University of Verona
Department of Computer Science

# Gestural interaction in Virtual Environments: user studies and applications

Fabio Marco Caputo

Supervisor: Prof. Andrea Giachetti

Submitted in part fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science of the University of Verona.

April 2019

## Abstract

With the currently available technology, there has been an increased interest in the development of virtual reality (VR) applications, some of those already becoming commercial products. This fact also rose many issues related to their usability. One of the main challenges is the design of interfaces to interact with these immersive virtual environments (IVE), in particular for those setups relying on hand tracking as a mean of input and in general with devices that stray away from the standard choices such as keyboards and mice. Finding appropriate ways to deal with the usability issues is key to the success of applications in VR, not only for entertainment purposes but also for professional application in medical and engineering field. The research on this topic in the last years has been quite active and several problems were identified as relevant to achieve satisfying usability results. This research project aims to tackle some of those critical aspects strictly related to interaction in IVEs. It is firstly focused on object manipulation. An initial evaluation allowed us to highlight the nature of some critical issues from which we derived some design guidelines for manipulation techniques. We proposed a different number of solutions and performed user tests to validate them. Secondly, we tried to verify the feasibility of gestural interfaces in applications by developing and testing a gesture recognition algorithm based on 3D hand trajectories. This work showed promising results in terms of accuracy hinting at the possibility of a reliable gestural interface for VR applications.

# Acknowledgements

These years for me have been the toughest but I was able to carry out the work presented in this thesis thanks to all the people that have been with me along the way. I'd like to express my deepest gratitude to my tutor Prof. Andrea Giachetti for showing and teaching me how to take my first steps in the world of research. His experience and dedication were key, in order to complete my work. I would also like to thank all the professors, researchers and all the people of our department that supported me with precious suggestions, feedback and help, all extremely useful for my work and professional growth. My sincere thanks will also go to Daniel Mendes, Prof. Alfredo Ferreira and Prof. Joaquim Jorge at INESC-ID of Lisbon for the opportunity of working together and sharing ideas on the research topics of our interest.

To my family and my friends: thank you for staying with me during the roughest times and now more than ever. I owe you everything. In particular to my parents and sister who never gave up on me. To my dear friend Paolo for always pushing me onward no matter what. To my beloved friend Arthur for being so close to me all the time despite the distance separating us: I can't wait to see you. Thank you so much!

*To my little bunny.*

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

## 1.1 Virtual Reality

Virtual Reality (VR), is a term used for those applications that can simulate physical presence in places of the real world or imagined worlds. These VR applications recreate sensory experiences, which could in theory also include virtual taste, sight, smell, sound, touch, etc. Most current VR environments are mainly composed of visual experiences, through the use of a video monitor or with special stereoscopic displays for a more immersive experience. Some additional sensory information like sound, through headphones or speakers orientated towards users, or special gloves that can provide haptic feedback, etc. can greatly enhance the experience. VR is an object of interest for a variety of applications. Besides amusement and entertainment, with VR movies and games there is a strong interest in its application for other purposes such as Virtual Training, integration with rehab programmes, immersive 3D design and modeling, 3D data visualization and so on.

VR however, in its general meaning, is quite a broad term including several kinds of experiences ranging from interactive to non-interactive, visual to non-visual, immersive to non-immersive ones, etc., with each condition having its own specific issues. In this thesis, we went through some of those related to the interactive immersive VR applications that feature what are called Immersive Virtual Environments (IVEs). IVEs are subject to a different number of problems that can either technical or interaction ones. Technical issues are those that have to do with hardware limits and derived software ones. In fact, only recently, the technological advancements brought back VR as a topic of interest not only in research but also on the market of technological entertainment. Examples of technical related issues are: low framerates, low resolution, non-stereo aliasing, and examples of software limits derived from these limits are all those rendering techniques such as texture mapping and bump mapping that don't work really well in VR compared to standard graphics applications. Interaction issues, on the other hand, are all those that fall in the HCI area and also the focus of our work.

Figure 1.1: Two of the most popular Head Mounted Displays (HDMS) and their controllers: Oculus Rift (left) and HTC Vive (right).

While it is important to keep in mind that both these kind of problems ultimately affect the usability of a VR application, at this time, the technical ones are considered solved at least when it comes to a certain type of setups, namely those involving

Head Mounted Displays (HDMs) for stereo visualization and particular controllers as input devices. To this day there are several solutions such as Oculus Rift, HTC Vive, PlayStation VR (Figure 1.1) and many others manufactured by other popular brands. From the hardware point of view, all these devices offer a decent user experience (UX) that is still, however, far from ideal. For example, these devices are still quite invasive with most of the HMDs being still too heavy for comfortable long sessions in IVEs. The controller devices provided, despite having very accurate tracking, may not be very practical to use for every task. An alternative to controller devices are all the available hand trackers that allow performing mid-air deviceless interaction with the user's hands. There are currently different low-cost hand trackers available such as Leap Motion Controller and Intel Real Sense (Figure 1.2). Most of them work similarly to Microsoft Kinect, by projecting infrared structured light and using infrared cameras to obtain 3D data in the form of point clouds. Usually, the Application Programming Interface (API) provided by the Software Development Kits (SDKs) of such devices are sufficient to track human hands but, while these devices introduce non-invasive and practicality features, they also come with a trade-off in terms of tracking accuracy. This lack of accuracy is often derived from specific conditions in which the power of the tracker and/or its API is simply not enough. The study of these kinds of conditions and the specific causes of tracking issues are also part of this research.

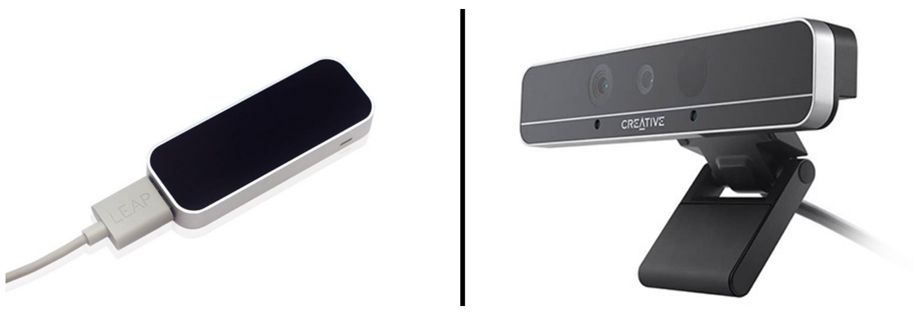


Figure 1.2: Examples of low-cost hand trackers: Leap Motion Controller (left) and Intel Real Sense (right).

## 1.2 Interaction Issues

Since the early days of virtual environments, interaction with virtual objects in a scene has been one of the main object of studies. Considering three-dimensional virtual environments, interaction isnt trivial, mainly due to the required mapping between traditional input devices (2D) and the virtual environment (3D). Most common solutions resort to techniques that somehow relate the actions performed in the two-dimensional space of the input device (e.g. mouse cursor or touch) to three-dimensional transformations. Since it is usual for people to interact with this kind of environments with traditional displays, 3D content is displayed in a 2D rendered image, which hinders contents perception. To overcome both the limitations of the input and the output devices, mainstream solutions for creating and editing 3D virtual content, namely computer-aided design (CAD) tools, resort to different orthogonal views of the environment. This allows a more direct two-dimensional interaction with limited degrees of freedom. Solutions that offer a single perspective view usually either apply the transformation in a plane parallel to the view plane or resort to widgets that constraint interactions and ease the 2D-3D mapping [33]. Research has shown that the first approach can sometimes result in unexpected transformations when users are allowed to freely navigate through the virtual environment, and that constrained interactions allow for more accurate manipulations [34]. Although mid-air interactions show promising results, the accuracy of human spatial interactions is limited. Moreover, the limited dexterity of mid-air hand gestures, aggravated by lack of precision from tracking systems and low-definition of current HMDs, constrain precise manipulations. This precision is of extreme importance when creating or assembling engineering models or architectural mock-ups, for instance, and these limits end up limiting the potential application of VR setups. Even if a large number of methods have been proposed in the literature, most demo systems do not provide easy to use interfaces and the possibility of reaching a high level of precision. This is also due to more intrinsic issues of the tracking task such as the inaccuracy in tracking, occlusions, difficulty in the segmentation of different gestural primitives, and other non-trivial problems. There is surely a lot of room for improvements, not necessarily related to a better tracking of body landmarks but also to a smart global processing of 3D data acquired through trackers. Several methods have been recently presented for characterizing salient points and for global shape description, with invariance properties and robustness against various kinds of perturbation. Methods with similar characteristics but adapted to the different kind of shape data provided by tracking devices could in principle be applied for the solution of open issues in gestural interaction. An example of smart application of simple geometric processing to realize effective interaction comes from 2D touchscreen interaction, where many gesture recognition applications are not based on complex time series analysis, but on the reduction of the problem to a simple template matching of 2D shapes. The popular 1-dollar recognizer [67] and similar derived methods (also proposed for 3D interaction [37]) are a clear demonstration of the usefulness of this simplification. This may seem to indicate that in this case the dynamic information may be neglected without losing the meaningful part of the signal. The research project will cover some of many aspects related to these topics with the possible development of design guidelines and the implementation of innovative interfaces for the interaction with objects in a virtual environment. The following list summarizes the main goals of the project:

- Derive useful and necessary guidelines to design novel interfaces aimed to achieve a natural interaction with different VR systems.
- Design and implement interactions paradigms and interface modules to be tested with users by creating a set of prototypes to use with appropriate hardware setup for the task.
- Test the validated paradigms and modules in real-world applications (e.g. 3D

scene design, virtual museum) with specific user categories.

- Investigate issues related to gesture-based interfaces and propose practical solutions such as novel gesture recognizer algorithms.

## 1.3 Thesis Statement and Hypothesis

The use of VR technologies is currently limited by the lack of interaction techniques that can feel appealing and practical for the users. The thesis statement we propose is the following:

> *It is possible to develop techniques that will enhance the usability of VR applications. This can be achieved by deriving guidelines to design better interaction techniques with items in VR scenes (i.e. manipulation techniques) and by exploiting known AI topics such as machine learning techniques to develop additional tools for interaction such as gestural commands.*

The challenge is anything but trivial considering the amount of research work already present in literature and the fact that there hasn't been any critical breakthrough leading to a comprehensive solution for the mid-air manipulation issues. This, among other things, also suggest that instead of trying to tackle all the issues all at once, a more efficient approach is to identify specific interaction scenarios and design appropriate solutions for their issues. Using this approach we decided to start this research work with the formulation of a series of hypothesis presented in the following:

**Hypothesis $\mathbf{H_1}$:** *The most critical transformation to perform in object manipulation is the rotation. The overall usability of a manipulation technique is highly* *dependent on the efficiency of its rotation technique in terms of learnability and practicality.*

We think the rotation is the hardest part to design properly because of the limitations of the available hand tracking solutions. In fact, while trackers seem to work rather well in tracking the 3 DOF related to the hand position in space, the same cannot be said for hand orientation tracking accuracy. The anatomical complexity of the hand plus the self-occlusion issues that often occur (e.g. one hand hiding the other from the tracker) make this job quite hard. These considerations lead us to the next hypothesis.

**Hypothesis $H_2$:** *Mapping the 3 DOF related to the orientation of an object to one or more positional DOF of the user's hand can substantially improve a manipulation technique usability.*

If the hypothesis $H_1$ holds true, verifying this hypothesis becomes critical in order to derive a solid guideline for the design manipulation techniques for hand tracker setups. This can be achieved mainly in two ways in order to avoid interference with other manipulation actions such as the translation: the first would be to design a two-handed technique and assign the 3 object orientation DOF to one or more of the positional DOF of the hand not involved in the translation action, the second would be to separate DOF by allowing only one action to be performed at the time (e.g. only translation, only rotation, etc.).

**Hypothesis $H_3$:** *Single-handed manipulation are preferred by users when they achieve a certain usability degree.*

The idea behind this hypothesis is rather straightforward: we think that users are likely to prefer to use just one hand to perform simple manipulation actions. What we want to verify whether this is the case and what, eventually, is the trade-off with the overall usability. In fact, if both this hypothesis and hypothesis $H_2$ hold true, the correct idea would be to design single-handed techniques featuring DOF separation as described before.

**Hypothesis $H_4$:** *Gesture recognition can be performed with relatively cheap computational cost and good accuracy results.*

A gesture recognition system could potentially be very useful in VR application interfaces when standard input devices such as controllers are not available or if, for some reason, a deviceless manipulation technique is the design choice. Gestural commands could be used for the activation of additional functionalities.

**Hypothesis $H_5$:** *A VR application implementing based on a manipulation technique can benefit from a set of gesture-based commands to enhance the overall usability and UX.*

The implementation of a successful manipulation technique alone is probably insufficient as the only mean of interaction in a more complex environment of a complete VR application. A gesture-based set of commands enabled by a gesture recognition algorithm together with a single-handed manipulation technique could provide a successful and complete interaction paradigm for VR applications.

## 1.4 Contributions

The research of this thesis resulted in different scientific contributions in the field of Human-Computer Interaction, in particular in interaction with IVEs. The summary of these contributions is listed in the following:

- The development of two novel manipulation techniques based on the insights provided by our tests and the results from the literature: the Knob and the Smart Pin. Both techniques aimed to overcome some of the common issues of previous methods and were designed to be used with just one hand (or a controller device) and featuring DOFs separation. While the Knob exploited the namesake metaphor to test a more direct approach the Smart Pin technique relies on a 3D widget that resulted in a good trade-off between naturalness and performance.

- The classification of a large number of research works on object manipulation including techniques using standard input devices, such as keyboards and mice and touchscreens, and more state of the art technologies such as hand trackers for mid-air techniques. This resulted in a survey that features a new taxonomy to classify all these methods and some design guidelines derived from the analysis of them.

- The development of an algorithm for gesture recognition from 3D trajectories (3cent) based on the dollar algorithm for the 2D case. The 3cent proved that gesture recognition from data acquired with hand trackers can be very accurate while being relatively cheap in terms of implementation and computational load. These results open the way to the design of gestural interfaces that could be integrated with manipulation techniques.

## 1.5 Publications

1. Fabio M Caputo, Marco Emporio, and Andrea Giachetti. The smart pin: an effective tool for object manipulation in immersive virtual reality environments. *Computers & Graphics*, 2018

2. Fabio M Caputo, Pietro Prebianca, Alessandro Carcangiu, Lucio D Spano, and Andrea Giachetti. Comparing 3d trajectories for simple mid-air gesture recognition. *Computers & Graphics*, 73:17–25, 2018

3. D Mendes, FM Caputo, A Giachetti, A Ferreira, and J Jorge. A survey on 3d virtual object manipulation: From the desktop to immersive virtual environments. In *Computer Graphics Forum*. Wiley Online Library, 2018

4. Fabio M Caputo, Daniel Mendes, Alessia Bonetti, Giacomo Saletti, and Andrea Giachetti. Smart choices for deviceless and device-based manipulation in immersive virtual reality. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 1–2. IEEE, 2018

5. Fabio Marco Caputo, Marco Emporio, and Andrea Giachetti. The smart pin: a novel object manipulation technique for immersive virtual environments. In *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology*, page 89. ACM, 2017

6. Fabio Marco Caputo, Pietro Prebianca, Alessandro Carcangiu, Lucio D Spano, and Andrea Giachetti. A 3 cent recognizer: Simple and effective retrieval and classification of mid-air gestures from single 3d traces. *Smart Tools and Apps for Graphics. Eurographics Association*, 2017

7. FM Caputo, M Emporio, and A Giachetti. Single-handed vs. two handed manipulation in virtual reality: A novel metaphor and experimental comparisons. *Smart Tools and Apps for Graphics. Eurographics Association*, 2017

8. Fabio Marco Caputo, Irina Mihaela Ciortan, Davide Corsi, Marco De Stefani, and Andrea Giachetti. Gestural interaction and navigation techniques for virtual museum experiences. In *AVI* CH*, pages 32–35, 2016

9. A Giachetti, FM Caputo, A Carcangiu, R Scateni, and LD Spano. Shape retrieval and 3d gestural interaction: position paper. In *Proceedings of the Eurographics 2016 Workshop on 3D Object Retrieval*, pages 1–4. Eurographics Association, 2016

10. Fabio Marco Caputo and Andrea Giachetti. Evaluation of basic object manipulation modes for low-cost immersive virtual reality. In *Proceedings of the 11th Biannual Conference on Italian SIGCHI Chapter*, pages 74–77. ACM, 2015

## 1.6 Dissertation Outline

The rest of this dissertation is organized as follows: In Chapter 2 we present the state of the art and our first approach on manipulation techniques with a work of comparison and evaluation of simple manipulation methods in IVEs. Chapter 3 follows, with a work on the design of our first novel manipulation technique. In Chapter 4 we present our work on a second novel technique with an additional study on input devices, at the end of the chapter. In particular, we present the results of a study analyzing the influence of combinations between input device choices and manipulation techniques including the one presented at the beginning of the chapter. In Chapter 5 we present our most recent development of the gesture recognition algorithm 3cent and the test results. Due to the two-folded nature of this research work (i.e. object manipulation and gesture recognition) the background theory and works are discussed in the relative chapters. In Chapter 6 we go over the summary of the research and discuss the results with respect to the initial hypothesis stated in relative section 1.3.

