

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Move Your Haptic: Enhancing Virtual Reality Experiences through Moving Haptic Devices

André Júlio Moreira



Mestrado em Engenharia Informática e Computação

Supervisor: Rui Pedro Amaral Rodrigues

Supervisor: Daniel Filipe Martins Tavares Mendes

July 26, 2024

Move Your Haptic: Enhancing Virtual Reality Experiences through Moving Haptic Devices

André Júlio Moreira

Mestrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

President: Prof. António Augusto de Sousa

Referee: Prof. Rui Pedro Amaral Rodrigues

Referee: Prof. Paulo Miguel de Jesus Dias

July 26, 2024

Abstract

In virtual reality experiences, the typical VR setup falls short in engaging senses beyond sight and sound the virtual environment. Various haptic devices have been developed and explored to enrich the immersive experience, spanning grounded, wearable, and graspable devices.

Despite advances in simulating intricate physical properties, challenges persist, particularly in engaging the user's sense of touch. A major hurdle with the majority of graspable devices is that they are stationary. Most of these devices must be retrieved from the precise spot where they were last dropped, significantly restricting the potential for diverse and dynamic interactions. On the other hand, innovative solutions like drones offer a great space simulation but currently fall short in providing the nuanced and meaningful interactions offered by more conventional graspable devices.

To achieve this goal, we propose a moving robot designed to accommodate and transport graspable devices. This robot can autonomously adjust its position based on the virtual object the user wants to interact with. This innovation enables the simulation of diverse virtual objects in different locations using a single moving device, aiming to create dynamic interactions within virtual environments with fewer physical components.

To evaluate our solution and determine if it offers a better alternative to the current state-of-the-art, we conducted user tests with a task comparing our approach with a baseline, in order to address our research questions. The baseline involved using multiple haptic props, while our robot transported a single haptic prop. The results indicated no statistical difference between the two approaches in terms of user experience. However, some participants were able to guess the approach behind each task after being informed how both approaches worked. This suggests that our approach is viable when users are fully immersed and it can create more dynamic experiences, but there is room for improvement, mainly in speeding up the positioning process.

ACM Classification: Human-centered computing → Human computer interaction (HCI); Haptic devices; Virtual reality.

Keywords: Virtual Reality, Haptic Devices, Immersive Experiences, Graspable Devices, Mobile Base, Spatial Interactions

Resumo

Nas experiências de realidade virtual, a configuração típica da RV não consegue envolver outros sentidos para além da visão e do som no ambiente virtual. Foram desenvolvidos e explorados vários dispositivos hápticos para enriquecer a experiência de imersão, abrangendo dispositivos fixos, vestíveis e agarráveis.

Apesar dos avanços na simulação de propriedades físicas complexas, persistem desafios, em particular no que diz respeito ao sentido do tato do utilizador. Um grande obstáculo na maioria dos dispositivos agarráveis é o facto de serem estacionários. A maioria destes dispositivos tem de ser recuperada do local exato onde foi pousada pela última vez, o que restringe significativamente o potencial para interações diversas e dinâmicas. Por outro lado, soluções inovadoras como os drones oferecem uma excelente simulação do espaço, mas atualmente não conseguem proporcionar as interações diferenciadas e significativas oferecidas pelos dispositivos que podem ser agarrados.

Para atingir este objetivo, propomos um robot móvel concebido para acomodar e transportar dispositivos agarráveis. Este robot pode ajustar autonomamente a sua posição com base no objeto virtual com o qual o utilizador pretende interagir. Esta inovação permite a simulação de diversos objectos virtuais em diferentes locais utilizando um único dispositivo móvel, com o objetivo de criar interações dinâmicas em ambientes virtuais com menos componentes físicos.

Para avaliar a nossa solução e determinar se ela oferece uma melhor alternativa ao atual estado da arte, realizámos testes de utilizador com uma tarefa que comparava a nossa abordagem com uma abordagem de referência, de modo a responder às nossas questões de investigação. A abordagem de referência envolvia a utilização de múltiplos adereços hápticos, enquanto o nosso robô transportava um único adereço háptico. Os resultados não indicaram qualquer diferença estatística entre as duas abordagens em termos de experiência do utilizador. No entanto, alguns participantes foram capazes de adivinhar a abordagem por detrás de cada tarefa depois de serem informados sobre o funcionamento de ambas as abordagens. Isto sugere que a nossa abordagem é viável quando os utilizadores estão totalmente imersos e pode criar experiências mais dinâmicas, mas há espaço para melhorias, principalmente na aceleração do processo de posicionamento.

ACM Classification: Human-centered computing → Human computer interaction (HCI); Haptic devices; Virtual reality.

Palavras-chave: Realidade Virtual, Dispositivos Hápticos, Experiências Imersivas, Dispositivos Agarráveis, Base Móvel, Interações Espaciais

Acknowledgements

First off, I want to thank my parents and family. Your endless support and encouragement have meant the world to me and made it possible for me to get the education I have today.

To all my friends, thank you for always being there for me, cheering me on, and making sure I had plenty of fun along the way. I couldn't have done it without you.

A big shoutout to NIAEFEUP for the last four amazing years. This experience has helped me grow so much, and I've learned more than I ever imagined. I hope I've managed to make a positive impact on others just as much as this place has on me.

Finally, I would like to extend my sincere appreciation to my supervisors, Daniel Mendes and Rui Rodrigues. Your guidance and insightful feedback have been crucial in steering me in the right direction. Thank you for your dedication and for trusting my abilities to take on your challenges.

André Júlio Moreira

*“These walls are funny. First you hate ’em, then you get used to ’em.
Enough time passes, you get so you depend on them.”*

Red - Shawshank Redemption

Contents

1	Introduction	1
1.1	Context	1
1.2	Motivation	2
1.3	Objectives	2
1.4	Document structure	2
2	State of the art	3
2.1	Haptic properties and techniques	3
2.1.1	Shape and Grasping	3
2.1.2	Texture	9
2.1.3	Muscle stimulation	11
2.1.4	Gravity and External Forces	12
2.1.5	Stiffness	14
2.1.6	Temperature	15
2.2	Grounded devices	16
2.3	Portable devices	17
2.3.1	Wearables devices	17
2.3.2	Graspable devices	18
2.4	Discussion	19
3	Move your Haptic	22
3.1	Problem statement and hypothesis	22
3.1.1	Our approach	23
3.2	Device	24
3.2.1	Hardware	25
3.2.2	Software	27
3.3	Controlling the Device	27
3.3.1	Remote start and stop	28
3.3.2	Movement by value	28
3.3.3	Rotation first	29
3.3.4	Segmented path	30
3.4	Communication protocol	31
3.5	Hand detection	31
3.6	Interaction	32
3.6.1	Height check	32
3.6.2	Vertical speed check	32
3.6.3	Using the hand position	33
3.6.4	Ensuring natural interactions with robot control	33

3.7	Candidate object detection	34
3.8	Custom Unity library	34
3.9	Architecture	35
3.10	Device limitations	37
3.11	Chapter summary	37
4	Evaluation and Results	39
4.1	Target haptic device	39
4.2	Setup and test environment	39
4.3	Task definition	41
4.4	Our approach	43
4.5	Baseline	44
4.6	Methodology	45
4.7	Data collection	46
4.8	Participants	47
4.9	Results	47
4.9.1	Analysis of tasks data	47
4.9.2	Analysis of the order identification question	49
4.9.3	Analysis of distance and time travelled	51
4.9.4	Results discussion	51
4.10	Chapter summary	52
5	Conclusion	54
5.1	Future Work	55
	References	57
A	Mecanum wheels	62

List of Figures

2.1	X-Rings simulating a radially symmetric object [18].	4
2.2	Several objects simulated by Shape-A-Getti [4].	5
2.3	Comparison between Shapeshift and PoCoPo	5
2.4	PuPoP simulating several shapes with different primitives [35].	6
2.5	InflatableBots changing to adapt to a 3D scenario [17].	6
2.6	Comparison between Haptobend, WeATaViX and Tangiball.	7
2.7	Haptic prop appropriation using a quadcopter [1].	8
2.8	Assembly process of a passive haptic prop [42].	8
2.9	Complex shapes simulation with Wolverine [9].	8
2.10	Comparison between Grability and Claw simulating a cube.	9
2.11	Wireality simulating a complex shape [14].	9
2.12	Snake Charmer changing the simulated surface [3].	10
2.13	Materializer showcasing three different textures [31].	11
2.14	Modularity of the Haptic Revolver [37].	11
2.15	Impact simulation using electrical muscle stimulation [25].	12
2.16	Muscle movement manipulation using electrical stimulation [26].	12
2.17	Thor's Hammer [19].	13
2.18	Shifty graspable haptic device [39].	14
2.19	The several possible states of the Drag:on device [40].	14
2.20	Bstick [23].	15
2.21	PATCH and its different contact points of temperature [16].	15
2.22	Effect produced by alternating the current of the device [24].	16
2.23	Process of providing chemicals to simulate different sensations [27].	16
2.24	Hiro-III haptic device [13].	17
3.1	Sketch of the idealized solution	24
3.2	Final implementation of the device transporting a haptic prop	24
3.3	Device hardware schematic	25
3.4	Components organization of the device	26
3.5	How to operate the driver to move the device.	26
3.6	Significant deviation caused by a misaligned route	29
3.7	Timeline that causes misalignment	29
3.8	Timeline with improved solution	30
3.9	Improved sequence of operations to ensure fast positionment	30
3.10	State machine of the haptic interactable state	33
3.11	Architecture of the solution	36
3.12	Detailed architecture of the library API	37

4.1	Haptic prop.	40
4.2	Virtual and physical environment used in the user tests.	40
4.3	Simple walls to prevent the device from falling from the platform.	41
4.4	Butterflies distracting the user.	42
4.5	Validating the apple with the supervisor.	42
4.6	Tube to place the apple.	43
4.7	Validating the apple with the supervisor.	43
4.8	Sequence of interaction from a user in the test.	44
4.9	The three haptic props used in the experience.	44
4.10	User in the experience using multiple haptic props	45
4.11	Histogram of the confidence levels.	50
A.1	Structure of a mecanum wheel [15].	62
A.2	How to achieve a desired position using mecanum wheels [21].	63

List of Tables

2.1	Overview of Haptic Devices	21
4.1	Median (IQR), and Wilcoxon Signed Ranks test results for each of the questions for both approaches.	49

Abreviaturas e Símbolos

VR	Virtual Reality
2DOF	two degrees of freedom
3DOF	three degrees of freedom
3D	three dimensions

Chapter 1

Introduction

Virtual reality (VR) is a powerful mechanism that significantly enhances virtual experiences. Despite head-mounted displays (HMDs) being relatively expensive for the general public and presenting challenges such as nausea and discomfort that still need to be addressed, large companies are making significant advancements in this area to make VR more accessible and comfortable.

Currently, the most common virtual experiences primarily provide visual and auditory feedback, with users interacting through gestures or controllers. However, the use of controllers, or just gestures, represents a major limitation in achieving truly immersive experiences. When interacting with the real world, the sense of touch complements visual and auditory feedback and is one of the most important types of feedback for humans [32]. Therefore, it is crucial to explore the field of haptics, which studies how to bring users closer to a virtual experience that closely mimics reality, alongside the ongoing improvements in head-mounted displays.

1.1 Context

The adoption of virtual reality (VR) technology has impacted interaction with digital content, offering immersive experiences that surpass traditional modes of interaction. While visual and auditory feedback are critical components of VR environments, they alone may not suffice to fully immerse users in virtual experiences. To address this limitation, researchers and developers have turned their attention to haptic feedback as a means to enhance the realism and immersion of VR interactions.

Although early implementations have primarily utilized simple forms of haptic feedback, such as vibrations commonly found in commercial controllers, advancements in haptic technology now introduce a broader collection of tactile sensations, including texture, temperature, stiffness, and the simulation of external forces.

1.2 Motivation

The current focus of VR technology primarily revolves around stimulating vision and sound, leaving a gap in providing comprehensive sensory experiences. This limitation has caused advancements in haptic devices aimed at enhancing immersion in virtual environments. Despite notable progress in VR technology, existing solutions still struggle to replicate the richness of real-world sensory interactions, particularly in terms of tactile feedback. While several haptic devices have been introduced to address this gap, there is yet a lot of space for improvement over current approaches.

Moreover, commercially available solutions aimed at sensory experiences in VR remain relatively underdeveloped. Existing headsets offer only basic tactile feedback, such as vibrations, which lack the complexity needed to simulate a wide range of sensations effectively. Consequently, users are deprived of the potential sensory experiences that could significantly enhance the immersive and engaging nature of VR environments.

1.3 Objectives

The primary aim of this research is to explore an approach that addresses one of the main issues with graspable devices: the lack of movement. We aim to overcome the limitation that requires having the same number of haptic devices as there are different virtual objects. Our goal is to develop a robot capable of moving haptic devices to mimic the location of virtual objects.

1.4 Document structure

The remainder of the document is structured into four main sections. Chapter 2, State of the Art, conducts a comprehensive analysis and research of the current landscape of haptic devices applied in virtual reality. This chapter thoroughly examines and discusses various types of haptic devices, highlighting their strengths and weaknesses. In Chapter 3, we reassess the existing landscape of haptic technology, identifying areas for improvement and defining the specific problem we aim to address. Through discussion and analysis, we formulate a potential hypothesis to guide our approach to developing a solution. In this Chapter, we also materialize and build upon our proposal, identifying key issues and implementation details along with the development of a proof of concept for a moving device. In Chapter 4, we evaluate and analyse the limitations and feasibility of our device, describing the steps taken to understand how the device performs. Finally, Chapter 5 serves as the conclusion of this preliminary report, providing a summary of the findings and insights gained throughout the research process.

Chapter 2

State of the art

This chapter explores advanced haptic devices, focusing on shape, grasping, texture, gravity, stiffness and temperature in immersive virtual experiences. It covers various approaches to simulating shapes, from radially symmetric objects to shape primitives. The discussion extends to devices addressing the sense of grasping and those exploring texture through vibrations or dynamic surface changes. Additionally, it touches on innovative solutions for interaction feedback, such as using drones or electrical muscle stimulation and chemical haptics. This overview provides a glimpse into the diverse landscape of state-of-the-art haptic technologies and concludes with a critical view and discussion about the current state and future directions of these technologies, offering insights into their potential applications and limitations.

2.1 Haptic properties and techniques

Understanding and replicating haptic properties is essential for creating a more immersive and realistic interaction in virtual environments. Haptic feedback adds new dimensions to digital experiences, allowing users to feel and interact with virtual objects in a way that goes beyond visual and auditory stimuli. Technology can effectively bridge the gap between the digital and physical worlds by simulating object properties such as shape and texture and sensations like grasping and muscle feedback. This integration provides users with a more engaging and enriching experience that combines the benefits of both realms.

2.1.1 Shape and Grasping

One of the most important things humans notice and connect with their visual stimuli when interacting with an object is the sense of touch, which might come from the object's shape and the feeling of grasping perceived by the user. Several approaches have been studied on how human senses can be stimulated to simulate the interaction with physical objects based on their shape.

A fundamental aspect that captures human attention and establishes a connection with the visual stimuli being perceived during object interaction is the sense of touch. This tangible experience arises from the object's form and the actual sensation of grasping the user undergoes.

Various studies have been carried out in this realm, and different approaches have been developed to engage the human senses, aiming to reproduce the interaction with physical objects accurately by considering their shapes and the nuanced dynamics of grasping.

To simulate simpler objects that pertain to the property of being radially symmetric, several devices have been introduced and studied to render these types of shapes. One might think it is insufficient, but several everyday objects have this property [18], such as bottles, cups, and fruits, among many others. Several devices manipulate this property to achieve the simulation of different objects using a single device.

The X-Rings device [18] is built with radially expandable layers. Each layer has a DC motor that controls the object's radius. This allows the device to simulate objects that are not cylindrical but still are radially symmetric, as seen in figure 2.1. It would even be possible to simulate objects that do not follow a radially symmetrical shape but still follow some circular shape if we expand each device pin individually instead of the whole layer. However, this would pose a significant challenge in this device due to the amount of space linear actuators would take [18]. Following a similar approach, we can find the BStick [23], where each finger is allocated to a linear actuator capable of changing its position to simulate different shapes.

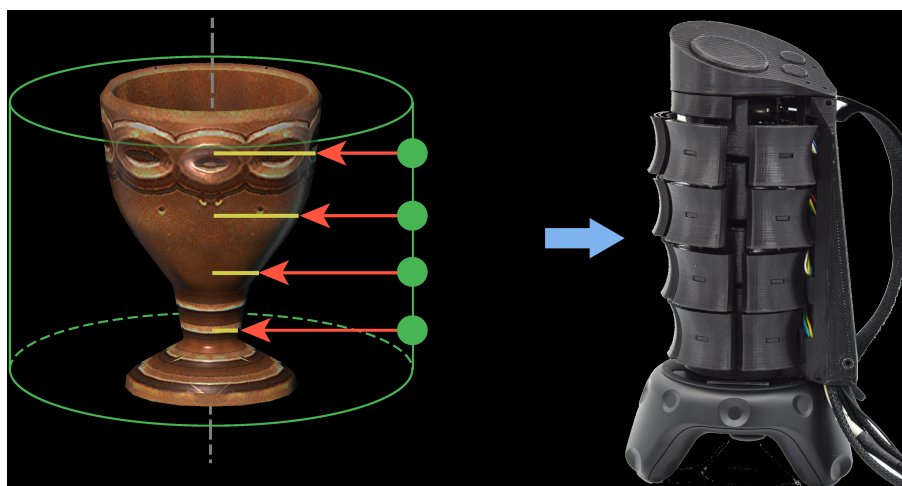


Figure 2.1: X-Rings simulating a radially symmetric object [18]

On a different approach, Shape-A-Getti [4] utilises twisting rods to simulate radially symmetric shapes effectively. It uniquely expands the radius, but exclusively at the device's centre. This distinctive feature sets it apart from the previous approaches, introducing the capability to simulate both convex and concave shapes and smoother curved surfaces, as seen in the diverse objects simulated in figure 2.2.

