Bimanual haptic interaction with virtual environments

Anthony Talvas

Thèse

THÈSE INSA Rennes
sous le sceau de l'Université Européenne de Bretagne
pour obtenir le grade de
DOCTEUR DE L'INSA DE RENNES
Spécialité : Informatique

présentée par
Anthony Talvas
ÉCOLE DOCTORALE : MATISSE
LABORATOIRE : IRISA – UMR6074

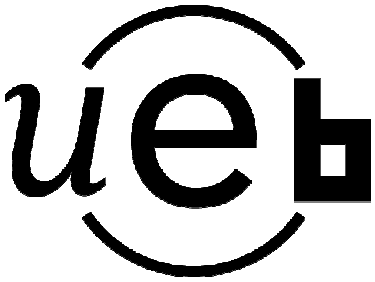
Bimanual Haptic Interaction with Virtual Environments

Thèse soutenue le 1er Décembre 2014
devant le jury composé de :

Bruno Arnaldi
Professeur, INSA Rennes / *Président*
François Faure
Professeur, Université Joseph Fourier, Grenoble / *Rapporteur*
Mehdi Ammi
Maître de conférences HDR, Université Paris-Sud / *Rapporteur*
Géry Casiez
Professeur, Université de Lille 1 / *Examinateur*
Paul G. Kry
Associate professor, McGill University, Montréal / *Examinateur*
Anatole Lécuyer
Directeur de recherche, Inria Rennes / *Directeur de thèse*
Maud Marchal
Maître de conférences, INSA Rennes / *Encadrante*

Interaction Haptique Bimanuelle avec des Environnements Virtuels

Anthony Talvas



En partenariat avec

Acknowledgements

First and foremost, I would like to thank my supervisors Maud Marchal and Anatole Lécuyer for having given me the opportunity to work on this PhD thesis, but also for their advice and support during the entirety of these three years. This has been a truly enriching experience for me on both a scientific and personal side, as well as an enjoyable time that went by faster than I had originally expected.

I would like to thank all the researchers who accepted to be part of my PhD commitee: Pr. François Faure, Dr. Mehdi Ammi, Pr. Géry Casiez, Dr. Paul G. Kry and Pr. Bruno Arnaldi. Thanks for their helpful comments on my manuscript, as well as their presence during my defense.

I would also like to thank the researchers with whom I have had the opportunity to collaborate during my thesis: Dr. Mathieu Emily, Dr. Christian Duriez and Pr. Miguel A. Otaduy. This PhD has also given me the chance to visit other research teams in the context of these collaborations, namely team SHACRA at Inria Lille and the Modeling and Virtual Reality Group at University Rey Juan Carlos in Madrid. My thanks go to the members of both of these teams for their welcome and help.

Speaking of welcoming teams, I also have to thank everyone in the Hybrid team for providing a cheerful environment to work in for three years. I want to especially mention those I have had the pleasure to share an office with: Pierre Gaucher, Fabien Danieau, Merwan Achibet and Adrien Girard. Their presence made this time in the team even more pleasant and enjoyable for me. My thanks also go to Loeiz Glondu and Gabriel Cirio who have given me invaluable help during my thesis.

Last but not least, I would like to thank my parents who have always been by my side, and whose support proved time and again invaluable. Also thanks to my friends from all over France and overseas, who have provided me with more encouragement than one could ever ask for. And of course, thanks to my dearest, without whom I would have never made it this far.

Contents

List of Figures

List of Tables

Introduction 1

This manuscript, entitled "Bimanual Haptic Interaction with Virtual Environments", presents research conducted in the context of Virtual Reality (VR). The goal of VR applications is to **allow a user to interact in real time with a Virtual Environment** (VE), as well as to **perceive that VE in the most immersive way possible**. This degree of immersion is achieved by substituting the stimuli provided by the real environment around the user for computer-generated stimuli reflecting the presence of the user inside the VE. Common applications of VR technologies include but are not limited to entertainment, education, training, computer-aided design, and medicine.

To this day, VR applications have most commonly used the visual and auditive modalities as ways to provide feedback from the VE to the user. This may prove sufficient to provide a sufficient level of immersion when the experience does not require direct interaction with the VE, such as in virtual tours. However, these sensorial modalities alone may fall short when actual interaction between the user's body and virtual objects is involved. Indeed, when we interact directly with objects in real life, a third sense is heavily involved in the perception of that interaction: the **haptic sense**, which encompasses two complementary senses. First, the tactile sense operates at the skin level, and allows to feel surfaces and textures. Secondly, the proprioceptive or kinesthetic sense is mediated by the inner ear as well as receptors inside the muscles and tendons, and is related to the perception of balance and posture of the different parts of the body relative to each other. When our hands interact with objects, local forces of contact and texture of the object are perceived through the tactile sense, while overall shape and properties like elasticity are perceived through the kinesthetic sense.

Due to its importance for interactive applications, the haptic modality has gained more and more attention over the last decades in VR applications. The inclusion of haptics in VR was made possible by **haptic devices**, which are human-computer interfaces designed to **detect a user's movements and stimulate the haptic sense**. Tactile interfaces stimulate the skin, mostly through the fingertips which are among the most sensitive parts on the entire body. Kinesthetic devices display net forces, either through a single effector like a stylus, or directly to the fingers (and potentially the palm), for instance using exoskeletons. These devices are coupled to virtual representations of the user inside the VE, also referred to as virtual proxies, which reproduce the user's motions and send the interaction forces back to the haptic device. In an ideal scenario, users would be able to interact with virtual objects directly with their hand, by grasping and moving them around, or doing precise manipulation tasks with them. Even more ideally, both hands could be involved in these interactions.

In our daily lives, **we commonly use both of our hands to perform all sorts of tasks**. Examples of this include holding a bottle with one hand while opening it with the other, holding a steering wheel while changing gears, keeping a nail straight while hammering it, or holding a fruit while cutting it. A fair amount of these tasks are performed two-handedly so naturally that we sometimes do not even notice that we use

two hands in the process. Part of why two-handed interaction is so common is because of specificities of the bimanual haptic system, such as different sensitivities between the dominant and non-dominant hand, as well as transfers of haptic information between both hands. In contrast, when it comes to haptic interaction with virtual environments, until recently the interaction happened mostly through one hand only, generally the one referred to as the dominant hand. Considering the importance of using two hands in real life, unimanual haptic interaction can prove less efficient and immersive for a certain number of interactive tasks in VR. This raises the need to **better integrate the use of two hands in haptics**.

Bimanual haptics, or *two-handed haptics*, refers to **haptic interaction through both hands of the same person**. While this field may bear similarities with other fields such as *multi-touch* or *multi-finger*, it differs in that the latter do not necessarily consider interaction through both hands. Similarly, while the field of *multi-user* interaction also involves multiple hands, it does not consider the interactions between both hands of the same user at a sensorimotor and cognitive level. Although *bimanual haptics* is a generic term for anything in the field of haptics that involves both hands, in this manuscript it is mostly used to refer more specifically to two-handed computer haptics, as it is the main focus of this thesis.

1.1 Challenges of Two-handed Haptic Interaction with Virtual Environments

When interacting in real time with a VE, a central question is what method to use to represent users inside the VE, and allow them to perform certain tasks. This is especially relevant with bimanual haptics, as the use of two hands widens significantly the range of tasks that can be performed. These methods, which combine hardware and software elements for achieving a specific set of tasks, are referred to as interaction techniques, or interaction metaphors (Figure 1.1).

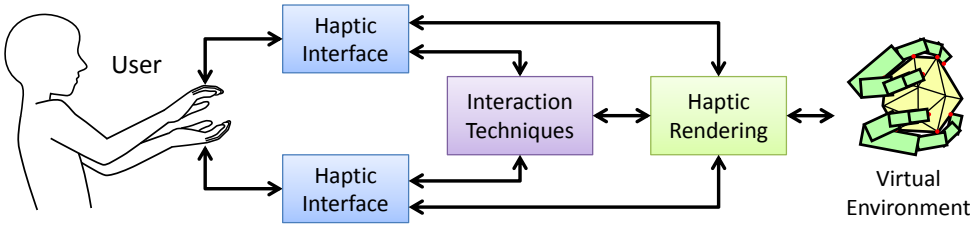


Figure 1.1 – Elements involved in the bimanual haptic interaction between a user and a virtual environment. The user (human layer) interacts with the haptic devices (hardware layer), which are coupled to the virtual environment through haptic rendering (software layer). Interaction techniques are linked to all of these elements and define how the user will be able to perform a given set of tasks.

A first set of tasks for which interaction techniques are required are haptic exploration tasks. Indeed, haptic devices have a **limited range of motion**, which is usually insufficient to interact with anything beyond the arms' reach. Additionally, bimanual devices may suffer from additional issues, as bimanual kinesthetic devices are usually made of robot arms that may collide with each other under certain circumstances, further reducing their range of motion. There is thus a need for interaction techniques to **extend the workspaces** of these devices, and allow the exploration or larger VEs. Additionally, such techniques should not rely on specific hand gestures, as a common scenario in bimanual

interaction would involve **navigating through a VE while holding objects** with one or two hands.

There are also two main interaction metaphors for the haptic manipulation of virtual objects, each having their own challenges. The most straightforward approach for interacting with virtual objects is the use of **virtual hands**, allowing direct interaction in a natural manner. Such models, however, rely on specific hardware for receiving hand configurations as well as sending forces to the palm and fingers of both hands, which can be both expensive and limited in terms of degrees of freedom. Moreover, when interacting with physically-based environments, virtual hands may also not be desirable due to the **high cost of simulating accurate hand models and resolving complex contact scenarios**. Another approach for manipulation is the use of simpler models, such as **rigid virtual proxies** representing a given tool that will be used to interact with the VE. Such models are easy to implement and fast to compute, but are often **limited to a single, specific task**, and may not be as immersive as virtual hands.

There are additional challenges in the field of bimanual haptics that are related to the haptic sense itself. The human haptic system is characterized by a high temporal and spatial sensitivity, a notable example being the finger pads [Srinivasan, 1995]. In order to comply with the high temporal resolution of the haptic system, **high haptic rendering rates are required on the hardware and software side**. This is required for instance to display forces resulting from short interactions with virtual objects (*e.g.* tapping an object) or sudden discontinuities in force (*e.g.* entering in contact with a rigid object).

Moreover, the common use of physically-based simulation within VEs leads to additional requirements that should be ideally met when interacting with such environments. Firstly, **high physical simulation rates** should be achieved, in order to handle short contacts and reduce latency with sudden force discontinuities. Moreover, smaller time steps help reducing numerical errors and may lead to a more stable simulation, avoiding artifacts that could be felt by the user. Secondly, **accurate contact models** should be used, to satisfy the fine spatial resolution on the hands and fingertips. This notably means resolving all penetrations between objects without inducing a certain springiness. Accurate friction models are also important to provide realistic haptic feedback to the user's fingertips, and for stable grasping of virtual objects. Finally, **realistic models for the physical behavior of the virtual proxies** should be used. Rigid models can be satisfactory when simple tools are used as proxies. However, when virtual hands are used, the deformation of finger pads should be simulated as well, as it softens the contact between fingers and objects, which stabilizes grasping and modulates the frictional behavior during contact.

Even when just considering unimanual haptics, these two requirements already antagonize each other strongly: more accurate physical models have the downside of being also more expensive, which clashes with the need for high simulation rates (Figure 1.2). Notably, the human hand has many joints each with their own articular limits, thus articulated hand models have to include many constraints to reflect these properties. This leads to more complex systems to solve numerically, as a contact happening on a fingertip can have repercussions all the way up to the wrist, and subsequently on the other fingers as well. Also, taking into account the finger pad deformation under contact requires simulating the deformation of the fingers themselves, which is in itself costly.

Having two hands in the simulation further adds to this complexity, by doubling the number of virtual proxies involved, and as a result the number of contacts generated through interaction with these proxies. When the two hands are interacting each with a separate virtual object, the resulting system to solve is simply the addition of two

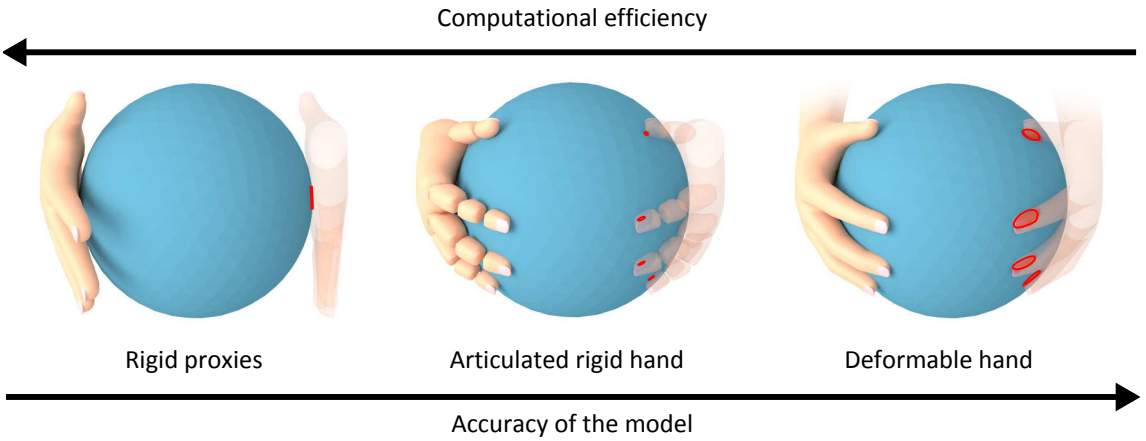


Figure 1.2 – Tradeoffs between realism and computational efficiency of different simulated models for interacting with virtual objects. The contact surfaces generated by the different models are highlighted in red on the left hand.

unimanual systems. However, if both hands are manipulating the same object, then the complexity increases even more, as the number of interconnected constraints increases. For instance, when two hands grasp firmly an object, the motion of one finger may affect all other fingers of both hands. Bimanual interaction thus leads to systems that are more computationally expensive to solve, with potential numerical issues notably when a high number of constraints affects a small number of degrees of freedom like when deformable fingers manipulate a rigid object [Miguel and Otaduy, 2011].

1.2 Thesis Objectives

This thesis focuses on enhancing several aspects of bimanual haptic interaction, which can be split into two main categories. Firstly, 3D interaction within VEs with two haptic devices is considered. Secondly, we consider the tradeoffs between the computational efficiency of bimanual haptic methods required for high simulation rates, and the realism of these methods required for natural and convincing interaction. Within this second category, two objectives are considered: improving the interactive capabilities of simpler models, and improving the computational efficiency of more complex models, such as finger pad contacts.

1.2.1 Improving Bimanual Haptic Interaction with Virtual Environments

Kinesthetic haptic interfaces tend to have limited workspaces, the most common desktop devices having ranges of motion that fit the amplitude of wrist motions, and overall most devices bearing workspaces no larger than the reach of human arms. This is problematic in most cases, as VEs tend to be much larger than these boundaries. Additional methods are thus required to allow navigation within VEs and interaction with larger objects. For instance, navigation tasks can be assigned to other body parts like the feet, which is a fairly obvious option as feet are naturally used to navigate, but this requires additional hardware that is not necessarily easily available. Exploration tasks can thus be assigned to the hands themselves, which then raises the need for interaction techniques that allow motions of the user within a VE while not hindering manipulation of virtual objects with one or two hands during these motions (Figure 1.3).

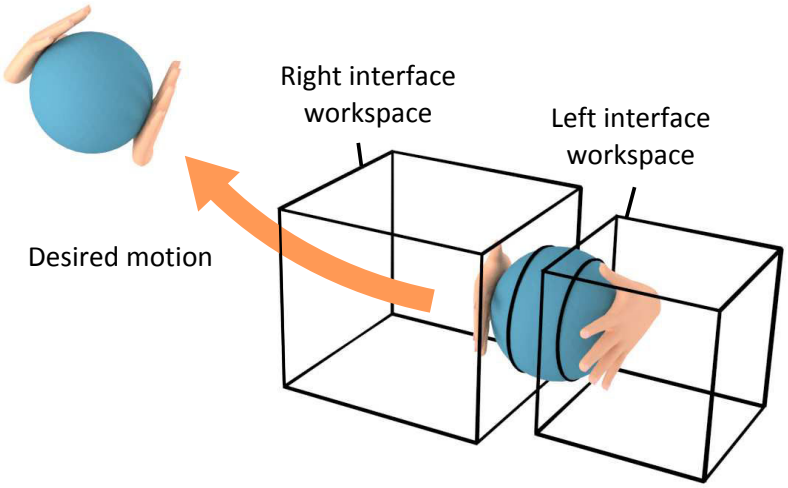


Figure 1.3 – First objective: achieving simultaneous exploration and manipulation of virtual objects with single-point haptic interfaces that have limited and potentially different workspaces.

This is especially true with rigid proxies, which are common due to the availability of haptic devices with single effectors, and as they allow to reduce the complexity of virtual proxies to simple unarticulated rigid objects. Bimanual manipulation of virtual objects with rigid proxies can be unpractical, as most of the time only two contacts can be generated with two proxies. This makes grasps more unstable than with hand models that stabilize the grasp with the fingers. Interaction techniques are thus needed to assist the user in the manipulation of virtual objects using those proxies, as well as for moving these objects within the VE without unintentionally dropping them.

1.2.2 Improving the Grasping of Virtual Objects

When grasping objects with two hands, whether through full virtual hands or a couple of phalanges, an important aspect is the rendering of the contact surfaces between finger pads and manipulated objects. These surfaces stabilize the grasping of objects by constraining the relative rotation of the objects through torsional friction. Rigid models fail to provide these surfaces due to their inability to deform under contact to match the shape of the object being touched. Soft models, on the other side, do simulate that deformation, but often require costly methods to simulate realistic deformation of the fingers. This raises the need for methods to render the contact surfaces of finger pads more efficiently (Figure 1.4).

Whichever method is used to simulate the deformation of fingers under contact, the resulting contact surfaces are classically stored as sets of multiple contact points. When solving these contacts with accurate methods such as constraint-based methods, this high number of contacts can prove complex to solve, notably when considering friction which adds more constraints per contact. It is especially true in bimanual scenarios, as more fingers produce more contacts with the same object. It is thus of interest to reduce the number of constraints to improve the computational efficiency of contact resolution. However, the multitude of contacts provides a good sampling of the contact surface which is required to simulate torsional friction over this surface by adding the individual contributions of each friction frame. There is thus a double objective in this topic: reducing the number of constraints, and at the same time, retaining rotational friction in a similar fashion to what a fine sampling would provide (Figure 1.5).

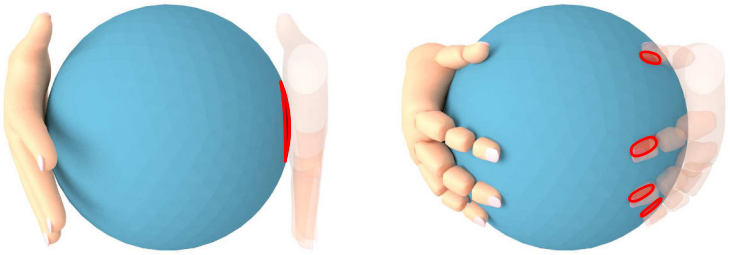


Figure 1.4 – Second objective: allowing the simulation of finger pad-like contact surfaces from rigid proxies and rigid articulated hand models. The desired contact surfaces are shown in red on the left hand.

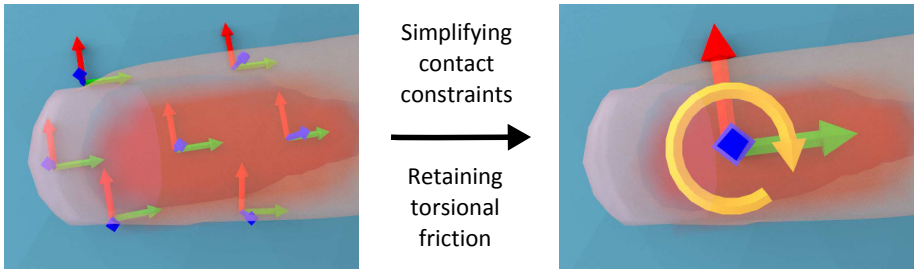


Figure 1.5 – Third objective: reducing numerous contact and friction constraints into a minimal set of constraints that still keep torsional friction over the contact surface. Contacts are represented by the sets of arrows, corresponding to separation and friction frames.

1.3 Approach and Contributions

This manuscript presents the research carried out to answer the previously described objectives. First, **Chapter 1** presents the related work on bimanual haptics, which is divided into four main parts. The current knowledge on the perceptual, motor and cognitive aspects of the bimanual haptic system is first described. Then, the available bimanual haptic hardware is presented, leading to the existing software solutions to allow the haptic rendering of forces provided by the VE and physically-based simulation of VEs. Finally, interaction techniques that allow users to perform specific bimanual tasks within a VE are presented.

The following chapters describe the proposed contributions to address the three objectives. The order of the chapters goes from techniques that enhance interaction with simple virtual proxies to methods that improve the efficiency of more complex models.

Chapter 2 addresses the issues with bimanual haptic exploration and manipulation in VEs using single-point haptic devices, and presents a set of novel interactive techniques adapted to two-handed manipulation of objects with simple rigid proxies. We first propose the *double bubble* technique for **bimanual haptic exploration of VEs through a combination of position and rate control**, notably with different devices for each hand. A viewport adaptation keeps both virtual proxies always visible in larger VEs. We also present a bimanual haptic manipulation technique which facilitates the grasping of virtual objects with simple rigid proxies: the *magnetic pinch*, which adds an **attractive link between proxies and grasped object to prevent undesired drops**. The *joint control*, which forces common control modes and Control/Display ratios for two proxies grasping

an object, allows **easier navigation** in the VE with the *double bubble* **during grasping**. A **user evaluation** was conducted to assess the **efficiency of these techniques** for pick-and-place tasks, by comparing the *double bubble* to a state-of-the-art technique for extending the workspaces, and by measuring the benefits of the *joint control* and *magnetic pinch*.

In **Chapter 3**, we focus on improving the computation of contact surfaces similar to human finger pads. We propose a method called the *God-finger* method to **simulate a finger pad-like contact area from a single contact point or small set of contact points** generated by point or rigid proxies. The method uses the local geometry of the object at the contact site as well as the normal and tangential force applied to it in order to provide additional contact points that emulate the deformation of a finger pad under contact. It improves both the unimanual and bimanual manipulation of virtual objects by constraining their rotation in a similar manner to actual finger pads, while being more computationally efficient than deformable body simulation methods. The method is adapted to both interaction with 3 degree-of-freedom devices using simple point proxies, and 6 degree-of-freedom interfaces with more complex rigid proxies. It is also well suited for interaction with rigid and deformable objects, including with rough surfaces. A visual rendering method provides feedback to the user on the shape of the contact surface and the amount of pressure applied.

Chapter 4 proposes novel contributions for improving the efficiency of finger pad contact resolution. We present an approach based on novel aggregate constraints for simulating dexterous grasping using soft fingers. It aims at improving the computation of contact mechanics when many contact points are involved, by **aggregating the multiple contact constraints into a minimal set of constraints**. We also introduce a method for **non-uniform pressure distribution over the contact surface**, to adapt the response when touching sharp edges. We use the **Coulomb-Contensou friction model to efficiently simulate tangential and torsional friction**. The approach is evaluated with an articulated hand model with deformable phalanges, that can be coupled with a data glove for real time interaction. We show through different use cases that our aggregate constraint formulation is well-suited for simulating dexterous manipulation of virtual objects through soft fingers, and efficiently reduces the computation time of constraint resolution.

Finally, **Chapter 5** concludes on the work presented in this manuscript, and discusses short and long term perspectives on the topic of bimanual haptics with VEs.

Related Work on Bimanual Haptics 2

Contents

The field of bimanual haptics is an emerging domain that focuses on haptic interaction through both hands of the same person. An increasing number of studies has been conducted on this topic in the last couple of decades, notably thanks to the increase in computational power available and the decrease of haptic device costs. This chapter reviews the current state of studies on bimanual haptics and discusses the current perspectives in this field.

It is structured so as to follow the different steps that allow a user to perform a bimanual task in a VE (Figure 1.1). The sensorimotor aspects of bimanual haptics are a central part this field as their clear specificities should provide guidelines for the design of hardware, software and interactive techniques. Then, the choice of haptic devices on the hardware side influences the physical models and haptic rendering methods that will be used on the software side, and subsequently, the interaction techniques used to allow the user to perform a certain set of tasks.

First, we describe in Section 2.1 the perceptive, motor and cognitive aspects of the human bimanual haptic system. We then present the hardware available for both single-point and multi-finger bimanual haptics in Section 2.2. The software aspects of bimanual haptic interaction is then described in Section 2.3, with physical simulation and haptic rendering methods. Finally, interaction techniques for navigation in a VE and manipulation of virtual objects are presented in Section 2.4.

2.1 Human Bimanual Haptic System

Working with two hands is more than simply "using twice one hand". First of all, there are clear differences between the way both hands function, notably relative to each other, as we have a dominant hand (DH) and a non-dominant hand (NDH). Then, the use of two hands acting in an integrated way allows to perform tasks that could hardly be done with a single hand, or with hands working on independent subtasks that involve no interaction between both hands. Finally, there are additional haptic cues provided by the joint use of two hands.

This section deals with the cognitive and motor aspects of the use of two hands. Key observations on bimanual haptic perception, motor action, and cognition are reviewed, before discussing some benefits of bimanuality over unimanuality.

2.1.1 Bimanual Haptic Perception

Bimanual actions involve a major form of haptic feedback: the ability to locate our hands relatively to each other. While the presence of visual feedback causes the visual sense to be predominantly used over the haptic sense, this bimanual proprioception fully comes in action when this visual feedback is absent, inconsistent or incomplete [Balakrishnan and Hinckley, 1999, Veit et al., 2008]. A notable example is that of blind people, for whom two-handed exploration represents a valuable means of estimating distances with their environment. Several studies showed that it is easier to follow the relative position between the hands rather than their individual position relatively to a 3D space [Hinckley et al., 1997a, 1998, Balakrishnan and Hinckley, 1999, Veit et al., 2008].

Some studies have also showed a certain level of specialization of the hands when it comes to haptic perception. It was suggested that the NDH has increased proprioceptive sensitivity, whether for right-handers [Goble and Brown, 2008] or left-handers [Goble et al., 2009]. This fact may be correlated with studies suggesting a specialization of the non-dominant hemisphere for proprioception processing [Leonard and Milner, 1991, Rains and Milner, 1994]. Brain studies also suggest that the parietal lobe is involved in the integration of sensorimotor information, and among other things for spatial perception [Wolpert et al., 1998]. Unilateral damage to this lobe has been shown to lead to a neglect of stimuli on the contralateral side of the lesions [Vallar, 1998].

The integration of different stimuli between both hands was also studied. Concerning the haptic perception of curvature, while a similar discrimination threshold for unimanual and bimanual curvature perception has been reported [Sanders and Kappers, 2006, Squeri et al., 2012], a more recent study actually observed a lower threshold for bimanual exploration of large cylinders [Panday et al., 2013]. This latter study clearly showed that this better discrimination is entirely due to integration of curvature perception between the two hands, as further experiments proved that the position discrimination has similar thresholds with one or two hands in this scenario. The bimanual perception of stiffness

was shown to be more precise than unimanual stiffness perception, and that the bimanual perception was the result of the combination of both unimanual inputs in a statistically optimal manner [Plaisier and Ernst, 2012]. Ballesteros et al. [1997] reported a better perception of symmetric shapes with bimanual exploration than unimanual, but not for asymmetric shapes. They also did not observe any significant difference of perception between the DH and NDH.

2.1.2 Bimanual Motor Control

The mechanisms that link the sensation in both hands to the motor control of these hands mostly remain to be studied. Experiments involving reaching tasks with each hand suggested transfers of learning between both hands [Criscimagna-Hemminger et al., 2003, Harley and Prilutsky, 2012]. Transfers of haptic information were also observed, with trajectory information transferred from the NDH to the DH, and endpoint position information transferred from the DH to the NDH [Sainburg and Wang, 2002, Wang and Sainburg, 2007, Harley and Prilutsky, 2012]. Brain studies further suggest the existence of neural pathways that mediate the coordination of both hands during bimanual tasks [Carson, 2005].

A topic that has received considerable attention when it comes to two-handed interaction is the sequence of action during a bimanual task, for which the most common model is Guiard's Kinematic Chain model [Guiard, 1987]. It makes an analogy of the DH and the NDH as two motors assembled in series, based on three major principles:

1. The NDH is a frame of reference for the DH.
2. The NDH precedes the DH in action.
3. Both hands act at different temporal and spatial scales, the NDH acting at a coarser scale.

The Kinematic Chain model applies to many real life tasks, such as sewing, which involves both hands. Handwriting is another example, where the DH moves the pen relatively to the paper, which is itself moved by the NDH relatively to the table [Guiard, 1987]. It was demonstrated that this model can be used effectively for reasoning about precision manipulation tasks [Hinckley et al., 1997b], as well as for designing two-handed interaction techniques [Cutler et al., 1997, Mine et al., 1997]. Ullrich et al. [2011a] evaluated the first principle of the model (frame of reference concept) for interaction tasks with haptics through a virtual needle insertion task. The needle was manipulated by the DH and the goal was to insert in a certain target. The NDH could be assigned an asynchronous pointing task, having it touching another target close to the first one. This allowed the creation of a frame of reference between the DH and NDH, and was shown to reduce significantly the completion times of the task without affecting precision.

Aside from Guiard's model, another observed property of bimanual action is the degree of symmetry used when performing increasingly difficult tasks, such as tasks that are more cognitively demanding, that need to be executed faster, or more precisely. Bimanual tasks can be performed either symmetrically, in which case both hands will perform the same task (*e.g.* rope skipping), or asymmetrically, where both hands perform different tasks (*e.g.* striking a match). Additionally, they can be performed synchronously, meaning the motions of both hands are in phase (*e.g.* weight lifting), or asynchronously, with antiphase movements of both hands (*e.g.* rope climbing) [Guiard, 1987, Balakrishnan and Hinckley, 2000]. Hinckley et al. [1997b] suggested that, for easy tasks, there is little asymmetry of

the hands, while harder tasks, notably requiring more precision, induce a stronger specialization of the hands, meaning that they take more specific, non-interchangeable roles. Further studies on a symmetric bimanual tracking task showed that increasing difficulty, necessity of dividing attention or lack of visual integration lead to a more sequential way of performing tasks as well as a slightly decreased parallelism (defined as the ratio of error rates between DH and NDH) [Balakrishnan and Hinckley, 2000]. Ulinski et al. [2007] experimented on bimanual selection techniques and reported better accuracy with symmetric techniques, while suggesting that asymmetric techniques are more suitable for tasks that are more cognitively demanding or that last for longer periods of time.

2.1.3 Bimanual Cognition

An important cognitive aspect in bimanual interaction is the notion of *task integration*, *i.e.* the compilation at a cognitive level of different subtasks into a single task, which can happen at three different levels [Owen et al., 2005]. First, it can consist of visual integration, like for instance when scaling a rectangle by sliding its corners [Casalta et al., 1999]. Then, there is motor integration, when different subtasks are combined into a single gesture [Leganchuk et al., 1998]. Finally, there is conceptual integration, which causes the user not to think of an activity as a set of subtasks but rather as a single task, such as with a 3D cross-section visualization task [Hinckley et al., 1997a, 1998] (Figure 2.1). Such a task requires, unimanually, to shift between a "moving the view" task and a "moving the cross-section plane" task, thus creating a third "change states" extraneous subtask. Using two hands, on the contrary, integrates these tasks into a single "cut relative to view" meta-task, which is performed in a single gesture (Figure 2.1).

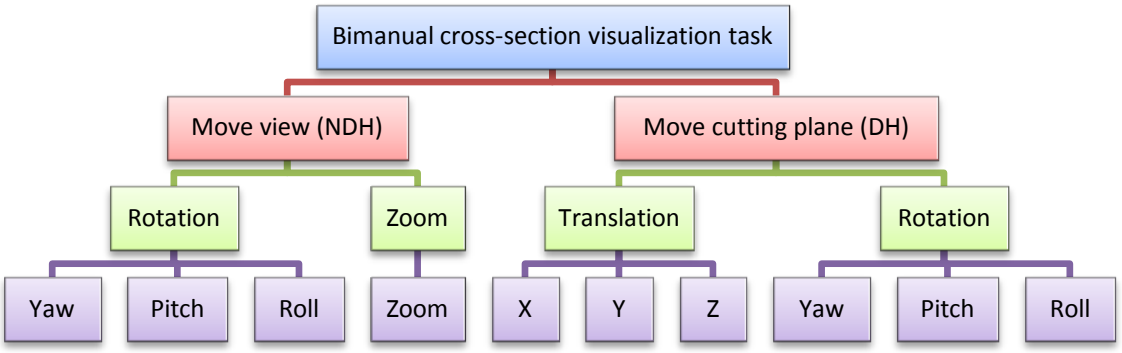


Figure 2.1 – Hierarchy of ten different subtasks making up a cross-section visualization task, that are integrated into a single bimanual task [Hinckley et al., 1997a].

It was also shown that bimanual interaction can improve cognitive efficiency. Hinckley et al. [1997a] performed an experiment in which participants had to align two virtual objects using either one or both hands, then try to reproduce the movement of the DH using only the haptic sense, without visual feedback. The results showed that unimanual performance was improved if the bimanual condition was used before, which was not observed the other way around. This indicates that using both hands changed the subjects' task-solving strategy at a cognitive level. Using two hands can also reduce the gap between novices and experts compared to the use of only one hand, as was shown with a navigation/selection task [Buxton and Myers, 1986]. Owen et al. [2005] hypothesized that part of the performance gain could be explained by the fact bimanual interaction leaves more room for epistemic actions, *i.e.* motor actions performed in order to improve

cognition, however the data obtained from their experiment was not fully conclusive.

2.1.4 Benefits of Bimanuality for Interaction

Most tasks operated naturally in real life can be naively thought as being unimanual, while they are in fact bimanual and often asymmetric [Guiard, 1987]. An example of this is writing, where the NDH has a role of orienting the paper for the DH to write on it. Some experiments showed that the use of two hands remains natural in human-computer interaction [Buxton and Myers, 1986, Cutler et al., 1997]. Thus, a first advantage of using two hands for haptic interaction is that it harnesses better our existing skills, that we use in everyday life.

Another benefit of bimanuality is that it increases task accuracy compared to the use of one hand through two-handed proprioceptive feedback, benefit which is more clearly visible when visual feedback is absent [Hinckley et al., 1998, Veit et al., 2008]. Moreover, tasks can be achieved faster when two hands are used [Buxton, 1988, Dillon et al., 1990, Cline, 2000, Owen et al., 2005]. This can be explained by the fact that two hands allow to realize tasks in less steps than with one hand, as illustrated with digital painting tasks, where several unimanual steps could be reduced to a single bimanual action [Buxton, 1988, Bier et al., 1993]. Also, two hands can be present at two separate points of action at the same time, which removes the time needed to switch positions between both, like for digital painting with the menus and workspace [Dillon et al., 1990].

However, there can also be negative effects of using two hands on cognitive efficiency in some cases. Notably, when the tasks applied by the two hands are not sufficiently integrated, an effect of "division of attention" occurs between the two tasks and performance may decrease significantly [Kantowitz, 1991]. Some known examples include the use of a mouse in each hand for independent tasks, which may lead to performances that do not exceed those of one-handed experiments [Dillon et al., 1990, Kabbash et al., 1994, Zeleznik et al., 1997]. For instance, the Toolglass metaphor [Bier et al., 1993], which integrates the selection of an operation and an operand by allowing users to click with their DH through buttons that are part of a see-through interface held by the NDH, performed better for digital painting than the use of two mice [Kabbash et al., 1994].

2.1.5 Discussion and Conclusion

A major element in bimanual haptic perception is our proprioception, which allows us to locate accurately both hands relatively to each other. Several studies have focused on the specialization of hands for haptic perception, and found increased proprioceptive sensitivity for the NDH [Goble and Brown, 2008, Goble et al., 2009], transfer of trajectory information from the NDH to the DH, and transfer of endpoint limb position information from the DH to the NDH [Harley and Prilutsky, 2012]. The integration of tasks between both hands appears to be a complex problem, as there are different possible integration schemes: visual [Casalta et al., 1999], motor [Leganchuk et al., 1998] and conceptual [Hinckley et al., 1997a, 1998].

The Kinematic Chain model implies a sequential and specialized role of both hands [Guiard, 1987], and was shown to be an efficient model for designing more natural and efficient bimanual interaction techniques [Hinckley et al., 1997b, Cutler et al., 1997, Mine et al., 1997, Ullrich et al., 2011a]. Furthermore, it was observed that tasks tend to be performed in a more asymmetric manner as they get more difficult [Hinckley et al., 1997b, Balakrishnan and Hinckley, 2000]. Two-handed interaction also allows better integration of

different subtasks at a cognitive level, making seemingly complex tasks more simple [Owen et al., 2005, Casalta et al., 1999, Buxton, 1986, Leganchuk et al., 1998, Hinckley et al., 1997a]. However, this is only observed when the attention is not too divided, otherwise tasks may actually become more cognitively demanding [Kantowitz, 1991, Dillon et al., 1990, Kabbash et al., 1994, Zeleznik et al., 1997, Leganchuk et al., 1998].

The use of two hands is a common occurence in our daily lives, that was shown to be applicable to the domain of human-computer interaction and more specifically haptics. Bimanuality brings a certain number of benefits, ranging from better accuracy [Hinckley et al., 1998, Veit et al., 2008] to faster realization of tasks [Buxton, 1988, Dillon et al., 1990, Cline, 2000, Owen et al., 2005]. Overall, taking into account these known elements about two-handed perception and action could greatly improve the efficiency of designed bimanual haptic hardware, software and interactive techniques.

2.2 Bimanual Haptic Hardware

In order to interact through both hands with a virtual or remote environment with haptic feedback, the first requirement is hardware that features both tracking of the position of two hands, possibly of the palms and fingers for more realistic interaction, and display of the computed feedback on these hands, being either kinesthetic, tactile, or both. Haptic hardware suited for bimanual interaction can be divided into two major categories. Single-point interfaces track the position and provide force feedback to a single effector for each hand. They can be grounded, meaning that their base is fixed on a static element like a desktop, or mobile, in which case they are mounted on an element that can move around. Multi-finger interfaces track and provide kinesthetic or tactile feedback to multiple fingers per hand, and can be either grounded, or body-based, such as with haptic gloves for instance. This section focuses on these interfaces, by going through the aforementioned categories: single-point grounded, single-point mobile, multi-finger body-based and multi-finger grounded interfaces.

2.2.1 Single-Point Grounded Interfaces

Haptic interfaces with a single end effector per hand, which we refer to as *single-point interfaces*, have been widely used for bimanual interaction, notably due to their abundance. A notable example is the Phantom series of devices, which is a originally unimanual commercial model that was also used in a fair amount of bimanual haptic studies. There are also other devices that were shown to be fit for bimanual interaction, some of which having an additional degree of freedom (DOF) for grasping.

2.2.1.1 Phantom devices

The Phantom family of devices (Geomagic, Durham, North Carolina, USA) has been used in several bimanual haptic studies. In all of these studies, these interfaces are underactuated, meaning that they have more DOFs in input (6DOF) than in output (3 translational DOF). However, 6DOF feedback versions of these devices do exist. The workspace includes various sizes depending on the model used, and can range from $16 \times 12 \times 7cm$ to $84 \times 58 \times 41cm$.

Dual Phantom devices were notably used in medical contexts, for instance with a simulator of ultrasound-guided needle puncture made with two PHANToM Omni devices [Vidal et al., 2008], or a *da Vinci* Surgical System simulator developed with the same

interfaces but improved with gripper devices [Sun et al., 2007]. Ullrich and Kuhlen used these devices as well, but replacing the stylus of one Omni device with a palpation pad, to simulate palpation interaction alongside needle insertion in a medical training simulator [Ullrich and Kuhlen, 2012] (Figure 2.2a).

Another example is a 3D mesh manipulation software using two Phantom Desktop devices in conjunction with stereo vision [Faeth et al., 2008] (Figure 2.2b). von der Heyde and HÃ¤ger-ross [1998] also used two Phantom 3.0 interfaces to study one-handed grip tasks and two-handed pointing tasks as well as part of a virtual laboratory for studying human behavior in complex tasks [Pelz et al., 1999]. Finally, Kron et al. [2004] used Phantom devices for teleoperation, but using a Phantom Desktop device for the NDH, and a Phantom 1.5, which features a larger workspace, for the DH.

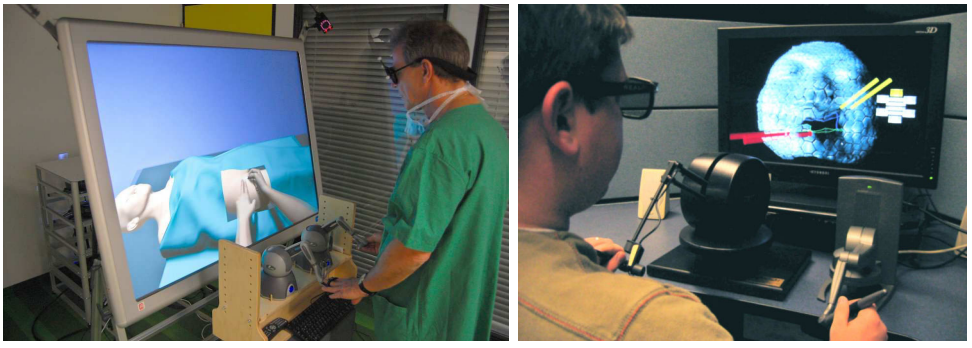


Figure 2.2 – Bimanual haptic applications with Phantom devices: medical training simulator [Ullrich and Kuhlen, 2012] and 3D mesh manipulation software [Faeth et al., 2008].

2.2.1.2 Other devices

While the Phantom devices are commonplace and widely used, other devices with single effectors were shown to be suitable for bimanual studies as well. The Virtuose 6D (Haption SA, Soulgé-sur-Ouette, France) features 6DOF in both input and output, as well as a workspace fitting the movement of human arms. It was notably used for a virtual snap-in task between a deformable clip and quasi-rigid pipe [Duriez et al., 2006], as well as for a virtual crepe preparation simulator, through the use of a virtual bowl in one hand and pan in the other hand for the manipulation of fluids [Cirio et al., 2011] (Figure 2.3a).

Similarly, the VISHARD10 interface is a robot arm that features 6DOF in both input and output, with 4 extra DOF to avoid arm collision and increase the size of the workspace up to a cylindrical workspace of 1.7 m of diameter and 0.6 m of height [Ueberle et al., 2004]. The GRAB device [Bergamasco et al., 2006] was built for either two-finger single-hand or two-handed one-finger interaction, with a focus on having a workspace large enough for two-handed manipulation with minimal encumbrance, while displaying forces representative of hand manipulation in unstructured environments. The fingertips fit into thimbles which serve as underactuated end effectors, with 6DOF sensing and 3DOF force display.

Some single-point devices were also designed with bimanual operation in mind, providing left-hand and right-hand versions of the interfaces. The omega.6 (Force Dimension, Nyon, Switzerland) is an underactuated device with similar workspace to the Phantom with increased force feedback. The Freedom 6S (MPB Technologies, Montréal, Quebec) allows 6DOF haptic interaction in a workspace fit to human forearms [Hayward et al., 1997, Demers et al., 1998]. A bimanual haptic interface was developed by the German Aerospace Center (DLR) using two light-weight robot arms attached to the same column

[Hulin et al., 2008] (Figure 2.3b). Each arm features 6DOF with an extra DOF that allows to avoid collisions between the two arms, the workspace provided by the combination of both being similar to that of human arms.

Figure 2.3 – Examples of single-point bimanual haptic interfaces: Virtuose 6D [Cirio et al., 2011] and robot arm-based interface from DLR [Hulin et al., 2008].

