.,

$$\underline{q}_{\mathrm{ref}} = \underline{q}_{T_{i_{\max}}} \quad \text{with} \quad i_{\max} = \underset{i \in \{1,\dots,n\}}{\operatorname{argmax}} \, \hat{p}_{[i]} \,. \tag{8.2}$$

   In the example from Figure 8.1, this strategy yields $\underline{q}_{\mathrm{ref}} = \underline{q}_{T_2}$. If the *no known target* output is available, (8.2) can be easily extended. In this case, the goal configuration for *no known target* can be defined according to the needs of the specific application.
2. The first method is suitable when the estimated probability vector has a distinct maximum. However, it results in volatile behavior when two targets are almost equally likely. To mitigate this issue, the estimated probabilities can be taken into account with the goal configuration

$$\underline{q}_{\mathrm{ref}} = \frac{1}{\sum_{i=1}^{n} \hat{p}_{[i]}} \sum_{i=1}^{n} \hat{p}_{[i]} \, \underline{q}_{T_i} \,,$$

   which can be derived by finding the joint configuration minimizing the expected value of the squared distance to the next target. Again, the distance metric used corresponds to the weighted Euclidean distance from Section 8.2.1. In Figure 8.1, this method returns a configuration that is located somewhere slightly to the right of target 2.

8.2.4 Evaluation

The proposed intention-based motion planning algorithms were implemented in Python with the aid of *graph-tool* [207] and *OMPL* [208] for the underlying calls to A* and RRT*, respectively. All evaluations were conducted in simulation and separately from the intention estimation. This implies that the processing of the intention estimation output from the previous section was not systematically evaluated, as the results depend heavily on the actual behavior of the intention estimation and the chosen scenario.

In the following experiments, the roadmap is configured to have a minimum node distance of 0.2 according to the weighted Euclidean distance. The number of manipulator configuration samples s is determined dynamically, so that the distance measure does not exceed 0.2. The weights of the cost function are set to $w_{\mathrm{col}} = 1000$, $w_{\mathrm{cfg}} = 0.1$, and $w_{\mathrm{dst}} = 1$. For the path tracking, $D = 0.6$ is chosen

Planning method	∅ distance cost $\sum c_{\text{dist}}$	∅ configuration cost $\sum c_{\text{cfg}}$	Total number of collisions	∅ planning duration in s
RRT*	2.69	16.16	4	10.76
Proposed method w/o heuristic	2.95	17.50	2	0.18
Proposed method with heuristic	3.03	43.48	18	0.06

Table 8.2: Quantitative evaluation of the proposed path planning algorithm for random queries. Adapted from [S6].

as a good compromise between fast and accurate tracking behavior. Lastly, the user is modeled as a cylindrical obstacle with a diameter of $1\,\mathrm{m}$ in combination with a user-aligned, conical costmap with a peak cost of 100 and a diameter of $2.4\,\mathrm{m}$.

Open-Loop Evaluation of the Path Planning

In the first part of the evaluation, the path planning algorithm is evaluated in an open-loop setup on an *Intel Core i5-4590* CPU. For this purpose, 1000 queries with uniformly drawn, valid start and end configurations are generated. In each query, the user position is randomly drawn from a workspace-centered normal distribution. Its variance is chosen so that $97.5\,\%$ of the samples fall within HapticGiant's workspace. The queries are sent to the variants of the proposed path planning algorithm from Section 8.2.1. Furthermore, a standard RRT* planner is used as a baseline algorithm. For a fair comparison, the returned paths are linearly interpolated and resampled with a constant step size.

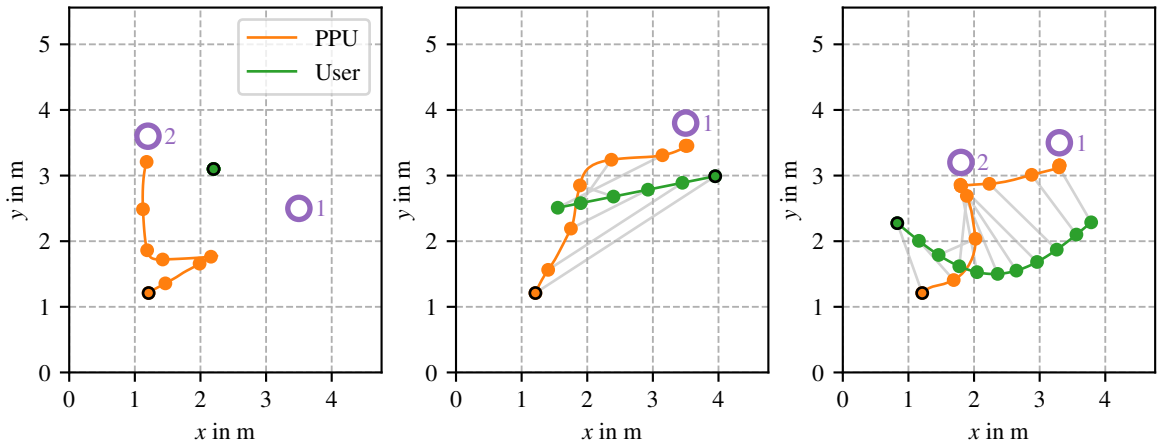
The resulting metrics are listed in Table 8.2. From there, it is evident that RRT* achieves the lowest average distance and configuration costs with a very low number of collisions. However, the average planning duration reveals that RRT* is much too slow for scenarios requiring dynamic replanning. The proposed path planning method without the heuristic is about 50 times faster, while the distance and configuration costs are only slightly increased. The number of detected collisions is comparable to RRT*. By enabling the proposed heuristic, the average planning duration can be further reduced by a factor of three, but this speedup comes at the cost of greatly increased configuration costs, indicating that collisions are more likely to occur when the user is moving.

Closed-Loop Evaluation with Mock Intention Estimates

The second part of this evaluation aims at demonstrating the dynamic, closed-loop behavior of the motion planning setup. For this purpose, the output of a mock intention estimation is continuously processed in a pipeline, whose steps correspond to Section 8.2.3, Section 8.2.1, and Section 8.2.2, respectively. This pipeline runs in parallel with the real-time system simulation from Chapter 5.

Three example scenarios are presented in the following:

- In Figure 8.9a, a static user can choose between two targets. Initially, target 1 has the highest probability of being the next target. For this reason, HapticGiant moves towards target 1. After $2\,\mathrm{s}$, the predicted user intention shifts to target 2. As a result, the manipulator is redirected to a path that does not interfere with the user.
- In Figure 8.9b, the user is on a collision course with the manipulator. The motion planning avoids the impending collision by successfully executing an evasive movement.
- The final scenario in Figure 8.9c contains a moving user, whose intention shifts from target 2 to target 1 after $7\,\mathrm{s}$. As expected, the continuously updated motion plan causes HapticGiant to approach target 2 with a small detour to avoid a user collision before pursuing target 1 due to the new intention.



(a) Fixed user with two targets. **(b)** Moving user with one target. **(c)** Moving user with two targets.

Figure 8.9: Exemplary behavior of the full motion planning pipeline with one or two targets (purple circles). The points on the trajectories are placed 1 s apart. In addition, black circles indicate the starting points and gray lines connect points with matching timestamps.

8.3 Discussion

In the first half of this chapter, three variants of an LSTM-based NN were presented as a novel method for estimating the intention of a human user in varying scenarios. The evaluation with a completely separate dataset shows that the trained models generalize well. In particular, the proposed *extended binary classifier* can outperform HMM-based approaches without relying on manual feature engineering. Moreover, the number of targets does not need to be known at training time, which is an advantage over existing NN-based methods. As an extension, the proposed *multi-class classifier* provides special handling for users that do not intend to approach any of the known targets.

In the second half, a custom intention-based motion planning algorithm was presented, whose goal is to position HapticGiant's end effector at the position of the user's intended target, ideally before the user arrives there. As part of this, two processing strategies for the estimated intentions, a custom path planning method for high-dimensional configuration spaces, and a path tracking method based on *ruckig* were introduced. The proposed path planning algorithm achieves update rates around 5 Hz. As a result, collisions with static and dynamic obstacles can be avoided in real time, while the remaining performance metrics are comparable to the much slower RRT* algorithm.

The results of Section 8.2.4 in conjunction with Section 8.1.4 suggest that the proposed intention estimation and motion planning algorithms are promising technologies. However, further research is required for the successful integration into an ETHD, that is capable of simultaneously rendering multiple DTs. For example, the current state of the intention estimation cannot detect users deliberately trying to trick the system, and the physical joint limits of HapticGiant restrict the PPU motion to speeds lower than comfortable human walking speeds. Additionally, extra safety measures, such as local collision avoidance and mitigation, are necessary to guarantee the well-being of the user.

Independently, the methods in this chapter can be used in applications beyond haptic displays. For instance, the intention estimation algorithm can be ported to the problem of motion compression or to collaborative robots. Similarly, the motion planning pipeline can be applied to manipulators on mobile platforms with little modification.

In future work, the intention estimation may be improved by using more recent network architectures, such as *PointNet* [209] or *Point Transformer* [210]. Futhermore, it should be investigated whether the

observed discrepancy between the quantitative and qualitative results of the multi-class classifier is a limitation of the chosen accuracy metric. Another research direction is the integration of additional input data sources, such as hand pose, posture, and locomotion status.

On the motion planning side, the closed-loop behavior with the full intention estimation should be studied in real scenarios. This way, the performance of the not yet evaluated intention processing strategies can be assessed. Currently, the path tracking does not fully utilize the maximum joint ratings. For this reason, more effort should be put into the path-to-trajectory conversion. Finally, the primitive conical costmap should be replaced by a more sophisticated model. For example, the target positions and the intention estimation output can be considered to proactively avoid evasive motions resulting from the intersection of user and manipulator paths.

CHAPTER
9

Conclusions

Contents

> Success is not final, failure is not fatal:
> it is the courage to continue that counts.
>
> WINSTON CHURCHILL (1874 – 1965)

The advent of consumer-grade augmented reality (AR) and virtual reality (VR) devices has ushered in a new era of immersive telepresence, where users can engage with virtual and remote environments in increasingly realistic ways. Visual immersion is now highly advanced, thanks to decades of research in motion tracking, display technology, and photo-realistic graphics. This positions haptic feedback as a critical component for the next leap regarding the level of immersion. In particular, kinesthetic haptic feedback is a promising approach that enables a user not only to see, but also to *feel* and *manipulate* virtual objects. Despite its importance, the technology required for this endeavor remains comparatively underdeveloped, especially when it comes to applications that require a large range of motion.

As a potential solution to this issue, the kinesthetic haptic interface *HapticGiant* was developed in this thesis. The resulting prototype has a room-sized workspace, enables the natural locomotion of the user, and achieves unmatched dexterity. Special emphasis was put on the high-level control architecture, which enables the rendering of arbitrary serial kinematic chains. Thereby, HapticGiant can be used to generate haptic feedback that facilitates the perception of a tangible *digital twin (DT)* without having to worry about mechanical system limitations. In the big picture, this allows HapticGiant to be readily used as an *encountered-type haptic display (ETHD)* in scenarios with a single DT. Beyond that, this thesis provides a perspective on how HapticGiant can handle scenarios with multiple, coexisting DTs to further close the sensory gap in telepresence applications.

Next, Section 9.1 summarizes the contributions of this thesis, followed by Section 9.2, which offers an outlook on future research and applications.

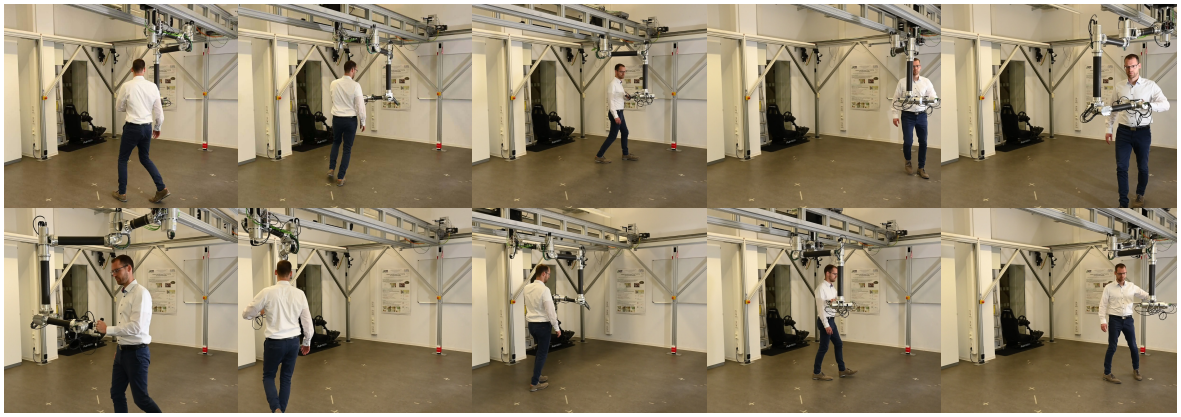
9.1 Contributions

In the first part of this thesis (Chapters 3 to 5), we covered the conception and realization of HapticGiant. To this end, the following key contributions were made:

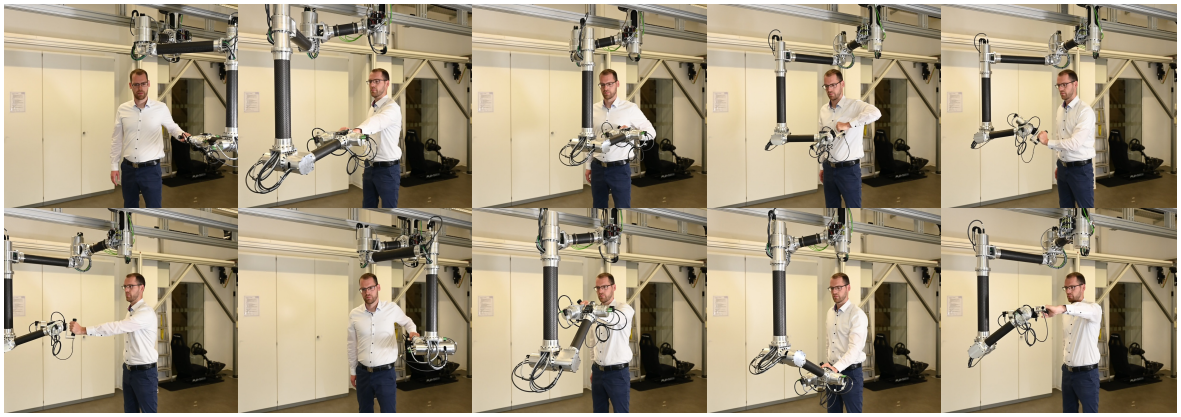
- The **kinematic topology of HapticGiant** was conceived based on a broad set of requirements, existing designs from the literature, and practical experience. The chosen topology consists of a gantry crane-like prepositioning unit (PPU) with two degrees of freedom (DOF) and a manipulator with six DOF in a user-enclosing configuration. Link lengths, joint sizes, and transmission ratios were chosen using a novel **optimization-driven dimensioning method**, whose objective is to maximize the coverage of reachable poses, velocities, and accelerations with respect to an anthropometric model of the human arm. The resulting manipulator design was successfully validated using a dataset with activities of daily living (ADL), resulting in coverage values of $94.0\,\%$, $93.7\,\%$, and $90.1\,\%$ for position, velocity, and acceleration, respectively.
- Based on this concept, a **physical prototype** was developed using the methods of System Generation Engineering (SGE). After manufacturing, the moving parts, weighing $36.8\,\mathrm{kg}$, span a total link length of $2.5\,\mathrm{m}$. Thanks to the optimized design and the inclusion of a slip ring in the manipulator construction, the system can handle a variety of hand poses in a very large workspace, including the examples in Figure 9.1. Other key features are the system-wide *EtherCAT* fieldbus, the torque sensors in the manipulator joints, and the 3D force-torque sensor located at the end effector. Wherever possible, the design uses off-the-shelf components to simplify maintenance and hardware expansion. Special attention has been paid to functional safety to ensure the user's well-being under all circumstances.
- On the software side, ***Real-Time Control Framework (RTCF)*** was developed as a novel tool to increase modularity, developer productivity, and interoperability while satisfying the performance requirements of the control system. In particular, RTCF is fully compatible with *Robot Operating System (ROS)* and benchmarks have proven its ability to handle control frequencies well beyond $2\,\mathrm{kHz}$ with low latency and jitter.
- In addition, HapticGiant comes with a **custom simulation environment** for system dynamics and sensor characteristics. Unlike existing solutions, the simulation considers effects such as joint flexibility and HapticGiant's velocity-controlled PPU. Furthermore, the simulation leverages the principle of maximum dissipation (PoMD) to efficiently simulate Coulomb friction in addition to linear friction effects, resulting in an average computation time of $64.8\,\mu\mathrm{s}$ for a full simulation update with a step size of $1\,\mathrm{ms}$.