Shapeshifting devices targeting a more extensive range of objects beyond simple radially symmetrical ones are problematic. The Shapeshift device [33] is a grounded system designed to facilitate adequate perception of the shape of horizontal surfaces and specific vertically flat objects. This is achieved by placing linear actuators that adjust the height of each pin on a horizontal surface, as seen in figure 2.3a. PoCoPo [38], on the other hand, prioritises portability and features

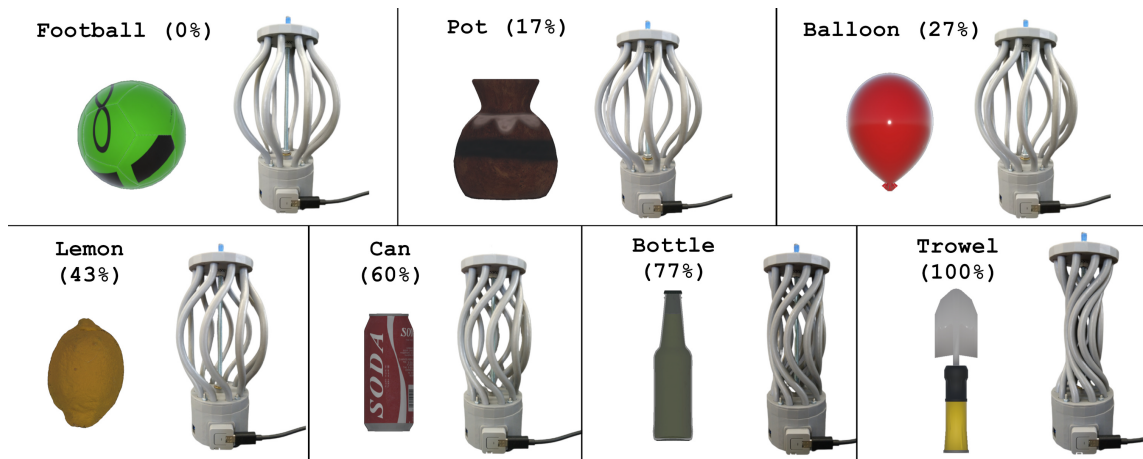


Figure 2.2: Several objects simulated by Shape-A-Getti [4].

two controllable opposing surfaces, as seen in figure 2.3b. This design enables more curious 3D shape definitions, such as cylinders and spheres, to overcome the constraints encountered with Shapeshift.

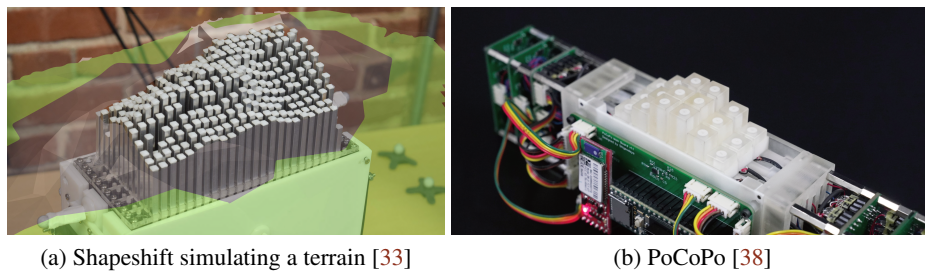


Figure 2.3: Comparison between Shapeshift and PoCoPo

Following more effectively the idea of primitive shape construction, PuPoP [35] is a lightweight and graspable device that is composed of several inflatable airbags that provide various shape primitives, such as cylinders, spheres and cubes stacked together. Figure 2.4 showcases that the device dynamically activates or reactivates different primitives through pressure control in each airbag, allowing the easy creation of distinct composite shapes. This device emphasises the importance of stimulating the whole hand. It recognises that users actively sense the environment with every part of their hand, setting it apart from other approaches that may overlook this property.

A similar approach is followed by InflatableBots [17], where a moving robotic device transports an inflatable airbag. This bag can change its vertical height to adapt to different scenarios by adjusting the amount of air inside, as shown in Figure 2.5. This type of device is particularly useful for simulating large-scale objects and environments.

To enhance immersion in the perception of shape or touch, a straightforward method involves using passive haptic props. WeATaViX [12] introduces a wearable device equipped with a passive prop positioned under the hand. When the user's hand comes into contact with a virtual object, the device elevates the prop, allowing the user to grasp it physically, as seen in 2.6b. Another

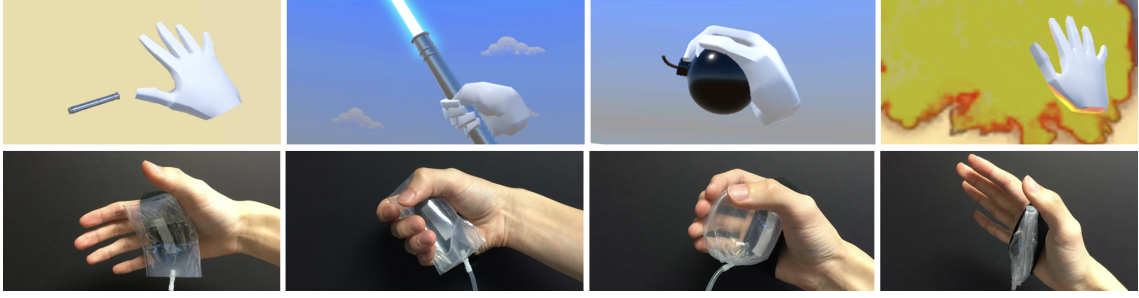


Figure 2.4: PuPoP simulating several shapes with different primitives [35].

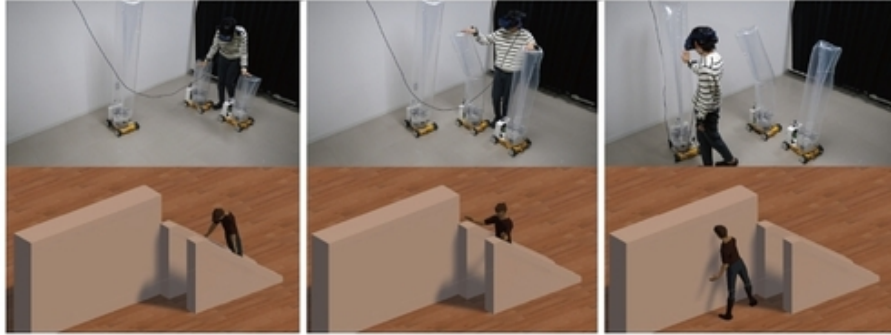


Figure 2.5: InflatableBots changing to adapt to a 3D scenario [17].

approach explored in Tangiball [6] takes a reverse perspective on interaction. This method involves mimicking the position of a haptic prop in the real environment to the virtual one, as shown in Figure 2.6c, essentially serving as a controller for manipulating virtual objects. Another potential approach within the domain of passive props, explored in Haptobend [29], involves using a prop that can be bent into various shapes. However, a significant problem arises in this method, as users need to manually shape the device into the correct form manually, as we can see in Figure 2.6a.

However, these approaches face notable constraints, evident in the restrictions on the range of interactions they can provide. To address these challenges, some solutions opt for using mobile robots. These robots navigate the environment, offering users more meaningful and diverse interactions, thereby mitigating the limitations of stationary designs. The prevalent solution commonly employed involves the use of quadcopters. The authors of Tactile Drones [22] present a novel approach where users receive feedback by interacting with the environment through a drone. This drone actively engages with users, offering tactile interactions with elements such as arrows, wildlife, and other virtual objects [22]. This approach is constrained in its capability, as it can only deliver impact feedback, limiting the potential for more meaningful interactions.

VRHapticDrones addresses this challenge [20], which explores diverse forms of interaction. It provides impact feedback and extends its functionality to control the virtual environment. Users can manipulate the drone's position and control objects within the virtual space, receiving instantaneous feedback from this interaction.

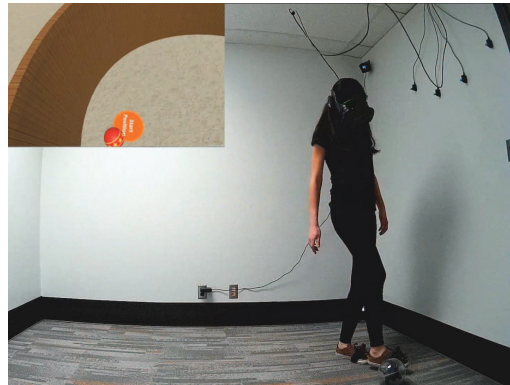
An interesting approach shown in figure 2.7 that addresses the same issue involves using drones to manipulate haptic props for simulating different virtual objects, as explored in Beyond



(a) Haptobend [29]



(b) WeATaViX [12]



(c) Tangiball [6]

Figure 2.6: Comparison between Haptobend, WeATaViX and Tangiball.

the Force [1]. This method allows for replicating virtual objects using corresponding real props and facilitates the generation of distinct texture sensations in the virtual environment.

The previously mentioned strategies lack the diversity of shapes that can be created. An approach studied the possibility of using tiny robots to dynamically assemble magnetic blocks [42] into a passive prop. As seen in figure 2.8, the system comprises several cubes and other shapes that improve the diversity of available surfaces. Every piece can magnetically attach to each other, creating a great variety of 3D shapes. Despite offering a wide range of possible builds, the drawback of this idea lies in the long construction time, posing challenges for simulating rapidly changing virtual environments.

Approaches focusing on the sense of grasping have been studied to tackle the lack of ability to create shapeshifting devices that can virtually simulate any shape by focusing on controlling and restraining the possible movements of the hand, forcing a sense of grasping into the users. The Wolverine [9] is a lightweight device that enhances user immersion by simulating the grasping of virtual objects by applying braking forces between the thumb and the other fingers. This technique, called pad opposition property, is similarly explored in the Rutgers Masters II [5] and DESR [41].

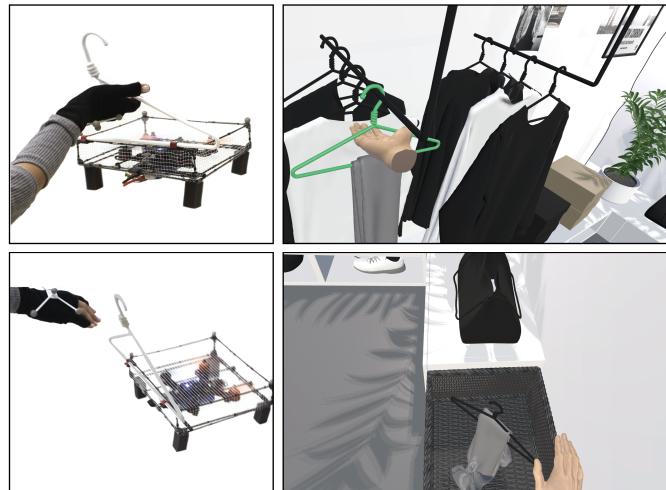


Figure 2.7: Haptic prop appropriation using a quadcopter [1].

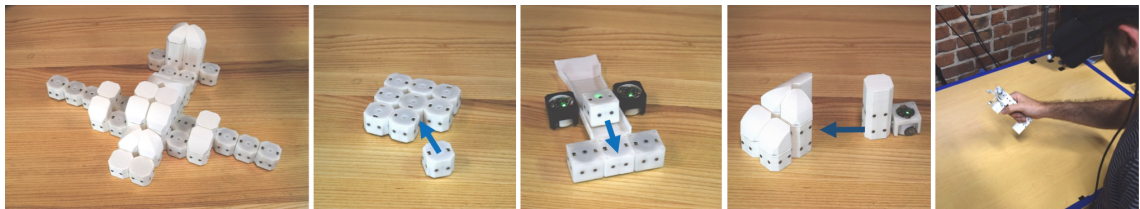


Figure 2.8: Assembly process of a passive haptic prop [42].

The 2DOF and 3DOF joints the device comprises are crucial for enabling users to control their hand movements. This freedom also allows the placement of fingers in various positions, allowing users to perceive the shape of virtual objects at different points. This technique makes it possible to simulate very different circular, rectangular and triangular shapes, as shown in some use cases in figure 2.9.

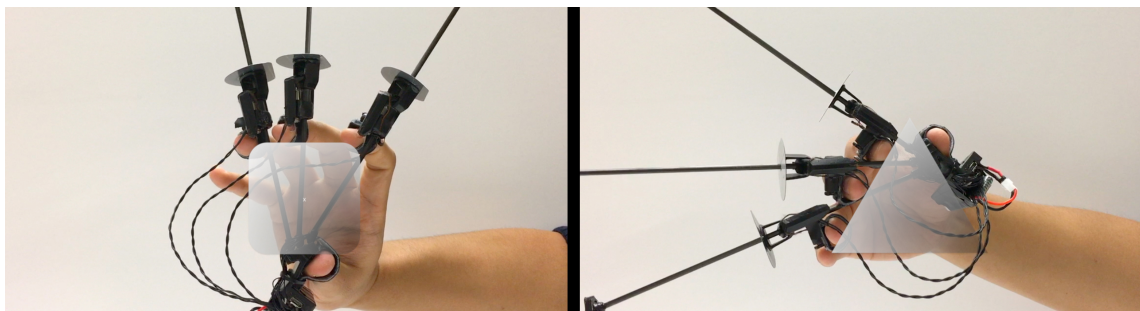


Figure 2.9: Complex shapes simulation with Wolverine [9].

Other approaches explore similar ways of exploring the pad opposition method, such as the Grability [8]. This device comprises opposed pads that prevent the user from closing their hand further, according to the virtual object shape. Similarly, Claw [10] is a handheld device that creates an opposition force between the index finger and the palm of the hand. As seen in figure 2.10, these devices have a lower resolution in the shape definition they can achieve, but they

explore other properties that will later be analysed.

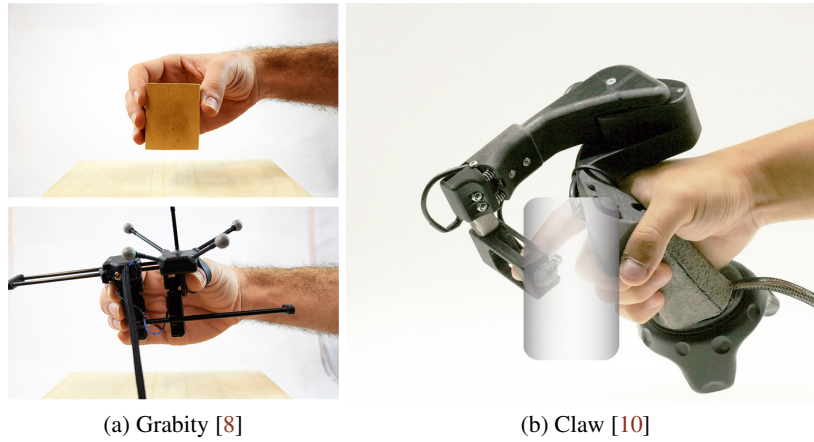


Figure 2.10: Comparison between Grabity and Claw simulating a cube.

The Wireality [14] device explores a different approach to how we can simulate interactions focusing on the whole hand. Instead, in this approach, wires are attached from a controller placed on the shoulder of the user connected to various points on the hand, including the fingers, palm, and wrist, as showcased in figure 2.11. The controller actively regulates the length of these wires, consequently limiting how much a user can move each part of their hand. This strategy facilitates the simulation of complex shapes by delivering feedback across several hand points.



Figure 2.11: Wireality simulating a complex shape [14].

2.1.2 Texture

While shape and grasping are fundamental aspects of creating immersive experiences, it is essential to explore other properties. An object's texture, while offering more subtle haptic information when compared to its shape, still serves as a potent haptic cue [32]. According to studies, texture recognition in humans is associated with the perception of light vibrations caused by an object's movement interaction and the user's skin [32]. Claw [10] explores this possibility using a voice coil actuator that generates vibrations. Whenever the user moves his finger across a virtual surface,

the device calculates the speed at which the movement is occurring and plays back a randomly selected sample from a vibration haptic pattern, modifying the speed and amplitude according to how much pressure the user is applying and to the general roughness of the surface. This solution can dynamically update the texture properties and is a base for simulating various textures.

High precision is needed to simulate interaction with textures by stimulating the user with vibrations. To solve this problem at the cost of loss of variety simulated, approaches to dynamically changing a physical surface have been researched.

As showcased in figure 2.12, the Snake Charmer [3] is a device that explores this approach. Consisting of a movable mechanical arm and an attachable cube featuring various surfaces, the device dynamically adjusts its setting based on the object the user is interacting with, rotating the cube and translating it to the position of the user's hand. This device focuses on adaptability and versatility as it can be attached to other objects with different shapes or textures. However, this device has drawbacks, as it is bulky and grounded, which might bring some difficulties and restrictions to virtual experiences.



Figure 2.12: Snake Charmer changing the simulated surface [3].

The concept behind graspable devices aims to address the earlier challenges, specifically those associated with the bulkiness and intrusiveness of specific existing devices. Materializer [31] similarly approaches the problem as Snake Charmer[3] while focusing on the device's portability. The device is structured by five identical triangular prisms that rotate around a vertical axis, as seen in Figure 2.13. Each face of these prisms accommodates three distinct surfaces that simulate different textures. The device then rotates the prisms whenever the interaction requires a different texture. This approach creates a more portable device but encounters limitations when considering the simultaneous simulation of textures, as it is restricted to accommodating only three textures at a time. Besides this, it doesn't provide a simple method for interacting with the surface, requiring users to hold the device while attempting to engage with it.

Haptic Revolver [37] is a handheld device that provides a similar modular approach with a distinct emphasis on interaction. Users place their index finger within an attachable circular base, rotating perpendicularly to the direction of the user's finger. Figure 2.14 showcases the development of this device with modularity in mind. The idea behind this is to simulate different objects with different properties. Unlike the prior approach presented by the Materializer [31], which lacked the means to interact with the surface easily, this device synchronises the rotation speed of the circular base with the user's finger movement in the virtual environment. This improvement in interaction is contrasted with the number of surfaces the device can simulate simultaneously.