# Chapter 2

# Object Manipulation in IVEs

## 2.1 Background Theory and Related Work

Interaction in IVEs has a large number of forms that can range from exploring the environment, retrieve informations, access to several application functionalities by interacting with widgets, etc.

One of the most challenging kind of interactions is the manipulation of virtual objects. Many useful and entertaining VR applications, heavily rely on the ability to grab, move, rotate and scale objects in the field of view. However, as shown in [38] "... development of interaction mechanisms for manipulation remains one of the greatest challenges for VR". In fact, while successful metaphors to control 3D manipulation using 2D desktop interfaces are well established and employed in many 3D editing environments, efficient and flexible solutions well suited for immersive VR are still missing, limiting the potential use of this emerging technology.

Many recent VR setups and applications rely on handheld devices to interact with the scene, e.g. Oculus Touch or HTC Vive. This choice is suitable for personal use and gaming, but is limiting the naturalism of the experience and not appropriate for many use cases (e.g. public displays, work environments, etc). Furthermore, mapping grabbing actions to device buttons and object orientation to the device's one does not solve, for example, issues related to continuous rotation control due to intrinsic limits of hand articulation.

When handheld devices are not an option, it is possible to rely on hand tracking systems and related APIs (Leap Motion, Kinect, RealSense) providing real-time hand and finger position mapping and some gesture recognition capabilities. These trackers, even if not always stable and accurate due to occlusions and noise, can be directly used to build gesture-based interfaces allowing direct object manipulation, similarly to the finger-based manipulation on touchscreens.

A significant hurdle to usability of natural techniques, such as the Simple Virtual Hand [39], is the reliability of hand orientation tracking required to map the rotation of the hand to that of the desired transformation. In order to avoid this problem, a possible solution is to design a manipulation technique that relies only on easy to track positional data, like the 3D coordinates of the hands' centroids. A very well known method of this kind is the so-called "Handlebar", adopted in several practical applications. In the Handlebar metaphor [59], the relative position of the two hands is exploited to derive the object transform mapping, allowing a reasonable accuracy even in case of rough tracking without accurate finger detection. However, this method requires the use of both hands, denying the availability of a free hand for any possible action to be performed in parallel. Furthermore, it can require significant hand displacements that may result in the so-called gorilla arm issue if performed for a long duration[38].

The possibility of manipulating objects with a single hand, as usual on 2D desktop interfaces, would certainly extend the potential applications of immersive VR based interfaces and visualization tools. In [44] a technique based on this concept, MAiOR, is presented. In this technique translations, rotations and scaling actions are controlled using only the information on user hand position tracked using an handheld controller (i.e. handles from the HTC Vive HMD).

Mid-air manipulation has been recently investigated in several papers proposing different kinds of solutions and trying to evaluate their usability in specific tasks for both standard or stereoscopic displays and HMD based immersive environments.

This kind of interaction can be based on different sorts of devices, resulting in differences in tracking accuracy, access to buttons, force feedback, etc., and presenting diverse specific problems. For this reason some interesting papers compare manipulation solutions based on different hardware setups, trying to derive useful hints for application design.

Mendes et al. [43] compared different manipulation methods using a 2D touch based approach and mid-air bi-manual methods with two variants: 6DOF Hand, which uses the dominant hand to grab, move and rotate objects, and the distance between both hands for scale and 3DOF Hand, where the dominant hand only moves the object, while scale and rotation are given by the non-dominant hand.

Vuibert et al. [65] also made a comparison of different methods including physically constrained devices, a wand-like device, tangible object replica and finger-based interaction. Another wide range study on direct mid-air manipulation is presented in [20]. Here manipulation with 6 degrees of freedom flystick, and finger-tracking systems are analyzed and many aspects of the problems are addressed, like grasp/release heuristics, a finite state machine for object manipulation, handling of complex hand/finger interactions with objects.

Results of these multi-modal comparisons show that users often like mid-air interactions more than other methods, finding the accuracy in manipulation sufficiently good for many proposed tasks.

This fact, together with application-driven constraints, suggested that we should focus our interest on mid-air deviceless interaction, using low-cost trackers to track hands and fingers with no need of wearing markers or using handheld devices and on co-located hands and objects. This kind of interaction is extremely important for emerging, partially or fully, immersive applications like those based on Head Mounted Displays (HMD), autostereoscopic monitors, or even light field monitors [1] where physical input systems or other input channels are not available.

One of the main problems tackled by mid-air interaction research is the search for more "natural" interfaces that can greatly enhance the performance of the system as well as the user experience [9]. This is a very challenging task, as it requires the study of optimal metaphors and implementations that should consider: the lack of buttons, object selection, the detection of the start and end of gestures as well as their decomposition, the difficulties and fatigue of mid-air gesturing, and the accurate mapping of hand motion to object motion with optimal constraints reducing the effect of sensor errors.

The simple attempt to directly map 5 or 6 DOF motion on virtual objects may, in fact, be "natural", but it is not able to create an effective interaction [3]. Furthermore, using low-cost trackers like the Leap Motion[4], but also due to the intrinsic difficulty in controlling hands in mid-air movements, it is necessary to cope with the relevant limits in selection accuracy in a setup with a single tracking sensor and the impossibility of directly tracking large rotations [47].

Another relevant problem for the usability of effective mid-air virtual object manipulation is the lack of haptic feedback, that should ideally be replaced by different feedback mechanisms [45]. The solutions proposed in the literature for mid-air manipulation can be classified into two categories, bi-manual and uni-manual techniques.

In the bi-manual category, the Handlebar metaphor [59], is probably the most intuitive and popular solution for manipulation in VR. The typical implementation of this method uses hand gestures for grabbing and estimates the object transform based on the hands' position, exploiting resemblance with real-world actions like rotating and stretching a bar. This particular mechanic allows the user to handle simultaneously 7 DOF. Several research works [5, 31] and also some commercial applications exploited it with slight variations. Another way to exploit the bi-manual interaction is to split the DOF control between the two hands. Caputo et al. [16] propose the 2HR technique, mapping the object rotation onto the user's non-dominant hand rotation. User tests demonstrated, however, the poor usability of the method. In the same work, the Finger-Point technique features a rotation action, also performed with the non-dominant hand, by mapping the index movement on a 2D plane into 3D rotations applied on the object. This indirect rotation showed promising results suggesting that indirect approaches for the rotation action are viable and even preferred by users. Following this kind of indirect approach, there are also bi-manual techniques relying on gestures to indirectly manipulate a selected object through the use of widgets or tools. Examples can be found in [61, 7]. Manipulation modes, transitions and accurate control are not, however, addressed. In the uni-manual category, it is possible to find a variety of solutions in the literature. The Simple Virtual Hand [39] is the most direct one. This technique enables object selection through a grasping gesture. After that, position and orientation changes of the object are directly mapped onto the hand's ones. This technique is quite intuitive, but it is not suitable whenever a robust hand tracking is not available, as in the case of a single low-cost device, where hand orientation tracking can be unreliable [16]. An alternative kind of approach is to design indirect techniques by either the use of metaphors or widgets. Kim and Park [35] proposed a Virtual Handle with a grabbing metaphor, based on a bounding sphere around the selected object and the creation of a local reference frame for the following transformations. The Crank Handle metaphor for rotation action, presented in [6], demonstrates the feasibility of single-hand manipulation, by exploiting DOF separation. The user must, however, activate the rotation mode, after selecting the translation mode, and has to split the transform into multiple rotations around primary axes.

Bossavit et al. [6] analyzed several critical choices for mid-air manipulation design. From the proposed analysis and the related experimental tests, they also derived some useful guidelines that were particularly inspiring for our work, like the potential use of single-hand methods and the benefits of separating translation and rotation.

Despite the increasing research efforts and the interesting solutions proposed, however, no widely accepted metaphors for deviceless and mid-air interaction emerged, and many open issues remain unsolved, e.g. accurate releasing of position control in uni-manual translation, the necessity of learning semaphoric gestures, potential unwanted activation of manipulation modes. Furthermore, selected hand gestures should, in theory, try to be consistent with ergonomic principles and biomechanical constraints, trying to minimize stress and fatigue. Analyses related to these factors in gestural interaction and manipulation have been presented in [48, 10].

## 2.2 First Approach and Evaluation

There are different ways to interact with virtual objects present in a VR scene. The type of interaction is determined by the kind of task the user is allowed to perform and ultimately by the purpose of the application featuring such interactions. One specific kind of interaction is the object manipulation. In this case, the user is allowed more or less directly, depending on the metaphor proposed by the implemented technique, to select an object in the scene and perform one or more different transformations (i.e. translation rather than rotations or scaling operations). However, this kind of interaction has many limitations. First, the user's arm has a limited range of interaction due to real-world constraints (e.g. it's safe to assume arms are always anchored to user's bodies) and sensors detection volume in case of deviceless setups. This limit can be worked around with the use of a navigation technique. Such a technique is required as it also allows different visual perspectives, but it should not be required depending on the application. Manipulation of large objects is also an issue as they tend to obscure the users view during positioning tasks unless additional workarounds are implemented to give the user a way to change the point of a view for a more convenient perspective. [8]

With our first work we tried to verify hypothesis $H_1$. In order to do so we designed an experiment featuring four manipulation techniques to be tested with users performing a simple docking task. The idea was to test different strategies for the rotation action like assigning it to the non-dominant user's hand or separating it from the translation action by assigning different activation commands.

### 2.2.1 List of Tested Techniques

**1. One-handed direct manipulation (1HR)** This can be considered a straightforward implementation of the Simple Virtual Hand method [39]. The interaction is directly mapped from the real world hand to virtual one. This fact, however, does not necessarily guarantee optimal execution performances. Critical limits and constraints are inevitably given by the type, setup, number and quality of the tracking devices (limited interaction range, low sensor resolution, occlusions, wrong signal interpretation). These problems clearly affect also other paradigms, but possibly with different impact on usability. In our implementation grab is obtained when a certain grab strength (a value proportional to fingers' curl measured by the device APIs) threshold is reached and the hand's palm position is within range of the object. Translation is then achieved by computing the translation vector from the previous tracking frame to the present one. The same mechanism is applied to the rotation using Euler angles giving hand orientation instead.

**2. One-handed translation and second hand rotation (2HR)** In this approach grab and translation are performed as before, but a bimanual rotation mechanism is introduced for the rotation around an arbitrary axis. The idea is to use the free hand, i.e. the one which is not holding the target object, as reference for rotation, meaning that the object will rotate by the amount of variation in orientation of the free hand as shown in **Figure 2.1**. The holding hand can focus on keeping a convenient position in order to ease holding-gesture recognition from the device. To quickly implement this method we changed the hand's reference for rotation while the object is selected.

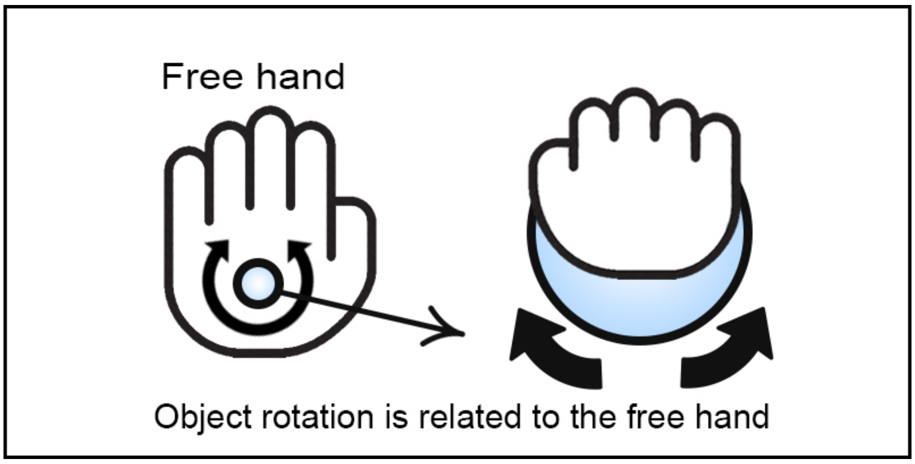


Figure 2.1: One-handed translation and second hand rotation technique: in this technique the dominant hand can control the translation of the object by a simple grab and drag action while the non-dominant hand has control over the object rotation.

**3. One-handed translation and Finger-Point rotation (FR)** This metaphor differs from the previous ones for the rotation mechanism only and is, to our knowledge, original. The idea is to implement in the 3D gestural interaction a 2D constrained mapped rotation similar to the one used in the Virtual Trackball metaphor for mouse interaction. The user is able to rotate the object by pointing one finger. We exploited the automatic recognition of the "the pointed-finger" gesture provided by the Leap Motion SDK: when the second hand is detected in this position, finger tip position is tracked and projected on a virtual vertical plane in front of the user. The wideness and direction of the fingertip movement on the plane determine the rotation angle. By moving along the two axes of the plane, two of the rotation axes are actually affected as in **Figure 2.2**. Although a DOF for rotation is indeed lost in this paradigm, the user can in this case benefit from the constraints and from the familiarity of the gesture used in 2D interfaces to deal with the most critical interaction with a virtual object, i.e. the rotation.

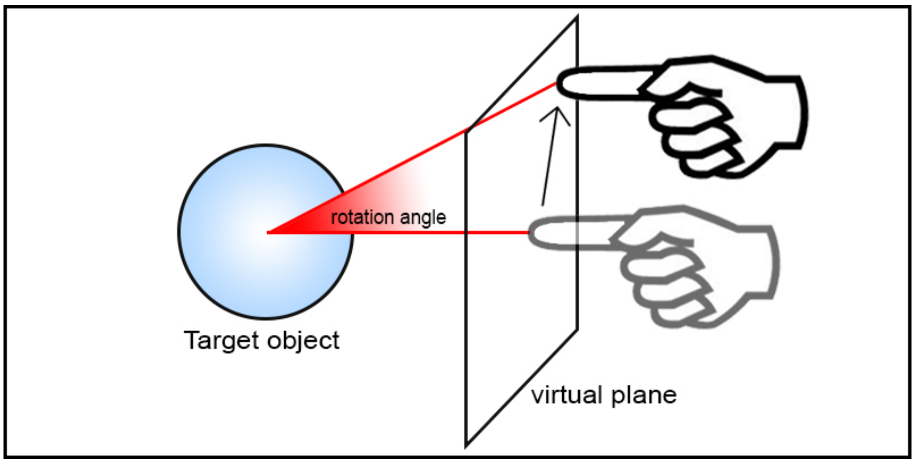


Figure 2.2: Fingerpoint Rotation technique: the user can rotate the object by sliding the index finger across a virtual plane and defining a direction and an angle.

**4. Two-Handed translation and handlebar rotation (HB)** The handlebar paradigm has been proposed and tested for virtual object manipulation in 3D in different works [35, 5]. With this paradigm, the user controls the selected object by holding two ends of a virtual rigid bar (i.e. the handlebar). Three DOF for the translation in the 3D space are achieved by mapping the translation of the handlebar directly to the selected object. As for the rotation, it is achieved for 2 of the 3 rotational DOF by leaning the handlebar on a horizontal or vertical plane. In our approach to the handlebar metaphor, we only have a small variation from the one proposed in [60] which is the way we handle the third DOF bound to the rotation about the x-axis. The approach proposed by Song will feature a “crank” metaphor to enable such rotation with either one of the user's hands (**Figure 2.3 a**). In our approach **2HR** we use direct rotation about the x-axis, enabled directly by the use of both or single hands exploiting the use of a motorcycle throttle metaphor (**Figure 2.3b**). To make the global interaction (grabbing, translation and rotation) coherent, in this case, we used a bi-manual grabbing and translation. Selection is still based on a grab strength threshold, measured for both hands at the same time. Instead of requiring the hands to be in range of the object, a constraint related to the distance between the hands, enforcing the "handlebar" simulation has to be satisfied in order to achieve selection.