2.2.1.3 Devices with grip-force interfaces

Some devices, while still only providing input and output on a single point, have added handles that allow to sense the grasping force applied on the interface by the user, and to provide resistive feedback to the user's grip. The omega.7 (Force Dimension, Nyon, Switzerland) is a version of the omega.6 with that capability, thus is also an underactuated device, while the sigma.7 has 6DOF in both sensing and actuation (Figure 2.4a). The omega.7 notably showed its bimanual capabilities during an experiment involving two-armed surgical robot manipulation in microgravity. Similarly, the Freedom 7S (MPB Technologies, Montréal, Quebec) is an extension of the Freedom 6S with scissors handles.

The DLR bimanual haptic interface allows the use of different kinds of handles: aside from a magnetic clutch for leaving the fingers free and a joystick handle featuring a mini-joystick, switch and buttons, it can also feature a grip-force interface as well [Hulin et al., 2008].

The SPIDAR-G&G [Murayama et al., 2004] consists of two SPIDAR-G devices [Kim et al., 2003a], which are string-based unimanual devices with 6DOF for both motion and force feedback, and an additional DOF provided by a spherical element made of two hemispheres that a user can grasp to reproduce the same action in the virtual environment (Figure 2.4b). The workspace of each interface is a cubic frame with 20 cm of side length, and the two interfaces were seperated by 40 cm in the SPIDAR G&G prototype, thus avoiding interface collision issues.

The bimanual surgical interface from Immersion Corp. [Waldron and Tollon, 2003] is a special case in that it is specifically designed for simulation of minimally invasive surgical procedures, and as such its design is close to that of the actual surgical tools with 5DOF for each manipulator (Figure 2.4c).

2.2.2 Single-Point Mobile Interfaces

Mobile devices, by using the mobility of a robot carrying the haptic arms, can obtain almost infinite planar workspaces. An example is the Mobile Haptic Grasper [Formaglio et al., 2006], which is made of two grounded Phantom devices mounted on a mobile robot (Figure 2.5a). This device being limited to 3DOF in actuation, Peer and Buss used the VISHARD7 device [Peer and Buss, 2008], which features full 6DOF capabilities and is

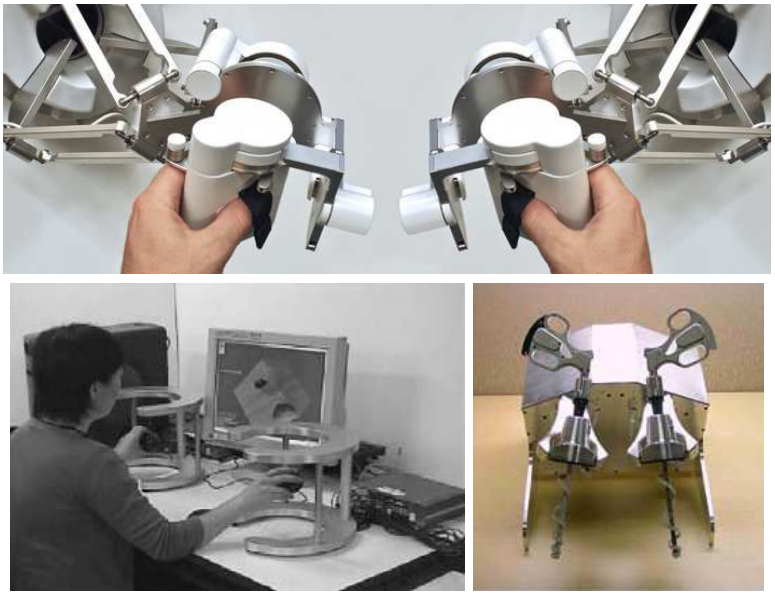


Figure 2.4 – Examples of single-point bimanual haptic interfaces with grip-force interfaces: sigma.7 (Force Dimension), SPIDAR-G&G [Murayama et al., 2004] and bimanual surgical interface [Waldron and Tollon, 2003].

mountable on a mobile robot unlike the VISHARD10 [Buss et al., 2009], thus leading to a mobile bimanual interface similar to the Mobile Haptic Grasper but with increased actuation (Figure 2.5b). The latter also features a local workspace around the robot that fits the reach of human arms, allowing fast movements of higher amplitude than those permitted by the Phantom devices.

Figure 2.5 – Examples of single-point mobile bimanual haptic interfaces: Mobile Haptic Grasper [Formaglio et al., 2006] and VISHARD7-based mobile haptic interface [Peer and Buss, 2008].

2.2.3 Multi-Finger Body-Based Interfaces

Multi-finger interfaces can take the form of gloves, that provide either kinesthetic or tactile feedback to the fingers while not hindering the motions of the user like grounded devices. For kinesthetic feedback, two commercial devices from CyberGlove Systems provide the

necessary components for both sensing of all fingers the position of each phalanx of all fingers as well as the palm and wrist with the CyberGlove, and resistive force feedback to each of the fingers with the CyberGrasp, which are both available in left-hand and right-hand versions (Figure 2.6a). For tactile display, the CyberTouch (CyberGlove Systems LLC, San Jose, California, USA) is the vibrotactile counterpart of the CyberGrasp, to be used conjointly with the CyberGlove as well.

For tactile display, the GhostGlove is a glove that displays 2DoF tactile feedback on the palm and fingers through pulleys and motors [Minamizawa et al., 2008a] (Figure 2.6b). Using a belt attached to dual motors, the device can display both vertical and shearing forces on each of the end effectors, however it is a pure display interface and as such does not provide sensing of the position and orientation of the palm and fingers by itself. The device was tested in a bimanual scenario of recognition of a virtual solid, leading to good recognition of the size of the object, though approximately 2 cm smaller for sizes varying between 19 and 28 cm [Minamizawa et al., 2008b].

The TactTip is a tactile module that can be attached to the fingertips with kinesthetic gloves such as the CyberGlove/CyberGrasp combination, providing both vibrotactile feedback and temperature feedback to all fingers [Kron and Schmidt, 2003]. While not tested in a bimanual context, it can be easily imagined that such modules could be used with two kinesthetic gloves at the same time. Hulin et al. [2011] noted a few tactile devices which could be used for bimanual interaction. The A.R.T. tactile finger feedback device uses optical tracking for finger position sensing and 3 wires around the fingertips that can contract and vibrate for tactile feedback [Scheibe et al., 2007, Moehring and Froehlich, 2011] (Figure 2.6c). The DLR vibrotactile feedback device provides tactile feedback to the entire forearm through two rings of vibration motors [Hulin et al., 2007]. Finally, the DLR VibroTac also provides vibrotactile feedback to the arms using an array of vibration motors, and can be attached to either the upper arm, forearm or wrist [Schatzle et al., 2010].

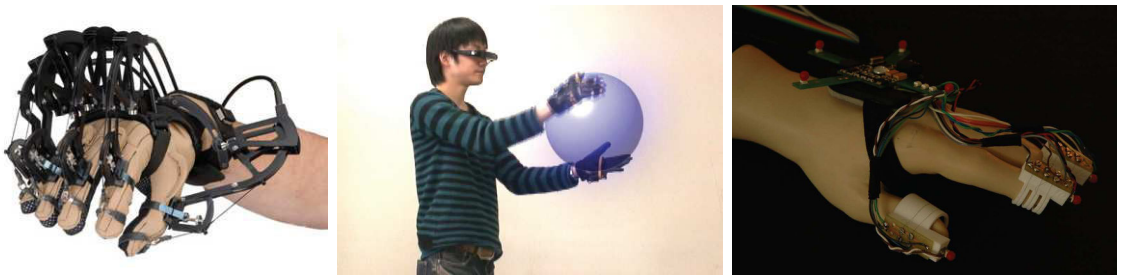


Figure 2.6 – Examples of multi-finger body-based bimanual haptic interfaces: CyberGlove with CyberGrasp (CyberGlove Systems) (kinesthetic), GhostGlove [Minamizawa et al., 2008a] (motor-based tactile) and A.R.T. tactile finger feedback device [Scheibe et al., 2007] (vibrotactile).

2.2.4 Multi-Finger Grounded Interfaces

Similarly to single-point interaction, Phantom devices were also used for bimanual multi-fingered interaction. Barbagli et al. [2003] developed a system for two-handed two-fingered grasping of virtual objects by adding motors and sensors to Phantom devices. A force reflecting gripper measures the relative position of the thumb and index, and gives force feedback when an object is grasped.

Three members of the SPIDAR family of string-based haptic display [Sato, 2002] were specifically designed for bimanual multi-fingered interaction. The first is Both-Hands

SPIDAR [Ishii et al., 1994] which is made of two SPIDAR-II devices [Ishii and Sato, 1994] combined into the same frame, with non-overlapping workspaces. The SPIDAR-II allows to grasp virtual objects using the index and thumb of one hand, and as such the Both-Hands SPIDAR device bears the same capabilities with two hands. Another SPIDAR system that provides the same kind of interaction is SPIDAR-8 [Walairacht et al., 2001] which is built on the same concept as Both-Hands SPIDAR but allowing interaction with four fingers per hand instead of two. The strings, however, could cause some interference with the user's fingers with certain hand poses, notably when rotating the hand as a whole. The SPIDAR-10 adds one rotary frame per hand to reduce those interference effects by rotating the frames following the hand rotations, also adding the remaining fingertips to the process for interaction with all 10 fingers [Maruyama et al., 2013] (Figure 2.7a). All of these systems have 3 degrees of freedom (DOF) in input and output for each finger.

The Bimanual HIRO system [Yoshikawa et al., 2009] (Figure 2.7b) was developed using two HIRO III devices [Endo et al., 2009], which are robotic arms similar to the DLR interface, with the difference that they are connected to all five fingertips of both hands of the user. The interface thus provides 3DOF in input and output for each of the fingertips, for a workspace covering the space of a desktop. The MasterFinger-2 was designed for two-fingered haptic interaction with the thumb and index, and its usability for two-handed manipulation was shown by using two of them jointly [Garcia-Robledo et al., 2009]. The MasterFinger-2 is underactuated, bearing 6DOF in sensing and 3DOF in actuation, with an additionnal DOF that helps increasing the workspace. The workspace for each finger is around 40 cm wide, but there should not be any interface collision if the bases are separated by more than 50 cm.

The Haptic Workstation from Immersion bears all the components necessary for a bimanual whole-hand haptic interaction [Ott et al., 2007]: additionally to the previously mentioned CyberGlove and CyberGrasp, the CyberForce provides 6DOF hand tracking and 3DOF force feedback on the wrists. The workspace is that of human arms when stretching them forward but slightly less on the sides, and is limited towards the torso, as well as subject to interface collision if the arms are crossed. Other devices were used conjointly with the CyberGlove and CyberGrasp for whole hand interaction in telepresence applications, such as a couple of VISHARD10 devices for 6DOF feedback on the wrist [Buss et al., 2009] (Figure 2.7c). Another example is the DeKiFeD4, a robot arm-based haptic device that provides 4DOF of sensing and actuation (3 translation, 1 rotation), coupled with additional 6DOF force/torque sensors for the wrist, to provide a whole-hand interface for telepresence [Kron et al., 2004]. In a similar way, since the DLR bimanual haptic interface uses a magnetic clutch to couple the robot arm and the user's hand, it could be extensible to whole-hand interaction if combined with kinesthetic gloves [Hulin et al., 2008] or tactile devices [Hulin et al., 2011].

2.2.5 Discussion and Conclusion

A certain number of bimanual haptic interfaces exist, and they display a wide diversity in a lot of characteristics, as summarized in Table 2.1 for single-point interfaces and Table 2.2 for multi-finger interfaces. The factor that has potentially the biggest impact is the number of degrees of freedom, which separates these two main categories, with the special case of whole-hand interfaces for the latter. However, even within each category there is a certain range of degrees of freedom. For instance, while some single-point kinesthetic interfaces are underactuated with 6DOF in sensing and 3DOF in actuation, others can have 6DOF for both and potentially an extra DOF for grasping. For multi-finger interfaces, most of

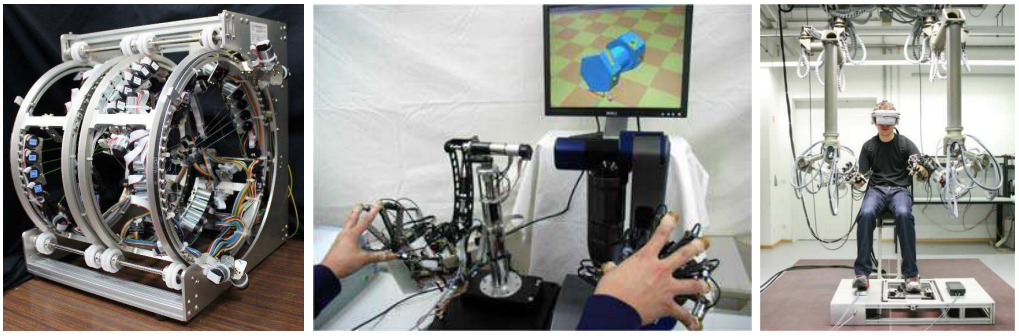


Figure 2.7 – Examples of multi-finger grounded bimanual haptic interfaces: SPIDAR-10 [Maruyama et al., 2013], Bimanual HIRO [Yoshikawa et al., 2009], and VISHARD-10 with CyberGlove and CyberGrasp [Buss et al., 2009]

them are 3DOF devices while the variety lies in the number of fingers supported, though the MasterFinger-2 [Garcia-Robledo et al., 2009] chose to trade the number of fingers for more DOF in sensing. Some of these multi-fingered interfaces also include the palm, increasing even further the number of DOF. Tactile devices have an even wider range of possible outputs, whether through motors with belts/wires pulling fingertips in one or more directions, thermoelectric elements for heat display on the fingertips, or vibrating motors for either the fingers or the arm. The great differences between all of these categories raise different problematics in terms of haptic rendering, which will be covered in the next section.

There is also a wide range of workspace sizes. Some devices have a small workspace (*e.g.* Phantom), while others offer a workspace adapted to movements of the forearm (*e.g.* Freedom 6S [Demers et al., 1998]) or the entire arm (*e.g.* Haptic Workstation [Ott et al., 2007]). Finally, some of them allow to move in an almost infinite horizontal space through mobile robots. An additional issue concerning workspaces is that of interface collision. Some devices with small workspaces, like the SPIDAR G&G [Murayama et al., 2004], are distant enough from each other to avoid such problems. Most interfaces with bigger workspaces, however, do encounter this issue, although some of them incorporate additional DOFs to avoid it, like the DLR bimanual haptic device. Great variability is also observed in terms of exertable forces, as the maximum peak forces can range from as low as 2.5N for the Freedom 6S [Demers et al., 1998], for instance, up to 154 N for the VISHARD7 [Peer and Buss, 2008].

This large spectrum of possibilities for all of these haptic devices makes it all the more difficult to develop generic software solutions that would encompass all of them. For instance, for mobile devices, a controller for the mobile robot has to be added. Grounded devices have workspace limits that require interaction techniques to overcome them. A whole-hand interface needs virtual hand models to handle the repartition of forces over the phalanges, palm and wrist. Tactile devices, whether they are based on vibrations, temperature or motors, may also require different kinds of controllers and require a finer rendering of haptic textures.

Another element worth noting is that most presented interfaces are symmetrical: in most cases the same device is used in each hand, at most with an adaptation for the left and right hand for some of them, *e.g.* gloves. The case where each hand holds a different interface has scarcely been investigated, apart from a few cases like the study of de Pascale et al. [2005] on grasping of locally deformable objects with two different PHANToM devices. Kron et al. [2004] pushed the concept further in teleoperation by providing two different grippers to the telemanipulator, while the work by Ullrich and

Table 2.1 – Overview of current bimanual single-point mobile haptic interfaces. All of the interfaces provide kinesthetic feedback. Interfaces marked with 7DOF have additional grip-force interfaces.

Device	Reference or Manufacturer	Input (DOF)	Output (DOF)
SPIDAR G&G	[Murayama et al., 2004]	7	7
omega.7	Force Dimension	7	4
sigma.7	Force Dimension	7	7
Freedom 7S	[Hayward et al., 1997]	7	7
DLR Bimanual Haptic Device	[Hulin et al., 2008]	6-7	6-7
omega.6	Force Dimension	6	3
Freedom 6S	[Demers et al., 1998]	6	6
Geomagic Touch	Geomagic	6	3-6
GRAB	[Bergamasco et al., 2006]	6	3
Virtuose 6D	Haption	6	6
VISHARD10	[Ueberle et al., 2004]	6	6
Mobile Haptic Grasper	[Formaglio et al., 2006]	6	3
Mobile VISHARD7	[Peer and Buss, 2008]	6	6

Table 2.2 – Overview of current bimanual multi-finger haptic interfaces.

Device	Reference or Manufacturer	Input (DOF)	Output (DOF)	Mobility and Feedback
Both-Hands SPIDAR	[Ishii et al., 1994]	3×2	3×2	Grounded Kinesthetic
SPIDAR-8	[Walairacht et al., 2001]	3×4	3×4	
SPIDAR-10	[Maruyama et al., 2013]	3×5	3×5	
Bimanual HIRO	[Yoshikawa et al., 2009]	3×5	3×5	
MasterFinger-2	[Garcia-Robledo et al., 2009]	6×2	3×2	
Haptic Workstation	[Ott et al., 2007]	$6 + 22$	$3 + 5 \times 1$	
CyberGlove + CyberGrasp	CyberGlove Systems	22	5×1	Body-based Kinesthetic
GhostGlove	[Minamizawa et al., 2008a]	N/A	$2 + 2 \times 5$	Body-based Tactile
CyberTouch	CyberGlove Systems	18-22	0-125 Hz	

Kuhlen assigned two different tasks in a virtual environment, with an adapted PHANToM Omni for palpation with the non-dominant hand [Ullrich and Kuhlen, 2012]. Considering the specificities of the NDH and DH as well as their interconnections described in this section, it would potentially be beneficial to use different devices for both hands more often by taking into account their respective roles.

2.3 Bimanual Haptic Software

Following the overview of the available two-handed haptic hardware, this section focuses on the software side of bimanual haptic interaction. First, the haptic rendering and virtual coupling techniques that allow a user to interact with a VE and receive force feedback from it are developed. Then, physically-based methods used to simulate VEs for haptic interaction are presented, encompassing the simulation of differents states of matter, resolution of contacts, and complex hand models for grasping. Existing haptic Application Programming Interfaces (APIs) are overviewed, as well as bimanual haptic software developed using these APIs and their architectures.

2.3.1 Haptic Rendering and Coupling

Haptic rendering refers to the methods used to generate and render haptic feedback to users, notably resulting from interaction with a VE. So far, techniques used in two-handed haptics for transmitting forces from either a remote or virtual environment to the user are similar to those used in unimanual haptics. For instance, in telepresence, the position of the hands of the operator are sent to the telemanipulator, and the interaction forces sensed by both arms of the latter are fed back to both haptic devices, with filtering of the received forces in order not to exceed the maximum capabilities of the devices [Kron et al., 2004, Peer et al., 2006].

A pioneering technique for haptic interaction and rendering with VEs is the *god-object* method, which uses a point representation of the haptic device in the simulation that will respond to physical contraints [Zilles and Salisbury, 1995]. The use of virtual proxies, *i.e.* representations of the haptic devices with an object instead of a point, was later proposed along with smoothing of object surfaces and the addition of friction [Ruspini et al., 1997]. A generalization of the *god-object* method for 6DOF haptic interaction with rigid proxies was later proposed, which notably performs well with objects of high complexity [Ortega et al., 2007].

The rendering of forces during interaction between a user and a VE usually follows a closed loop [Carignan and Cleary, 2000]. Motions of the device, constrained by the user, are sent to the proxy in the VE, and the results from the interaction of that proxy are fed back to the device. There are two main classes of closed-loop rendering schemes: admittance and impedance. Admittance control measures the forces applied by the user and control the position and/or velocity of the end effector of the device. Inversely, impedance control detects the motions commanded by the user and controls the forces applied by the device.

A main issue that arises with closed-loop rendering is that the introduction of instabilities may cause oscillations that amplify over time. Virtual coupling was introduced to stabilize the loop by applying a spring-damper link between the haptic device and its virtual counterpart [Colgate et al., 1995]. The 6DOF *god-object* method by Ortega et al. [2007] iterated on the virtual coupling approach by separating the computation of the motion of the proxy and that of force feedback. This suppresses some force artifacts found

with the regular virtual coupling approach (Figure 2.8).

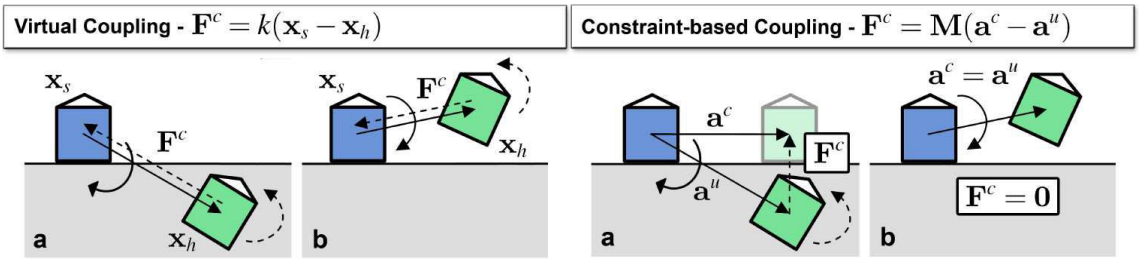


Figure 2.8 – Principle of the virtual coupling method and the constraint-based method by Ortega *et al.* (from Ortega et al. [2007]). **(a)** The constraint-based approach removes unwanted tangential forces in the force feedback. **(b)** It also removes artificial forces when moving away from objects.

Haptic rendering in the specific case of bimanual haptics has greatly focused on the rendering of grasping of virtual objects. One way to achieve this is through the use of virtual coupling with virtual hands, such as with the mass-spring hand model developed by Ott for the Haptic Workstation [Ott et al., 2007], in which the palm and each phalanx are represented by rigid bodies, linked by joints with springs and damping. The spring-hand model follows the movements of the tracked hand while being physically constrained by contact with virtual objects, and collision information is used to compute force feedback on the palm and individual fingers.

Two heuristic approaches were also proposed for haptic rendering of multi-finger grasping of objects [Garcia-Robledo et al., 2009]. The first method consists in separating objects in two halves and connecting those halves by springs. This solution works well if the object is to be grasped from opposite points, but not when it is grasped from neighbor points. The second method proposes to simulate springs between all fingers that manipulate an object, and have the force feedback be that of the spring and no longer that of the collision response with the object.

2.3.2 Physically-based Simulation

In virtual reality, the interaction between different objets of a virtual environment can be handled in a realistic way by physical simulation. To this date, the methods used for physical simulation in bimanual haptics are exactly the same as the ones used in unimanual context.

The physically-based simulation of multiple bodies usually involves three main steps per iteration. First, the position of all the degrees of freedom in the next time step is computed without any constraints. Then, the violation of constraints is determined, which includes collision detection for the interpenetration between bodies. Finally, the constraints are enforced using several methods, ensuring separation between bodies, friction under sliding contact, and other phenomena that constrain the relative motion between bodies.

2.3.2.1 Simulation of the mechanical behavior of bodies

Rigid bodies assume that the particles that constitute them are linked by springs of infinite stiffness, the distance between those particles remaining the same regardless of the forces applied to them. This allows rigid bodies to be defined by only 6 degrees of freedom describing their position and orientation. The surface of rigid bodies is usually modeled as polyhedra, but parametric surfaces can also be used [Kry and Pai, 2003]. This simplicity

of rigid body systems makes them highly efficient for real time interaction, and notably for haptics. Most commonly known physics engines, such as Open Dynamics Engine [Smith, 2011], Bullet Physics [Coumans, 2012], PhysX (NVIDIA Corporation) or Havok Physics (Havok), handle rigid body dynamics. A comprehensive introduction to rigid body dynamics can be found in [Baraff, 1997].

Deformable bodies, or soft bodies, are defined by their stress and strain as well as their displacement with respect to an undeformed configuration. Mass-spring models discretize matter into multiple nodes linked by springs and dampers to model deformation. They were among the first to be used in haptics due to their simplicity and computational efficiency. Finite Element Methods (FEM), on the other hand, are based on continuum mechanics. They model bodies as sets of contiguous elements, usually tetrahedra, that carry the aforementioned rheological properties [Zhuang and Canny, 1999, Felippa, 2000, Picinbono et al., 2003, Irving et al., 2004]. They tend to be preferred over mass-spring models, producing more realistic deformation at a higher cost. A comprehensive review on deformable body simulation can be found in [Nealen et al., 2006].

Haptic interaction with soft bodies has received certain attention, notably for medical simulation. A corotational approach was proposed to decouple rigid global motion from local deformation [Nesme et al., 2005], allowing to maintain haptic frame rates even with the high cost of FEM models [Duriez et al., 2006]. This approach was tested on a bimanual haptic simulation, with a snap-in task between a deformable clip and a quasi-rigid or deformable pipe. Several studies also proposed the use of local approximations of the deformation at the site of contact, either for deformable bodies [Jacobs and Cavusoglu, 2007, Cavusoglu and Tendick, 2000] or quasi-rigid bodies [de Pascale et al., 2004]. The medical simulation framework SOFA [Allard et al., 2007, Faure et al., 2012] implements a few deformable models such as FEM, and was used in bimanual haptics for a medical training simulator [Ullrich et al., 2011b] as well as for an immersive ultrasound-guided needle puncture simulator [Vidal et al., 2009].

Fluid simulation has, in comparison, received less attention than other kinds of simulation for haptic interaction. Some recent studies have focused on using precomputation [Dobashi et al., 2006] or approximations [Mora and Lee, 2008] for fast computation of the fluids for haptic rates, while others focused more on accurate models [Baxter and Lin, 2004, Yang et al., 2009]. A method has also been proposed for simulation of both rigid bodies and fluids through smoothed-particle hydrodynamics, shown to be suitable for bimanual haptics [Cirio et al., 2011].

2.3.2.2 Resolution of contacts between objects

Several methods allow to resolve interpenetrations and friction between objects in contact, which can be classified in three main categories: penalty-based, impulse-based, and constraint-based methods. Penalty-based methods attempt to separate objects using spring-damper forces at contacts, based on penetration depth and velocity [Moore and Wilhelms, 1988]. Though not straightforward, static and dynamic friction can also be handled with such methods [Yamane and Nakamura, 2006]. While these methods are fast to compute, the reliance on stiffness and damping parameters decreases the stability and perceived rigidity of contacts [Constantinescu et al., 2005]. Impulse-based methods solve multiple contacts as a sequence of individual collisions over several iterations [Mirtich and Canny, 1995, Chang and Colgate, 1997]. This allows to simulate much stiffer contacts, but may result in a high number of iterations when many contacts are involved.

Penalty-based and impulse-based methods are generally fast to compute but approximate the contact simulations. Constraint-based methods attempt to precisely solve all

penetrations and friction between objects in contact by solving a linear complementarity problem (LCP) [Baraff, 1996]. The resulting system can be solved either in one step using a direct solver [Baraff, 1991, 1994, Pauly et al., 2004], or by processing contacts sequentially using an iterative solver [Duriez et al., 2006, Erleben, 2007, Otaduy et al., 2009]. For friction, a pyramid discretization of the friction cone has to be used to keep the problem as a LCP [Baraff, 1991, Milenkovic and Schmidl, 2001], but an exact cone can also be used by using iterative methods to solve the resulting non-linear complementarity problem (NLCP) [Duriez et al., 2006]. Methods have also been proposed to more specifically handle contact constraints with articulated bodies, either rigid [Duriez et al., 2008] or deformable [Galoppo et al., 2007].

2.3.2.3 Methods for improving contact solving efficiency

Unlike rigid body simulations which have a very limited amount of contacts, scenarios involving deformable objects tend to produce large numbers of contact points due to the molding of the objects they are colliding with. This may have a significant impact on computation times, becoming an issue when attempting to maintain high simulation rates for haptics. This raises the need for methods to efficiently handle such complex scenarios. When deformable objects such as FEM models collide, each contact point tends to affect a small number of degrees of freedom. However, when a deformable object collides with a rigid body, a large number of contacts affects a small number of degrees of freedom, which may cause computational problems when attempting to solve the resulting LCP. Separating constraint sets containing rigid-rigid, deformable-deformable and rigid-deformable contacts was shown to be an efficient way to solve large and dense LCPs involving all these types of contacts simultaneously [Miguel and Otaduy, 2011].

Contact reduction methods can also be used for speeding up the resolution of contacts, by reducing the number of contacts to be solved. They are commonly used in physics engines for video games [Moravanszky and Terdiman, 2004], but they are also suitable for haptic interaction [Kim et al., 2003b]. Some preprocessing can be applied at the collision detection level to ensure that the least amount of redundant contacts gets generated in the first place. Then, contact clustering methods can be used to further reduce the number of contacts, for instance using the Euclidian distance between contacts, keeping a single weighted average contact point and contact normal for each cluster, with the maximum penetration depth of the contact points within the cluster [Kim et al., 2003b]. Such methods are best suited for rigid body interaction however, as deformable bodies require a fine sampling over the entire contact surface.

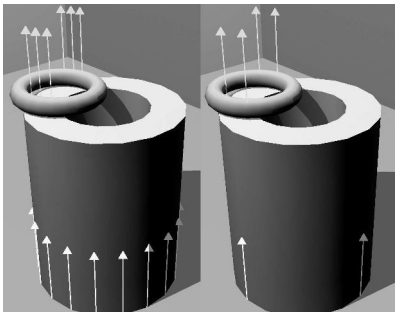


Figure 2.9 – Contact clustering with two rigid bodies resting on a plane (from [Moravanszky and Terdiman, 2004]). Contacts are represented by arrows, before and after reduction.

Another method consists in reducing the number of constraints generated during contact by formulating contact as *volume constraints*, as opposed to *point constraints* [Allard

et al., 2010]. Contact constraints are no longer defined as numerous penetration distances, but rather as a much reduced number of penetration volumes to be solved, number which is defined by the arbitrary resolution of a regular grid. This method also gathers friction frames at the different contact points into one frame per volume constraint, which may cause torsional friction to be lost if only one volume is used, as this type of friction is usually obtained by the combination of all the friction forces at each contact point.

A few models exist for simulating torsional friction from a single contact. Barbagli et al. [2004] derived a simple *soft finger* model from the study of human fingertips to achieve two-finger grasping of virtual objects with only one rigid contact per finger. This work was later extended to account for the coupling between tangential and torsional friction forces [Frisoli et al., 2006] using the concept of limit curve [Howe and Cutkosky, 1996]. Ciocarlie et al. [2007] use approximations of the local geometry of the finger and object in contact for their soft finger model, also achieving grasping with torsional friction using rigid fingers. Finally, the Coulomb-Contensou friction model [Contensou, 1963, Leine and Glocker, 2003] is a model that extends regular Coulomb friction by formulating tangential and torsional friction as a function of sliding and angular velocity, also taking into account the interpendencies between both types of friction.

2.3.2.4 Hand models for grasping

One of the most straightforward methods to perform physically-simulated hand interaction is the use of hand models based on articulated bodies, where palm and phalanges are represented by interconnected rigid bodies (Figure 2.10). Borst and Indugula introduced the use of such models, by using linear and torsional spring-dampers between the simulated palm and phalanges and their tracked configurations [Borst and Indugula, 2005]. This effectively allowed grasping of objects, though this required high friction coefficients which could result in excessive sticking. This method was shown to be sufficiently efficient to allow real-time dexterous interaction, notably with a pressure-sensitive glove-free device Kry et al. [2008], or with haptic feedback on the palm and fingers [Ott et al., 2007]. Ortega et al. [2007] improved on this model by building upon the 6DOF *god-object* method to build a fully constrained *god-hand* model, which solves the interdependencies between the multiple rigid bodies of the hand using Gauss's principle of least constraint [Jacobs et al., 2012]. Several studies have focused on synthesizing plausible grasping motions with rigid body hands [Kry and Pai, 2006, Liu, 2008, Ye and Liu, 2012]. Overall, models based on articulated bodies are computationally efficient, but they do not account for skin deformation under contact, which is not only necessary for more realistic-looking interaction and motion [Jain and Liu, 2011], but also to simulate friction at contact surfaces more accurately.

The hand is a complex structure made of a skeleton, muscles, tendons and skin, each with their own physical properties, making it challenging to accurately simulate the deformation of the hand as a whole during motion and contact. While it is possible to simulate the biomechanical behavior of the tendons and muscles of the hand [Sueda et al., 2008], such methods do not meet the performance requirements for interactive simulation and are more suited for offline rendering. Thus, some more approximate and more efficient methods have been proposed to simulate finger deformation in real time (Figure 2.11). Gourret et al. [1989] proposed a model for hand animation made of a FEM-based deformable hand with an underlying rigid skeleton, allowing more realistic contact with deformable objects. The rigid skeleton was made of realistic bones, so as to better simulate the restriction of flesh deformation due to bones. Other deformable hand models use conjointly a rigid hand to simulate the back of the hand, which is the least deformable part, and soft bodies

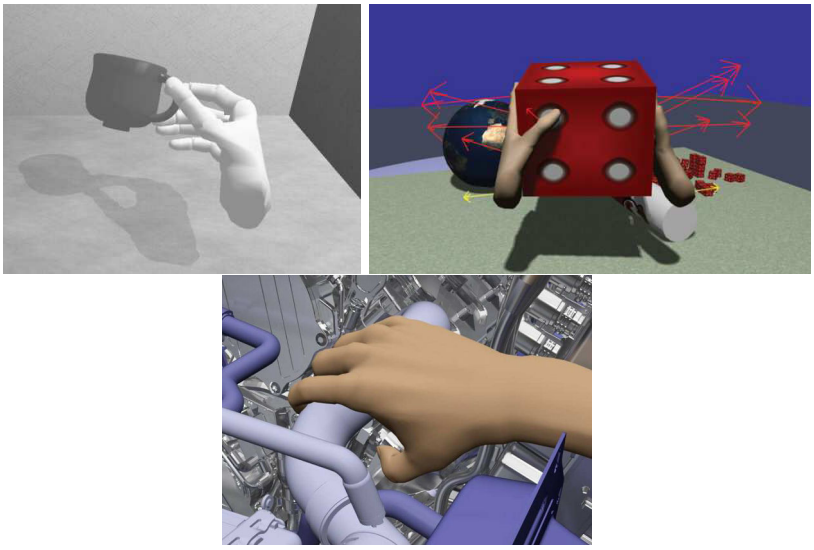


Figure 2.10 – Examples of rigid hand models for interaction with virtual environments: spring-damper models for unimanual interaction [Borst and Indugula, 2005] and bimanual haptic interaction [Ott, 2009], and *God-hand* model [Jacobs et al., 2012].

to simulate the finger pads, either using FEM [Pouliquen et al., 2005] or lattice shape matching with adaptive stiffness [Jacobs and Froehlich, 2011]. Garre et al. [2011] proposed another way to couple a FEM-based hand with a rigid body hand, by using springs between nodes of the deformable hand and rest positions skinned from the rigid body configuration. These methods effectively allow dexterous manipulation of objects with fairly realistic deformation under contact. Contact is also more realistic due to the larger contact surfaces compared to articulated bodies models. However, the number of contacts could become huge when taking into account different fingers, preventing real-time performances.

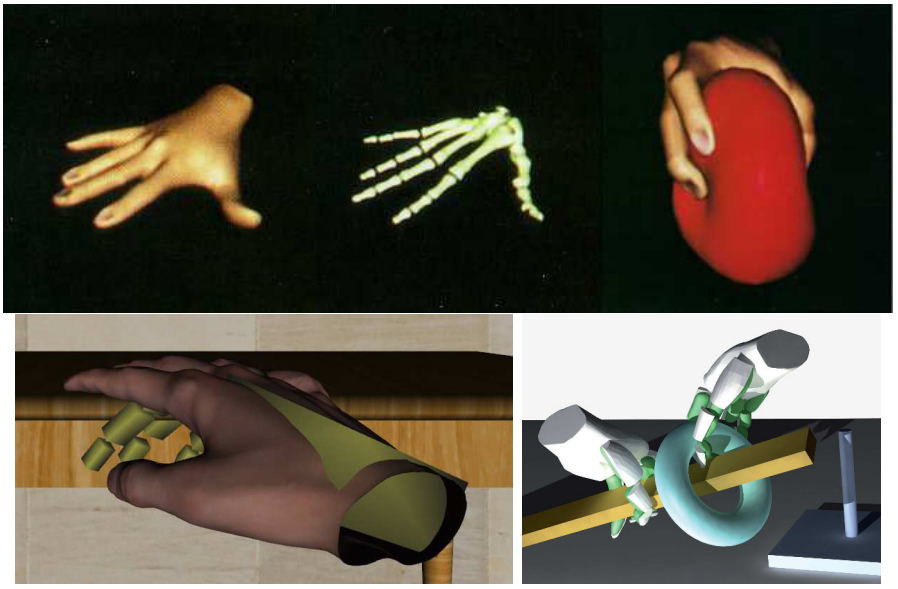


Figure 2.11 – Examples of deformable hand models for interaction with virtual environments: FEM-based models with underlying rigid skeleton [Gourret et al., 1989] or coupling to a rigid skeleton with springs [Garre et al., 2011], and lattice shape matching-based model [Jacobs et al., 2012].

2.3.3 Bimanual Haptic Applications

Several APIs and frameworks are available to handle haptics in applications, however they tend not to explicitly show if they support multiple devices at the same time, which may confuse a developer interested in bimanual haptics. We review the available APIs for bimanual haptics, as well as software developed using those APIs and their architectures.

2.3.3.1 Haptic APIs

Haptic APIs can be divided in two major categories: device-specific and generic APIs. A summary of most existing APIs for both of these categories is shown on Table 2.3, all of which being able to connect to multiple devices simultaneously. Device-specific APIs provide low-level access to one device or series of devices. Most of them are developed for unimanual devices, but MHaptic in particular is bimanual-oriented, the Haptic Workstation being a two-handed device [Ott, 2009]. Generic libraries have the advantage over device-specific APIs of supporting different haptic devices.

Table 2.3 – Overview of APIs usable for bimanual haptics. Device-specific APIs are first listed, then generic APIs.

API	Supported devices	Other features
Haptic SDK (Force Dimension)	Force Dimension	
OpenHaptics (Geomagic)	Geomagic	Haptic rendering
Freedom 6S API (MPB Technologies)	MPB Technologies	
HDAL SDK (Novint)	Novint	
libnifalcon [Machulis, 2009]	Novint	
MHaptic [Ott, 2009]	Haptic Workstation	Haptic rendering, physics, graphics
JTouchToolkit	Geomagic, Novint	Graphics
HAPI (SenseGraphics AB)	Geomagic, Novint, Moog FCS Robotics, Force Dimension	Haptic rendering
Haptik Library [de Pascale and Prattichizzo, 2007]	Geomagic, Haption, Force Dimension, MPB Technologies, Novint	Networking
CHAI 3D [Conti et al., 2003]	Force Dimension, Novint, MPB Technologies, Geomagic	Haptic rendering, graphics
H3DAPI (SenseGraphics AB)	Geomagic, Force Dimension, Novint, Moog FCS Robotics	Haptic rendering, graphics

A few of these APIs have been used as a framework to develop bimanual haptic software. An example is the M4 system for visuo-haptic manipulation of 3D meshes [Faeth et al., 2008], for which the software layer was developed on top of the H3D API (SenseGraphics). Some H3D nodes were extended to provide grabbing ability for the non-dominant hand and mesh manipulation abilities (cutting, deforming and painting) for the dominant hand, although two handed manipulation of the mesh is also made possible. The Haptik Library was also used to develop the software layer of the Mobile Haptic Grasper [Formaglio et al., 2006]. A plugin was specifically designed to control the mobile part of the system and added to the library, allowing any user to use mobile interfaces with the simple principle that a mobile device can be considered a grounded device with a huge workspace. Grasping was included in the software using the previously mentioned soft fingers method.

2.3.3.2 Software architectures

The software architectures for bimanual haptics differ whether we look at the cases of telepresence, or virtual reality, due to the different requirements of both contexts. While both share haptic control threads, on one side the teleoperation case has to deal with networking and teleoperator arm configurations, and on the other side the virtual reality case has to deal with collision detection and simulation. In both cases, however, the architectures tend not to be very different from those used for unimanual haptics.

Bimanual haptic teleoperation follows the same software structure as for unimanual telepresence, that is to say a haptic control loop communicating with the haptic devices, a teleoperator control loop which communicates with the manipulator arms, and data sent through UDP communication between the two [Kron et al., 2004, Peer et al., 2006]. It can optionally include collision avoidance algorithms to detect dangerous configurations of the manipulator arms and provide force feedback to avoid those [Kron et al., 2004].

Architectures used in bimanual interaction with virtual environments are rarely explicited, however three papers described those architectures in a comprehensive way, by Ott et al. [2007] for the MHaptic library, Garcia-Robledo et al. [2011] for the presentation of the MasterFinger-2 used in a bimanual context, and Ullrich et al. [2011b] for the Bimanual Haptic Simulator.

The two first architectures bear certain similarities, notably the fact they follow the same scheme: a simulation thread, two haptic threads, and a visual rendering thread. The simulation thread manages the virtual environment by detecting collisions and then solving object-object and device-object collisions. This thread includes a haptic hand model in the case of MHaptic while the MasterFinger-2 includes a grasping detector. The haptic threads read raw positions and orientations from the sensors and manage the actuators according to the results of the simulation, the special case of MHaptic integrating an anti-gravity software to improve user comfort in these threads. The visual rendering thread simply reads the data from the simulation (and the hand model in the case of MHaptic) and displays it to the user.

The architecture of the Bimanual Haptic Simulator is fairly similar to the two others but with clear differences as well, aside the fact it simulates deformable bodies. The similar parts are the presence of the simulation and visual rendering threads, though their refresh rates are drastically different from the previous ones. Notably, the physics thread runs at 25 Hz instead of 200-300 Hz due to the computationally intensive nature of deformable body simulation. Also, while the previous systems had a single thread for each haptic device, this one has two threads per device: an interaction thread and a thread for force algorithms and haptic rendering. While the latter could be considered similar to the haptic threads of the previous systems, running at 1 kHz and taking care

of force feedback (with special force effects), the former is completely different, handling interactions between haptic device and simulated tissue, which was a part that was present in the single simulation thread in the other architectures.

Table 2.4 – Thread refresh rates for three bimanual haptic software architectures.

Software	Simulation	Haptic	Interaction	Physics	Visual
MHaptic [Ott, 2009]	Rigid body	950 Hz	N/A	300 Hz	60 Hz
MasterFinger-2 [Garcia-Robledo et al., 2011]	Rigid body	1,000 Hz	N/A	200 Hz	50 Hz
Bimanual Haptic Simulator [Ullrich et al., 2011b]	Deformable	1,000 Hz	120 Hz	25 Hz	Unlimited

2.3.4 Discussion and Conclusion

Bimanual haptic rendering is mostly similar to the unimanual case, with notably universal techniques like the virtual coupling, *god-object* method and hand models. Actual bimanual techniques have also been developed, focusing mostly on the rendering of grasping with single-point or multi-finger interfaces. However, very much like bimanual hardware, those techniques consider the two hands equally, not taking into account any form of asymmetry of the hands.

Physically-based simulation techniques for bimanual haptics are also the same as those used for unimanual haptics, most often geometry-based rigid body simulation. However, other kinds of simulation have been explored, most notably deformable body simulation, which can be used to model hand deformation. Several methods allow to solve contacts, which make for different tradeoffs between computational efficiency (penalty-based methods) and physical accuracy (constraint-based methods).

Deformable objects increase the number of contacts in the simulation, which take considerably more time to solve for more accurate methods that operate at least in quadratic time to the number of contact points, and decrease stability of the less accurate methods [Moravanszky and Terdiman, 2004]. Additionally, contact solvers may encounter issues with strongly overdetermined systems. Bimanual interaction adds to this complexity by further increasing the number of contacts, especially with deformable hand models. This raises the need for more efficient contact solving methods, heuristics, or interaction techniques to solve these issues.