In the second part of this thesis (Chapters 6 and 7), we dealt with suitable algorithms to enable and improve the haptic rendering of serial kinematic chains. This resulted in the following contributions:

- A novel **kinematic state estimator** based on the extended Kalman filter (EKF) was developed to improve the available information on joint positions, velocities, and accelerations by incorporating gyroscope and accelerometer measurements. The inclusion of bias states for each inertial sensor axis enables arbitrary inertial sensor configurations without the need for sensor calibration. Simulations with an eight DOF manipulator were performed to compare the estimator with approaches from the literature, showing superior performance. With an average update time of less than $560\,\mu\mathrm{s}$ per iteration, the estimator is suitable to be used in environments with real-time requirements. These results were also confirmed experimentally on a real selective compliance assembly robot arm (SCARA) manipulator with two DOF. On the downside, the integration of the estimator into HapticGiant failed due to an unexpected mismatch between accelerometer and encoder readings.
- Motivated by the flexibility of the proposed state estimator, the question of the ideal sensor configuration was addressed. In this context, an efficient **method for calculating the maximal**



(a) Natural user locomotion while haptic rendering is active. The time between two consecutive frames is 1 s.



(b) A selection of user-reachable hand poses.

Figure 9.1: Examples of the user interaction with HapticGiant. Adapted from [O1].

absolute readings of triaxial inertial sensors at a given position was presented, alongside an approximate observability measure based on the observability Gramian. By combining these methods, it is possible to identify and rank feasible sensor configurations, leading to a procedure for the **optimal placement of inertial sensors**. This procedure was successfully validated on a SCARA manipulator and HapticGiant by comparing the predicted estimation quality with results from the kinematic state estimator mentioned above. Ultimately, the proposed procedure has resulted in the sensor configuration that is installed on HapticGiant.

- HapticGiant must provide haptic feedback that matches the kinematic and dynamic properties of the DT. Therefore, a novel **hierarchical optimal force controller** was developed to render arbitrary serial kinematic chains. In this controller, the rendering is formulated as a set of prioritized equality and inequality tasks, which take care of the kinematic coupling, manipulator joint limits, singularity avoidance, Cartesian limits, and DT joint limits. As a result, any serial kinematic chain can be presented to the user through a corresponding non-linear admittance that behaves in a dynamically consistent manner when a hardware limitation is reached. This in turn ensures that the motion of the DT aligns with the user's intuition. The controller is complemented by a custom PPU optimization strategy and suitable low-level controllers. In total, the average computation time for a DT with six joints is 529 µs, indicating feasible update rates of at least 1 kHz.
- The proposed control scheme was successfully put to **test in real-world experiments**. As part of these, the *Haptify* benchmarking method was utilized. The resulting metrics show superior workspace volume and stiffness rendering, with free-space forces comparable to commercial

devices with considerably smaller workspaces. However, these performance advantages come at the cost of significantly higher vibration levels.

The last part of this thesis (Chapter 8) explored the seamless rendering of multiple DTs, resulting in the following contributions:

- A novel approach for **intention estimation** was developed to predict which target, i.e., which DT, the user is likely going to approach. To achieve this, the available sensor and target data is fed into a neural network (NN) with a long short-term memory (LSTM) architecture after an ego-centric transformation, yielding a probability distribution over the targets. Due to a lack of suitable datasets in the literature, **two datasets with intention data** were created. The performance of the intention estimation was found to be comparable to existing methods, but without the need for manual feature engineering, a fixed number of targets, or predefined target positions.
- Based on the intention estimation, a **motion planning pipeline** was developed to pave the way for rendering multiple DTs in an ETHD setup. As part of this, HapticGiant's eight-dimensional configuration space was separated into two subspaces, which were then reconnected using a novel extension to roadmap-based path planning. Compared to RRT*, this reduces the average computation time by a factor of about 50 without a significant loss of path quality. The subsequent path tracking was realized using *ruckig* in combination with a constant lookahead to achieve fast reaction times and smooth trajectories. Closed-loop tests in simulation confirmed the desired behavior. Moreover, two approaches for the conversion of the intention estimation output were presented.

9.2 Future Research

Throughout this thesis, future research directions were already discussed at the end of each chapter. Nevertheless, this section will recall aspects that the author considers to be crucial for the further development and the performance of HapticGiant. In addition, a brief outlook on potential research topics beyond the scope of HapticGiant will be given.

Practical experience with the prototype suggests that the low-level controllers are currently the main bottleneck affecting haptic quality. In particular, vibrations and stick-slip behavior limit the achievable level of transparency. As a countermeasure, the low-level controllers may be enhanced by adding more feed-forward terms, which are based on the available system models, or by incorporating joint torque feedback. Additionally, the controller performance may be improved by taking into account joint flexibility and, if relevant, link flexibility. This also requires extensive system identification and the addition of further effects to the simulation.

Another way to improve HapticGiant's control performance is to complete the integration of the proposed kinematic state estimation. In order to benefit from its full potential, the mismatch between the accelerometer and encoder readings must be resolved first. Next, a partial, initial calibration of the inertial sensors with respect to biases and scale-factor errors appears to be a simple yet effective method for further reducing the estimation errors, from which the low-level controllers will benefit.

At a higher level, the current limitations of the hierarchical optimal force controller from Section 7.3.3 need to be addressed. This includes avoiding infeasible manipulator trajectories and improving the limit switching strategy to avoid oscillatory behavior. Beyond that, a collision mitigation strategy can be implemented as a measure to improve machine safety, ideally as a separate task in the hierarchical quadratic programming (HQP) formulation.

The most important step towards a fully operational demonstrator that can render multiple DTs in an ETHD setup is a closed-loop test of the proposed motion planning strategy in conjunction with the presented variants of the intention estimation. In particular, this test will shed light on how the proposed methods for processing the intention estimation output perform in practice. Given the typical human walking speed, the repositioning process will likely be too slow to provide a seamless user experience in many cases. To improve this situation, the intention estimation may be enhanced to the point where its multi-class classifier variant achieves the same performance as the other variants. Furthermore, the path tracking may be improved to gain a higher utilization of the joint velocity range. If the system is still too slow, the goal of simultaneously rendering multiple DTs must also be taken into account during the design of the target environment (TE) or in future hardware iterations. In any case, high-speed motion demands enhanced safety precautions and tight switching between rendering and repositioning, making a future integration of the path or trajectory tracking into the HQP formulation an attractive goal.

Depending on the insights gained from the above steps, HapticGiant's hardware may need to be revised. If this is the case, the design should be refined as well by integrating the gained knowledge into the proposed optimization-driven design method.

From an application perspective, HapticGiant is on the verge of being ready for many different practical use cases. For instance, the system can be coupled with state-of-the-art haptic rendering software to enable a multitude of existing haptic applications. Moreover, HapticGiant opens up new perspectives on classical teleoperation by enabling not only manipulator movements but also platform movements over large distances. Thanks to the intrinsic handling of the DT joint limits with regard to velocity and acceleration, the presented hierarchical force control scheme can also be used for teleoperating *sluggish* systems, such as the excavator shown in Figure 1.3b. Returning to the overall goal of highly immersive telepresence, HapticGiant giant's end effector can be replaced with a tool turret, which is equipped with different props. As a result, the realistic rendering of additional objects, such as walls, buttons, and other rigid items, becomes possible without any need for disrupting the user experience.

Beyond the scope of HapticGiant, the knowledge and experience gained in this thesis can contribute to the development of other, potentially very-large-scale kinesthetic haptic interfaces. In particular, the proposed hierarchical optimal force controller as well as the optimization-driven design method are adaptable to other requirements and kinematic topologies. An interesting extension is the addition of bimanual manipulation capabilities, although this introduces additional complexity, especially when the ability for natural user locomotion must be maintained. Regarding potential applications, the achievable level of immersion in the general context of very-large-scale ETHDs should be studied in detail. This will not only identify potential improvements, such as the addition of tactile feedback, but also use cases where devices such as HapticGiant provide the greatest benefit.

APPENDIX A

Partial Derivatives of the Kinematic Equations

This chapter is based on results presented in the author's publication [O2, O3].

In this chapter, the theory from Sections 2.2 and 2.3.2 is used to derive the first- and second-order partial derivatives of quantities that occur in the forward kinematics of an articulated body up to acceleration level. Hereinafter, the skew-symmetric matrix of the joint rotation axes is abbreviated as $\mathbf{R}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} = \left[\underline{r}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}}\times\right]$. Likewise, $\mathbf{T}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} = \left[\underline{t}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}}\times\right]$ is defined for the joint translation axes.

Theorem A.1:
The partial derivative of the joint rotation matrix $\mathbf{C}_{\beta_j}^{\epsilon_{j-1}}$ with respect to the joint configuration $\underline{q}_{[j]}$ is

$$\frac{\partial \mathbf{C}_{\beta_j}^{\epsilon_{j-1}}}{\partial \underline{q}_{[j]}} = \mathbf{R}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \mathbf{C}_{\beta_j}^{\epsilon_{j-1}} \,.$$

Proof. The rotation matrix $\mathbf{C}_{\beta_j}^{\epsilon_{j-1}}$ can be converted to axis-angle notation with the axis $\underline{r}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}}$ and the angle $\alpha_{\epsilon_{j-1}\beta j} = \underline{q}_{[j]}$. Therefore, $\underline{\omega}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} = \dot{\underline{q}}_{[j]}\underline{r}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}}$. With (2.2), this yields

$$\frac{\partial \mathbf{C}_{\beta_j}^{\epsilon_{j-1}}(\underline{q}_{[j]})}{\partial t} = \mathbf{\Omega}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \mathbf{C}_{\beta_j}^{\epsilon_{j-1}} = \dot{\underline{q}}_{[j]} \mathbf{R}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \mathbf{C}_{\beta_j}^{\epsilon_{j-1}} \,.$$

Direct application of the chain rules results in

$$\frac{\partial \mathbf{C}_{\beta_j}^{\epsilon_{j-1}}(\underline{q}_{[j]})}{\partial t} = \frac{\partial \mathbf{C}_{\beta_j}^{\epsilon_{j-1}}(\underline{q}_{[j]})}{\partial \underline{q}_{[j]}} \dot{\underline{q}}_{[j]} \,.$$

Comparing both equations yields the desired result. As defined in Section 2.3.1, $\underline{r}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} = \underline{0}_3$ is true for prismatic joints. Hence, the found expression is also correct for prismatic joints. □

Lemma A.1:
The partial derivative of the joint rotation matrix $\mathbf{C}_{\epsilon_{j-1}}^{\beta_j}$ with respect to the joint configuration $\underline{q}_{[j]}$ is

$$\frac{\partial \mathbf{C}_{\epsilon_{j-1}}^{\beta_j}}{\partial \underline{q}_{[j]}} = -\mathbf{R}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \mathbf{C}_{\epsilon_{j-1}}^{\beta_j} .$$

Proof. We obtain

$$\frac{\partial \mathbf{C}_{\epsilon_{j-1}}^{\beta_j}}{\partial \underline{q}_{[j]}} = \mathbf{R}_{\beta_j\epsilon_{j-1}}^{\beta_j} \mathbf{C}_{\epsilon_{j-1}}^{\beta_j} .$$

by swapping ϵ_{j-1} with β_j in Theorem A.1. Applying (2.1) directly yields the desired result. □

Theorem A.2:
The partial derivatives of the angular velocity in the body frame $\underline{\omega}_{W\beta_j}^{\beta_j}$ with respect to the joint configuration $\underline{q}$ and its derivatives $\dot{\underline{q}}$ and $\ddot{\underline{q}}$ can be calculated using the recursive expressions

$$\frac{\partial \underline{\omega}_{W\beta_j}^{\beta_j}}{\partial \underline{q}_{[i]}} = \begin{cases} \mathbf{C}_{\beta_{j-1}}^{\beta_j} \frac{\partial \underline{\omega}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \underline{q}_{[i]}} , & \text{if } i < j \\ -\mathbf{R}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \underline{\omega}_{W\beta_j}^{\beta_j} , & \text{if } i = j \\ \underline{0} , & \text{if } i > j \end{cases} ,$$

$$\frac{\partial \underline{\omega}_{W\beta_j}^{\beta_j}}{\partial \dot{\underline{q}}_{[i]}} = \begin{cases} \mathbf{C}_{\beta_{j-1}}^{\beta_j} \frac{\partial \underline{\omega}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \dot{\underline{q}}_{[i]}} , & \text{if } i < j \\ \mathbf{C}_{\epsilon_{j-1}}^{\beta_j} \underline{r}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} , & \text{if } i = j \\ \underline{0} , & \text{if } i > j \end{cases} , \quad \textit{and}$$

$$\frac{\partial \underline{\omega}_{W\beta_j}^{\beta_j}}{\partial \ddot{\underline{q}}_{[i]}} = \underline{0}_3 .$$

Proof. Changing the resolving frame of (2.5) to β_j yields

$$\underline{\omega}_{W\beta_j}^{\beta_j} = \underbrace{\mathbf{C}_{\epsilon_{j-1}}^{\beta_j}}_{\underline{q}_{[j]}} \mathbf{C}_{\beta_{j-1}}^{\epsilon_{j-1}} \Big(\underbrace{\underline{\omega}_{W\beta_{j-1}}^{\beta_{j-1}}}_{\underline{q}_{[:j-1]},\, \dot{\underline{q}}_{[:j-1]}} + \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underline{r}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \underbrace{\dot{\underline{q}}_{[j]}}_{\dot{\underline{q}}_{[j]}} \Big) . \tag{A.1}$$