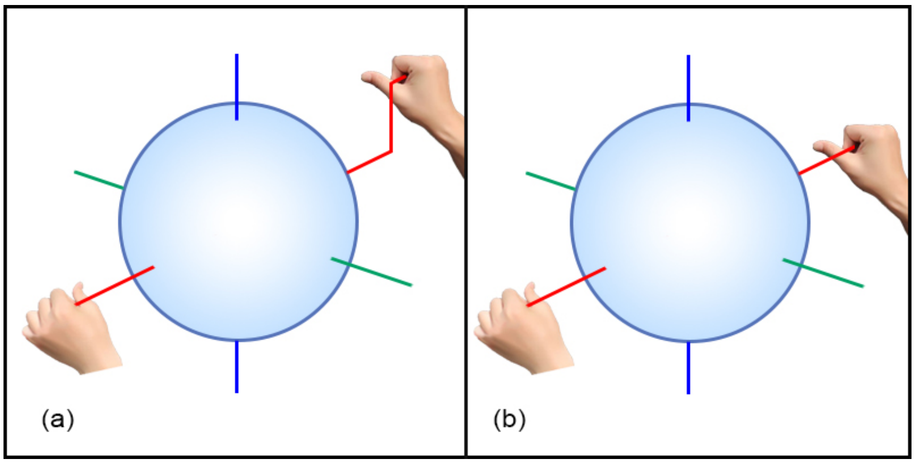


Figure 2.3: Handlebar technique: the user can manipulate the object by grabbing a imaginary handlebar passing through the selected object.

### 2.2.2 Experiment Setup

To test these interaction modes, we implemented from scratch a simple modular VR platform that we called Manipulation Interaction Test Platform (MITP). In this platform, the hand tracking and rendering processes are developed from the interaction management so that it was possible to create an independent interface for paradigm behaviour definition. The greatest advantage of such architecture is the possibility to modify, add or remove the interaction modes without really touching the data process for both tracking and rendering. The platform supports the loading of multiple modes, allowing the construction of a paradigms' library.

In our test setup (Figure 2.4) hands tracking is realized with a Leap Motion controller placed on the desk in front of the monitor, creating a fixed interaction space that can be reached by the user hands and where the scene (object, virtual hands, target) was placed in the fixed desktop reference system. Immersive 3D rendering of the interaction region from the user eyes viewpoints is performed on an Oculus DK2 headset.

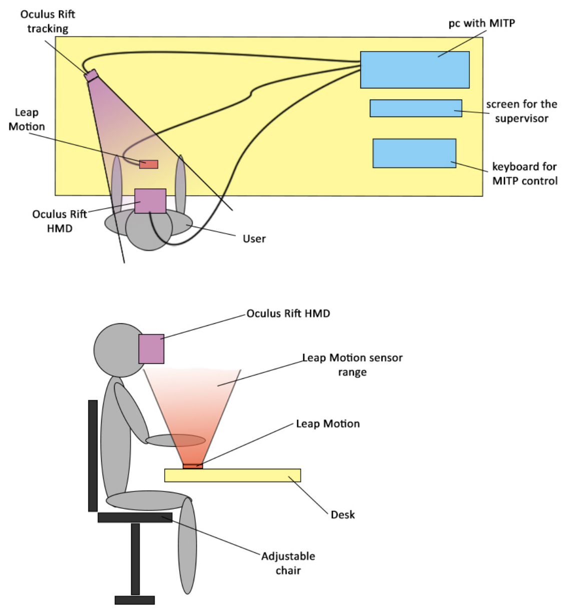


Figure 2.4: Experiment setup for the designed user task.

In the practical virtual task, the object to be grabbed and moved was a sphere with a black dot on the surface and the target position was identified by a yellow semi-transparent sphere (Figure 2.5). Upon reaching the target position the target object was locked in place and a visual feedback regarding the accomplished objective was returned by changing the color of the target position sphere from yellow to green. The user had then to adjust the object pose to align the black dot on it with a red dot on the target sphere using the active interaction technique. Upon reaching the correct pose a visual feedback informing about the completed objective and task was returned by changing the color of the target position sphere from green to blue. This choice was done to separately evaluate grabbing/translation and rotation modes. Future works will evaluate the effect of gesture interactions and modality changes.

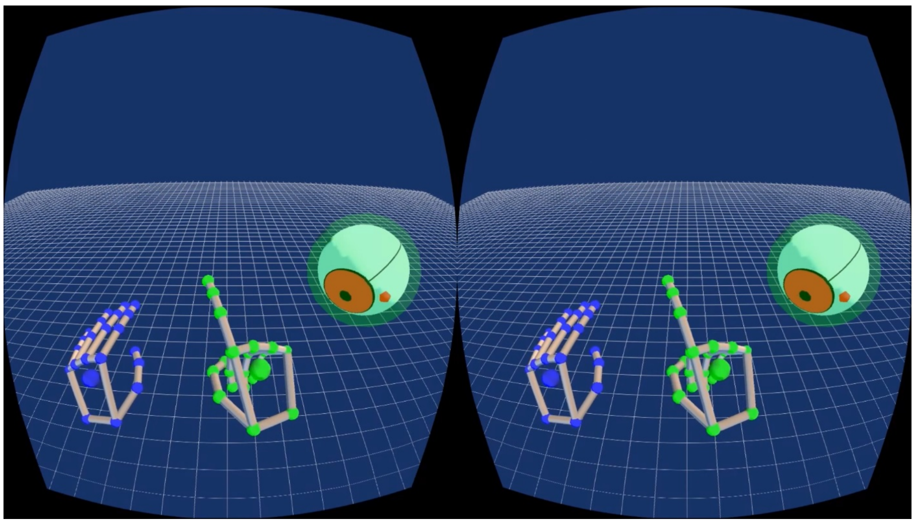


Figure 2.5: Snapshot of the stereo pair sent to the headset display during a Finger-Point rotation gesture.

The interaction modes implemented have been finally tested on a group of 24 people (aged from 22 to 39 years) with no previous experience with VR environments. We used both right-hand and left-hand users to test our "symmetrical" implementation of techniques. Results confirmed that no correlation emerged between the preferred hand and performances. Each user, after training sessions where no specific goal was set and the user was allowed to explore the mechanism each different interaction mode, had to complete the previously described task with all the 4 different interaction techniques. To avoid biases, the execution order of the techniques was randomized so that each subject had to use them in a different order. The platform automatically recorded all the relevant variables of the interaction sessions: time necessary to complete grabbing and translation (T1), the time required by the rotation (T2), total execution time (Tot), number of grasping events (NGrab). After the end of the test, each subject had to compile a questionnaire to provide additional data about her/his personal profile and user experience. Questions covered also information about the preferred method, regardless of performance, with respect to perceived stress and a subjective evaluation of the visual feedback provided.

### 2.2.3 Evaluation Results

Figure 2.6 shows the average values of the automatically estimated task completion performances with the four interaction modes. A general improvement in performance is visible from direct-manipulation-type of interaction towards more abstract-metaphors ones. The average time for the translation step T1 is approximately the same not only for the modes using the same single hand grab and translation mode (1HR, 2HR, FR) the average (T1), but also for the two-handed grab and translation mode used in HB. Differences are not statistically significant, meaning that bimanual grab and translate does not make the subtask more complex.

Relevant differences are instead reported for the rotation time T2. In particular, the only non-significant difference according to a repeated-measures ANOVA is the one between 1HR and 2HR techniques with the same situation between the difference in the tot time.

The problem of direct one-handed rotation seems strongly correlated with the high number of grabbing actions recorded with that mode (Ngrabs) at least in the case of 1HR which average has a significant difference with all the other techniques. This is clearly due not only to limitations of the tracking device, with non-trivial problems related also to the accuracy of the grabbing and release procedure, but also to the limits of arm mobility creating the necessity of mapping a relevant rotation over multiple gestures. This fact is confirmed by the poor performance of 2HR, that should limit the interference between grabbing, release and rotation procedure, but has the same problems related to the limited hand mobility. This is probably the reason why more indirect interaction metaphors seem to provide the best performance for rotation. FR, that maps rotation on a 2D translation not directly corresponding to the 3D object, provided the best results in every data category. HB gave better results than the techniques applying directly the rotation on the locally "touched" object, however, the coordination of the two hands may limit the accuracy of this method. The indirect rotation is easier to be controlled as the applied rotation angle is given by a translation vector of the finger point once the user activates the rotation mode by pointing the finger. Once the rotation is activated it is controlled by a translation with no need for bimanual coordination and the potential mapping of a large rotation in a unique gesture.

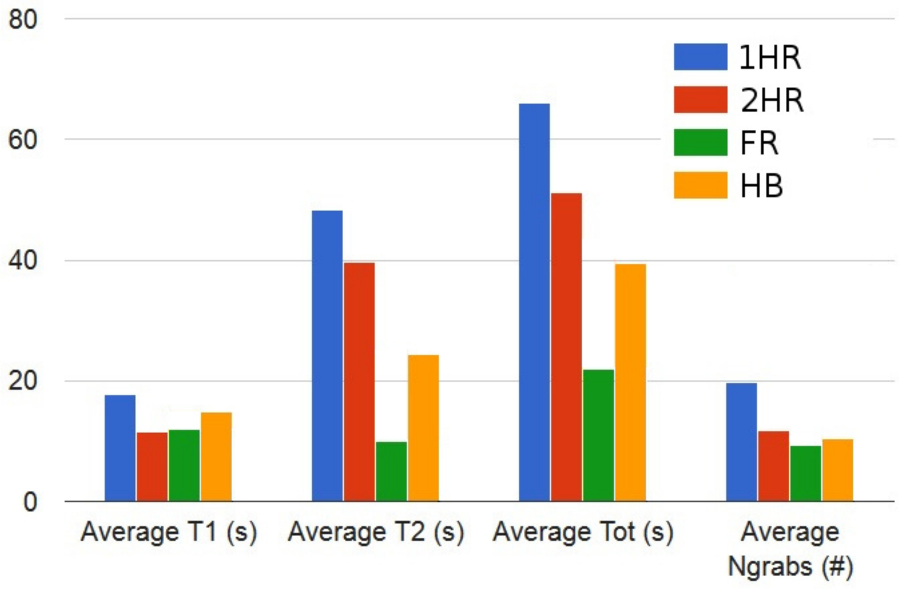


Figure 2.6: Average values of automatically collected performance measurements for the test group

Answers given to the questionnaire (the main results are summarized in Figure 2.7) mostly confirm the outcomes of automatic measurements, but also give additional insights. For example, it is possible to see that the perceived learnability is not too different for the various methods, but actually is a bit lower for HB, and this may account for the lower performances with respect to FR. Actually, even if users have a strong preference for FR method for rotation completion, the overall usability of HB and FR were scored similarly. This may be interpreted by considering that, despite the better performances obtained, the FR interaction may be felt as less satisfactory as less coherent, mixing different paradigms for translation (direct, 3D) and rotation (indirect, 2D).

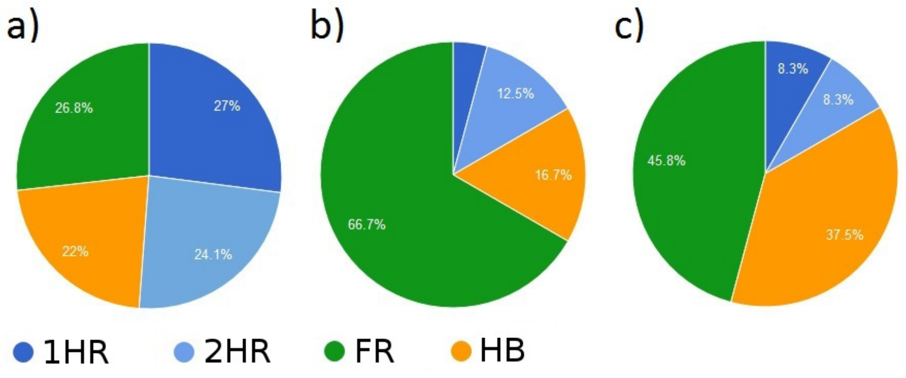


Figure 2.7: Preferred interaction modes indicated in the questionnaire with respect to to a) learnability, b) ease of rotation task completion, c) overall usability

.

### 2.2.4 Evaluation Discussion

Our study was focused on the individual execution of simple manipulation tasks with a low cost VR setup (OculusVR+Leap Motion).

The first is that, with this setup, direct manipulation methods are not the most effective considering that, according to our results, more indirect methods (i.e. Handlebar, Finger-Point) provided better results in terms of both performances and UX. Experimental results show that this is due to the hard realization of an effective single-hand object rotation. In fact, the Handlebar technique gives better results, but even better results can be obtained with our method mapping constrained rotation on translation similarly to what is done in the virtual trackball metaphor[21, 57]. These results seem to be consistent with our $H_1$ considering the fact that significant differences in performance showed in the rotation task and the questionnaire further confirm this showing a relevant correlation between the two most preferred techniques in both the questions about the rotation task and the overall usability preference.

# Chapter 3

# The Knob Metaphor

Considering the results of our first evaluation we designed a tentative technique that would address some of the problems highlighted by our previous study. Our goal was to create as simple as possible solutions to allow an intuitive and easy manipulation of objects, especially when it comes to critical actions such as the rotation. Following both our findings and those provided by the literature, we aimed at:

- performing the manipulation control with a single hand
- separating translation from rotation and constrain rotation axes
- being robust against the limits of tracking and gesture segmentation accuracy provided by the chosen device/API, making difficult to control object release in the desired position
- finding intuitive metaphors for rotation and scaling
- avoiding as much as possible the necessity of non-intuitive gestures or use of different modalities to switch interaction modes

The scheme of the designed interaction solution is represented in Figure 3.1. The system starts in idle state when hands are tracked and their position is displayed in the scene but no interaction with the objects is activated. However, if the hand falls in the region of interaction with an object, specific gestures can trigger the switch to different modes. Furthermore, if the hand is in the right position to start a rotation gesture on the object, a slightly different state is enabled, giving also visual feedback about the enabled rotation axis. In this state a knob rotation gesture starts the rotation as described in the following subsections, locking a rotation direction. Translation and scaling can be activated as well, when the hand is in the interaction region, with unambiguous gestures.

In this way, basic manipulation modes are completely decoupled,but can be activated easily with a single hand gesture. In the following we described the proposed solutions to manage the specific manipulation modes.

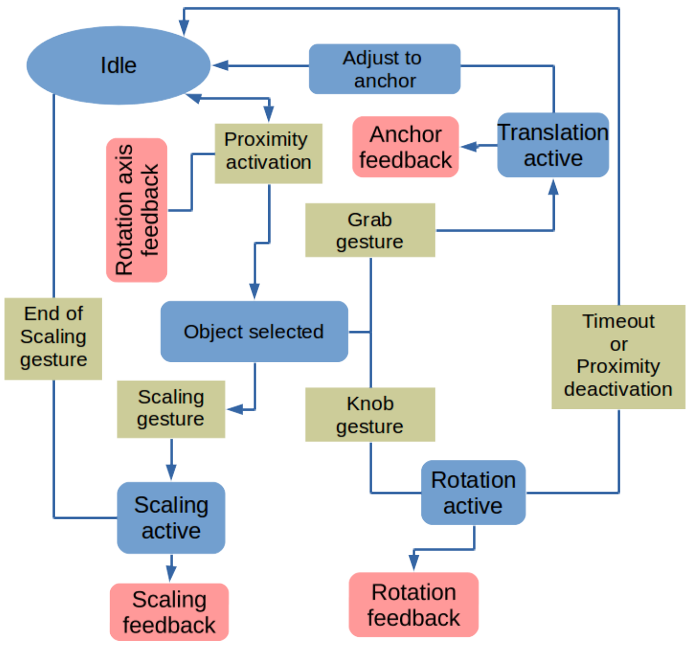


Figure 3.1: Block diagram of the proposed interaction method. Blue rounded corner rectangles represent system states, while green rectangles indicate hand gestures triggering transitions.