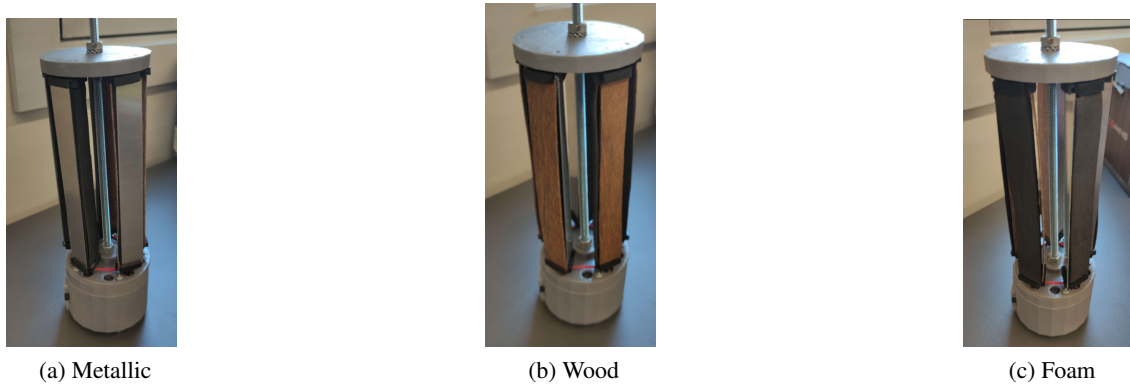


Figure 2.13: Materializer showcasing three different textures [31].

However, this challenge could be addressed by actuating the wheel axis forward and backwards, automatically switching between multiple wheels installed on the device [37].



Figure 2.14: Modularity of the Haptic Revolver [37].

2.1.3 Muscle stimulation

A critical aspect of exploring our surroundings is the interaction we establish with the objects in our environment. While the previously mentioned properties play a crucial role in creating realistic sensations, how we receive feedback from our engagements with the virtual world adds an additional layer of immersion.

Accurately simulating feedback for impact and resistance forces poses a complex challenge, often necessary to use externally stationary devices to apply force from an external source [25]. Some previously proposed solutions successfully evoke the presence of an object but fall short in providing accurate feedback when the user exerts force in unintended ways, primarily due to the absence of external resistance forces [26]. This limitation may lead to interactions that are not intended within the virtual environment.

Impacto [25] addresses the challenge of impact simulation and explores simulating impact forces without using external tethered devices but, instead, by manipulating the movement of users' limbs. This is achieved by electrically stimulating muscles responsible for limb movement, enabling the creation of a backward movement, thus simulating impact, as shown in figure 2.15.

The effectiveness of this approach was tested in various use cases and body parts, with a primary focus on simulating a fight in the arms and mimicking the kick of a ball on the foot.

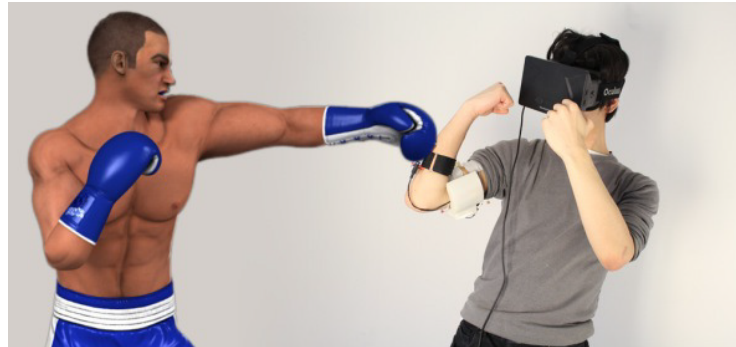


Figure 2.15: Impact simulation using electrical muscle stimulation [25].

In a later approach, the same principle of electrical stimulation addresses the significant challenge of restricting physical interactions by limiting the user's movement without relying on external forces. The authors of [26] built a device that influences the user's shoulder, arm, and wrist using electrodes to stimulate the muscles, inducing a counteractive push, as shown in figure 2.16. This design allows for generating either solid or subtle resistance forces, effectively simulating powerful repulsive forces or simple constraints similar to object surfaces that prevent further movement forward. As a result, interactions become more realistic and accurate in how users engage with stationary objects.

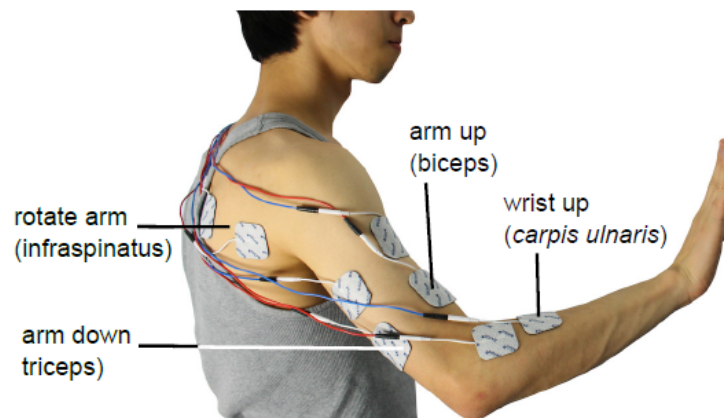


Figure 2.16: Muscle movement manipulation using electrical stimulation [26].

2.1.4 Gravity and External Forces

Some intriguing aspects that come to mind during interactions with objects in the environment are their weight, which is influenced by gravity, and the impact of external forces like friction and drag when moving them around. While these properties might go unnoticed for lighter objects, they enhance the sense of immersion. This is a particularly challenging problem in non-grounded devices because alternatives to simulate them must be found.

One potential method to apply external forces without being grounded is using propellers, commonly found in various vehicles. Thor’s Hammer [19] explores this concept by incorporating a three-degrees-of-freedom force feedback system featuring a handheld cage with six propellers, as shown in Figure 2.17. This setup enables the device to simulate diverse forces, including water and air drag, pulling sensations, resisting forces like touching, and gravitational forces when manipulating objects. While this solution yields compelling results, it does introduce a notable drawback — increased noise levels when the propellers operate at higher force levels. A comparable approach was explored by ProppellerHand [2], which, in contrast to Thor’s Hammer, is a wearable device.

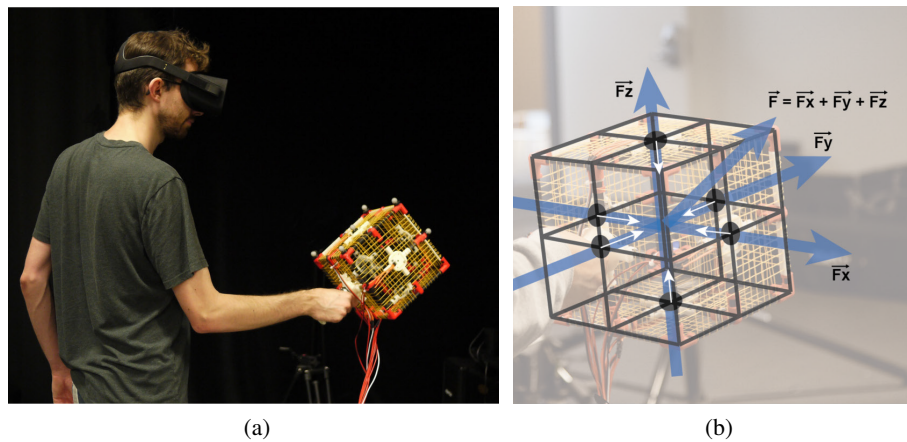


Figure 2.17: Thor’s Hammer [19].

Shifty [39], as seen in Figure 2.18, is a graspable device shaped like a rod featuring a movable weight piece along its axis. This device capitalises on a fundamental physical property to simulate weight perception. When interacting with an object, manipulating the centre of mass creates a distinct perception. As the centre of mass moves farther away from a person’s hand, which essentially serves as the object’s rotational axis, the moment of inertia increases, making it more challenging to move. Consequently, users must exert greater force to manipulate it, fostering the illusion of different weights. The device addresses the challenge of gravity by leveraging this physical phenomenon. While the device’s actual weight remains constant, altering the position of the weight will change its centre of mass, generating a perceived variation in weight without the use of any external force.

Drag:on [40] extends the exploration of weight shifting and integrates it with air drag-resisting forces. The device can adjust its surface area on demand by conveying two expandable fans, as depicted in Figure 2.19. In addition to altering the device’s centre of gravity, as previously explored by Shifty [39], this adaptation also changes the drag-resisting forces from the air when the object is in motion. This enhancement contributes to a more immersive and realistic sense of weight and external forces during object manipulation.

Gravity is also addressed through an exploration of asymmetric skin stretching. Asymmetric vibrations, distinguished by large positive acceleration peaks and small negative acceleration



Figure 2.18: Shifty graspable haptic device [39].

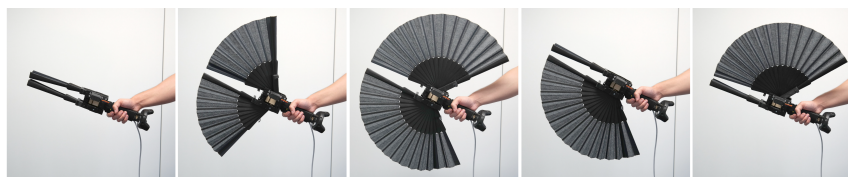


Figure 2.19: The several possible states of the Drag:on device [40].

peaks, create a compelling sensation of being pulled toward the larger acceleration [11]. This phenomenon serves as the foundation for WAVES [11], a wearable device that offers users three-dimensional translation cues, facilitating natural guidance. This approach is further employed by Gravity [8] to convey a sense of gravity, as it supplies asymmetric skin stretching in the fingertips, eliciting a pulling sensation aligned with the direction of gravity.

Triggermuscle [34] explores a less accurate approach but is very easily applicable since it is a simple extension of the standard VR handheld controller. It adds a spring mechanism inside the trigger of an HTC Vive controller whose resistive force changes according to the weight of a virtual object. This way, in order to move virtual objects around, they must exert more considerable force the heavier these objects are. This more straightforward approach, however, did not yield optimal results, as many users failed to perceive distinct weights using the new method.

2.1.5 Stiffness

In the realm of haptic devices, while the emphasis often lies on shape and external forces, the consideration of stiffness becomes particularly significant in certain situations. Although it may not take precedence in every scenario, stiffness is a crucial and robust haptic cue, contributing to the overall tactile experience in specific contexts.

The CLAW multifunctional device [10], in addition to offering previously mentioned properties, can provide haptic cues related to stiffness. Functioning as a controller for the virtual hand, it detects the force exerted by the user and, based on this information, provides a force feedback response between the thumb and index finger tailored to the properties of the simulated object. This capability empowers the device to calculate and counteract the user's force, facilitating the

nuanced simulation of various stiffness properties. Bstick [23] adopts a comparable approach, assigning each finger to a linear actuator that can alter its position for simulating multiple shapes, as shown in Figure 2.20. This device can also read the force exerted by the user on the device, enabling it to offer force feedback and evoke a sense of stiffness.



Figure 2.20: Bstick [23].

2.1.6 Temperature

When an individual interacts with an object and encounters a significant temperature difference, it becomes an intriguing property to explore as a potent haptic cue. The sense of touch studied in previous sections can be highly stimulated by also incorporating temperature since human skin is very sensitive to temperature variations.

One of the simplest and more straightforward approaches is using fluids to transport, manipulate and transfer energy to and from the user. Two devices that explore this approach are the ThermAirGlove [7] and the PATCH [16], which rely on the transport of fluids of different temperatures to transfer heat between the user and the device, to convey a sensation of difference of temperature. Their main difference is in the fluid conducted through the pipes, which in ThermAirGlove is air, and in PATCH is water. Besides this, the devices also vary in format, where ThermAirGlove provides a more user-friendly solution in the format of a glove instead of an armband, as seen in Figure 2.21.

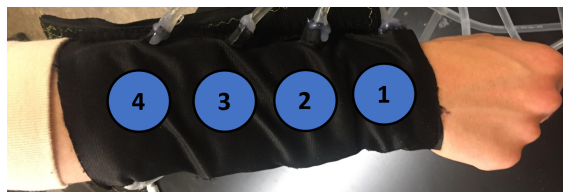


Figure 2.21: PATCH and its different contact points of temperature [16].

An alternative approach leverages two physical phenomena: the Joule effect, which generates heat by passing current through electrical resistances, and the Peltier effect, where charge carriers move within thermoelectric pellets in response to the current direction [24]. This device enables heating and cooling sensations through an electrical circuit that adaptively changes its current direction to influence the electrical system, as seen in Figure 2.22. Comprising two layers with distinct electron types, the device can switch temperatures by reversing the current direction,

prompting electrons to relocate to the alternate layer. This method proves notably more straightforward than solutions involving fluids, eliminating the need for fluid storage and bulky pumping systems.

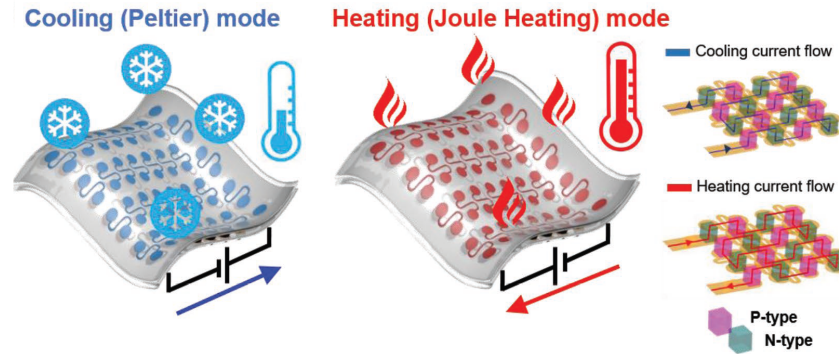


Figure 2.22: Effect produced by alternating the current of the device [24].

These approaches are standard and adapted from systems outside virtual reality and haptic devices. The paper entitled Chemical Haptics [27] explores a novel approach using skin-safe chemicals to convey sensations to the users, as shown in Figure 2.23. Instead of using electrical or mechanical systems, these sensations are induced when the skin reacts to specific chemicals, generating haptic feedback. In addition to examining sensations like stinging and numbness, the paper also investigates the simulation of temperature changes using chemicals, using capsaicin to create warming sensations and menthol to produce cooling effects.



Figure 2.23: Process of providing chemicals to simulate different sensations [27].

2.2 Grounded devices

Grounded devices, which establish a physical connection with external elements such as the ground or walls, can significantly exert external forces on users. This characteristic enables the accurate simulation of environmental forces, contributing to a more nuanced and realistic haptic experience.

One of the first approaches studied in the realm of haptics was the Phantom device [28], which is able to provide high precision forces on the fingertip to simulate virtual environments. A later

approach extends this behaviour by creating HIRO III [13], as seen in figure 2.24, a 15 degrees-of-freedom device that provides an actuated force on the five fingers of the user. Since both of these devices are grounded, they are highly capable of showcasing high performance and accuracy in force display.



Figure 2.24: Hiro-III haptic device [13].

These types of devices may also accurately simulate the movement of specific objects within a virtual space, thanks to their steady bases. An example is the Snake Charmer [3], which can effortlessly navigate around obstacles and adapt to different surfaces, as previously mentioned.

Besides these exciting properties, grounded devices benefit from fewer spatial constraints and can simulate larger objects with greater precision, focusing more on accuracy and less on space conservation. This distinction is evident when comparing Shapeshift [33] and PoCoPo [38]. Shapeshift's expansive design can simulate more complex shapes and exhibit significant height variations. In contrast, PoCoPo, constrained by its smaller size, operates within more confined parameters, limiting its ability to replicate smaller objects.

Despite these advantages, there are some drawbacks. The direct connection of these devices to the real environment and their potential bulkiness can make these devices invasive. Thus, developing these devices with user safety in mind is crucial.

2.3 Portable devices

To tackle the challenges grounded devices encounter, research has increasingly concentrated on developing smaller and more convenient alternatives. The primary solutions designed revolve around two device paradigms: wearable devices and graspable devices.

2.3.1 Wearables devices

The primary limitation of portable devices in comparison to grounded ones is the absence of externally applied forces, making the restriction of limb movement a challenging task. In order to

address these problems, several approaches in the realm of wearable devices have been studied.

A practical approach involves restricting movement along specific axes to address the limitation arising from the absence of external forces. This concept relies on applying forces between different body parts, providing a means to simulate grasping and representing varying shapes by exerting forces between the palm of the hand and the fingers. Notable examples showcasing this idea include the Wolverine [9], Grabity [8], and Claw [10]. In a similar vein, Wireality [14] applies force between the user's back and various points on the hand to constrain hand movement. When comparing them to grounded haptic devices, a noticeable decline in accuracy and a reduction in the number of simulated degrees of freedom become evident due to the absence of external forces. However, due to their compact size, these portable devices showcase increased versatility and user-friendly controls.

Another proposed solution involves using variable fluid containers in the user's hand to restrict the extent of grasping, thereby conveying a sense of shape and constraint. The PuPop device [35] explores this concept by employing variable air-pumped shape primitives. While this approach is distinct, it falls short in terms of the effectiveness provided by other methods.

A final and intriguing approach investigated by wearable devices involves muscle stimulation through electrical current. This method was explored in Impacto [25] and in a subsequent study in [26], as previously described in subsection 2.1.3.

Despite the existence of intriguing solutions within this paradigm, some drawbacks persist. The primary concern is that wearables may induce significant discomfort due to their continuous contact with the user. A notable example is the Wireality device [14], which is restrained by its considerable bulkiness despite being a portable solution.

2.3.2 Graspable devices

Graspable devices emerge as a solution to address various challenges the previously presented alternatives pose. Compared to wearable devices, graspable devices offer a less invasive approach by mirroring the natural way we interact with objects – by picking them up. Unlike wearables that are already in hand, graspable devices prioritise a more intuitive and familiar interaction, enabling users to engage with technology in a way that echoes their daily interactions with physical objects.