The underbraces contain the dependencies on the kinematic state for the individual terms. Thus, we see that

$$\frac{\partial \underline{\omega}_{W\beta_j}^{\beta_j}}{\partial \underline{q}_{[i]}} = \mathbf{C}_{\beta_{j-1}}^{\beta_j} \frac{\partial \underline{\omega}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \underline{q}_{[i]}}$$

for $i < j$ and

$$\frac{\partial \underline{\omega}_{W\beta_j}^{\beta_j}}{\partial \underline{q}_{[j]}} = -\mathbf{R}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \underline{\omega}_{W\beta_j}^{\beta_j}$$

for $i = j$ using Lemma A.1. The remaining derivatives with respect to the joint position are zero. Analogously, the derivatives with respect to the joint velocities are

$$\frac{\partial \underline{\omega}_{W\beta_j}^{\beta_j}}{\partial \dot{\underline{q}}_{[i]}} = \mathbf{C}_{\beta_{j-1}}^{\beta_j} \frac{\partial \underline{\omega}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \dot{\underline{q}}_{[i]}}$$

for $i < j$,

$$\frac{\partial \underline{\omega}_{W\beta_j}^{\beta_j}}{\partial \dot{\underline{q}}_{[j]}} = \mathbf{C}_{\epsilon_{j-1}}^{\beta_j} \underline{r}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}}$$

for $i = j$, and zero for $i > j$. The derivatives with respect to the joint accelerations are all zero, because the joint accelerations do not appear in (A.1). □

Theorem A.3:

The partial derivatives of the angular acceleration in the body frame $\underline{\alpha}_{W\beta_j}^{\beta_j}$ with respect to the joint configuration $\underline{q}$ and its derivatives $\dot{\underline{q}}$ and $\ddot{\underline{q}}$ can be calculated using the recursive expressions

$$\frac{\partial \underline{\alpha}_{W\beta_j}^{\beta_j}}{\partial \underline{q}_{[i]}} = \begin{cases} \mathbf{C}_{\beta_{j-1}}^{\beta_j} \left(\frac{\partial \underline{\alpha}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \underline{q}_{[i]}} + \left[\frac{\partial \underline{\omega}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \underline{q}_{[i]}} \times \right] \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underline{r}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \dot{\underline{q}}_{[j]} \right), & \text{if } i < j \\ -\mathbf{R}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \underline{\alpha}_{W\beta_j}^{\beta_j}, & \text{if } i = j \\ \underline{0}, & \text{if } i > j \end{cases},$$

$$\frac{\partial \underline{\alpha}_{W\beta_j}^{\beta_j}}{\partial \dot{\underline{q}}_{[i]}} = \begin{cases} \mathbf{C}_{\beta_{j-1}}^{\beta_j} \left(\frac{\partial \underline{\alpha}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \dot{\underline{q}}_{[i]}} + \left[\frac{\partial \underline{\omega}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \dot{\underline{q}}_{[i]}} \times \right] \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underline{r}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \dot{\underline{q}}_{[j]} \right), & \text{if } i < j \\ \mathbf{C}_{\beta_{j-1}}^{\beta_j} \boldsymbol{\Omega}_{W\beta_{j-1}}^{\beta_{j-1}} \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underline{r}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}}, & \text{if } i = j \\ \underline{0}, & \text{if } i > j \end{cases}, \quad \textit{and}$$

$$\frac{\partial \underline{\alpha}_{W\beta_j}^{\beta_j}}{\partial \ddot{\underline{q}}_{[i]}} = \begin{cases} \mathbf{C}_{\beta_{j-1}}^{\beta_j} \frac{\partial \underline{\alpha}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \ddot{\underline{q}}_{[i]}}, & \text{if } i < j \\ \mathbf{C}_{\epsilon_{j-1}}^{\beta_j} \underline{r}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}}, & \text{if } i = j \\ \underline{0}, & \text{if } i > j \end{cases}.$$

Proof. Changing the resolving frame of (2.6) to β_j yields

$$\underline{\alpha}_{W\beta_j}^{\beta_j} = \underbrace{\mathbf{C}_{\epsilon_{j-1}}^{\beta_j}}_{\underline{q}_{[j]}} \mathbf{C}_{\beta_{j-1}}^{\epsilon_{j-1}} \Big(\underbrace{\underline{\alpha}_{W\beta_{j-1}}^{\beta_{j-1}}}_{\underline{q}_{[:j-1]}, \dot{\underline{q}}_{[:j-1]}, \ddot{\underline{q}}_{[:j-1]}} + \underbrace{\boldsymbol{\Omega}_{W\beta_{j-1}}^{\beta_{j-1}}}_{\underline{q}_{[:j-1]}, \dot{\underline{q}}_{[:j-1]}} \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underline{r}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \underbrace{\dot{\underline{q}}_{[j]}}_{\dot{\underline{q}}_{[j]}} + \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underline{r}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \underbrace{\ddot{\underline{q}}_{[j]}}_{\ddot{\underline{q}}_{[j]}} \Big).$$

Again, the underbraces show how the terms depend on the kinematic state. The desired partial derivatives follow immediately with the same scheme as in Theorem A.2, except that the angular acceleration also depends on the joint accelerations. □

Theorem A.4:

The partial derivatives of the linear acceleration in the body frame $\underline{a}_{W\beta_j}^{\beta_j}$ with respect to the joint

configuration $\underline{q}$ and its derivatives $\dot{\underline{q}}$ and $\ddot{\underline{q}}$ can be calculated using the recursive expressions

$$\frac{\partial \underline{a}_{W\beta_j}^{\beta_j}}{\partial \underline{q}_{[i]}} = \begin{cases} \mathbf{C}_{\beta_{j-1}}^{\beta_j} \Bigg(+ \dfrac{\partial \underline{a}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \underline{q}_{[i]}} + 2 \left[\dfrac{\partial \underline{\omega}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \underline{q}_{[i]}} \times \right] \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underline{t}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \dot{\underline{q}}_{[j]} \\ \qquad + \left(\left[\dfrac{\partial \underline{\alpha}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \underline{q}_{[i]}} \times \right] + \left[\dfrac{\partial \underline{\omega}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \underline{q}_{[i]}} \times \right] \boldsymbol{\Omega}_{W\beta_{j-1}}^{\beta_{j-1}} + \boldsymbol{\Omega}_{W\beta_{j-1}}^{\beta_{j-1}} \left[\dfrac{\partial \underline{\omega}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \underline{q}_{[i]}} \times \right] \right) \underline{x}_{\beta_{j-1}\beta_j}^{\beta_{j-1}} \Bigg), & \text{if } i < j \\ -\mathbf{R}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \underline{a}_{W\beta_j}^{\beta_j} + \mathbf{C}_{\beta_{j-1}}^{\beta_j} \left(\left[\underline{\alpha}_{W\beta_{j-1}}^{\beta_{j-1}} \times \right] + \boldsymbol{\Omega}_{W\beta_{j-1}}^{\beta_{j-1}} \boldsymbol{\Omega}_{W\beta_{j-1}}^{\beta_{j-1}} \right) \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underline{t}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}}, & \text{if } i = j \\ \underline{0}, & \text{if } i > j \end{cases},$$

$$\frac{\partial \underline{a}_{W\beta_j}^{\beta_j}}{\partial \dot{\underline{q}}_{[i]}} = \begin{cases} \mathbf{C}_{\beta_{j-1}}^{\beta_j} \Bigg(+ \dfrac{\partial \underline{a}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \dot{\underline{q}}_{[i]}} + 2 \left[\dfrac{\partial \underline{\omega}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \dot{\underline{q}}_{[i]}} \times \right] \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underline{t}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \dot{\underline{q}}_{[j]} \\ \qquad + \left(\left[\dfrac{\partial \underline{\alpha}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \dot{\underline{q}}_{[i]}} \times \right] + \left[\dfrac{\partial \underline{\omega}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \dot{\underline{q}}_{[i]}} \times \right] \boldsymbol{\Omega}_{W\beta_{j-1}}^{\beta_{j-1}} + \boldsymbol{\Omega}_{W\beta_{j-1}}^{\beta_{j-1}} \left[\dfrac{\partial \underline{\omega}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \dot{\underline{q}}_{[i]}} \times \right] \right) \underline{x}_{\beta_{j-1}\beta_j}^{\beta_{j-1}} \Bigg), & \text{if } i < j \\ 2\, \mathbf{C}_{\beta_{j-1}}^{\beta_j} \boldsymbol{\Omega}_{W\beta_{j-1}}^{\beta_{j-1}} \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underline{t}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}}, & \text{if } i = j \\ \underline{0}, & \text{if } i > j \end{cases},$$

$$\frac{\partial \underline{a}_{W\beta_j}^{\beta_j}}{\partial \ddot{\underline{q}}_{[i]}} = \begin{cases} \mathbf{C}_{\beta_{j-1}}^{\beta_j} \left(\dfrac{\partial \underline{a}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \ddot{\underline{q}}_{[i]}} + \left[\dfrac{\partial \underline{\alpha}_{W\beta_{j-1}}^{\beta_{j-1}}}{\partial \ddot{\underline{q}}_{[i]}} \times \right] \underline{x}_{\beta_{j-1}\beta_j}^{\beta_{j-1}} \right), & \text{if } i < j \\ \mathbf{C}_{\epsilon_{j-1}}^{\beta_j} \underline{t}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}}, & \text{if } i = j \\ \underline{0}, & \text{if } i > j \end{cases}$$

with

$$\underline{x}_{\beta_{j-1}\beta_j}^{\beta_{j-1}} = \underline{x}_{\beta_{j-1}\epsilon_{j-1}}^{\beta_{j-1}} + \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underline{t}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \underline{q}_{[j]} \,.$$

Proof. Changing the resolving frame of (2.8) to β_j yields

$$\underline{a}_{W\beta_j}^{\beta_j} = \underbrace{\mathbf{C}_{\epsilon_{j-1}}^{\beta_j}}_{\underline{q}_{[j]}} \mathbf{C}_{\beta_{j-1}}^{\epsilon_{j-1}} \Bigg(+ \underbrace{\underline{a}_{W\beta_{j-1}}^{\beta_{j-1}}}_{\underline{q}_{[:j-1]},\, \dot{\underline{q}}_{[:j-1]},\, \ddot{\underline{q}}_{[:j-1]}} + 2 \underbrace{\boldsymbol{\Omega}_{W\beta_{j-1}}^{\beta_{j-1}}}_{\underline{q}_{[:j-1]},\, \dot{\underline{q}}_{[:j-1]}} \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underline{t}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \underbrace{\dot{\underline{q}}_{[j]}}_{\dot{\underline{q}}_{[j]}} + \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underline{t}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \underbrace{\ddot{\underline{q}}_{[j]}}_{\ddot{\underline{q}}_{[j]}} + \Big(\Big[\underbrace{\underline{\alpha}_{W\beta_{j-1}}^{\beta_{j-1}}}_{\underline{q}_{[:j-1]},\, \dot{\underline{q}}_{[:j-1]},\, \ddot{\underline{q}}_{[:j-1]}} \times \Big] + \underbrace{\boldsymbol{\Omega}_{W\beta_{j-1}}^{\beta_{j-1}} \boldsymbol{\Omega}_{W\beta_{j-1}}^{\beta_{j-1}}}_{\underline{q}_{[:j-1]},\, \dot{\underline{q}}_{[:j-1]}} \Big) \Big(\underline{x}_{\beta_{j-1}\epsilon_{j-1}}^{\beta_{j-1}} + \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underline{t}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \underbrace{\underline{q}_{[j]}}_{\underline{q}_{[j]}} \Big) \Big)$$

using

$$\mathbf{C}_W^{\beta_{j-1}} \boldsymbol{\Omega}_{W\beta_{j-1}}^W \boldsymbol{\Omega}_{W\beta_{j-1}}^W \mathbf{C}_{\beta_{j-1}}^W = \mathbf{C}_W^{\beta_{j-1}} \boldsymbol{\Omega}_{W\beta_{j-1}}^W \Big(\mathbf{C}_{\beta_{j-1}}^W \mathbf{C}_W^{\beta_{j-1}} \Big) \boldsymbol{\Omega}_{W\beta_{j-1}}^W \mathbf{C}_{\beta_{j-1}}^W = \boldsymbol{\Omega}_{W\beta_{j-1}}^{\beta_{j-1}} \boldsymbol{\Omega}_{W\beta_{j-1}}^{\beta_{j-1}} \,,$$

which stems from (2.3). The remainder of the proof is analogous to Theorem A.3. □

Theorem A.5:

The first-order partial derivative of the link start frame orientation $\mathbf{C}_{\beta_j}^W$ with respect to the joint configuration $\underline{q}$ can be calculated using the recursive expression

$$\frac{\partial \mathbf{C}_{\beta_j}^W}{\partial \underline{q}_{[i]}} = \begin{cases} \dfrac{\partial \mathbf{C}_{\beta_{j-1}}^W}{\partial \underline{q}_{[i]}} \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \mathbf{C}_{\beta_j}^{\epsilon_{j-1}}, & \text{if } i < j \\ \mathbf{C}_{\beta_{j-1}}^W \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \mathbf{R}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \mathbf{C}_{\beta_j}^{\epsilon_{j-1}}, & \text{if } i = j \\ \underline{0}, & \text{if } i > j \end{cases} .$$

Proof. According to (2.4) the orientation of the body frame is

$$\mathbf{C}_{\beta_j}^W = \underbrace{\mathbf{C}_{\beta_{j-1}}^W}_{\underline{q}_{[:j-1]}} \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underbrace{\mathbf{C}_{\beta_j}^{\epsilon_{j-1}}}_{\underline{q}_{[j]}}$$

As before, the underbraces contain the dependencies on the joint configuration $\underline{q}$ for the individual terms. The application of the product rule yields the derivative

$$\frac{\partial \mathbf{C}_{\beta_j}^W}{\partial \underline{q}_{[i]}} = \underbrace{\frac{\partial \mathbf{C}_{\beta_{j-1}}^W}{\partial \underline{q}_{[i]}}}_{\underline{q}_{[:j-1]}} \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underbrace{\mathbf{C}_{\beta_j}^{\epsilon_{j-1}}}_{\underline{q}_{[j]}} + \underbrace{\mathbf{C}_{\beta_{j-1}}^W}_{\underline{q}_{[:j-1]}} \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \underbrace{\frac{\partial \mathbf{C}_{\beta_j}^{\epsilon_{j-1}}}{\partial \underline{q}_{[i]}}}_{\underline{q}_{[j]}} . \tag{A.2}$$