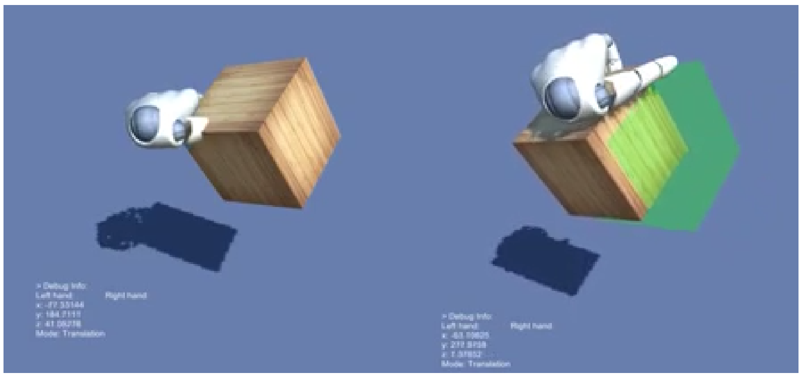
## 3.1 Translation: Release Anchor and Visual Feedback

If we separate translation from rotation and scaling, the natural solution to grab objects and translate them is to check for hand closure and then put the system in translation mode, anchoring the displacement of the virtual object to the displacement of the hand (palm). This method is intuitive and also clearly outperformed other methods in many experiments, e.g. in docking tasks [6, 16]. The only problem with this method, not fully addressed in the literature, is how to deal with object release in uni-manual, deviceless interaction.

The grab detection is currently realized with a simple thresholding $t = 0.5$ on the "grab strength" value, accessible through the Leapmotion API, when the hand is superimposed to the object. This choice provides reasonable accuracy, but it is not free from missed and false detections, so that further investigations will surely be performed to improve the usability of the system.

The biggest issue with this grab/release mechanism is that, if we recognize a simple finger gesture to exit from translation mode (e.g. opposite thresholding of the strength value), we could have a large displacement from the palm position before the release gesture and the tracked. A drop position detector based on trajectory speed analysis can reduce this problem by giving a batter localization, however, it could create other problems: the visual feedback is not synchronized with the internal object position status and this is particularly bad when the user changes idea and, after stopping the translation, he does not release the grabbing and starts again translating the object.

For this reason, we not only designed a translation stop/restart detector based on hand speed but also a specific feedback to show, in this case, the detected release position (visual anchor) and the current object with standard rendering.



```
> Debug Info:
Left hand:          Right hand
x: -77.33144
y: 184.7111
z: 41.08278
Mode: Translation
```

```
> Debug Info:
Left hand:          Right hand
x: -63.10625
y: 277.8738
z: 1.37832
Mode: Translation
```

Figure 3.2: The translation action can be started with a simple grab gesture (left side). The release anchor location is showed through a visual feedback (right side).

The release anchor detection and update is realized as follows: once the translation mode has been initiated by the user, the system will constantly try to detect and guess the intended drop point. If the velocity is above a small threshold $v_t$, the release anchor is constantly moved constrained to the detected palm position at time $t - dt$ where $dt$ is a speed dependent value avoiding unwanted displacements due to the release gesture. When the speed is under the threshold $v_t$ the detected drop point is fixed in the position of the palm at the threshold crossing time. In this case, a semi-transparent and monochromatic replica of the object is displayed (visual anchor) as showed in Figure 3.2, so that the user can see where the real one would snap to in space. This way the user can decide if they want to confirm the drop position by releasing there the object, by simply interrupting the grab gesture, or keep moving the object. If the object is not released, the anchor is kept for a fixed time, then the tracking is restored. If the velocity remains slow, the visual feedback of the release anchor position is maintained for some time also to allow a more accurate positioning in a feedback loop.

## 3.2 Rotation: the Knob

To perform rotation we introduced a novel metaphor that we called ”knob” metaphor. The most typical uni-manual rotation gesture, in fact, is the action we do when turning a volume control. A similar single-handed gesture emerged also as the most intuitive rotation gesture in the elicitation study by Aigner et al. [2], with a clear preference over bi-manual gestures or non-iconic ones.

The idea is to map this gesture on the rotation of the object (selected by proximity when the gesture is started), automatically selecting a constrained rotation axis on the basis of the palm orientation. To make the system simpler to understand we force rotation to be around the z-axis or x-axis. As in the real world interaction, users can then perform multiple rotations of the knob in sequence to obtain large rotations. To avoid unwanted forward and back rotations, the system is locked in a specific rotation direction after the gesture starts and exits from the lock only after a fixed time step passes without further rotation.

This method allows the realization of arbitrary rotations with sequences of x and z rotations. Despite the fact that in specific cases in which the user would like to rotate exactly about the y-axis, it can take a fairly long sequence of x/z rotations we believe the fluency provided by this rotation mechanism is enough to compensate for the extra number of steps needed to obtain the desired pose.

### 3.2.1 Rotation Feedback

A further help for the understanding of the interaction mechanism without training is given by the visual feedback of potential rotation (Figure 4.3(b)). This is designed as follows: when the system is in the idle state and the user’s hand enters the object’s selection range with the gesture associated with the knob rotation method, a double

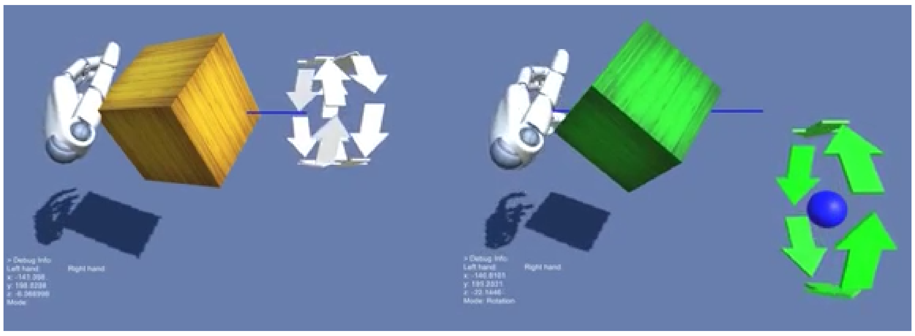


Figure 3.3: The visual feedback for rotation shows the possible directions before the knob gesture is applied (left side). A visual feedback is provided showing the current locked direction of rotation (right side).

set of arrows rotating around the selected axis is displayed. This notifies the user that both directions for rotation are available. Upon rotating the hand, one of the two directions is confirmed and the system will only keep the corresponding set of arrows and discard the other one to make clear which direction the user has locked onto.

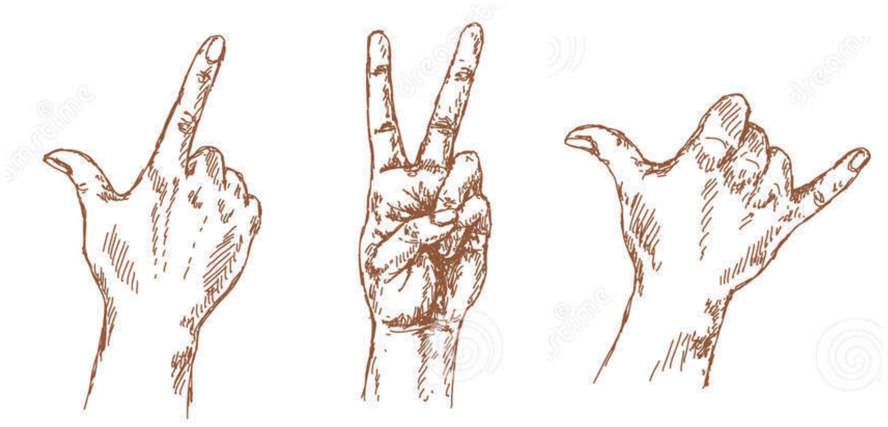


Figure 3.4: Equivalent hand poses starting scaling mode when hand is over the object. Temporal coherence avoids unwanted manipulation due to sensor errors.

## 3.3 Scaling: Iconic Gesture

The solution found to perform object scaling without using the second hand consists of using a reasonably intuitive and non-ambiguous hand pose that the system will recognize in order to switch to the scaling mode and then the actual scaling is mapped on horizontal hand translations while the recognized hand pose is kept. As we want to drive the selection of rotation/translation/scaling with just one hand over the object, the gesture should not be similar to object grabbing. We opted for a gesture consisting of putting two fingers straight and extended. This choice is due to the sufficiently low stress deriving from the pose [7], the possibility of realizing it in different ways (see Figure 3.4, allowing an easy user adaptation. To avoid ambiguities and problems due to finger tracking errors, to switch to the scaling mode, we require that the fingers' position is maintained for at least 1 second before switching mode. After this time, horizontal motion is mapped into scale: moving the hand to the right the object is enlarged, moving to the left it is downscaled. Visual feedback of the scaling activation is provided by slightly increasing the brightness of the selected object color. To exit this mode it is sufficient to change hand pose.

## 3.4 Implementation and Results

We implemented versions of the interaction solutions proposed in Virtual Reality environments realized with Unity 5.3 (`www.unity3d.com`), using the Leapmotion (`www.leapmotion.com`) controller and Orion beta API along with the OVR plug-in (`developer.oculus.com`) in order to enable the use of a HMD for stereoscopic display through an Oculus Rift DK2. Since the task presented to our test users is designed to be performed on a workbench-like virtual scene, we opted for a fixed position of the Leapmotion controller on the desk, so that its detection volume could include all the relevant space required by the user's hands for the task completion.

## 3.5 Software Application and Task

The whole application used for the test was built with Unity 5.3 (`www.unity3d.com`) framework. Our application features the full implementation of our Knob metaphor along with a Handlebar implementation, following the description in the original paper s [5, 59] regarding the single-object manipulation. Besides the techniques, the application also features a task used to evaluate and compare users' performances. We designed an information searching task, with the idea of stressing on both accuracy and interaction fluency. The idea was to have users performing the same task multiple times in series and the task can be summarized in two main parts:

- **I**nformation retrieval: A colored cube is presented at the center of the scene between 4 numbered tiles that are used later for the "information check" subtask (Figure 3.5a). A different texture is loaded on the cube from a pool of 4 with each run. All the 4 textures in the pool look exactly the same besides a small detail and are designed to color each of the cube faces with the same tint. They only variant between them is a small digit, ranging from 1 to 4, and appearing on a different face for each texture. Once the test begins, the user is asked to use all the necessary transformations actions available for the current mode (i.e. knob metaphor or handlebar) to manipulate the cube in order search for the small number. The initial position of the cube is set so that at least a rotation and a scaling action (to magnify the number) are required to identify the digit value.

- **I**nformation Check: Once the value is discovered the user has to position the cube on one of the 4 tiles, also numbered from 1 to 4. If the number on the tile matches the number on the cube texture, the task is successfully completed (Figure 3.5b). The correct tile will also change its color from white to green once the cube is placed, but only after it has been released, in order to prevent

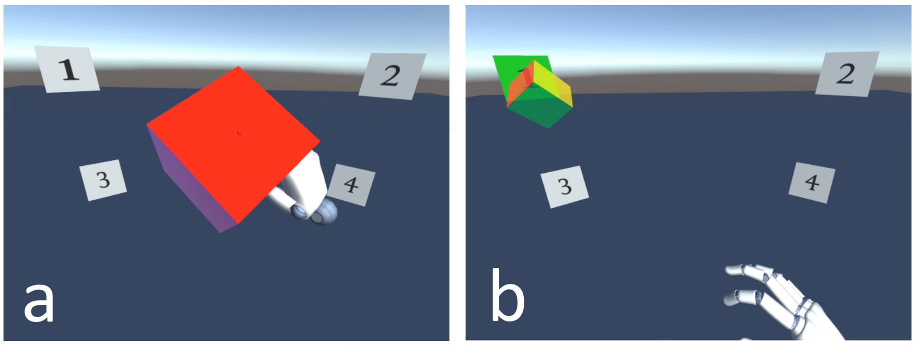


Figure 3.5: Task sequence: a) Information retrieval, a small number "1" is present on the red face. b) Information check, the cube is correctly placed on the tile labeled with the corresponding number.

the user from getting unwanted hints on the solution by simply hovering the cube across the tiles. Furthermore, to avoid accidental activation of the tiles during the execution of the first part of the task due to an excessive scaling of the cube, the collision detection is done relatively to the center of the cube rather than its bounding volume.

## 3.6 Experiment Overview

To perform the experiment, we selected a group of 24 subjects, with no experience of 3D interfaces and VR in general. Each user performed the described task in 2 series of 4 runs each. Half the subjects started with the Knob metaphor as their first technique while the second half started with the Handlebar. For both techniques, regardless of the order, all users were given a 5 minutes time to familiarize with the manipulation of a neutral object (i.e. a white cube) with no specific task to perform and a small break in between the two series. To further remove biases, each user, for each technique, performed the 4 runs going through all the 4 different textures. The order in which they were assigned to the runs was based on a latin square design.

During the experiments, the application developed for the task automatically collected data about execution performance such as the execution, the time spent in idle, and for the Knob metaphor, the whole sequence of transitions between actions with associated timestamps. This recording was not possible with enough reliability for the Handlebar technique due to the overall uncertainty of the action in progress derived by not having separated DOFs.

At the end of the two series, each user provided further data with questionnaires surveying their test experience. We collected data for both techniques, in particular about: learnability, as in how easy/hard was to understand the interaction mechanism, usability in terms of how practical they were in order to complete the task and gestural comfort to estimate the perceived fatigue. The questions were asked so that, by using a Likert scale ranging from 1 to 5 the lower value was always considered the negative one for the judged parameters.

## 3.7 Results

We examined the data collected by the task application. For each run of each series/technique, we averaged the times of all users, thus having 4 averages for each technique. These results are shown in Figure 3.6. Users took on average 17.6 s to complete the task with the Handlebar method and 44.6 with the single-handed one. It is a large difference, but the trend suggested by the curve indicates that, while in successive trials the handlebar performances are approximately constant, there is a steep decrease in the task completion time with the single-handed method. This suggests that, while the handlebar is more intuitive and immediately learned, the single-handed manipulation proposed requires a small training phase, but then could be probably similarly efficient. This argument is further supported by the decrease of deviation in subsequent runs. In fact, by considering completion times excluding the "idle" time segments of each run, the averages of times decrease uniformly across the runs of both technique of a factor of 0.6, but more interestingly the variability of data

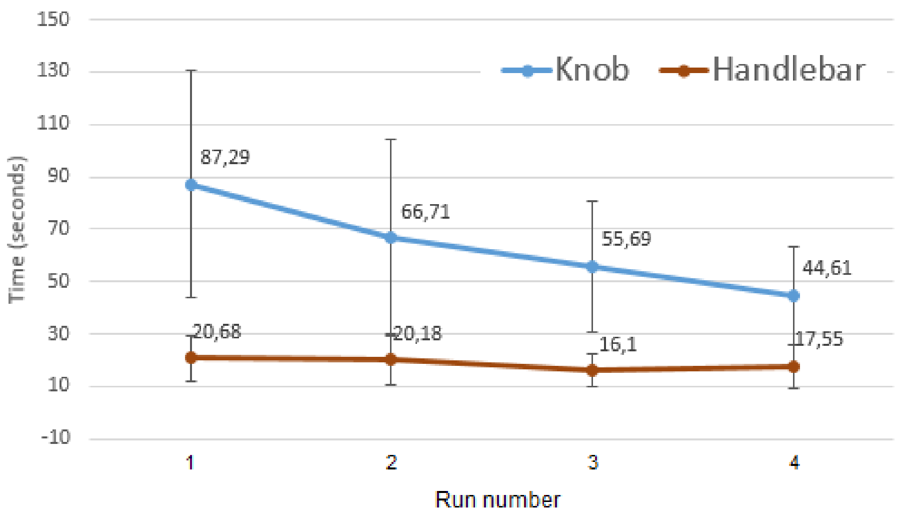


Figure 3.6: Task completion times (Knob vs Handlebar)

lowers with subsequent runs (Figure 3.7), suggesting that not only the performances of the Knob method improve with more runs but it improves overall across all the subjects. This effect might be due to the difference in learning speed between users but, nonetheless, data suggest a certain convergence towards improved performances regardless of that.

An interesting observation can be made on the task completion recordings is that user did not follow a fixed sequence of actions on the different manipulation DOF, as it could be expected. The action sequence could in principle have been split in a single rotation and a single scaling for the information retrieval part, followed by a final translation for the information check part.

Users performing the task with the Knob technique did not act in this way and performed on average sequences of 15.9 different actions (excluding idle) and in different orders to complete the task, with an average deviation of 5.6.

This means that they did not take full advantage of the DOF separation and that the improvements of the performances with repeated trials could have been obtained with an optimization of the sequences.