A notable distinction arises in comparing the wearable device X-Rings [18] with the graspable Shape-A-Getti, both designed for radially symmetric shapes. While X-Rings demands constant usage and incorporates intricate mechanisms with radially expandable layers and motors, Shape-A-Getti [4] sets itself apart by prioritising a more user-friendly experience. Shape-A-Getti simplifies interaction as it makes it more natural, mimicking how users pick up objects in the real world. This pivotal difference mitigates the complexity of continuous usage and significantly improves overall accessibility and convenience when engaging with simulated shapes.

Numerous instances reveal the application of similar properties and methodologies on a smaller scale when contrasting grounded devices with graspable counterparts. As previously explained, the PoCoPo [38], for instance, can be viewed as a scaled-down version of Shapeshift [33]. At the expense of a reduced simulated area and available precision, which was seen as a possible

advantage of Shapeshift, PoCoPo offers enhanced usability by eliminating the need to manipulate a weighty object.

2.4 Discussion

Table 2.1 presents a comprehensive analysis of haptic devices, grouped into four main categories: Graspable, Grounded, Wearable, and Movable. Grounded devices establish a physical connection with external elements, enabling precise simulations of environmental forces. Wearable devices, characterised by their lightweight design but requiring continuous user engagement, offer a less intrusive option for haptic interactions. Graspable devices, known for their portability and versatility, allow users to engage by picking them up and putting them down. Movable devices, not directly handled by the user but mobile, extend the scope of haptic interactions. With this classification in mind, we can convey valuable insights into the diverse capabilities of state-of-the-art haptic technologies. The following columns identify if the devices in question target the properties of shape and grasping, texture, mimicking the positioning of virtual objects, temperature, stiffness, and lastly external forces and gravity.

This analysis reveals the absence of perfect solutions in the current landscape of haptic devices for virtual reality. A notable example is evident in the comparison between Shapeshift and PoCoPo. While Shapeshift offers higher precision and the capability to simulate more complex shapes, its bulkiness and spatial constraints limit its usability. This example underscores the ongoing challenge of balancing precision, usability, and the devices' size.

Considering this trade-off, there is still a noticeable preference for non-grounded devices, which emerges upon closer examination of available solutions. This preference is reinforced by the substantial number of devices developed within this category, indicating a marked shift away from conventional grounded solutions. While grounded devices excel in accurately simulating properties like external forces, the change towards non-grounded alternatives underscores a strategic move to overcome limitations associated with spatial constraints and setup challenges. This strategic shift signifies a pursuit of solutions that balance accuracy with improved user experience.

Another evident challenge this analysis highlights is the absence of devices capable of fully simulating all the necessary elements for a truly immersive experience. Notably, only a few devices manage to simulate more than one property, leaving a gap in achieving comprehensive haptic feedback. The current landscape of haptic devices for virtual reality presents a situation where devices offer mere approximations, failing to address the comprehensive requirements for a fully immersive encounter. This underscores a critical, unaddressed issue in the realm of haptic technologies for virtual reality, encouraging further research and innovation to bridge the existing gaps and deliver more cohesive and realistic virtual experiences.

Mobile devices, as demonstrated by solutions using drones [20, 22, 1] and robotic assembly of shapes [42], provide exciting solutions to challenges found in both grounded and portable devices. They harness the advantages of each category, utilising the movement capabilities of grounded devices to convey external forces and motions while capitalising on the scale and user-friendliness

of portable devices. This collaborative approach highlights the potential for innovative solutions by integrating features from both device types.

Graspable devices exhibit compelling properties that closely mimic the natural way humans interact with physical objects, providing an intuitive and familiar user experience. Compared to grounded devices, their reduced size enhances their portability and convenience, allowing users to engage with haptic feedback more easily. Additionally, graspable devices offer a better user experience by minimising invasiveness when compared to wearables, as they do not require continuous contact with the user's body. However, a notable challenge arises from these devices needing to be manually picked up and placed in the desired location, limiting the range and spontaneity of possible interactions.

With this in mind, while graspable devices present remarkable advancements in mimicking natural human interactions and enhancing user experiences, the inherent challenge of manual repositioning poses a significant limitation. We recognise this as one of the issues in the current literature and that it can be addressed by drawing inspiration from movable devices that can offer innovative solutions. Exploring the realm of movable devices provides a promising avenue for addressing the limitations associated with the stationary nature of graspable devices, thereby paving the way for more dynamic and immersive virtual experiences.

Device	Shape	Texture	Positioning	Temperature	Stiffness	Forces
Grounded Devices						
Phantom [28]						•
HIRO III [13]						•
Shapeshift [33]	•					
Snake Charmer [3]	•	•	•			
PATCH [16]				•		
Movable Devices						
Tactile Drones [22]			•			
VRHapticDrones [20]			•			
InflatableBots [17]	•		•			
Beyond the Force [1]			•			
Robotic Assembly [42]	•		•			
Tangiball [6]	•		•			
Wearable Devices						
PropellerHand [2]						•
Impacto [25]						•
Triggermuscle [34]						•
Muscle Stimulation [26]	•					•
Gravity [8]	•					•
Claw [10]	•				•	
Wolverine [9]	•					
Wireality [14]	•					
PuPoP [35]	•					
WeATaViX [12]	•					
X-Rings [18]	•					
ThermAirGlove [7]				•		
Stretchable Device [24]				•		
Chemical Haptics [27]				•		
Graspable Devices						
Shape-A-Getti [4]	•					
PoCoPo [38]	•					
Haptobend [29]	•					
Bstick [23]	•				•	
Haptic Revolver [37]		•				
Materializer [31]		•				
Shifty [39]						•
Drag:on [40]						•
Thor's Hammer [19]						•

Table 2.1: Overview of Haptic Devices

Chapter 3

Move your Haptic

This chapter primarily focuses on pinpointing the problem chosen for investigation following an evaluation of the state-of-the-art. To this end, we articulate a more concrete problem, hypothesis, and research questions. Furthermore, we propose a possible solution to address this challenge. Finally, we will review the implementation details and issues we encountered while developing a proof of concept.

3.1 Problem statement and hypothesis

Upon conducting a comprehensive analysis of the current state-of-the-art and available solutions, we have identified specific areas that present opportunities for further development. Graspable devices provide a more natural way of interacting with devices, as users can pick them up and put them down, as opposed to wearable devices. However, these devices must be manually relocated to the virtual object's target position. We recognise the powerful interactions a graspable device can provide, but we acknowledge their limitations in terms of mobility, which constrains the breadth of possible interactions, mainly dynamic ones. In a pursuit to advance research in this domain, our primary objective is to create a solution that can add the dimension of mobility to existing graspable devices. Considering the identified challenge, the hypothesis that we will be trying to test and evaluate is defined as follows:

It is possible to simulate multiple virtual objects in a convincing way using a smaller number of movable devices.

Several challenges must be overcome to address the hypothesis. Two primary issues stand out in the idealisation of a device that targets this hypothesis. The first challenge involves accurately predicting which object the user intends to interact with so that the devices understand which virtual object to replicate. The second challenge lies in accurately tracking the device's position and moving it to the target position, replicating the virtual object's location. Both of these issues pose substantial challenges, but we believe in prioritising the resolution of the challenge of accurate positioning and movement, given its central role in our device's functionality. To navigate this

challenge, we propose abstracting and simplifying the recognition process to a straightforward and naive approach.

To more clearly understand how we can evaluate our hypothesis, we define the research questions as the following:

RQ1 Is it possible to simulate a diverse range of virtual objects using a reduced number of mobile devices in a way that remains imperceptible to the user?

RQ2 What are the limitations on the capabilities of the mobile device that allow it to still precisely replicate the positioning of virtual objects in a timely manner?

3.1.1 Our approach

Inspired by haptic devices that have incorporated mobility solutions, such as shape assembly using robots [42] and drone-based solutions [20, 22, 1], we conceived the idea of developing a mobile base capable of transporting graspable devices or haptic props. This solution, as it is not moved in the air, provides a more precise and controllable way of moving objects around. To address the hypothesis at hand, our idea is to control a swarm of movable devices that control the position of multiple graspable devices.

The device's overall design uses two parallel bases held together by vertical rods. In this approach, the lower layer supports the wheels and houses various solution components, including the microcontroller, batteries, and cable management. The upper layer is dedicated to securing the haptic gadget in place. One possible way to achieve this is to establish two circular layers separated by springs, generating a centring force at the edges of the haptic apparatus's base. This problem, however, will be a secondary focus in the initial development of the prototype and will be reevaluated later when trying to integrate a use-case haptic gadget.

A simple design for addressing the mobility problem involves employing independent wheels on each side of the vehicle. This configuration enables the vehicle to move forward, backwards, and rotate, allowing it to attain any position within the environment. However, such vehicles introduce additional complexity when aiming for precise positioning. Determining the initial rotation needed to reach the correct direction accurately becomes crucial. If the device requires adjustments upon nearing the target, it has to execute a rotation before proceeding forward. The complexity of the movements necessary to correct errors and the substantial delay incurred during the rotation process may render this solution impractical for the high-speed requirements of human interactions.

A microcontroller is essential to control the wheels. This component is responsible for translating the required movements of the device into the torques to be applied to the wheels and for actuating them accordingly. After implementing the fundamental movement functionalities, our next step involves comprehensive testing of critical characteristics. These tests will assess parameters such as weight transport capacity, speed, and the precision with which we can reach desired positions. The goal is to gather valuable data that will enable us to fine-tune and optimise our device. If needed, we will reevaluate our approach and consider alternative solutions.

Another crucial challenge to address is ensuring the accurate alignment of virtual objects with the natural environment, which is the primary focus of our work. Initially, we must establish a method for mapping the position of the virtual device to the real world and collecting information on the user's hand position. To define the target position, we must analyse which virtual object the user is moving towards to place the device in that position. Once we successfully track the correct position of our device, we can initiate an iterative process to calculate the direction required to move the device to the target position. In each iteration, we validate whether the target position is met and readjust the movement direction if necessary. For clarity, we sketched this current idea of the prototype, as seen in Figure 3.1.

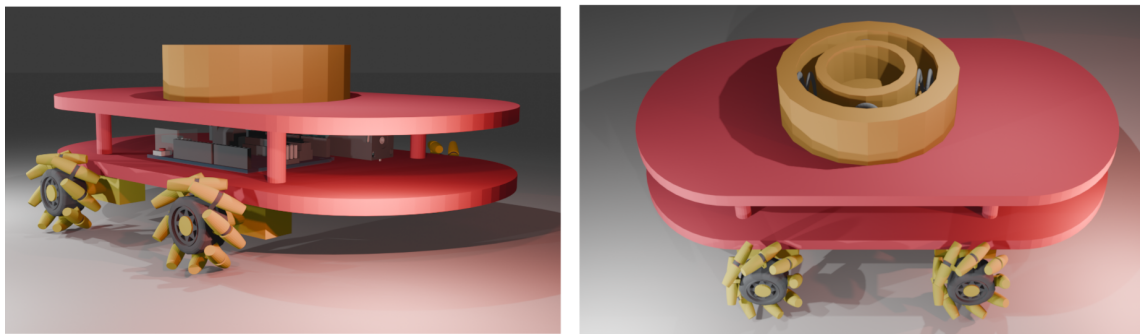


Figure 3.1: Sketch of the idealized solution

3.2 Device

The device developed in this research is designed to interact with virtual environments through precise movement and control. We aimed to create a device that performs reliably in various conditions and integrates easily with existing experiences to provide a robust user and developer experience. The final implementation of the device in action, moving a haptic prop, can be seen in Figure 3.2.

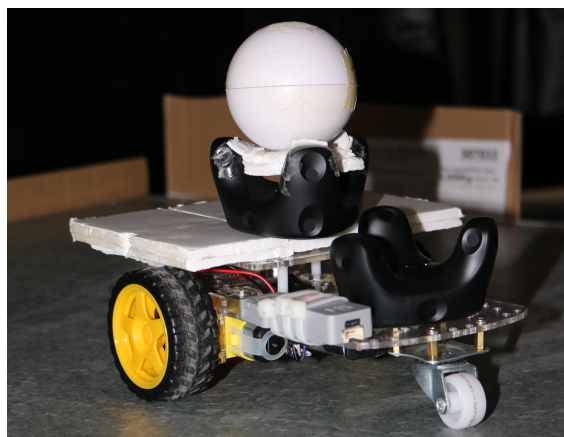


Figure 3.2: Final implementation of the device transporting a haptic prop

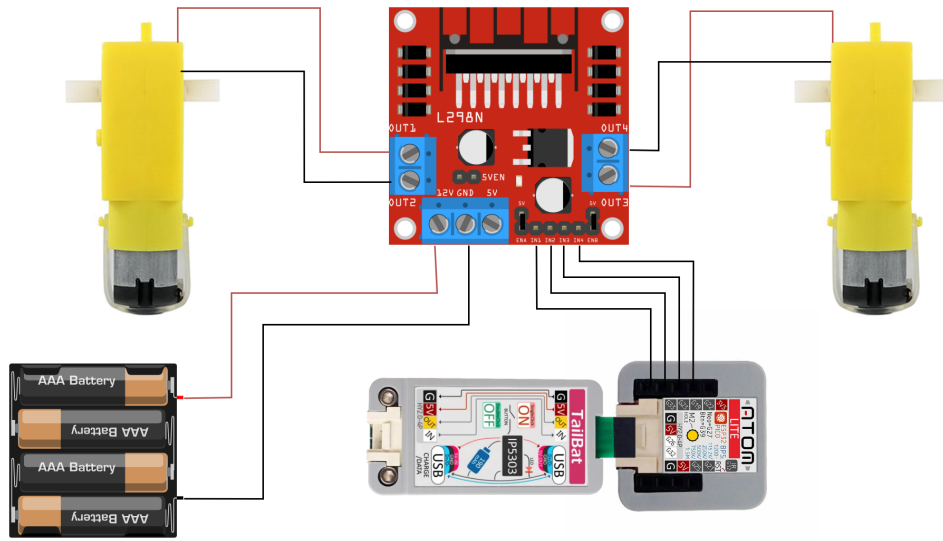


Figure 3.3: Device hardware schematic

3.2.1 Hardware

The device hardware schematic, shown in Figure 3.3, illustrates the components used in the physical device. The device comprises several key components, including a motor driver (LN298N) that controls the motors with attached wheels, and a free wheel, batteries and microcontroller.

This composition forms a differential drive vehicle, which consists of two active wheels and an additional, yet optional, free-rotating wheel, similar to our design. Such vehicles are subject to a non-holonomic constraint, meaning that while they can reach any position and orientation within their configuration space, they are limited in their movement capabilities. Specifically, these vehicles cannot move sideways and can only change direction by rotating.

The motor driver allows for easy voltage and current direction adjustment in each motor, enabling control over movement and speed by combining different signals. The driver can control up to two motors. It contains two pins, *ENA* and *ENB*, to control the speed of each motor. However, in this implementation, we did not use these pins and left them connected to the 5-volt pin to achieve maximum speed.

Our device also includes a pack of AAA batteries that power both the motors and the motor driver. We use an M5 Stack Atom Lite microcontroller powered by a TailBait, a rechargeable battery, to control these components. The motor driver is controlled by connecting four GPIO inputs from the microcontroller to the driver's *IN* connections. When active, these inputs apply the necessary current direction and voltage to the motors to achieve the desired wheel movement direction.

In order to save space on top of the device, so that we can put the HTC Vive tracker, which is a tracking device, and the base for the haptic prop, we decided to place most of the electronic components under the acrylic base, as seen in Figure 3.4. This didn't affect the wheels or the

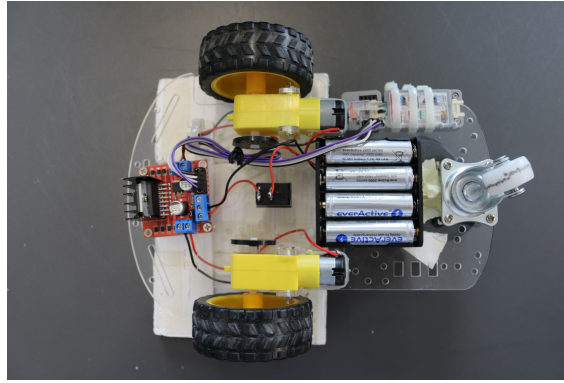


Figure 3.4: Components organization of the device

movement as the cables were well attached, tight and organized.

As illustrated in Figure 3.5, we can control the movement of the device by individually manipulating the current in the motor driver pins IN1, IN2, IN3, and IN4. The principle is to control the direction of rotation for each wheel. The two basic movements are forward movement and right rotation.

For forward movement, both wheels need to rotate in the same direction. This is achieved by turning on the IN1 and IN3 pins of the motor driver, as shown in Subfigure 3.5a. To rotate the device to the right, the wheels must rotate in opposite directions. This can be accomplished by turning on the IN1 and IN4 pins of the driver, as depicted in Subfigure 3.5c.

For other movements, we simply need to activate the opposite pins for each wheel. For instance, to move backward, we would turn on the IN2 and IN4 pins, and to rotate to the left, we would turn on the IN2 and IN3 pins. These configurations are also illustrated in Subfigures 3.5b and 3.5d.

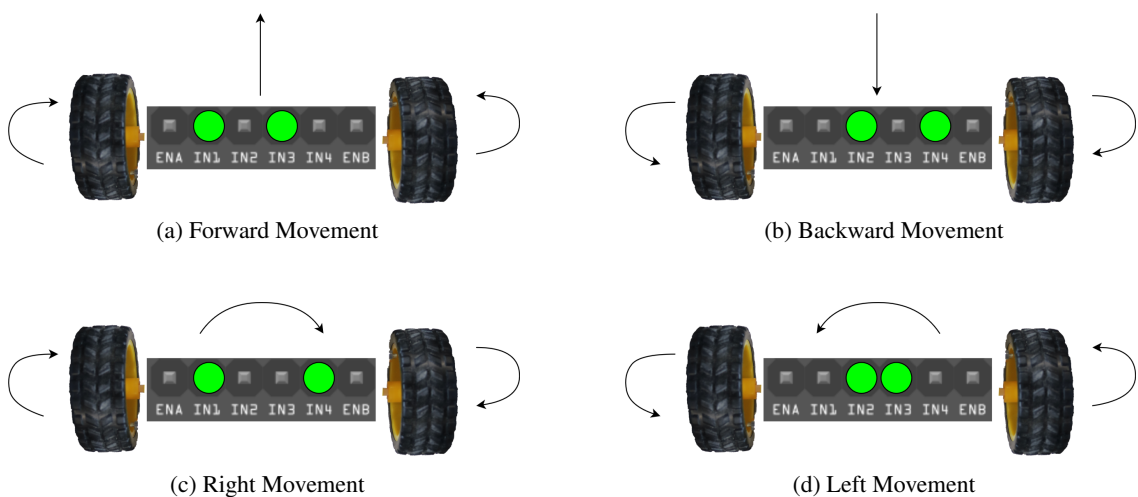


Figure 3.5: How to operate the driver to move the device.