Thus, if $i < j$

$$\frac{\partial \mathbf{C}_{\beta_j}^W}{\partial \underline{q}_{[i]}} = \frac{\partial \mathbf{C}_{\beta_{j-1}}^W}{\partial \underline{q}_{[i]}} \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \mathbf{C}_{\beta_j}^{\epsilon_{j-1}}$$

remains. For $i = j$, the derivative is

$$\frac{\partial \mathbf{C}_{\beta_j}^W}{\partial \underline{q}_{[i]}} = \mathbf{C}_{\beta_{j-1}}^W \mathbf{C}_{\epsilon_{j-1}}^{\beta_{j-1}} \mathbf{R}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} \mathbf{C}_{\beta_j}^{\epsilon_{j-1}}$$

after the application of Theorem A.1. For $i > j$, the derivative is zero. This result is also correct for prismatic joints because $\underline{r}_{\epsilon_{j-1}\beta_j}^{\epsilon_{j-1}} = \underline{0}$ in this case. □

Theorem A.6:

The second-order partial derivative of the link start frame orientation $\mathbf{C}_{\beta_k}^W$ with respect to the joint configuration $\underline{q}$ can be calculated using the recursive expression

$$\frac{\partial^2 \mathbf{C}_{\beta_k}^W}{\partial \underline{q}_{[i]} \partial \underline{q}_{[j]}} = \begin{cases} \underline{0}\,, & \text{if } k < i \vee k < j \\ \mathbf{C}_{\beta_{k-1}}^W \mathbf{C}_{\epsilon_{k-1}}^{\beta_{k-1}} \mathbf{R}_{\epsilon_{k-1}\beta_k}^{\epsilon_{k-1}} \mathbf{R}_{\epsilon_{k-1}\beta_k}^{\epsilon_{k-1}} \mathbf{C}_{\beta_k}^{\epsilon_{k-1}}\,, & \text{if } k = i = j \\ \frac{\partial \mathbf{C}_{\beta_{k-1}}^W}{\partial \underline{q}_{[j]}} \mathbf{C}_{\epsilon_{k-1}}^{\beta_{k-1}} \mathbf{R}_{\epsilon_{k-1}\beta_k}^{\epsilon_{k-1}} \mathbf{C}_{\beta_k}^{\epsilon_{k-1}}\,, & \text{if } k = i \wedge k > j \\ \frac{\partial \mathbf{C}_{\beta_{k-1}}^W}{\partial \underline{q}_{[i]}} \mathbf{C}_{\epsilon_{k-1}}^{\beta_{k-1}} \mathbf{R}_{\epsilon_{k-1}\beta_k}^{\epsilon_{k-1}} \mathbf{C}_{\beta_k}^{\epsilon_{k-1}}\,, & \text{if } k = j \wedge k > i \\ \frac{\partial^2 \mathbf{C}_{\beta_{k-1}}^W}{\partial \underline{q}_{[i]} \partial \underline{q}_{[j]}} \mathbf{C}_{\epsilon_{k-1}}^{\beta_{k-1}} \mathbf{C}_{\beta_k}^{\epsilon_{k-1}}\,, & \text{if } k > i \wedge k > j \end{cases} .$$

The required first-order partial derivatives of $\mathbf{C}_{\beta_k}^W$ can be calculated using Theorem A.5.

Proof. With (A.2), we know that

$$\frac{\partial \mathbf{C}_{\beta_k}^W}{\partial \underline{q}_{[i]}} = \underbrace{\frac{\partial \mathbf{C}_{\beta_{k-1}}^W}{\partial \underline{q}_{[i]}}}_{\underline{q}_{[:k-1]}} \mathbf{C}_{\epsilon_{k-1}}^{\beta_{k-1}} \underbrace{\mathbf{C}_{\beta_k}^{\epsilon_{k-1}}}_{\underline{q}_{[k]}} + \underbrace{\mathbf{C}_{\beta_{k-1}}^W}_{\underline{q}_{[:k-1]}} \mathbf{C}_{\epsilon_{k-1}}^{\beta_{k-1}} \underbrace{\frac{\partial \mathbf{C}_{\beta_k}^{\epsilon_{k-1}}}{\partial \underline{q}_{[i]}}}_{\underline{q}_{[k]}} .$$

Deriving this with respect to $\underline{q}_{[j]}$ yields

$$\begin{aligned} \frac{\partial^2 \mathbf{C}_{\beta_k}^W}{\partial \underline{q}_{[i]} \partial \underline{q}_{[j]}} = &+ \underbrace{\frac{\partial^2 \mathbf{C}_{\beta_{k-1}}^W}{\partial \underline{q}_{[i]} \partial \underline{q}_{[j]}}}_{\underline{q}_{[:k-1]}} \mathbf{C}_{\epsilon_{k-1}}^{\beta_{k-1}} \underbrace{\mathbf{C}_{\beta_k}^{\epsilon_{k-1}}}_{\underline{q}_{[k]}} + \underbrace{\frac{\partial \mathbf{C}_{\beta_{k-1}}^W}{\partial \underline{q}_{[i]}}}_{\underline{q}_{[:k-1]}} \mathbf{C}_{\epsilon_{k-1}}^{\beta_{k-1}} \underbrace{\frac{\partial \mathbf{C}_{\beta_k}^{\epsilon_{k-1}}}{\partial \underline{q}_{[j]}}}_{\underline{q}_{[k]}} \\ &+ \underbrace{\frac{\partial \mathbf{C}_{\beta_{k-1}}^W}{\partial \underline{q}_{[j]}}}_{\underline{q}_{[:k-1]}} \mathbf{C}_{\epsilon_{k-1}}^{\beta_{k-1}} \underbrace{\frac{\partial \mathbf{C}_{\beta_k}^{\epsilon_{k-1}}}{\partial \underline{q}_{[i]}}}_{\underline{q}_{[k]}} + \underbrace{\mathbf{C}_{\beta_{k-1}}^W}_{\underline{q}_{[:k-1]}} \mathbf{C}_{\epsilon_{k-1}}^{\beta_{k-1}} \underbrace{\frac{\partial^2 \mathbf{C}_{\beta_k}^{\epsilon_{k-1}}}{\partial \underline{q}_{[i]} \partial \underline{q}_{[j]}}}_{\underline{q}_{[k]}} . \end{aligned}$$

With the dependencies on the joint configuration in underbraces, it is easy to see that

$$\frac{\partial^2 \mathbf{C}_{\beta_k}^W}{\partial \underbrace{q}_{[i]} \partial \underbrace{q}_{[j]}} = \underline{0}$$

if $k < i$ or $k < j$. This also matches the observation that a joint can only affect the part of the kinematic chain that comes after the joint.

If $i = k$, the first and second term are zero. Now, two subcases have to be considered separately: In the first case, $j = k$ holds. This causes the third term to be zero as well, yielding

$$\frac{\partial^2 \mathbf{C}_{\beta_k}^W}{\partial \underbrace{q}_{[i]} \partial \underbrace{q}_{[j]}} = \mathbf{C}_{\beta_{k-1}}^W \mathbf{C}_{\epsilon_{k-1}}^{\beta_{k-1}} \mathbf{R}_{\epsilon_{k-1}\beta_k}^{\epsilon_{k-1}} \mathbf{R}_{\epsilon_{k-1}\beta_k}^{\epsilon_{k-1}} \mathbf{C}_{\beta_k}^{\epsilon_{k-1}}$$

after applying Theorem A.1 twice. In the second case, $k > j$ holds, causing the fourth term to vanish. The third term remains and is simplified to

$$\frac{\partial^2 \mathbf{C}_{\beta_k}^W}{\partial \underbrace{q}_{[i]} \partial \underbrace{q}_{[j]}} = \frac{\partial \mathbf{C}_{\beta_{k-1}}^W}{\partial \underbrace{q}_{[j]}} \mathbf{C}_{\epsilon_{k-1}}^{\beta_{k-1}} \mathbf{R}_{\epsilon_{k-1}\beta_k}^{\epsilon_{k-1}} \mathbf{C}_{\beta_k}^{\epsilon_{k-1}}$$

using Theorem A.1.

Next, $j = k$ is considered in two subcases, where the first and the third term are zero. The subcase $i = k$ is already covered. The subcase $k > i$ removes the fourth term and yields

$$\frac{\partial^2 \mathbf{C}_{\beta_k}^W}{\partial \underbrace{q}_{[i]} \partial \underbrace{q}_{[j]}} = \frac{\partial \mathbf{C}_{\beta_{k-1}}^W}{\partial \underbrace{q}_{[i]}} \mathbf{C}_{\epsilon_{k-1}}^{\beta_{k-1}} \mathbf{R}_{\epsilon_{k-1}\beta_k}^{\epsilon_{k-1}} \mathbf{C}_{\beta_k}^{\epsilon_{k-1}} .$$

What remains is the case $k > i$ and $k > j$, where all terms except for the first one are zero. Thus,

$$\frac{\partial^2 \mathbf{C}_{\beta_k}^W}{\partial \underbrace{q}_{[i]} \partial \underbrace{q}_{[j]}} = \frac{\partial^2 \mathbf{C}_{\beta_{k-1}}^W}{\partial \underbrace{q}_{[i]} \partial \underbrace{q}_{[j]}} \mathbf{C}_{\epsilon_{k-1}}^{\beta_{k-1}} \mathbf{C}_{\beta_k}^{\epsilon_{k-1}}$$

is true for this case. □

Theorem A.7:

The second-order partial derivative of the link start frame position $\underbrace{x}_{W\beta_k}^W$ with respect to the joint configuration $\underbrace{q}$ can be calculated using the recursive expression

$$\frac{\partial^2 \underbrace{x}_{W\beta_k}^W}{\partial \underbrace{q}_{[i]} \partial \underbrace{q}_{[j]}} = \begin{cases} \underline{0}\,, & \text{if } k < i \vee k < j \\ \underline{0}\,, & \text{if } k = i = j \\ \frac{\partial \mathbf{C}_{\beta_{k-1}}^W}{\partial \underbrace{q}_{[j]}} \mathbf{C}_{\epsilon_{k-1}}^{\beta_{k-1}} \underbrace{t}_{\epsilon_{k-1}\beta_k}^{\epsilon_{k-1}}\,, & \text{if } k = i \wedge k > j \\ \frac{\partial \mathbf{C}_{\beta_{k-1}}^W}{\partial \underbrace{q}_{[i]}} \mathbf{C}_{\epsilon_{k-1}}^{\beta_{k-1}} \underbrace{t}_{\epsilon_{k-1}\beta_k}^{\epsilon_{k-1}}\,, & \text{if } k = j \wedge k > i \\ \frac{\partial^2 \underbrace{x}_{W\beta_{k-1}}^W}{\partial \underbrace{q}_{[i]} \partial \underbrace{q}_{[j]}} + \frac{\partial^2 \mathbf{C}_{\beta_{k-1}}^W}{\partial \underbrace{q}_{[i]} \partial \underbrace{q}_{[j]}} \left(\underbrace{x}_{\beta_{k-1}\epsilon_{k-1}}^{\beta_{k-1}} + \mathbf{C}_{\epsilon_{k-1}}^{\beta_{k-1}} \underbrace{t}_{\epsilon_{k-1}\beta_k}^{\epsilon_{k-1}} \underbrace{q}_{[k]} \right)\,, & \text{if } k > i \wedge k > j \end{cases} .$$

The required first- and second-order partial derivatives of $\mathbf{C}_{\beta_k}^W$ can be calculated using Theorems A.5 and A.6.

Proof. According to (2.7), the position of frame β_k is

$$\underline{x}^W_{W\beta_k} = \underbrace{\underline{x}^W_{W\beta_{k-1}}}_{\underline{q}_{[:k-1]}} + \underbrace{\mathbf{C}^W_{\beta_{k-1}}}_{\underline{q}_{[:k-1]}} \Big(\underline{x}^{\beta_{k-1}}_{\beta_{k-1}\epsilon_{k-1}} + \mathbf{C}^{\beta_{k-1}}_{\epsilon_{k-1}} \underline{t}^{\epsilon_{k-1}}_{\epsilon_{k-1}\beta_k} \underbrace{\underline{q}_{[k]}}_{\underline{q}_{[k]}} \Big) .$$

The derivative with respect to $\underline{q}_{[i]}$ is then

$$\frac{\partial \underline{x}^W_{W\beta_k}}{\partial \underline{q}_{[i]}} = \underbrace{\frac{\partial \underline{x}^W_{W\beta_{k-1}}}{\partial \underline{q}_{[i]}}}_{\underline{q}_{[:k-1]}} + \underbrace{\frac{\partial \mathbf{C}^W_{\beta_{k-1}}}{\partial \underline{q}_{[i]}}}_{\underline{q}_{[:k-1]}} \Big(\underline{x}^{\beta_{k-1}}_{\beta_{k-1}\epsilon_{k-1}} + \mathbf{C}^{\beta_{k-1}}_{\epsilon_{k-1}} \underline{t}^{\epsilon_{k-1}}_{\epsilon_{k-1}\beta_k} \underbrace{\underline{q}_{[k]}}_{\underline{q}_{[k]}} \Big) + \underbrace{\mathbf{C}^W_{\beta_{k-1}}}_{\underline{q}_{[:k-1]}} \mathbf{C}^{\beta_{k-1}}_{\epsilon_{k-1}} \underline{t}^{\epsilon_{k-1}}_{\epsilon_{k-1}\beta_k} \underbrace{\frac{\partial \underline{q}_{[k]}}{\partial \underline{q}_{[i]}}}_{\underline{q}_{[k]}} .$$

Deriving this with respect to $\underline{q}_{[j]}$ yields

$$\begin{aligned}
\frac{\partial^2 \underline{x}^W_{W\beta_k}}{\partial \underline{q}_{[i]} \partial \underline{q}_{[j]}} = &+ \underbrace{\frac{\partial^2 \underline{x}^W_{W\beta_{k-1}}}{\partial \underline{q}_{[i]} \partial \underline{q}_{[j]}}}_{\underline{q}_{[:k-1]}} + \underbrace{\frac{\partial^2 \mathbf{C}^W_{\beta_{k-1}}}{\partial \underline{q}_{[i]} \partial \underline{q}_{[j]}}}_{\underline{q}_{[:k-1]}} \Big(\underline{x}^{\beta_{k-1}}_{\beta_{k-1}\epsilon_{k-1}} + \mathbf{C}^{\beta_{k-1}}_{\epsilon_{k-1}} \underline{t}^{\epsilon_{k-1}}_{\epsilon_{k-1}\beta_k} \underbrace{\underline{q}_{[k]}}_{\underline{q}_{[k]}} \Big) \\
&+ \underbrace{\frac{\partial \mathbf{C}^W_{\beta_{k-1}}}{\partial \underline{q}_{[i]}}}_{\underline{q}_{[:k-1]}} \mathbf{C}^{\beta_{k-1}}_{\epsilon_{k-1}} \underline{t}^{\epsilon_{k-1}}_{\epsilon_{k-1}\beta_k} \underbrace{\frac{\partial \underline{q}_{[k]}}{\partial \underline{q}_{[j]}}}_{\underline{q}_{[k]}} + \underbrace{\frac{\partial \mathbf{C}^W_{\beta_{k-1}}}{\partial \underline{q}_{[j]}}}_{\underline{q}_{[:k-1]}} \mathbf{C}^{\beta_{k-1}}_{\epsilon_{k-1}} \underline{t}^{\epsilon_{k-1}}_{\epsilon_{k-1}\beta_k} \underbrace{\frac{\partial \underline{q}_{[k]}}{\partial \underline{q}_{[i]}}}_{\underline{q}_{[k]}} \\
&+ \underbrace{\mathbf{C}^W_{\beta_{k-1}}}_{\underline{q}_{[:k-1]}} \mathbf{C}^{\beta_{k-1}}_{\epsilon_{k-1}} \underline{t}^{\epsilon_{k-1}}_{\epsilon_{k-1}\beta_k} \underbrace{\frac{\partial^2 \underline{q}_{[k]}}{\partial \underline{q}_{[i]} \partial \underline{q}_{[j]}}}_{\underline{q}_{[k]}} .
\end{aligned}$$