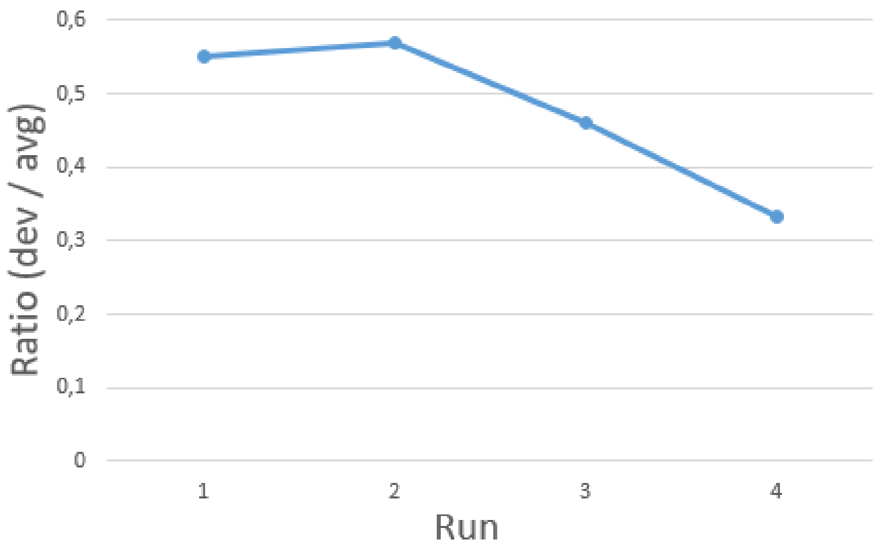


Figure 3.7: Times averages deviations excluding idle time.

Regarding questionnaires data, we report the averages of answers to questions previously described, in Figure 3.8. Despite the outcome not being very encouraging for the Knob metaphor we can still notice, based on the Likert scale ranging from 1 to 5 that our method hasn't received overall negative scores. This leads us to believe that the method is still viable and could probably be better evaluated when applied to tasks taking advantage of the free hand to perform other actions simultaneously, compared against bi-manual techniques like Handlebar, requiring such actions to be performed in sequence.

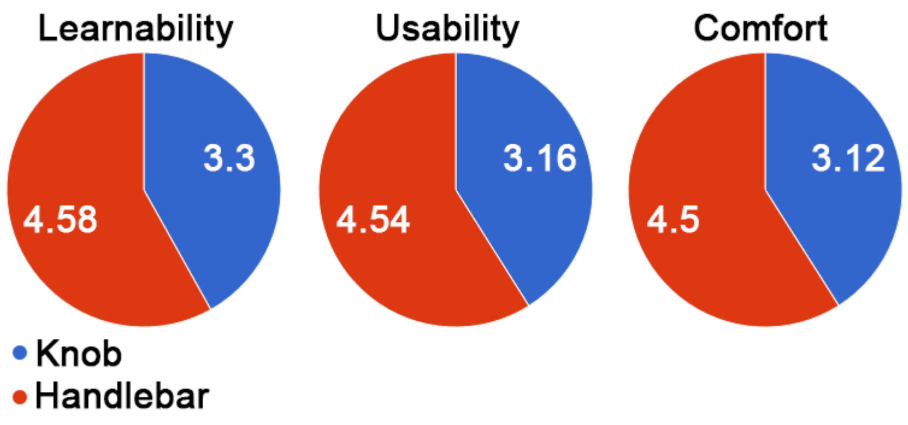


Figure 3.8: Questionnaires answers regarding the evaluation of techniques.

## 3.8 Knob Discussion

We this technique We tried to provide a viable uni-manual technique to perform mid-air manipulation in immersive VE (i.e. the Knob metaphor). To ensure the method provides enough accuracy for generic tasks we designed the interaction with minimal interferences between the different actions through DOF separation. We evaluated the performance of our technique by comparing against the Handlebar metaphor its performance in an information searching task and its viability through evaluation questionnaires. Our findings led to believe that the proposed technique is sufficiently usable but not outstanding, in fact, without training, it is not as effective as the bi-manual manipulation with the Handlebar metaphor in classic manipulation tasks. However, experimental results also show that the difference in performance clearly scales down as the users become more accustomed to the Knob technique. Considering these results it's reasonable to believe that not only the Knob technique is at least a viable technique in general, but also that it can be a method of choice for all the applications combining manipulation with other actions potentially performed with the non-dominant hand.

# Chapter 4

# The Smart Pin

While our previous approach enabled the rotation action through a metaphor meant to be intuitive by recalling a real-world typical action with knobs, in this next work we tested the effects of choosing a more indirect approach through the use of a 3D widget to enable all the manipulation actions. In fact, by looking at the results of our work on the Knob technique, our hypothesis was that users might prefer a more streamilined interaction. This idea was supported mainly by the questionnaire data showing excellent scores for the Handlebar technique. The main goal of our new method is to provide a single-hand interface to perform translation/rotation/scaling, separating degrees of freedom and allowing both robust and effortless mode change and intuitive and easy to learn gesture mapping for the separated DOF. The solution proposed is based on a simple widget that allows mode selection with simple grabbing actions in different positions and on the implementation of separate and effective mappings for translation (direct mapping), rotation (Arcball inspired) as in the Finger-Point described in 2.2.1 and scaling (uniform rescaling proportional to the distance from the pin center). The algorithm controlling the manipulation is, again, a state machine, starting in the idle mode (see Figure 4.1). Upon approaching with the hand the target object, the system status is changed from idle to "object selected", and the user is prompted with a pin with two caps pinning the object in its center of mass and orienting itself towards the user's hand (see Figure 4.2). The pin keeps following the hand and remains active as long as the hand is in a close distance range from the object. In case of multiple objects within the distance activating the "object selected" mode activation, the closest one generates the shown pin.

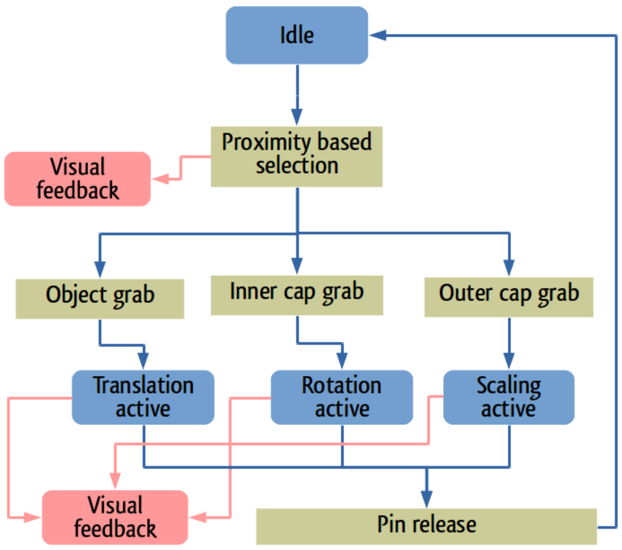


Figure 4.1: Smart Pin algorithm flow chart. Thanks to the visual feedback, once the potential object manipulation is activated by proximity, the user can activate easily and unambiguously one of the three manipulation modes with a simple grab gesture in different positions. Grab release gesture deactivates the current transform mode restoring the idle state.

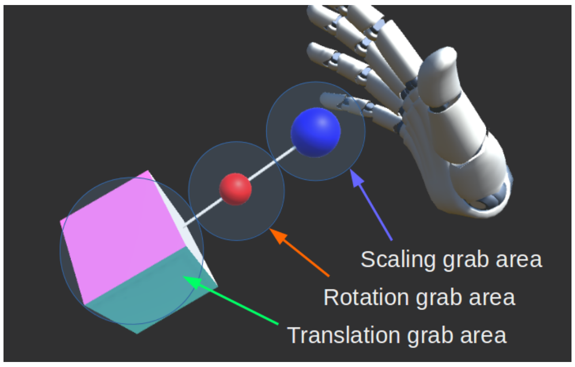


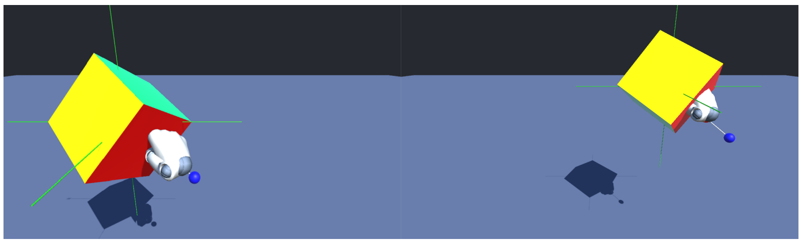
Figure 4.2: The Smart Pin widget. In our tests visual feedback shows the tracked hand.

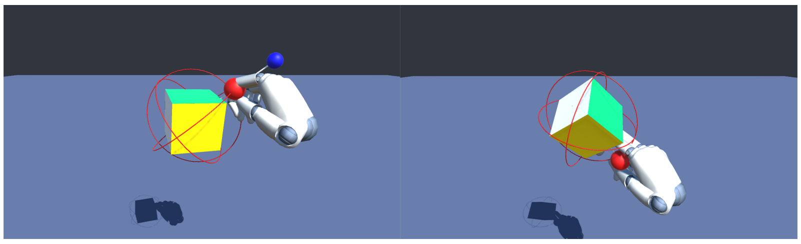
On this widget, the user can easily activate one of the three available manipulation modes by grabbing the object center (translation) or one of the two caps (rotation and scaling). Position and color of the pins allow an unambiguous selection as the regions where the grab gesture activate the modes (shown in Figure 4.2) are non-overlapping spheres around each cap plus one around the target object center, with size depending on the object scale. By releasing the grab gesture any action is interrupted and the state of the pin will return to idle. The relative position of rotation and translation pin caps has been chosen according to user preferences in preliminary tests. As the pin orientation follows the hand-center direction, the user can move from the activation zone of one mode to that of another one just putting the hand closer or farther from the object center. To help the user to understand the system status, the pin cap corresponding to the mode that would be activated by grabbing in the current position is highlighted by temporarily increasing its size. The length of the pin is scaled with the object so as to keep the same visible part outside the object. In particular, the red cap (innermost) and the blue cap (outermost) positions are described by the following formulas:

$$BCP = OBSR + OPL$$

$$RCP = OBSR + (OPL - \Delta) \quad \text{with} \quad \Delta < OPL$$

where BCP and RCP are respectively the blue cap position and the red cap position, OBSR is the selected object bounding sphere radius, and OPL is the outer pin length which is the fixed minimum length of the pin that should never fall inside the object. Other visual feedback mechanisms typical of manipulation widgets are then associated with the different modes when active.


(a) Translation


(b) Rotation

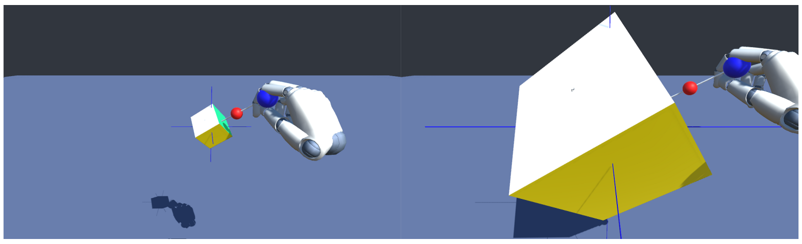

(c) Scaling

Figure 4.3: Couple of snapshots from (a) translation, (b) rotation, (c) scaling manipulation actions performed with the Smart Pin.

## 4.1 Translation

Grabbing inside the object, the user activates the translation mode, linking the subsequent hand palm translation to the object's displacement. When the grab is released, the object is no longer moved and the system returns to "object selected" status. During the translation, Cartesian axes visualization provides a spatial reference to the free direct manipulation (axes don't constrain the motion), as shown in Figure 4.3(a).

## 4.2 Rotation

The rotation mapping provides an intuitive control exploiting that exploits the ideas used in 2D Virtual Trackball/Arcball interfaces [21, 57]. As soon as the rotation mode is activated, the object is locked to the pin and rotates around its center. Its orientation is smoothly interpolated according to the movements of the pin projection on the unit sphere. This choice is motivated by the fact that direct mapping of hand orientation did not demonstrate effectiveness in previous works, due to physical constraints and the lack of robustness when using hand tracking devices. Mapping the orientation on the sphere relying on positional tracking of the user's hand, however, allows an intuitive and robust control, as the success of Arcball in 2D demonstrates. With the mapping to unit sphere projection, the user can control the orientation in a comfortable way with smaller or larger motions according to the distance from the center, and the visual feedback helps in understanding the rotation center position and the effects of hand motion. In the rotation mode, a gridded sphere is overlaid to highlight orientation changes (Figure 4.3(b)).

## 4.3 Pin Scaling

Finally, by grabbing the outer pin cap, it is possible to activate the scaling mode. In the current implementation, when this mode is active, the object is interactively uniformly scaled along its dimensions according to the ratio between the distance of the grabbing point from the object center and the one measured when the scaling action started. This choice intuitive and familiar to many users, as scaling according to the distance from the center is common in many visualization interfaces.

We designed a 7-DOF manipulation with uniform scaling, but, if necessary, a mechanism to scale only along the pin axis could be easily implemented and made selectable by interpreting two different grabbing styles/gestures or by adding a third handle. In the scaling mode, the Cartesian axes are displayed to provide spatial feedback, as in the translation mode, but using a different color (Figure 4.3(c)).

## 4.4 Technique limitations

The implementation of the technique comes with few limitations. First of all, both selection and manipulation actions can only be performed within the user's arm reach range. This design choice was made to preserve the natural style of interaction when possible. This limit becomes particularly relevant to any task that would require the user to perform these actions across a space wider than the arm's reach. It is possible to overcome this limit if the user is able to navigate in the virtual environment or with ad hoc selection and indirect manipulation metaphors (e.g. world in miniature [62]). Another limit of our current implementation is related to the selection method. In fact, the proximity condition to satisfy in order to select an object is not suitable when the density of the objects in a certain volume of space becomes too high. In this case, the selection may be not stable and sensitive to small position changes, making it presumably hard to work with. Different solutions can be implemented for selection in those cases, as done, for example, in [59]. In any case, the specific manipulation tasks used in our tests do not present these critical scenarios.

## 4.5 Implementation notes and experimental setup

We implemented the Smart Pin and Handlebar manipulation methods in VR environments realized with Unity 5.3 (`www.unity3d.com`) and using the Leap Motion Controller and Orion beta API. Finger tracking provided by the device is not always accurate [4] but, as our manipulation methods do not rely on finger tracking except for grabbing gesture detection, the interaction results quite smooth (even if errors may be annoying for visual feedback). Of course, the use of a single sensor may be problematic, in particular cases, for two-handed manipulation systems due to occlusions. This occurs rarely in handlebar interaction with the sensor under the user's arms as it would imply to assume uncomfortable positions.

The experiments took place in a laboratory with a desk and chair setup (Figure 2.4) to make users comfortable. The choice was not critical since the tasks required to interact in a limited amount of space. The Leap Motion was placed on the desk, at a distance of 25-30cm from the user in order to match the sensor detection cone with the virtual scene operation space and to prevent users from having to stretch or crunch their arms while performing the task. Chair height was also adjusted accordingly to users' body size and preferences to guarantee the best possible comfort, prior to the beginning of the test.

## 4.6 Evaluation

To compare the novel manipulation method with the Handlebar technique, we designed two different tasks. The first is a classical docking task, aimed at evaluating the accuracy in reproducing position, scale, and orientation of a reference object. The second is an information search task, testing the efficiency of the method when provided with the necessity of a quick and greedy sequence of geometrical transforms. We proposed both tasks to a group of 24 subjects (16 males, 8 females), all of them without previous experience of immersive VR applications.

Each subject completed the two tasks both with the Handlebar control, implemented as described in [59] and the Smart Pin control. The order of execution of the tests has been inverted on half the subjects to avoid cross-training effects. Before starting the tasks' execution with each method, a short standardized description was provided to the subjects, who had then a fixed training time of 2 minutes to familiarize with the techniques, manipulating a cubic shape without any task to perform. Each task execution has been repeated by each user several times during each session. After task completion, the subjects were asked to fill a questionnaire to record their preferences and subjective opinions about the methods. Furthermore, each subject went back to the lab after one week to repeat the whole test. For each run of both tasks, we recorded the execution time as a direct performance measurement. Following the completion of all the runs, the users had to fill a questionnaire with personal data and scores to evaluate different aspects of the experience.

## 4.7 Search task

The search task was designed with the same characteristics of the one described in subsection 3.5. In Figure 4.8 the steps to the task completion are shown in the case of the Smart Pin. Each user repeated the task four times per session, each time with a different position/value of the searched number, and changing the order of presentation of the configurations for the different subjects with a Latin square design.