3.2.2 Software

The software supporting this project primarily manages two key functions: operating the motor driver and listening to/reporting the device status via MQTT. Operating the motor driver involves adjusting the output voltage of each GPIO pin. We utilized the Arduino framework with Platform.io as our development environment for programming the microcontroller. The Arduino framework simplifies interaction with GPIO pins on compatible microcontroller boards. Specifically, the `digitalWrite(pinNo, 0/1)` method provided by Arduino allows us to toggle the voltage of output pins, turning them ON or OFF to control the motor driver directly. Additionally, we configure the GPIO pins for output using `pinMode(pinNo, OUTPUT)`. For the first motor, we control the states of GPIO pins connected to IN1 and IN2, and for the second motor, IN3 and IN4 are controlled in a similar manner.

The current implementation relies on MQTT to receive instructions via topics for movement and rotation, subsequently setting the current state to the desired GPIOs. Upon receiving new instructions, the device defers processing until the current operation completes. After each movement, the device signals readiness to accept additional instructions, thereby enhancing efficiency. This approach prevents the device from becoming overwhelmed by a flooded message queue, ensuring optimal message processing speed.

To enable this functionality, it was crucial to determine the linear and rotational velocities of the device. Since the motor driver cannot directly receive movement instructions and the microcontroller lacks speed sensors, the device estimates the necessary duration to keep the desired GPIO pins active to achieve specific movement or rotation values. After measuring the required parameters, as discussed later in Subsection 3.3.2, we can calculate the time required to complete the operation to achieve the desired rotation or distance. Our device operates with an acceleration of 0, allowing us to utilize the basic velocity formulas. Let v denote the calculated velocity, and Δs represent the desired distance to be travelled. Before stopping, the device activates the required GPIO pins for t seconds, calculated as $\Delta t = \frac{\Delta s}{v}$.

Similarly, it is possible to calculate the time the device needs to have the GPIO pins active to rotate after measuring the desired rotational speed. We can assume the simplified formula for angular velocity. Let ω be the measured angular velocity and $\Delta \theta$ be the desired rotation to perform, we can leave the device actuating for Δt seconds, calculated as $\Delta t = \frac{\Delta \theta}{\omega}$.

3.3 Controlling the Device

Interaction with the device is facilitated through the MQTT protocol. Given that the device lacks knowledge of the virtual or physical world's mapping, Unity manages its movement. The device's position is tracked using an HTC Vive Tracker within Unity, enabling the transmission of both the desired target direction and periodic position updates to the device.

Theoretically, the device could calculate the necessary movements to reach a specified destination based on the target direction and position information provided. However, transmitting

such a large volume of data would severely affect the microcontroller's performance, leading to potential operational issues.

To address this challenge, we have decided to transfer the computation of movement actions to the client's software rather than the device itself. By centralizing movement calculations within Unity, we minimize data transmission overhead and ensure smoother operation. This approach leverages the computational capabilities of the client's software to handle complex calculations, thus optimizing the performance and responsiveness of the device.

3.3.1 Remote start and stop

Initially, a straightforward and naive approach was tested. We implemented three basic actions for the device: move, rotate, and stop. This method involved the device continuously performing one of these actions until it received a stop message. However, due to the nature of communication via MQTT and WiFi, there was a significant delay between sending the stop message and processing it by the client.

As a result, when the stop message was received and acted upon, the device had already exceeded the desired rotation or movement value. This delay made it impossible to position the device as intended accurately. The inherent latency in the communication process thus rendered this approach impractical for precise control of the device.

3.3.2 Movement by value

In our subsequent approaches, it was essential to accurately determine the velocities of the device to instruct it on movement and rotation properly. To achieve this, we created a scene in Unity with a user interface designed to help calculate the necessary values.

The procedure began by placing the device in its starting position and initiating the process through the user interface. The initial position was stored for later calculations. We then set the device to move mode, allowing it to move for 5 seconds. The final position was also recorded after clicking the finish measurement button. The calculated distance travelled was then divided by 5 seconds, providing the velocity in Unity units per second. This value was subsequently stored in the microcontroller for precise movement control. Angular velocity was calculated similarly by recording initial and final rotation values, although this was often measured and fine-tuned manually due to being simpler and more effective.

With these velocity values, we could instruct the device on the distances and rotations required to reach its final destination. At regular intervals, such as every 200 milliseconds, Unity would send updated instructions to the device to guide it toward the destination. If the device were already acting on any action, it would ignore subsequent requests until it was done. The device's first step was to align within a specific angle threshold, typically 15 degrees, before receiving the command to move the necessary distance to the target point.

However, this approach had limitations. If the device deviated from the intended path due to the third wheel causing undesired rotation or inaccuracies in the motor driver or the motors,

it could end up far away from the desired position. As illustrated in Figure 3.6, even a small misalignment could result in significant deviation. Repeated occurrences of such deviations could cause the device to take a long time to realign with the target position correctly.

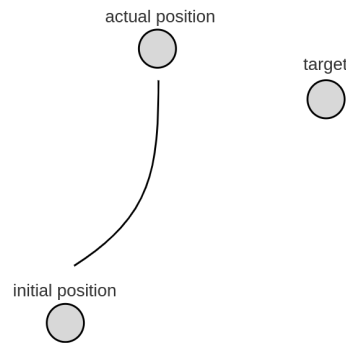


Figure 3.6: Significant deviation caused by a misaligned route

3.3.3 Rotation first

To address the problems raised in the approach described in Subsection 3.3.2, we prioritized rotation operations for the device, only ignoring new movement messages. This meant the device would realign itself whenever it exceeded the expected 15° threshold. However, this approach introduced new challenges.

The device received a high volume of messages, which caused it to become easily flooded and operate more slowly. Another significant problem was the reliance on outdated angle messages. The device would be moving, and only a few milliseconds later, it would receive the rotation message, which was meant to be understood from a few positions before, as seen in Figure 3.7. This delay would cause severe rotation errors, making it difficult to place the device in the desired position correctly.

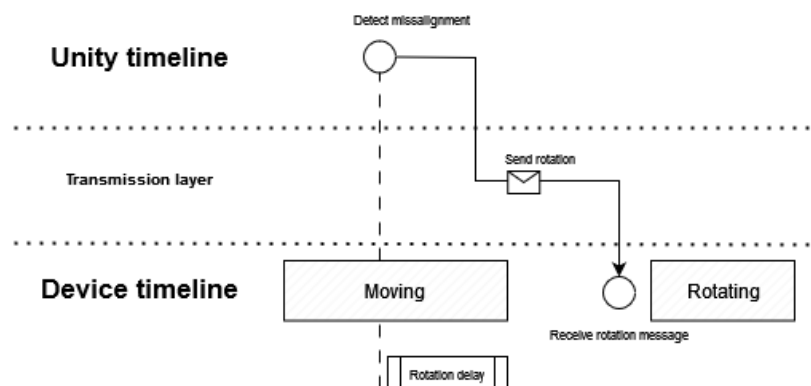


Figure 3.7: Timeline that causes misalignment

3.3.4 Segmented path

The optimal solution arises from integrating the best aspects of both approaches discussed in Subsections 3.3.3 and 3.3.2 to address previous issues. To mitigate the problem of outdated messages, we introduced a new message type that informs our software when the device stops moving. This enables the software to send new moves or rotate actions only when the device is stationary and awaiting instructions. The device provides this information as Figure 3.8 illustrates. With this knowledge, the device could accurately rotate in the correct directions.

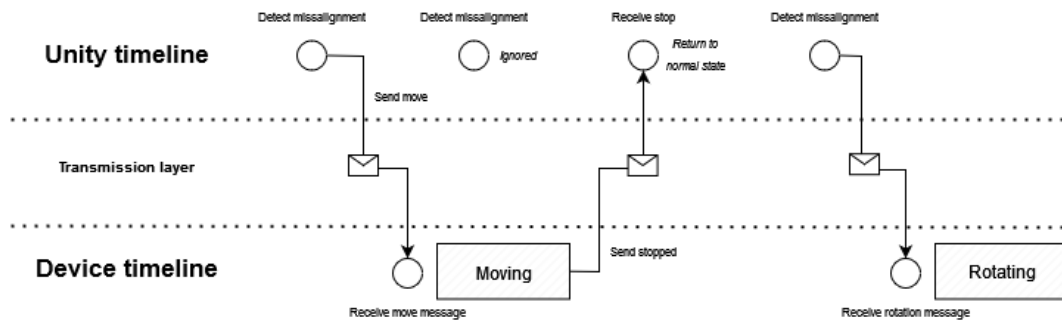


Figure 3.8: Timeline with improved solution

Another challenge addressed is adapting the solution from Subsection 3.3.3 to periodically adjust the direction over time, gradually correcting its movement direction as needed. We configured and parameterized a maximum step distance in the software, determining how far the device can move before reevaluating its rotation values. This capability significantly improves device movement, as shown in Figure 3.9, which greatly improves the solution over what happens in Figure 3.6.

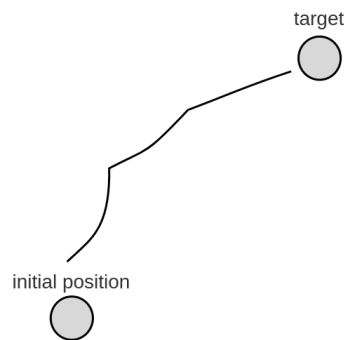


Figure 3.9: Improved sequence of operations to ensure fast positionment

3.4 Communication protocol

As mentioned in previous sections, we use the MQTT protocol [30] to support communication between the several components of our solution. Given that we use a microcontroller, we must ensure low energy consumption and minimal processing power requirements. MQTT is particularly well-suited for such constraints due to its lightweight and efficient design. MQTT clients are very small and require minimal resources, making them ideal for small microcontrollers.

Another advantage of MQTT is its support for bi-directional communications, allowing the device to send and receive messages reliably. This is crucial for maintaining a consistent and responsive interaction with the device. The reliability of message delivery is also a key feature, with MQTT offering three defined Quality of Service levels. In our solution, we decided to use the exactly once delivery (QoS level 2), ensuring that each message is delivered precisely once. This level of reliability is essential for ensuring that the correct instructions are communicated, thereby enhancing the overall robustness and accuracy of the system.

3.5 Hand detection

Later in this chapter, we will explain methodologies that require the software to know the precise location of the user's hand, which is crucial for maintaining accurate and smooth interactions. Our current approach involves using the Leap Motion Controller 2 [36] to detect hand positions. This device provides a Unity library with numerous prefabs to visualize the hands and a `LeapProvider` class that can be used to obtain information such as position, rotation, and other properties of hand gestures, such as pinching and whether the hand is closed.

However, we encountered challenges using the device out of the box, as the virtual hand often felt misaligned when attempting to grasp a prop. To resolve this, we needed to calibrate the hand position. The class provided by Leap Motion supports adding middleware to adjust certain properties, which we used to modify the hand's position. We could calculate a correction offset and apply it during the middleware step.

The HTC Vive Tracker was used to calculate the offset. Our first implementation calculated the distance from the palm to the tracker and applied this offset. However, this approach only worked when the hand was held in a specific direction, and the hand would incorrectly move backwards when the user rotated. This happened because the offset was calculated in the 3D world space and applied in the same direction.

To address this issue, we calculated the offset in a different 3D space: the view camera space. This involves transforming the offset vector into the 3D world with the origin at the view camera, considering the height and distance relative to the direction of the view. This method ensures that the hand remains correctly aligned in various orientations and positions, enhancing the interactions' accuracy and realism.

We later developed a simple Unity scene where the user can calibrate their hand by placing the tracker in their hand and pinching with the left hand, which triggers the correction calculations.

3.6 Interaction

We researched this topic using haptic props as a case study. Haptic props are simple physical objects designed to mimic specific properties of virtual objects, such as their shape or texture. Our approach involved utilizing HTC Vive Trackers to attach these haptic props, enabling us to interact with virtual objects within a virtual environment using physical counterparts.

Controlling these virtual objects was straightforward. By setting them as a child of the tracker object in Unity, it followed the movements of its parent during interaction. Initially, we considered calculating angle and position offsets to ensure smooth transitions between different states. However, we found that directly linking the objects to the tracker object was effective in practice. This simplified approach worked well without the need for complex calculations.

Our primary challenge was determining when the user began interacting with an object. Due to tracker reading errors, it was decided to keep virtual objects stationary and separate from the tracker object to prevent noticeable flickering. These errors also posed a significant challenge in accurately detecting when the user initiated interaction with an object.

3.6.1 Height check

Initially, we attempted a simple height check with a threshold to determine interaction. However, this approach made interactions feel unnatural, as the object would only begin moving milliseconds after the user touched it.

Additionally, it was difficult to calibrate the point at which the object should remain stationary due to tracker errors. These issues impacted the overall immersive experience for the user.

3.6.2 Vertical speed check

To address the issues described in Subsection 3.6.1, we attempted to implement a vertical speed check to determine if the object was being manipulated, coupled with a check on its vertical position relative to a specific threshold. This approach aimed to ensure the software could detect when the device was picked up and relocated to another position. However, this method introduced numerous interaction errors, resulting in a poor user experience.

The main challenge derived once again from the slightly unstable position readings provided by the trackers. Occasionally, the software would misinterpret minor movements or errors as significant vertical shifts, making it difficult to calibrate the speed threshold accurately for smooth interactions.

This inconsistency led to another problem. If a user picked up an object and paused mid-air before reaching the height threshold, the object would abruptly stop moving, disrupting the natural flow of interaction and causing a jarring sensation.

3.6.3 Using the hand position

Our latest approach proved the most accurate, offering smoother and more natural interactions. We leveraged hand position to initiate interactions, activating the virtual object whenever our hand approached it. This method ensured that the virtual object was dynamically placed as a child of the tracker game object precisely when interaction began.

Typically, when someone picks up an object, they focus their gaze directly on it, allowing us to assume that the hand will be visible during the initial interaction phase. However, as users move around with the object, they may lower their arm or focus on other aspects of the environment, causing the hand to move out of range of the Leap Motion camera and potentially breaking the interaction. Besides this, the Leap Motion device may also struggle to track the hand accurately when partially obscured by the object being held.

To address these issues, we built upon the solution described in Subsection 3.6.1 to maintain object interaction even when the hand is out of range. By implementing a height threshold, we can determine when the user has picked up the object and is still interacting with it, even if their hand is no longer detectable.

3.6.4 Ensuring natural interactions with robot control

Another important consideration is ensuring that the robot does not initiate movement prematurely. Such premature movement could create an unnatural experience for the user. They may feel weird vibrations or observe the device moving unexpectedly, particularly when picking up or setting down an object. To address this, the interactable object script can be simplified to incorporate three distinct states: idle, interacting, and airborne.

As depicted in Figure 3.10, objects start idle and transition to interacting when the user's hand approaches the virtual object within a specified threshold. In the interacting state, the object follows the parent (tracker). However, to prevent the robot from moving prematurely while the object is still near the device, movement is only triggered when the object transitions to the airborne state, detected when it reaches a certain height. This state change is communicated to listeners, including the robot controller.

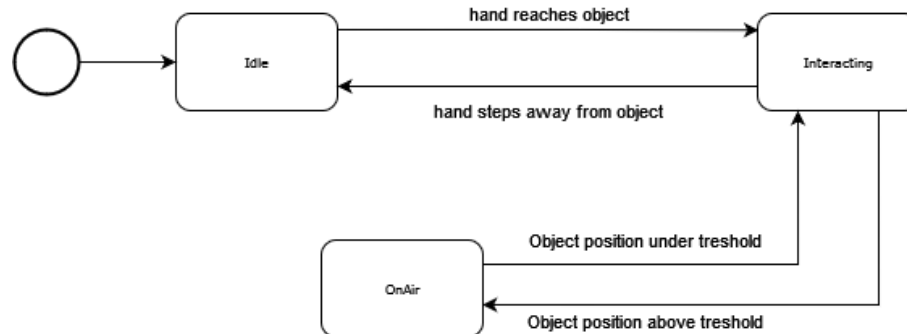


Figure 3.10: State machine of the haptic interactable state

The process is reversed to determine when an object has been safely placed down by the user. Movement only resumes when the object returns to the idle state, indicating that the user is no longer interacting with it and the hand is already far away. This approach ensures the robot remains stationary while the user maintains contact with the object.

This approach is naive and relies on height thresholds, potentially limiting the scope of interactions possible. One possible enhancement could involve integrating sensors, such as pressure or weight sensors, to accurately detect when the virtual object is no longer in contact with the device, thereby refining the interaction experience.

3.7 Candidate object detection

A critical challenge for this methodology is ensuring the software identifies the object the user intends to interact with. This knowledge is required to inform the device on where to move. One hypothetical solution could limit interactions to a single object at a time, transitioning to a new object only when the current one is no longer interactable. While this approach simplifies device positioning towards a single virtual object, it would restrict the range of interactions. Our goal, however, is to enable more meaningful interactions and enhance the state-of-the-art in handling graspable objects' limitations.

Our primary focus was developing the device from the beginning of our solution's design, relegating object selection to a secondary priority. That's why we devised a straightforward yet effective solution to address these challenges and facilitate diverse interactions. Since users interact with objects using their hands, a reasonable assumption is that they will interact with the closest object to their hand next.