Analog to the proof of Theorem A.6, all terms become zero when $k < i$ or $k < j$. If $k = i = j$, the result is zero as well, because $\frac{\partial^2 x}{\partial x \partial x} = 0$. If $k = i$ in combination with $k > j$, only the fourth term remains. In this case,

$$\frac{\partial^2 \underline{x}^W_{W\beta_k}}{\partial \underline{q}_{[i]} \partial \underline{q}_{[j]}} = \frac{\partial \mathbf{C}^W_{\beta_{k-1}}}{\partial \underline{q}_{[j]}} \mathbf{C}^{\beta_{k-1}}_{\epsilon_{k-1}} \underline{t}^{\epsilon_{k-1}}_{\epsilon_{k-1}\beta_k}$$

is the desired derivative. Likewise, the condition $k = j$ and $k > i$ eliminates all terms except for the third and yields

$$\frac{\partial^2 \underline{x}^W_{W\beta_k}}{\partial \underline{q}_{[i]} \partial \underline{q}_{[i]}} = \frac{\partial \mathbf{C}^W_{\beta_{k-1}}}{\partial \underline{q}_{[j]}} \mathbf{C}^{\beta_{k-1}}_{\epsilon_{k-1}} \underline{t}^{\epsilon_{k-1}}_{\epsilon_{k-1}\beta_k} .$$

Finally, if $k > i$ and $k > j$, the first and the second term are non-zero. The result is then

$$\frac{\partial^2 \underline{x}^W_{W\beta_k}}{\partial \underline{q}_{[i]} \partial \underline{q}_{[i]}} = \frac{\partial^2 \underline{x}^W_{W\beta_{k-1}}}{\partial \underline{q}_{[i]} \partial \underline{q}_{[j]}} + \frac{\partial^2 \mathbf{C}^W_{\beta_{k-1}}}{\partial \underline{q}_{[i]} \partial \underline{q}_{[j]}} \left(\underline{x}^{\beta_{k-1}}_{\beta_{k-1}\epsilon_{k-1}} + \mathbf{C}^{\beta_{k-1}}_{\epsilon_{k-1}} \underline{t}^{\epsilon_{k-1}}_{\epsilon_{k-1}\beta_k} \underline{q}_{[k]} \right) .$$

□

APPENDIX
B

Example Code for RTCF

```
<launch>
    <node name="rt_runner" pkg="rtcf" type="rt_runner" output="screen">
        <rosparam param="mode">"wait_for_components"</rosparam>
        <rosparam param="ros_mapping_whitelist">".*ros.*"</rosparam>
        <rosparam param="num_components_expected">2</rosparam>
        <rosparam param="frequency">1000.0</rosparam>
    </node>
    <node name="example1" pkg="rtcf" type="rt_launcher" args="rtcf_examples
    ↪ PaperExample">
        <remap from="out_port" to="/tmp"/>
        <remap from="in_port" to="/ros/in"/>
        <rosparam param="is_first">True</rosparam>
    </node>
    <node name="example2" pkg="rtcf" type="rt_launcher" args="rtcf_examples
    ↪ PaperExample">
        <remap from="out_port" to="/ros/out"/>
        <remap from="in_port" to="/tmp"/>
    </node>
</launch>
```

Listing B.1: Example of a ROS-compatible launch file with two real-time components that are loaded into an rt_runner and executed at 1 kHz. Taken from [O6].

```
#include <std_msgs/Float32.h>

#include <rtcf/macros.hpp>
#include <rtcf/rtcf_extension.hpp>
OROCOS_HEADERS_BEGIN
#include <rtt/Component.hpp>
#include <rtt/Port.hpp>
#include <rtt/RTT.hpp>
OROCOS_HEADERS_END

class PaperExample : public RTT::TaskContext, public RtcfExtension
{
public:
  PaperExample(std::string const &name) :
      TaskContext(name), port_out_("out_port"), port_in_("in_port") {}

  bool configureHook()
  {
    this->ports()->addPort(port_in_);
    this->ports()->addPort(port_out_);
    double param = this->getNodeHandle().param("/test_param", 0.0);
    NON_RT_INFO_STREAM("Fetched param with value " << param);
    return true;
  }
  void updateHook()
  {
    RT_INFO("Update hook called!");
    port_in_.read(msg_);
    port_out_.write(msg_);
  }

  bool startHook() { return true; }
  void stopHook() {}
  void cleanupHook() {}

private:
  RTT::OutputPort<std_msgs::Float32> port_out_;
  RTT::InputPort<std_msgs::Float32> port_in_;
  std_msgs::Float32 msg_;
};

ORO_CREATE_COMPONENT(PaperExample)
```

Listing B.2: A minimal working example of an RTCF component. Taken from [O6].

List of Figures

List of Tables

List of Algorithms

List of Listings

Bibliography

[1] J. V. Draper, D. B. Kaber, and J. M. Usher, "Telepresence," *Human Factors*, vol. 40, no. 3, pp. 354–375, Sep. 1998.

[2] Meta Platforms Inc., "Introducing Meta: A social technology company," https://about.fb.com/news/2021/10/facebook-company-is-now-meta/, Oct. 2021, accessed: 2024-10-19.

[3] Apple Inc., "Introducing Apple Vision Pro: Apple's first spatial computer," https://www.apple.com/newsroom/2023/06/introducing-apple-vision-pro/, Jun. 2023, accessed: 2024-10-19.

[4] M. A. Srinivasan and C. Basdogan, "Haptics in virtual environments: Taxonomy, research status, and challenges," *Computers & Graphics*, vol. 21, no. 4, pp. 393–404, Jul. 1997.

[5] T. A. Kern, C. Hatzfeld, and A. Abbasimoshaei, Eds., *Engineering Haptic Devices*, 3rd ed., ser. Springer Series on Touch and Haptic Systems. Cham: Springer Nature, 2023.

[6] B. Hannaford and A. M. Okamura, "Haptics," in *Springer Handbook of Robotics*, 2nd ed., B. Siciliano and O. Khatib, Eds. Berlin, Heidelberg: Springer, 2016, pp. 1063–1083.

[7] S. Choi and K. J. Kuchenbecker, "Vibrotactile display: Perception, technology, and applications," *Proceedings of the IEEE*, vol. 101, no. 9, pp. 2093–2104, Sep. 2013.

[8] Ultraleap Inc., "Turning ultrasound into virtual touch," https://www.ultraleap.com/haptics/, accessed: 2024-10-19.

[9] Weart S.r.l., "TouchDIVER Pro haptic gloves," https://weart.it/touchdiver-pro-haptic-gloves/, accessed: 2024-11-12.

[10] Kuka AG, "RoboGym: Revolutionary training – Not just for top athletes," https://www.kuka.com/en-de/company/press/news/2020/10/robogym, Nov. 2020, accessed: 2024-11-12.

[11] J. Petereit, J. Beyerer, T. Asfour, S. Gentes, B. Hein, U. D. Hanebeck, F. Kirchner, R. Dillmann, H. H. Götting, M. Weiser, M. Gustmann, and T. Egloffstein, "ROBDEKON: Robotic systems for decontamination in hazardous environments," in *Proceedings of the 2019 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR 2019)*, Sep. 2019, pp. 249–255.

[12] M. Caeiro-Rodríguez, I. Otero-González, F. A. Mikic-Fonte, and M. Llamas-Nistal, "A systematic review of commercial smart gloves: Current status and applications," *Sensors*, vol. 21, no. 8, p. 2667, Apr. 2021.

[13] Y. Yokokohji, R. L. Hollis, and T. Kanade, "What you can see is what you can feel – Development of a visual/haptic interface to virtual environment," in *Proceedings of the 1996 IEEE Virtual Reality Annual International Symposium*, Mar. 1996, pp. 46–53.

[14] V. R. Mercado, M. Marchal, and A. Lécuyer, ""Haptics On-Demand": A survey on encountered-type haptic displays," *IEEE Transactions on Haptics*, vol. 14, no. 3, pp. 449–464, Feb. 2021.

[15] A. Adilkhanov, M. Rubagotti, and Z. Kappassov, "Haptic devices: Wearability-based taxonomy and literature review," *IEEE Access*, vol. 10, pp. 91 923–91 947, Aug. 2022.

[16] H. Seifi, F. Fazlollahi, M. Oppermann, J. A. Sastrillo, J. Ip, A. Agrawal, G. Park, K. J. Kuchenbecker, and K. E. MacLean, "Haptipedia: Accelerating haptic device discovery to support interaction & engineering design," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, May 2019, pp. 1–12.

[17] R. Q. van der Linde and P. Lammertse, "HapticMaster – A generic force controlled robot for human interaction," *Industrial Robot*, vol. 30, no. 6, pp. 515–524, Jan. 2003.

[18] M. Ueberle, N. Mock, and M. Buss, "VISHARD10, a novel hyper-redundant haptic interface," in *Proceedings of the 12th International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (HAPTICS 2004)*, Mar. 2004, pp. 58–65.

[19] M.-W. Ueberle, "Design, control, and evaluation of a family of kinesthetic haptic interfaces," Ph.D. dissertation, Technical University Munich, 2006.

[20] B. Dellon and Y. Matsuoka, "Modeling and system identification of a life-size brake-actuated manipulator," *IEEE Transactions on Robotics*, vol. 25, no. 3, pp. 481–491, Jun. 2009.

[21] M. V. Weghe, B. Dellon, S. Kelly, R. Juchniewicz, and Y. Matsuoka, "Demonstration of a large dissipative haptic environment," in *Proceedings of the 2006 14th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (HAPTICS 2006)*, Mar. 2006, pp. 329–330.

[22] P. Rößler, T. Armstrong, O. Hessel, M. Mende, and U. D. Hanebeck, "A novel haptic interface for free locomotion in extended range telepresence scenarios," in *Proceedings of the 3rd International Conference on Informatics in Control, Automation and Robotics (ICINCO 2006)*, Aug. 2006, pp. 148–153.

[23] A. Pérez Arias and U. D. Hanebeck, "A novel haptic interface for extended range telepresence: Control and evaluation," in *Proceedings of the 6th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2009)*, Jul. 2009, pp. 222–227.

[24] M. Zinn, O. Khatib, B. Roth, and J. K. Salisbury, "Large workspace haptic devices – A new actuation approach," in *Proceedings of the 2008 Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (HAPTICS 2008)*, Mar. 2008, pp. 185–192.

[25] J. Perret and L. Dominjon, "The INCA 6D: A commercial stringed haptic system suitable for industrial applications," in *Proceedings of the 2009 Joint Virtual Reality Conference (JVRC 2009)*, Dec. 2009, pp. 72–75.

[26] T. Hulin, K. Hertkorn, P. Kremer, S. Schätzle, J. Artigas, M. Sagardia, F. Zacharias, and C. Preusche, "The DLR bimanual haptic device with optimized workspace," in *Proceedings of the 2011 IEEE International Conference on Robotics and Automation (ICRA 2011)*, May 2011, pp. 3441–3442.

[27] M. Abdullah, M. Kim, W. Hassan, Y. Kuroda, and S. Jeon, "HapticDrone: An encountered-type kinesthetic haptic interface with controllable force feedback: Example of stiffness and weight rendering," in *Proceedings of the 2018 IEEE Haptics Symposium (HAPTICS 2018)*, Mar. 2018, pp. 334–339.

[28] A. Horie, M. H. D. Y. Saraiji, Z. Kashino, and M. Inami, "EncounteredLimbs: A room-scale encountered-type haptic presentation using wearable robotic arms," in *Proceedings of the 2021 IEEE Conference on Virtual Reality and 3D User Interfaces (VR 2021)*, Mar. 2021, pp. 260–269.

[29] R. Suzuki, H. Hedayati, C. Zheng, J. L. Bohn, D. Szafir, E. Y.-L. Do, M. D. Gross, and D. Leithinger, "RoomShift: Room-scale dynamic haptics for VR with furniture-moving swarm robots," in *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, Apr. 2020, pp. 1–11.

[30] Y. Yixian, K. Takashima, A. Tang, T. Tanno, K. Fujita, and Y. Kitamura, "ZoomWalls: Dynamic walls that simulate haptic infrastructure for room-scale VR world," in *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology (UIST 2020)*, Oct. 2020, pp. 223–235.

[31] S. Mortezapoor, K. Vasylevska, E. Vonach, and H. Kaufmann, "CoboDeck: A large-scale haptic VR system using a collaborative mobile robot," in *Proceedings of the 2023 IEEE Conference Virtual Reality and 3D User Interfaces (VR 2023)*, Mar. 2023, pp. 297–307.

[32] E. Bouzbib, G. Bailly, S. Haliyo, and P. Frey, "CoVR: A large-scale force-feedback robotic interface for non-deterministic scenarios in VR," in *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology (UIST '20)*, Oct. 2020, pp. 209–222.

[33] A. Pérez Arias, "Haptic guidance for extended range telepresence," Ph.D. dissertation, Karlsruhe Institute of Technology, 2013.

[34] P. D. Groves, *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*, 2nd ed. Boston: Artech House, 2013.

[35] S. Kim and M. Kim, "Rotation representations and their conversions," *IEEE Access*, vol. 11, pp. 6682–6699, Jan. 2023.

[36] K. Waldron and J. Schmiedeler, "Kinematics," in *Springer Handbook of Robotics*, 2nd ed., B. Siciliano and O. Khatib, Eds. Berlin, Heidelberg: Springer, 2016, pp. 11–36.

[37] R. Featherstone and D. E. Orin, "Dynamics," in *Springer Handbook of Robotics*, 2nd ed., B. Siciliano and O. Khatib, Eds. Berlin, Heidelberg: Springer, 2016, pp. 37–66.

[38] T. Arens, F. Hettlich, C. Karpfinger, U. Kockelkorn, K. Lichtenegger, and H. Stachel, *Mathematik*, 5th ed. Berlin, Heidelberg: Springer Spektrum, 2022.

[39] A. Hourtash, "The kinematic Hessian and higher derivatives," in *Proccedings of the 2005 IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA 2005)*, Jun. 2005, pp. 169–174.

[40] R. Featherstone, *Rigid Body Dynamics Algorithms*, 1st ed. New York: Springer, 2007.

[41] D. C. Bellicoso, C. Gehring, J. Hwangbo, P. Fankhauser, and M. Hutter, "Perception-less terrain adaptation through whole body control and hierarchical optimization," in *Proceedings of the 16th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2016)*, Nov. 2016, pp. 558–564.