## 4.8 Accurate docking task

The docking task requires precise matching of the manipulated shape over the reference one, with small tolerances for positioning, orientation, and scaling. The subject perceives through the HMD a scene with a white cat model on the left and an identical red and semi-transparent shape, with different size and orientation, on the right (Figure 4.5(a)). The initial position, orientation, and scale for both the models was predetermined and based on the run number, to avoid random bias in the run complexity. Subjects have to manipulate the white cat in order to superimpose it over the red one. When the correct centroid position, orientation, and scale are reached, corresponding bright green signs are turned on as a feedback for the user (Figure 4.5(b)). When all the three variables are in the correct range (i.e. ±2.5mm centroid distance for translation, ±10 degrees for rotation and ±0.1 scale ratio from the target scale), the task is completed Figure 4.5(c)). The docking execution requires the use of all the three transformation modes, but, differently to the search case, it also requires accuracy in placement. This task has been repeated three times by each user in each session.

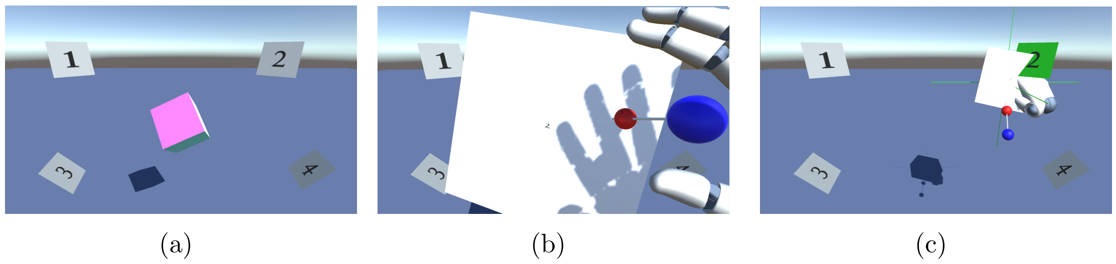


Figure 4.4: Sequence showing the search task phases. (a) Initial position. The subject must find a small number in the range 1-4 displayed on a cube side by rotating, translating or scaling the shape (b). When the number is recognized, he should move the cube on the target with the corresponding value (c).

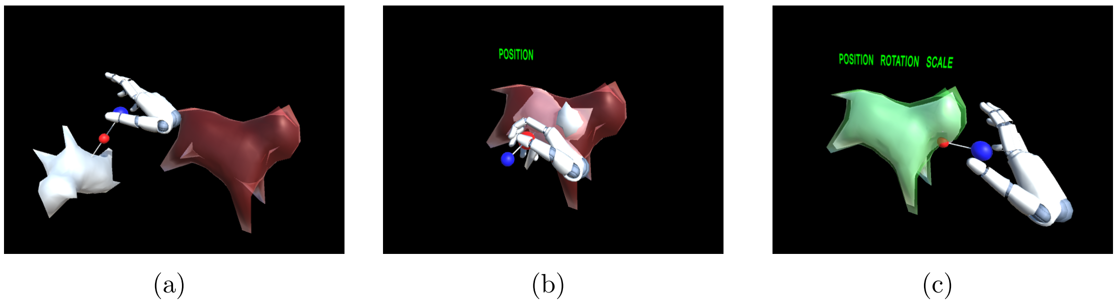


Figure 4.5: The docking task proposed to the test subjects. A white cat model must be picked up from the original position (a), translated to match the reference position of the target model (a similar red and transparent cat) (b) and orientated and scaled in order to be well superimposed to the target model (c).

## 4.9 Questionnaire

A questionnaire was given to the participants to assess their preferences about learnability, easiness of use and fatigue. It also aimed at capturing potential nuisance variables that may influence results (e.g. previous experience of gaming). Participants (after the first and the second session) were asked to give a score (from 1 to 5, higher is better) to the learnability (perceived easiness of the learning process), the ease of use and the physical comfort (lack of perceived physical fatiguing and stress during) of the two manipulation methods, as well as a global indication of preference between them.

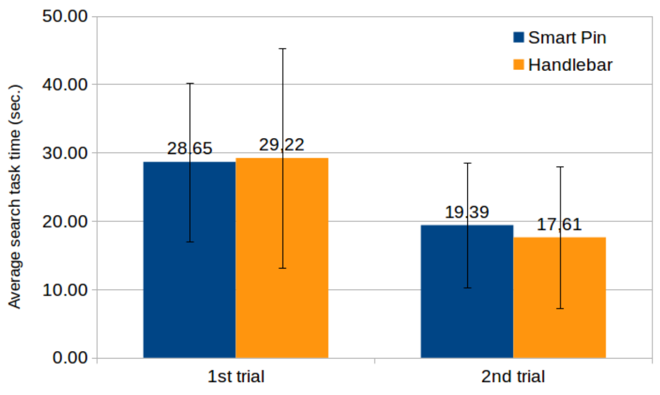


Figure 4.6: Average task completion times for the two trials of the search task. Vertical bars show the standard deviations. Average values are quite similar for the two methods and are similarly decreased in the second trial (one week after the first).

## 4.10 Smart Pin Results: Execution Times

For both tasks, we aimed at testing the null hypothesis that there is no significant difference in task completion times with the novel method and the Handlebar metaphor. We also wanted to test if the novel technique could prove to be more intuitive, less fatiguing and better learnable.

Figure 4.19 shows average execution times with standard deviations for the search

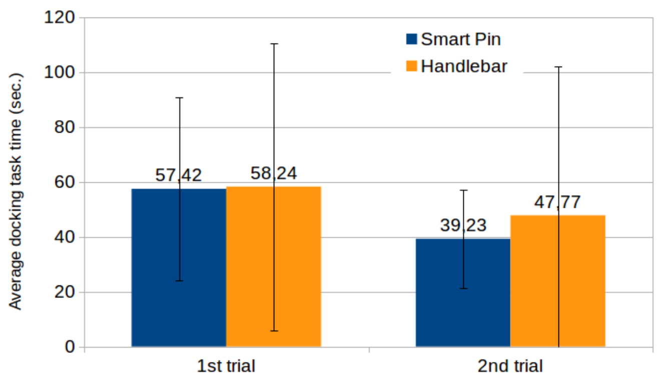


Figure 4.7: Average task completion times for the two trials for the docking task. Vertical bars show the standard deviations. Average values are quite similar for the two methods, and are similarly decreased in the second trial.

task in the first and second session. It is possible to see that the results of the compared techniques are quite close and there is no statistical difference in the efficiency of the two methods both in the first and in the second trial. Furthermore, the methods show a similar learning trend as in both cases we have a similar reduction of the average task completion times.

Similar behavior is seen also for the docking task (Figure 4.20): in both tasks and both trial sessions, at a significance level of 5% we cannot reject the null hypothesis, which is, in some senses, unexpected considering the differences in the two methods. In our tests, we modified three independent variables, namely the manipulation technique (Smart Pin or Handlebar), the task (search and docking) and the different week in which users performed the test (first and second). To evaluate the potential interference between these factors, we performed a Three-Way (within subjects) ANOVA test, resulting in a non-significant score (F=1.506, p=0.2321). We also performed an equivalence test on the time averages with a two-one-sided t-tests (TOST) procedure [55]. We set the lower and upper bounds at $-5$ and 5 seconds. We compared the two techniques within the same week, and type of task for a total of 4 tests and we rejected the null hypothesis (the difference between the means falling outside the equivalence interval) in all of them.

If we analyze more in-depth the outcomes of the experiments we can see that, even if means and variances are quite close in the two conditions, there is a considerably larger variance for Handlebar in the docking task (52.3sec. vs 33.2 sec. in the first trial, 54.0 sec vs 17.9 sec in the second one).

The reason may be due to the fact that the efficiency of this method heavily depends on the spatial abilities of the subjects, that are not uniform in the population. A box and whisker plot (Figure 4.8) shows that even if also the median values are similar and similarly improved in the second trial, the distribution of the quartiles is quite different for the two methods, with the Smart Pin providing, especially in the second trial, most of the values concentrated around the median time (central line), while the Handlebar has first and third quartile values (top and bottom lines of the boxes) still quite spread.

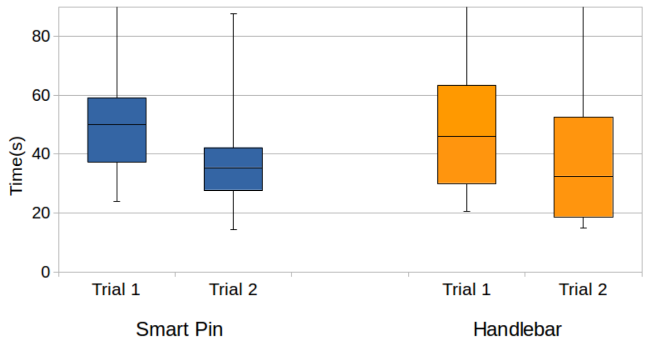


Figure 4.8: Box and whisker plots for the accurate docking task in the two trials with the two compared methods. Median values are similar and similarly improved in the second trial, but, in this case, the Handlebar method reveals a quite larger distance between first and third quartile values (boxes extrema), especially after the second trial.

Note that, especially for the accurate docking task, the completion times of some trials are quite higher than the average/median values. We considered, for this reason, the idea of removing from the statistical analysis the involved subjects, treating them as outliers. However, we verified that removing outliers with a standard criterion the statistical outcomes would not have changed, and the removal would have also broken the Latin square design of the test, so we finally decided to show the analysis performed on the full set.

## 4.11 Smart Pin Results: Questionnaires Outcomes

If we look at the ratings obtained with the questionnaires, we are finally able to see statistically significant differences between the tested methods. While the scores given to the learnability of the systems are quite similar (coherently with the similar improvements in the execution times), those given to the ease of use and to the physical comfort of the methods were significantly different (Figures 4.9,4.10,4.11).

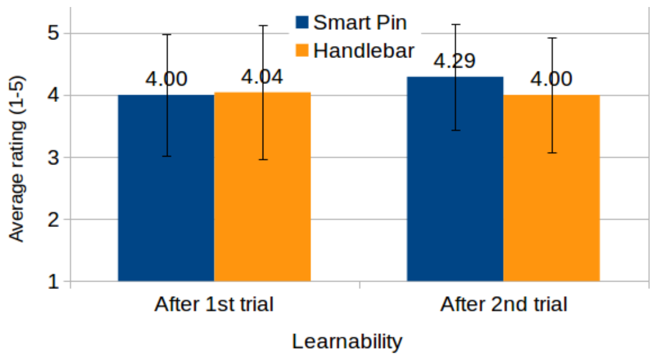


Figure 4.9: Bar charts showing the differences in the subjective scores given by subjects about learnability. Scores range from 1 (worst) to 5 (best). Both methods are appreciated, and the average scores for the Smart Pin method are higher after the second trial. There are no statistically significant differences between the two methods.

At a significance level of 5%, a Wilcoxon signed-rank test indicates that we should reject the null hypothesis on equal ease of use after the second trial (dof=22, z-valu=-2.29, p-value=0.021).

The hypothesis of equivalent physical comfort is also rejected for both trials (dof=22, z-value=-2.15 and p-value=0.031 for the first; dof=22, z-value=-2.58 and p-value=0.010 for the second). This was expected, as the widget based solution reduces the need for large motions and hands' coordination. Note that average scores of both methods are quite good and similarly increasing in the second test.

Another question presented to the subjects was to give an overall preference between the two methods. After the first trial, 16 subjects (66.7%) indicated the Smart Pin as the favorite method and 8 (33.3%) the Handlebar. After the second one,19 subjects (79.1%) declared a preference for the Smart Pin method and 5 (20.9%) for Handlebar (Figure 4.12).

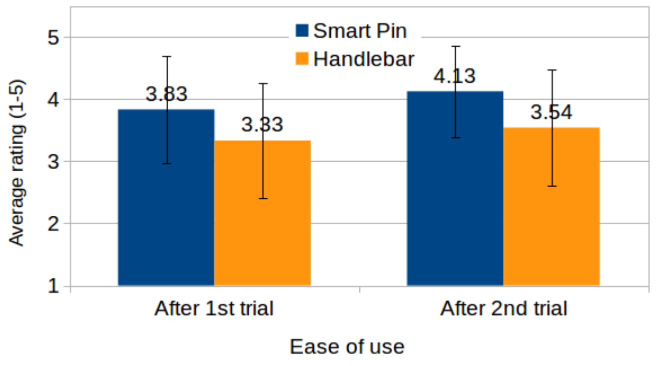


Figure 4.10: Bar charts showing the differences in the subjective evaluation of ease of use. Scores range from 1 (worst) to 5 (best). There are statistically significant differences between for the second trial (p=0.02 in a Wilcoxon Signed-Rank Test).

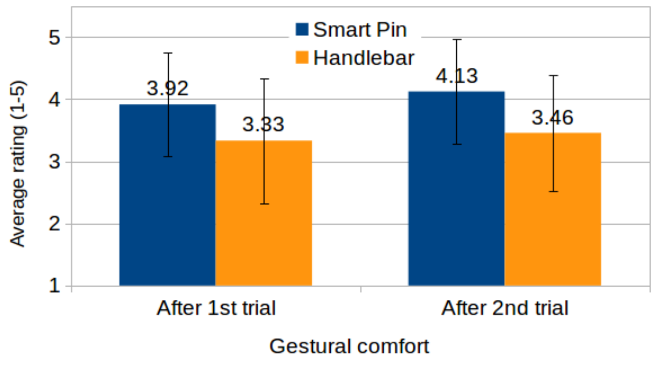


Figure 4.11: Bar charts showing the differences in the subjective scores given by subjects about gestural comfort. Scores range from 1 (worst) to 5 (best). There are statistically significant differences between the two methods both after the first (p=0.03 in a Wilcoxon Signed-Rank Test) and after the second trial (p=0.01 in a Wilcoxon Signed-Rank Test).

## 4.12 Smart Pin Results: Effects of gaming experience

Even if the subject sample is limited, and results should be considered with care, we investigated the potential influence of other variables collected from subjects.

It is well known that gaming experience can enhance spatial abilities like mental rotation [23]. The subject set included 15 elements with a regular gaming activity (identified by a minimum of two hours of playing per week), and 9 not using video games on a regular basis. Looking at the execution times, it is possible to see that there is a significant difference (dof=22, T=3.07547, p-value=0.002 in a t-test) between the two groups in the search task completion times with the Smart Pin method. Experienced subjects took 21.1 sec. (std. dev. 9.0) on average while non-game-experienced subjects took 30.7 sec. (std. dev. 12.4).

The average execution time in the docking task is also lower in the experienced group, but there isn't a statistically significant difference, due to the large standard deviation of the Handlebar average times. No statistically significant differences were recorded between the groups on the docking task.

Looking at the questionnaire outcomes, there is a visible and coherent reduction of all the scores in the non-experienced group for the Handlebar method (see Figure4.13), while the ratings given to the Smart Pin are similar. In the case of the Handlebar, the difference in the learnability score is statistically significant (dof=22, Z-Score=2.21, p-value=0.027 in a Mann-Whitney-Wilcoxon U-Test). This may support the claim that this method requires an enhanced spatial ability and it is perceived as more difficult/less comfortable by people with lower spatial skills' training. Instead, the Smart Pin is perceived similarly easy to learn by trained and untrained people, independently on task performances.

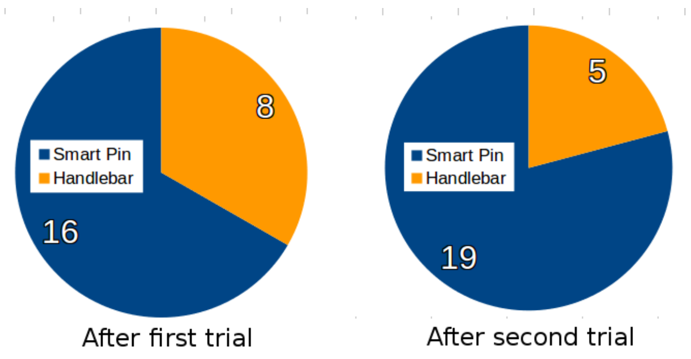


Figure 4.12: After the second trial, the number of subjects declaring an overall preference for the Smart Pin method raised from 68% to 82%.

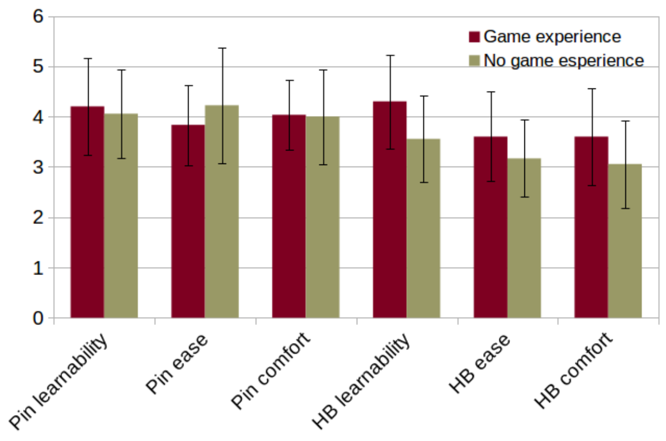


Figure 4.13: Subjective grades seems coherently lower for non-game experienced people only for the Handlebar method, suggesting that the Smart Pin may be more acceptable for people with reduced spatial ability.