This part of the solution already considers the possibility of using multiple moving robots. To support this, we iterate through all available devices to assign each device to a virtual object and find the closest active virtual object. This process could be further optimized by developing an algorithm that minimizes the total travel distance for all devices, prevents unnecessary device changes, and avoids collisions.

We also added support for device types. For example, if we want to simulate more than one object, we can specify the type of object each device can simulate. The query for virtual objects then considers both the type of virtual objects and the device.

This method provides a straightforward yet effective solution for our current implementation. Future enhancements could incorporate gaze direction, as users typically look at objects they intend to interact with, but this remains a topic for future exploration.

3.8 Custom Unity library

From the outset, we prioritized not only user experience but also developer experience. We aimed for extensibility to ensure that new developers could create virtual experiences without modifying

our library code. To achieve this, we designed our game library to adhere to principles similar to event-driven architectures.

We utilized Unity events, which resemble aspects of the observer and decorator patterns. The observer pattern allows for clean and straightforward behaviour connections between different components by enabling objects to subscribe to events. When changes occur, the producers notify these subscribed objects. This approach fosters a modular and maintainable codebase.

Unity events also present similarities to the decorator pattern, allowing further functionalities to be added to a specific class. Developers can select which functions should be called when a Unity event occurs directly within the user interface. This design choice enhances flexibility and promotes a robust development environment, ultimately supporting a wide range of user experiences without necessitating changes to the core library code.

For example, let's say we wanted to transport a haptic device that morphs its shape according to different properties of a virtual object instead of our simple haptic prop. Our moving device is informed of an object change through this method to know where to move, but we could add additional callbacks to that functionality. The new haptic device could seamlessly adapt by incorporating a callback to the method that informs the haptic device of the new properties of the next target. This would work out of the box without modifying the library code.

This extensible approach allows developers to introduce new functionalities, such as shape-morphing haptic devices or dynamically changing virtual environments based on the behaviour of our moving device, without altering the core library. By leveraging Unity events, we ensure the library remains flexible and scalable, efficiently accommodating future enhancements and new virtual experiences.

However, as we mentioned, some parts of our solution were implemented more simply due to less priority in our vision for this prototype. That's why, for each solution, we created abstract classes, as interfaces are not very well supported in the unity editor, which allows developers to create custom ways of interacting with the device or to find out what object will be interacted with next. Instead of making each class depend on a specific implementation, they rely on abstract classes or interfaces, so it is fast and easy to change the implementation in the unity UI by dragging a different concrete class.

3.9 Architecture

The solution comprises three main modules: the application part, the device, and the headset module, as depicted in Figure 3.11. The device module includes a microcontroller responsible for actuating the motors and a position tracker. Additionally, the headset incorporates a hand-tracking module to enable the solution to determine which objects the user will interact with. Finally, the application layer encompasses the application API, which parses all information received from both trackers and sends actions to the microcontroller.

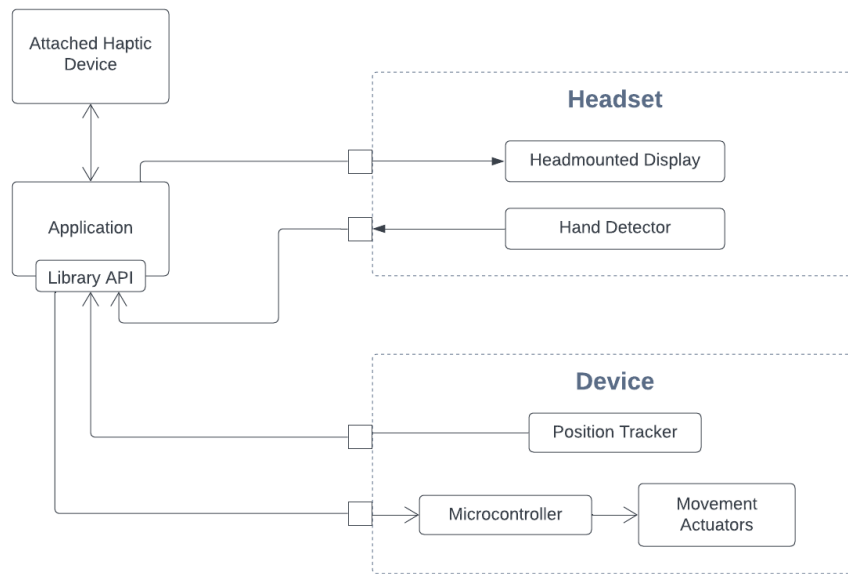


Figure 3.11: Architecture of the solution

In a more detailed way, the Library API can be further deconstructed into several modules, as seen in Figure 3.12. These modules are hand detection, object detection, and the device module, each with specific responsibilities and implementations.

The hand detection module is responsible for accurately detecting the position and rotation of the user's hand. This module is mainly represented by the `IHandDetector` interface and its implementation, `LeapHandDetector`, which relies on the Leap Motion Controller module to gather that information.

The object detection module determines which object the user will interact with next. It is represented by the `IObjectDetection` interface and its implementation `ClosestObjectDetection`. This module analyses the virtual environment and predicts user actions based on hand movements and object proximity, ensuring the device can prepare for future interactions.

The device module controls the physical device and is represented by the `IDevicePositioner` abstract class, which has two extensions: `MovableDevicePositioner` and `VirtualDevicePositioner`. `MovableDevicePositioner` interacts with the physical device, managing its movements to ensure it reaches the desired positions. `VirtualDevicePositioner`, on the other hand, manipulates a virtual fake device, which is useful for testing other parts of the program before developing the actual physical device.

To support these implementations, we rely on some external dependencies. The Leap Motion Controller module integrates with the Leap Motion Controller 2 device, providing the necessary data for hand detection. For communication, the M2Mqtt library for Unity ensures efficient and reliable data transmission between the device and the software. Lastly, the SteamVR library supports the HTC Vive Tracker, providing tracking data crucial for device positioning and enabling precise control and interaction within the virtual environment.

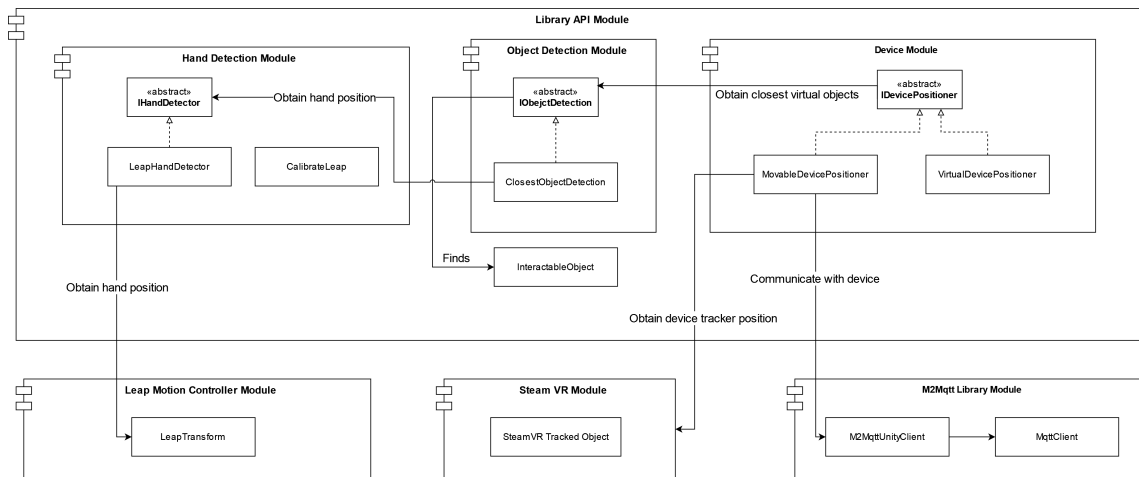


Figure 3.12: Detailed architecture of the library API

3.10 Device limitations

After completing the first version of our solution, we were able to quickly identify some of the device's strengths and weaknesses. We could place the device in the desired locations, usually less than 5 centimetres from the correct position.

However, some issues arose due to the device's movement constraints. The device could travel at approximately 30 centimetres per second. However, the device must first rotate toward the destination and then move forward, and would sometimes have some trajectory errors caused by the third free-rotating wheel. This resulted in longer travel times between different objects.

Another problem we encountered was the difficulty in precisely placing the device when the user wanted to set down the haptic prop. High precision is needed for this task, and natural human shaking when dropping the device made it hard for the device to stop moving promptly.

These challenges led us to consider possible solutions to address these limitations. To ensure the device reaches the desired virtual object on time, distractions can be created within the virtual experience, such as entertaining and captivating side quests that conclude once the device reaches the desired position. To mitigate the issue of the device not ensuring a valid drop zone, we could guide the user and the device to a specific drop zone using virtual elements such as a tube or small plate.

3.11 Chapter summary

This chapter focused on evaluating the state-of-the-art and proposing a new approach to be tested against our hypotheses and research questions.

Later, we developed a novel approach to enhance graspable haptic devices by incorporating mobility. Our objective was to simulate multiple virtual objects convincingly using fewer movable

devices, which posed several challenges. These challenges included predicting user interaction, accurately tracking and positioning the device, and managing communication protocols from low-level hardware to high-level interaction components. Throughout the chapter, we meticulously detailed the evolution and final implementation of each aspect that addressed these challenges.

At the end of this chapter, we also analyzed some of the known limitations of the device, which will be considered in evaluating our approach.

Chapter 4

Evaluation and Results

This chapter analyses whether we can answer our research questions to evaluate our device's effectiveness in creating immersive virtual experiences and improving the current state of the art. To draw conclusions, we had users perform a task using two different approaches.

We compared our approach of transporting a single haptic device with an approach using several haptic devices. Our goal was to determine if users noticed any differences between the two runs of the same task and if our device's functioning remained discreet while minimizing the resources needed to create engaging, immersive, and dynamic experiences.

4.1 Target haptic device

In this case, we decided to use a haptic prop as the haptic device. Haptic props are basic physical objects designed to replicate certain characteristics of virtual objects, such as their shape or texture. The reasoning behind this choice is that it significantly enhances the realism of the virtual experience while being straightforward to implement and create.

Our device consisted of an HTC Vive Tracker 2 attached to a small base made out of KLine. These components were securely joined using strong double-sided duct tape, which proved durable through several experiences without failure. As we will explain later, we chose virtual apples as our target objects. To replicate some of the apple properties, we used a simple ball similar in shape to an apple. Simulating the shape alone was sufficient, as it was not the main focus of the tests. Figure 4.1 shows the entire device setup.

4.2 Setup and test environment

The virtual environment consisted of an outdoor establishment with trees and rocks as scenery. It also included a table, some apples, and a juice machine. The virtual table mimicked the physical environment. The main purpose of using a table was to prevent the user from stepping on the device, thereby protecting it.

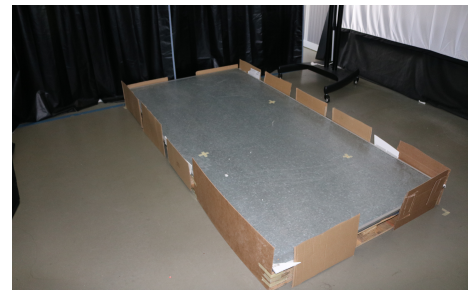


Figure 4.1: Haptic prop.

To ensure user safety, we designed the virtual table slightly larger than the physical one, providing extra space to prevent users from tripping. The comparison between the virtual and physical environments can be seen in [Figure 4.2](#).



(a) Virtual environment



(b) Physical environment

Figure 4.2: Virtual and physical environment used in the user tests.

Another issue we had to consider was the possibility of the device falling from the elevated platform. To prevent this, we added cardboard walls around the table to protect the device from potential falls, as shown in [Figure 4.3](#).

The participants used the HTC Vive Pro 2 with a Leap Motion Controller 2, both connected to a computer equipped with 32GB of RAM, an i9-13900KF CPU, and an NVIDIA GeForce RTX 4090 GPU. The users interacted with HTC Vive Trackers, versions 1 and 2 during the experiments.



Figure 4.3: Simple walls to prevent the device from falling from the platform.

4.3 Task definition

We used the same task but with different physical implementations to compare both methodologies. The objective of the task was to pick up three apples, one at a time, from a table and insert them into a machine to produce juice.

However, due to some of the constraints of our device, we had to introduce side quests. As previously explained in Section 3.10, our device presented significant challenges that needed to be mitigated to create a meaningful virtual experience. One challenge was the device's inability always to position itself on time. To address this, we introduced captivating side quests to distract the users while keeping them engaged in the experience.

In the user tests, we implemented butterflies flying over the apples, as seen in Figure 4.4. Users were instructed to wave their hands to send away the butterflies. This side quest served as a placebo effect to entertain users while the device positioned itself correctly.

Another addition to enhancing interaction and fun was requiring users to show the apple to a supervisor via a video surveillance camera before inserting it into the tube. This served the same purpose as the butterflies. This step was authorized when a green light appeared, as seen in Figures 4.5. This mechanism was designed to give the device additional time to position itself while keeping the user entertained.

One last important issue to address was the device's inability to correctly position itself when the user wanted to place the object down. To solve this, we indicated that the experience should guide the user and the device to the same location. We directed the users to drop the apple into a small tube, as shown in Figure 4.6, minimizing the margin for error in positioning the device. This was achieved by using Unity events to monitor the interaction steps and introducing a custom behaviour in the device, which allows it to follow a specific object, in this case, the tube, rather than selecting candidate objects from the object detector described in Section 3.7.



Figure 4.4: Butterflies distracting the user.



(a) Waiting for authorization



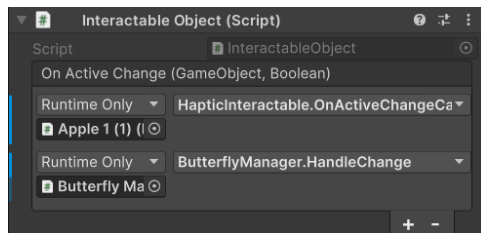
(b) Authorized

Figure 4.5: Validating the apple with the supervisor.

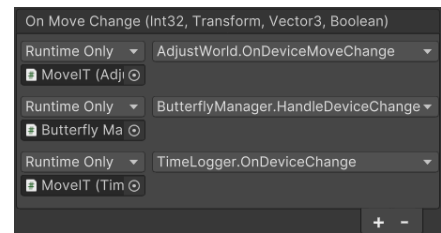
Each interaction point was straightforward to implement due to our library's extensibility and Unity events, as explained in Section 3.8. We only needed to create script files for the butterflies, the camera and the tube door. Then, we defined functions that matched the Unity event method signatures for interactable object state changes and device movement updates. When the candidate object changed, the butterflies would switch to a new position, in which the Unity event regarding this can be seen in Figure 4.7a. The butterflies would only disappear once the Unity event signalled that the device had stopped, as shown in Figure 4.7b.



Figure 4.6: Tube to place the apple.



(a) When candidate object changes



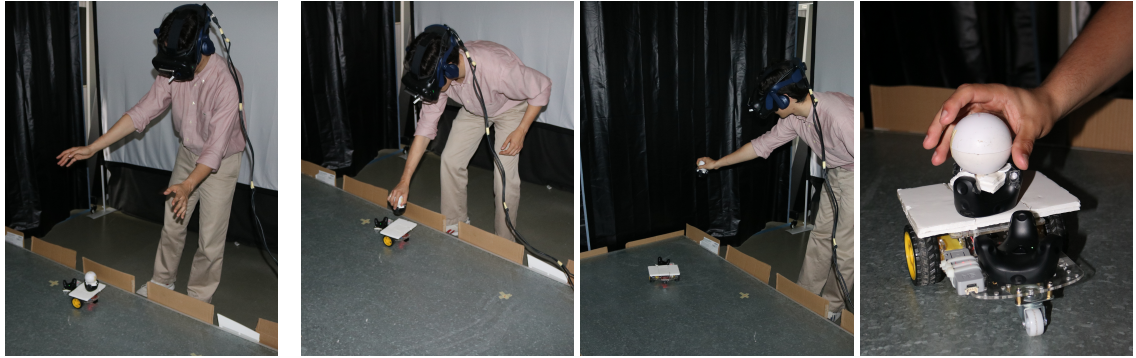
(b) Callback when device stops moving

Figure 4.7: Validating the apple with the supervisor.

4.4 Our approach

Our approach involved moving a single haptic prop throughout the physical environment to mimic the position of the virtual devices the user intended to interact with. Since the device produced some noise, we masked this with ambient sounds, such as forest and bird chirps, to ensure the user could not detect the methodology of the task from the noise.

Throughout our experiment, the users moved around the table to pick up objects from the moving robot after sending away the butterflies. Next, they showed the apple to the supervisor while the robot moved to the drop-off point. Finally, the users placed the apple back into the device. This procedure is illustrated in Figure 4.8. Our approach allowed the experience to continue seamlessly without any external influence, demonstrating the capability to create dynamic and autonomous experiences.



(a) User interacting with the butterflies. (b) User picking up the apple. (c) User showing the apple to the supervisor. (d) User placing the apple down in the tube.

Figure 4.8: Sequence of interaction from a user in the test.

4.5 Baseline

To evaluate our approach, we wanted to compare it with a baseline. We aimed to demonstrate that we can simulate multiple virtual objects with fewer physical devices without users noticing. Therefore, the baseline approach involved using individual haptic props for each virtual object. This required an equal number of haptic props to match the number of virtual objects.

Due to material constraints, we had to replicate the haptic prop described in Section 4.1 using different HTC Vive Trackers. The differences in weight and size were minimal, as shown in Figure 4.9, so they should go unnoticed by the users.



Figure 4.9: The three haptic props used in the experience.

To implement this, we added a Unity script that, at the beginning of the experience, aligns each virtual object to the correct position in the physical world. This script sets the XZ position of the virtual object to match the XZ position of the corresponding HTC Vive Tracker. This way, when the user interacts with a virtual apple, it will also pick up the haptic prop.