[42] A. Herzog, N. Rotella, S. Mason, F. Grimminger, S. Schaal, and L. Righetti, "Momentum control with hierarchical inverse dynamics on a torque-controlled humanoid," *Autonomous Robots*, vol. 40, no. 3, pp. 473–491, Mar. 2016.

[43] A. Bambade, S. El-Kazdadi, A. Taylor, and J. Carpentier, "PROX-QP: Yet another quadratic programming solver for robotics and beyond," in *Proceedings of Robotics: Science and Systems 2022 (RSS 2022)*, Jun. 2022.

[44] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: An operator splitting solver for quadratic programs," *Mathematical Programming Computation*, vol. 12, pp. 637–672, Dec. 2020.

[45] P. Rößler, F. Beutler, U. D. Hanebeck, and N. Nitzsche, "Motion compression applied to guidance of a mobile teleoperator," in *Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005)*, Aug. 2005, pp. 2887–2892.

[46] N. Nitzsche, U. D. Hanebeck, and G. Schmidt, "Motion compression for telepresent walking in large-scale remote environments," *Presence: Teleoperators and Virtual Environments 2004*, vol. 13, no. 1, pp. 44–60, Feb. 2004.

[47] M. Ueberle and M. Buss, "Design, control, and evaluation of a new 6 DOF haptic device," in *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2002)*, Sep. 2002, pp. 2949–2954.

[48] Y. Zimmermann, A. Forino, R. Riener, and M. Hutter, "ANYexo: A versatile and dynamic upper-limb rehabilitation robot," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3649–3656, Oct. 2019.

[49] Y. Zimmermann, M. Sommerhalder, P. Wolf, R. Riener, and M. Hutter, "ANYexo 2.0: A fully actuated upper-limb exoskeleton for manipulation and joint-oriented training in all stages of rehabilitation," *IEEE Transactions on Robotics*, vol. 39, no. 3, pp. 2131–2150, Jun. 2023.

[50] N. Nitzsche, U. D. Hanebeck, and G. Schmidt, "Mobile haptic interaction with extended real or virtual environments," in *Proceedings of the 10th IEEE International Workshop on Robot and Human Interactive Communication (ROMAN 2001)*, Sep. 2001, pp. 313–318.

[51] ——, "Design issues of mobile haptic interfaces," *Journal of Robotic Systems*, vol. 20, no. 9, pp. 549–556, Aug. 2003.

[52] S. Knopp, M. Lorenz, L. Pelliccia, and P. Klimant, "Using industrial robots as haptic devices for VR-training," in *Proceedings of the 2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR 2018)*, Mar. 2018, pp. 607–608.

[53] Y. Kim, M. C. Castillo Silva, S. Anastasi, and N. Deshpande, "Towards immersive bilateral teleoperation using encountered-type haptic interface," in *Proceedings of the 2023 IEEE International Conference on Systems, Man, and Cybernetics (SMC 2023)*, Oct. 2023, pp. 1354–1359.

[54] S. Dai, J. Smiley, T. Dwyer, B. Ens, and L. Besancon, "RoboHapalytics: A robot assisted haptic controller for immersive analytics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 29, no. 1, pp. 451–461, Jan. 2023.

[55] F. Zacharias, I. S. Howard, T. Hulin, and G. Hirzinger, "Workspace comparisons of setup configurations for human-robot interaction," in *Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2010)*, Oct. 2010, pp. 3117–3122.

[56] T. Yoshikawa, "Manipulability of robotic mechanisms," *The International Journal of Robotics Research*, vol. 4, no. 2, pp. 3–9, Jun. 1985.

[57] ——, "Dynamic manipulability of robot manipulators," in *Proceedings of the 1985 IEEE International Conference on Robotics and Automation (ICRA 1985)*, Mar. 1985, pp. 1033–1038.

[58] V. Hayward, J. Choksi, G. Lanvin, and C. Ramstein, "Design and multi-objective optimization of a linkage for a haptic interface," in *Advances in Robot Kinematics and Computational Geometry*, J. Lenarčič and B. Ravani, Eds. Dordrecht: Springer, 1994, pp. 359–368.

[59] O. Khatib and J. Burdick, "Optimization of dynamics in manipulator design: The operational space formulation." *International Journal of Robotics and Automation*, vol. 2, no. 2, pp. 90–98, 1987.

[60] L. J. Stocco, S. E. Salcudean, and F. Sassani, "Mechanism design for global isotropy with applications to haptic interfaces," in *Proceedings of the ASME 1997 International Mechanical Engineering Congress and Exposition (IMECE 1997)*, Nov. 1997, pp. 115–122.

[61] J. C. Perry, J. M. Powell, and J. Rosen, "Isotropy of an upper limb exoskeleton and the kinematics and dynamics of the human arm," *Applied Bionics and Biomechanics*, vol. 6, no. 2, pp. 175–191, Jun. 2009.

[62] M. Chadwick, H. Kolvenbach, F. Dubois, H. F. Lau, and M. Hutter, "Vitruvio: An open-source leg design optimization toolbox for walking robots," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6318–6325, Oct. 2020.

[63] *DIN 33402-2:2020-12: Ergonomics – Human body dimensions – Part 2: Values*, German Institute for Standardization Std., 2020.

[64] D. A. Winter, *Biomechanics and Motor Control of Human Movement*, 4th ed. Hoboken, NJ: Wiley, 2009.

[65] J. Rosen, J. C. Perry, N. Manning, S. Burns, and B. Hannaford, "The human arm kinematics and dynamics during daily activities – Toward a 7 DOF upper limb powered exoskeleton," in *Proceedings of the 12th International Conference on Advanced Robotics (ICAR 2005)*, Jul. 2005, pp. 532–539.

[66] I. A. Kapandji, *The Physiology of the Joints: Upper limb*, 5th ed. Edinburgh: Churchill Livingstone, 2002, vol. 1.

[67] J. L. Pons, *Wearable Robots: Biomechatronic Exoskeletons*, 1st ed. Chichester: Wiley, 2008.

[68] R. J. Purser, "Generalized Fibonacci grids: A new class of structured smoothly adaptive multi-dimensional computational lattices," U.S. Department of Commerce, National Oceanic and Atmospheric Administration, National Weather Service, National Centers for Environmental Prediction, Tech. Rep. Office Note 455, Jan. 2008.

[69] D. Frisch and U. D. Hanebeck, "Deterministic Gaussian sampling with generalized Fibonacci grids," in *Proceedings of the 2021 IEEE 24th International Conference on Information Fusion (FUSION 2021)*, Nov. 2021, pp. 340–347.

[70] D. Frisch, "Transformable deterministic sampling," Ph.D. dissertation, Karlsruhe Institute of Technology, 2024, to appear.

[71] H. Kim, L. Miller, A. Al-Refai, M. Brand, and J. Rosen, "Redundancy resolution of a human arm for controlling a seven DOF wearable robotic system," in *Proceedings of the 2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC 2011)*, Aug. 2011, pp. 3471–3474.

[72] T. Kang, J. He, and S. I. Helms Tillery, "Determining natural arm configuration along a reaching trajectory," *Experimental Brain Research*, vol. 167, pp. 352–361, Oct. 2005.

[73] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, 26th ed. Boston, Munich: Addison-Wesley, 2004.

[74] R. Diankov, "Automated construction of robotic manipulation programs," Ph.D. dissertation, Carnegie Mellon University, 2010.

[75] The MathWorks Inc., "Robotics system toolbox," https://www.mathworks.com/products/robotics.html, accessed: 2024-07-12.

[76] ——, "Global optimization toolbox," https://www.mathworks.com/products/global-optimization.html, accessed: 2024-07-29.

[77] ——, "Parallel computing toolbox," https://www.mathworks.com/products/parallel-computing.html, accessed: 2024-07-29.

[78] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, Nov. 1987.

[79] A. Saudabayev, Z. Rysbek, R. Khassenova, and H. A. Varol, "Human grasping database for activities of daily living with depth, color and kinematic data streams," *Scientific Data*, vol. 5, p. 180101, May 2018.

[80] D. Jansen and H. Büttner, "Real-time Ethernet: The EtherCAT solution," *Computing and Control Engineering*, vol. 15, no. 1, pp. 16–21, Feb. 2004.

[81] A. Albers and S. Rapp, "Model of SGE: System Generation Engineering as basis for structured planning and management of development," in *Design Methodology for Future Products: Data Driven, Agile and Flexible*, 1st ed., D. Krause and E. Heyden, Eds. Cham: Springer, 2022, pp. 27–46.

[82] A. Albu-Schäffer, S. Haddadin, Ch. Ott, A. Stemmer, T. Wimböck, and G. Hirzinger, "The DLR lightweight robot: Design and control concepts for robots in human environments," *Industrial Robot: An International Journal*, vol. 34, no. 5, pp. 376–385, Aug. 2007.

[83] M. Maier, T. Bahls, R. Baver, M. Bihler, M. Chalon, W. Friedl, N. Hoeger, C. Hofmann, A. Kolb, A. M. Sundaram, M. Pfanne, H.-J. Sedlmayr, and N. Seitz, "TINA: The modular torque controlled robotic arm – A study for mars sample return," in *Proceedings of the 2021 IEEE Aerospace Conference*, Mar. 2021, pp. 1–10.

[84] S. Rader, L. Kaul, H. Fischbach, N. Vahrenkamp, and T. Asfour, "Design of a high-performance humanoid dual arm system with inner shoulder joints," in *Proceedings of the 2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids 2016)*, Nov. 2016, pp. 523–529.

[85] C. Benson, "Atherton bikes goes crowdfunding, micro investors to share risk & rewards of growth," *Bikerumor*, Nov. 2020, https://bikerumor.com/atherton-bikes-goes-crowdfunding-micro-investors-to-share-risk-rewards-of-growth/, accessed: 2024-07-10.

[86] Pilz GmbH & Co. KG, "Safe Stop 2 (SS2)," https://www.pilz.com/en-INT/lexicon/sicherer-stopp-2-ss2, accessed: 2024-07-08.

[87] J. Pan, S. Chitta, and D. Manocha, "FCL: A general purpose library for collision and proximity queries," in *Proceedings of the 2012 IEEE International Conference on Robotics and Automation (ICRA 2012)*, May 2012, pp. 3859–3866.

[88] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "ROS: An open-source Robot Operating System," in *Proceedings of the 2009 ICRA Workshop on Open Source Software*, Jan. 2009.

[89] P. Gerum, "The XENOMAI project – Implementing a RTOS emulation framework on GNU/Linux," http://archive.opengroup.org/public/member/q202/documentation/forums/rtf/Copy%20of%20xenomai%20white%20paper%208%20Apr%2002.pdf, accessed: 2024-11-12.

[90] The Linux Foundation, "Real-time Linux," https://wiki.linuxfoundation.org/realtime/start, accessed: 2024-11-12.

[91] S. Chitta, E. Marder-Eppstein, W. Meeussen, V. Pradeep, A. R. Tsouroukdissian, J. Bohren, D. Coleman, B. Magyar, G. Raiola, M. Lüdtke, and E. Fernández Perdomo, "ros_control: A generic and simple control framework for ROS," *The Journal of Open Source Software*, vol. 2, no. 20, p. 456, Dec. 2017.

[92] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot Operating System 2: Design, architecture, and uses in the wild," *Science Robotics*, vol. 7, no. 66, p. eabm6074, May 2022.

[93] Open Robotics, "ROS 2 documentation: Jazzy – Understanding real-time programming," https://docs.ros.org/en/jazzy/Tutorials/Demos/Real-Time-Programming.html, accessed: 2024-11-12.

[94] H. Wei, Z. Shao, Z. Huang, R. Chen, Y. Guan, J. Tan, and Z. Shao, "RT-ROS: A real-time ROS architecture on multi-core processors," *Future Generation Computer Systems*, vol. 56, pp. 171–178, Mar. 2016.

[95] K. Belsare, A. C. Rodriguez, P. G. Sánchez, J. Hierro, T. Kołcon, R. Lange, I. Lütkebohle, A. Malki, J. M. Losa, F. Melendez, M. M. Rodriguez, A. Nordmann, J. Staschulat, and J. von Mendel, "Micro-ROS," in *Robot Operating System (ROS): The Complete Reference*, A. Koubaa, Ed. Cham: Springer, 2023, vol. 7, pp. 3–55.

[96] R. Delgado, B.-J. You, and B. W. Choi, "Real-time control architecture based on Xenomai using ROS packages for a service robot," *Journal of Systems and Software*, vol. 151, pp. 8–19, May 2019.

[97] H. Bruyninckx, "Open robot control software: The OROCOS project," in *Proceedings of the 2001 IEEE International Conference on Robotics and Automation (ICRA 2001))*, May 2001, pp. 2523–2528.

[98] Open Robot Control Software, "rtt_ros_integration – Orocos RTT / ROS integration packages," https://github.com/orocos/rtt_ros_integration, accessed: 2021-06-28.

[99] A. B. Kahn, "Topological sorting of large networks," *Communications of the ACM*, vol. 5, no. 11, pp. 558–562, Nov. 1962.

[100] Open Robotics, "ROS.org – nodelet," http://wiki.ros.org/nodelet, accessed: 2021-06-28.

[101] ——, "ROS.org – dynamic_reconfigure," http://wiki.ros.org/dynamic_reconfigure, accessed: 2021-06-28.

[102] C. I. King, "stress-ng (stress next generation)," https://github.com/ColinIanKing/stress-ng, accessed: 2024-11-12.

[103] M. Bartelmann, B. Feuerbacher, T. Krüger, D. Lüst, A. Rebhan, and A. Wipf, *Theoretische Physik 1: Mechanik*, 1st ed. Berlin, Heidelberg: Springer, 2018.

[104] S. Singh and A. Kaur, "Game development using Unity game engine," in *Proceedings of the 2022 3rd International Conference on Computing, Analytics and Networks (ICAN 2022)*, Nov. 2022, pp. 148–153.

[105] A. Sanders, *An Introduction to Unreal Engine 4*, 1st ed. New York: CRC Press, 2016.

[106] Open Robotics, "ROS.org – urdf," http://wiki.ros.org/urdf, accessed: 2021-11-12.

[107] A. Zea and U. D. Hanebeck, "iviz: A ROS visualization app for mobile devices," *Software Impacts*, vol. 8, p. 100057, May 2021.

[108] The MathWorks Inc., "Simulink," https://www.mathworks.com/products/simulink.html, accessed: 2024-07-12.

[109] ——, "Simscape Multibody," https://www.mathworks.com/products/simscape-multibody.html, accessed: 2024-07-12.

[110] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," in *Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004)*, Sep. 2004, pp. 2149–2154.

[111] T. Erez, Y. Tassa, and E. Todorov, "Simulation tools for model-based robotics: Comparison of Bullet, Havok, MuJoCo, ODE and PhysX," in *Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA 2015)*, May 2015, pp. 4397–4404.

[112] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2012)*, Oct. 2012, pp. 5026–5033.

[113] A. De Luca and W. Book, "Robots with flexible elements," in *Springer Handbook of Robotics*, 2nd ed., B. Siciliano and O. Khatib, Eds. Berlin, Heidelberg: Springer, 2016, pp. 243–282.