Separating constraint sets between same-state and multi-state contacts improves the contact solving in scenarios involving both rigid and deformable objects, with no cost in physical accuracy. Contact reduction methods simplify the systems to solve by removing redundant contacts, but losing some information compared to a fine sampling in the process. Formulating contacts as penetration volumes instead of penetration depths over multiple points allows to further reduce the number of constraints to solve, but multiple volumes are still required in order to maintain torsional friction over the contact surfaces.

Most existing haptic APIs support multiple point-based devices. Generic APIs were shown as suitable for developing two-handed haptic software, as illustrated with a 3D modeling software and the controller of a mobile bimanual haptic device. Bimanual haptic software follows a multithreaded architecture with common visual rendering and physical

simulation threads, though the handling of haptic devices may differ slightly. While haptic rendering is always performed separately for each device at a high rate, interactions between haptic devices and the objects of the simulation can be handled either in the common simulation thread, or in one specific thread for each device. Refresh rates also vary, showing different tradeoffs between higher update rates of the simulation and smoother visual rendering.

2.4 Interaction Techniques

The previous sections dealt with three separate elements of bimanual haptic interaction: user, hardware and software. Interaction techniques gather all of these elements, combining hardware and software solutions to allow a user to perform a specific task within the VE. There are currently very few interaction techniques that can be considered purely bimanual haptic. However, this list can be enriched by looking at close fields such as unimanual haptic interaction, which techniques can be used in a dual way for bimanual haptics. Non-haptic two-handed interaction techniques also bear principles that could be used as inspirations for designing novel bimanual haptic techniques.

An element of classification for 3D interaction tasks is the taxonomy of Bowman [1999], which defines four main classes of interaction: navigation, selection, manipulation and system control. The interaction techniques that are reviewed in this section unevenly fit these categories, hence these techniques will be reviewed following two main categories. Firstly, we consider the parts of the interaction that do not involve direct interaction with virtual objects, encompassing navigation and system control. Then, we consider the interaction with virtual objects, including selection and manipulation.

2.4.1 Navigation and System Control

Most haptic devices have limited workspaces that do not allow the exploration of large VEs by themselves, which raises the need for methods to extend these workspaces. While there are hardware approaches that increase the workspaces of bimanual haptic hardware, most of them do not increase those workspaces infinitely, and may require devices that are not necessarily widespread. This leads to the use of navigation techniques for handling the extension of these workspaces. They have the advantage of being generic and applicable to any haptic device available to the user with no further requirements, although the majority of currently existing techniques are not specific to two-handed haptics, and are rather found in the more general field of two-handed 3D interaction.

An early bimanual technique in the field of 3D graphics interfaces proposed the use of the NDH for controlling the camera while the DH executes the tasks [Balakrishnan and Kurtenbach, 1999]. While this technique could be hardly used in conjunction with tasks requiring both hands in a VE, a scenario could be considered where only one hand is performing a task, while a simultaneous navigation task (*e.g.* moving a small object over a long distance) could be assigned to the other hand. The *bulldozer* metaphor was also proposed, for navigation in 3D environments with 2D interfaces [Zhai et al., 1999]. Using a dual joystick configuration, the user is given 4DOF navigation capabilities (3DOF in translation and 1DOF in rotation), using a metaphor similar to the handling of a shopping cart: pushing both joysticks forward for moving forward, pulling them both to move backwards, pushing both on the same side to move sideways and pulling one while pushing the other to turn (Figure 2.12). Translation on the vertical axis was added by

having the user push both joysticks on opposite sides. That interaction technique being close to a real life task, it was shown to perform well, and could potentially be adapted for 3D interfaces.

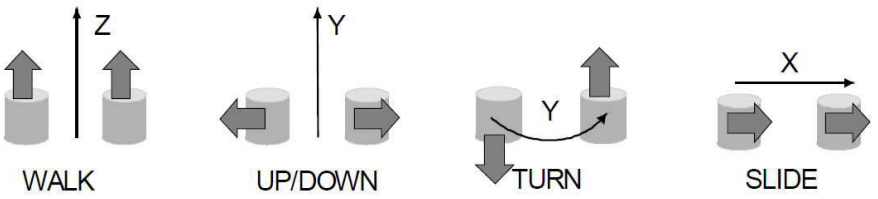


Figure 2.12 – Control schemes of the *bulldozer* metaphor for exploration of 3D virtual environments with 2D interfaces (from Zhai et al. [1999]).

A few unimanual techniques also exist for extending workspaces of haptic interfaces, one of the simplest being the use of a scaling factor to match the real workspace provided by the haptic devices with a virtual volume defined in the VE [Fischer and Vance, 2003]. A major drawback of this technique, however, is that the accuracy of motions in the virtual space decreases with the scaling factor, making precise manipulation and navigation in a large VE incompatible. Another approach is the *clutching* technique, which consists in holding down a button to temporarily interrupt the coupling between the device and the proxy while the user recenters the device. The *Dual Shell* method is an extension of this technique, that automatically handles the clutching when predefined boundaries are reached, without requiring the potentially counterintuitive manipulation of a button [Isshiki et al., 2008]. The use of rate control was also proposed to control the velocity of the virtual proxy through the position of the haptic device [Zhai, 1998]. This technique infinitely increases the workspace in all directions, although seeming to be far from intuitive notably for precise manipulation tasks.

The *Bubble* metaphor [Dominjon et al., 2005] uses direct control of the position of the proxy in the VE while the device remains inside predefined spherical boundaries (defined as the *bubble*). When the device moves outside this *bubble*, the latter is translated throughout the VE, with a small elastic force that gives the feel of the surface of the *bubble* to the user (Figure 2.13). For a unimanual painting task, a user experiment showed that this technique was faster, more accurate and more appreciated than the scaling factor and clutching techniques. The *Bubble* technique was used for bimanual multi-finger interaction with complex VEs through the Haptic Workstation [Ott et al., 2007]. This implementation allowed users to translate and rotate the camera by moving both hands outside the bubble in the same direction. It was shown to be efficient for simultaneous navigation and manipulation with this specific device, as it allows interaction with the palms and fingers of both hands. However, it remains difficult to use with single-point interfaces, as picked objects tend to be frequently dropped during the translations of the virtual workspace through rate control.

Finally, an alternate and generic way to facilitate navigation in VEs is to increase the number of degrees of freedom available to the user. The use of a 3DOF foot pedal was proposed in a context of teleoperation, in order to move the controlled robot while having both hands occupied simultaneously by other tasks [Peer et al., 2006]. More recently, two foot pedals were also integrated into the DLR bimanual haptic device, which could be assigned to different tasks [Hulin et al., 2011].

As far as system control techniques are concerned, in the context of 3D interaction techniques, Cutler et al. [1997] proposed ways to apply transitions between different tools, or subtasks, proposed to the user with the Responsive Workbench. They defined explicit

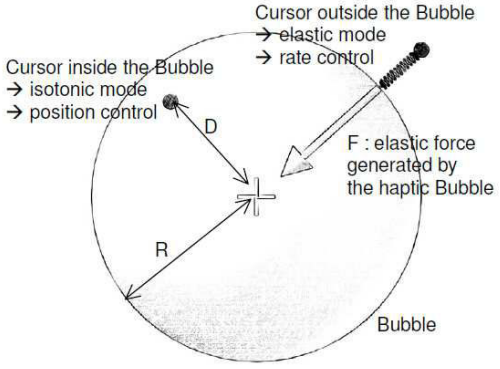


Figure 2.13 – Principle of the *bubble* technique for haptic exploration of virtual environments (from [Dominjon et al., 2005]).

transitions, like picking up a tool in a toolbox, with a default behaviour when no tool has been picked up yet, that should be specific of the application. Another example of explicit transition, more practical in that it does not require several movements between the workspace and the toolbox, is the use of hand gestures. Implicit transitions were also defined, in which switching from a subtask to another happens seamlessly, almost imperceptibly. An example of this is the switch from a unimanual grabbing technique to a bimanual grab-and-twirl technique, that occurs naturally as the user reaches in with the other hand to help the manipulation (Figure 2.14).

Figure 2.14 – Example of implicit transition between unimanual and bimanual tools. (from [Cutler et al., 1997]). While performing a unimanual 6DOF grab of a tool, the user approaches their other hand and implicitly switches to a bimanual 6DOF grab of that tool.

2.4.2 Selection and Manipulation

The act of "selecting" an object in a VE with haptic devices is often implicit. Touching an object with the representation of a haptic device in the VE, such as the previously mentioned *god-objects* [Zilles and Salisbury, 1995, Ruspini et al., 1997, Ortega et al., 2007], leads to the generation of one or more contact points by the collision detection engine. Doing so with a second virtual proxy simply adds more contact points, which in the case of physically-based VEs leads to a more complex system that may be resolved in the same manner or require additional methods to maintain haptic rates.

Some techniques were proposed to detect when a user attempts to grasp a physically-simulated object, in order to apply specific manipulation techniques to facilitate grasping and interaction. Most of these techniques focus on unimanual multi-finger interfaces, but

can be used effectively for bimanual interaction as well. Zachmann and Rettig [2001] used the distribution of contacts among finger phalanges, thumb phalanges and palm to discriminate between push, precision grasp and power grasp. Moehring and Froehlich [2005] used a similar approach, but instead using ray tests between distal phalanges to determine the fingers that participate in the grasping, with a hierarchy that prioritizes more stable grasps. Grasping can also be detected through the use of *grasp pairs*, which are defined as pairs of fingers contacting with an object for which the line between both contacts is included in both friction cones [Holz et al., 2008, Moehring and Froehlich, 2010] (Figure 2.15).

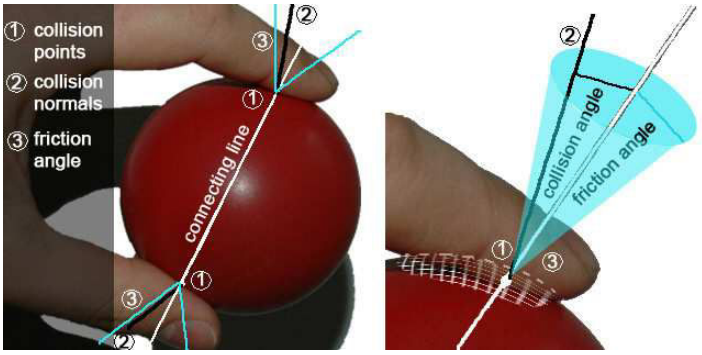


Figure 2.15 – Principle of the grasp pairs for multi-finger grasping (from [Moehring and Froehlich, 2010]). A grasp pair is valid when both collision normals lie within both friction cones.

For haptic manipulation of virtual objects, current interaction techniques mostly consist in the use of a unimanual technique in a dual way, or with an adaptation for two-handed interaction. *God-object* methods represent the hand and/or fingers with single points or simple meshes that are often mechanically the same. They are the only physically-based interaction technique that allows manipulation of virtual objects with single-point interfaces, though bimanual manipulation of virtual objects this way can prove challenging due to the minimal amount of contact points, often insufficient for stable grasping. Rigid virtual hand models increase sufficiently the number of contact points in order to achieve grasping, especially in a two-handed scenario, but contact with such models can be unrealistic, as discussed in Section 2.3.2.4. Deformable hand models solve this issue, but suffer from the very large quantity of contacts to solve as a result of the conformation of the fingerpad surfaces to the object surfaces.

Grasping can be facilitated by using heuristics once the intent of the user to grasp an object is detected using the aforementioned selection techniques. A common technique for unimanual multi-finger grasping consists in controlling the motion of the grasped object by that of the fingers considered as taking part in the grasping. This can be achieved by constraining the relative position and orientation of the object with respect to the palm (Figure 2.16), and potentially constraining additional axes for specific interaction scenarios (*e.g.* turning a key in a car, which involves only one axis) [Zachmann and Rettig, 2001, Moehring and Froehlich, 2005, Holz et al., 2008, Moehring and Froehlich, 2010]. While these methods allow stable grasps of virtual objects, they often have to sacrifice physical correctness in the process, often leading to unrealistic contact between hand and object and sometimes ignoring object-object collision. Furthermore, they may have to approximate the hand motions of the user as well.

Virtual proxy methods also allow to assign two different tools to each hand (Figure 2.17). For instance, the multi-modal mesh manipulation system [Faeth et al., 2008],

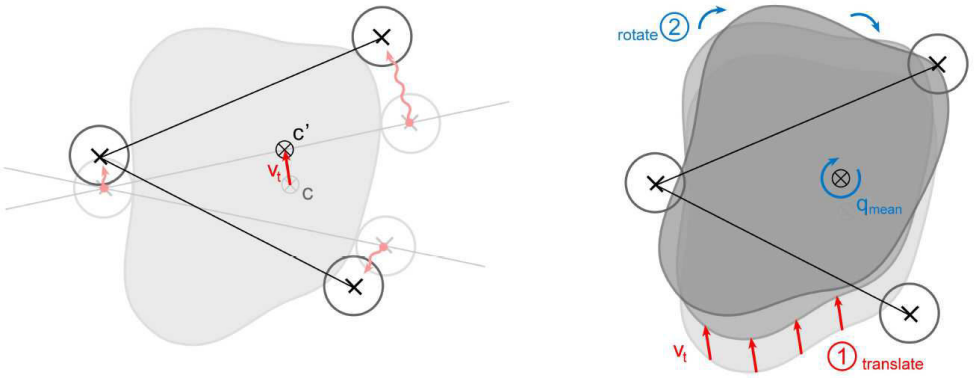


Figure 2.16 – Constraining the motion of a virtual object to the motion of grasp pairs (from [Holz et al., 2008]). Given a motion of the proxies grasping the object (left), a mean translation and rotation is determined for the object itself (right).

a bimanual 3D modeling software, uses a tool-object metaphor. It can assign a grabbing task to the non-dominant hand and different manipulation tools to the dominant hand, or a tool in each hand for bimanual manipulation of the mesh (*e.g.* stretching, folding, or tearing). Similarly, the Bimanual Haptic Simulator for medical training [Ullrich et al., 2011a] assigns a palpation task to the non-dominant hand and a needle insertion task to the dominant hand. The use of passive interfaces, or *props*, can also allow to assign different tasks to each hand. For instance, Lindeman et al. [1999] proposed hand-held windows on the non-dominant hand with which the dominant hand can interact, allowing the use of 2D interaction techniques within a 3D virtual environment.

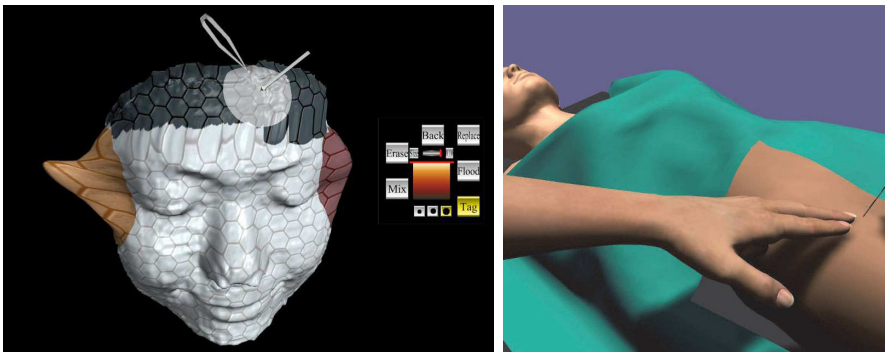


Figure 2.17 – Examples of multi-tool bimanual haptic applications. Bimanual 3D modeling software, where one hand manipulates the mesh while the other hand modulates its malleability by painting (from [Faeth, 2009]), and bimanual haptic simulator for medical training, with a needle insertion task with auxiliary palpation task (from [Ullrich and Kuhlen, 2012]).

2.4.3 Discussion and Conclusion

Interaction techniques can be used to enhance the bimanual haptic interaction of a user with a virtual environment, by combining hardware and software solutions to produce efficient metaphors. The amount of two-handed haptic interaction techniques remains fairly limited to this date, with some pioneering works starting to emerge.

As far as navigation techniques are concerned, most techniques are unimanual, and early methods tended to favor either accuracy of the motions in the VE, or smoothness of the navigation. The clutching method is precise but requires numerous back-and-forth

motions for navigation, the use of a scaling factor is better suited for large VEs but makes precise manipulation impossible, and the use of rate control allows smooth navigation but unintuitive precise manipulation. The *bubble* technique [Dominjon et al., 2005] proposed a better tradeoff by switching between position and rate control, allowing both precise motions in a small workspace and smooth navigation in infinite spaces. However, the bimanual implementation of the technique in MHaptic [Ott, 2009] barely exploited the capabilities proposed by having two hands, simply requiring to push both in the same direction in order to move or turn towards that direction.

Navigation techniques in VEs could be improved by taking from metaphors like the *bulldozer* metaphor, which originally allowed more DOFs for navigation in the virtual environment using less DOFs in input, and was also adapted for two hands. The asymmetry of the hands could also be taken into account by, for instance, only assigning navigation tasks to one hand, leaving the other hand free for other tasks. The use of foot pedals for navigating seems to be one of the most natural options, as we use usually our feet to navigate, however it is only possible with the appropriate hardware.

Selection techniques tend to be focused on multi-finger unimanual interaction, but their principles could be adapted for bimanual interaction as well. The information of which fingers are in contact with an object, as well as of the contact positions and normals involved, can be used to determine when the user attemps to grasp an object. Bimanual haptic methods for detecting grasping are also starting to emerge, notably through the study of the different steps of grasping [Garcia-Robledo et al., 2011]. These selection methods can be used to trigger manipulation techniques, which in turn may simplify the manipulation of the grasped object.

Manipulation techniques are currently not much different from the unimanual haptics field either. Generic methods like god-objects as well as rigid and deformable hand models are often simply duplicated and thus providee the same interaction possibilities to both hands, not taking into account the possibilities given by the combination of both. Still, some works have been using two different tools in each hand, such as assigning a pointing task to the non-dominant hand to benefit from the frame of reference of that hand for gaining accuracy with the dominant hand, as suggested by Guiard's kinematic chain model [Ullrich et al., 2011a]. Some pioneering work has been done on the topic of transitioning between tools in a two-handed scenario, in the more general context of 3D interaction [Cutler et al., 1997]. The adaptation of these transitions to the bimanual haptic context remains however to be done, and in general system control techniques in bimanual haptics are lacking as well.

2.5 Conclusion

Haptics can greatly enhance the immersion of a user into a simulated environment by stimulating the tactile and proprioceptive senses. Therefore, it has many applications in several fields such as medicine or industry. For the longest time, haptic studies focused on the use of a single device, but an increasing attention has been given to two-handed haptics in more recent years, as the use of two hands is a common happening in our daily lives. It was shown that there are clear benefits of bimanual interaction, which are related to the speed, accuracy and mental schemes with which tasks are carried out. These benefits were also shown to carry to the field of human-computer interaction, making bimanual haptics particularly relevant.

The sensorimotor and cognitive aspects of bimanual haptics have been well studied,

highlighting some specificities of the human bimanual haptic system. Notably, the hands have a clear specialization in terms of sensitivity, and perceptual information is transferred between both hands. The Kinematic Chain model Guiard [1987] defines major principles concerning the relationship between the dominant hand and the non-dominant hand, on a spatial and temporal scale. It was also observed that the use of two hands helps to compile different subtasks into single, less cognitively heavy tasks. The level of asymmetry deployed to perform a task was shown to be correlated to the difficulty of that task. There is a wide amount of data on the human bimanual haptic system, and it was shown that taking this data into account when designing interaction techniques could enhance their efficiency.

Several bimanual haptic devices have been created, whether mechanically designed to be bimanual or being unimanual devices used in pairs. All of these devices show a great variety in terms of degrees of freedom in both sensing and actuation, a major separation being made between devices with single effectors and multi-finger interfaces. There is also a wide variety in workspace sizes, between grounded interfaces that have limited workspaces, mobile interfaces, and haptic gloves. This wide variety of devices requires interaction metaphors and haptic coupling methods that are adapted to each of them, and that can possibly help overcome their shortcomings. However, most of these interfaces are kinesthetic, with only a handful of devices shown to be suitable for bimanual tactile feedback. Furthermore, most of them are symmetrical, the same device being used in each hand, and the influence of having two different devices in each hand has to the best of our knowledge not been studied.

As far as the bimanual haptic software is concerned, to this day little has been done to develop techniques that differ truly from those used in unimanual haptics. Haptic rendering and coupling is similar, though more specific techniques for haptic rendering of grasping exist for both single-point and multi-finger interfaces. Same as for hardware, it is worth noting that haptic rendering techniques are also symmetrical, considering both hands equally. Physically-based simulation techniques are also no different from unimanual haptics, the major difference being that bimanual interaction increases the complexity of contact scenarios. Methods to improve the performance of physical simulations, such as contact reduction, constraint aggregation or soft finger approximation methods, may thus become more relevant in the context of bimanual haptics. Current haptic APIs support the use of two interfaces conjointly, and shown to be suitable to develop bimanual software. Bimanual haptic software for interaction with VEs usually follows a multithreaded architecture with a simulation thread, a visual rendering thread, and dual haptic threads.

Bimanual haptic interaction techniques are still scarce, but unimanual techniques can be of interest for the design of new methods. For navigation, the *bubble* technique for navigation in VEs was adapted for bimanual navigation with haptic gloves. The use of foot pedals was also proposed to allow navigation without monopolizing the hands, though it is dependent on the availability of such hardware. Selection techniques are mostly focused on the detection of grasping, for instance by analyzing the repartition of forces between proxies and objects, which can be used to trigger manipulation techniques. Manipulation techniques are highly dependent on the type of device used, rigid and deformable hand models being the straightforward choice for haptic gloves. For devices with single effectors, rigid proxies representing a given tool are a common choice, and asymmetric tasks can be performed by assigning different proxies to each hand. These proxies are however usually limited to a single task, which raises the need for more techniques to improve their interaction capabilities or handle transitions between different tools.

Interaction Techniques for Bimanual Haptic Interfaces

3

Contents

Haptic devices with single effectors are commonplace, and an efficient way to interact with virtual environments two-handedly using rigid proxies. Due to the simplicity of these models as well as the low number of contact points generated by such proxies, haptic rates can be more easily achieved using them. However, some issues arise when interacting with a VE using single-point interfaces, which are two-fold.

Firstly, most of these interfaces are grounded and have very limited workspaces, which is problematic when attempting to navigate within a moderately large VE. Some workspace extension techniques have been developed in order to alleviate this issue, but they also suffer from some limitations. Increasing the scaling factor between the motions of the device and those of the proxy allows to increase the size of the workspace to some extent, but the accuracy of motions decreases with the increase of that factor. Rate control was also proposed, using the displacement of the device to control the velocity of the proxy, which increases the workspace infinitely but may prove unintuitive for precise manipulation. The clutching technique uses a button press on the device to stop the coupling between device and proxy while the user pulls the device back to a central position. It is the technique that conserves the most accuracy of all of these, but navigation can be tedious due to the repetitive back-and-forth motions required.

Secondly, the low number of interaction points leads to more difficult manipulation of virtual objects with simple proxies. Picking a virtual object two-handedly with single-point devices can be compared to picking a real object with only one fingertip of each hand, which is in itself challenging. With rigid proxies, the task gets even more complicated as

there is no deformation of the proxy to match the surface of the object. With 3DOF interfaces, the proxies can no longer be oriented, making it even more difficult to grasp an object from an optimal angle.

In this chapter, we present bimanual haptic interaction techniques that address these two specific issues, and take into account hand asymmetry as well. These techniques include a bimanual navigation technique, a method to improve navigation during grasping, a grasping detection method, and a two-handed haptic manipulation technique. We introduce the ***double bubble***, based on the *bubble* technique, which **allows free motion with both hands in a VE**, with a viewport adaptation method that maintains both virtual proxies within the field of view. We then present the ***joint control*, which facilitates the exploration of larger VEs while manipulating objects** by ensuring control modes and velocities are the same when interacting with the same object. A grasping detection method allows to determine when a user attemps to effectively pick an object between two rigid proxies. Finally, we describe the ***magnetic pinch***, which uses either springs or constraints to **keep the virtual proxies from unintentionally dropping an object** grasped with both hands. The techniques were evaluated against state of the art navigation and manipulation techniques, in terms of speed, accuracy, and user appreciation.

3.1 Navigation in Virtual Environments with Dual Haptic Interfaces

In the context of one-handed haptics, the *bubble* technique [Dominjon et al., 2005] was shown to propose an efficient tradeoff between precise movements when interacting with a VE, and smooth navigation within that VE. Precise manipulation is achieved by using position control within certain spherical boundaries, meaning that motions of the proxy are directly mapped to motions of the device within that *bubble*. When reaching outside of the *bubble*, the control scheme switches to rate control, in which the *bubble* moves within the VE at a constant velocity depending on the distance from the interface to the boundaries of the *bubble*. Haptic feedback is also provided to the user when reaching those boundaries, to clearly indicate the switch to rate control. This feedback is a unilateral spring that pulls the device back towards the surface of the *bubble*, which also helps the user get back to position control more easily once the *bubble* has reached the desired location.

A first bimanual implementation of the technique was proposed in the context of the MHaptic framework for the Haptic Workstation [Ott et al., 2007]. Both devices had workspaces fitting the range of arm movements, hence the *bubble* technique was used to make the user itself navigate within a VE. This was achieved by providing a common *bubble* for both hands, the user moving forward by pushing both hands forward, and turning around by pushing both hands to the same side. This proved to be a suitable navigation technique for this specific device.

However, this approach cannot be considered generic, as it uses the best case scenario where the user is provided with devices that fit the reach of their arms. It would not be suitable for devices with smaller workspaces, unless high scaling factors are used in which case accuracy is lost in the process. Additionally, since the Haptic Workstation is a multi-finger interface, it is easy to navigate while grasping an object as the hand models ensure that the grasping is stable in most circumstances. Overall, for the case we are interested in of interaction with generic single-point devices, possibly a different device per hand, and grasping with simple rigid proxies, novel methods are required to handle navigation. We

thus propose a different approach to the *bubble* technique, called the *double bubble*, where two independent *bubbles* are used instead of a common *bubble* for both hands.

3.1.1 Double Bubble

The aforementioned bimanual implementation of the *bubble* technique had one common *bubble* for both hands. In the *double bubble* technique, both devices have their own *bubble*, similarly divided into two areas associated to a control mode. For each device, an inner area controls directly the position of the proxy, and an outer area moves the corresponding *bubble* in velocity within the VE (Figure 3.1). Additionally to the haptic feedback of the original *bubble* technique provided during rate control, a visual feedback is also added in the form of a motion trail behind the proxy.

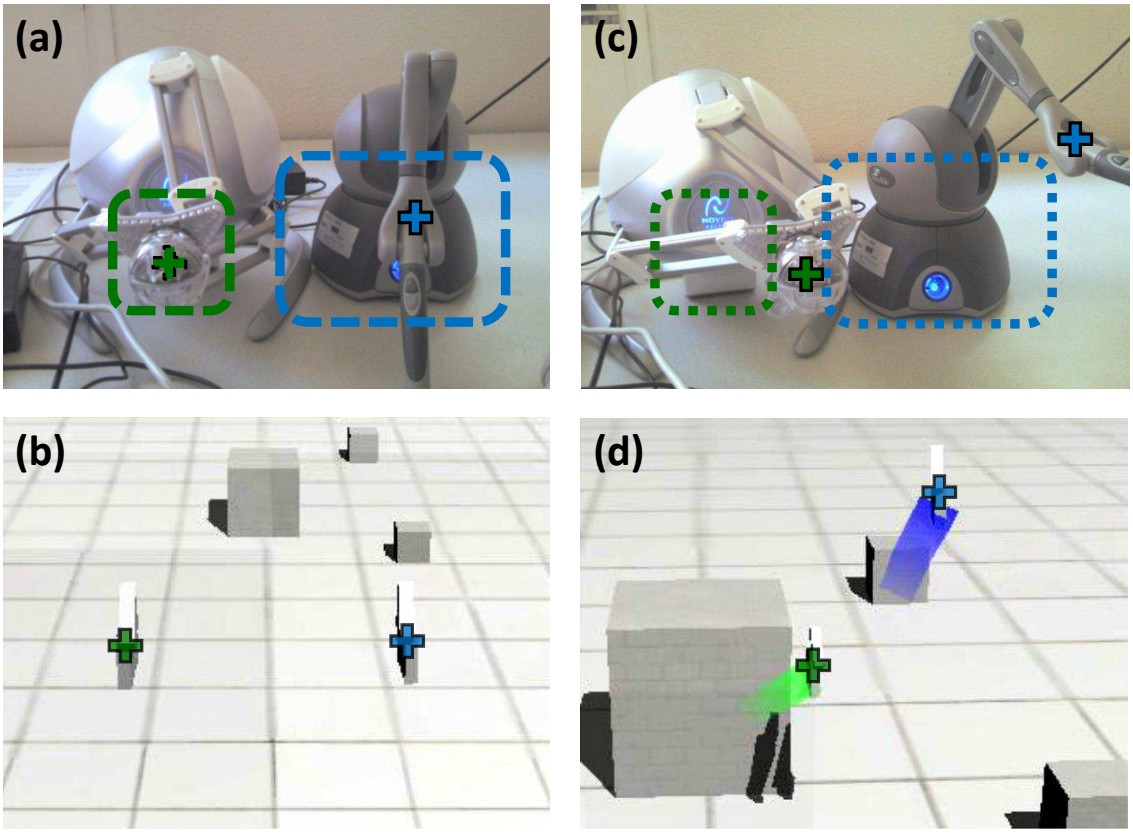


Figure 3.1 – Control modes of the *double bubble*. The top row represents the haptic devices with the workspaces in dashed lines. The bottom row represents the corresponding virtual environment. (a-b) Devices inside the *bubbles*: position control. (c-d) Devices outside the *bubbles*: rate control, indicated by colored trails.

Since the method is designed to be generic, it is expected that it can be used with devices that have non-spherical workspaces. A notable example is that of Phantom devices, which have a higher width than their height or depth. In order to better fit the physical workspaces of such devices, any convex hull (*e.g.* a cube or box) can be used as boundaries of the *bubble*, instead of just spherical boundaries (Figure 3.2).

The *double bubble* allows to move both hands independently within a VE, meaning that the right proxy can end up far at the left of the left proxy and vice versa. It was observed that this could lead to certain confusion for some users, who had a harder time figuring

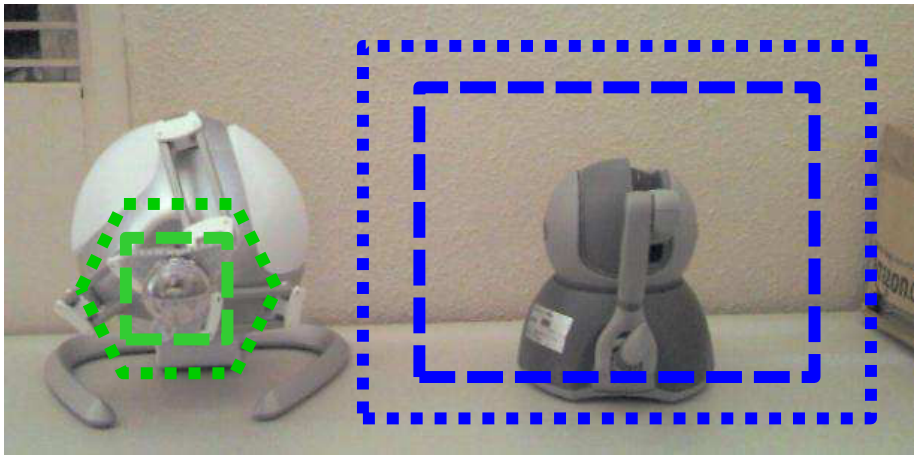


Figure 3.2 – Approximate *bubble* and physical workspace sizes of a Novint Falcon and a PHANToM Omni. The inner rectangles represent the *bubbles*, while outer polygons represent the physical workspaces (mechanical stops of the devices).

out which was which in such a scenario. A possible solution to avoid this confusion is to prevent the hands from crossing each other at all.

This is achieved by simulating an invisible "separation plane" that prevents the centers of the *bubbles* from crossing, by negating the horizontal component of the *bubble* velocities when they are about to violate that constraint. Given the left *bubble* of center $\mathbf{l} = (l_x, l_y, l_z)$ and its displacement at the next simulation step $\mathbf{d} = (d_x, d_y, d_z)$, and the right *bubble* of center $\mathbf{r} = (r_x, r_y, r_z)$, the constraint is applied following Equation (3.1). The same constraint is applied to the right *bubble*.

$$\mathbf{d} = (r_x - l_x, d_y, d_z) \quad if \quad l_x + d_x > r_x \ . \tag{3.1}$$

3.1.2 Viewport Adaptation

Since each device is attached to a *bubble* moving independently from the other, they can end up moving very far from each other, possibly out of the initial field of view. A method is thus required to constantly adapt the viewpoint so as to keep both proxies visible. This could be achieved by adapting the viewpoint to the position of both proxies, however this would lead to constant motions of the camera even during precise manipulation tasks. Instead, we adapt the viewpoint to the position of the *bubbles*, which limits the motions of the camera to when one or both of the *bubbles* are actually moving within the VE (Figure 3.3).

This is accomplished by setting the distance between the camera and the center of the both *bubbles* to a value proportional to the distance between the leftmost border of the left *bubble* and the rightmost border of the right *bubble*, plus an arbitrary margin (Figure 3.4).

3.1.2.1 Translational motion

Given the left *bubble* of center $\mathbf{l} = (l_x, l_y, l_z)$ and width w_l, and the right *bubble* of center $\mathbf{r} = (r_x, r_y, r_z)$ and width w_r, the position of the camera is computed following Equations (3.2)-(3.4).

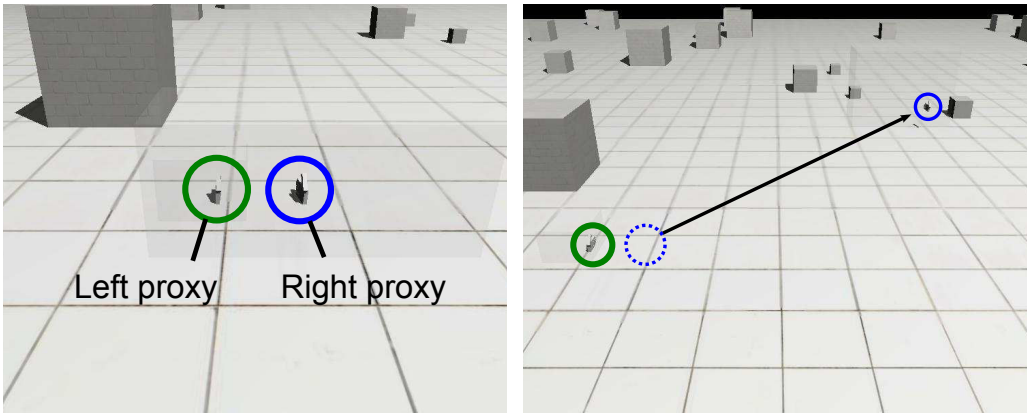


Figure 3.3 – Result of the viewport adaptation with different relative positioning of the proxies (circled).

The center of the scene **s** is first computed from both workspace centers following:

$$\mathbf{s} = \left(\frac{l_x + r_x}{2}, \frac{l_y + r_y}{2}, max(l_z, r_z) \right) . \tag{3.2}$$

The width of the displayed scene w_s is then computed from the widths of both workspaces w_l and w_r, as well as an arbitrary margin m that ensures that the virtual workspace boundaries do not leave the borders of the screen:

$$w_s = \sqrt{(r_x - l_x)^2 + (r_y - l_y)^2} + w_l/2 + w_r/2 + 2m . \tag{3.3}$$

Finally, the position of the camera **c** is computed following:

$$\mathbf{c} = \mathbf{s} + w_s d\mathbf{a} . \tag{3.4}$$

where d is a scalar that depends on the camera field of view (larger fields of view requiring smaller distance to the *bubbles*), and **a** is an arbitrary unit vector that determines the angle from which the scene is displayed.

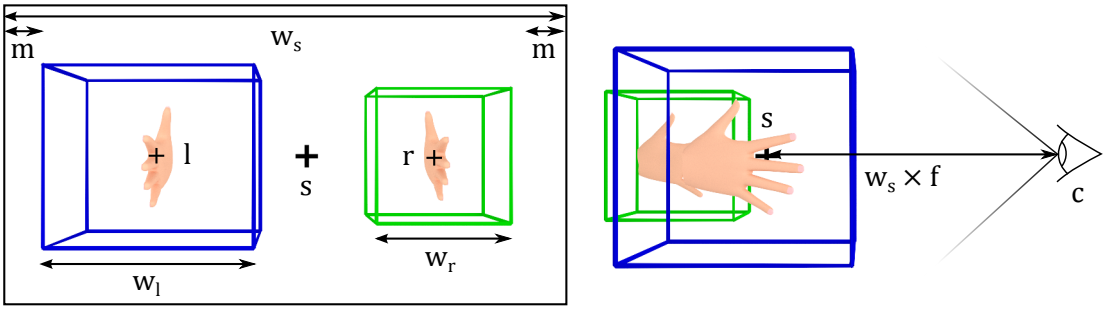


Figure 3.4 – Computation of the camera position for viewport adaptation. **(left)** View of the two *bubbles* from the computed camera position. The points l, r, s are the centers of the left bubble, right bubble and scene, and w_l, w_r, w_s their respective widths. **(right)** Side view showing the camera position c with respect to the center of the scene.

3.1.2.2 Rotational motion

A limitation of this method is that it does not allow rotations of the viewport. A first method to add rotation is to replace some translational DOF of the camera by rotational DOF. For instance, pushing a *bubble* forward could rotate the viewpoint around the vertical

axis between both *bubbles*, while pushing both forward at the same time would translate both bubbles forward. This effectively trades the ability to have two proxies at different depths from the viewpoint for the ability to rotate the viewpoint.

Another way to handle these rotations without sacrificing any DOFs is to use the previously mentioned "separation plane", which prevents both bubbles from crossing each other. Pushing on this plane with a *bubble* allows to extract an angular velocity from the horizontal component of the displacement of the *bubble* (Figure 3.5). That angular velocity can then be applied to the vector **a** so as to turn the viewpoint around the center of the scene. The displacement of both proxies can be used, making the viewport rotation faster if both hands push in opposite directions simultaneously. While not losing any DOFs with this method, it is limited in that both bubbles have to be close to this separation plane in order to initiate rotation.

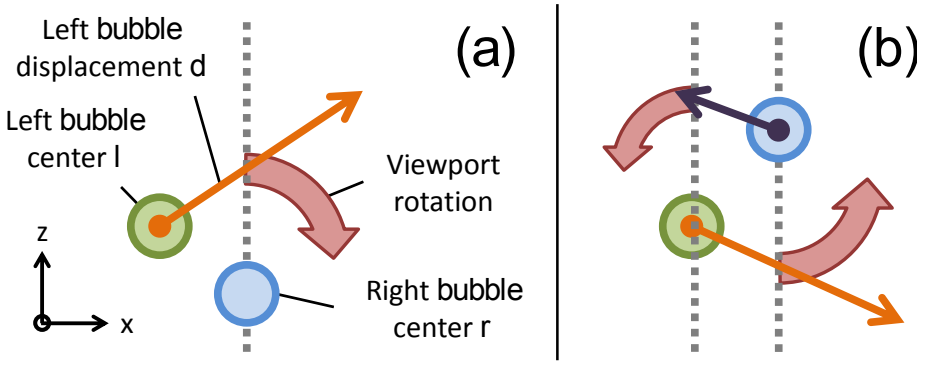


Figure 3.5 – Rotation of the viewport by pushing on the separation plane with the proxies. (a) One-handed case. (b) Two-handed case.

Whichever method is used, the previous equations for translational motion still apply with rotation added, but have to be considered in local coordinates of the camera.

3.1.3 Joint Control

The *double bubble* metaphor may introduce a difference in control modes and/or *bubble* velocities when activated. While this is not a problem in free motion, this may cause some issues when attempting to grasp an object with both proxies (Figure 3.6a-b). Notably, a device with a smaller workspace will tend to reach the boundaries of its bubble faster than the other device. Also, depending on the position of the bubbles, one device could start grasping while very close to the border of its *bubble*, also causing it to reach rate control faster than the other.

In order to reduce the impact of these differences when picking and placing a virtual object, *joint control* was introduced. During grasping, joint control enforces a common control scheme for both bubbles, meaning that both *bubbles* enter rate control as soon as at least one device reaches its boundaries. It also enforces a common velocity during rate control, which is the average of both velocities. This technique thus allows easier exploration of a VE while holding an object between rigid proxies (Figure 3.6c-d).

When the boundaries of the *bubbles* are represented by simple rectangular parallelepipeds, the effective size of the *bubbles* during joint control can be easily computed as the intersection of both bubbles in local coordinates of their respective proxies (Figure 3.7). The visual representation of the *bubbles* on the screen can thus temporarily be changed to reflect this effective size, which in turn leads to a zooming of the camera due

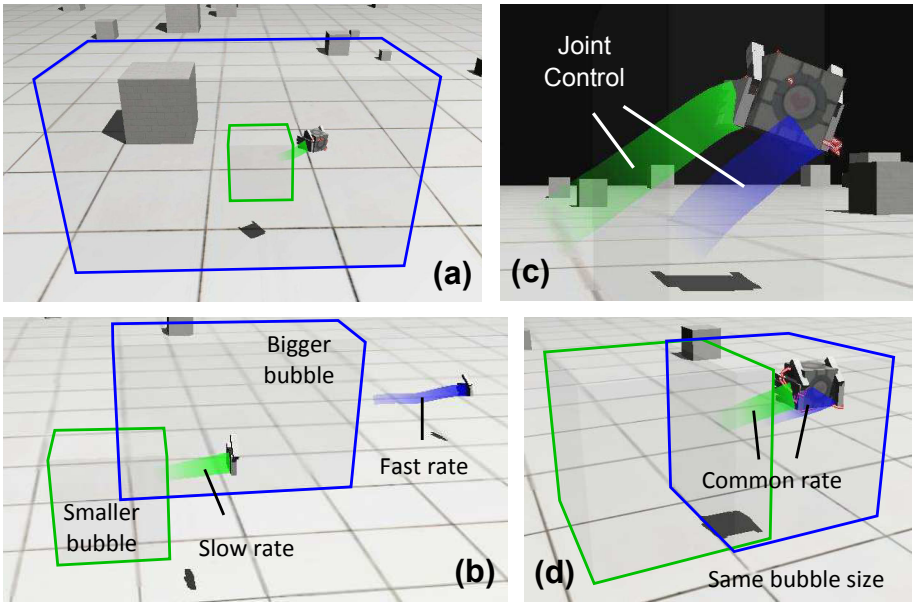
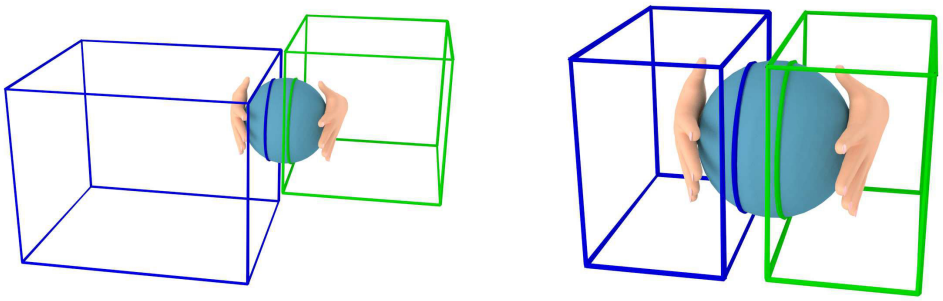


Figure 3.6 – Illustration of *joint control*. **(a)** Issues when carrying an object without *joint control*: the smaller *bubble* is in rate control while the bigger *bubble* is in position control. **(b)** Difference in *bubble* size and workspace translation speed without *joint control*. **(c-d)** Carrying an object with *joint control*.

to the viewport adaptation. The amount of zooming provides additional feedback to the user as of the actual workspace available to them during grasping.