## 4.13 Smart Pin Discussion

The solution we proposed has, in our opinion, several interesting aspects that could make it a successful choice for many VR applications, and could be easily adapted to a variety of environments. First, it requires a single hand and decouples degrees of freedom, but does not require gesture recognition or sequences of actions to switch modes as other single-handed methods [6]. Using this method, then, rotations are performed in a very intuitive way, familiar to most users due to the close similarity to the Arcball/Spaceball widgets used in 2D interfaces.

The DOF selection based on grabbing the pin caps or the object volume is not ambiguous and offers clear visual feedback. The shape of the pin caps is spherical in the current implementation but could be changed to increase the affordance. We are currently testing different appearances for the two pin caps. The idea is to replace the spheres with shapes suggesting the associated actions, or showing billboards with classical manipulation icons near the caps. We show examples of affordance-enhanced pin caps in Figure 4.14. These solutions will be tested with users to find the preferred option.

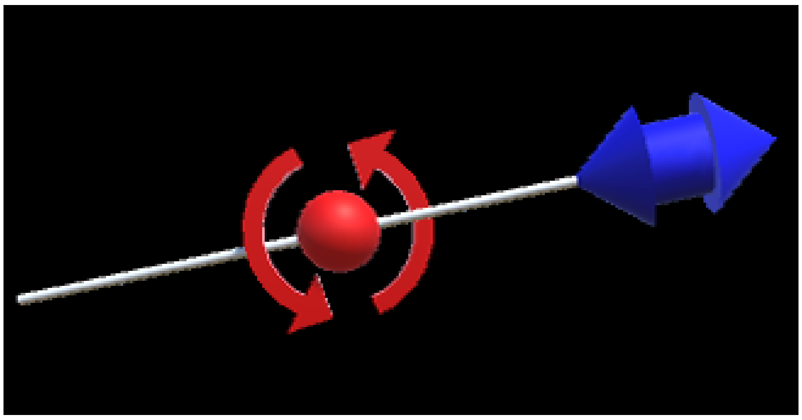


Figure 4.14: Examples of affordance-enhanced pin caps.

We evaluated the usability of the Smart Pin on a simple setup to focus on specific interaction aspects, but the method can be applied as well on more generic setups. Using a hand tracker attached to the HMD, it could be, for example, exploited in applications allowing the user to stand and walk in the 3D space. The widget, using the cursor feedback mode, could be employed as well in non-immersive applications based on stereoscopic or standard displays.

With the current implementation, the method only allows the manipulation of objects of the same size of those that could be manipulated in the real world. However, the technique could be easily modified to handle objects of different size, possibly combining the Smart Pin with other metaphors like the worlds in miniature [62].

## 4.14 Other studies: Device vs. Deviceless interaction

In order to maximize the usability of VR applications, different kind of manipulation techniques can be chosen. Researchers have proposed a variety of metaphors and developers are adopting different choices in the new applications that are emerging thanks to the diffusion of low-cost VR setups.

The success of a manipulation method, however, does not depend on the metaphor only, but also on constraints that are coming from the specific hardware solutions and may depend also on visualization/feedback issues. The effect of these factors, however, is much less investigated in the literature.

By analyzing the outcome of the user test, using a typical docking task controlled with the different methods, we can derive useful hints and guidelines that can help developers with design choices to achieve good performance results for the manipulation tasks involved in a specific VR application.

In this work we designed experimental tests using two different manipulation techniques (Simple Virtual Hand [39] and Smart Pin widget [?]), two different control systems (Leap Motion and Oculus Touch) and providing two different kinds of visual feedback (simple cursor and tracked hand visualization when possible).

### 4.14.1 Experimental design

We tested device and deviceless input by using respectively an Oculus Touch Controller and a Leap Motion Controller. In the deviceless condition, we also tested two different types of visual feedback: the first showing a 3D hand model of a hand rigged on the joints tracked by the sensor and the second displaying a 3D cursor represented by a sphere with a diameter approximatively corresponding to the finger width. These different combinations generated three different experimental conditions: Leap Motion with 3D hand model visualized (L-H), Leap Motion with a 3D cursor feedback (L-C) and Oculus Touch with 3D cursor (T-C). All these three conditions were examined in a user test consisting of a simple docking task to be performed with two different manipulation techniques: the Simple Virtual Hand [39] (SVH) to test a more direct method and the Smart Pin [11] (SPin) to test a widget-based method. Our intention with this choice was to verify if any possible emerging trend would show across both techniques based on a direct approach (i.e. SVH) and indirect ones (i.e. SPin). To obtain an IVE we used the Oculus Rift head-mounted display as a stereoscopic output device.

Our test group consisted of 18 people (12 males and 6 females), with no previous experience in VR contexts and with age ranging from 20 to 29 years (average 23.1). The group was divided into two sets: the first performed the docking task using the Smart Pin technique first while the second one started with Simple Virtual Hand version. For each technique, each user was asked to complete the task 3 times for each modality (L-H, L-C, T-C) with a different order, following a latin square design. Each modality consisted of a learning session of fixed time (1 minute) with no docking involved and three runs of the actual task for a total of 18 runs per user. W collected the execution times for all the tasks and the number of actions taken to complete the docking. When all the required tasks were completed, each user had to fill a questionnaire to evaluate different aspects of their experience during the test.

### 4.14.2 Experimental results

Figure 4.15 shows box plots representing the distribution of the average task completion times of the 18 users with the three different implementations of the SVH technique. It is evident using the Touch performances are significantly better, while

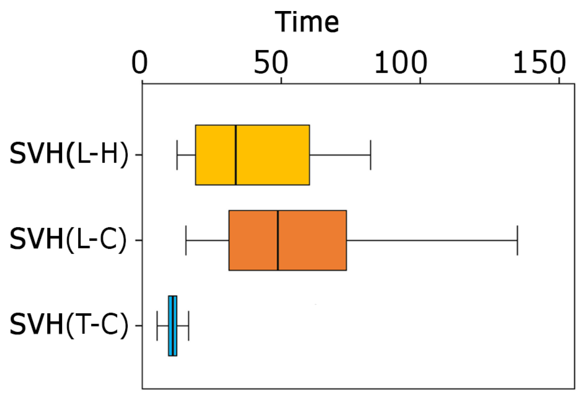


Figure 4.15: Box plot showing task's completion time, in seconds, using the SVH technique in the different conditions. The chart presents the median, 1st and 3rd interquartile ranges (boxes) and 95% confidence interval (whiskers).

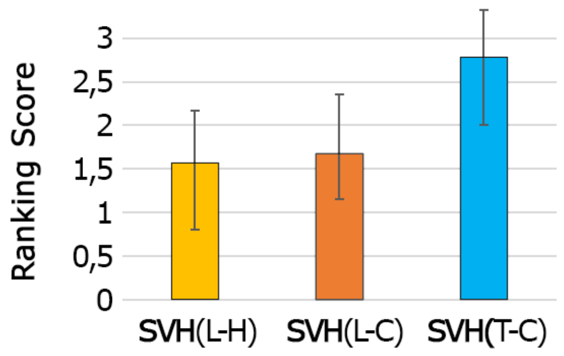


Figure 4.16: Average ranking scores (higher is better) given by users sorting by preference the different combinations of Controller/Feedback options in the Simple Virtual Hand task.

different feedback solutions for the Leap Motion mode does not create statistically significant differences. Touch-based interaction was ranked as the preferred method by 15 subjects (Figure 4.16), while there is no clear preference between the two visual feedback choices with Leap Motion.

In the Smart Pin case the task required also a scaling of the object before docking, however, this did not create a large increase of the completion time with Leap Motion interaction. Device based interaction is still significantly faster (see Figure 4.17), but the difference with the hand tracking-based methods is decreased. No significant differences between the visual feedback choices are measured. Touch-

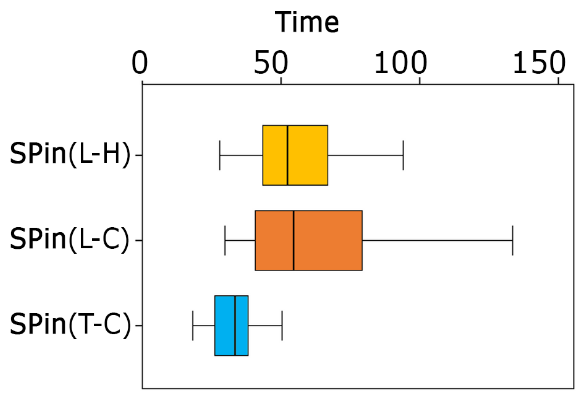


Figure 4.17: Box plot showing task's completion time, in seconds, using the Smart Pin technique in the different conditions. Times are not comparable with those in Figure 4.15 as the task included scaling.

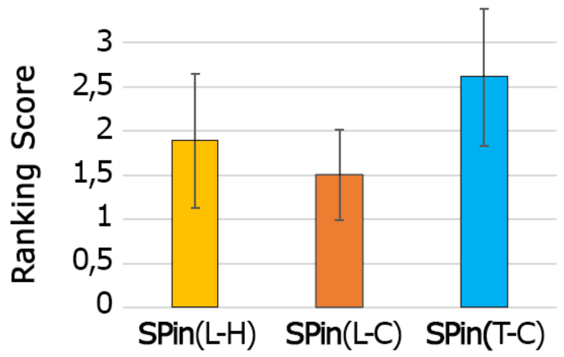


Figure 4.18: Average ranking scores (higher is better) given by users sorting the different combinations of Controller/Feedback options in the Smart Pin.

based interaction was ranked as the preferred method by 15 subjects (Figure 4.18), while there is no clear preference between the two visual feedback choices with Leap Motion.

### 4.14.3 Design Choices

In this work, we presented an initial effort aimed at evaluating different choices related to tracking devices, techniques, metaphors and feedback in manipulation tasks. Despite the limited number of comparisons and user, the collected data and analysis hint to at least two design guidelines and confirm the tight interaction between all the different design aspects investigated:

- Handheld devices, that guarantee shorter manipulation tasks' completion times, should be preferred if available in the planned setup and if the efficiency is a critical factor for the target application.
- Direct manipulation (VH) guarantees shorter execution times with the devices, but are not necessarily preferred to a widget based method by users.

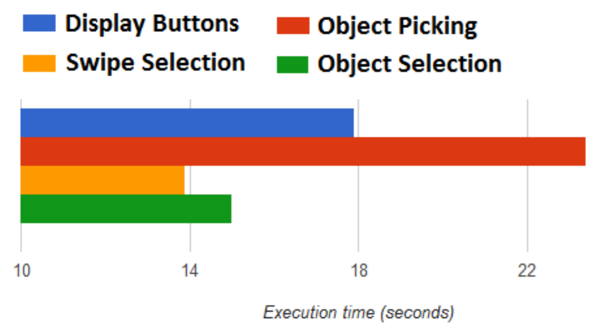


Figure 4.19: First task completion times

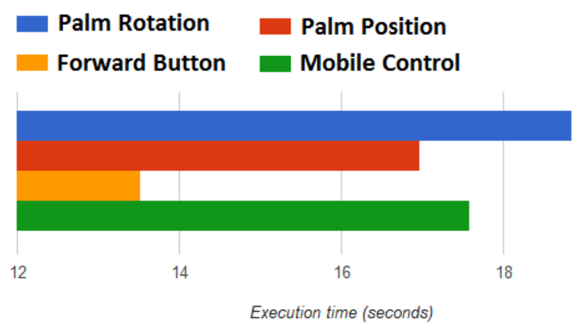


Figure 4.20: Second task completion times

# Chapter 5

# Gesture Recognition

Gestural interaction is particulary successful on 2D touchscreens where it is easy to determine the beginning and the end of the gesture by using the contact between fingers and display surface. In 3D deviceless interaction, the beginning of a gesture must be automatically detected from the gesture itself. The use of pattern recognition tools may allow a robust recognition of a gesture learned after the gesture realization, that may be good for a sign language interpreter, for example, but not for a manipulation tool that should give visual feedback within a reasonable time. A possible trick to solve this issue typically applied to interfaces is to use a second hand or a vocal interface, or the recognition of a coded gesture to tell the system that the gesture is starting or it is finished. Furthermore, in a manipulation interaction gesture start involves also the "grabbing" of the object of interest and gesture end involves also its release. This means that the algorithm should localize with a sufficient accuracy the position of the grab and, more difficult, the desired location of the object release. Especially the last task requires, in our opinion, both a smart geometrical representation of the hand/finger trajectories, possibly invariant to users gestures realization and a smart learning procedure in order to characterize the key actions and the corresponding desired object position. It is a really challenging task, but the ideas that could be applied to find a reasonable solution may be the same applied in classical robust point matching and landmark location. Keypoint trajectories could be simplified and normalized, mappings between trajectories could be encoded as functions, the evolution of connected point could be treated as a surface. And, in the same way learning approaches are used to find discriminative keypoints for specialized recognition tasks on 3D meshes [25], it is possible to think that on a geometric encoding of hand trajectories, keypoints that have a user-independent recognition of gesture limits could be learned through the collection of example data. The use of data collection to learn gestural feature is already applied in HCI, for example in gesture elicitation experiments [49, 2], and, in some sense, with the same approach, we could learn how users behave when doing "naturally" simple gestures like grabbing, translating and rotating a virtual object. Registering example gestures and de-coupling 3D trajectories and velocity patterns it would be possible to find specific and invariant keypoints to identify the beginning and end of gestures.

Another big problem is related to the accuracy of gestures, that in manipulation tasks is particularly important. The accuracy of object positioning in manipulation is limited by the lack in accuracy of tracking and the possible occlusions of keypoints. However, this problem could be mitigated by the redundancy of the data, and all the research on robust descriptors or partial retrieval can surely help in finding ad hoc solutions for the task. Approaches for partial shape retrieval, like Bag of Words [40, 66] could be, for example, applied to select only the partial information correctly describing the gesture and captured by the device/tracking library used. Furthermore, learning from example, using regression techniques, the relationships between keypoint positions and desired manipulation position could help in improving the accuracy of the gesture localization. Another problem in this particular case is the delay in visual feedback, as the detection of the release gesture requires a backwards analysis and if the grabbed object is moved together with the hand, there is a discrepancy between the expected manipulated object position and the visualized position in the virtual representation. In this work, we developed a simple algorithm based on the dollar algorithm family for 2D cases and explore several metrics that can be used for mid-air gesture comparison. By adopting smart choices in gesture traces processing and comparing, it was possible to obtain very good results in the retrieval and recognition of simple command gestures, from complete or even partial hand trajectories. The approach was also extended in order to recognize gestures characterized by both hand and finger motions and tested on a recent benchmark, reaching state of the art performances.

## 5.1 Background Theory and Related Work

There is a large amount of work in the literature on gesture recognition, with applications based on different kinds of input data and designed for a variety of applicative domains.

Many gesture recognition methods are based on image sequences processing, exploiting well-known tools such as Hidden Markov models [68], Dynamic Time Warping (DTW), Time-Delay Neural Networks (TDNN) and Finite-State Machines (FTM) [46]. Generic surveys on gesture recognition can be found, for example in [52, 53, 22].

Focusing on dynamic gesture recognition, it should be noted that only a few methods are based on processing hand position trajectories or hand skeleton data.

The use of trajectory analysis for gesture recognition has been proposed in sign language decoding for example, where relatively long sequences with several features determine the gestures' semantic.

In [41] heterogeneous features and HMM are used for sign characterization. In [56] a descriptor for multiple motion trajectories based on the kinematic relation among different moving parts is proposed. In [69] a method to parse trajectories to derive a gesture descriptor based on sequences of segmented primitives is presented and tested also on a dataset of Australian sign language gestures.

Some of the cited gesture recognition methods based on the encoding of multiple trajectories of body keypoints, provided by trackers and depth sensors APIs, are similarly used for action recognition. Several datasets providing labeled trajectories of human skeleton joints have been proposed and a really huge number of encoding methods for these sequences of skeletal nodes' paths has been proposed. Interested readers can look at specific surveys, e.g. [32].

However, the rapid diffusion of applications like virtual and mixed reality simulators, virtual museums, immersive visualization systems, public displays, and the availability of 3D tracked devices and low-cost body skeleton and hand trackers suggest that the use of 3D paths analysis will be quite common in the near future not only for the labeling of multiple complex trajectories but also for the development of interfaces based on the recognition of short command gestures performed with hands and fingers.