[114] F. S. Guzmán, *Numerical Methods for Initial Value Problems in Physics*, 1st ed. Cham: Springer, 2023.

[115] R. Kikuuwe, N. Takesue, A. Sano, H. Mochiyama, and H. Fujimoto, "Fixed-step friction simulation: From classical Coulomb model to modern continuous models," in *Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005)*, Aug. 2005, pp. 1009–1016.

[116] E. Drumwright and D. Shell, "Modeling contact friction and joint friction in dynamic robotic simulation using the principle of maximum dissipation," in *Algorithmic Foundations of Robotics IX*, ser. Springer Tracts in Advanced Robotics, D. Hsu, V. Isler, JC. Latombe, and M. C. Lin, Eds. Berlin, Heidelberg: Springer, 2010, vol. 68, pp. 249–266.

[117] P. Dupont, "Friction modeling in dynamic robot simulation," in *Proceedings of the 1990 IEEE International Conference on Robotics and Automation (ICRA 1990)*, vol. 2, May 1990, pp. 1370–1376.

[118] N. Farhat, V. Mata, Á. Page, and M. Díaz-Rodríguez, "Dynamic simulation of a parallel robot: Coulomb friction and stick-slip in robot joints," *Robotica*, vol. 28, no. 1, pp. 35–45, Jan. 2010.

[119] K. J. Åström and C. Canudas-de-Wit, "Revisiting the LuGre friction model: Stick-slip motion and rate dependence," *IEEE Control Systems Magazine*, vol. 28, no. 6, pp. 101–114, Dec. 2008.

[120] J. Carpentier, G. Saurel, G. Buondonno, J. Mirabel, F. Lamiraux, O. Stasse, and N. Mansard, "The Pinocchio C++ library: A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives," in *Proceedings of the 2019 IEEE/SICE International Symposium on System Integration (SII 2019)*, Jan. 2019, pp. 614–619.

[121] O. Muvengei, J. Kihiu, and B. Ikua, "Dynamic analysis of planar multi-body systems with LuGre friction at differently located revolute clearance joints," *Multibody System Dynamics*, vol. 28, pp. 369–393, Mar. 2012.

[122] Y. Wu, T.-J. Tarn, N. Xi, and A. Isidori, "On robust impact control via positive acceleration feedback for robot manipulators," in *Proceedings of the 1996 IEEE International Conference on Robotics and Automation (ICRA 1996)*, vol. 2, Apr. 1996, pp. 1891–1896.

[123] W. L. Xu and J. D. Han, "Joint acceleration feedback control for robots: Analysis, sensing and experiments," *Robotics and Computer Integrated Manufacturing*, vol. 16, no. 5, pp. 307–320, Oct. 2000.

[124] J. D. Han, Y. C. Wang, D. L. Tan, and W. L. Xu, "Acceleration feedback control for direct-drive motor system," in *Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000)*, vol. 2, Oct. 2000, pp. 1068–1074.

[125] A. Calanca and P. Fiorini, "A rationale for acceleration feedback in force control of series elastic actuators," *IEEE Transactions on Robotics*, vol. 34, no. 1, pp. 48–61, Feb. 2018.

[126] Y. Zimmermann, E. B. Küçüktabak, F. Farshidian, R. Riener, and M. Hutter, "Towards dynamic transparency: Robust interaction force tracking using multi-sensory control on an arm exoskeleton," in *Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2020)*, Oct. 2020, pp. 7417–7424.

[127] S. Haddadin, A. De Luca, and A. Albu-Schäffer, "Robot collisions: A survey on detection, isolation, and identification," *IEEE Transactions on Robotics*, vol. 33, no. 6, pp. 1292–1312, Dec. 2017.

[128] Q. Leboutet, J. Roux, A. Janot, J. R. Guadarrama-Olvera, and G. Cheng, "Inertial parameter identification in robotics: A survey," *Applied Sciences*, vol. 11, no. 9, p. 4303, May 2021.

[129] F. Nori, S. Traversaro, and M. Fallon, "Sensor fusion and state estimation of the robot," in *Humanoid Robotics: A Reference*, A. Goswami and P. Vadakkepat, Eds. Dordrecht: Springer, 2017, pp. 1–29.

[130] W.-H. Zhu and T. Lamarche, "Velocity estimation by using position and acceleration sensors," *IEEE Transactions on Industrial Electronics*, vol. 54, no. 5, pp. 2706–2715, Oct. 2007.

[131] E. Hedberg, J. Norén, M. Norrlöf, and S. Gunnarsson, "Industrial robot tool position estimation using inertial measurements in a complementary filter and an EKF," in *Proceedings of the 20th IFAC World Congress (IFAC 2020)*, ser. IFAC-PapersOnLine, vol. 50, no. 1, Jul. 2017, pp. 12 748–12 752.

[132] W. Chen and M. Tomizuka, "Direct joint space state estimation in robots with multiple elastic joints," *IEEE/ASME Transactions on Mechatronics*, vol. 19, no. 2, pp. 697–706, Apr. 2014.

[133] J. Vihonen, J. Honkakorpi, J. Mattila, and A. Visa, "Geometry-aided angular acceleration sensing of rigid multi-body manipulator using MEMS rate gyros and linear accelerometers," in *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2013)*, Nov. 2013, pp. 2514–2520.

[134] N. Rotella, S. Mason, S. Schaal, and L. Righetti, "Inertial sensor-based humanoid joint state estimation," in *Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA 2016)*, May 2016, pp. 1825–1831.

[135] S. A. B. Birjandi, J. Kühn, and S. Haddadin, "Joint velocity and acceleration estimation in serial chain rigid body and flexible joint manipulators," in *Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2019)*, Nov. 2019, pp. 7503–7509.

[136] ——, "Observer-extended direct method for collision monitoring in robot manipulators using proprioception and IMU sensing," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 954–961, Apr. 2020.

[137] S. Chen and Y. Li, "A method of automatic sensor placement for robot vision in inspection tasks," in *Proceedings of the 2002 IEEE International Conference on Robotics and Automation (ICRA 2002)*, vol. 3, May 2002, pp. 2545–2550.

[138] D. Frisch, K. Li, and U. D. Hanebeck, "Optimal sensor placement for multilateration using alternating greedy removal and placement," in *Proceedings of the 2022 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI 2022)*, Sep. 2022.

[139] I. B. Wijayasinghe, M. N. Saadatzi, S. Abubakar, and D. O. Popa, "A study on optimal placement of accelerometers for pose estimation of a robot arm," in *Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA 2018)*, May 2018, pp. 1444–1451.

[140] W. Niswander, W. Wang, and K. Kontson, "Optimization of IMU sensor placement for the measurement of lower limb joint kinematics," *Sensors*, vol. 20, no. 21, p. 5993, Oct. 2020.

[141] A. W. Mahoney, T. L. Bruns, P. J. Swaney, and R. J. Webster, "On the inseparable nature of sensor selection, sensor placement, and state estimation for continuum robots or "Where to put your sensors and how to use them"," in *Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA 2016)*, May 2016, pp. 4472–4478.

[142] O. Föllinger, U. Konigorski, B. Lohmann, G. Roppenecker, and A. Trächtler, *Regelungstechnik: Einführung in die Methoden und ihre Anwendung*, 12th ed. Berlin: VDE Verlag GmbH, 2016.

[143] B. T. Hinson, "Observability-based guidance and sensor placement," Ph.D. dissertation, University of Washington, 2014.

[144] B. Rakhim, A. Zhakatayev, O. Adiyatov, and H. A. Varol, "Optimal sensor placement of variable impedance actuated robots," in *Proceedings of the 2019 IEEE/SICE International Symposium on System Integration (SII 2019)*, Jan. 2019, pp. 141–146.

[145] J. Carpentier and N. Mansard, "Analytical derivatives of rigid body dynamics algorithms," in *Proceedings of Robotics: Science and Systems (RSS 2018)*, Jun. 2018.

[146] Eigen Developers, "libeigen – eigen," https://gitlab.com/libeigen/eigen, accessed: 2024-08-19.

[147] R. A. Horn and C. R. Johnson, *Matrix Analysis*, 2nd ed. New York: Cambridge University Press, 2013.

[148] C.-T. Chen, *Linear System Theory and Design*, 3rd ed. New York: Oxford Univ. Press, 1999.

[149] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation: Theory, Algorithms and Software*. New York: John Wiley & Sons, 2002.

[150] J. Steinbring and U. D. Hanebeck, "S2KF: The smart sampling Kalman filter," in *Proceedings of the 16th International Conference on Information Fusion (FUSION 2013)*, Jul. 2013, pp. 2089–2096.

[151] D. E. Whitney, "Historical perspective and state of the art in robot force control," in *Proceedings of the 1985 IEEE International Conference on Robotics and Automation (ICRA 1985)*, Mar. 1985, pp. 262–268.

[152] B. Siciliano and L. Villani, *Robot Force Control*, 1st ed. New York: Springer, 2000.

[153] L. Villani and J. De Schutter, "Force control," in *Springer Handbook of Robotics*, 2nd ed., B. Siciliano and O. Khatib, Eds. Berlin, Heidelberg: Springer, 2016, pp. 195–219.

[154] M. Vukobratovic, D. Surdilovic, Y. Ekalo, and D. Katic, *Dynamics and Robust Control of Robot-Environment Interaction*, ser. New Frontiers in Robotics. New Jersey: World Scientific, 2009, vol. 2.

[155] C. Ott, R. Mukherjee, and Y. Nakamura, "Unified impedance and admittance control," in *Proceedings of the 2010 IEEE International Conference on Robotics and Automation (ICRA 2010)*, May 2010, pp. 554–561.

[156] G. Zeng and A. Hemami, "An overview of robot force control," *Robotica*, vol. 15, no. 5, pp. 473–482, Sep. 1997.

[157] S. D. Laycock and A. M. Day, "A survey of haptic rendering techniques," *Computer Graphics Forum*, vol. 26, no. 1, pp. 50–65, Mar. 2007.

[158] C. B. Zilles and J. K. Salisbury, "A constraint-based god-object method for haptic display," in *Proceedings of the 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 1995)*, vol. 3, Aug. 1995, pp. 146–151.

[159] D. C. Ruspini, K. Kolarov, and O. Khatib, "The haptic display of complex graphical environments," in *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH 1997)*, Aug. 1997, pp. 345–352.

[160] F. Conti, F. Barbagli, R. Balaniuk, M. Halg, C. Lu, D. Morris, L. Sentis, E. Vileshin, J. Warren, O. Khatib, and K. Salisbury, "The CHAI libraries," in *Proceedings of Eurohaptics 2003*, Jul. 2003, pp. 496–500.

[161] P. Kadleček, "A practical survey of haptic APIs," Bachelor's thesis, Charles University in Prague, 2010.

[162] C. L. Clover, G. R. Luecke, J. J. Troy, and W. A. McNeely, "Dynamic simulation of virtual mechanisms with haptic feedback using industrial robotics equipment," in *Proceedings of the 1997 IEEE International Conference on Robotics and Automation (ICRA 1997)*, vol. 1, Apr. 1997, pp. 724–730.

[163] L. B. Rosenberg, "Virtual fixtures: Perceptual tools for telerobotic manipulation," in *Proceedings of the 1993 IEEE Virtual Reality Annual International Symposium*, Sep. 1993, pp. 76–82.

[164] L. Tching, G. Dumont, and J. Perret, "Interactive simulation of CAD models assemblies using virtual constraint guidance," *International Journal on Interactive Design and Manufacturing (IJIDeM)*, vol. 4, pp. 95–102, May 2010.

[165] J. J. Abbott and A. M. Okamura, "Virtual fixture architectures for telemanipulation," in *Proceedings of the 2003 IEEE International Conference on Robotics and Automation (ICRA 2003)*, vol. 2, Sep. 2003, pp. 2798–2805.

[166] N. Zafer, "Constraint-based haptic rendering of a parametric surface," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 221, no. 3, pp. 507–517, May 2007.

[167] P. Marayong, M. Li, A. M. Okamura, and G. D. Hager, "Spatial motion constraints: Theory and demonstrations for robot guidance using virtual fixtures," in *Proceedings of the 2003 IEEE International Conference on Robotics and Automation (ICRA 2003)*, vol. 2, Sep. 2003, pp. 1954–1959.

[168] A. Escande, N. Mansard, and P.-B. Wieber, "Hierarchical quadratic programming: Fast online humanoid-robot motion generation," *The International Journal of Robotics Research*, vol. 33, no. 7, pp. 1006–1028, Jun. 2014.

[169] A. Del Prete, "Joint position and velocity bounds in discrete-time acceleration/torque control of robot manipulators," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 281–288, Jan. 2018.

[170] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed., ser. Springer Series in Operations Research and Financial Engineering. New York: Springer, 2006.

[171] W. S. Newman, "Stability and performance limits of interaction controllers," *Journal of Dynamic Systems, Measurement, and Control*, vol. 114, no. 4, pp. 563–570, Dec. 1992.

[172] F. Cordoni, L. Persio, and R. Muradore, "A variable stochastic admittance control framework with energy tank," in *Proceedings of the 21st IFAC World Congress (IFAC 2020)*, ser. IFAC-PapersOnLine, vol. 53, no. 2, Jul. 2020, pp. 9986–9991.

[173] C. Schindlbeck and S. Haddadin, "Unified passivity-based Cartesian force/impedance control for rigid and flexible joint robots via task-energy tanks," in *Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA 2015)*, May 2015, pp. 440–447.

[174] C. T. Landi, F. Ferraguti, L. Sabattini, C. Secchi, and C. Fantuzzi, "Admittance control parameter adaptation for physical human-robot interaction," in *Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA 2017)*, May 2017, pp. 2911–2916.

[175] C. T. Landi, F. Ferraguti, L. Sabattini, C. Secchi, M. Bonfè, and C. Fantuzzi, "Variable admittance control preventing undesired oscillating behaviors in physical human-robot interaction," in *Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2017)*, Sep. 2017, pp. 3611–3616.

[176] A. Albu-Schäffer, "Regelung von Robotern mit elastischen Gelenken am Beispiel der DLR-Leichtbauarme," Ph.D. dissertation, Technische Universität München, 2002.

[177] B. Armstrong-Helouvry, "Stick-slip arising from Stribeck friction," in *Proceedings of the 1990 IEEE International Conference on Robotics and Automation (ICRA 1990)*, vol. 2, May 1990, pp. 1377–1382.

[178] L. Di Gaspero, "QuadProg++," https://github.com/liuq/QuadProgpp, accessed: 2024-06-14.

[179] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpOASES: A parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, pp. 327–363, Apr. 2014.

[180] E. Samur, *Performance Metrics for Haptic Interfaces*, 1st ed. London: Springer, 2012.

[181] F. Fazlollahi and K. J. Kuchenbecker, "Haptify: A measurement-based benchmarking system for grounded force-feedback devices," *IEEE Transactions on Robotics*, vol. 39, no. 2, pp. 1622–1636, Apr. 2023.