(a) Actual *bubble* sizes **(b)** Adjusted *bubble* sizes

Figure 3.7 – Adaptation of *bubble* boundaries during joint control. The *bubble* sizes are adjusted to the effective workspace of both proxies while grasping, reflecting workspace differences and proximity to the *bubbles* of both devices.

3.2 Bimanual Manipulation of Virtual Objects with Single-Point Haptic Interfaces

Grasping with single-point interfaces is a challenging task, as rigid proxies provide a small amount of contact points that is often insufficient to perform stable grasping. Objects tend to be dropped unintentionally while attempting to move them, which raises the need for manipulation techniques to keep objects attached to the proxies. This, in turn, requires grasping detection techniques, in order to know when to activate and deactivate said manipulation techniques.

3.2.1 Grasping Detection

A grasping detection method is required in order to detect when a user is actually attempting to pick an object and not simply touching it. We consider three conditions to determine whether both hands are grasping an object or not, according to the contact normals, the contact forces, and the relative position of both hands (Figure 3.8):

1. The angle between the contact normals must be under a certain threshold.
2. Both contact forces must exceed a threshold in order to discriminate simple contacts with an object from a true intent of grasping the object.
3. Two cylinders projected from both proxies following the contact normal must intersect. Their radii match the approximate sizes of the proxies by default, but can be tuned to make the detection more or less sensitive.

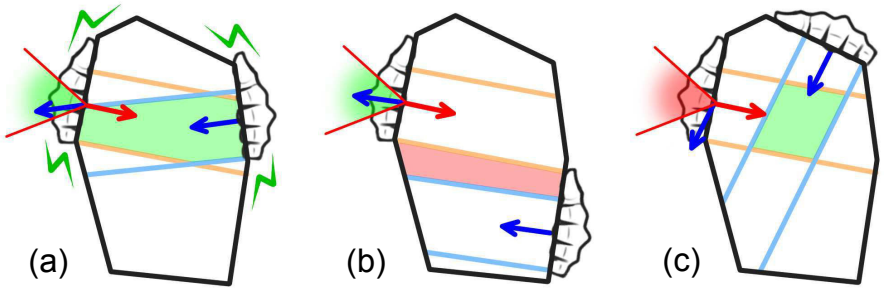


Figure 3.8 – Different cases of dual contact with a virtual object, case (a) being appropriate for grasping and cases (b-c) not being as such: (a) Normals nearly colinear and hands face-to-face, (b) Hands not in front of each other, (c) Normals far from colinearity.

3.2.2 Magnetic Pinch

Once a grasping situation is detected, the *magnetic pinch* takes effect, which "magnetizes" both hands to the picked object to prevent unintentional drops from happening. A visual feedback is also added to the haptic feedback in the form of red bolts, emphasizing the activation of the technique to the user (Figure 3.9). Two implementations were considered for the magnetic pinch: one based on springs and the other based on constraints.

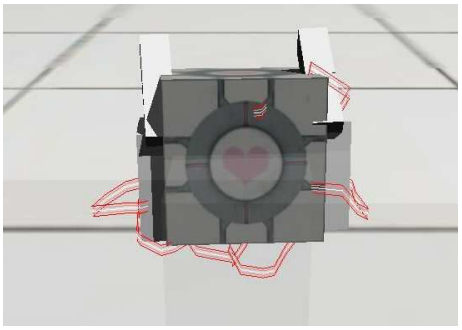


Figure 3.9 – Visual feedback of the *magnetic pinch*, symbolized by red bolts between virtual proxies and picked object.

3.2.2.1 Spring-based approach

The first approach simulates a spring pulling both hands towards the picked object (Figure 3.10a). For each haptic device, a force $\mathbf{F_h}$ is generated following:

$$\mathbf{F_h} = -k_h \left(1 - \frac{g_s}{\|\mathbf{o} - \mathbf{p}\|}\right)(\mathbf{o} - \mathbf{p}) \,. \tag{3.5}$$

Here, $\mathbf{p}$ is the position of the first interface, $\mathbf{o}$ is the position of the second interface, g_s is the size of the grasped object (the distance between the two contact points when the grasping is initiated), and k_h is the stiffness of the spring. The spring is removed as soon as the user provides enough force to end the contact between the proxies and the object, hence dropping the object.

Additionally, the position of the center of mass of the grasped object $\mathbf{g}_p$ can be pulled to the central point between the positions of the two virtual proxies $\mathbf{l}$ and $\mathbf{r}$, further reducing the risk of unwanted drops (Figure 3.10b). This mostly works for simple objects that are grasped around the center of mass, but more complex objects may require using the center position between both proxies at grasping initiation instead of the center of mass for $\mathbf{g}_p$. Whichever method is used, a spring of stiffness k_o is added with a force $\mathbf{F}_o$ following:

$$\mathbf{F_o} = -k_o \cdot \left(\frac{\mathbf{l} + \mathbf{r}}{2} - \mathbf{g_p}\right) \,. \tag{3.6}$$

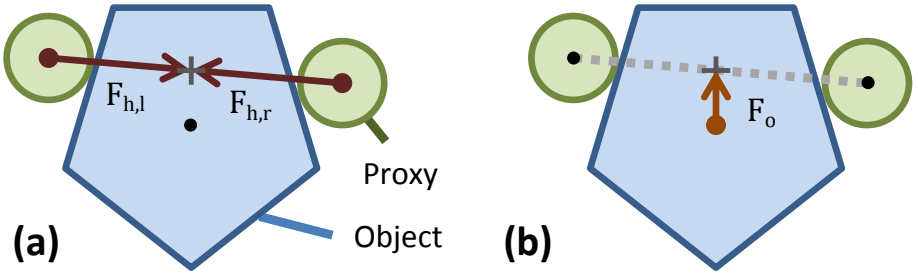


Figure 3.10 – Forces applied by the spring-based approach of the *magnetic pinch*. **(a)** $\mathbf{F_{h,l}}$ and $\mathbf{F_{h,r}}$ are the forces applied on the centers of mass of both proxies to pull them towards each other. **(b)** $\mathbf{F_o}$ is the force applied on the center of mass of the picked object to pull it towards c, the middle point between both virtual proxies.

This approach helps stabilizing the grasping of virtual objects in a soft manner as long as the stiffnesses used are not too high. However, low stiffnesses also make it more difficult to lift heavier objects. The efficiency of this method is thus highly dependent on the choice of stiffness values. Adapting the stiffness to the weight of the grasped object could help to solve this issue.

Moreover, when it comes to rotating an object between both hands, the method is most suited for 6DOF interfaces, which can naturally rotate along with the object. However, it still relies on friction to handle rotation around the grasping axis (along both proxies), which is low due to the low amount of contact points. With 3DOF interfaces, rotating objects between both hands remains challenging, which leads to the constraint-based approach.

3.2.2.2 Constraint-based approach

The second approach constrains the relative position and orientation between both proxies and the picked object. Both constraints are removed as soon as at least one proxy provides an outwards force which value exceeds an arbitrary threshold. This provides a feel of a

stronger grip of the object than the previous approach, as constraints are used to transfer directly forces from the grasped object to the proxies, instead of relying on friction and springs.

This method allows the rotation of these objects around two axes (orthogonal to the grasping axis) with 3DOF devices, by using the relative position between both proxies (Figure 3.11a). Since the relative orientation is constrained, translational motions in opposite directions become angular velocities of the grasped object, and the proxies rotate along with the object even though the devices do not provide any orientation. In addition, if at least one of the two devices has 6DOF input (possibly underactuated), then it is possible to handle rotations around the grasping axis as well (Figure 3.11b).

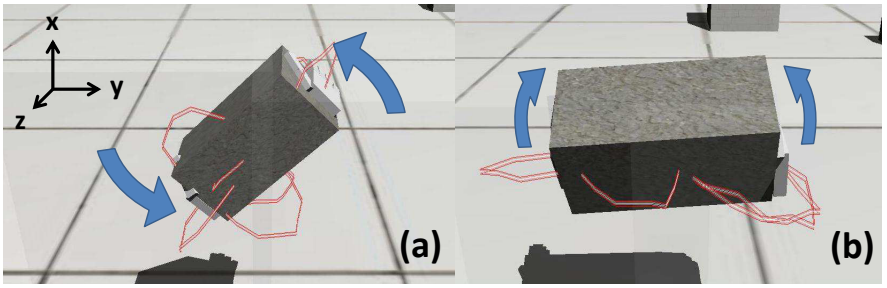


Figure 3.11 – Rotation of a virtual object using the constraint-based approach of the *magnetic pinch*: (a) Rotation around the depth axis using the relative position between both hands. (b) Rotation around the grasping axis using the torque of the right device.

The choice of the threshold determines the weight of the objects that can be picked with the magnet effect: the higher the value, the heavier the object that can be lifted. However, higher values also imply that strong forces must be applied to release even lighter objects, which can give an unnatural feel of being abnormally "glued" to the object in these cases. Once again, this effect could be reduced by dynamically modulating the threshold with the weight of the picked object.

3.3 Evaluation

An experiment was performed to assess the efficiency of the previously mentioned techniques in terms of performance and subjective appreciation in comparison to state-of-the-art methods.

3.3.1 Method

The experiment involved a simple pick-and-place task, where users had to pick a cube and place it on a given target. The *double bubble* technique was compared to the *clutching* technique for workspace extension, and the benefits of the *magnetic pinch* (spring-based approach) and *joint control* were also measured for grasping facilitation.

3.3.1.1 Population

Thirteen participants (2 females and 11 males) aged from 20 to 26 (mean = 22.8, SD = 1.7) performed the experiment. None of the participants had any known perception disorder. All participants were naïve with respect to the proposed techniques, as well as to the experimental setup and the purpose of the experiment.

3.3.1.2 Experimental Apparatus

The participants were seated at 1 m in front of a 24 inch widescreen monitor. The experiment was conducted using two different haptic interfaces. The participants manipulated a Falcon (Novint Technologies Inc.) in their left hand, and a Phantom Omni (Geomagic) in their right hand (used as a 3DOF device), both placed in front of the screen as shown in Figure 3.12. Visual feedback was rendered at a refresh rate of 50 Hz, while the haptic rendering rate was 1,000 Hz. Physical simulation was performed using Nvidia PhysX at a rate of 1,000 Hz to match the update frequency of the haptic loop. A virtual coupling mechanism [Colgate et al., 1995] was used between the haptic interfaces and the virtual proxies by simulating a spring-damper system between each haptic device and its corresponding proxy.

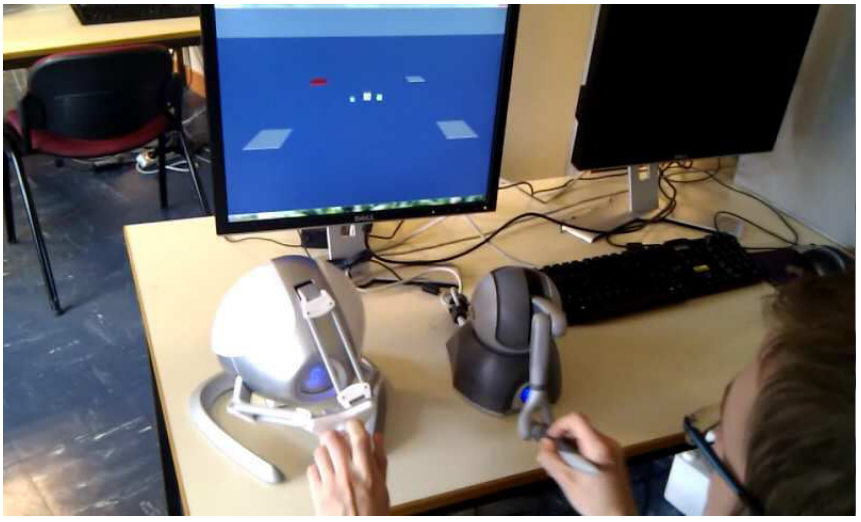


Figure 3.12 – Apparatus used in the experiment. The user manipulates a Falcon (Novint Technologies Inc.) in their left hand, and a Phantom Omni (Geomagic) in their right hand, coupled to rigid proxies in the virtual environment.

3.3.1.3 Virtual Environment

The VE was composed of a 100 m-wide ground plane with four potential target planes, of 1m of width, placed at the corners of a 6 m-wide square around the center of the VE. The target plane of each trial was colored in red, and the other planes were colored in white. The cube to be manipulated had a width of 30 cm and a mass of 3 g, and was placed at the center of the VE. The proxies controlled by each haptic device were physically represented by cubes of 20 cm of width, and were positioned 2 m away from each other and 5 m away from the central cube at the start of each trial. The sizes of the cube, targets and proxies were chosen so as to match a real life task. The distances between them were chosen so as to require both proxies to get out of the workspace borders to reach the cube and targets. The proxy controlled by the left device was visually represented by a blue left hand, and the right proxy was represented by a green right hand. Figure 3.13 shows the scene as displayed at the beginning of a trial.

3.3.1.4 Procedure

At the beginning of each trial, both haptic devices and proxies were set to their starting positions. The subject had to pick the cube from both sides, carry it towards the red

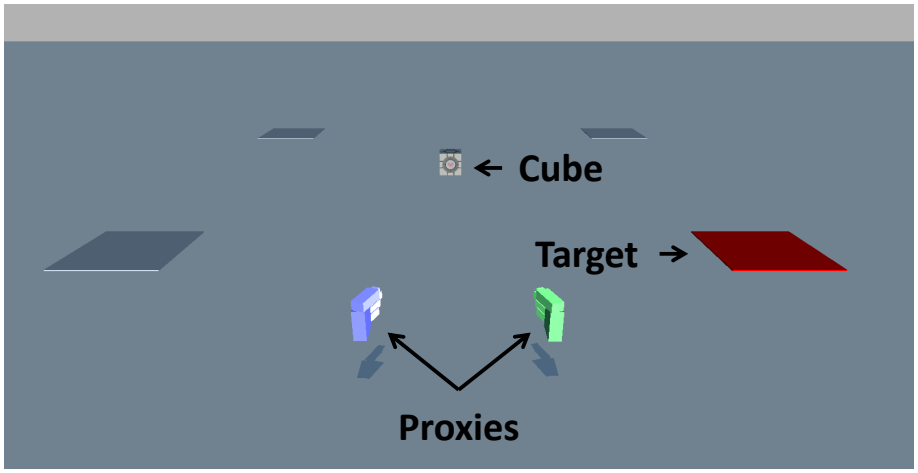


Figure 3.13 – Virtual environment used in the experiment.

target and put the cube on the target, thus ending the trial. A black screen warned the subject about the beginning of the next trial.

3.3.1.5 Experimental Conditions

A within-subject design was used to evaluate the four different conditions:

1. *Ctrl*: Control condition (navigation with clutching, no manipulation techniques)
2. *DB*: Double Bubble (no manipulation techniques)
3. *MP*: Magnetic Pinch (navigation with clutching)
4. *DB+MP*: Double Bubble with Magnetic Pinch and joint control

There were a total of 176 trials, corresponding to 4 conditions × 4 targets × 11 trials. The order between the different conditions was counterbalanced across participants, and for each condition, the order between the targets was randomized. The experiment lasted around 1 hour.

3.3.1.6 Collected Data

For each trial and each participant, the completion time and number of drops were recorded. The completion time is the time elapsed between the moment the proxies leave their starting positions and the moment the cube touches its target plane. The number of drops is the number of hits recorded between the cube and any part of the ground plane that is not the target plane. At the end of the experiment, participants had to complete a subjective questionnaire in which they had to grade the different techniques according to different criteria. The participants could rate the criteria from 1 (very bad) to 7 (very good). The different criteria were: (1) Global appreciation, (2) Efficiency, (3) Learning, (4) Usability, (5) Fatigue, and (6) Realism.

The following questions were also asked to the participants:

1. Have you encountered difficulties during the experiment?
2. Did you perceive differences between the different conditions?
3. What manipulation strategies did you use for the different conditions?

4. Which condition do you prefer and why?

5. If you were to perform a more complicated manipulation task such as manipulating with obstacles or realizing assembly tasks, which condition would you have preferred?

6. Do you have remarks, suggestions?

3.3.2 Experiment Results

Concerning the data, we performed a Shapiro test that rejected the normality hypothesis on the data distribution. Thus, we used a non-parametric Friedman test for differences among the different projections. Post-hoc comparisons were performed using Wilcoxon signed-rank tests with a threshold of 0.05 for significance. Reported p-values are adjusted for multiple comparisons.

3.3.2.1 Completion Time and Number of Drops

A statistical analysis was conducted from the completion time data collected during the experiment. The results are displayed on Figure 3.14. For each participant, statistics (mean M, standard deviation SD) were computed on the 44 trials in each condition. A Friedman test on the completion time (in seconds) revealed a significant effect of the technique ($\chi^2 = 27.66$, $p < 0.001$). Follow-up post-hoc analysis revealed that completion time in both the *MP* ($M = 14.16$, $SD = 7.14$) and *DB/MP* ($M = 8.43$, $SD = 2.91$) conditions were significantly shorter that in the control ($M = 21.41$, $SD = 13.19$) and *DB* ($M = 20.06$, $SD = 14.63$) conditions ($p < 0.001$ in all cases), and that the *DB+MP* condition led to significantly shorter times than the *MP* condition as well ($p < 0.001$).

Similarly, a statistical analysis was conducted on the number of drops for all trials of each participant. A Friedman test showed a significant effect of the technique ($\chi^2 = 25.52$, $p < 0.001$). Post-hoc analysis showed that the *MP* ($M = 4.22$, $SD = 9.45$) and *DB/MP* ($M = 2.36$, $SD = 2.33$) conditions led to significatively less drops than the control ($M = 7.88$, $SD = 6.37$) and *DB* ($M = 8.79$, $SD = 6.77$) conditions ($p < 0.001$ in all cases).

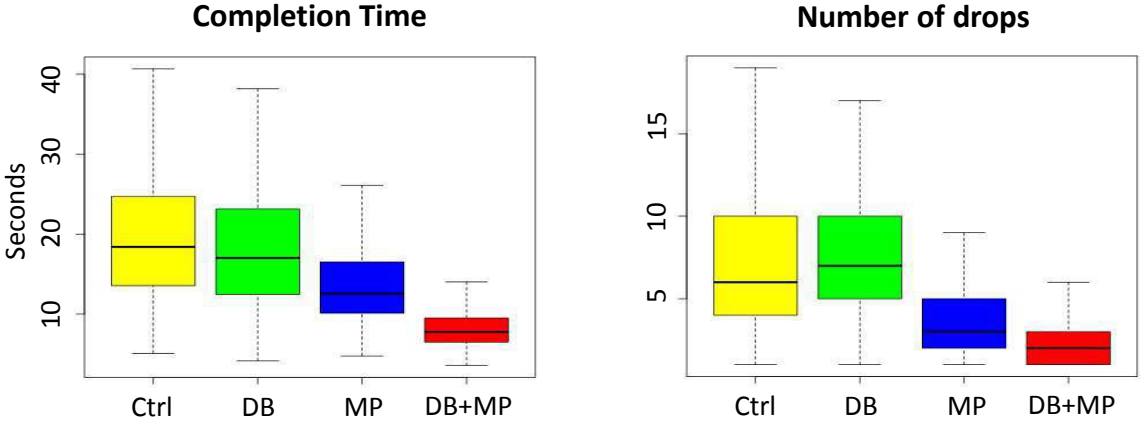


Figure 3.14 – Box plots of the completion times and number of drops for all conditions. They are delimited by the quartile (25% quantile and 75% quantile) of the distribution of the condition over the individuals. The median is represented for each trial.

3.3.2.2 Subjective Questionnaire

The results for the subjective questionnaire are displayed on Figure 3.15. A Friedman test was performed to analyze the answers of the participants. The reported p-values were adjusted for multiple comparisons (alpha-level p=0.05). A significant effect was found for 5 criteria: Global appreciation ($\chi^2 = 4.62$, $p < 0.001$), Efficiency ($\chi^2 = 4.92$, $p < 0.001$), Learning easiness ($\chi^2 = 4.50$, $p < 0.001$), Use easiness ($\chi^2 = 4.80$, $p < 0.001$) and Fatigue ($\chi^2 = 4.46$, $p < 0.001$). No significant effect was found for the Realism criterion.

Post-hoc analysis showed that the *DB+MP* condition was preferred to both the control and *DB* for all criteria: Global appreciation ($p < 0.001$ and $p < 0.001$ respectively), Efficiency ($p < 0.001$ and $p < 0.001$), Learning ($p < 0.001$ and $p < 0.001$), Usability ($p < 0.001$ and $p < 0.001$) and Fatigue ($p < 0.001$ and $p < 0.001$). The *MP* condition was also preferred over the control and *DB* for 3 criteria: Global appreciation ($p = 0.029$ and $p = 0.028$), Learning ($p = 0.032$ and $p = 0.009$) and Usability ($p = 0.027$ and $p = 0.008$), plus a fourth criterion for the *DB*: Efficiency ($p = 0.020$).

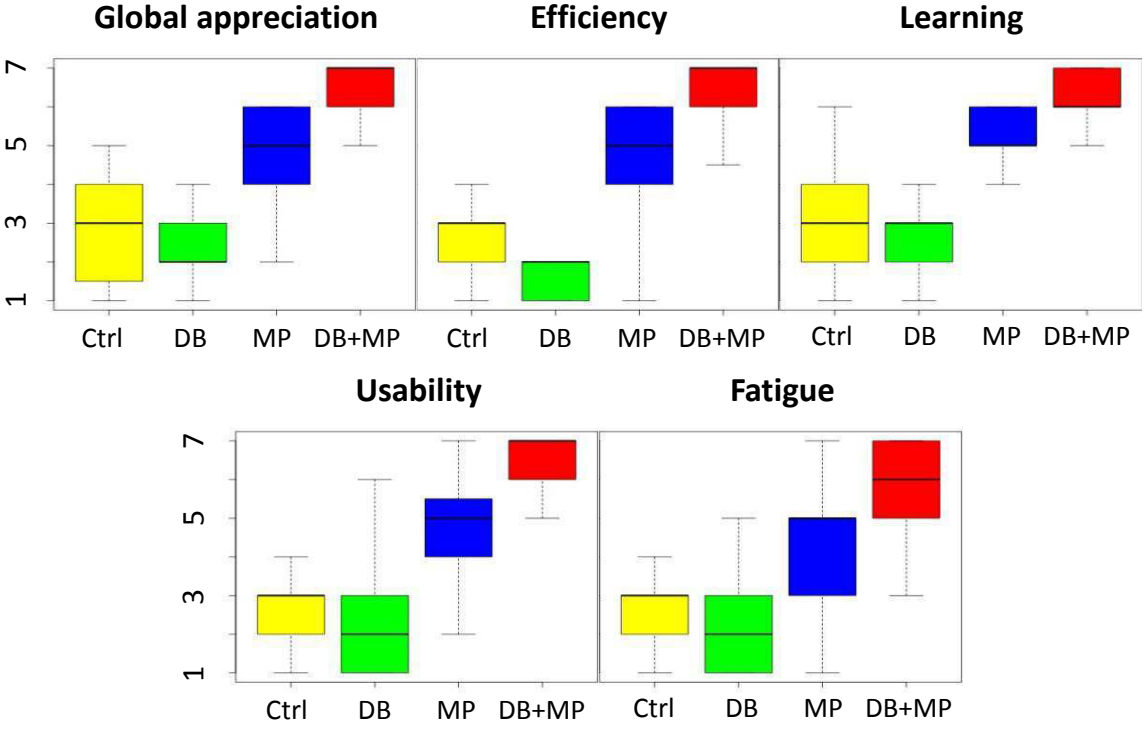


Figure 3.15 – Box plots of the subjective ratings for the significative criteria, for all conditions. They are delimited by the quartile (25% quantile and 75% quantile) of the distribution of the condition over the individuals. The median is represented for each trial.

All participants perceived well the differences between the four conditions. Most participants stated that they preferred the *DB+MP* condition over all the others, reported as faster and easier to use, as well as more efficient and less tiring. They also mentioned this combination as being their condition of choice for more complex manipulation tasks. One participant, however, preferred the *MP* condition, mentioning its behaviour was close to that of a mouse. For the conditions without grasping techniques, it was observed that some participants had a tendency to throw the cube towards the target rather than actually putting it down with both hands like they did with the grasping techniques. It was also noted that the lack of *magnetic pinch* sometimes led users to apply excessively

strong forces on the cube, causing it to drop. A few participants also noted the clutching technique as being overall difficult to use, while others mentioned the difficulty of having two different interfaces regardless of the technique.

3.3.3 Discussion

The experiment showed that the *magnetic pinch* and *joint control* improved performances and subjective appreciation for a pick-and-place task over the *double bubble* and *clutching* navigation techniques, while the combination of these techniques with the *double bubble* led to the best results.

The *double bubble*, used alone, performed as good as the *clutching* technique without outperforming it, in terms of completion time, drop rate, and subjective appreciation. While the technique allows to translate the workspaces in a VE more smoothly than the *clutching* technique by removing the need to move the devices back and forth several times, synchronizing individual velocities in rate control may prove challenging for grasping.

The experiment showed that the *magnetic pinch* and *joint control* significantly improved completion times and reduced dropping rates compared to the conditions that did not use them. In addition, the subjective appreciation also favored the conditions which used these techniques over those that did not. These results strongly indicate that the *magnetic pinch* and *joint control* techniques, by stabilizing the grasping of a virtual object with virtual proxies, are efficient for facilitating pick-and-place tasks. Additionally, while the *magnetic pinch* inherently adds an unrealistic behaviour through the magnetic attraction, it does not seem to hinder the global realism of the scene, as no significant difference in the participants perception of realism was reported for the different conditions.

The *double bubble* showed its full potential when used jointly with the *magnetic pinch* and *joint control*, outperforming the combination of the latter techniques with *clutching*. The *magnetic pinch*, by stabilizing the grasping, also helps to maintain the *joint control* active, which solves the issues encountered with the *double bubble* alone.

3.4 Conclusion

In this chapter, we presented novel interaction techniques to improve bimanual haptic navigation and manipulation of virtual objects with single-point haptic interfaces coupled with rigid proxies. The *double bubble* allows one to move the workspaces of two devices within a VE through hybrid position/rate control, while adapting the viewpoint to the distance between both *bubbles*. The method is designed to be generic, handling any combination of two haptic devices, including different devices in each hand. The *joint control* eases navigation while grasping objects by enforcing common control modes and velocities. When a grasping scenario is detected, the *magnetic pinch* prevents dropping of picked objects using either virtual springs or constraints.

A user experiment showed that the manipulation techniques could lead to faster completion of pick-and-place tasks, with less undesired drops of the object and overall better user appreciation compared to conditions that did not use them. They are thus efficient for simplifying the picking and carrying of an object. The *double bubble*, when used jointly with these techniques using *joint control*, reduced even further the time needed to complete the task, outperforming the *clutching* technique. Overall, the combination of all of these techniques was shown to be efficient for extending the workspaces of different haptic interfaces and enhancing bimanual manipulation with single-point devices in VEs.

God-finger: Rendering of Contact Surfaces

4

Contents

Real life interaction using our hands involves the ability of our fingers to deform in order to generate a contact surface with the objects we touch. In contrast, haptic interaction with physically-based virtual objects often involves the use of rigid proxies which lack this deforming ability of human fingers. As a result, they produce small groups of contact points with virtual objects that barely define a contact surface, which is not sufficient to simulate the friction expected between finger pads and real objects. The use of deformable proxies allows to generate realistic contact areas, with the major tradeoff of requiring soft body simulation methods, which can be more difficult to calibrate as well as more expensive in terms of computation, especially in bimanual scenarios.

We thus propose a method for improving the tradeoff between realistic friction and computational cost, by **generating finger pad-like contact surfaces with rigid proxies**, using a simple heuristic (Figure 4.1). The method, called ***god-finger* method, computes a contact area from the collision information of a single contact point**. This surface is rendered by additional contact points determined from the normal force applied and the geometry of the touched object, making the method less expensive than deformable body simulation and better suited to achieve haptic rates.

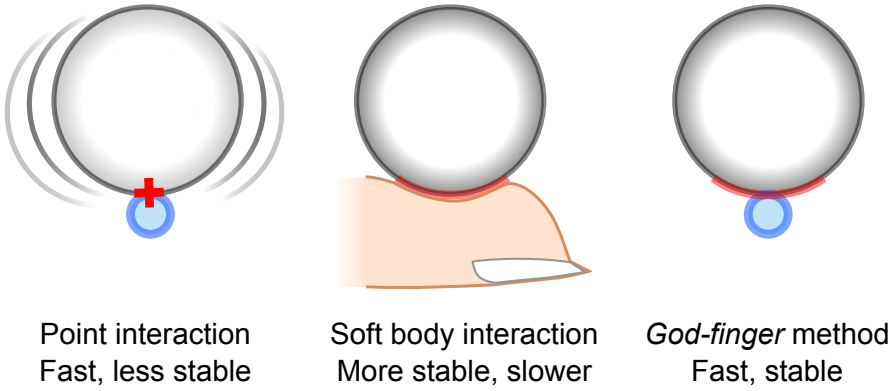


Figure 4.1 – Concept of the *god-finger* method. **(left)** Lifting a virtual object with a single point is challenging, causing the object to eventually fall. **(middle)** Soft body methods simulate the deformation of human fingers under contact, stabilizing those contacts, but are more expensive. **(right)** The *god-finger* method renders a contact surface from a single contact point using a simple heuristic, making it possible to lift the object without dropping.

In this chapter, we present the *god-finger* method with several extensions for specific interaction scenarios. We first describe the core algorithm of the technique, suited for point interaction with rigid objects using 3DOF interfaces. Then, extensions of the method for interaction using 6DOF interfaces and more complex proxies are presented, as well as for interaction with deformable and non-smooth surfaces. We then present some improvements of visual and haptic feedback with the method. Finally, some results of the method on different interaction scenarios are discussed.

4.1 Generating Contact Surfaces from 3DOF Point Contacts

The *god-finger* method computes a contact surface similar to that of a finger pad by scanning the local geometry of a contacting object from a single contact point. This computation is performed in several steps summarized in Figure 4.2. First, a flat fingerprint is generated on the tangent plane to the contact, representing the contact surface that would be expected on a flat surface. It is represented by vectors stemming from the contact point and spanning the surface of the fingerprint, called *radial vectors*. The second step is the projection of this flat fingerprint onto the colliding object, in order to get the actual contact surface. This is done by performing a local geometry scan around the contact point, following the radial vectors. This scan can get prematurely stopped in order to prevent the generated surface from having excessive bumps, keeping it similar to a finger pad contact. Finally, the points resulting from this scan, called *sub-god-objects*, become new contacts between both objects that define the whole contact surface.

4.1.1 Fingerprint generation

Once a contact between the *god-object* and another object occurs, collision detection provides the contact point c_p and normal c_n, which we consider here in local coordinates of the collided object. It also provides information on whether the collision occured on a face, edge or vertex of the object's geometry. Radial vectors are first generated, defining a fingerprint as would be observed if the contact surface was planar.

This process starts by determining the contact radius, which depends on the force

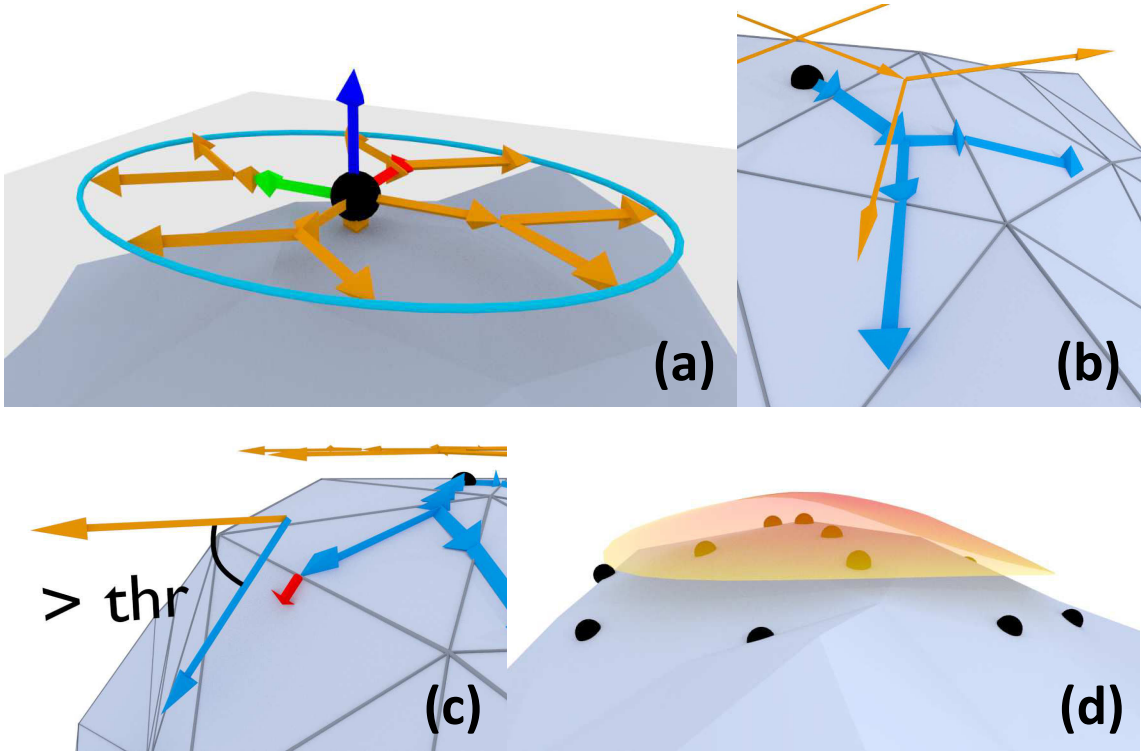


Figure 4.2 – Steps of the *god-finger* method. **(a)** A fingerprint is generated as if the object was planar. **(b)** The contact area is then fit to the geometry of the object. **(c)** Bumps on the generated surface are prevented by stopping the scan at sharp edges. **(d)** The result of the scan provides new contact points which describe the contact surface.

applied by the *god-object* to the virtual object. This force is, due to the virtual coupling, proportional to the distance between the position of the interface and that of the *god-object*. Barbagli *et al.* performed experimental measures of the contact area of the human finger and derived a model between the normal force and the contact radius [Barbagli et al., 2004]. We thus use that model to determine the contact radius r from the distance between the *god-object* position $\mathbf{p_{GO}}$ and the interface position $\mathbf{p_I}$ (expressed in local coordinates), considering an maximum radius of the fingerpad r_{max} and distance constant D'_0 determining how fast the radius increases with normal force. Considering $\mathbf{\Delta p} = \mathbf{p_I} - \mathbf{p_{GO}}$, the contact radius is computed as:

$$r = r_{max}\left(1 - e^{-\mathbf{\Delta p} \cdot \mathbf{c_n}/D'_0}\right). \tag{4.1}$$

Then, given a number n_{SGO} of desired *sub-god-objects* (at least 3), we generate n_{SGO} unit vectors orthogonal to the contact normal, defined as the radial vectors $\mathbf{r_i}$ (Figure 4.3a). If the initial contact occurs on a face, then the radial vectors can be used as is for the following computations. Otherwise, if it occurs on an edge or vertex, the vectors need to be projected onto the surrounding faces (Figure 4.3b). We also define an angular threshold τ_a beyond which we consider that the contact area can not spread anymore over the surface of the object. Thus, if the angle between the contact normal and one of the surrounding faces normals exceeds that threshold, the corresponding radial vectors are dropped.

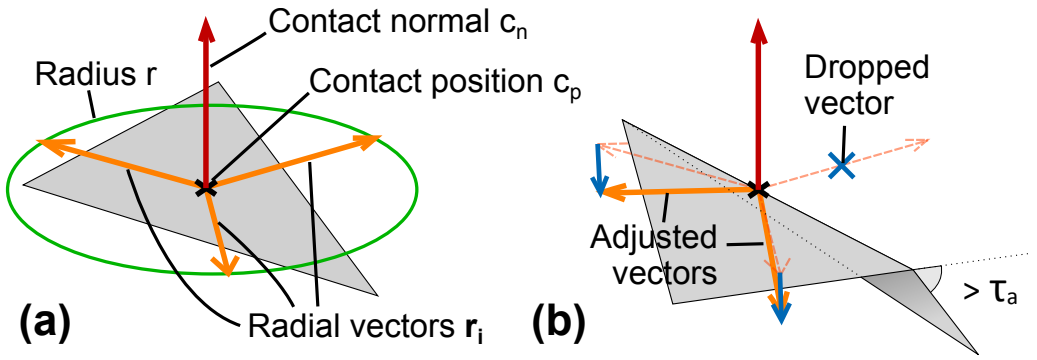


Figure 4.3 – Generation of radial vectors from the contact point (black cross) and normal (red arrow). **(a)** Contact point on a face, the radial vectors (orange arrows) can be used as is. **(b)** Contact point on an edge or a vertex. For faces with normals under a threshold τ_a, the vectors are projected on the faces (blue arrows). For the other faces, the corresponding vectors are dropped (blue cross).

4.1.2 Local geometry scan

Once the initial fingerprint with radial vectors and distances is obtained, the next step is to scan the local geometry of the object to find the positions of the *sub-god-objects*. In order to make the search efficient, the neighbor information is precomputed for the meshes of all virtual objects when they are loaded into the simulation. This means for a triangular mesh that, for every triangle, the indices of the three neighboring triangles (*i.e.*, those sharing an edge with the considered triangle) are stored. The method works the same for both convex and concave virtual objects, making it suitable for interaction with deformable objects as well.

For each radial vector r_i and distance to travel along that vector d_i, the position of the corresponding *sub-god-object* p_{SGO_i} is computed. The search starts at a *sub-god-object* position p_{SGO} equal to the contact position c_p, on a face f with normal f_n determined previously. The object geometry is scanned along r_i over a distance d_i while checking if one of the edges of the triangle was reached. If no edge was reached, then the *sub-god-object* is located on the same face, at the end of the current scan vector. If an edge was reached, however, the *sub-god-object* is positioned on the point of the edge that was reached. Then, it is checked if the normal of the neighboring face is within the angular threshold τ_a defined previously. To take into account friction, it is also checked if it lies within the friction cone of friction coefficient μ. If both conditions are verified, then the geometry search continues, by setting the current face as the neighboring face, and projecting the radial vector r_i on that new face.

This process loops until the full distance was travelled, or until a face with a normal exceeding the angular threshold is found or when the normal is no more in the friction cone. The full algorithm to get the position of a *sub-god-object* $\mathbf{p_{SGO_i}}$ from a starting point $\mathbf{c_p}$ on face f, and a radial vector $\mathbf{r_i}$ with scanning distance d_i is described in Algorithm 1.

Once the *sub-god-object* positions are obtained, they are provided to the simulation engine as additional contacts between *god-object* and colliding object. The simulation then solves the contact as if the *god-object* was a soft finger that had deformed to match the shape of the object.

Algorithm 1 Local geometry scan

Input: Starting point $\mathbf{c_p}$ on face f, radial vector $\mathbf{r_i}$ with scanning distance d_i
Output: *Sub-god-object* $\mathbf{p_{SGO_i}}$

```
p_SGO_i ← c_p
while d_i > 0 and angle(f_n, c_n) < min(τ_a, tan^-1(μ)) do
    r_i,f ← projection of r_i on f
    if an edge e of f is found between p_SGO_i and p_SGO_i + d_i r_i,f then
        p_SGO_i ← point reached on e
        f ← neighboring face
        f_n ← neighboring face normal
        d_i ← d_i − distance between new and old p_SGO
    else
        d_i = 0
        p_SGO_i = p_SGO_i + d_i × r_i,f
    end if
end while
```

4.2 Fingerprint Generation with 6DOF Interfaces and Complex Proxies

The core algorithm presented in the previous section is the most simple version of the algorithm, only handling interaction with points or very simple proxies due to the fact it can only handle one initial contact for generating a contact surface. However, even a single point proxy may generate more than one contact in concave areas depending on the collision detection method used. More complex proxies are bound to generate multiple contact points as well. To handle these cases, we propose an extension of the fingerprint generation algorithm when more than one contact is detected.

Additionally, the base method does not take into account the orientation of the proxy, as it always generates a circular contact surface around the initial contact point. This produces a visual result that does not truly evoke an actual fingerprint. It also fails to provide variations in friction force when sliding against a surface with different finger orientations. We thus also propose a method for generating contact surfaces that are closer to finger pad contacts, by taking into account the orientation of the proxy during 6DOF interaction.

4.2.1 Multiple contact point handling

We consider a *god-object* in contact with another body in the simulation, generating multiple contact points $\mathbf{c_i}$ with contact normals $\mathbf{n_i}$. The radial vectors are first generated on the tangent plane to the average $\mathbf{n}$ of the contact normals $\mathbf{n_i}$. Then, given $\mathbf{c}$ the barycenter of the contact points $\mathbf{c_i}$, the geometry scan is initiated for each radial vector $\mathbf{r_i}$ from the contact point $\mathbf{c_i}$ that is closest to that vector, *i.e.*, that maximizes the dot product $(\mathbf{c_i} - \mathbf{c}) \cdot \mathbf{r_i}$. In order to keep the overall same scan distance from the contact barycenter, the distance d_i travelled for each vector $\mathbf{r_i}$ during the geometry scan has then to be reduced by $(\mathbf{c_i} - \mathbf{c}) \cdot \mathbf{r_i}$. Results of this adaptation are shown in Figure 4.4.

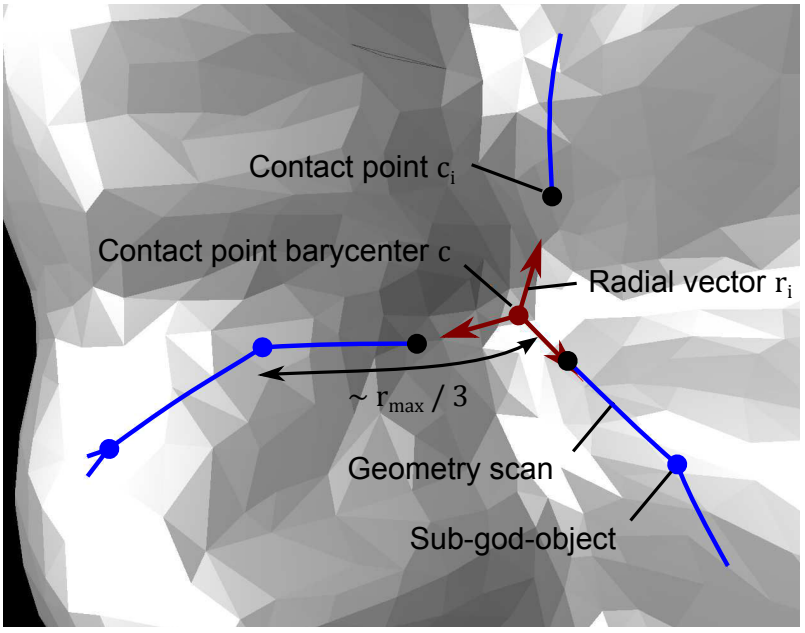


Figure 4.4 – Adaptation of the starting points of the geometry scan with three original contact points. The radial vectors are generated from the barycenter of the contact points, and the geometry scans are started from the contact points closest to each radial vector.

4.2.2 6DOF fingerprint generation

In order to generate more accurate fingerprints with 6DOF interfaces, we propose a modification of the radial vector generation that models the finger as an ellipsoid of dimensions $\mathbf{r} = (r_x, r_y, r_z)$. This ellipsoid, which takes into account the orientation of the device, is then projected onto the tangent plane to provide a fingerprint that takes the form of an ellipse, more similar to the contact surface of a finger pad in the given orientation.