As mentioned earlier, we introduced side quests to mitigate some device issues. However, with this method, the haptic props consistently remain in the correct positions, rendering these distractions unnecessary. We needed a solution to ensure users remained engaged without noticing the underlying tasks due to the lack of these distracting elements. We developed a virtual moving device early in our development process to help us develop the project without the physical device materials. Reusing this approach allowed us to simulate the behaviour of the physical device. As a result, we could maintain the distracting elements so that users couldn't identify the task underneath.

A problem with this methodology is the inability to create dynamic experiences, as the haptic props cannot be moved autonomously after use. In the context of the experience, once the user placed the apple in the juice machine, that apple would be gone and no longer interactable. With this approach, once the user places down one of the haptic props, it remains in the same place, and the user will likely try to place other haptic props on top of the first one. This can be observed in Figure 4.10, where both haptic props will collide if there is no external intervention. There is no optimal way to address this issue dynamically. Consequently, we had to manually remove the prop after the user stopped interacting with it, requiring external intervention during the experience, as opposed to our dynamic approach.



Figure 4.10: User in the experience using multiple haptic props

4.6 Methodology

The user tests consisted of two runs of the same task, each using a different methodology. The order of the methodologies for each run was chosen randomly and noted on the form.

Before starting the experiments, participants were asked to fill out a consent form, allowing us to record images and gather data. They also provided information about their background with virtual reality and their demographic data. After this, participants entered the virtual scene to adjust the headsets and familiarize themselves with the environment and virtual hands. Before

performing the task, they were instructed on the task and shown a video tutorial to understand the steps needed to complete the experience successfully.

The participants were then ready to start the tasks. We guided them to the scene and signalled when to begin the task. This information was important, as it sometimes took additional time to calibrate the user test due to the trackers' lack of fidelity. Moreover, it was crucial that participants left the scene without seeing the physical environment to avoid revealing which task was which. After each task, participants filled out a form to answer important questions regarding their experience.

Upon completing both tasks, participants were instructed on how each methodology functioned. Once they understood clearly, they were asked to identify the sequence of tests they had experienced. This assessment helped us determine if they could distinguish between the methodologies and the reasons behind it.

4.7 Data collection

To comprehensively evaluate and compare both methodologies, we collected participant data. This approach allowed us to gain insights into the performance and effectiveness of our method.

Most of our metrics were subjective and aimed at understanding the user experience during the tasks. To gauge any differences in user perception between the tasks, we collected data on how natural interactions felt and the number of real objects they interacted with. This data was gathered before participants were informed about the workings of each method, ensuring their responses were unbiased by prior knowledge. The questions the user had to answer were the following:

- *How successful were you in accomplishing what you were asked to do?* - scale from Perfect (1) to Failure (5)
- *How insecure, discouraged, irritated, stressed, and annoyed were you?* - scale from very low (1) to very high (5)
- *How natural did the interactions feel?* - scale from Very natural (1) to Not natural (5)
- *Did you notice anything strange during the task? If so, please explain.*
- *How many different real physical objects did you interact with?* - numeric from 0 to 4
- *What is the level of confidence of your previous answer?* - scale from very low (1) to very high (5)

In the end, participants would also have to answer a question regarding the order of the tests, based on their experience and our explanation of both methodologies: *What was the order of the conditions you experienced?*

We also gathered the distances and time the device travelled from one place to another. With this information, we could calculate the velocities of the device, including the adjustments, not only the linear velocity previously mentioned.

4.8 Participants

Throughout our experiments, a total of 32 people went through our process. All the participants consented to the process of collecting their data for anonymous usage. Only 1 user did not allow the collection and usage of pictures during the experience. Of these participants, 22 were between 21 and 30 years old, while the remaining 10 were between 16 and 20.

Regarding educational background, most participants held a Bachelor's degree or were pursuing one. There were 21 participants with a Bachelor's degree and 10 with secondary education. Additionally, 1 participant held a Master's degree.

When examining the participants' experience with VR, it was clear that the group had varying levels of familiarity. 3 participants frequently used VR (several times a month), 1 who used it monthly (once or twice a month), and 4 who used it annually (once or twice a year). A significant portion of the participants, 13 in total, had only used VR once or a few times before, while 10 participants had never used VR.

Among the 32 participants, the occupational distribution showed that 26 were students, 2 were workers, and 6 of them were both working and studying.

Out of the 32 participants, 2 tests were considered invalid. One test was invalid because the juice machine did not trigger the juice-making process. This occurred because the user's hand was not visible to the system. The second invalid test happened because the interactions were disrupted when the user did not follow the correct order of actions, preventing them from completing the task correctly.

4.9 Results

By examining subjective metrics and employing statistical methods and descriptive analysis, we can understand how users felt during both runs of the task. We also analyzed open-ended questions to allow users to express any unusual situations they might have experienced.

4.9.1 Analysis of tasks data

To understand if users felt differently when completely immersed in the virtual environment and the activity, we asked them about their experience after each task without prior knowledge of the order identification question. This approach ensured that users focused more on the tasks and the experience than on potential external factors.

4.9.1.1 Participants observations through an open-ended question

At this stage, asking participants if they had felt anything weird so far was important. This answers the question *Did you notice anything strange during the task? If so, please explain.* This allows us to understand what might trigger the user to realize what methodology they are experiencing

and if it changes between tasks. We can also measure how much the device will impact on the immersiveness of the experience.

Some users mentioned that the texture or shape of the apples was not quite right, as it was a rubbery ball. However, both experiences used the same props, so this didn't trigger a different reaction in either run.

Some users also noticed tracker errors, such as the apple floating away and returning to the hand position. They also noticed that sometimes the apples were not picked up when they actually did due to trackers not being visible. However, this could happen in both runs, so it wasn't a reason to suspect any particular methodology.

One user noticed that the supervisor already confirmed one of the apples, probably because the device was already in position. This user also noticed that the table heights differed as the device transported the props in a higher position. These made them feel suspicious, but they could not identify the correct order of the tasks.

Some users also mentioned that sometimes the butterflies were annoying and took a lot of time to send away, but they did not mention that in the reason behind choosing the order of the tasks in the later question.

Some users noted that occasionally, the physical apples were not yet synchronized with their virtual counterparts or that they were misaligned. These were the only consistent observations between this question and the last one regarding the order of the tasks.

4.9.1.2 Closed-ended questions

We employed statistical methods to analyze the remaining data and to understand if there were significant differences between the two methodologies. We used an alpha value of 5% to draw statistically relevant conclusions.

The questions focused on how participants felt during the experiment and the overall pleasantness of the user interaction. We wanted to understand if the two methodologies felt different to the participants. We had two runs of the same task, meaning the answers were paired samples. Additionally, since the questions were qualitative and the answers were subjective to their personal experience, we considered the data did not follow a normal distribution. Due to these reasons, we decided to use the Wilcoxon signed-rank test.

Performing the Wilcoxon test on the 30 valid tests produced the results shown in Table 4.1. The *Robot* column refers to the experience where the robot transported the haptic prop, while the *Static* column refers to the results from the experience using the three static haptic props. That column also contains the median and the interquartile range. From the table, we can observe that the values are not statistically significantly different, as all medians are the same and the p-values from the Wilcoxon test are consistently above 0.05.

Another interesting observation is that most users reported interacting with three physical objects during the robot experience. This suggests that users did not realize they were interacting with a single haptic prop in the robot task.

Question	Median (IQR)		Wilcoxon Signed Ranks
	Robot	Static	
How successful were you in accomplishing what you were asked to do?	1 (1)	1 (0)	$Z = -0.642, p = 0.521$
How insecure, discouraged, irritated, stressed, and annoyed were you?	1 (1)	1 (1)	$Z = -1.403, p = 0.161$
How natural did the interactions feel?	2 (2)	2 (2)	$Z = -0.975, p = 0.329$
How many different real physical objects did you interact with?	3 (0)	3 (0)	$Z = -.378, p = 0.705$
What is the level of confidence of your previous answer?	4.5 (1)	4 (1)	$Z = -0.225, p = 0.822$

Table 4.1: Median (IQR), and Wilcoxon Signed Ranks test results for each of the questions for both approaches.

4.9.2 Analysis of the order identification question

To better understand how effectively our approach masked its functionality, we asked users to identify the order in which the tasks were performed, even after they were given a detailed explanation of how the system worked. Initially, users may not have noticed anything unusual, but after being informed about the methodology, they might become aware of details they had previously overlooked.

This question was paired with a confidence level rating to measure how certain users were of their responses. To facilitate analysis, we categorized the answers into four groups: wrong or clueless (confidence level of 1), correct with some certainty (confidence levels of 2 and 3), correct with a lot of certainty (confidence level of 4), and correct with absolute certainty (confidence level of 5).

Out of the 30 valid experiences, 18 users (60%) answered with minimal confidence (confidence level 1) about the order of the methodologies or guessed wrongly, even after understanding the intricacies of the experiences. From the remaining 12 users who correctly guessed and were confident about the answer (confidence level above 1) the order (40%), only 1 (3.33%) was absolutely certain of their answer, while 4 (13.33%) correctly identified the order with a lot of certainty. The remaining 7 users (23.33%) correctly guessed the order with some certainty.

We also analyzed the answers based on the level of confidence, obtaining a median confidence level of 2 with an interquartile range of 2. This indicates that the confidence levels of the participants were slightly skewed towards the lower values, as shown in Figure 4.11. This analysis suggests that even though some participants guessed correctly, the majority remained uncertain and slightly reluctant when choosing the order of the approaches. This implies that both approaches felt quite similar to the participants.

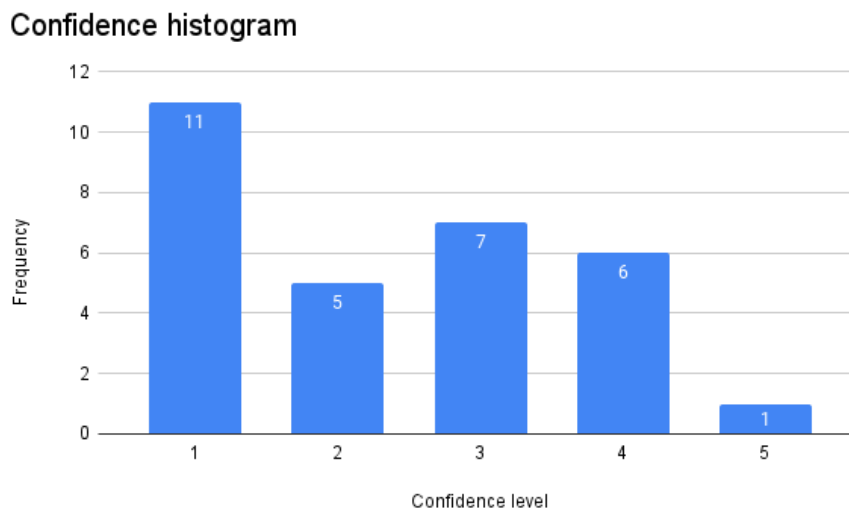


Figure 4.11: Histogram of the confidence levels.

4.9.2.1 Participants justifications

To support their answers, participants had to justify and identify the aspects that gave away the methodology used in each task.

The only participant who was absolutely certain of their correct guess (confidence level 5) based their answer on the fact that the robot task took longer to be ready for interaction. This user understood the reason behind the butterflies and the cameras. Another participant used this justification but was only slightly certain (confidence level 2).

Another participant, who was somewhat certain of their guess (confidence level 3), mentioned that in the multiple haptic props experience, they felt a slight difference between the objects. This was the only case where a user noticed this particular difference.

Three participants, one almost certain (confidence level 4) and two slightly certain (confidence levels 2 and 3), mentioned that the virtual and physical objects in the robot experience were slightly misaligned. This led them to believe that the robot had misplaced the objects. However, two other participants used this justification but incorrectly guessed the methodology. This indicates that sometimes, the robot was more precise than the multiple haptic props experience, possibly due to tracker errors.

Five participants correctly guessed the order of methodologies because the robot had not arrived at the location yet, leading them to try to grasp nothing initially and then, after some time, grab the haptic prop in the same place. Three of those were almost certain of their answer (confidence level 4), and the other two were somewhat confident (confidence level 3). This might have occurred because the butterflies were arriving but were still not within the user's field of view, causing them to attempt to pick up the apple before sending the butterflies away.

Another participant, who correctly guessed with some certainty (confidence level 3), claims to have felt a magnet sensation, stating that the device probably had a magnetic way to clamp the

haptic prop in place. This was probably due to the car's material being softer than the material of the virtual table.

Finally, one participant who correctly guessed with slight confidence (confidence level 2) stated that they could not distinguish between the two conditions.

4.9.3 Analysis of distance and time travelled

Calculating the linear velocity is not enough to understand our device's limitations. Due to the complex movements the device has to make to adjust the direction of the target object, it usually takes much longer.

After excluding distances smaller than 5 centimetres, typically resulting from adjustments, we calculated the velocities of the device. From 162 distance and time measurements, we determined the average velocity of the device. It could move at approximately 13.5 centimetres per second, with a standard deviation of 5 centimetres. This value is significantly lower than the previously calculated linear velocity of 30 centimetres per second.

4.9.4 Results discussion

Following the analyses of the three types of data recovered, we now have enough information to further comprehend how our approach behaves compared to a simpler one.

The first study suggests that participants did not perceive a significant difference between both approaches when fully immersed in the virtual environment. This indicates that most users were unaware of the underlying mechanics of the interactions and were more focused on the actual tasks. Additionally, there is no statistically significant difference in how natural the interactions felt to the participants, which is a strong indicator that our approach effectively masked its underlying technology.

One notable observation is that users were quite certain they were interacting with three physical objects, even in the robot experience. This means they did not realize they were repeatedly interacting with the same physical device, further demonstrating the effectiveness of the masking strategy.

Following this analysis, we also sought to determine if users could distinguish between the two approaches. This understanding was crucial to identify weak points in our method and to see if any particular aspects revealed the underlying approach. Our findings showed that most participants could not correctly identify the methodology, even after being informed about the intricacies of the approaches. However, some participants, when given more time and perspective, could think harder and pinpoint potential giveaway elements that might reveal the approach behind the experience.

We also noted that the responses to this final question were easily influenced by our explanations. For instance, one participant claimed to have felt magnet sensations, which was not part of our implementation.

The last study concludes that the actual physical capabilities of the device are not fully maximized. Although the vehicle can achieve a velocity of 30 centimeters per second, the complex movements required to rotate in the proper directions significantly reduce its actual speed. This indicates that even if we increase the power of the current materials, it is unlikely to substantially improve the overall positioning speed of the device.

Overall, we believed that most of the correctly identified task runs could be addressed in the future by enhancing the speed at which the device moves and by improving the quality of the materials used. Additionally, to address issues related to the robot not arriving on time, we could enhance the distracting elements, making them more engaging and effective at capturing the users' attention. This would encourage users to wait for the robot to reach the correct destination before continuing their task.

4.9.4.1 Research questions

With this comprehensive analysis of the data gathered from participants, we can now try to answer the research questions we proposed.

RQ1 Is it possible to simulate a diverse range of virtual objects using a reduced number of mobile devices in a way that remains imperceptible to the user? Overall, yes, since there was no significant difference in the way users felt about both runs of the same task in terms of how natural and how stressed they felt. However, given more information and time to think, some participants were able to identify certain giveaways to the approach.

RQ2 What are the limitations on the capabilities of the mobile device that allow it to still precisely replicate the positioning of virtual objects in a timely manner? The major limitation is the device's speed, which can prevent it from being in the correct place in time for the user to interact with it. However, we can create distracting interaction elements to make the user wait while the device moves into position.

4.10 Chapter summary

In this chapter, we delve into the practical evaluation of our approach. We conducted experiments with participants to address our research questions and assess the practicality and limitations of our device.

Firstly, we provide an overview and characterization of the haptic device chosen for our study, detailing the physical setup and virtual test environment. The experiments involved performing identical tasks using two different approaches: our novel approach versus a baseline utilizing multiple haptic props.

We outline the types of questions and data collected from participants, which primarily focused on subjective user experiences. This data was crucial for determining if users perceived any differences between the two approaches.

Following the experiments, we thoroughly analysed the gathered data to ascertain participant perceptions. We found that participants were deeply engaged in both experiences, with no statistical difference in their subjective feedback between the two approaches. However, after receiving explanations and additional reflection, some participants were able to accurately identify the sequence of approaches used, while many remained uncertain or guessed incorrectly.

Chapter 5

Conclusion

In our work we have laid the groundwork for innovation in the field of haptic devices within virtual reality. By conducting a comprehensive examination of the state-of-the-art, we have identified various types of haptic devices, including grounded, wearable, graspable, and movable devices. Through meticulous analysis, we have evaluated the unique characteristics, implementations, and respective advantages and disadvantages of each of these types.

Our research has led us to identify a specific problem deserving further investigation: the limited mobility of graspable haptic devices. While graspable devices hold promise for portraying natural interactions within VR environments, their stationary nature poses challenges when simulating interactions with multiple virtual objects. For this to be possible, we need to rely on multiple stationary devices, restricting the breadth of possible interactions.

Building on these research findings, we have defined a clear problem statement and formulated a hypothesis to guide our efforts in addressing this challenge. To study our approach, we defined the concept of Move Your Haptic, a moving device that can move graspable haptic devices to the position of virtual objects. This way, when users interact with virtual objects, they will interact with the physical counterparts.

We have also developed a Unity library to enable developers to easily integrate various types of interactions. The extensibility of the library is powered by Unity Events, which allow developers to add callbacks to our custom events. This enables developers to extend the behavior of our library without modifying its core code. One of the most promising use cases of this feature is when our moving device transports a complex haptic device. If the haptic device needs to be informed about the properties of a virtual object that the user is about to interact with, it can be easily managed. This can be simply achieved through the provided events.

To evaluate our approach, we conducted user tests comparing our method of moving a single haptic prop against an approach using multiple haptic props. The experiment consisted of participants performing two runs of the same task, transporting an apple to a tube, using both approaches. Our goal was to determine if users felt a difference between the two methods, collecting data on stress and the naturalness of the interactions, for example. At the end of both experiences, we also wanted to see if users could identify the order in which the approaches were presented.