[182] N. Landin, J. M. Romano, W. McMahan, and K. J. Kuchenbecker, "Dimensional reduction of high-frequency accelerations for haptic rendering," in *Haptics: Generating and Perceiving Tangible Sensations, Eurohaptics 2010, Part II*, Jul. 2010, pp. 79–86.

[183] M. Awais, M. Y. Saeed, M. S. A. Malik, M. Younas, and S. R. I. Asif, "Intention based comparative analysis of human-robot interaction," *IEEE Access*, vol. 8, pp. 205 821–205 835, Nov. 2020.

[184] N. Stefanov, A. Peer, and M. Buss, "Online intention recognition for computer-assisted teleoperation," in *Proceedings of the 2010 IEEE International Conference on Robotics and Automation (ICRA 2010)*, May 2010, pp. 5334–5339.

[185] D. Gehrig, P. Krauthausen, L. Rybok, H. Kuehne, U. D. Hanebeck, T. Schultz, and R. Stiefelhagen, "Combined intention, activity, and motion recognition for a humanoid household robot," in *Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2011)*, Sep. 2011, pp. 4819–4825.

[186] P. Rößler, O. C. Schrempf, and U. D. Hanebeck, "Stochastic prediction of waypoints for extended-range telepresence applications," in *Proceedings of the 2nd International Workshop on Human Centered Robotic Systems (HCRS 2006)*, Oct. 2006, pp. 85–89.

[187] S. Jain and B. Argall, "Recursive bayesian human intent recognition in shared-control robotics," in *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2018)*, Oct. 2018, pp. 3905–3912.

[188] T. Petković, D. Puljiz, I. Marković, and B. Hein, "Human intention estimation based on hidden Markov model motion validation for safe flexible robotized warehouses," *Robotics and Computer-Integrated Manufacturing*, vol. 57, pp. 182–196, Jun. 2019.

[189] D. Puljiz, B. Zhou, K. Ma, and B. Hein, "HAIR: Head-mounted AR intention recognition," in *Proceedings of the 4th International Workshop on Virtual, Augmented, and Mixed Reality for Human-Robot Interaction (VAM-HRI 2021)*, Mar. 2021.

[190] J. Pettersson and P. Falkman, "Human movement direction classification using virtual reality and eye tracking," *Procedia Manufacturing*, vol. 51, pp. 95–102, Jan. 2020.

[191] A. Clarence, J. Knibbe, M. Cordeil, and M. Wybrow, "Unscripted retargeting: Reach prediction for haptic retargeting in virtual reality," in *Proceedings of the 2021 IEEE Conference on Virtual Reality and 3D User Interfaces (VR 2021)*, Mar. 2021, pp. 150–159.

[192] C. Gomez Cubero and M. Rehm, "Intention recognition in human robot interaction based on eye tracking," in *Human-Computer Interaction – INTERACT 2021, 18th IFIP TC 13 International Conference, Part III*, Aug. 2021, pp. 428–437.

[193] A. Gasparetto, P. Boscariol, A. Lanzutti, and R. Vidoni, "Path planning and trajectory planning algorithms: A general overview," in *Motion and Operation Planning of Robotic Systems: Background and Practical Approaches*, 1st ed., ser. Mechanisms and Machine Science, G. Carbone and F. Gomez-Bravo, Eds. Cham: Springer, 2015, vol. 29, pp. 3–27.

[194] S. M. LaValle, *Planning Algorithms*. Cambridge: Cambridge University Press, 2006.

[195] L. E. Kavraki and S. M. LaValle, "Motion planning," in *Springer Handbook of Robotics*, 2nd ed., B. Siciliano and O. Khatib, Eds. Berlin, Heidelberg: Springer, 2016, pp. 139–161.

[196] T. Sandakalum and M. H. Ang, Jr., "Motion planning for mobile manipulators – A systematic review," *Machines*, vol. 10, no. 2, p. 97, Jan. 2022.

[197] A. Hermann, F. Drews, J. Bauer, S. Klemm, A. Roennau, and R. Dillmann, "Unified GPU voxel collision detection for mobile manipulation planning," in *Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2014)*, Sep. 2014, pp. 4154–4160.

[198] T. Bandyopadhyay, K. S. Won, E. Frazzoli, D. Hsu, W. S. Lee, and D. Rus, "Intention-aware motion planning," in *Algorithmic Foundations of Robotics X: Proceedings of the Tenth Workshop on the Algorithmic Foundations of Robotics*, Feb. 2013, pp. 475–491.

[199] J. S. Park, C. Park, and D. Manocha, "I-Planner: Intention-aware motion planning using learning-based human motion prediction," *The International Journal of Robotics Research*, vol. 38, no. 1, pp. 23–39, Jan. 2019.

[200] T. J. D. Berstad, M. Riegler, H. Espeland, T. de Lange, P. H. Smedsrud, K. Pogorelov, H. Kvale Stensland, and P. Halvorsen, "Tradeoffs using binary and multiclass neural network classification for medical multidisease detection," in *Proceedings of the 2018 IEEE International Symposium on Multimedia (ISM 2018)*, Dec. 2018, pp. 1–8.

[201] R. Kothari, Z. Yang, C. Kanan, R. Bailey, J. B. Pelz, and G. J. Diaz, "Gaze-in-wild: A dataset for studying eye and head coordination in everyday activities," *Scientific Reports*, vol. 10, p. 2539, Feb. 2020.

[202] V. Sitzmann, A. Serrano, A. Pavel, M. Agrawala, D. Gutierrez, B. Masia, and G. Wetzstein, "Saliency in VR: How do people explore virtual environments?" *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 4, pp. 1633–1642, Apr. 2018.

[203] K. J. Emery, M. Zannoli, L. Xiao, J. Warren, and S. S. Talathi, "OpenNEEDS: A dataset of gaze, head, hand, and scene signals during exploration in open-ended VR environments," in *ETRA '21 Short Papers: ACM Symposium on Eye Tracking Research and Applications*, May 2021, pp. 1–7.

[204] TensorFlow Developers, "TensorFlow: Large-scale machine learning on heterogeneous systems," https://www.tensorflow.org/, accessed: 2024-11-13.

[205] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, no. 85, pp. 2825–2830, Oct. 2011.

[206] L. Berscheid and T. Kröger, "Jerk-limited real-time trajectory generation with arbitrary target states," in *Proceedings of Robotics: Science and Systems 2021 (RSS 2021)*, Jul. 2021.

[207] T. P. Peixoto, "The graph-tool Python library," Figshare, doi: 10.6084/m9.figshare.1164194, Sep. 2014.

[208] I. A. Şucan, M. Moll, and L. E. Kavraki, "The open motion planning library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, Dec. 2012.

[209] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2017)*, Jul. 2017, pp. 77–85.

[210] H. Zhao, L. Jiang, J. Jia, P. Torr, and V. Koltun, "Point transformer," in *Proccedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV 2021)*, Oct. 2021, pp. 16 239–16 248.

Datasheets

[D1] Haption, *Scale1™*, https://downloads.haption.com/marketing/datasheet/Datasheet-Hardware-Scale1-2023.pdf, 2023, accessed: 2024-10-12.

[D2] Universal Robots, *UR16e Technical Specification*, https://www.universal-robots.com/media/1811483/ur16e-rgb-fact-sheet-landscape-a4.pdf, Nov. 2023, accessed: 2024-07-19.

[D3] Sensodrive, *Sensojoint*, https://www.sensodrive.de/products/torque-technology-senso-joint.php, accessed: 2021-09-14.

[D4] HTC Corporation, *HTC VIVE Tracker (2018) Developer Guidelines*, https://dl.vive.com/Tracker/Guideline/HTC_Vive_Tracker(2018)_Developer+Guidelines_v1.0.pdf, Mar. 2018, accessed: 2024-07-31.

[D5] SEW Eurodrive, *Movidrive MDX60B/61B*, https://download.sew-eurodrive.com/download/pdf/16838017.pdf, Sep. 2010, accessed: 2024-08-29.

[D6] Sick, *TIM781S-2174104 – TiM-S*, https://cdn.sick.com/media/pdf/9/49/149/dataSheet_TIM781S-2174104_1096363_en.pdf, Jun. 2024, accessed: 2024-06-26.

[D7] Elgo, *EMAX*, https://www.elgo.de/fileadmin/user_upload/pdf/flyer/sensors/EMAX-EMAL-00-FL-E.pdf, 2019, accessed: 2024-06-26.

[D8] Botasys, *SensONE*, https://www.botasys.com/force-torque-sensors/sensone, accessed: 2024-06-26.

[D9] Beckhoff, *EP3752-0000 2 x 3-axis accelerometers*, https://download.beckhoff.com/download/document/io/ethercat-box/ep3752en.pdf, Sep. 2022, accessed: 2024-07-04.

[D10] Gable Systems, *Gable IMU ONE SERIES Datasheet*, https://gable-imu.nl/ONE-SERIES_Datasheet.pdf, Nov. 2022, accessed: 2024-07-04.

[D11] Invensense, *MPU-9250 Product Specification*, https://invensense.tdk.com/wp-content/uploads/2015/02/PS-MPU-9250A-01-v1.1.pdf, Jun. 2016, accessed: 2024-07-04.

[D12] B-Command, *rotarX – Through-Bore Slip Rings*, https://www.rotarx.com/wp-content/uploads/B-COMMAND-rotarX-03-Through-Bore-Slip-Rings.pdf, 2019, accessed: 2024-06-26.

[D13] Nanotec, *Betriebsanleitung Brems-Chopper*, https://www.nanotec.com/fileadmin/files/Handbuecher/accessories/brake-chopper/BC72_50_Betriebsanleitung_V1.2.0.pdf, Dec. 2021, accessed: 2024-07-04.

[D14] PILZ, *PNOZmulti 2 Technical catalogue*, https://www.pilz.com/download/open/PNOZmulti_2_Techn_Catalogue_1006421-EN-01.pdf, Feb. 2023, accessed: 2024-06-26.

[D15] Microsoft, *Microsoft HoloLens 2 Fact Sheet*, https://news.microsoft.com/wp-content/uploads/prod/2019/02/Fact-Sheet_HoloLens2.pdf, Feb. 2019, accessed: 2024-07-15.

[D16] XSENS, *MTi 1-series Datasheet*, https://www.xsens.com/hubfs/Downloads/Manuals/MTi-1-series-datasheet.pdf, Dec. 2019, accessed: 2024-08-28.

[D17] STMicroelectronics, *LIS3DH Datasheet*, https://www.st.com/resource/en/datasheet/lis3dh.pdf, Dec. 2016, accessed: 2024-08-28.

[D18] 3D Systems, *Haptic Devices: Add the sense of Touch to your digital world*, https://de.3dsystems.com/sites/default/files/2019-04/3d-systems-haptic-device-en-letter-web-2018-08-06.pdf, Aug. 2018, accessed: 2024-09-18.

Supervised Student Theses

For privacy reasons, the names of the students are omitted in the following list.

[S1] "Referenzen-basierte Entwicklung einer kinästhetisch-haptischen Schnittstelle für die Anwendung in der virtuellen Realität," Bachelor's thesis, Karlsruhe Institute of Technology, May 2022.

[S2] "Entwurf und Implementierung einer Regelung für eine große haptische Schnittstelle mit sechs Freiheitsgraden," Master's thesis, Karlsruhe Institute of Technology, Jan. 2022.

[S3] "Sensoreinsatzplanung und Realisierung eines Zustandsschätzers für eine kinematische Kette," Master's thesis, Karlsruhe Institute of Technology, May 2022.

[S4] "Aufbau einer Simulation für einen XR-Prototyp eines Robotersystems," Bachelor's thesis, Karlsruhe Institute of Technology, Apr. 2023.

[S5] "Maschinelles Lernen zur Intentionserkennung in 3D-Umgebungen," Master's thesis, Karlsruhe Institute of Technology, Jun. 2022.

[S6] "Intentionsbasierte Echtzeit-Bewegungsplanung für mobile Manipulatoren in der Mensch-Roboter-Interaktion," Master's thesis, Karlruhe Institute of Technology, Jun. 2023.

[S7] "A nonlinear model predictive controller for various ground vehicles emulated on a holonomic robot," Master's thesis, Karlsruhe Institute of Technology, Jan. 2021.

[S8] "Enhanced drive train control for a holonomic robot," Bachelor's thesis, Karlsruhe Institute of Technology, Dec. 2020.

[S9] "Verfeinerung des Dynamikmodells eines autonomen Hydraulikbaggers durch Identifikation der Dynamikparameter sowie unbekannter Einflussfaktoren auf die Kraftschätzung am Endeffektor," Bachelor's thesis, Karlsruhe Institute of Technology, Jan. 2024.

The Author's Publications

[O1] M. Fennel, M. Walker, and U. D. Hanebeck, "HapticGiant: A novel very large kinesthetic haptic interface with hierarchical force control," *IEEE Transactions on Haptics*, 2024, under review.

[O2] M. Fennel, L. Driller, A. Zea, and U. D. Hanebeck, "Calibration-free IMU-based kinematic state estimation for robotic manipulators," in *Proceedings of the 2022 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI 2022)*, Sep. 2022, winner of the best paper award.

[O3] ——, "Observability-based placement of inertial sensors on robotic manipulators for kinematic state estimation," in *Proceedings of the 22nd IFAC World Congress (IFAC 2023)*, ser. IFAC-PapersOnLine, vol. 56, no. 2, Jul. 2023, pp. 5293–5299.

[O4] M. Fennel, A. Zea, and U. D. Hanebeck, "Optimization-driven design of a kinesthetic haptic interface with human-like capabilities," *IEEE Transactions on Haptics*, vol. 15, no. 1, pp. 45–50, Dec. 2021.

[O5] M. Fennel, A. Zea, J. Mangler, A. Roennau, and U. D. Hanebeck, "Haptic rendering of arbitrary serial manipulators for robot programming," *IEEE Control Systems Letters*, vol. 6, pp. 716–721, Jun. 2021.

[O6] M. Fennel, S. Geyer, and U. D. Hanebeck, "RTCF: A framework for seamless and modular real-time control with ROS," *Software Impacts*, vol. 9, p. 100109, Aug. 2021.

[O7] M. Fennel, S. Garbay, A. Zea, and U. D. Hanebeck, "Intention estimation with recurrent neural networks for mixed reality environments," in *Proceedings of the 26th International Conference on Information Fusion (FUSION 2023)*, Jun. 2023.

[O8] ——, "Time series data of gaze, head pose, hand pose, and object positions for object approaches with a given intention," Zenodo, doi: 10.5281/zenodo.7687773, Mar. 2023.

[O9] M. Fennel, A. Zea, and U. D. Hanebeck, "Haptic-guided path generation for remote car-like vehicles," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 4088–4095, Apr. 2021.

[O10] ——, "Intuitive and immersive teleoperation of robot manipulators for remote decontamination," *at – Automatisierungstechnik*, vol. 70, no. 10, pp. 888–899, Oct. 2022.

[O11] A. Zea, M. Fennel, and U. D. Hanebeck, "Robot joint tracking with mobile depth cameras for augmented reality applications," in *Proceedings of the 25th International Conference on Information Fusion (FUSION 2022)*, Jul. 2022.