We consider the *god-object* to be a rigid body, with a local frame $B = (\mathbf{b_x}, \mathbf{b_y}, \mathbf{b_z})$, where $\mathbf{b_z}$ is considered the axis closest to the contact normal. Given a contact point $\mathbf{c}$ with normal $\mathbf{n}$ (possibly averaged over multiple contacts), we project the $\mathbf{b_x}$ component onto the tangent plane n to get a unit vector $\mathbf{t}$, which is one of the two axes of the projection of the ellipsoid onto the tangent plane. The other axis is given by the unit vector $\mathbf{s}$, orthogonal to $\mathbf{t}$ on the tangent plane. To ensure consistency between the sub-god-objects between consecutive iterations of the simulation, the first radial vector is generated following $\mathbf{t}$, and the other vectors are always generated in the same order.

Radial vectors $\mathbf{r_i}$ are first generated as usual as a circular fingerprint on the tangent plane, with scan distances $d_i = max(r_x, r_y, r_z) = r_{max}$. This fingerprint is then turned into the desired ellipse by scaling the radial vectors with respect to the tangent frame $(\mathbf{t}, \mathbf{s})$, with the scaling factors $scale_t$ and $scale_s$ computed following:

$$\begin{aligned} scale_t &= \frac{r_x \|\mathbf{b_x} \cdot \mathbf{t}\| + r_y \|\mathbf{b_y} \cdot \mathbf{t}\| + r_z \|\mathbf{b_z} \cdot \mathbf{t}\|}{r_{max} (\|\mathbf{b_x} \cdot \mathbf{t}\| + \|\mathbf{b_y} \cdot \mathbf{t}\| + \|\mathbf{b_z} \cdot \mathbf{t}\|)} \\ scale_s &= \frac{r_x \|\mathbf{b_x} \cdot \mathbf{s}\| + r_y \|\mathbf{b_y} \cdot \mathbf{s}\| + r_z \|\mathbf{b_z} \cdot \mathbf{s}\|}{r_{max} (\|\mathbf{b_x} \cdot \mathbf{s}\| + \|\mathbf{b_y} \cdot \mathbf{s}\| + \|\mathbf{b_z} \cdot \mathbf{s}\|)}. \end{aligned} \tag{4.2}$$

Finally, the distances d_i are scaled by a factor $\|scale_t(\mathbf{r} \cdot \mathbf{t})\mathbf{t} + scale_s(\mathbf{r} \cdot \mathbf{s})\mathbf{s}\|$. This effectively rescales the circular fingerprint into an ellipse that adapts to the orientation of the interface, as shown on Figure 4.5.

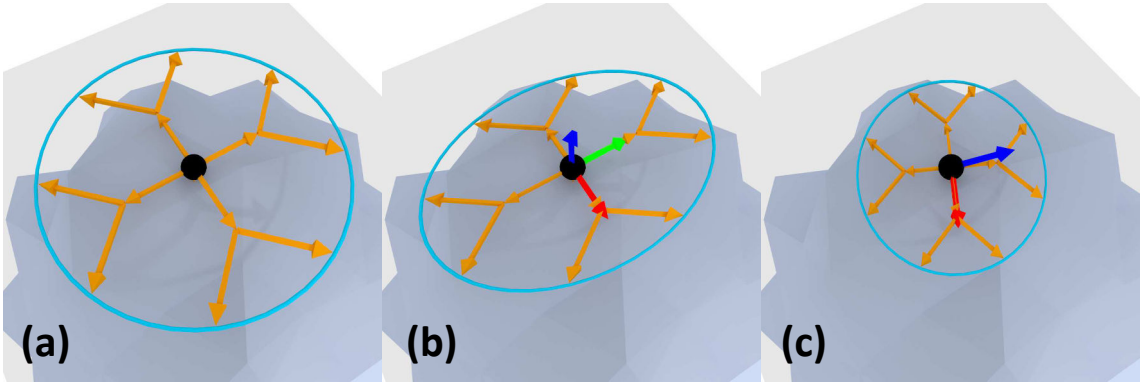


Figure 4.5 – Generation of the initial fingerprint with 6DOF interfaces. (a) The radial vectors are generated as a circular fingerprint. (b) The fingerprint becomes an ellipse with 6DOF interfaces. (c) The orientation of the interface is used to change the shape of the fingerprint accordingly, *e.g.* when tilting the finger forward.

4.3 Local Geometry Scan for Deformable and Rough Surfaces

The base geometry scan algorithm of the *god-finger* method produces contact points along the outline of the contact surface. While this may be a sufficient sampling for contact with rigid objects, it produces an irregular pressure distribution over the contact surface. This may notably cause some artifacts when interacting with deformable objects. We thus propose an alternate scan algorithm that computes iteratively a more uniform distribution of contact points over the surface.

Another limitation of the core method is the case of non-smooth surfaces. Since the geometry scan ends when the angle between the reached face and the contact face is too high, the search may end prematurely when going through rugged areas, which locally have hollows that satisfy the end search condition. The algorithm may therefore be modified to take into account these local irregularities.

4.3.1 Iterative scan

In order to generate more uniform distributions of the sub-god-objects over the contact surfaces, the geometry scans can be performed through a radial tree, adding a *sub-god-object* at each node of the tree (Figure 4.6). The number of iterations and radial vectors can then be increased to provide a finer sampling of the contact surface

We consider a number of radial vectors n_{RV} and a number of iterations n_{it} that defines the density of the *sub-god-objects* in the contact surface. For each radial vector r_i with its associated scanning distance d_i, the regular geometry scan is performed over a distance of d_i/n_{it}. If the search ended by reaching the full distance, a *sub-god-object* is added and the projection of the radial vector on the last face reached is extracted. Two new vectors are obtained by rotating the projected vector around the normal of this face by $\pi/\left(n_{RV}(1+0.2(n_{it}-2))\right)$, and new geometry scans are initiated from the new sub-god-object following these vectors. This process is done iteratively, applying at each iteration i a rotation of $\pi/n_{RV}i$ until either the geometry scan ends before reaching the full distance, or the maximum number of iterations is reached.

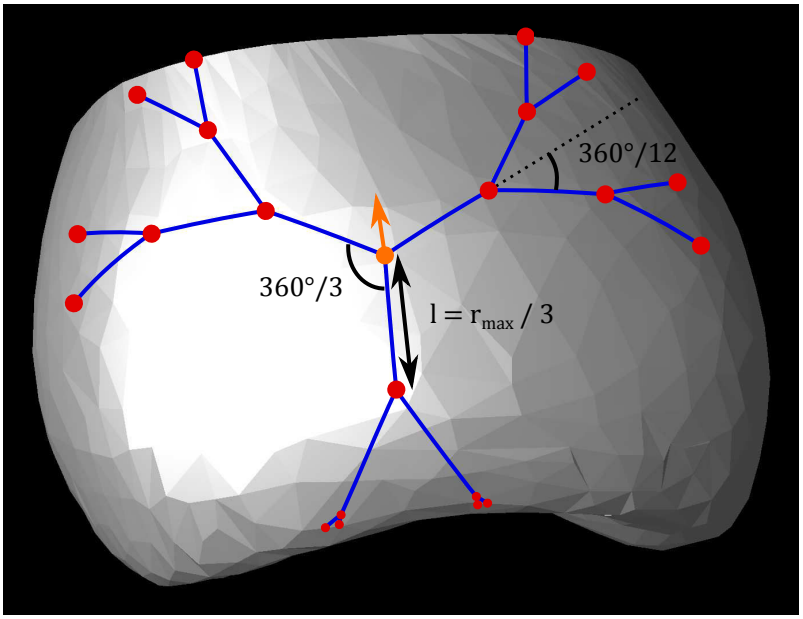


Figure 4.6 – Result of the iterative geometry scan for 3 radial vectors, 3 iterations, and a scan distance of r_{max}

4.3.2 Rough surface compensation

In order to prevent the geometry scan from stopping prematurely when reaching hollows that locally satisfy the end condition, we propose an alternative algorithm that keeps the search going temporarily in these cases. When a face exceeding the angular threshold is met, the search continues until either the full distance is met, in which case the search definitely ends, or a point that satisfies the angular threshold when disregarding the in-between points that did not (Figure 4.7). If such a point if found, then the search continues from that position using the regular algorithm.

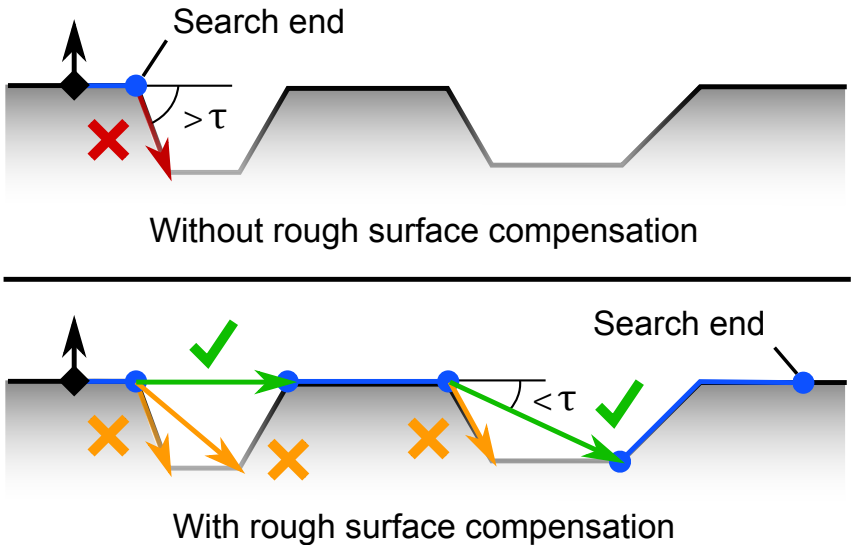


Figure 4.7 – Result of a single geometry scan on a rough surface with and without rough surface compensation. The arrows represent the vectors whose normals are compared to the contact normal.

When the scan initiated by the radial vector $\mathbf{r_i}$ reaches, at step t, a face f_{t+1} which normal exceeds the threshold τ, we consider $\mathbf{p_t}$ the position reached, $\mathbf{n_t}$ the normal of the last face reached f_t, and $\mathbf{s_{t+1}}$ the projection of $\mathbf{r_i}$ on f_{t+1}. We only consider the case where $\mathbf{s_{t+1}} \cdot \mathbf{n_t} < 0$, which corresponds to reaching a hollow, as we do want the search to end when reaching a bump. In this case, the geometry scan is allowed to continue from $\mathbf{p_t}$ by ignoring the angular threshold condition. At each step $t + j$, instead of checking if $angle\,(\mathbf{n_{t+j}}, \mathbf{f_n}) < \tau$ like usual, we instead perform the comparison $angle\,((p_{t+j} - p_t), \mathbf{r_i}) < \tau$. If the full distance is reached before this condition is met, then the search is considered to end at $\mathbf{p_t}$, otherwise it continues from the point that met the condition using the regular algorithm.

The modified algorithm for determining the position of a potential new point $\mathbf{p_{t+j}}$ using rough surface compensation from a starting point $\mathbf{p_t}$, and a radial vector $\mathbf{r_i}$ with scanning distance d_i is described in Algorithm 2.

Algorithm 2 Rough surface compensation

Require: Point $\mathbf{p_t}$ on face f_{t+1}, radial vector $\mathbf{r_i}$ with scanning distance d_i
Ensure: Complete stop or pursuit of geometry scan from point $\mathbf{p_{t+j}}$

```
p_{t+j} ← p_t
f_{t+j} ← f_{t+1}
while d_i > 0 do
  r_{i,f} ← projection of r_i on f_{t+j}
  if an edge e_{t+j} of f_{t+j} is found between p_{t+j} and p_{t+j} + d_i r_{i,f} then
    p_{t+j} ← point reached on e_{t+j}
    f_{t+j} ← neighboring face
    if |p_{t+j} − p_t| ≤ d_i then
      if angle((p_{t+j} − p_t), r_i) < τ then
        d_i ← d_i − distance between new and old p_SGO
        Geometry scan continues with the regular algorithm
      end if
    else
      Geometry scan ends
    end if
  else
    Geometry scan ends
  end if
end while
```

4.4 Improvement of Haptic and Visual Feedback

The god-finger method, by itself, only emulates the presence of a contact surface between a rigid proxy representing a finger and a virtual object. However, finger deformation does not limit itself to the ability to generate contact surfaces, as the finger itself deforms during sliding contact. We then propose a method to improve the haptic feedback of the method by approximating finger deformation using friction. Moreover, since the proxy does not visually deform to match the generated contact surface, there is a lack of visual feedback for the method as well, which can be solved by displaying the outline of the contact surface.

4.4.1 Rendering of finger deformation through friction

Human fingers deform slightly when applying tangential force over the surface of an object, until reaching a point where the finger no longer deforms and sticks to the surface, and further force will lead the finger to slip (Figure 4.8). Previously, these deformations of the finger were handled by deforming the contact surface in the opposite direction of the force applied. Here, we apply a more simple and accurate model to account for this phenomenon: the way the center of mass of the finger moves in certain boundaries while the contact surface remains unmoving is akin to friction. Hence, considering a coefficient e that represents the elasticity of the finger, and the contact radius r_{max}, we modulate the friction coefficients of the god-objects by er_{max}.

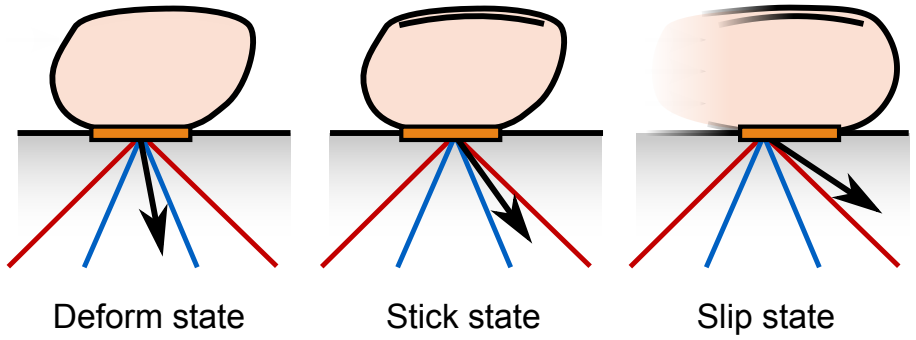


Figure 4.8 – The three friction states of the god-finger method. The contact area is represented by a rectangle and the contact force is represented by an arrow stemming from the contact area.

4.4.2 Visual feedback of the contact surface

The use of a radial tree for the geometry scan allows to display outlines of the contact surface at different pressure levels, corresponding to the consecutive iterations of the local geometry scan. However, it is not possible to simply draw straight lines between the consecutive sub-god-objects of a given iteration, as the lines would clip through the surface of the object in convex areas. Bézier curves can thus be used, by computing tangents on each point, and linking a point to its neighbors through quadratic Bézier curves (Figure 4.9). Another problem that may arise with this approach, though, is that clipping could occur in concave areas. We thus propose a hybrid approach: for convex areas, points are linked to their neighbors using quadratic Bézier curves with tangents, while for concave areas, points are linked using Bézier curves without tangents.

4.5 Results

4.5.1 3DOF interaction with rigid objects

The method can be implemented in any common rigid body physics engine that allows modifications of the contact points between collision detection and contact response. We chose Havok Physics for this evaluation. Concave virtual objects were represented as collections of convex rigid bodies, notably using convex decomposition. Interaction scenarios were tested with two haptic devices: a Falcon (Novint Technologies Inc.) and a Phantom

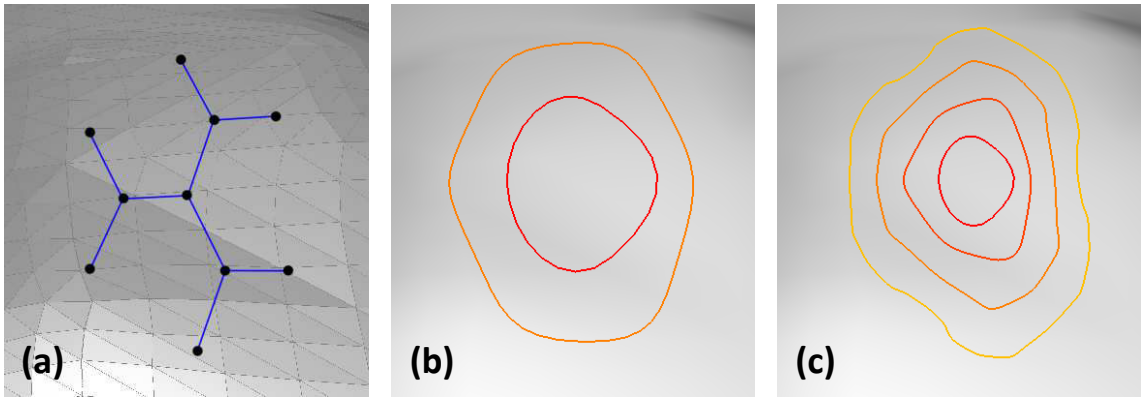


Figure 4.9 – Visual feedback of the god-finger with Bézier curves. **(a)** Geometry scans and sub-god-objects obtained from an initial contact point, with 2 iterations. **(b)** Corresponding visual representation with Bézier curves. **(c)** Visual feedback with the same initial contact and force applied, with 4 iterations.

Omni (Geomagic). The Falcon is a 3DOF device, and the Omni was used as a 3DOF device as well. The haptic devices and corresponding virtual proxies were linked through a virtual coupling mechanism [Colgate et al., 1995], simulating a spring-damper link between them.

The evaluation of the method was performed on a 2.2 GHz quad-core PC, with both haptic devices directly connected to it. Visual rendering was fixed to a 60 Hz rate, and both the physical simulation rates and haptic device update rates were fixed to 1000 Hz, which is a suitable rate for haptic interaction.

4.5.1.1 Unimanual manipulation

Without the *god-finger* method, simply sliding and rotating a spherical object over a flat surface with a single rigid proxy can prove tedious. Notably, the object has a strong tendency to roll as soon as any force is applied, therefore it is extremely difficult to make it only slide while keeping the rolling minimal. The lack of contact surface also tends to amplify the torques applied to the object, making it also challenging to apply rotations of small amplitude. The *god-finger* method allows to better constrain the object, limiting the amount of uncontrolled rolling while still allowing large rotations of the object. It also makes it easier to let a curved object slide along a surface without rolling (Figure 4.10).

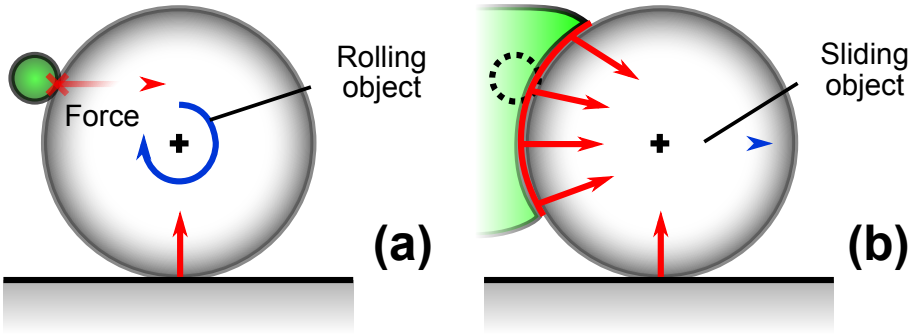


Figure 4.10 – Responses of a virtual object with a regular point *god-object* and with the *god-finger* method. **(a)** With a point *god-object*, the object rolls. **(b)** With the contact surface generated by the *god-finger* method, the object slides.

Another task made possible only with the *god-finger* method is that of lifting objects

with only one interface, when the contact area is sufficiently large compared to the size of the object (Figure 4.11). It is not a trivial task as moving too fast with the interface will cause the god-finger to drop the object. However, this behaviour could be considered realistic, as for instance a hand can not be moved too fast with an object on the palm without dropping it.

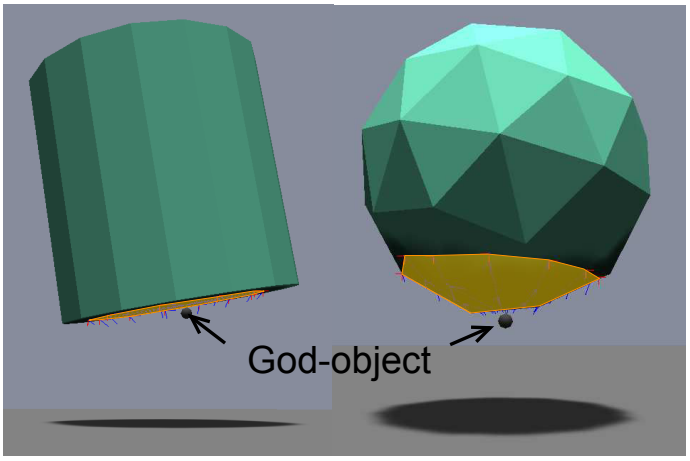


Figure 4.11 – Lifting of virtual objects using a single *god-finger*. The contact areas are represented in yellow.

4.5.1.2 Bimanual manipulation

Rigid objects can be lifted with two interfaces as well with the *god-finger* method (Figure 4.12). Like previously, objects can be lifted from under the object, and in some cases it is also possible to drop one of the interfaces during the lift to switch to a unimanual lifting scenario.

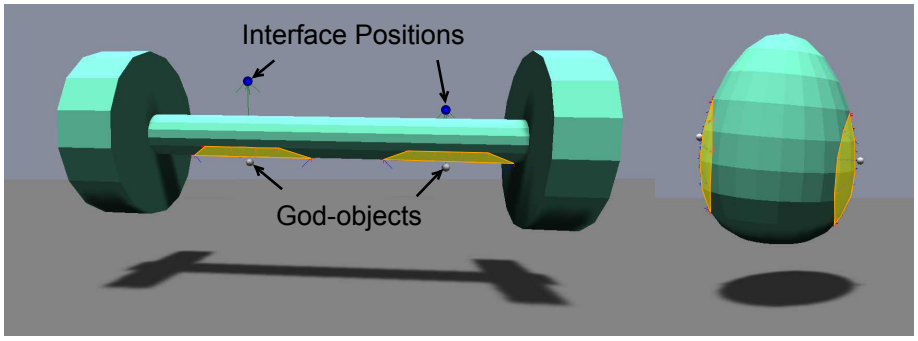


Figure 4.12 – Lifting of virtual objects using two *god-fingers*. The contact areas are represented in yellow.

Objects can also be grasped from the sides with two devices. The use of *god-fingers* instead of regular *god-objects* allows to restrain the torques around the contact normals, thus allowing to pick an object up more easily without resorting to unrealistically high friction values.

The classical *god-object* method and our novel *god-finger* method were directly compared. A cylindrical object was picked by two *god-objects* or *god-fingers* that followed the same motion. First, a fully horizontal motion brings both interface positions inside the object, leading both proxies to pick the object slightly under the center of mass. A vertical motion then causes both proxies to lift the object for 2 seconds, before stopping

their motion completely. The force applied on the center of mass of the object was measured for 25 seconds, and the results are shown in Figure 4.13. With regular *god-objects*, there is no resistance to torques around the contact normals, causing the object to rotate in order to place its center of mass under the grasping axis, and subsequently to start oscillating. These oscillations keep going even long after the lifting is done. With the *god-finger* method, the object never rotates around the contact normals, the oscillations measured at the beginning being only translational and due to the spring-damper of the virtual coupling.

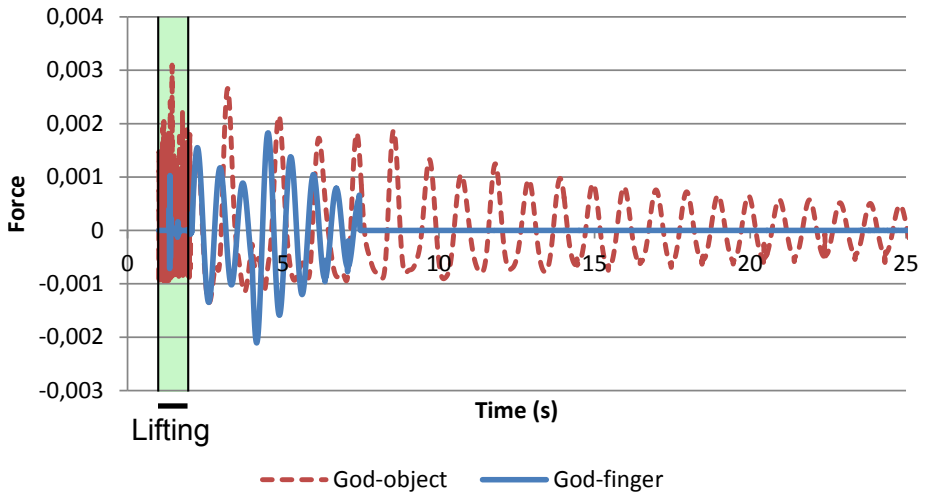


Figure 4.13 – Force (vertical component) applied on the center of mass of a cylindrical object picked and lifted by two *god-objects* or two *god-fingers* along the horizontal axis, slightly under the center of mass.

4.5.2 Performance measurements

The performance of the *god-finger* method was measured using the SOFA framework [Faure et al., 2012], in which the method was implemented. All contacts, including those provided by the method, were solved using a constraint-based approach. The computation time of the whole method was measured, including the fingerprint computation and geometry scans. Simulations were run on a Intel Core i7-2720QM CPU with a Nvidia Quadro 4000M GPU.

We measured how the method scaled with the number of iterations and initial radial vectors from a single contact point, on a moderately even surface. The computation time of the method was measured, as well as the time with respect to the total time taken by the simulation step. In this case, the geometry scans were performed sequentially, with no parallelism used to speed up the computation. Results are shown in Figure 4.14.

As would be expected, the method scales linearly with the number of geometry scans, with an initial cost of around 44 μs and each additional scan taking around 3.6 μs. For the minimal amount of scans required for the method to work, *i.e.*, 1 iteration and 3 radial vectors, the method takes as little as 3.6% of the total simulation time. The method takes the most relative time at around 35 scans (3 iterations and 5 radial vectors), taking around 8.5% of the simulation step. Beyond that point, the method starts taking less relative time as an increasing amount of time is taken for the resolution of the contact constraints added by the method.

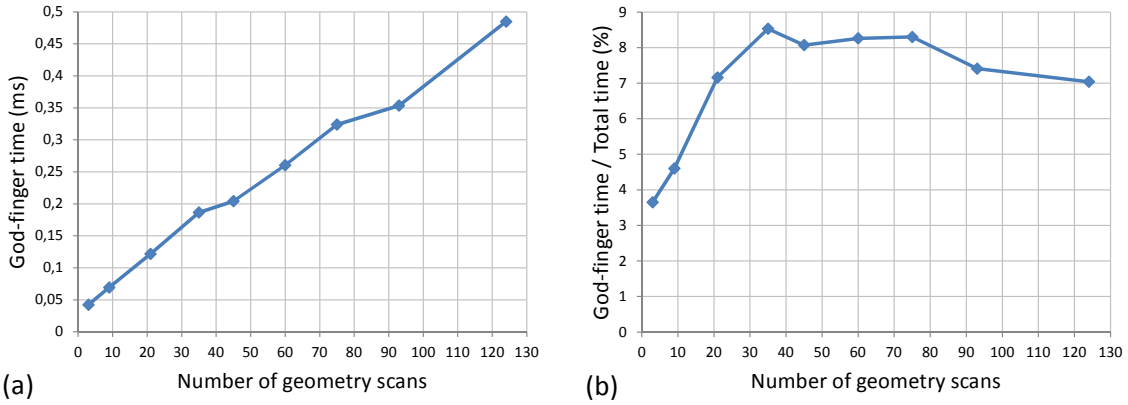


Figure 4.14 – Scaling of the computation time of the *god-finger* algorithm with the number of geometry scans. **(a)** Absolute time. **(b)** Relative time to the total simulation step.

4.6 Conclusion

We have proposed a *god-finger* method for rendering finger pad contact surfaces using information obtained from rigid proxy contacts. The contact area is generated by scanning the local geometry of the object over a radius depending on the force applied to the object, and stopping at positions that would lead to unnatural deformations or friction. The most basic algorithm allows 3DOF haptic manipulation of rigid objects using point proxies, with an increased stability provided by finger pad contacts.

Extensions of the base method allow interaction with 6DOF interfaces and complex rigid proxies that generate more than one contact point. Iterative scanning can be used to provide a finer sampling of the contact surface, which provides more convincing results with deformable objects, and an alternate algorithm allows to handle rough surfaces as well. Finger deformation under sliding contact can be approximated by modulating friction values, and visual feedback can be provided by displaying the outline of the contact surface with Bézier curves.

The method maintains high simulation rates due to the non-requirement for soft body simulation. It has a low cost that scales linearly with the number of scans performed, though it could be improved even further by parallelizing the scans. Similarly to soft body simulation methods, the computational cost of the method becomes more and more negligible compared to the constraint resolution times when finer sampling is used.

Dexterous Manipulation with Soft Fingers

5

Contents

One of the most natural and immersive ways of interacting with physically-based virtual environments is the use of virtual hands that mimic the behavior of our own hands. Dexterous virtual object manipulation is however a complex task that requires appropriate models of hand and contact mechanics to be simulated in real time. A common approach in virtual grasping is to model the hand as an articulated body made of rigid phalanges, which is a computationally efficient method that has the main issue of underestimating contact surfaces. The unrealistic ridigity of such models leads to more difficult grasping of virtual objects, notably when picking an object from sharp edges. Moreover, forces tend to be larger than necessary, often leading to sticking friction artifacts.

Deformable fingerpads offer a solution to better simulate friction between a hand model and virtual objects, but modeling soft fingers has its own difficulties as well. The softness of the fingerpads leads to a fast expansion of the contact surface at initial contact, which makes the resulting grasping more stable, but also increases the number of contact points, making computation more intensive. Moreover, precise friction models need to be computed at each contact point, which further increases the complexity of the system to solve. As a result, the solvers for frictional contact mechanics with deformable objects scale poorly with the resolution of the objects and the number of contacts.

Allard et al. [2010] introduced volume contact constraints for improving the performance of contact handling with deformable objects. The method used volumetric collision detection in order to formulate contact constraints as penetration *volumes* between objects instead of penetration *distances* over several contact points. The number of constraints is thus no longer dependent on the resolution of the collision meshes, but instead on a regular grid which determines how the penetration volumes are divided.

Despite its benefits in terms of efficiency, this approach suffers from two important limitations for grasping simulation. The first one is that pressure is uniform at all contact points aggregated into the same volume constraint, ignoring differences between contact against a flat surface, a curved surface, or a sharp edge. The second one is that, per penetration volume, friction acts only on linear velocities, not accounting for twists over the contact surface. In order to retain some pressure distribution as well as torsional friction between objects, the number of volume constraints needs to be increased, which in turn decreases the efficiency of the method.

In this chapter, we introduce novel aggregate constraints inspired by the work of Allard *et al.* for simulation of dexterous grasping. We augment their volume contact constraint formulation in multiple ways, aimed at improving the computation of hand and contact mechanics during grasping, while maintaining efficient resolution of penetrations and realistic friction. We first describe the formulation of volume contact constraints from point contact-based collision detection. We then introduce a method to compute a non-uniform pressure distribution within aggregate constraints, which adapts to the local geometry of the objects in contact, capturing the grasping differences between flat and sharp surfaces. Then, we detail the addition of torsional friction to aggregated volume contact constraints, with a minimal added computational cost. Finally, we describe a deformable hand model for manipulation of virtual objects using our constraint method, and we evaluate the method on several interaction scenarios.

5.1 Volume Contact Constraints

In the simulation of grasping using deformable fingers, fingerpads adapt smoothly to the surfaces of grasped objects. The resulting large contact areas help stabilize grasping, but also induce a large number of point contacts that affect simulation performance, because with typical constrained dynamics methods the number of constraints is proportional to the number of contacts. Instead, we build on volume contact constraints [Allard et al., 2010] as a more efficient way to handle soft-finger contacts. With our approach, the number of constraints scales with the number of phalanges in contact and is independent from the discretization of collision meshes.

We overcome limitations of previous volume constraint methods by integrating torsional friction within the individual constraints, using the Coulomb-Contensou friction model. We also integrate a non-uniform distribution of forces over the contact surface within the formulation of the constraints, based on the penetration between objects during unconstrained motion. By doing so, we obtain results similar to those obtained with a multi-volume contact constraint approach, while removing the need to divide the contact volumes using a regular grid, thus further reducing the number of constraints involved to a total of 4 per phalanx. This section describes the formulation of these constraints.

5.1.1 Numerical Integration with Constraints

Before describing our approach to volume contact constraints, we first formulate the dynamics of a physical system with generic constraints. According to Newton's second law, the dynamics of a discretized body within the simulation follows:

$$\mathbf{M}\dot{\mathbf{v}} = \mathbf{f}(\mathbf{q}, \mathbf{v}) + \mathbf{f_{ex}}, \tag{5.1}$$

where $\mathbf{q}$ is the vector of degrees of freedom, $\mathbf{v} = \dot{\mathbf{q}}$ is the vector of velocities, $\mathbf{M}$ is the

mass matrix, $\mathbf{f}$ provides the internal forces, and $\mathbf{f_{ex}}$ the external forces (due to contact and other constraints). Using implicit Euler integration, and given a time step h and current state $(\mathbf{q_0}, \mathbf{v_0})$, we obtain the linearized system:

$$\underbrace{\left(\mathbf{M} - h\frac{\partial \mathbf{f}}{\partial \mathbf{v}} - h^2\frac{\partial \mathbf{f}}{\partial \mathbf{q}}\right)}_{\mathbf{A}} d\mathbf{v} = \underbrace{h\mathbf{f}(\mathbf{q}_0, \mathbf{v}_0) + h^2\frac{\partial \mathbf{f}}{\partial \mathbf{q}}\mathbf{v}_0}_{\mathbf{b}} + h\mathbf{f_{ex}}, \tag{5.2}$$

The solution $d\mathbf{v}$ is then used to update the velocity: $\mathbf{v}_{t+h} = \mathbf{v}_t + d\mathbf{v}$, and then the position implicitly: $\mathbf{q}_{t+h} = \mathbf{q}_t + h\mathbf{v}_{t+h}$.

We now consider a deformable phalanx with a triangular surface mesh whose vertices collide with triangles of another body. In order to prevent their interpenetration, as well as to simulate friction at contacts, we solve the previous system for both bodies under contact constraints. These constraints provide the external forces $\mathbf{f_{ex}}$, which are defined as the product of $\mathbb{H}$, a matrix of constraint directions, and λ, a vector of constraint force intensities. This leads to the following linear system for two bodies in contact:

$$\begin{aligned} \mathbf{A}_1 d\mathbf{v}_1 &= \mathbf{b}_1 + h\mathbb{H}_1^T\boldsymbol{\lambda}, \\ \mathbf{A}_2 d\mathbf{v}_2 &= \mathbf{b}_2 + h\mathbb{H}_2^T\boldsymbol{\lambda}. \end{aligned} \tag{5.3}$$

Let us define as $\dot{\delta}$ a vector of the relative velocities at the contact points expressed in the frames defined by the constraint directions $\mathbb{H}$. By separating the unconstrained velocities of the bodies, $\mathbf{v}_1^{free}$ and $\mathbf{v}_2^{free}$ (*i.e.* with $\lambda = \mathbf{0}$), from the velocity update due to contact forces, we can express the relative velocities at contact points as:

$$\dot{\boldsymbol{\delta}} = \underbrace{\mathbb{H}_1\mathbf{v}_1^{free} - \mathbb{H}_2\mathbf{v}_2^{free}}_{\boldsymbol{\delta}^{free}} + h\left[\mathbb{H}_1\mathbf{A}_1^{-1}\mathbb{H}_1^T + \mathbb{H}_2\mathbf{A}_2^{-1}\mathbb{H}_2^T\right]\boldsymbol{\lambda}. \tag{5.4}$$

To solve for the velocity update $d\mathbf{v}$ subject to non-penetration constraints of the form $\dot{\boldsymbol{\delta}} \geq 0$ corresponds to solving a mixed linear complementarity problem (MLCP). In our implementation, we do this by first computing the constraint compliance matrix $\mathbb{H}\mathbf{A}^{-1}\mathbb{H}^T$, and then solving the resulting LCP using Projected Gauss-Seidel (PGS) relaxation. More details on this formulation can be found in [Duriez et al., 2006, Erleben, 2007, Otaduy et al., 2009].

The non-penetration constraints defined above act only on relative velocities, and the final position integration may introduce drift over time. We avoid this through a post-stabilization step where non-penetration constraints are solved on positions after the velocity solve. Given a vector $\boldsymbol{\Lambda}$ of constraint force intensities for the position solve, the constrained update of body positions can be expressed as the linear system:

$$\begin{aligned} \mathbf{A}_1 d\mathbf{q}_1 &= h\mathbb{H}_1^T\boldsymbol{\Lambda}, \\ \mathbf{A}_2 d\mathbf{q}_2 &= h\mathbb{H}_2^T\boldsymbol{\Lambda}. \end{aligned} \tag{5.5}$$

Then, non-penetration constraints on relative positions at contact points, $\boldsymbol{\delta}$, can be expressed by summing up the relative positions after the velocity update, $\boldsymbol{\delta}^{vel}$, plus the position update:

$$\boldsymbol{\delta} = \underbrace{\mathbb{J}_1(\mathbf{q}_1^{free}) - \mathbb{J}_2(\mathbf{q}_2^{free})}_{\boldsymbol{\delta}^{free}} + h^2\left[\mathbb{H}_1\mathbf{A}_1^{-1}\mathbb{H}_1^T + \mathbb{H}_2\mathbf{A}_2^{-1}\mathbb{H}_2^T\right]\boldsymbol{\Lambda}. \tag{5.6}$$

We solve the resulting MLCP using the same approach as for the velocity update.

To summarize, the global animation loop for obtaining the velocities $\mathbf{v_{t+h}}$ and positions $\mathbf{q_{t+h}}$ at the next iteration from the current velocities $\mathbf{v_t}$ and positions $\mathbf{q_t}$ thus follows Algorithm 3.

Algorithm 3 Animation loop for velocity and position update with constraints

Input: Current velocities $\mathbf{v_t}$ and positions $\mathbf{p_t}$
Output: Velocities $\mathbf{v_{t+h}}$ and positions $\mathbf{p_{t+h}}$ at next iteration

$\mathbf{dv^{free}} \leftarrow$ solving (5.3) with $\boldsymbol{\lambda} = \mathbf{0}$
$\mathbf{v_{t+h}^{free}} \leftarrow \mathbf{v_t} + \mathbf{dv^{free}}$
$\mathbf{q_{t+h}^{free}} \leftarrow \mathbf{q_t} + h\,\mathbf{v_{t+h}^{free}}$
Compute collision detection
$\mathbb{H} \leftarrow$ constraint directions
$\dot{\boldsymbol{\delta}}^{free} \leftarrow$ constraint violations in velocity
$\boldsymbol{\lambda} \leftarrow$ MLCP solve in velocity using PGS
$\mathbf{v_{t+h}} \leftarrow \mathbf{v_{t+h}^{free}} + h\,\mathbf{A}^{-1}\mathbb{H}^T\boldsymbol{\lambda}$
$\mathbf{q_{t+h}^{vel}} \leftarrow \mathbf{q_t} + h\,\mathbf{v_{t+h}}$
$\boldsymbol{\delta}^{vel} \leftarrow$ constraint violations in position
$\boldsymbol{\Lambda} \leftarrow$ MLCP solve in position using PGS
$\mathbf{q_{t+h}} \leftarrow \mathbf{q_{t+h}^{vel}} + h^2\,\mathbf{A}^{-1}\mathbb{H}^T\boldsymbol{\Lambda}$

5.1.2 Volume-based Separation Constraints

The constrained dynamics algorithm described in the previous section is valid for generic formulations of non-penetration constraints. Typically, such constraints are formulated individually for each contact point, using the separation gap as a metric for non-penetration. In this section, we describe volume constraints instead. In addition, we formulate a single constraint for an entire contact surface, thus dramatically reducing the cost of solving constrained dynamics. Figure 5.1 shows in a schematic way the process for our constraint formulation.

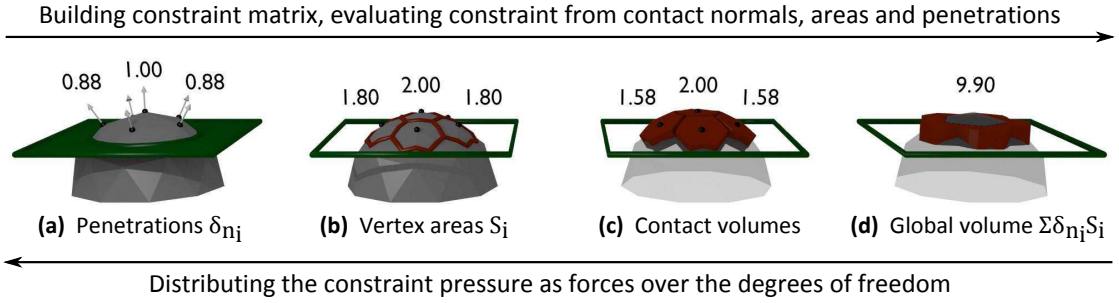


Figure 5.1 – Formulation of separation contact constraints as a volume constraint. **(a)** Two bodies in contact with penetrations evaluated with respect to contact normals. **(b)** Areas computed for every contact point from the local geometry. **(c)** Formulation of contacts as penetration volumes. **(d)** Summing these volumes to formulate a single constraint, which results to a corrective pressure upon solving.

5.1.2.1 Constraint matrix

We denote as $\mathbb{H}_n$ the rows of the constraint matrix corresponding to normal directions at contacts, in contrast to tangential directions to be used later in Section 5.1.4. And we denote as λ_n the normal forces applied by the constraints on the bodies in contact.

In previous work [Allard et al., 2010], penetration volumes between each pair of bodies were provided directly by specific collision detection methods. Here, we resort to more generic collision detection methods, which provide a set of contact points $\mathbf{q}_i$ with penetration distances $\delta_{n,i}^{free}$ and penetration distance gradients (*i.e.* transpose of contact normals): $\mathbf{n}_i^T = \partial\delta_{n,i}^{free}/\partial\mathbf{q_i}$. To define a penetration volume from these contact points, we first compute for each vertex $\mathbf{q}_i$ on the surface of a phalanx an area: $S_i = 1/3\sum_j S_{T_j}$, where S_{T_j} denote the areas of the triangles incident on the vertex. These vertex areas can then be summed to obtain an area for the entire contact surface: $S = \sum S_i$. As a result, we can obtain, for each contact i, the contact volumes and contact volume gradients:

$$\begin{aligned} V_i &= S_i\delta_{n,i}^{free}, \\ J_{V_i} &= \partial V_i/\partial\mathbf{q_i} = S_i\mathbf{n_i}^T. \end{aligned} \tag{5.7}$$

These individual volumes and volume gradients are then summed to formulate an aggregate volume constraint for both contacting bodies: $V = \sum V_i$ and $J_V = \sum J_{V_i}$ (Figure 5.1). For each contact point, a term is added to the constraint matrix as follows:

$$\mathbb{H}_{n,i} = S_i\mathbf{n}_i^T. \tag{5.8}$$

5.1.2.2 Constraint evaluation

Signorini's law is formulated as a complementarity relation (noted $\perp$) between the constraint force and the penetration volume [Duriez et al., 2006]. On the velocity solve this means that the constraint force must be repulsive or null ($\lambda_n \geq 0$), the volume must not increase ($\dot{V} \leq 0$), and the constraint applies pressure only if the penetration volume is null.

$$\lambda_n \geq 0 \perp J_V(\dot{q_0} + \Delta\dot{q}) \leq 0. \tag{5.9}$$

As discussed in Section 5.1.1, the addition of this condition to Eq. 5.3 leads to an MLCP. For post-stabilization, the complementarity condition can be reformulated such that penetration volume is removed:

$$\Lambda_n \geq 0 \perp V(\mathbf{q_0}) + J_V d\mathbf{q} \leq 0. \tag{5.10}$$

5.1.3 Non-uniform Pressure Distribution

With point contact approaches, each one of the contact constraints that sample a contact area undergoes a different contact pressure. Contact points with a larger penetration depth are expected to receive a higher pressure to resolve non-penetration. With the standard volume constraints formulated in Allard et al. [2010], on the other hand, all point contacts aggregated into the same constraint undergo exactly the same pressure. With our basic formulation above, contact pressure is modulated by the area associated to each contact point, but it ignores the amount of penetration (Figure 5.2). Next, we propose a method to support non-uniform pressure distribution within aggregate constraints.