After analyzing the results, we found that our approach and the baseline did not present significant differences in how users felt throughout the interactions. However, after instructing participants on how both approaches worked, some were able to distinguish between them after thinking for some time, though not with much confidence. Throughout the experiences, when participants were fully immersed, they did not feel a difference, validating our hypothesis in situations where users are fully engaged in the experience.

Throughout the experiments, we also noticed a lot of room for improvement. As mentioned when answering the limitations of the device, one of the most important things to solve is the speed of the device, as some users tried to interact with virtual objects whose physical counterpart was still arriving at the location.

With our research, we successfully created a working proof of concept that can enhance the current state-of-the-art in graspable devices. Our approach allows the reuse of a single haptic device to create dynamic experiences where users can interact with multiple objects. In traditional experiences using multiple haptic devices, the devices must be placed correctly from the beginning of the experience. With our approach, virtual experiences can change over time, enabling more expressive and versatile interactions, proving to be a great approach to augment the powerful interactions that graspable devices bring.

5.1 Future Work

We believe that this research represents a significant advancement in the area of graspable haptic devices. It presents a great opportunity to make graspable devices a more viable option for haptic interaction. However, due to time constraints, we had to limit the scope of our ideas. This means that there are several potential directions to further improve and refine our approach.

One of the most crucial aspects to address is the current speed limitations. This can be improved by using better components and adding two additional motors and wheels to provide more stability to the device. Sometimes we noticed that the device would misalign its trajectory due to the free third wheel. However, even with these enhancements, it might not be enough to significantly improve our vehicle's speed. The fact that the device first has to rotate to the correct direction before moving forward causes it to lose a lot of time. To further improve this, we suggest using mecanum wheels, which are capable of moving in any direction, as explained in [A](#). This would likely address the issue of having to set a specific drop point, allowing users to drop objects anywhere instead.

In our use case, we used a simple haptic prop consisting of a lightweight soft ball and an HTC Vive Tracker. These components provided excellent grip, so we did not need a complex solution to hold the device in place. However, for larger and more complex devices with higher centers of mass, further research is needed to ensure the devices remain stable and secure when moving the robot. This will be even more important when the device can travel at higher speeds, increasing the chance of objects falling.

Currently, for interactions and the moving device to function properly, knowledge of the hand position is required to determine if the user has released the device. To improve this, one could employ a simple weight sensor detect if the device is still on top of the robot. This is crucial to ensure that the moving device only moves after the user is no longer in contact with it, preventing unexpected movements.

To ensure we can provide multiple types of haptic feedback in a physical environment, it is important to fully develop the multi-device approach. However, this introduces additional challenges, particularly the need to avoid collisions between multiple devices. Solving the issue of device collisions would also likely address potential collisions with users, representing a significant advancement in the safety and functionality of the system.

Finally, our approach currently only works when the virtual objects we want to represent are all at the same vertical position, in the virtual environment. To provide a more immersive experience, it would be beneficial to research an approach where the transported haptic device could also change its vertical position, using a vertical lift, for example. This would expand the range of possible interactions and enhance the overall user experience.

References

- [1] Parastoo Abtahi, Benoit Landry, Jackie (Junrui) Yang, Marco Pavone, Sean Follmer, and James A. Landay. Beyond The Force: Using Quadcopters to Appropriately Objects and the Environment for Haptics in Virtual Reality. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, pages 1–13, New York, NY, USA, May 2019. Association for Computing Machinery.
- [2] Alexander Achberger, Frank Heyen, Kresimir Vidakovic, and Michael Sedlmair. PropellerHand: A Hand-Mounted, Propeller-Based Force Feedback Device. In *Proceedings of the 14th International Symposium on Visual Information Communication and Interaction*, VINCI '21, pages 1–8, New York, NY, USA, November 2021. Association for Computing Machinery.
- [3] Bruno Araujo, Ricardo Jota, Varun Perumal, Jia Xian Yao, Karan Singh, and Daniel Wigdor. Snake Charmer: Physically Enabling Virtual Objects. In *Proceedings of the TEI '16: Tenth International Conference on Tangible, Embedded, and Embodied Interaction*, TEI '16, pages 218–226, New York, NY, USA, February 2016. Association for Computing Machinery.
- [4] Filipe Barbosa, Daniel Mendes, and Rui Rodrigues. Shape-A-Getti: A haptic device for getting multiple shapes using a simple actuator. *Computers & Graphics*, 117:42–50, December 2023.
- [5] M. Bouzit, G. Burdea, G. Popescu, and R. Boian. The Rutgers Master II-new design force-feedback glove. *IEEE/ASME Transactions on Mechatronics*, 7(2):256–263, June 2002. Conference Name: IEEE/ASME Transactions on Mechatronics.
- [6] Lal Lila Bozgeyikli and Evren Bozgeyikli. Tangiball: Foot-Enabled Embodied Tangible Interaction with a Ball in Virtual Reality. In *2022 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 812–820, March 2022. ISSN: 2642-5254.
- [7] Shaoyu Cai, Pingchuan Ke, Takuji Narumi, and Kening Zhu. ThermAirGlove: A Pneumatic Glove for Thermal Perception and Material Identification in Virtual Reality. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 248–257, March 2020. ISSN: 2642-5254.
- [8] Inrak Choi, Heather Culbertson, Mark R. Miller, Alex Olwal, and Sean Follmer. Gravity: A Wearable Haptic Interface for Simulating Weight and Grasping in Virtual Reality. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, pages 119–130, Québec City QC Canada, October 2017. ACM.
- [9] Inrak Choi and Sean Follmer. Wolverine: A Wearable Haptic Interface for Grasping in VR. In *Adjunct Proceedings of the 29th Annual ACM Symposium on User Interface Software*

- and Technology*, UIST '16 Adjunct, pages 117–119, New York, NY, USA, October 2016. Association for Computing Machinery.
- [10] Inrak Choi, Eyal Ofek, Hrvoje Benko, Mike Sinclair, and Christian Holz. CLAW: A Multifunctional Handheld Haptic Controller for Grasping, Touching, and Triggering in Virtual Reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 1–13, New York, NY, USA, April 2018. Association for Computing Machinery.
- [11] Heather Culbertson, Julie M. Walker, Michael Raitor, and Allison M. Okamura. WAVES: A Wearable Asymmetric Vibration Excitation System for Presenting Three-Dimensional Translation and Rotation Cues. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, pages 4972–4982, New York, NY, USA, May 2017. Association for Computing Machinery.
- [12] Xavier de Tinguay, Thomas Howard, Claudio Pacchierotti, Maud Marchal, and Anatole Lécuyer. WeATaViX: WEearable Actuated TAngibles for VIRTual Reality eXperiences. In Ilana Nisky, Jess Hartcher-O'Brien, Michaël Wiertelowski, and Jeroen Smeets, editors, *Haptics: Science, Technology, Applications*, Lecture Notes in Computer Science, pages 262–270, Cham, 2020. Springer International Publishing.
- [13] Takahiro Endo, Haruhisa Kawasaki, Tetsuya Mouri, Yasuhiko Ishigure, Hisayuki Shimomura, Masato Matsumura, and Kazumi Koketsu. Five-Fingered Haptic Interface Robot: HIRO III. *IEEE Transactions on Haptics*, 4(1):14–27, January 2011.
- [14] Cathy Fang, Yang Zhang, Matthew Dworman, and Chris Harrison. Wireality: Enabling Complex Tangible Geometries in Virtual Reality with Worn Multi-String Haptics. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, pages 1–10, New York, NY, USA, April 2020. Association for Computing Machinery.
- [15] A. Gferrer. Geometry and kinematics of the Mecanum wheel. *Computer Aided Geometric Design*, 25(9):784–791, December 2008.
- [16] Dustin T. Goetz, David K. Owusu-Antwi, and Heather Culbertson. PATCH: Pump-Actuated Thermal Compression Haptics. In *2020 IEEE Haptics Symposium (HAPTICS)*, pages 643–649, March 2020. ISSN: 2324-7355.
- [17] Ryota Gomi, Ryo Suzuki, Kazuki Takashima, Kazuyuki Fujita, and Yoshifumi Kitamura. InflatableBots: Inflatable Shape-Changing Mobile Robots for Large-Scale Encountered-Type Haptics in VR. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, CHI '24, pages 1–14, New York, NY, USA, May 2024. Association for Computing Machinery.
- [18] Eric J Gonzalez, Eyal Ofek, Mar Gonzalez-Franco, and Mike Sinclair. X-Rings: A Hand-mounted 360° Shape Display for Grasping in Virtual Reality. In *The 34th Annual ACM Symposium on User Interface Software and Technology*, pages 732–742, Virtual Event USA, October 2021. ACM.
- [19] Seongkook Heo, Christina Chung, Geehyuk Lee, and Daniel Wigdor. Thor's Hammer: An Ungrounded Force Feedback Device Utilizing Propeller-Induced Propulsive Force. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 1–11, New York, NY, USA, April 2018. Association for Computing Machinery.

- [20] Matthias Hoppe, Pascal Knierim, Thomas Kosch, Markus Funk, Lauren Futami, Stefan Schneegass, Niels Henze, Albrecht Schmidt, and Tonja Machulla. VRHapticDrones: Providing Haptics in Virtual Reality through Quadcopters. In *Proceedings of the 17th International Conference on Mobile and Ubiquitous Multimedia*, MUM '18, pages 7–18, New York, NY, USA, November 2018. Association for Computing Machinery.
- [21] Joe Siang Keek, Ser Lee Loh, and Shin Horng Chong. Comprehensive Development and Control of a Path-Trackable Mecanum-Wheeled Robot. *IEEE Access*, 7:18368–18381, 2019. Conference Name: IEEE Access.
- [22] Pascal Knierim, Thomas Kosch, Valentin Schwind, Markus Funk, Francisco Kiss, Stefan Schneegass, and Niels Henze. Tactile Drones - Providing Immersive Tactile Feedback in Virtual Reality through Quadcopters. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '17, pages 433–436, New York, NY, USA, May 2017. Association for Computing Machinery.
- [23] Ginam Ko and SangHun Nam. Bstick: Handheld Virtual Reality Haptic Controller for Hand Rehabilitation. *Systems*, 10(3):54, June 2022. Number: 3 Publisher: Multidisciplinary Digital Publishing Institute.
- [24] Jinwoo Lee, Heayoun Sul, Wonha Lee, Kyung Rok Pyun, Inho Ha, Dongkwan Kim, Hyojoon Park, Hyeonjin Eom, Yeosang Yoon, Jinwook Jung, Dongjun Lee, and Seung Hwan Ko. Stretchable Skin-Like Cooling/Heating Device for Reconstruction of Artificial Thermal Sensation in Virtual Reality. *Advanced Functional Materials*, 30(29):1909171, 2020. _eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/adfm.201909171>.
- [25] Pedro Lopes, Alexandra Ion, and Patrick Baudisch. Impacto: Simulating Physical Impact by Combining Tactile Stimulation with Electrical Muscle Stimulation. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, UIST '15, pages 11–19, New York, NY, USA, November 2015. Association for Computing Machinery.
- [26] Pedro Lopes, Sijing You, Lung-Pan Cheng, Sebastian Marwecki, and Patrick Baudisch. Providing Haptics to Walls & Heavy Objects in Virtual Reality by Means of Electrical Muscle Stimulation. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, pages 1471–1482, New York, NY, USA, May 2017. Association for Computing Machinery.
- [27] Jasmine Lu, Ziwei Liu, Jas Brooks, and Pedro Lopes. Chemical Haptics: Rendering Haptic Sensations via Topical Stimulants. In *The 34th Annual ACM Symposium on User Interface Software and Technology*, UIST '21, pages 239–257, New York, NY, USA, October 2021. Association for Computing Machinery.
- [28] Thomas H Massie and J Kenneth Salisbury. The phantom haptic interface: A device for probing virtual objects. volume 55, pages 295–300. Chicago, IL, 1994. Issue: 1.
- [29] John C. McClelland, Robert J. Teather, and Audrey Girouard. Haptobend: shape-changing passive haptic feedback in virtual reality. In *Proceedings of the 5th Symposium on Spatial User Interaction*, SUI '17, pages 82–90, New York, NY, USA, October 2017. Association for Computing Machinery.
- [30] MQTT.org. Mqtt: The standard for iot messaging. <https://mqtt.org>, 2023. Accessed: 2024-06-22.

- [31] Henrique Melo Ribeiro. Material Changing Haptics for VR. July 2023. Accepted: 2023-08-09T23:12:56Z.
- [32] Aaron Raymond See, Jose Antonio G. Choco, and Kohila Chandramohan. Touch, Texture and Haptic Feedback: A Review on How We Feel the World around Us. *Applied Sciences*, 12(9):4686, January 2022. Number: 9 Publisher: Multidisciplinary Digital Publishing Institute.
- [33] Alexa F. Siu, Eric J. Gonzalez, Shenli Yuan, Jason B. Ginsberg, and Sean Follmer. shapeShift: 2D Spatial Manipulation and Self-Actuation of Tabletop Shape Displays for Tangible and Haptic Interaction. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 1–13, New York, NY, USA, April 2018. Association for Computing Machinery.
- [34] Carolin Stellmacher, Michael Bonfert, Ernst Kruijff, and Johannes Schöning. Triggermuscle: Exploring Weight Perception for Virtual Reality Through Adaptive Trigger Resistance in a Haptic VR Controller. *Frontiers in Virtual Reality*, 2, 2022.
- [35] Shan-Yuan Teng, Tzu-Sheng Kuo, Chi Wang, Chi-huan Chiang, Da-Yuan Huang, Liwei Chan, and Bing-Yu Chen. PuPoP: Pop-up Prop on Palm for Virtual Reality. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*, pages 5–17, Berlin Germany, October 2018. ACM.
- [36] Ultraleap. Leap motion controller, 2024. Accessed: 2024-06-23.
- [37] Eric Whitmire, Hrvoje Benko, Christian Holz, Eyal Ofek, and Mike Sinclair. Haptic Revolver: Touch, Shear, Texture, and Shape Rendering on a Reconfigurable Virtual Reality Controller. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 1–12, New York, NY, USA, April 2018. Association for Computing Machinery.
- [38] Shigeo Yoshida, Yuqian Sun, and Hideaki Kuzuoka. PoCoPo: Handheld Pin-based Shape Display for Haptic Rendering in Virtual Reality. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, pages 1–13, New York, NY, USA, April 2020. Association for Computing Machinery.
- [39] André Zenner and Antonio Krüger. Shifty: A Weight-Shifting Dynamic Passive Haptic Proxy to Enhance Object Perception in Virtual Reality. *IEEE Transactions on Visualization and Computer Graphics*, 23(4):1285–1294, April 2017. Conference Name: IEEE Transactions on Visualization and Computer Graphics.
- [40] André Zenner and Antonio Krüger. Drag: on: A Virtual Reality Controller Providing Haptic Feedback Based on Drag and Weight Shift. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, pages 1–12, New York, NY, USA, May 2019. Association for Computing Machinery.
- [41] Rui Zhang, A. Kunz, P. Lochmatter, and G. Kovacs. Dielectric Elastomer Spring Roll Actuators for a Portable Force Feedback Device. In *2006 14th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 347–353, March 2006. ISSN: 2324-7355.

- [42] Yiwei Zhao, Lawrence H. Kim, Ye Wang, Mathieu Le Goc, and Sean Follmer. Robotic Assembly of Haptic Proxy Objects for Tangible Interaction and Virtual Reality. In *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces, ISS '17*, pages 82–91, New York, NY, USA, October 2017. Association for Computing Machinery.

Appendix A

Mecanum wheels

Mecanum wheels, portrayed in Figure A.1 empower robots with instantaneous movement capabilities in any direction from any starting configuration. Their unique design might provide higher precision and speed in achieving specific positions by eliminating the need for the vehicle to rotate before moving and before trying to adjust the trajectory, a limitation present in traditional wheels. However, certain uncertainties might hold back the development of our device using these wheels, as they have low friction [21], which might be problematic for our use case.

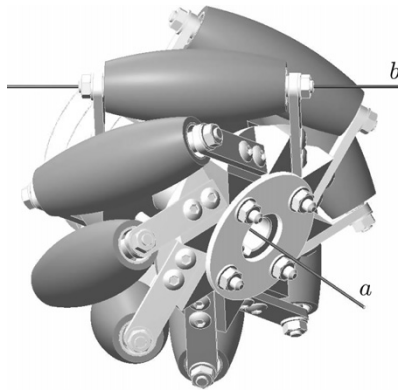


Figure A.1: Structure of a mecanum wheel [15].

As illustrated in Figure A.2, applying distinct torques to different wheels allows us to directly reach the same target position without additional rotation steps. The capability of unrestricted movement also opens up the possibility for future exploration into object avoidance.

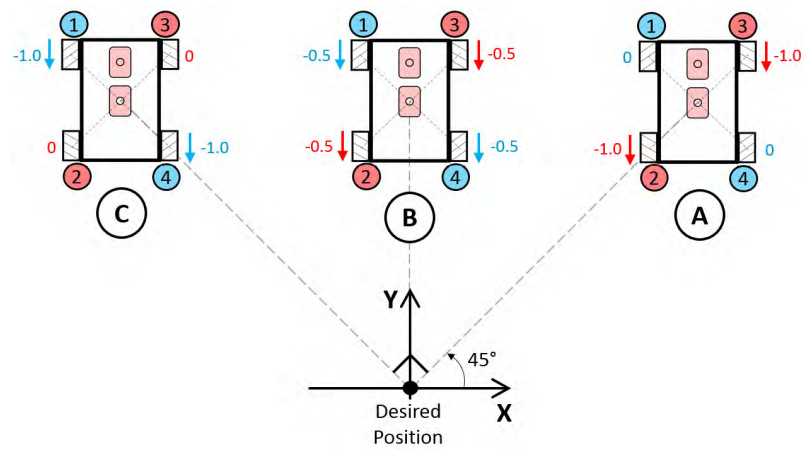


Figure A.2: How to achieve a desired position using mecanum wheels [21].