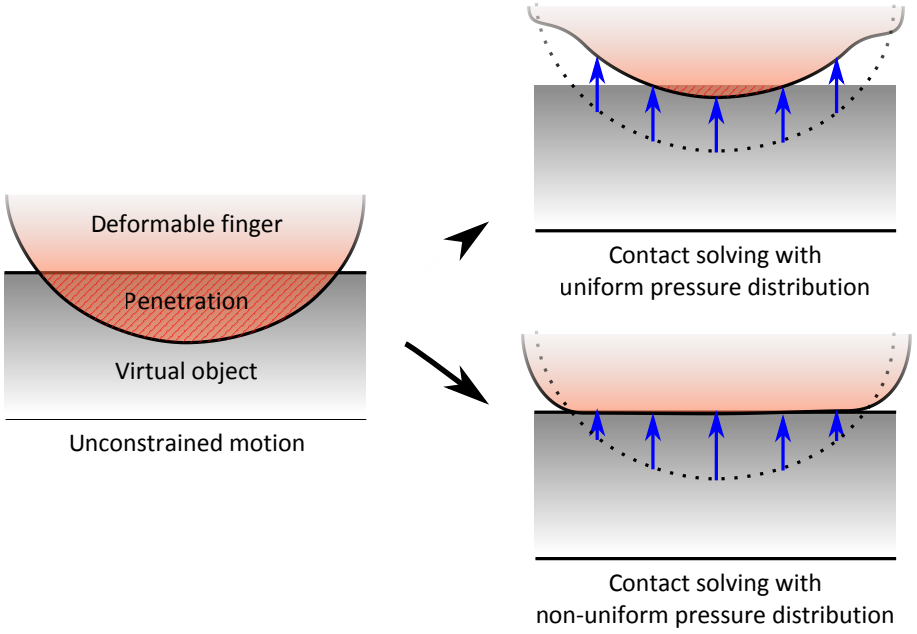


Figure 5.2 – Differences in contact solving with volume contact constraints using uniform and non-uniform pressure distribution.

Given multiple contact points on the skin of a phalanx, the amount of pressure force on each contact point indicates the force needed to enforce non-penetration for that particular contact. Assuming that the tissue in the contact area is roughly homogeneous, and that the object under contact is also homogeneous around the contact area, removing the contact forces would produce larger penetrations on points with higher contact pressure. Based on this observation, we have designed a method that applies a non-uniform distribution of pressure within each aggregate volume constraint as a function of the interpenetration during constraint-free motion.

Specifically, for an aggreate constraint with n_j contact points, we compute for each contact point i a weight w_i proportional to its penetration distance:

$$w_i = n_j \delta_{n,i}^{free} / \sum_{j=1}^{n_j} \delta_{n,j}^{free}. \tag{5.11}$$

We use these weights to rescale the computation of penetration volumes in Eq. 5.7 as:

$$\mathbb{H}_{n,i} = w_i S_i \mathbf{n}_i^T. \tag{5.12}$$

And as a result the entries of the constraint matrix from Eq. 5.8 are also rescaled as:

$$\begin{aligned} V_i &= w_i S_i \delta_{n,i}^{free}, \\ J_{V_i} &= w_i S_i \mathbf{n_i}. \end{aligned} \tag{5.13}$$

5.1.4 Friction Constraints

When an object is grasped with two fingers, torsional friction between the fingerpads and the object plays a key role in maintaining grasping stability. Using point contact constraints is sufficient to model regular tangential friction, and torsional friction is naturally

obtained by accumulating all forces. With volume constraints, on the other hand, a naïve application of point-like friction constraints ignores all torsional friction.

We propose to incorporate an aggregate torsional friction constraint to each aggregate volume constraint, in way similar to regular tangential friction constraints, using the Coulomb-Contensou friction model [Contensou, 1963, Leine and Glocker, 2003]. For simple deformable tissue such as finger phalanges, this allows to apply both tangential and torsional friction using a single set of 3 constraints per phalanx. The aggregation of torsional friction constraints is depicted in an example in Figure 5.3.

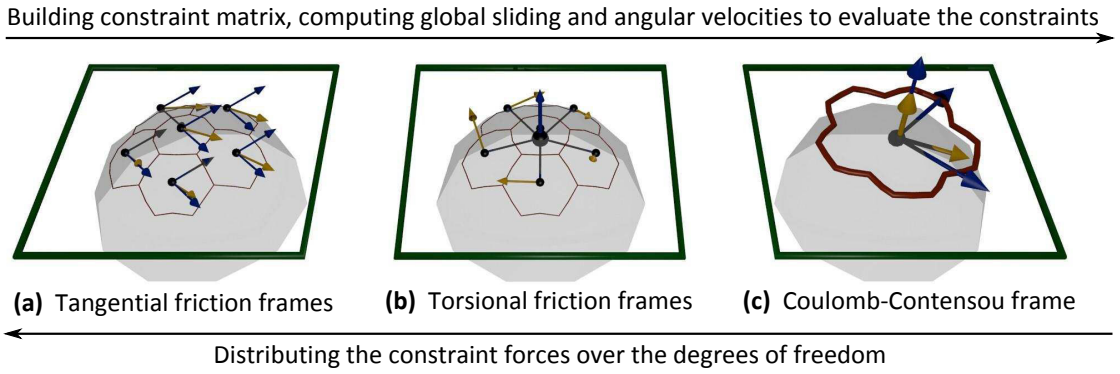


Figure 5.3 – Formulation of Coulomb-Contensou friction for the contact surface with our approach. **(a)** Individual Coulomb friction frames with sliding velocities at each contact point. **(b)** Torsional velocities evaluated from vectors orthogonal to the contact surface normal and lever arms between the contact points and their barycenter. **(c)** Summing these individual frames to evaluate linear and angular velocities for the contact surface.

5.1.4.1 Constraint matrix

Tangential friction is expressed as two constraints spanning the tangent plane of the contact surface, represented by two unit vectors $\mathbf{t}$ and $\mathbf{s}$ orthogonal to the average volume contact normal $\mathbf{n}$. For each contact point, two terms are added to the constraint matrix as follows:

$$\begin{aligned} \mathbb{H}_{t,i} &= \frac{S_i}{S}\mathbf{t}^T, \\ \mathbb{H}_{s,i} &= \frac{S_i}{S}\mathbf{s}^T. \end{aligned} \tag{5.14}$$

To model torsional friction, for each aggregate constraint we compute an area-weighted centroid of the contact points, $\mathbf{p_c} = \sum S_i \mathbf{p}_i / S$, and an area-weighted contact normal $\mathbf{n_c} = \sum S_i \mathbf{n}_i / S$ (followed by normalization). For each contact point in the aggregate constraint, we define a lever arm $\mathbf{r_i} = \mathbf{p}_i - \mathbf{p_c}$. Torsional friction terms are then added to the constraint matrix as follows:

$$\mathbb{H}_{\omega,i} = \mathbf{n_c} \times \mathbf{r_i}. \tag{5.15}$$

5.1.4.2 Constraint evaluation

Regular Coulomb friction aims to maximize the dissipation of relative tangential velocities subject to a constraint on the required tangential force, which should be smaller than a friction coefficient μ times the normal force. The Coulomb-Contensou model extends this principle while taking into account the relationship between tangential and torsional

friction: the faster an object slides along a surface, the less inclined it will be to rotate around the contact normal, and vice versa.

The application of the Coulomb-Contensou model to our aggregate constraint setting requires the evaluation of relative angular velocities between the objects in contact. First, for each contact point we compute an angular velocity with respect to the torsional friction frame defined in Equation 5.15, using its relative angular velocity $\dot{\delta}_{\omega,i}^{free}$ and its lever arm $\mathbf{r_i}$ (Figure 5.3). We estimate the relative angular velocity of the aggreate contact as the area-weighted sum of individual velocities:

$$\omega^{free} = \left(\sum_i \frac{S_i}{S} \frac{\mathbf{r}_i \times \dot{\delta}_{\omega,i}^{free}}{\|\mathbf{r}_i\|^2} \right) \cdot \mathbf{n}. \tag{5.16}$$

The constraints of the Coulomb-Contensou model through a velocity potential are defined in Leine and Glocker [2003]. Given the sliding velocity at each contact point i, computed as $\mathbf{v}_s = \dot{\delta}_t^{free}\mathbf{t} + \dot{\delta}_s^{free}\mathbf{s} + \omega^{free}\mathbf{n} \times \mathbf{r_i}$, the velocity potential is:

$$\Phi = \sum_i \frac{S_i}{S} \mu \lambda_n \|\mathbf{v_s}\| \tag{5.17}$$

The admissible values for the tangential and torsional friction forces are then computed from derivatives of this velocity potential w.r.t. the tangential and angular velocity respectively:

$$\begin{aligned} \nabla_{\dot{\delta}_{\vec{T}}^{free}} \Phi &= \textstyle\sum_i \frac{S_i}{S} \mu \lambda_n \frac{\mathbf{v}_s}{\|\mathbf{v}_s\|} \\ \nabla_{\omega^{free}} \Phi &= \textstyle\sum_i \frac{S_i}{S} \mu \lambda_n \mathbf{r_i^*} \frac{\mathbf{v}_s}{\|\mathbf{v}_s\|}, \end{aligned} \tag{5.18}$$

where $\mathbf{r_i^*}$ is the cross-product matrix of $\mathbf{r_i}$.

Coulomb-Contensou friction constraints are defined as a complementarity condition, similarly to separation constraints. For tangential friction, either the friction force is strictly included inside the friction cone defined by the velocity potential derivative, in which case objects stick together, otherwise the objects slip tangentially and the force is on the border of the cone, along the direction of motion. The condition for the friction force $\lambda_{\vec{T}} = \sqrt{\lambda_t^2 + \lambda_s^2}$ is thus added to the resulting NLCP following:

$$\begin{aligned} \dot{\boldsymbol{\delta}}_{\vec{T}} = \vec{0} &\Rightarrow \left\|\lambda_{\vec{T}}\right\| < \left\|\partial_{\dot{\delta}_{\vec{T}}^{free}} \Phi\right\| \text{ (stick)} \\ \dot{\boldsymbol{\delta}}_{\vec{T}} \neq \vec{0} &\Rightarrow \left\|\lambda_{\vec{T}}\right\| = - \left\|\partial_{\dot{\delta}_{\vec{T}}^{free}} \Phi\right\| \frac{\dot{\delta}_{\vec{T}}}{\|\dot{\delta}_{\vec{T}}\|} \text{ (slip).} \end{aligned} \tag{5.19}$$

For the friction torque λ_ω, the condition is formulated in a similar manner, with either torsional sticking when the friction torque is within admissible values, or slipping otherwise. This is added to the MLCP following:

$$\begin{aligned} \dot{\boldsymbol{\delta}}_{\omega} = \vec{0} &\Rightarrow \|\lambda_\omega\| < \|\partial_{\omega^{free}} \Phi\| \text{ (stick)} \\ \dot{\boldsymbol{\delta}}_{\omega} \neq \vec{0} &\Rightarrow \|\lambda_\omega\| = - \|\partial_{\omega^{free}} \Phi\| \frac{\dot{\delta}_{\vec{T}}}{\|\dot{\delta}_{\vec{T}}\|} \text{ (slip).} \end{aligned} \tag{5.20}$$

5.2 Hand model

We propose a deformable hand model that leverages our aggregate constraint method for interactive grasping and dextrous manipulation of virtual objects. The model consists of 5

interconnected layers: tracked hand data, reduced coordinates model, mapped rigid body skeleton, deformable phalanges, and surface collision/visual model (Figure 5.4).

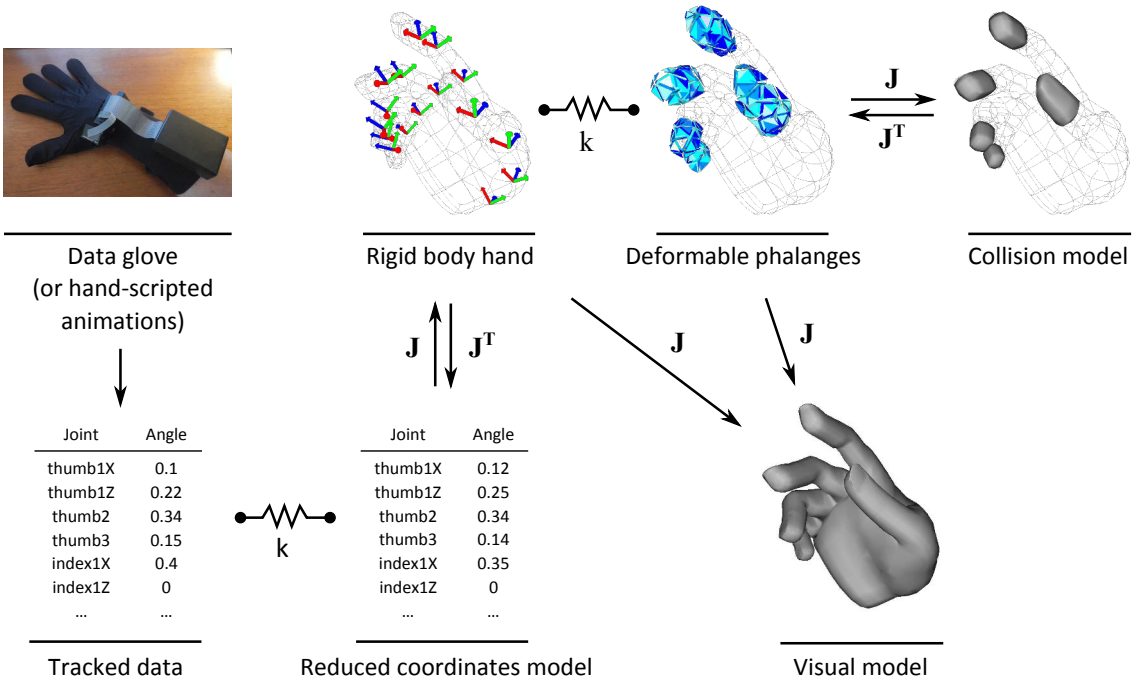


Figure 5.4 – Different layers of our proposed hand model. The reduced coordinates model follows the tracked data with springs, while the rigid body hand is mapped to it. The deformable phalanges are linked to the rigid hand with springs, and the collision models are mapped to them. The visual model is skinned from the rigid body hand and deformable phalanges.

The tracked data is first stored as angular values at all joints of the hand, considering two degrees of freedom at the base of each finger (flexion and abduction) and one degree of freedom for both joints of each finger. The position and orientation of the palm are stored as well, to account for motions of the hand as a whole. The simulated hand is also first modeled in reduced coordinates, with 20 finger joint values and a 6DOF base value. Stiff springs are used on each joint to make this hand model follow the tracked hand as closely as possible. Unilateral springs are used in absence of haptic feedback, but bilateral springs can also be used as a virtual coupling scheme if haptic feedback is provided to the hand and/or fingers. Joints are also assigned minimum and maximum angular limits, which are enforced using stiff springs once either of these values are exceeded.

A mapping function is then used to determine the position of the palm and phalanges of the hand skeleton from the reduced coordinates model. Forces and constraints can be mapped both ways using jacobians and transpose jacobians, as described in Duriez et al. [2008]. This model thus serves as a proxy between the reduced-coordinate model and 3D space. The deformable phalanges are modeled as coarse tetrahedral meshes (1513 tetrahedra for all 15 phalanges), and simulated using linear corotational FEM [Müller and Gross, 2004]. Nodes whose positions match the anatomical position of bones are linked to the mapped rigid body skeleton using bilateral springs.

For handling collisions using our approach, we assign one collision model per phalanx, hence all point contacts acting on the same phalanx are aggregated into a single volume

constraint. Each collision model is a triangular surface mesh which is mapped to its respective deformable phalanx. Similarly to the mapping between the reduced-coordinate model and the rigid body model, the use of transpose jacobians allows to map contacts defined at skin level to the outer nodes of the underlying deformable model. The visual model also uses the same mapping function for the phalanges, and additionally uses regular skinning with linear blending for parts of the hand which are not given a collision model.

5.3 Results

Our approach was implemented into the SOFA framework for physical simulation of both rigid and deformable bodies [Faure et al., 2012]. Simulations were run on a Intel Core i7-2720QM CPU, with a Nvidia Quadro 4000M GPU (used only for visual rendering). For real-time hand tracking, we used a Data Glove 5 (5DT), which provides 1DOF per finger. In addition, a combination of a GameTrak position tracker (In2Games) and a Colibri inertial motion tracker (TRIVISIO Prototyping GmbH) was used to track the 6DOF position of the hand. The limited tracking capability of the data glove allows only simple interactions, such as enclosing and opening motions of the hand (see Figure 5.7c). To demonstrate our methods on more dexterous interactions, we also designed benchmarks with scripted animations, executing the simulations in real time and with the scripted hand configurations substituting the input from the data glove.

5.3.1 Illustrative use cases

Grasping a cube from its edges is a typical scenario showing the need for deformable fingers, as rigid fingers are unable to produce a sufficient contact area for stable grasping. This scenario also showcases the importance of pressure distribution in our contact model (Figure 5.5). With uniform pressure over the contact surface, grasping is not successful, similar to the result with rigid fingers. However, our weighting of contact points within the aggregate constraint formulation shown in Section 5.1.3 restores the pressure distribution that would be expected with point contact constraints, and allows the cube to be lifted.

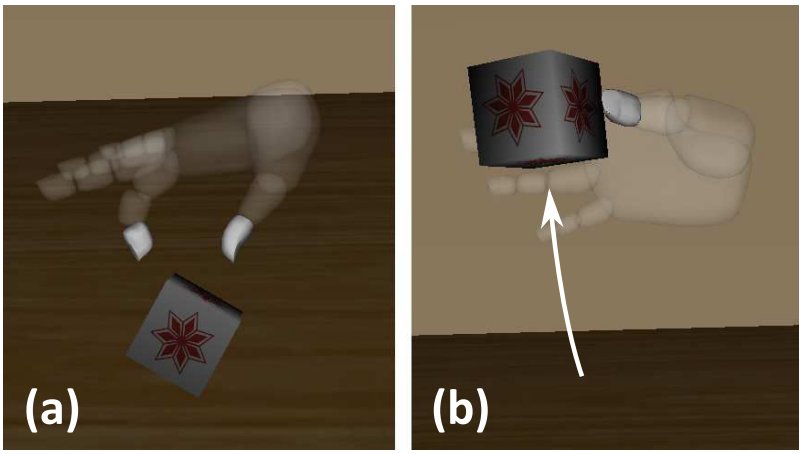


Figure 5.5 – Grasping a cube from the edges with our approach. (a) Uniform pressure distribution is not sufficient to lift the object. (b) Non-uniform pressure distribution allows grasping.

The Coulomb-Contensou friction model accurately represents both the friction forces and torques at fingers in contact, as shown when grasping a long object from one of its

ends with two fingers (Figure 5.6). The object stays in a horizontal position as long as the fingers apply a sufficient force, and good control of rotations around the grasping axis is provided as well.

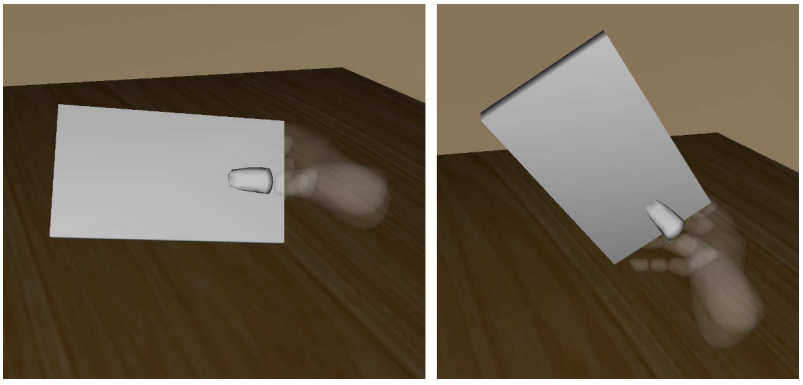


Figure 5.6 – Example of Coulomb-Contensou friction with our approach based on volume contact constraints. The fingers resist to moments around the grasping axis, and allow to rotate objects around that axis.

Our method is suitable for simulating interaction scenarios that would be computationally challenging with point contact forces, such as enclosing and grasping a rigid ball with all fingers deformable (Figure 5.7a), or lifting a dumbbell with two hands (Figure 5.7c). Both scenes present a large number of degrees of freedom and contact constraints. Highly dexterous manipulation is also possible, as demonstrated by a scenario involving the spinning of a pencil (Figure 5.8). Finally, our method is suitable for real-time interaction with data gloves, as shown by a full-hand interaction scenario with a deformable ball (Figure 5.7b).

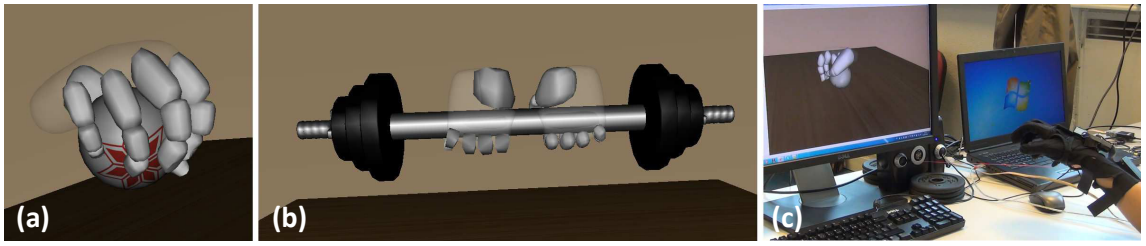


Figure 5.7 – Some object manipulation scenarios with our method. (a) Grasping a rigid ball with all deformable fingers. (b) Manipulating a deformable ball with a data glove. (c) Lifting a dumbbell with two fully deformable hands.

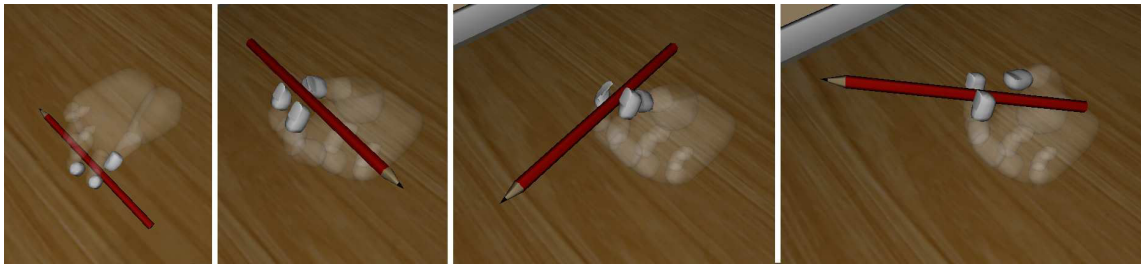


Figure 5.8 – Illustration of our aggregate constraint approach for dexterous manipulation of a pen using soft fingers. The pen is picked and spun with the hand modeled with soft fingerpads.

5.3.2 Computation times

Performance for the aforementioned simulations was measured in terms of computation times for the constraint solve and the entire simulation step w.r.t. the complexity of the scene, measured in terms of the number of deformable phalanges and contact points. The results shown in Table 5.1 have been averaged over 100 iterations.

Table 5.1 – Performance of classical point contact constraints and our aggregate contact constraint approach on different manipulation scenarios.

Scenario	Fig.	Constraints (Phalanges)	Constraints		Constraint solving (ms)		Total time (ms)	
			Point	Agg.	Point	Agg.	Point	Agg.
Pencil spin	5.8	13 (3)	37	12	4.2	2.32	13.88	12.19
Edge grasp	5.5	12 (2)	37	8	9.64	2.23	17.64	10.22
		21 (2)	65	8	23.01	3.45	31.17	11.8
Rigid ball	5.7a	27 (15)	81	23	24.73	9.44	59.98	45.81
		176 (15)	528	51	223.69	54.75	262.13	93.63
Dumbbell	5.7c	86 (30)	251	96	130.45	68.45	201.86	147.03

In the edge grasping scene, only the last phalanges of the thumb and index were simulated. Two slightly different grasps were used to compare performance under a varying number of contact points but the same number of phalanges in contact. With 12 contacts on average, constraint solving times were reduced by 77%, a reduction that increased to 85% with 21 contacts. The overall simulation time was similarly reduced by 42% and 62% respectively.

For the rigid ball grasping simulation, the hand had all phalanges simulated. Two cases were considered: a first one where the ball is lifted with the fingertips, generating 27 contacts in average during grasping, and a second one where the hand grasps the ball more closely, with more phalanges involved and 176 contacts in average. In both cases, the simulation result is the same as with point contact constraints. Similarly to the edge grasping scene, increasing the number of contacts also increases the gain of our method for constraint solving performance, from 62% to 76%. Moreover, in the more complex scene, constraint solving was a clear bottleneck with point contact constraints, taking 85% of the simulation step. With our aggregate constraints, it took only 58%. In this scene, our method provides a dramatic improvement of the frame rate, increasing from 3.81 fps to 10.68 fps.

For simulations with a low number of phalanges in contact, and hence less contact points, such as the pen spinning scene, our method still improves greatly constraint solving times, which are almost halved in this case. However, the impact is then smaller on the overall performance, as the simulation cost is shared with other components, such as FEM simulation.

The dumbbell lifting scenario is a more challenging scenario for our method, as it contains 30 deformable phalanges during two-handed interaction. This scenario also leads to many collisions between several phalanges and the bar, pushing the number of contacts to an average of 86 during lifting. Constraint solving thus becomes a critical part in the simulation step, taking 65% of the time with point contact constraints. Even on such a

difficult case for our method, it almost halves constraint solving time, and in turn reduces the overall computation time by 27%, for the same visual result.

5.3.3 Discussion

Our aggregate contact constraint method effectively reduces the number of constraints induced by contact with deformable phalanges during grasping and dexterous manipulation of virtual objects. This leads to a reduction of constraint solving times as well as overall simulation times on scenarios with moderate to high complexity. Simulations with our method are otherwise visually very similar to simulations obtained with point contact constraints, with pressure distribution and torsional friction being accurately represented. The method is especially efficient when the number of contacts per phalanx is high, and its cost grows with the number of phalanges in contact, but even in the latter case it still leads to a noticeable improvement in computation times.

Our method allows to attain the interactive frame rates suitable for real-time interaction with fully deformable fingers using a data glove (over 60 fps for some scenarios). The aggregate constraints provide a global force and average direction for an entire phalanx, which could be used to display kinesthetic feedback to the fingers of the user. The computation of pressure distribution, on the other hand, could be used for tactile feedback on the finger pads.

5.4 Conclusion

We presented a novel contact constraint method dedicated to the simulation of interaction with virtual objects using hand models with deformable phalanges. Our constraint-based formulation aggregates all contacts between two bodies into a single set of separation and friction constraints. We introduced a weighting method for simulating the non-uniform pressure distribution over the contact surface of finger pads. This better simulates the response to contact with varying penetrations over the contact surface, such as when grasping objects by sharp edges. We proposed the use of Coulomb-Contensou friction in our constraint formulation in order to simulate torsional friction at contact surfaces without requiring multiple friction frames.

Our approach thus uses a minimal set of constraints for accurately simulating contact and friction for contact surfaces generated by deformable finger pads, which generally generate numerous contacts during grasping. Especially in complex scenarios, our method leads to a significant improvements in terms of constraint solving times, as well as global simulation times. The performance gain provided by our method can therefore be used to simulate more complex real-time scenarios such as bimanual interaction, or to attain the high simulation rates required by haptics.

Conclusion

6

In this manuscript, we focused on the topic of two-handed haptic interaction with virtual environments. The main goal of our research was to **improve the interaction with virtual objects within physically-based VEs using two haptic devices**. This sort of interaction can be performed through virtual hand models, which ideally allow direct and natural manipulation of virtual objects. They may however be computationally expensive to simulate, which makes it more difficult to retain haptic rates. Alternatively, simple rigid proxies can be used, which are efficient but limited in terms of interaction possibilities, requiring additional techniques in order to perform specific tasks within VEs. The main issues encountered with both interaction metaphors were tackled by following two main objectives. The first objective focused on **enhancing bimanual haptic navigation and manipulation using interaction techniques**, notably with simple devices and virtual proxies. The second objective consists in **improving the computational efficiency of more complex models, such as virtual hands**, in order to attain haptic rates in complex bimanual scenes.

We first proposed novel interaction techniques for **improving bimanual exploration in VEs and haptic manipulation with single-point haptic devices**, as well as both of these tasks performed simultaneously (Chapter 3). The haptic navigation technique, called *double bubble*, extends the workspaces of grounded devices by using a combination of position and rate control, leading to smoother and more intuitive navigation compared to techniques that use either of those exclusively. As the virtual proxies for both hands can move infinitely in all directions, an adaptation of the viewport is proposed to keep track of both regardless of the distance between them. Optionally, as users may get confused when both proxies cross each other, a separation plane can be used as well to keep both hands on their respective sides of the field of view. This plane can also be used to handle rotations of the viewport, or translational DOF can be substituted for rotational DOF for the same effect. Since differences in control modes can be introduced with the *double bubble* which can be an issue when grasping objects, a *joint control* is proposed to ensure common velocities during grasping. The manipulation technqiue is the *magnetic pinch*, which uses springs or constraints to pull both hands and an object together to stabilize the grasping of that object. The technique is triggered when a grasping situation is detecting, using the relative position of the proxies and the forces applied on the object. A user experiment showed that the combination of these techniques improved the completion times and grasping stability for pick and place tasks, and were overall more appreciated compared to state-of-the-art methods.

We then presented a method for **enhancing interaction using point and rigid proxies as well as rigid hand models, by rendering finger pad-like contact surfaces** without using computationally expensive soft body simulation methods (Chapter 4). This *god-finger* method uses a heuristic that scans the local geometry of the touched object to provide additional contact points spanning the generated contact surface. It keeps the generated surfaces plausible by taking into account the force applied on the object, and by

stopping the propagation of the surface in positions that would lead to unnatural shapes. The method is adapted for 3DOF haptic interfaces, in which case the generated surfaces are circular, as well as for 6DOF devices, that produce elliptic surfaces that are closer to finger pad contacts. By default it handles interaction using point proxies, but it can also be extended to work with most kinds of convex rigid proxies by changing the starting points of the scan. Similarly, the base algorithm assumes that the touched object is locally smooth, but an alternate algorithm can be used to take into account local holes in the geometry. As interaction with deformable bodies requires a finer sampling of the contact surface, an iterative scan can be used to scan the surface in a radial tree pattern, providing a more homogeneous repartition of contact points over the surface, scaling linearly with the number of scans performed. A haptic feeling of deformation of a finger under sliding contact can be added by modulating friction coefficients, while visual feedback is provided through Bézier curves. The contact surfaces added by the *god-finger* method allow to better restrain the rotations of objects under contact, facilitating grasping when a low number of proxies is involved.

Finally, we proposed a method for **improving the computation of contact between deformable hand models and virtual objects**, which generates a large amount of constraints which prove challenging to solve (Chapter 5). Our novel contact constraint formulation aggregates the multiple contacts within each phalanx into a single set of constraints per phalanx, which include both separation and friction. We preserve the pressure distribution over the contact surface by weighting each individual constraint within the aggregate formulation. The Coulomb-Contensou friction law is used in order to preserve torsional friction on each contact without requiring multiple sets of tangential friction constraints. As a result, each phalanx generates a maximum of 4 constraints, independently of the resolution of the collision mesh, which greatly reduces constraint solving times in complex scenarios. This performance gain can prove desirable for real time interaction using haptic gloves, which require high simulation rates. Even when the number of phalanges is increased, our method still performs remarkably, making it of interest for bimanual interaction with deformable hands as well.

Future work

The work presented in this manuscript left some open questions, which would be the focus of short-term future work. We discuss some research axes worth studying for each of the proposed contributions.

Interaction techniques for bimanual haptic navigation and manipulation

- **Immersion.** In its current form, the *double bubble* method allows to move both hands infinitely inside the VE, causing them to be at variable distances from each other and from the point of view of the user. While practical, this interaction metaphor may not feel immersive, as it would be expected that the hands would stay within arms' reach at all times. The method could thus be adapted to allow only motions of the workspaces within that range, but this raises some additional questions, notably how to handle the motions of the users themselves inside the VE. A user evaluation could also be conducted to compare the efficiency and sense of immersion provided by both approaches.
- **Grasping facilitation.** Currently, while the grasping detection method is potentially able to detect the grasping of multiple objects at once, the *magnetic pinch*

is only able to handle one grasped object at a time. It could thus be potentially extended to handle these cases, by adding constraints or springs between the several objects involved. Another issue with the *magnetic pinch* is the choice of stiffness parameters, which can potentially either introduce stickiness with light virtual objects, or make the method ineffective with heavier objects. Those parameters could thus be modulated by the weight of the picked objects, while making sure that the hands do not get stuck on heavier objects.

- **Evaluation.** The presented evaluation focused mostly on the advantages of the proposed techniques compared to state-of-the-art navigation techniques and virtual manipulation without any manipulation technique. Further evaluations could be performed to compare the different variants of the techniques, whether being on the method for handling rotations of the viewport, or the method for linking hands and object with the magnetic pinch. The methods were evaluated on a simple pick-and-place task, but they could also be tested on other interaction scenarios such as peg-in-a-hole tasks to better assess the efficiency of the manipulation techniques. Similarly, the navigation methods could be better evaluated on more complex VEs, for instance with obstacles to avoid during navigation and grasping.

Simulation of finger pad contact surfaces from rigid contacts

- **Contact surface computation.** While the current algorithm can be adapted to handle multiple initial contact points, it works best when those points are fairly close to each other. If those points are further away, then the propagation of the contact surface from these points would lead to surfaces that have more polygonal shapes instead of ellipses. A new propagation algorithm could thus be developed to generate the contact surface from arbitrary initial contact positions. Optionally, the method could be adapted to generate contact surfaces with arbitrary shapes as well, for instance to simulate more complex shapes like the palm. Furthermore, the rough surface compensation only takes into account local holes in the geometry so far, as such holes are not expected to hinder the propagation of the contact surface of an actual finger pad. Local bumps, on the other hand, could potentially be an obstacle to that propagation due to their height, which the current algorithm does not handle properly thus far, and could be improved in future work.

- **Visual feedback.** Bézier curves provide a clear indication of the contact surface generated by the *god-finger* method, but it is also a visually unrealistic representation. The visual feedback could be improved by choosing more natural metaphors, like the shadow of the emulated finger pad, or a fingerprint. This may however prove challenging, as the algorithm only checks partial information about the geometry of the touched object. Fetching the missing information in order to, for instance, apply a fingerprint texture on the touched object, could lead to a noticeable impact on performance. Moreover, the method used to simulate the deformation of the simulated finger only provides haptic feedback to the user while the visual feedback remains the same, which could lessen the impact of the method due to this discrepancy. This issue could possibly be solved with rigid proxies by applying some basic visual deformation of the proxy as well, to emphasize the difference between the finger deformation state and sticking friction state.

- **Evaluation.** So far the method has been mostly compared to rigid proxies in terms of interaction capabilities and performance, and a comparison to soft body simula-

tion methods would also be needed. However, in terms of performance, comparing both methods would require finding a relation between parameters of the *god-finger* method and resolution of FEM models, which is not trivial.

Efficient contact constraint formulations for deformable hand grasping

- **Hand model.** The proposed hand model was only made of the fingers and was missing the palm, which could be added as future work. The palm, however, is a more complex structure that cannot simply be mapped to a single rigid body, unlike the phalanges, due to its more complex underlying skeletal structure. Also, linear FEM is used to simulate phalanges, while finger deformation is in fact non-linear, becoming very stiff as the contact forces increase. The phalanges are also completely separate from each other, not providing a good visual deformation of the joints between them. Additional methods could thus be used to simulate finger deformation more realistically.

- **Performance.** Our aggregate contact constraint method is especially efficient on complex scenarios where the contact solving step is the most computationally expensive part of the simulation step. Its impact is however more limited on simpler scenes where other factors such as FEM simulation become more expensive in comparison. We could thus expect better global performance and an even higher impact of our approach by using parallel computation for those parts of the simulation.

- **Evaluation.** The method was mostly evaluated on interaction scenarios involving objects that are locally smooth or having only one sharp edge. While our approach is robust in these cases, it could potentially encounter some issues when sliding along irregular surfaces made of many ridges. As contacts are added and removed from the aggregate constraint, the centroid and average normal would move in a non-smooth way. The impact of this could thus be evaluated. The method could also be more directly compared to other methods meant to reduce constraint solving times, such as contact culling and contact clustering.

Long-term perspectives

More integration of psychophysical data into bimanual computer haptics

When it comes to bimanual haptics with VEs, very little emphasis has been put so far on taking into account specificities of the human bimanual haptic system in the design of hardware, software and interactive techniques. There is however considerable potential to be exploited in this area, not only to improve two-handed interaction with VEs, but also to benefit from the proprioceptive cues of the bimanual haptic system for unimanual tasks. Overall, the field as a whole will benefit from a clearer understanding of these perceptual and motor aspects. For instance, while unimanual grasping has been extensively studied, bimanual grasping lacks clear classifications of the different types of grasping, which could help for the design of grasping detection methods.

Concerning hardware, the influence of having different devices in each hand could be better studied. Since the NDH is known to act at a coarser scale than the DH, and to also have higher proprioceptive sensitivity, the design of bimanual hardware could be altered

to reflect these specificities. Similarly, since the NDH is often used to hold objects that the DH manipulates, it could be hypothesized that devices for the NDH would benefit from a better ability to render continuous forces. On the software side, these differences between both hands could be reflected on the virtual proxies used, for instance by assigning different stiffnesses to both virtual couplings.

It has also been shown that studies on bimanual motor control can be used to design more efficient two-handed interaction techniques, leading to easier and more natural manipulation of virtual objects [Cutler et al., 1997]. The frame of reference principle was especially shown to improve the design of bimanual interaction techniques, even for seemingly unimanual tasks [Ullrich et al., 2011a]. This principle could be further used to detect the intent of the user to interact bimanually with an object, if the NDH is already in contact with it and the DH is at proximity. This could be used for instance for haptic guidance of the user, even before the DH actually touches the object. Techniques that prioritize the NDH for navigation tasks or coarse motions of objects, and further methods that allow the DH to use the frame of reference created by the NDH, could also be investigated.

Interaction techniques adapted for different bimanual tasks

Ideally, bimanual haptic interaction with VEs would be performed through haptic devices providing tactile and kinesthetic feedback to the fingers and palm. Tools and objects would be picked and manipulated as in real life, a single interaction metaphor allowing the full range of tasks expected with two hands. This kind of interaction has already been studied for rigid hands interacting with simple rigid objects, with convincing results at haptic rates [Ott et al., 2007].

However, this kind of generic approach may take some time to be popularized, for both hardware and software reasons. On the hardware side, devices allowing whole-hand bimanual haptic interaction are both expensive and cumbersome, and they suffer from collision issues between both interfaces due to the robot arms used for kinesthetic feedback on the palm. Additionally, these interfaces limit navigation due to their grounded nature. On the software side, the performance and stability of physical simulations suffer when more accurate hand models or more complex objects are involved. Interaction techniques will thus remain required in order to alleviate some of the issues met with bimanual whole-hand haptic interaction.

Navigation techniques are the most obviously needed due to the fact that kinesthetic interfaces tend to be grounded. Approaches like the *bubble* technique [Dominjon et al., 2005] may become a standard for immersive navigation in large VEs while grasping small objects, due to their simplicity of implementation and use. The question of how to adapt such methods to the handling of heavy or bigger objects that do not fit within arms' reach remains mostly open, though our joint control approach starts offering a solution to this issue. Likewise, workspace extension techniques may be required to handle interface collision issues between larger haptic devices.

As for manipulation, techniques that approximate the physical interaction between hand and objects may also be used to solve the issues with the computation cost of hand models. For instance, when considering firm, stable grasps of rigid objects, it may not be required to simulate accurately the contacts on the fingers and the palm. Instead, the control scheme could switch to a direct control of the motions of the object itself, with contact forces on the object mapped back to the hands. Grasping detection methods would ensure the transitions between the control schemes. This kind of approach has been already validated for non-haptic unimanual and bimanual interaction with virtual cars [Moehring and Froehlich, 2010], and could be generalized to all sorts of tasks.

Interfaces with single effectors may also remain relevant on the long term as alternatives to whole-hand devices, due to their affordability and smaller sizes. These devices suffer from the opposite problem however, as they allow very simple tasks by default which need to be enhanced using interaction techniques. Generic techniques like the *magnetic pinch* will allow the grasping of virtual objects and overall simple tasks that just require moving objects around. They may however fall short when it comes to tasks that require a fine manipulation of objects, which will probably require more specialized interaction metaphors. This, in turn, raises the question of how to handle transitions between these different metaphors in a way that feels natural to the user.

Adaptive physical models for more complex virtual environments

As more computational power will become accessible over time, we will be inclined to build more complex VEs so as to interact haptically with them using both hands. Complexity can be increased by building larger physically-based VEs with more objects interacting with each other and with which a user can interact. It can also be increased by refining the resolution of current scenes and hand models, or using more physically-accurate models, for more realistic visual, kinesthetic and tactile rendering. Soft bodies may also gain more interest, notably in the medical field for simulating living tissue and interactions between different organs, in virtual assembly for simulating deformable and quasi-rigid tools, and more generally for simulating the compliance of human hands.

However, in a lot of cases, this increase in complexity is reflected on the number of contacts involved in the physical simulation. Especially with two-handed interaction using virtual hands, systems involving many bodies interacting with each other through an even higher amount of contacts are very challenging to solve, even with increased computational power. As a result, methods that allow to enhance the tradeoffs between performance and physical accuracy may gain more interest as well.

This is an issue that the graphics community has had to deal with as well for real-time applications. With the increase in graphical power, bigger virtual environments are displayed to the user, while higher fidelity is also desired, the both of them antagonizing each other. However, the simple observation that elements that are further away from the viewpoint do not need such a high fidelity allowed the development of level-of-detail (LOD) algorithms. A parallel could be made with physically-based simulations: objects that are not visible or in contact with a haptic proxy may not require the same level of physical accuracy. For instance, it is possibly not required to ensure perfect non-penetration between colliding objects, or use the most physically-accurate deformation models for objects that are not visible.

Some multi-rate simulation approaches were already proposed to increase the simulation rates in subspaces around haptic interaction points, in order to achieve haptic rates at a more limited cost [Jacobs and Cavusoglu, 2007, Glondu et al., 2010]. These approaches could be extended to dynamically choose the simulation methods and adapt the resolution of the models depending on the accuracy required for the user to have a believable and immersive experience. Constraint-based contact resolution could be used where good visual or haptic fidelity is required, while penalty-based or impulse-based methods could otherwise be used if they do not hinder the stability of the global simulation. For virtual hands, deformable models with aggregate contact constraints could be used where good visual deformation is required, while rigid models with heuristic-based finger pad contacts could be used on parts that just require haptic feedback. If only kinesthetic feedback is displayed, rigid models without heuristics could also be sufficient when interacting with soft or quasi-rigid bodies. Overall, a dynamic choice of simulation methods could improve

the performance and stability of physically-based VEs, providing the user with enriched VEs and smoother feedback while keeping a comparable believability of the experience.

Appendix: Résumé Long en Français A

Contents

Dans ce manuscrit de thèse, nous présentons des travaux réalisés dans le contexte de la Réalité Virtuelle (RV). Les applications de RV visent à **permettre à un utilisateur d'interagir en temps réel avec un Environnement Virtuel** (EV), ainsi que de **percevoir cet EV de la manière la plus immersive possible**. Ce degré d'immersion est atteint par une substitution des stimuli de l'environnement réel autour de l'utlisateur par des stimuli générés par ordinateur, reflétant la présente de l'utilisateur dans l'EV. Jusqu'ici, les applications de RV ont le plus souvent utilisé les modalités visuelle et auditive pour fournir les retours de l'EV vers l'utilisateur. Cependant, ces modalités seules peuvent se révéler insuffisantes quand il y a une interaction physique entre l'utilisateur et des objets virtuels. Ce type d'interaction met à contribution le **sens haptique**, qui inclut la perception tactile des surfaces et textures, et la proprioception, perception de l'équilibre et de la posture.

De par son importance dans les applications interactives, la modalité haptique a reçu une attention croissante en RV dans les dernières décennies. L'inclusion de l'haptique en RV a été rendue possible par les interfaces haptiques, des interfaces homme-machine permettant de détecter les mouvements d'un utilisateur et de stimuler le sens haptique. Ces interfaces sont couplées à des représentations virtuelles de l'utilisateur dans l'EV, appelées proxys virtuels, qui reproduisent les mouvements de l'utilisateur et renvoient les forces d'interaction à l'interface haptique. Dans un cas idéal, un utilisateur serait capable d'interagir avec des objets virtuels directement avec leur main, en les saisissant et déplaçant, ou bien en effectuant des tâches de manipulation précises. Encore plus idéalement, les deux mains seraient impliquées dans ces interactions.

Dans la vie quotidienne, **nous utilisons couramment nos deux mains pour réaliser une grande variété de tâches**. Un certain nombre de ces tâches sont effectuées de manière bimanuelle si naturellement que l'on peut ne pas se rendre compte que les deux mains sont mises à contribution. L'interaction à deux mains est quelque chose de courant, en partie grâce à des spécificités du système bimanuel haptique, telles que des différences de sensibilité entre les mains. Cependant, dans le contexte de l'interaction haptique avec des EV, les interactions impliquaient majoritairement une seule main jusqu'à récemment. Sachant l'importance de l'utilisation de deux mains dans la vie courante, l'interaction haptique unimanuelle peut se révéler moins efficace et immersive pour un certain nombre de tâches interactives en RV. Cela soulève l'intérêt d'une meilleure intégration des deux mains en haptique. L'***haptique bimanuelle***, ou *haptique à deux mains*, désigne **l'interaction haptique à travers les deux mains d'une même personne**, et est le sujet principal de cette thèse.

Défis de l'Interaction Haptique Bimanuelle avec des Environnements Virtuels

Dans le cadre d'une interaction en temps réel avec un EV, une question centrale est le choix de la méthode de représentation de l'utilisateur dans l'EV, qui lui permettra de réaliser un certain nombre de tâches. Cette question est particulièrement pertinente en haptique bimanuelle, car l'utilisation de deux mains permet de réaliser un champ de tâches assez large. Ces méthodes, qui combinent des éléments matériels et logiciels pour réaliser des tâches spécifiques, sont connues sous le nom de techniques d'interaction (Figure A.1).

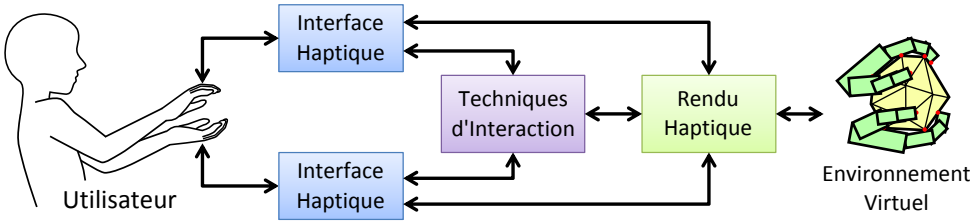


Figure A.1 – Eléments impliqués dans l'interaction haptique bimanuelle entre un utilisateur et un environnement virtuel. L'utilisateur (couche humaine) interagit avec les interfaces haptiques (couche matérielle), qui sont couplées à l'environnement virtuel par du rendu haptique (couche logicuelle). Les techniques d'interaction sont liées à tous ces éléments et définissent comment l'utilisateur pourra réaliser un certain nombre de tâches.

Des techniques d'interaction sont tout d'abord requises pour les tâches d'exploration haptique. En effet, les interfaces haptiques ont une **amplitude de mouvement limitée**, généralement insuffisante pour interagir avec des objets au-delà de l'amplitude des bras. Aussi, les interfaces à retour de force sont généralement des bras robotiques, qui peuvent rencontrer des problèmes de collision sous certaines circonstances, réduisant d'autant plus leur amplitude. Des techniques d'interactions sont donc nécessaires pour **étendre les espaces de travail** de ces appareils.

Deux métaphores d'interaction principales permettent la manipulation haptique d'objets virtuels, chacune avec ses propres défis. Les **mains virtuelles** permettent une interaction directe et naturelle, mais sont dépendantes de gants haptiques qui sont potentiellement coûteux et limités en degrés de liberté (DDL). De plus, les mains virtuelles peuvent se révéler coûteuses pour de l'interaction avec des EV basés physique, à cause du haut coût de simulation de modèles de main réalistes, ainsi que pour la résolution de scénarios de contact complexes. A l'opposé, les proxys virtuels rigides représentent un outil particulier qui servira à interagir avec l'EV. Ces modèles sont simples et rapides, mais sont sou-

vent **limités à une tâche unique**, et peuvent se révéler moins immersifs que les mains virtuelles.

Des défis supplémentaires sont liés au sens haptique humain, qui a une haute sensibilité temporelle et spatiale, ce qui nécessite de **hautes fréquences de rendu haptique**. Pour les EV basés physique, de **hautes fréquences de simulation physique** devraient aussi être visées pour répondre à cette haute sensibilité temporelle, notamment pour simuler des contacts ponctuels. Des modèles de contact réalistes devraient aussi être choisis pour respecter la sensibilité spatiale du sens haptique. Enfin, des modèles réalistes du comportement physique des proxys virtuels devraient être utilisés, notamment en prenant en compte la déformation des doigts avec des mains virtuelles. Cependant, des modèles physiques réalistes sont généralement incompatibles avec la nécessité de taux de simulation élevés. L'inclusion de deux mains dans la simulation augmente d'autant plus le temps de calcul, en doublant le nombre de proxys impliqués dans l'interaction et le nombre de contacts générés.

Objectifs de la Thèse et Contributions

Les travaux réalisés dans cette thèse cherchent à améliorer plusieurs aspects de l'interaction haptique bimanuelle, qui peuvent être rassemblés en deux catégories principales. Tout d'abord, l'interaction 3D dans des EV avec deux interfaces haptiques est considérée. Ensuite, nous considérons les compromis entre temps de calcul des différentes méthodes bimanuelles haptiques et le réalisme de ces méthodes.

Les interfaces à retour de force ont généralement des espaces de travail limités, allant de l'amplitude des mouvements du poignet à celle des bras. Cela se révèle insuffisant dans la plupart des cas, comme les EV ont tendance à être plus larges que ces limites. Des techniques d'interaction sont donc nécessaires pour permettre l'exploration d'EV et la manipulation d'objets virtuels larges, tout en ne gênant pas cette manipulation. Ceci est particulièrement le cas pour la manipulation bimanuelle avec des proxys rigides, qui peut être difficile du fait que seulement deux points de contact servent à la saisie d'objets. Cela mène à des saisies moins stables, qui nécessitent des techniques d'interaction pour assister l'utilisateur dans la manipulation des objets virtuels.

Lors de la saisie d'objets à deux mains, que ce soit avec des mains virtuelles ou juste deux phalanges, un aspect important est le rendu des surfaces de contact entre les doigts et les objets manipulés. Ces surfaces permettent de stabiliser la prise des objets en contraignant la rotation relative des objets grâce au frottement en torsion. D'un côté, les modèles rigides ne peuvent pas générer ces surfaces comme ils ne peuvent pas se déformer pour épouser la forme de l'objet en contact. D'un autre côté, les modèles souples simulent cette déformation, mais sont généralement coûteux en temps de calcul. Des méthodes sont donc nécessaires pour rendre ces surfaces de contact plus efficacement. D'autre part, ces surfaces sont généralement simulées par des ensembles de points de contact, dont la multitude peut se révéler complexe à résoudre. Il est donc intéressant de réduire le nombre de contraintes de contact pour réduire le temps de calcul pris par leur résolution. Cependant, il est aussi nécessaire de conserver le frottement en torsion qui est habituellement amené par l'addition des frottements tangentiels sur chaque point de contact.

Nous détaillons nos contributions par la suite, en partant des techniques qui améliorent l'interaction avec des proxys virtuels simples jusqu'aux méthodes qui améliorent l'efficacité de modèles plus complexes.

Nous abordons d'abord des problèmes avec l'interaction bimanuelle dans des EV avec des interfaces à effecteur unique. Nous proposons une technique d'interaction nommée la

double bulle pour l'exploration d'EV avec une combinaison de contrôle en position et en vitesse. Nous présentons aussi une technique de manipulation nommée *prise magnétique* qui facilite la saisie d'objets virtuels avec des proxys rigides simples. Des modes de contrôle communs sont utilisés pour améliorer la saisie et l'exploration simultanées. Une évaluation utilisateur a été réalisée pour mesurer l'efficacité de ces techniques.

Nous nous intéressons ensuite au calcul de surfaces de contact. Nous proposons une technique nommée *god-finger* pour rendre des surfaces similaires à celles générées par des doigts à partir d'un unique point de contact. Elle est basée sur un simple parcours de la géométrie locale de l'objet en contact, et est donc moins coûteuse que des méthodes de simulation de corps souples. La méthode est adaptée pour l'interaction avec des proxys rigides simples ou plus complexes, ainsi qu'avec des objets rigides ou déformables, y compris avec des surfaces rugueuses. Une méthode de rendu visuel donne un retour à l'utilisateur sur la forme de la surface de contact.

Enfin, nous abordons la résolution de contacts durant la manipulation dextre d'objets virtuels avec des doigts souples. Le calcul des mécaniques de contact est amélioré en agrégeant les multiples contraintes de contact concernées. Une méthode de distribution de pression non uniforme sur la surface de contact adapte la réponse lors d'un contact contre des arêtes aigües. Nous utilisons le modèle de frottement de Coulomb-Contensou pour simuler efficacement le frottement en torsion. L'approche est évaluée avec un modèle de main déformable pour de l'interaction en temps réel.

A.1 Techniques d'Interaction pour les Interfaces Haptiques Bimanuelles

Les interfaces haptiques à effecteur unique sont communes, et un moyen efficace d'interagir avec des EV à deux mains avec des proxys rigides. Cependant, certains problèmes apparaissent en interagissant avec ces interfaces, qui sont de deux ordres. D'abord, ces interfaces ont des espaces de travail limités, qui limitent l'exploration dans des EV modérément larges. Ensuite, le nombre limité de points d'interaction rend la manipulation d'objets virtuels plus difficile avec des proxys simples. Dans cette section, nous présentons des techniques d'interaction haptique bimanuelle qui résolvent ces deux problèmes, tout en prenant en compte l'asymétrie entre les mains.

A.1.1 Navigation dans des Environnements Virtuels avec Deux Interfaces Haptiques

Nous proposons une technique d'exploration d'EV appelée *double bulle*, inspirée par la technique de la *bulle* tirée de l'haptique unimanuelle [Dominjon et al., 2005]. Chaque interface possède une certaine zone (ou *bulle*) à l'intérieur de laquelle le proxy correspondant est contrôlé en position, permettant une manipulation précise d'objets virtuels. En-dehors de cette zone, les *bulles* se déplacent dans l'EV à une vitesse constante, fonction de la distance de l'interface aux limites de la *bulle* (Figure A.2). La méthode se voulant générique, les bulles peuvent prendre n'importe quelle forme convexe. Un retour haptique attire les interfaces vers les surfaces de leurs *bulles*, et un retour visuel dénote du passage en contrôle en vitesse sous la forme d'une traînée de mouvement.

La *double bulle* permet de bouger les deux interfaces indépendamment l'une de l'autre dans un espace potentiellement infini. Cela signifie que leurs proxys peuvent se retrouver très éloignés l'un de l'autre, notamment en dehors du champ de vision d'origine. Les deux

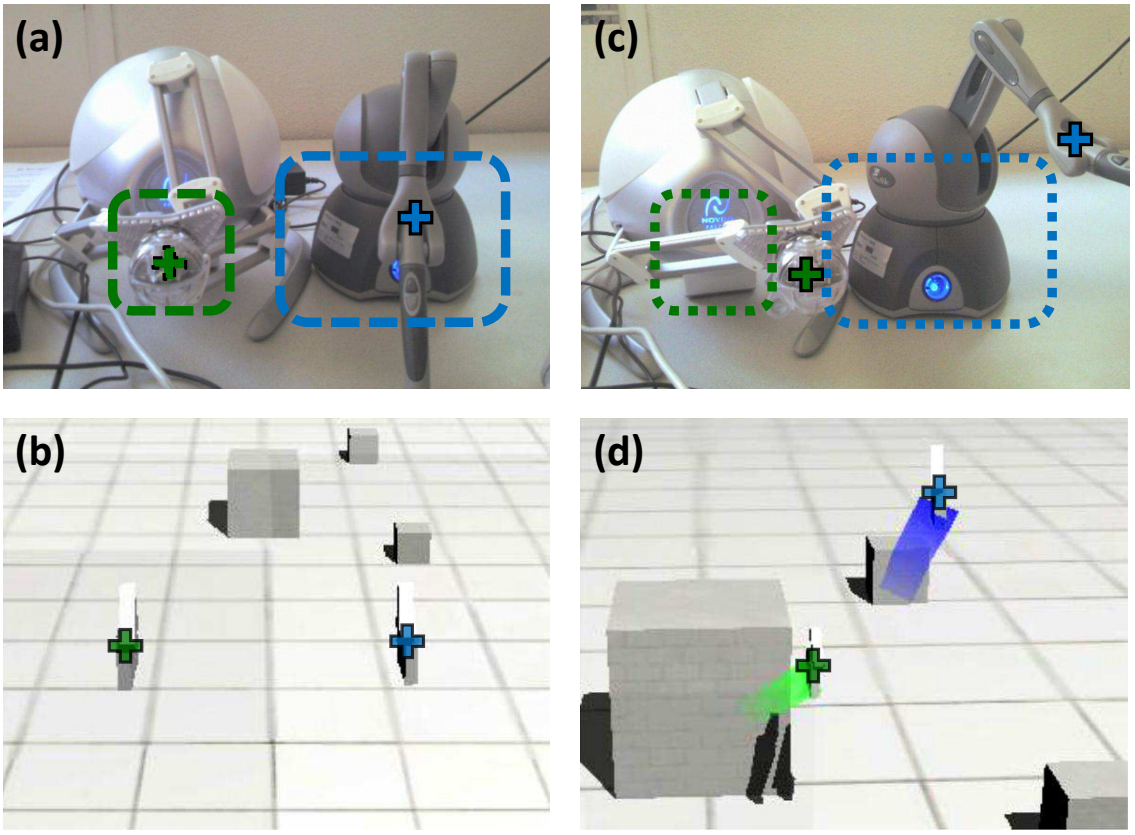


Figure A.2 – Modes de contrôle de la *double bulle*. La ligne supérieure représente les interfaces haptiques avec les espaces de travail en lignes pointillées. La ligne du dessous représente l'environnement virtuel correspondant. (a-b) Interfaces à l'intérieur des *bulles* : contrôle en position. (c-d) Interfaces en dehors des *bulles* : contrôle en vitesse, indiqué par des traînées colorées.

proxys peuvent être maintenus dans le champ de vision en adaptant le point de vue par rapport à la position des *bulles*.

Il est aussi possible que la *double bulle* introduise des différences de mode de contrôle et/ou de vitesse de déplacement entre les deux *bulles*, ce qui peut se révéler problématique lors de la saisie d'un objet à deux mains. Nous proposons donc un *contrôle commun*, qui force une même taille de bulle, un même mode de contrôle, et une même vitesse de déplacement de bulles lorsqu'un objet est saisi à deux mains.

A.1.2 Manipulation Bimanuelle d'Objets Virtuels avec des Interfaces à Effecteur Unique

Avant d'appliquer des méthodes pour améliorer la manipulation d'objets virtuels, il est nécessaire de détecter l'intention de l'utilisateur de saisir un objet. Nous proposons donc une méthode de détection de saisie d'objet à deux mains, basée sur la position relative entre les deux mains, qui doivent être face à face, et un seuil de forces de contact.

Une fois la saisie détectée, la *prise magnétique* assiste l'utilisateur dans cette saisie en attirant les mains de l'utilisateur et l'objet ensemble. Cette aide peut être réalisée par des ressorts entre les deux mains ainsi qu'entre l'objet et la position centrale entre les deux mains, donnant une sensation d'aimantation. Une autre possibilité est l'utilisation de contraintes entre les mains et l'objet, permettant de mieux manipuler l'objet en rotation grâce à la position relative entre les deux proxys, et possiblement l'orientation des interfaces si disponible (Figure A.3).

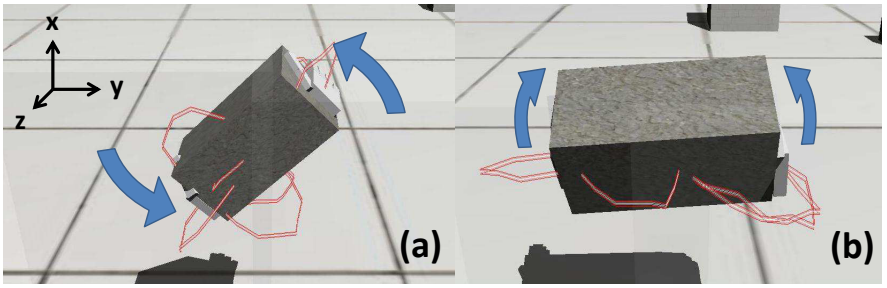


Figure A.3 – Rotation d'un objet virtuel avec l'approche basée contrainte de la *prise magnétique*. (a) Rotation autour de l'axe de profondeur en utilisant la position relative entre les deux mains. (b) Rotation autour de l'axe de saisie en utilisant le moment de l'interface de droite.

A.1.3 Evaluation

Afin de mesurer l'efficacité de ces techniques, une évaluation utilisateur a été menée avec 13 participants sur une tâche de prise d'objet à deux mains et déplacement jusqu'à une cible. Les conditions testées étaient : (Ctrl) navigation avec techniques de l'état de l'art, (DB) navigation avec *double bulle*, (MP) manipulation avec *prise magnétique*, (DB+MP) *double bulle* avec *prise magnétique*. Les résultats montrent que les temps de complétion et le nombre de pertes de l'objet sont améliorés avec les techniques de manipulation, et que l'ajout des techniques de navigation réduit d'autant plus les temps de complétion (Figure A.4). Un questionnaire subjectif montre aussi que les conditions avec techniques de manipulation sont préférées à celles qui ne les utilisent pas.

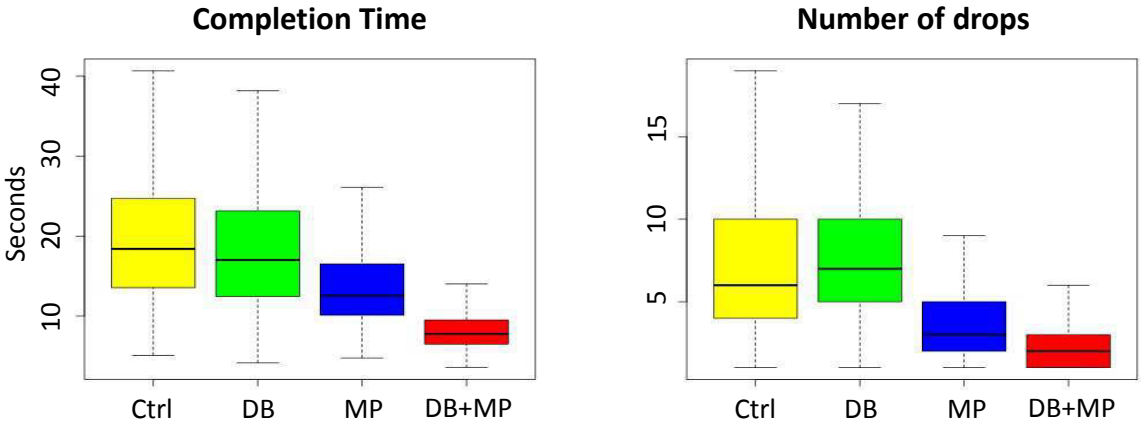


Figure A.4 – Box plots des temps de complétion et nombre de pertes de l'objet pour toutes les conditions.

A.2 God-finger: Rendu de Surfaces de Contact

Lorsque nous interagissons avec nos mains, nos doigts se déforment pour créer des surfaces de contact avec les objets. En interaction haptique, les proxys rigides ne possèdent pas cette capacité à se déformer, et simulent mal le frottement entre doigts et objets virtuels. A l'inverse, les modèles basés corps souples simulent des surfaces de contact réalistes, mais sont lourdes en temps de calcul. Nous proposons donc une méthode appelée *god-finger* pour améliorer le compromis entre frottement réaliste et temps de calcul, en **générant des surfaces de contact similaires à celles produites par des doigts humains à partir de proxys rigides**, à partir d'une simple heuristique (Figure A.5).

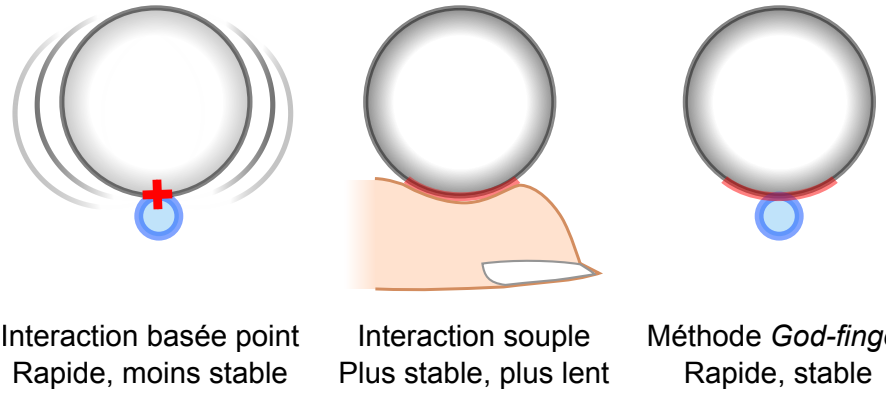


Figure A.5 – Concept de la méthode du *god-finger*. **(gauche)** Il est difficile de soulever un objet virtuel avec un point unique, ce qui cause la chute de l'objet. **(milieu)** Des méthodes basées corps souples simulent la déformation des doigts humains en contact, stabilisant ces contacts, mais étant plus coûteuses. **(droite)** La méthode du *god-finger* rend une surface de contact à partir d'un seul point de contact à partir d'une simple heuristique, permettant de soulever l'objet sans le perdre.

A.2.1 Génération de Surfaces de Contact avec des Contacts Ponctuels à 3DDL

La méthode du *god-finger* rend une surface de contact similaire à celles générées par des doigts, en parcourant la géométrie locale de l'objet touché à partir d'un seul point

de contact. La génération de cette surface se fait en plusieurs étapes résumées dans la Figure A.6. Tout d'abord, une empreinte plate est générée sur le plan tangent au contact, représentant la surface de contact qui serait attendue sur une surface plane. Elle est représentée par des vecteurs émanant du point de contact et qui recouvrent la surface de cette empreinte, appelés *vecteurs radiaux*. La seconde étape consiste à projeter cette empreinte plane sur l'objet en contact afin d'obtenir la surface de contact réelle. Cette projection se réalise en parcourant la géométrie locale autour du point de contact, en suivant les *vecteurs radiaux*. Ce parcours peut être prématurément arrêté afin d'empêcher que la surface ne possède de fortes irrégularités, afin qu'elle reste similaire à un contact de doigt. Enfin, les points résultant de ce parcours, nommés *sub-god-objects*, deviennent de nouveaux contacts entre les deux objets qui définissent la surface de contact.

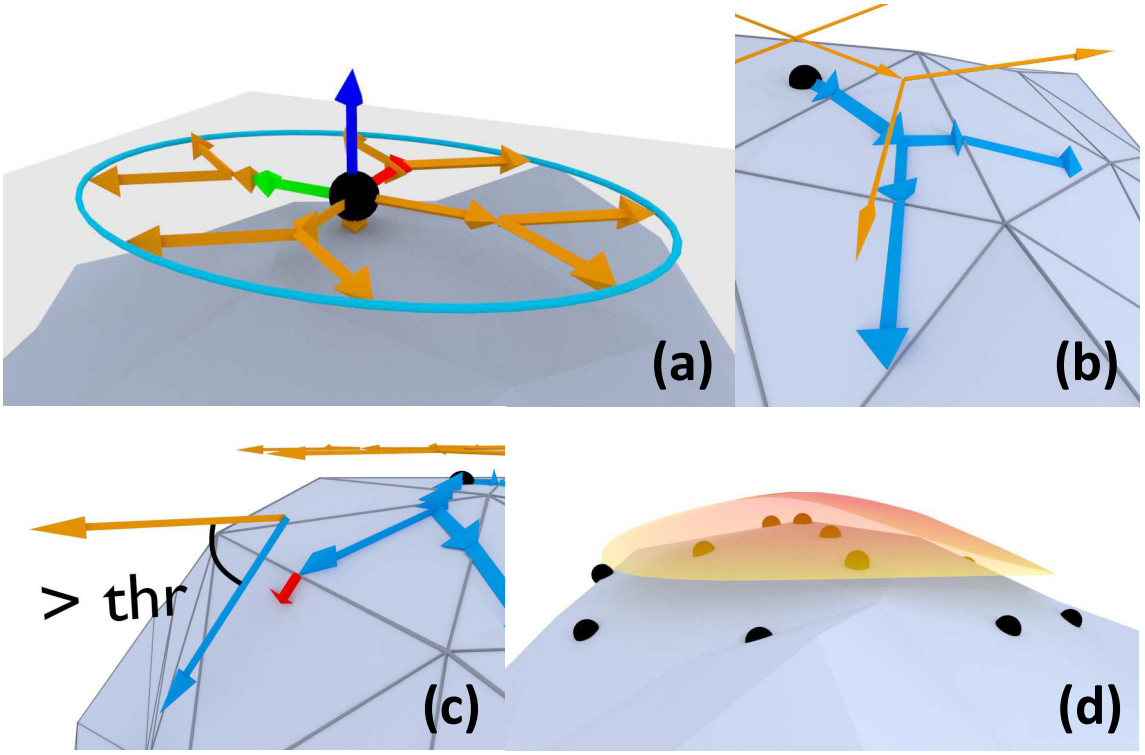


Figure A.6 – Etapes de la méthode du *god-finger*. **(a)** Une empreinte est générée comme si l'objet était plat. **(b)** La surface de contact est adaptée à la géométrie de l'objet. **(c)** En arrêtant le parcours aux arêtes aiguës, on évite des bosses irrégulières sur la surface générée. **(d)** Le résultat du parcours donne de nouveaux points de contact qui décrivent la surface de contact.

A.2.2 Améliorations de la Méthode pour des Proxys et Objets Complexes

L'algorithme de base du *god-finger* ne supporte que l'interaction avec des points uniques ou des proxys virtuels très simples, du fait qu'il ne considère qu'un seul point de contact à l'origine pour générer une surface. De plus, l'orientation des interfaces n'est pas prise en compte dans la méthode, qui génère par défaut une surface de contact circulaire, ce qui produit un rendu visuel et haptique qui n'évoque pas réellement un doigt. Il est possible d'améliorer la méthode pour interagir avec des proxys plus complexes et en prenant en compte l'orientation de l'interface. Pour les proxys complexes, la méthode peut être adaptée pour démarrer le parcours de la géométrie locale à partir de plusieurs points de contact, en faisant démarrer chaque parcours depuis le point de contact le plus proche de

chaque *vecteur radial*. Pour la prise en compte de l'orientation, le doigt peut être modelé comme une ellipsoïde, générant des surfaces de contact elliptiques plus proches des surfaces de contact de doigts.

La version basique de l'algorithme de parcours de la géométrie de l'objet se révèle efficace en contact avec des objets rigides et relativement lisses. Du fait que les points de contact générés se situent à la périphérie de la surface de contact, certains artefacts peuvent apparaître lors de contacts avec des objets déformables. Ceci peut être évité en affinant l'échantillonage de la méthode, en réalisant le parcours de manière itérative, donnant un arbre radial de points de contact qui décrit la surface de manière plus homogène. D'autre part, étant donné que le parcours s'arrête lorsqu'une arête aiguë est atteinte, la propagation de la surface de contact peut s'arrêter prématurément sur des surfaces rugueuses, qui ont localement des creux qui satisfont cette condition de terminaison. L'algorithme peut donc être modifié pour continuer temporairement la recherche une fois qu'une arête aiguë est atteinte, jusqu'à ce que soit atteint un point de l'autre côté du creux qui est considéré comme valide, ou que la distance maximale de parcours est atteinte.

A.2.3 Retour Visuel et Haptique

Les doigts humains se déforment légèrement quand une force tangentielle est appliquée sur la surface d'un objet, jusqu'à un point où le doigt ne se déforme plus et adhère à la surface, et appliquer une force supplémentaire causera un glissement du doigt sur la surface. Cette déformation du doigt n'est pas simulée, mais peut être rendue haptiquement par un modèle simple. La manière dont le centre de masse du doigt bouge dans certaines limites tandis que la surface de contact ne bouge pas peut être assimilé à du frottement, ainsi la déformation du doigt peut être rendue en modulant le coefficient de frottement du contact.

Si l'algorithme itératif de génération de surface est utilisé, l'arbre radial de points de contact peut être utilisé pour afficher les contours de la surface de contact à différents niveaux de pression, correspondant aux différentes itérations du parcours. Ces contours peuvent être affichés par des courbes de Bézier (Figure A.7).

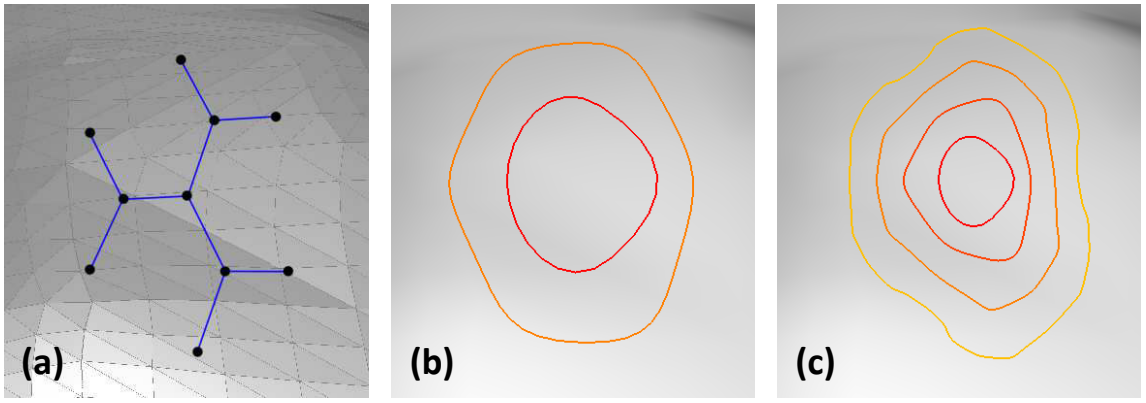


Figure A.7 – Retour visuel du *god-finger* avec des courbes de Bézier. **(a)** Points de contact générés par la méthode. **(b)** Représentation visuelle correspondante avec des courbes de Bézier. **(c)** Retour visuel avec le même contact initial et la même force appliquée, et plus d'itérations.

A.3 Manipulation Dextre avec des Doigts Déformables

L'utilisation de modèles de doigts déformables permet de mieux simuler le frottement entre une main et des objets virtuels, rendant notamment la saisie d'objets plus stable. Cependant, le grand nombre de points de contact généré rend les calculs de résolution de contact plus coûteux, ce qui est d'autant plus vrai quand chaque contact est résolu avec frottement. Les solveurs de contact avec frottement classiques pour des objets déformables ont une tendance à prendre un temps quadratique au nombre de points de contact. Allard et al. [2010] ont proposé des contraintes de contact en volume pour améliorer la performance de la résolution de contact en formulant les contraintes de contact en *volumes* de pénétration au lieu de plusieurs *distances* de pénétration. Le nombre de contraintes n'est donc plus dépendant de la résolution des maillages de collision, mais d'une grille qui détermine la manière dont les volumes de pénétration sont divisés.

La méthode souffre cependant de deux limitations importantes pour la simulation de saisie d'objets. D'une part, la pression est uniforme sur l'ensemble de la surface de contact, ce qui ne prend donc pas en compte les différences entre un contact sur une surface plane, courbée, ou pointue. D'autre part, le frottement n'agit que sur les vitesses linéaires, et ne prend pas en compte les moments sur la surface de contact sans avoir à multiplier les contraintes.

Nous proposons donc de nouvelles contraintes de contact agrégées inspirées par les travaux de Allard et al., et formulées à partir de points de contact classiques. Nous améliorons les méthodes existantes en introduisant une distribution de pression non uniforme, et en ajoutant le frottement en torsion directement à la formulation des contraintes. Ces nouvelles formulations permettent de diminuer le nombre de contraintes à un minimum de 4 par phalange, permettant d'obtenir des taux de simulation suffisamment élevée pour de la manipulation d'objets virtuels en temps réel (Figure A.8).

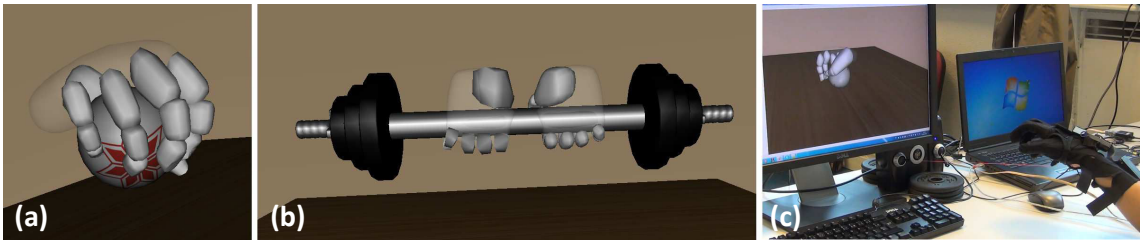


Figure A.8 – Quelques scénarios de manipulation d'objets avec notre méthode. (a) Saisie d'une balle rigide avec tous les doigts déformables. (b) Manipulation d'une balle déformable avec un gant de données. (c) Soulèvement d'haltères avec deux mains déformables.

A.3.1 Contraintes de Contact en Volume

Afin de formuler des contraintes de non-pénétration en volume à partir de contacts classiques, une aire est attribuée à chaque point de contact à partir de leur géométrie locale. Cette aire ainsi que la distance de pénétration permettent de calculer des volumes de pénétration individuels, qui sont ensuite sommés pour obtenir un unique volume de pénétration (Figure A.9). Cependant, si l'on ne fait que sommer les contraintes individuelles dans la matrice de contraintes, la même pression est appliquée sur chaque point de contact lors de la résolution des contacts. Nous proposons donc de pondérer les contributions de chaque contrainte dans cette matrice proportionnellement à la distance de pénétration de chaque

point en mouvement non contraint.

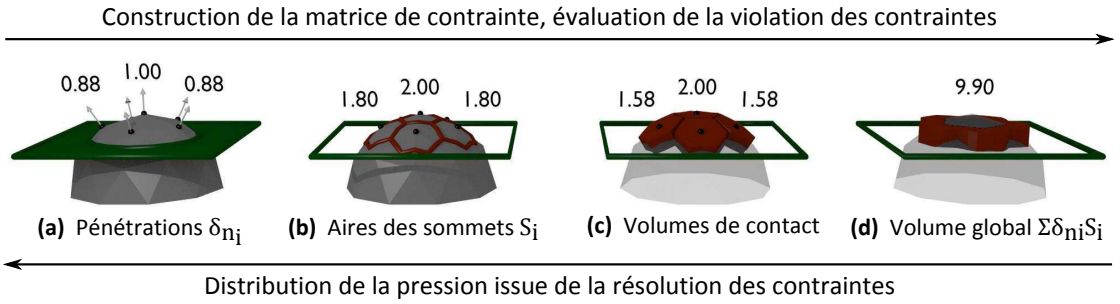


Figure A.9 – Formulation de contraintes de séparation comme contrainte en volume. **(a)** Deux corps en contact avec des pénétrations évaluées par rapport aux normales de contact. **(b)** Aires calculées pour chaque point de contact à partir de la géométrie locale. **(c)** Formulation des contacts comme volumes de pénétration. **(d)** Sommation de ces volumes pour formuler une seule contrainte.

Le frottement en torsion est ajouté à la formulation des contraintes grâce au modèle de frottement de Coulomb-Contensou [Contensou, 1963], qui calcule conjointement le frottement tangentiel et torsionnel pour prendre en compte leurs relations. La formulation des contraintes de frottement est basée sur le calcul de "potentiel de vitesse" proposé par Leine and Glocker [2003]. Pour l'application du frottement en torsion, une vitesse angulaire est évaluée comme somme pondérée des vitesses angulaires obtenues pour chaque point de contact par rapport à leur barycentre (Figure A.10).

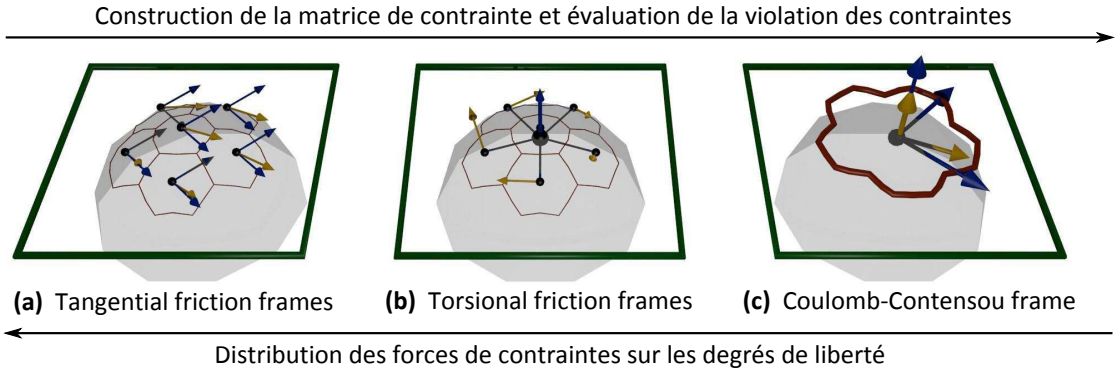


Figure A.10 – Formulation de frottement Coulomb-Contensou friction sur la surface de contact avec notre approche. **(a)** Référentiels de frottement de Coulomb individuels, avec des vitesse tangentielles à chaque point de contact. **(b)** Vitesses en torsion évaluées à partir de vecteurs orthogonaux à la normale de la surface de contact et aux bras de levier entre les points de contact et leur barycentre. **(c)** Sommation de ces référentiels pour évaluer les vitesses linéaire et angulaire pour la surface de contact.

A.3.2 Modèle de Main Déformable

Pour évaluer notre méthode de contraintes agrégées pour la manipulation d'objets virtuels, nous proposons un modèle de main déformable. Ce modèle est composé de plusieurs couches interconnectées : données de tracking, modèle en coordonnées réduites, squelette rigide, phalanges déformables, et modèle de surface pour le retour visuel et les collisions

(Figure 5.4).

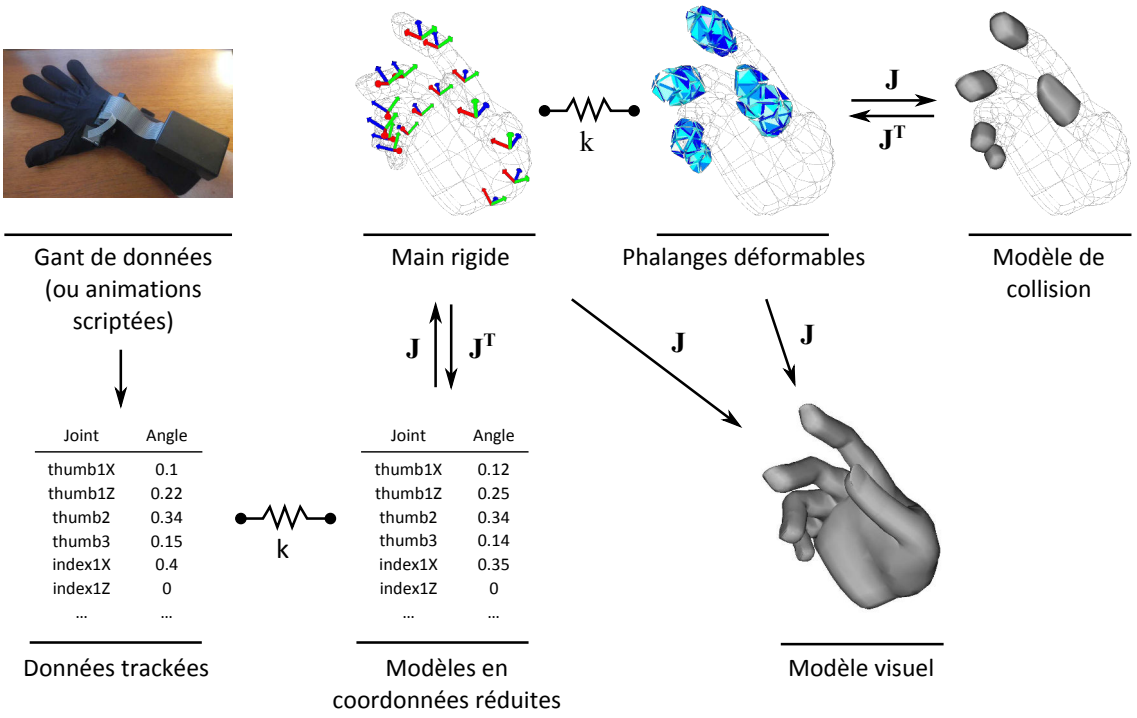


Figure A.11 – Différentes couches du modèle de main proposé. Le modèle en coordonnées réduites suit les données de tracking grâce à des ressorts, et le modèle de main rigide y est associé par un mapping. Les phalanges déformables sont liées à la main rigide par des ressorts, et les modèles de collision y sont liés par un mapping. Le modèle visuel est obtenu à partir des modèles rigides et déformable par du skinning.

A.4 Conclusion

Dans ce manuscrit, nous avons étudié l'interaction haptique à deux mains avec des environnements virtuels. L'objectif principal de cette thèse était d'**améliorer l'interaction avec des objets virtuels basés physique avec deux interfaces haptiques**. Ce type d'interaction peut être réalisé avec des modèles de main virtuels, qui permettent idéalement une manipulation d'objets virtuels directe et naturelle. Ces modèles peuvent cependant être coûteux à simuler, ce qui rend plus difficile le maintien de taux de rendu haptiques. Autrement, de simples proxys rigides peuvent être utilisés, qui sont efficaces mais limités en terms de possibilités d'interaction, nécessitant des techniques supplémentaires pour réaliser des tâches spécifiques dans des EV. Les problèmes majeurs rencontrés avec les deux métaphores d'interaction ont été abordés suivant deux objectifs. Le premier objectif a consisté à **améliorer la navigation et la manipulation bimanuelle haptique avec des techniques d'interaction**. Le second objectif a consisté à **améliorer l'efficacité en temps de calcul de modèles plus complexes, tels que les mains virtuelles**, afin d'atteindre des fréquences de rend haptiques dans des scènes bimanuelles complexes.

Nous avons d'abord proposé de nouvelles techniques d'interaction pour **améliorer l'exploration bimanuelle d'EV et la manipulation haptique avec des interfaces à effecteur unique**, ainsi que la combinaison simultanée de ces deux tâches. La *double*

bulle est une techique de navitation qui étend les espaces de travail des interfaces fixes par une combinaison de contrôle en position et en vitesse. Une adaptation du point de vue permet de maintenir les deux proxys en vue quelle que soit la distance qui les sépare. Les rotations du champ de vision peuvent être réalisées en substituant des degrés de liberté en translation, ou en utilisant un plan de séparation qui sert à empêcher les proxys de se croiser. Un *contrôle commun* permet d'éviter les problèmes de saisie d'objets liés à une différence de modes de contrôle ou de vitesse des bulles. La *prise magnétique* utilise des ressorts ou des contraintes pour assister l'utilisateur dans la prise d'objets en stabilisant la saisie. Une évaluation montre que la combinaison de ces techniques améliore les temps de complétion et la stabilité de saisie pour des tâches de déplacement d'objet, et qu'elles sont globalement plus appréciées que les méthodes de l'état de l'art.

Ensuite, nous avons présenté une méthode pour améliorer l'interaction avec des points ou des proxys rigides ainsi que des modèles de main rigides, en **rendant des surfaces de contact similaires à des contacts de doigts humains** sans utiliser des méthodes coûteuses de simulation de corps souples. La méthode du *god-finger* utilise une heuristique qui parcourt la géométrie locale de l'objet touché pour ajouter des points de contact qui décrivent la surface de contact générée. Ces surfaces demeurent plausibles en prenant en compte la force appliquée sur l'objet, et en arrêtant la propagation de la surface quand des irrégularités seraient générées. La méthode peut être adaptée pour supporter des interfaces à 6DDL, en générant des surfaces de contact elliptiques plus proches de contacts de doigts, ainsi que des proxys rigides plus complexes, en changeant les points de départ de la propagation de la surface. Le contact avec les objets déformables peut aussi être amélioré en affinant l'échantillonage de la surface de contact par un parcours itératif, ainsi qu'avec les objets rugueux en modifiant l'algorithme de parcours. Une sensation haptique d'une déformation du doigt en contact peut être ajoutée par une modulation du coefficient de frottement, et le retour visuel est fourni par des courbes de Bézier. Les surfaces de contact ajoutées par la méthode permettent de mieux contraindre la rotation des objets en contact, facilitant la saisie notamment quand un faible nombre de proxys virtuels est impliqué.

Enfin, nous avons proposé une méthode pour améliorer le calcul de contacts entre des modèles de main déformables et des objets virtuels, qui génèrent un nombre élevé de contraintes qui se révèlent complexes à résoudre. Notre formulation de contraintes agrège les contacts multiples dans chaque phalange en un unique groupe de contraintes, incluant la séparation et le frottement. Nous préservons la distribution de pression sur la surface de contact en pondérant chaque contrainte individuelle dans la formulation agrégée. Le loi de frottement de Coulomb-Contensou est utilisée pour préserver le frottement en torsion sur chaque contact sans nécessiter une division en plusieurs contraintes de frottement tangentiel. Chaque phalange génère donc un maximum de 4 contraintes, indépendamment de la résolution du maillage de collision, ce qui réduit grandement les temps de résolution de contrainte dans des scénarios complexes. Ce gain de performance est intéressant en interaction temps réel avec des gants haptiques, qui nécessitent des taux de simulation élevés.

Author's publications

Journal papers

- **A. Talvas**, M. Marchal, and A. Lécuyer. A Survey on Bimanual Haptic Interaction. *IEEE Transactions on Haptics*, 99:285–300, 2014.
- **A. Talvas**, M. Marchal, C. Duriez, and M. A. Otaduy. Aggregate Constraints for Virtual Manipulation with Soft Fingers. *IEEE Transactions on Visualization and Computer Graphics*, 2015.

Book chapters

- **A. Talvas**, M. Marchal, G. Cirio, and A. Lécuyer. *Multi-finger Haptic Interaction*, chapter 3D Interaction Techniques for Bimanual Haptics in Virtual Environments, pages 31–53. Springer Series on Touch and Haptic Systems. Springer, 2013.

International conferences

- **A. Talvas**, M. Marchal, C. Nicolas, G. Cirio, M. Emily, and A. Lécuyer. Novel Interactive Techniques for Bimanual Manipulation of 3D Objects with Two 3DOF Haptic Interfaces. In *Proc. of Eurohaptics*, pages 552–563, 2012.
- **A. Talvas**, M. Marchal, and A. Lécuyer. The God-finger Method for Improving 3D Interaction with Virtual Objects through Simulation of Contact Area. In *Proc. of IEEE Symposium on 3D User Interfaces*, pages 111–114, 2013.
- M. Achibet, A. Girard, **A. Talvas**, M. Marchal, and A. Lécuyer. Elastic-Arm: Human-Scale Elastic Feedback for Augmenting Interaction and Perception in Virtual Environments. *IEEE Virtual Reality*, 2015.

National conferences

- **A. Talvas**, M. Marchal, C. Nicolas, G. Cirio, M. Emily, and A. Lécuyer. Novel Interactive Techniques for Bimanual Haptic Manipulation in Virtual Environments. In *Proc. of the 7th annual conference of the AFRV*, 2012.

Bibliography

J. Allard, S. Cotin, F. Faure, P. j. Bensoussan, F. Poyer, C. Duriez, H. Delingette, and L. G. B. SOFA: an open source framework for medical simulation. *Studies in Health Technolology and Informatics*, 125:13–18, 2007. 24

J. Allard, F. Faure, H. Courtecuisse, F. Falipou, C. Duriez, and P. G. Kry. Volume contact constraints at arbitrary resolution. *ACM Transactions on Graphics*, 29(4):82:1–82:10, 2010. 25, 69, 70, 73, 100

R. Balakrishnan and K. Hinckley. The role of kinesthetic reference frames in two-handed input performance. In *Proc. of ACM Symposium on User Interface Software and Technology*, pages 171–178, 1999. 10

R. Balakrishnan and K. Hinckley. Symmetric bimanual interaction. In *Proc. of ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 33–40, 2000. 11, 12, 13

R. Balakrishnan and G. Kurtenbach. Exploring bimanual camera control and object manipulation in 3d graphics interfaces. In *Proc. of ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 56–62, 1999. 31

S. Ballesteros, D. Manga, and J. Reales. Haptic discrimination of bilateral symmetry in 2-dimensional and 3-dimensional unfamiliar displays. *Perception & Psychophysics*, 59 (1):37–50, 1997. 11

D. Baraff. Coping with friction for non-penetrating rigid body simulation. *Computer Graphics*, 25(4):31–40, 1991. 25

D. Baraff. Fast contact force computation for nonpenetrating rigid bodies. In *Proc. of 21st Annual Conference on Computer Graphics and Interactive Techniques*, pages 23–34, 1994. 25

D. Baraff. Linear-time dynamics using lagrange multipliers. In *Proc. of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pages 137–146, 1996. 25

D. Baraff. An introduction to physically based modeling: Rigid body simulation. In *SIGGRAPH '97 Course Notes*, page 97, 1997. 24

F. Barbagli, J. Salisbry, K., and R. Devengenzo. Enabling multi-finger, multi-hand virtualized grasping. In *Proc. of IEEE International Conference on Robotics and Automation*, volume 1, pages 809–815, 2003. 18

F. Barbagli, A. Frisoli, K. Salisbury, and M. Bergamasco. Simulating human fingers: a soft finger proxy model and algorithm. In *Proc. of 12th International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 9 – 17, 2004. 26, 57

W. Baxter and M. C. Lin. Haptic interaction with fluid media. In *Proc. of Graphics Interfaces*, 2004. 24

M. Bergamasco, C. A. Avizzano, A. Frisoli, E. Ruffaldi, and S. Marcheschi. Design and validation of a complete haptic system for manipulative tasks. *Advanced Robotics*, 20: 367–389, 2006. 15, 21

E. A. Bier, M. C. Stone, K. Pier, W. Buxton, and T. D. DeRose. Toolglass and magic lenses: the see-through interface. In *Proc. of Conference on Computer Graphics and Interactive Techniques*, pages 73–80, 1993. 13

C. Borst and A. Indugula. Realistic virtual grasping. In *Proc. of IEEE Virtual Reality Conference*, pages 91–98, 2005. 26, 27

D. A. Bowman. *Interaction techniques for common tasks in immersive virtual environments: design, evaluation, and application.* PhD thesis, Georgia Institute of Technology, 1999. 31

M. Buss, A. Peer, T. Schauß, N. Stefanov, U. Unterhinninghofen, S. Behrendt, J. Leupold, M. Durkovic, and M. Sarkis. Development of a multi-modal multi-user telepresence and teleaction system. *The International Journal of Robotics Research*, 29:1298–1316, 2009. 17, 19, 20

W. Buxton. Chunking and phrasing and the design of human-computer dialogues. In *Proc. of IFIP World Computer Congress*, pages 475–480, 1986. 14

W. Buxton. The Natural Language of Interaction: A Perspective on Non-Verbal Dialogues. *INFOR: Canadian Journal of Operations Research and Information Processing*, 26(4): 428–438, 1988. 13, 14

W. Buxton and B. Myers. A study in two-handed input. In *Proc. of ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 321–326, 1986. 12, 13

C. R. Carignan and K. R. Cleary. Closed-loop force control for haptic simulation of virtual environments. *Haptics-e*, 1:01–1, 2000. 22

R. Carson. Neural pathways mediating bilateral interactions between the upper limbs. *Brain Research Reviews*, 49:641–662, 2005. 11

D. Casalta, Y. Guiard, and M. Beaudouin-Lafon. Evaluating two-handed input techniques: rectangle editing and navigation. In *CHI Extended Abstracts on Human Factors in Computing Systems*, pages 236–237, 1999. 12, 13, 14

M. Cavusoglu and F. Tendick. Multirate simulation for high fidelity haptic interaction with deformable objects in virtual environments. In *Proc. of IEEE International Conference on Robotics and Automation*, volume 3, pages 2458–2465, 2000. 24

B. Chang and J. E. Colgate. Real-time impulse-based simulation of rigid body systems for haptic display. In *Proc. of ASME International Mechanical Engineering Congress and Exhibition*, 1997. 24

M. Ciocarlie, C. Lackner, and P. Allen. Soft finger model with adaptive contact geometry for grasping and manipulation tasks. In *Proc. of World Haptics Conference*, pages 219–224, 2007. 26

G. Cirio, M. Marchal, S. Hillaire, and A. Lécuyer. Six degrees-of-freedom haptic interaction with fluids. *IEEE Transactions on Visualization and Computer Graphics*, 17:1714–1727, 2011. 15, 16, 24

M. Cline. Higher degree-of-freedom bimanual user interfaces for 3-d computer graphics. In *Proc. of Conference on Human Interface Technologies*, pages 41–46, 2000. 13, 14

J. Colgate, M. Stanley, and J. Brown. Issues in the haptic display of tool use. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 140 –145 vol.3, 1995. 22, 49, 65

D. Constantinescu, S. Salcudean, and E. Croft. Haptic rendering of rigid contacts using impulsive and penalty forces. *IEEE Transactions on Robotics*, 21(3):309–323, 2005. 24

P. Contensou. Couplage entre frottement de glissement et frottement de pivotement dans la théorie de la toupie. In *Kreiselprobleme / Gyrodynamics*, pages 201–216. Springer Berlin Heidelberg, 1963. 26, 75, 101

F. Conti, F. Barbagli, R. Balaniuk, M. Halg, C. Lu, D. Morris, L. Sentis, J. Warren, O. Khatib, and K. Salisbury. The CHAI libraries. In *Proc. of Eurohaptics*, pages 496–500, 2003. 28

E. Coumans. Bullet Physics. `www.bulletphysics.org`, 2012. 24

S. E. Criscimagna-Hemminger, O. Donchin, M. S. Gazzaniga, and R. Shadmehr. Learned dynamics of reaching movements generalize from dominant to nondominant arm. *Journal of Neurophysiology*, 89(1):168–176, 2003. 11

L. D. Cutler, B. Fröhlich, and P. Hanrahan. Two-handed direct manipulation on the responsive workbench. In *Proc. of ACM Symposium on Interactive 3D Graphics*, pages 107–114, 1997. 11, 13, 32, 33, 36, 87

M. de Pascale and D. Prattichizzo. The haptik library: A component based architecture for uniform access to haptic devices. *IEEE Robotics & Automation Magazine*, 14(4): 64–74, 2007. 28

M. de Pascale, G. de Pascale, D. Prattichizzo, and F. Barbagli. A GPU-friendly Method for Haptic and Graphic Rendering of Deformable Objects. In *Proc. of Eurohaptics*, pages 44–51, 2004. 24

M. de Pascale, G. Sarcuni, and D. Prattichizzo. Real-time soft-finger grasping of physically based quasi-rigid objects. *World Haptics Conference*, 0:545–546, 2005. 20

J.-G. S. Demers, J. Boelen, and I. Sinclair. Freedom 6s force feedback hand controller. In *Proc. of IFAC Workshop on Space Robotics*, 1998. 15, 20, 21

R. F. Dillon, J. D. Edey, and J. W. Tombaugh. Measuring the true cost of command selection: techniques and results. In *Proc. of ACM SIGCHI Conference on Human Factors in Computing Systems: Empowering People*, pages 19–26, 1990. 13, 14

Y. Dobashi, M. Sato, S. Hasegawa, T. Yamamoto, M. Kato, and T. Nishita. A fluid resistance map method for real-time haptic interaction with fluids. In *Proc. of the ACM Symposium on Virtual Reality Software and Technology*, pages 91–99, 2006. 24

L. Dominjon, A. Lécuyer, J.-M. Burkhardt, G. Andrade-Barroso, and S. Richir. The "bubble" technique: Interacting with large virtual environments using haptic devices with limited workspace. In *Proc. of World Haptics Conference*, pages 639–640, 2005. 32, 33, 36, 40, 87, 94

C. Duriez, S. Member, F. Dubois, A. Kheddar, and C. Andriot. Realistic haptic rendering of interacting deformable objects in virtual environments. *IEEE Transactions on Visualization and Computer Graphics*, 12:36–47, 2006. 15, 24, 25, 71, 73

C. Duriez, H. Courtecuisse, J. Alcalde, and P. Bensoussan. Contact skinning. In *Proc. of Eurographics*, volume 27, pages 313–320, 2008. 25, 77

T. Endo, H. Kawasaki, T. Mouri, Y. Doi, T. Yoshida, Y. Ishigure, H. Shimomura, M. Matsumura, and K. Koketsu. Five-fingered haptic interface robot: Hiro iii. In *Proc. of World Haptics - Third Joint EuroHaptics conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 458 –463, 2009. 19

K. Erleben. Velocity-based shock propagation for multibody dynamics animation. *ACM Transactions on Graphics*, 26(2), 2007. 25, 71

A. Faeth, M. Oren, J. Sheller, S. Godinez, and C. Harding. Cutting, deforming and painting of 3d meshes in a two handed viso-haptic vr system. In *Proc. of IEEE Virtual Reality Conference*, pages 213 –216, 2008. 15, 29, 34

A. J. Faeth. Expressive cutting, deforming, and painting of three-dimensional digital shapes through asymmetric bimanual haptic manipulation. Master's thesis, Iowa State University, 2009. 35

F. Faure, C. Duriez, H. Delingette, J. Allard, B. Gilles, S. Marchesseau, H. Talbot, H. Courtecuisse, G. Bousquet, I. Peterlik, and S. Cotin. SOFA: A multi-model framework for interactive physical simulation. In *Soft Tissue Biomechanical Modeling for Computer Assisted Surgery*, volume 11, pages 283–321. Springer Berlin Heidelberg, 2012. 24, 67, 78

C. A. Felippa. A systematic approach to the element-independent corotational dynamics of finite element. Technical report, Center for Aerospace Structures, 2000. 24

A. Fischer and J. M. Vance. Phantom haptic device implemented in a projection screen virtual environment. In *Proc. of Workshop on Virtual Environments*, EGVE '03, pages 225–229. ACM, 2003. 32

A. Formaglio, M. de Pascale, and D. Prattichizzo. A mobile platform for haptic grasping in large environments. *Virtual Reality, Springer - Special Issue on Multisensory Interaction in Virtual Environments*, 10:11–23, 2006. 16, 17, 21, 29

A. Frisoli, F. Barbagli, E. Ruffaldi, K. Salisbury, and M. Bergamasco. A limit-curve based soft finger god-object algorithm. In *14th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 217–223, 2006. 26

N. Galoppo, M. A. Otaduy, S. Tekin, M. Gross, and M. C. Lin. Soft articulated characters with fast contact handling. *Computer Graphics Forum*, 26(3):243–253, 2007. 25

P. Garcia-Robledo, J. Ortego, J. Barrio, I. Galiana, M. Ferre, and R. Aracil. Multifinger haptic interface for bimanual manipulation of virtual objects. In *Proc. of IEEE International Workshop on Haptic Audio-visual Environments and Games*, pages 30 –35, 2009. 19, 20, 21, 23

P. Garcia-Robledo, J. Ortego, M. Ferre, J. Barrio, and M. Sanchez-Uran. Segmentation of bimanual virtual object manipulation tasks using multifinger haptic interfaces. *IEEE Transactions on Instrumentation and Measurement*, 60(1):69 –80, 2011. 29, 30, 36

C. Garre, F. Hernandez, A. Gracia, and M. Otaduy. Interactive simulation of a deformable hand for haptic rendering. In *Proc. of World Haptics Conference*, pages 239–244, 2011. 27

L. Glondu, M. Marchal, and G. Dumont. A new coupling scheme for haptic rendering of rigid bodies interactions based on a haptic sub-world using a contact graph. In A. M. Kappers, J. B. van Erp, W. M. Bergmann Tiest, and F. C. van der Helm, editors, *Haptics: Generating and Perceiving Tangible Sensations*, volume 6191 of *Lecture Notes in Computer Science*, pages 51–56. Springer Berlin Heidelberg, 2010. 88

D. Goble, B. Noble, and B. S. H. Proprioceptive target matching asymmetries in left-handed individuals. *Experimental Brain Research*, 197(4):403–408, 2009. 10, 13

D. J. Goble and S. H. Brown. Upper limb asymmetries in the matching of proprioceptive versus visual targets. *Journal of Neurophysiology*, 99:3063–3074, 2008. 10, 13

J.-P. Gourret, N. M. Thalmann, and D. Thalmann. Simulation of object and human skin formations in a grasping task. *SIGGRAPH Computer Graphics*, 23(3):21–30, 1989. 26, 27

Y. Guiard. Asymmetric division of labor in human skilled bimanual action: the kinematic chain as a model. *Journal of Motor Behavior*, 19(4):486–517, 1987. 11, 13, 37

L. R. Harley and B. I. Prilutsky. Transfer of learning between the arms during bimanual reaching. In *Proc. of IEEE Engineering in Medicine and Biology Society*, pages 6785–6788, 2012. 11, 13

V. Hayward, P. Gregorio, O. Astley, S. Greenish, M. Doyon, L. Lessard, J. Mcdougall, I. Sinclair, S. Boelen, X. Chen, J. P. Demers, J. Poulin, I. Benguigui, N. Almey, B. Makuc, and X. Zhang. Freedom-7: A high fidelity seven axis haptic device with application to surgical training. In *Lecture Notes in Control and Information Science 232*, pages 445–456, 1997. 15, 21

K. Hinckley, R. Pausch, and D. Proffitt. Attention and visual feedback: the bimanual frame of reference. In *Proc. of Symposium on Interactive 3D Graphics*, pages 121–ff., 1997a. 10, 12, 13, 14

K. Hinckley, R. Pausch, D. Proffitt, J. Patten, and N. Kassell. Cooperative bimanual action. In *Proc. of ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 27–34, 1997b. 11, 13

K. Hinckley, R. Pausch, D. Proffitt, and N. F. Kassell. Two-handed virtual manipulation. *ACM Transactions on Computer-Human Interaction*, 5:260–302, 1998. 10, 12, 13, 14

D. Holz, S. Ullrich, M. Wolter, and T. Kuhlen. Multi-contact grasp interaction for virtual environments. *Journal of Virtual Reality and Broadcasting*, 5(7), 2008. 34, 35

R. D. Howe and M. R. Cutkosky. Practical force-motion models for sliding manipulation. *International Journal of Robotics Research*, 15(6):557–572, 1996. 26

T. Hulin, P. Kremer, R. Scheibe, S. Schaetzle, and C. Preusche. Evaluating two novel tactile feedback devices. In *Proc. of International Conference on Enactive Interfaces*, 2007. 18

T. Hulin, M. Sagardia, J. Artigas, S. Schätzle, P. Kremer, and C. Preusche. Human-scale bimanual haptic interface. In *Proc. of International Conference on Enactive Interfaces*, pages 28–33, 2008. 16, 19, 21

T. Hulin, K. Hertkorn, P. Kremer, S. Schatzle, J. Artigas, M. Sagardia, F. Zacharias, and C. Preusche. The dlr bimanual haptic device with optimized workspace. In *Proc. of IEEE International Conference on Robotics and Automation*, pages 3441 –3442, 2011. 18, 19, 32

G. Irving, J. Teran, and R. Fedkiw. Invertible finite elements for robust simulation of large deformation. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 131–140, 2004. 24

M. Ishii and M. Sato. A 3d spatial interface device using tensed strings. *Presence*, 3(1): 81–86, 1994. 19

M. Ishii, P. Sukanya, and M. Sato. A virtual work space for both hands manipulation with coherency between kinesthetic and visual sensation. In *Proc. of Fourth International Symposium on Measurement and Control in Robotics*, pages 84–90, 1994. 19, 21

M. Isshiki, T. Sezaki, K. Akahane, N. Hashimoto, and S. M. A proposal of a clutch mechanism for 6dof haptic devices. In *Proc. of 18th International Conference on Artificial Reality and Telexistence*, pages 57–63, 2008. 32

J. Jacobs and B. Froehlich. A soft hand model for physically-based manipulation of virtual objects. In *Proc. of IEEE Virtual Reality Conference*, pages 11 –18, 2011. 27

J. Jacobs, M. Stengel, and B. Froehlich. A generalized god-object method for plausible finger-based interactions in virtual environments. In *Proc. of IEEE Symposium on 3D User Interfaces*, pages 43–51, 2012. 26, 27

P. Jacobs and M. Cavusoglu. High fidelity haptic rendering of stick-slip frictional contact with deformable objects in virtual environments using multi-rate simulation. In *Proc. of IEEE International Conference on Robotics and Automation*, pages 117–123, 2007. 24, 88

S. Jain and C. K. Liu. Controlling physics-based characters using soft contacts. In *Proc. of SIGGRAPH Asia Conference*, pages 163:1–163:10, 2011. 26

P. Kabbash, W. Buxton, and A. Sellen. Two-handed input in a compound task. In *Proc. of ACM SIGCHI Conference on Human Factors in Computing Systems*, pages 417–423, 1994. 13, 14

B. H. Kantowitz. Effects of response symmetry upon bi-manual rapid aiming. In *Proc. of Human Factors and Ergonomics Society Annual*, volume 35, pages 1541–1545, 1991. 13, 14

S. Kim, J. J. Berkley, and M. Sato. A novel seven degree of freedom haptic device for engineering design. *Virtual Reality*, 6(4):217–228, 2003a. 16

Y. J. Kim, M. A. Otaduy, M. C. Lin, and D. Manocha. Six-degree-of-freedom haptic rendering using incremental and localized computations. *Presence: Teleoperators and Virtual Environments*, 12(3):277–295, 2003b. 25

A. Kron and G. Schmidt. Multi-fingered tactile feedback from virtual and remote environments. In *Proc. of Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 16–23, 2003. 18

A. Kron, G. Schmidt, B. Petzold, M. Zah, P. Hinterseer, and E. Steinbach. Disposal of explosive ordnances by use of a bimanual haptic telepresence system. In *Proc. of IEEE International Conference on Robotics and Automation*, volume 2, pages 1968–1973, 2004. 15, 19, 20, 22, 29

P. G. Kry and D. K. Pai. Continuous contact simulation for smooth surfaces. *ACM Transactions on Graphics*, 22(1):106–129, 2003. 23

P. G. Kry and D. K. Pai. Interaction capture and synthesis. *ACM Transactions on Graphics*, 25(3):872–880, 2006. 26

P. G. Kry, A. Pihuit, A. Bernhardt, and M.-P. Cani. Handnavigator: Hands-on interaction for desktop virtual reality. In *Proc. of ACM Symposium on Virtual Reality Software and Technology*, pages 53–60, 2008. 26

A. Leganchuk, S. Zhai, and W. Buxton. Manual and cognitive benefits of two-handed input: an experimental study. *ACM Transactions on Computer-Human Interaction*, 5: 326–359, 1998. 12, 13, 14

R. I. Leine and C. Glocker. A set-valued force law for spatial Coulomb-Contensou friction. *European Journal of Mechanics - A/Solids*, 22(2):193–216, 2003. 26, 75, 76, 101

G. Leonard and B. Milner. Contribution of the right frontal lobe to the encoding and recall of kinesthetic distance information. *Neuropsychologia*, 29(1):47–58, 1991. 10

R. Lindeman, J. Sibert, and J. Hahn. Hand-held windows: towards effective 2d interaction in immersive virtual environments. In *Proc. of IEEE Virtual Reality*, pages 205 –212, 1999. 35

C. K. Liu. Synthesis of interactive hand manipulation. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 163–171, 2008. 26

K. Machulis. libnifalcon. `qdot.github.com/libnifalcon`, 2009. 28

N. Maruyama, L. Liu, K. Akahane, and M. Sato. A proposal of two-handed multi-finger haptic interface with rotary frame. In *Proc. of 23rd International Conference on Artificial Reality and Telexistence*, pages 40–45, 2013. 19, 20, 21

E. Miguel and M. A. Otaduy. Efficient simulation of contact between rigid and deformable objects. In *ECCOMAS - Multibody Dynamics*, 2011. 4, 25

V. J. Milenkovic and H. Schmidl. Optimization-based animation. In *Proc. of 28th Annual Conference on Computer Graphics and Interactive Techniques*, pages 37–46, 2001. 25

K. Minamizawa, S. Kamuro, S. Fukamachi, N. Kawakami, and S. Tachi. Ghostglove: haptic existence of the virtual world. In *Proc. of ACM SIGGRAPH*, page 134, 2008a. 18, 21

K. Minamizawa, S. Kamuro, N. Kawakami, and S. Tachi. A palm-worn haptic display for bimanual operations in virtual environments. In *Haptics: Perception, Devices and Scenarios*, volume 5024, pages 458–463. Springer, 2008b. 18

M. R. Mine, F. P. Brooks Jr., and C. H. Sequin. Moving objects in space: exploiting proprioception in virtual-environment interaction. In *Proc. of Conference on Computer Graphics and Interactive Techniques*, pages 19–26, 1997. 11, 13

B. Mirtich and J. Canny. Impulse-based simulation of rigid bodies. In *Proc. of Symposium on Interactive 3D Graphics*, 1995. 24

M. Moehring and B. Froehlich. Pseudo-physical interaction with a virtual car interior in immersive environments. In *Proc. of Eurographics Conference on Virtual Environments*, pages 181–189, 2005. 34

M. Moehring and B. Froehlich. Enabling functional validation of virtual cars through natural interaction metaphors. In *Proc. of IEEE Virtual Reality Conference*, pages 27–34, 2010. 34, 87

M. Moehring and B. Froehlich. Effective manipulation of virtual objects within arm's reach. In *Proc. of IEEE Virtual Reality Conference*, pages 131–138, 2011. 18

M. Moore and J. Wilhelms. Collision detection and response for computer animation. *SIGGRAPH Computer Graphics*, 22(4):289–298, 1988. 24

J. Mora and W.-S. Lee. Real-time 3d fluid interaction with a haptic user interface. In *Proc. of IEEE Symposium on 3D User Interfaces*, pages 75–81, 2008. 24

A. Moravanszky and P. Terdiman. Fast contact reduction for dynamics simulation. In A. Kirmse, editor, *Game Programming Gems 4*, pages 253–263. Charles River Media, 2004. 25, 30

M. Müller and M. Gross. Interactive virtual materials. In *Proc. of Graphics Interface*, pages 239–246, 2004. 77

J. Murayama, L. Bouguila, Y. Luo, K. Akahane, S. Hasegawa, B. Hirsbrunner, and M. Sato. SPIDAR G&G: A two-handed haptic interface for bimanual vr interaction. In *Proc. of EuroHaptics*, pages 138–146, 2004. 16, 17, 20, 21

A. Nealen, M. Müller, R. Keiser, E. Boxerman, and M. Carlson. Physically based deformable models in computer graphics. *Computer Graphics Forum*, 25(4):809–836, 2006. 24

M. Nesme, M. Marchal, E. Promayon, M. Chabanas, Y. Payan, and F. Faure. Physically realistic interactive simulation for biological soft tissues. *Recent Research Developments in Biomechanics*, 2:1–22, 2005. 24

M. Ortega, S. Redon, and S. Coquillart. A six degree-of-freedom god-object method for haptic display of rigid bodies with surface properties. *IEEE Transactions on Visualization and Computer Graphics*, 13(3):458–469, 2007. 22, 23, 26, 33

M. A. Otaduy, R. Tamstorf, D. Steinemann, and M. Gross. Implicit contact handling for deformable objects. *Computer Graphics Forum*, 28(2):559–568, 2009. 25, 71

R. Ott. *Two-handed haptic feedback in generic virtual environments*. PhD thesis, Ecole Polytechnique Fédérale de Lausanne, 2009. 27, 28, 30, 36

R. Ott, V. De Perrot, D. Thalmann, and F. Vexo. MHaptic: a haptic manipulation library for generic virtual environments. In *Proc. of International Conference on Cyberworlds*, pages 338–345, 2007. 19, 20, 21, 23, 26, 29, 32, 40, 87

R. Owen, G. Kurtenbach, G. Fitzmaurice, T. Baudel, and B. Buxton. When it gets more difficult, use both hands: exploring bimanual curve manipulation. In *Proc. of Graphics Interface*, pages 17–24, 2005. 12, 13, 14

V. Panday, W. Tiest, and A. Kappers. Bimanual integration of position and curvature in haptic perception. *IEEE Transactions on Haptics*, 6(3):285–295, 2013. 10

M. Pauly, D. K. Pai, and L. J. Guibas. Quasi-rigid objects in contact. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 109–119, 2004. 25

A. Peer and M. Buss. A new admittance-type haptic interface for bimanual manipulations. *IEEE/ASME Transactions on Mechatronics*, 13(4):416 –428, 2008. 16, 17, 20, 21

A. Peer, U. Unterhinninghofen, and M. Buss. Tele-assembly in wide remote environments. In *Proc. of 2nd International Workshop on Human-Centered Robotic Systems*, 2006. 22, 29, 32

J. B. Pelz, M. M. Hayhoe, D. H. Ballard, A. Shrivastava, J. D. Bayliss, and M. von der Heyde. Development of a virtual laboratory for the study of complex human behavior. In *Proc. of SPIE*, volume 3639B, 1999. 15

G. Picinbono, H. Delingette, and N. Ayache. Non-linear anisotropic elasticity for real-time surgery simulation. *Graphical Models*, 65(5):305–321, 2003. 24

M. A. Plaisier and M. O. Ernst. Two hands perceive better than one. In *Haptics: Perception, Devices, Mobility, and Communication*, volume 7283, pages 127–132. Springer, 2012. 11

M. Pouliquen, C. Duriez, C. Andriot, A. Bernard, L. Chodorge, and F. Gosselin. Real-time finite element finger pinch grasp simulation. In *Proc. of World Haptics Conference*, pages 323–328, 2005. 27

G. Rains and B. Milner. Right-hippocampal contralateral-hand effect in the recall of spatial location in the tactual modality. *Neuropsychologia*, 32:1233–1242, 1994. 10

D. C. Ruspini, K. Kolarov, and O. Khatib. The haptic display of complex graphical environments. In *Proc. of 24th Annual Conference on Computer Graphics and Interactive Techniques*, pages 345–352, 1997. 22, 33

R. L. Sainburg and J. Wang. Interlimb transfer of visuomotor rotations: independence of direction and final position information. *Experimental Brain Research*, 145(4):437–447, 2002. 11

A. F. Sanders and A. M. Kappers. Bimanual curvature discrimination of hand-sized surfaces placed at different positions. *Perception & Psychophysics*, 68(7):1094–1106, 2006. 10

M. Sato. Spidar and virtual reality. In *Proc. of the 5th Biannual World Automation Congress*, volume 13, pages 17 – 23, 2002. 18

S. Schatzle, T. Ende, T. Wusthoff, and C. Preusche. Vibrotac: An ergonomic and versatile usable vibrotactile feedback device. In *Proc. of IEEE International Workshop on Robots and Human Interactive Communications*, pages 670–675, 2010. 18

R. Scheibe, M. Moehring, and B. Froehlich. Tactile feedback at the finger tips for improved direct interaction in immersive environments. In *Proc. of IEEE Symposium on 3D User Interfaces*, pages 123–130, 2007. 18

R. Smith. Open Dynamics Engine. www.ode.org, 2011. 24

V. Squeri, A. Sciutti, M. Gori, L. Masia, G. Sandini, and J. Konczak. Two hands, one perception: how bimanual haptic information is combined by the brain. *Journal of Neurophysiology*, 107:544–550, 2012. 10

M. A. Srinivasan. *Virtual Reality: Scientific and Technical Challenges*, chapter Haptic interfaces, pages 161–187. National Academy Press, 1995. 3

S. Sueda, A. Kaufman, and D. K. Pai. Musculotendon simulation for hand animation. *ACM Transactions on Graphics*, 27(3):83:1–83:8, 2008. 26

L.-W. Sun, F. Van Meer, Y. Bailly, and C. K. Yeung. Design and development of a da vinci surgical system simulator. In *Proc. of International Conference on Mechatronics and Automation*, pages 1050 –1055, 2007. 15

M. Ueberle, N. Mock, and M. Buss. Vishard10, a novel hyper-redundant haptic interface. In *Proc. of International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 58–65, 2004. 15, 21

A. Ulinski, C. Zanbaka, Z. Wartell, P. Goolkasian, and L. Hodges. Two handed selection techniques for volumetric data. In *Proc. of IEEE Symposium on 3D User Interfaces*, pages 107–114, 2007. 12

S. Ullrich and T. Kuhlen. Haptic palpation for medical simulation in virtual environments. *IEEE Transactions on Visualization and Computer Graphics*, 18(4):617–625, 2012. 15, 22, 35

S. Ullrich, T. Knott, Y. Law, O. Grottke, and T. Kuhlen. Influence of the bimanual frame of reference with haptics for unimanual interaction tasks in virtual environments. In *Proc. of IEEE Symposium on 3D User Interfaces*, pages 39 –46, 2011a. 11, 13, 35, 36, 87

S. Ullrich, D. Rausch, and T. Kuhlen. Bimanual haptic simulator for medical training: System architecture and performance measurement. In *Joint Virtual Reality Conference of EuroVR*, 2011b. 24, 29, 30

G. Vallar. Spatial hemineglect in humans. *Trends Cogn. Sci. (Regul. Ed.)*, 2(3):87–97, 1998. 10

M. Veit, A. Capobianco, and D. Bechmann. Consequence of two-handed manipulation on speed, precision and perception on spatial input task in 3d modelling applications. *Journal of Universal Computer Science*, 14(19):3174–3187, 2008. 10, 13, 14

F. Vidal, P. Villard, R. Holbrey, N. John, F. Bello, A. Bulpitt, and D. Gould. Developing an immersive ultrasound guided needle puncture simulator. *Studies in Health Technolology and Informatics*, 142:398–400, 2009. 24

F. P. Vidal, N. W. John, D. A. Gould, and A. E. Healey. Simulation of ultrasound guided needle puncture using patient specific data with 3D textures and volume haptics. *Computer Animation and Virtual Worlds*, 19(2):111–127, 2008. 14

M. von der Heyde and C. HÃ¤ger-ross. Psychophysical experiments in a complex virtual environment. In *Proc. of Third PHANToM User Group Workshop*, 1998. 15

S. Walairacht, M. Ishii, Y. Koike, and M. Sato. Two-handed multi-fingers string-based haptic interface device. In *Proc. of IEICE Transactions on Information and Systems*, volume E84D, pages 365–373, 2001. 19, 21

K. Waldron and K. Tollon. Mechanical characterization of the immersion corp. haptic, bimanual, surgical simulator interface. In B. Siciliano and P. Dario, editors, *Experimental Robotics VIII*, volume 5, pages 106–112. Springer, 2003. 16, 17

J. Wang and R. Sainburg. The dominant and nondominant arms are specialized for stabilizing different features of task performance. *Experimental Brain Research*, 178(4): 565–570, 2007. 11

D. M. Wolpert, S. J. Goodbody, and M. Husain. Maintaining internal representations: the role of the human superior parietal lobe. *Nature Neuroscience*, 1(6):529–533, 1998. 10

K. Yamane and Y. Nakamura. Stable penalty-based model of frictional contacts. In *Proc. of IEEE International Conference on Robotics and Automation*, pages 1904–1909, 2006. 24

M. Yang, A. Safonova, J. Lu, K. J. Kuchenbecker, and Z. Zhou. A gpu-based approach for real-time haptic rendering of 3d fluids. In *Proc. of ACM SIGGRAPH Asia Sketches*, 2009. 24

Y. Ye and C. K. Liu. Synthesis of detailed hand manipulations using contact sampling. *ACM Transactions on Graphics*, 31(4):41:1–41:10, 2012. 26

T. Yoshikawa, T. Endo, T. Maeno, and H. Kawasaki. Multi-fingered bimanual haptic interface with three-dimensional force presentation. In *Proc. of International Symposium on Robot Control*, pages 811–816, 2009. 19, 20, 21

G. Zachmann and A. Rettig. Natural and robust interaction in virtual assembly simulation. In *Proc. of ISPE International Conference on Concurrent Engineering: Research and Applications*, 2001. 34

R. C. Zeleznik, A. S. Forsberg, and P. S. Strauss. Two pointer input for 3d interaction. In *Proc. of Symposium on Interactive 3D Graphics*, pages 115–120, 1997. 13, 14

S. Zhai. User performance in relation to 3d input device design. *SIGGRAPH Comput. Graph.*, 32(4):50–54, 1998. 32

S. Zhai, E. Kandogan, B. A. Smith, and T. Selker. In search of the 'magic carpet': Design and experimentation of a bimanual 3d navigation interface. *Journal of Visual Languages and Computing*, 10:3–17, 1999. 31, 32

Y. Zhuang and J. Canny. Real-time simulation of physically realistic global deformation. In *Proc. of IEEE Visualization Conference*, 1999. 24

C. Zilles and J. Salisbury. A constraint-based god-object method for haptic display. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 146–151, 1995. 22, 33

INSA de RENNES
Service des Formations

AVIS DU JURY SUR LA REPRODUCTION DE LA THESE SOUTENUE

Titre de la thèse:
Interaction haptique bimanuelle avec des environnements virtuels

Nom Prénom de l'auteur : TALVAS ANTHONY

Membres du jury :
- Monsieur AMMI Medhi
- Monsieur KRY Paul G.
- Monsieur LECUYER Anatole
- Monsieur ARNALDI BRUNO
- Madame MARCHAL Maud
- Monsieur CASIEZ Géry
- Monsieur FAURE François

Président du jury : Bruno ARNALDI

Date de la soutenance : 01 Décembre 2014

Reproduction de la these soutenue

Thèse pouvant être reproduite en l'état
~~Thèse pouvant être reproduite après corrections suggérées~~

Fait à Rennes, le 01 Décembre 2014

Signature du président de jury

Le Directeur,

M'hamed DRISSI

Résumé

En Réalité Virtuelle (RV), le sens haptique accroît l'immersion d'un utilisateur dans un Environnement Virtuel (EV) avec lequel il interagit en temps réel. Dans cette thèse, nous proposons des approches pour améliorer l'interaction haptique à deux mains avec des EV.

Nous abordons d'abord des problèmes avec l'interaction bimanuelle dans des EV avec des interfaces à effecteur unique. Nous proposons une technique d'interaction nommée la *double bulle* pour l'exploration d'EV avec une combinaison de contrôle en position et en vitesse. Nous présentons aussi une technique de manipulation nommée *prise magnétique* qui facilite la saisie d'objets virtuels avec des proxys rigides simples. Des modes de contrôle communs sont utilisés pour améliorer la saisie et l'exploration simultanées. Une évaluation utilisateur a été réalisée pour mesurer l'efficacité de ces techniques.

Nous nous intéressons ensuite au calcul de surfaces de contact. Nous proposons une technique nommée *god-finger* pour rendre des surfaces similaires à celles générées par des doigts à partir d'un unique point de contact. Elle est basée sur un simple parcours de la géométrie locale de l'objet en contact, et est donc moins coûteuse que des méthodes de simulation de corps souples. La méthode est adaptée pour l'interaction avec des proxys rigides simples ou plus complexes, ainsi qu'avec des objets rigides ou déformables, y compris avec des surfaces rugueuses. Une méthode de rendu visuel donne un retour à l'utilisateur sur la forme de la surface de contact.

Enfin, nous abordons la résolution de contacts durant la manipulation dextre d'objets virtuels avec des doigts souples. Le calcul des mécaniques de contact est amélioré en agrégeant les multiples contraintes de contact concernées. Une méthode de distribution de pression non uniforme sur la surface de contact adapte la réponse lors d'un contact contre des arêtes pointues. Nous utilisons le modèle de frottement de Coulomb-Contensou pour simuler efficacement le frottement en torsion. L'approche est évaluée avec un modèle de main déformable pour de l'interaction en temps réel.

Les travaux présentés dans ce manuscrit ouvrent de nouvelles perspectives dans le contexte de l'haptique bimanuelle et de la RV, en permettant une interaction plus naturelle avec des EV plus complexes.

Abstract

In Virtual Reality (VR), the haptic sense increases the immersion of users in a Virtual Environment (VE) with which they interact in real time. In this Ph.D thesis, we propose contributions to improve two-handed interaction in haptics with VEs.

We first address issues with bimanual interaction in VEs using haptic devices with single effectors. We propose an interaction technique called *double bubble* for exploration of VEs through a combination of position and rate control. We also present a manipulation technique called *magnetic pinch* which facilitates the grasping of virtual objects with simple rigid proxies. Simultaneous grasping and exploration of the VE is enhanced using common control modes. A user evaluation was conducted to assess the efficiency of these techniques.

We then focus on improving the computation of contact surfaces. We propose a *god-finger* method to render finger pad-like surfaces from a single contact point. It relies on a simple scan of the local geometry of the object in contact, and is this less costly than soft body simulation methods. The method is adapted for interaction using simple or more complex rigid virtual proxies, and with rigid or deformable objects, including rough surfaces. A visual rendering method provides feedback on the shape of the contact surface.

Finally, we address the resolution of contacts during dexterous manipulation of virtual objects through soft fingers. The computation of contact mechanics is improved by aggregating the multiple contact constraints involved. A method for non-uniform pressure distribution over the contact surface adapts the response when touching sharp edges. We use the Coulomb-Contensou friction model to efficiently simulate torsional friction. The approach is evaluated with a deformable hand model for real time interaction.

The contributions of this manuscript open novel perspectives in the context of bimanual haptics and VR, allowing more natural interaction with more complex VEs.

N° d'ordre : **14ISAR 38 – D14 - 38**

Institut National des Sciences Appliquées de Rennes
20, Avenue des Buttes de Coëmes • CS 70839 • F-35708 Rennes Cedex
Tel : 02 23 23 82 00 - Fax : 02 23 23 83 96