In this case, the use of gesture decomposition, local to global features encoding or the use of advanced pattern recognition for path classification may be not necessary, and a simple approach similar to the one used in the popular "dollar" algorithm family in the 2D case may be feasible.

This class of algorithms, based on a simple approximate spatial registration and the subsequent pairwise comparison of gestures shapes, is quite popular for 2D touch-based gestural interface design as it allows a quick prototyping of specific recognizers given a small set of example gestures without the use of complex classifiers libraries or large training sets.

This kind of approach has been already extended to 3D trajectories in the "3 dollar recognizer" [36] and in the following Protractor 3D [37] algorithm. These methods, however, rely on unusual normalization options for gestures probably derived by the adaptation of the original 2D methods and haven't been tested on large datasets related to specific applications.

In order to have a better evaluation of the feasibility of a "dollar"-like approach in the design of 3D mid-air interface, we created a novel dataset with 3D tracked hand trajectories related to command gestures and tested a relevant number of solutions for path processing in order to find the most suitable solution for the practical task of recognizing short mid-air gestures.

Furthermore, we considered also another recent dataset, coming from the SHREC'17 contest on Hand Gesture Recognition Using a Depth and Skeletal Dataset [58]. This dataset features short gestures similar to those we are interested in, but where the gestures are not all characterized by the hand trajectory only, as it includes also "fine" gestures where differences can be seen looking at multiple finger motions. The dataset includes a complete tracking of hands' joints.

The outcomes of our work are the demonstration that using very simple trajectory processing options it is possible to obtain a method ("3 cent recognizer") with very good gesture recognition/retrieval performances on gesture dictionaries including interface-like commands defined by a single hand trajectory. These performances remain quite good even if only a small part of the gesture is considered and if the sampling of the trajectory is quite coarse.

Furthermore, exploiting the SHREC'17 dataset and adding to the hand trajectory comparison the comparison of the corresponding sequences of finger orientations and joint finger angles, we developed an extension of the method able to process and recognize gestures characterized by both hand trajectory and finger motion. The method, still simple and easy to implement, outperforms the state-of-the-art on the contest data without using depth images information.

## 5.2 Command gesture datasets

We are interested in recognizing short gestures that can be interpreted as commands for a user interface in an immersive virtual reality environment to select different interaction options, or in a mixed-reality environment to control smart devices.

We created, with this goal, a novel dataset including 3D hand trajectories corresponding to 26 different “interface command” gestures. These trajectories were acquired using a Leap Motion device while 13 different subjects were performing the gestures.

During the acquisition process, each user was instructed with an animation showing the ideal trajectory, which s/he had to replicate immediately afterwards. The gesture set includes semi-circular arcs (arc3Dleft, arc3Dright), symbols drawn in a 3D space (caret, check, curly-bracket-left, curly-bracket-right, delete, pigtail, square-bracket-left, square-bracket-right, star, v, x), simple geometric figures (left-swipe, right-swipe, circle, rectangle, zig-zag), 3D polygonal chains (poly3Dxyz, poly3Dxzy, poly3Dyxz, poly3Dyzx, poly3Dzxy, poly3Dzyx) and a 3D spiral. Each gesture is encoded as a sequence of 3D points, representing the position of the dominant-hand forefinger, together with the sample timestamp. Example gesture trajectories are shown in Figure 5.1 .

The dataset is publicly available on the web at the address `https://github.com/davidespano/3cent-dataset`.

Gestures of our dataset are given already segmented, as the data are used to evaluate only methods for retrieval and classification of gesture patterns, not the detection/localization. It is clear that for a practical application, these patterns should be detected/segmented in longer tracked sequences. However if an effective and efficient method for matching gesture words is available, an online gesture recognizer could be easily implemented exploiting a sliding window approach, optimized search

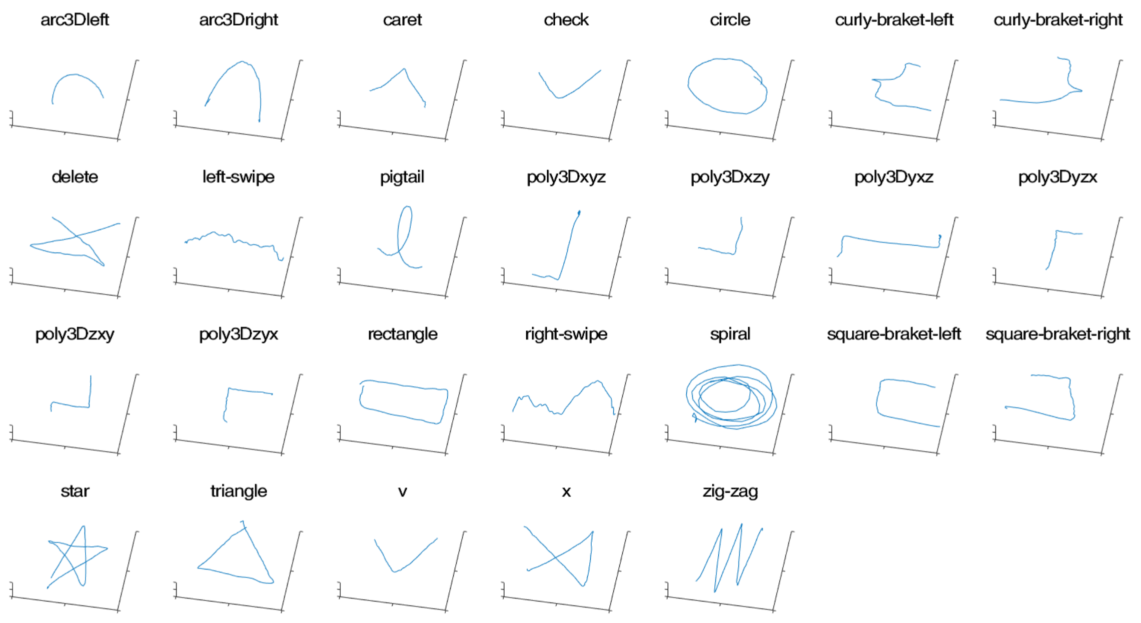


Figure 5.1: Example trajectories from the 26 gesture dataset.

as proposed in [64] or a fast stroke segmentation method similar to the one proposed in [70].

A public dataset with similar features, but including multi-finger gestures has been proposed for the SHREC'17 contest on 3D Hand Gesture recognition. This dataset is also based on 3D hand trajectories, even if it includes gestures characterized by both single hand trajectories and fine movements of the fingers. Authors, in fact, provided hand skeleton movements for 14 gesture classes, 9 “coarse”, e.g. not characterized by finger movements: Tap (label 2), Swipe Right (7), Swipe Left (8), Swipe Up (9), Swipe Down (10), Swipe X (11), Swipe + (12), Swipe V (13), Shake (14), and 5 ”fine”, e.g. characterized by finger movements: Grab (1), Expand (3), Pinch (4), Rotation Clockwise (5), Rotation Counter Clockwise (6), where gesture should be characterized by relative finger movements.

Example coarse gesture trajectories are shown in Figure 5.2.

While coarse gestures can be used similarly to our novel dataset to assess the performance of simple trajectory based gesture recognizers, we exploited the fine gesture

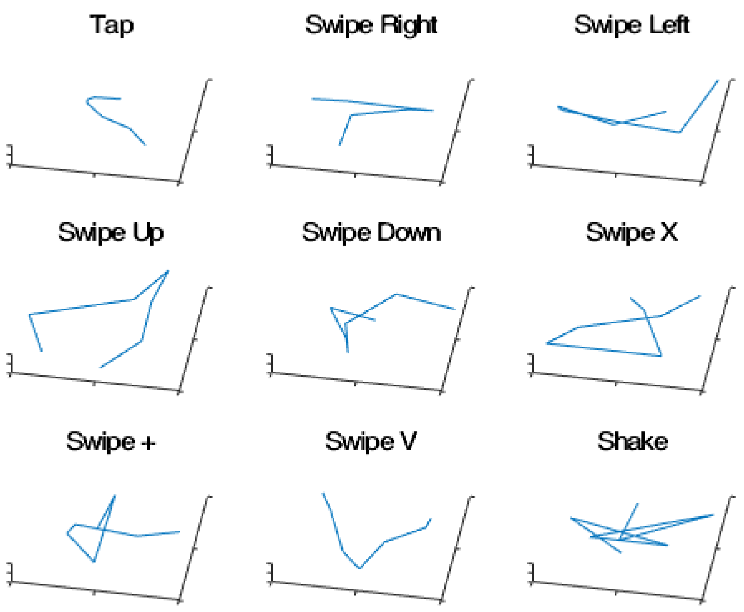


Figure 5.2: Example palm trajectories from the SHREC'17 "coarse" gesture classes.

to demonstrate the possibility of separating and then combining the main hand trajectory and fingers' information comparisons in order to derive a simple, integrated gesture distance measure that gives very good performances in the 14 labels classification task.

## 5.3 3D trajectory processing and comparison

Let us consider pre-segmented 3D command gesture trajectories. Our goal is to find the "optimal" way to compare two of these paths with a dissimilarity measure based on point distances. "1-dollar"-like algorithms are based on a processing chain consisting of resampling the gesture usually with a fixed number of points, centering and rotating it to match a template, scaling it to a reference size, and define a metric to estimate pre-processed gesture distances. These steps can be performed in different ways, and there is not a "right" way to perform them, as the choice is specific to the problem of interest. However, different choices result in quite different performances, and it is necessary to find the correct method for our application.

### 5.3.1 Resampling and smoothing

In order to allow an easy comparison, paths are usually resampled using a fixed number of points equally spaced in arc length or in time. While the arc length resampling seems natural to compare gesture geometry, time-based sampling would provide more information about the gesture evolution. We decided therefore to compare the approaches for our application. Furthermore, it is also interesting to evaluate the effect of changing the sampling density. Another issue to be considered is the acquisition or gestural noise. Different strategies for data simplification and smoothing have been applied, but clearly, the solution for this depends on the quality of the acquisition data and on the kind of motion involved in the gesture. It is therefore interesting to know if for our tasks a trajectory smoothing may be useful to improve the quality of the gesture difference estimation.

### 5.3.2 Spatial registration of gestures: centering, rotation and scaling

All the “dollar”-like methods proposed in the literature perform geometrical transforms of the gestures paths to align them before comparison. The 3 dollar method [36] relies on a first gesture reorientation, bounding-box based rescaling and a further tricky realignment before comparison, while Protractor 3D [37] on a more efficient Procrustes-like gesture alignment finding rotation minimizing sum of squared distances between points, that is the output distance. These choices are inherited by the corresponding 2D methods, but, as we will see in the experimental section, are not effective for our gestures, as the independent scaling on each axis distorts paths and the rotation remove the directional information of the gestures. We considered, therefore, several alternative choices to find out the best options for our retrieval task and compared them in our experiments.

To align the trajectory positions, we tested both translation to match gesture centroids and translation matching gesture origin. To scale the trajectories, we tested the bounding box method used in Protractor 3D as well as a scaling making the maximal distance of the gesture from its origin uniform and another one making gestures of uniform length. Finally, a rotation transform is usually applied to measure orientation-invariant gesture differences. We implemented and tested the option in our analysis using a Procrustean alignment method as done in [37]. But, as command gesture classes are usually characterized by the different orientation, we considered interesting to understand if the point-to-point gesture comparison could work well without matching orientation, but only translating and scaling.

### 5.3.3 Point chains distance metrics

Finally, gesture distances are estimated by comparing corresponding points. This can be done using the Euclidean distances or other metrics. As a gesture can be composed of parts that may appear shifted in the sequence, a typical approach that is applied to overcome this issue is the estimation of a warping matching better the sequence. Dynamic Time Warping (DTW) [54] is a well known algorithm often applied for this purpose. The warping estimated with DTW is expected to improve the quality of the matching, but clearly increases the computational cost. The advantages, then, may not be relevant in the case of simple trajectories not characterized by relevant features that may appear temporally shifted. For this reason, we decided to see on our dataset if the use of the warping can give advantages, estimating its effect combined with the other different path processing options.

## 5.4 Adding fingers' information

To recognize both coarse gestures characterized by hand motion only and finger gestures, where specific fingers' motion is relevant for the correct labeling using a single method, we extended our simple path comparison approach adding two more "trajectories" representing the evolution of hand orientation and hand articulation to the main hand path.

Using only directional vectors, thus being invariant to fingers' size, we defined a way to create a trajectory encoding the time evolution of hand parts' orientation and another one encoding the time evolution of finger articulation. To compare gestures, we estimate distances between the palm trajectories as well as differences in the two additional vectors, obtaining three distance measurements. The final dissimilarity value is simply obtained by multiplying the three distances.

### 5.4.1 Orientation evolution distance

Command gestures can be characterized by changes in hand orientation, as in the case of the "rotation" gestures in the SHREC'17 dataset. The evolution of the orientation of hand parts can be expressed by the trajectories of a few unit vectors characterizing it. We choose a few hand keypoints that are assumed to characterize the gestures of interest and we estimated all the unit vectors of the directions joining them. These points are index and thumb bases and tips, wrist and palm.

For each unit vector, we stored the three components, creating a time sequence of vectors. These sequences are compared using a Dynamic Time Warping approach to take into account the fact that the gesture parts are not necessarily performed with a pre-defined timing. We call the distance obtained "orientation evolution distance (OED)".

### 5.4.2 Finger vectors' distance

To characterize the finger movements related to gestures such as pinch and grab, independently on hand orientation, we considered angles between selected directions joining skeletal nodes. In particular, we considered the angles between a selection of vector directions relate to finger movements, namely: thumb base to thumb tip direction, index base to index tip direction, wrist to thumb tip direction, wrist to index tip direction, palm to thumb tip direction, palm to index tip direction, palm to wrist and normal to palm (indirectly estimated from joints). With 8 directions we have a time vector of 36 elements that can still be compared with different metrics. In this case, to cope with possible differences in gesture parts timing, we compare the vectors using Dynamic Time Warping. We call the distance obtained Finger Angles Distance (FAD).

The comparison using DTW of vectors with a large number of elements adds, in this case, a relevant computational complexity. However, as the trajectories can be encoded with a limited number of samples, as we will see in Section 5.5, this complexity is still limited. Furthermore, it could be heavily reduced using other strategies, e.g. dimensionality reduction using PCA or LDA projection.

## 5.5 Experimental Results

On the two 3D mid-air command gesture datasets described, we performed retrieval and classification tests comparing the effects of the different options on retrieval scores and classification accuracy. On the SHREC'17 dataset we could also test the novel multi-finger gesture recognition approach.

Tests have been implemented with simple Matlab code, available at the web address `https://github.com/giach68/gesturesCodes`.

| **Center.** | **Scaling** | **Rot.** | **NN** | **FT** | **ST** | **E** | **DCG** | **MAP** |
|---|---|---|---|---|---|---|---|---|
| Centroid | Length | No | **0,967** | **0,827** | **0,919** | **0,516** | **0,950** | **0,846** |
| Centroid | Length | Yes | 0,639 | 0,494 | 0,669 | 0,389 | 0,732 | 0,509 |
| Centroid | B. Box | No | 0,905 | 0,686 | 0,816 | 0,467 | 0,874 | 0,709 |
| Centroid | B. Box | Yes | 0,583 | 0,400 | 0,563 | 0,340 | 0,677 | 0,415 |
| Centroid | Max D. | No | 0,962 | 0,779 | 0,882 | 0,494 | 0,929 | 0,805 |
| Centroid | Max D. | Yes | 0,663 | 0,462 | 0,630 | 0,371 | 0,718 | 0,476 |
| Origin | Length | No | 0,950 | 0,794 | 0,905 | 0,510 | 0,939 | 0,822 |
| Origin | Length | Yes | 0,639 | 0,494 | 0,669 | 0,389 | 0,732 | 0,509 |
| Origin | B. Box | No | 0,905 | 0,686 | 0,816 | 0,467 | 0,874 | 0,709 |
| Origin | B. Box | Yes | 0,583 | 0,400 | 0,563 | 0,340 | 0,677 | 0,415 |
| Origin | Max D. | No | 0,938 | 0,771 | 0,889 | 0,503 | 0,931 | 0,804 |
| Origin | Max D. | Yes | 0,663 | 0,462 | 0,630 | 0,371 | 0,718 | 0,476 |

Table 5.1: Retrieval scores obtained on the new 26-gesture dataset using the different methods to perform geometrical registration of gestures before estimating distances. Bold fonts indicate best results.

### 5.5.1 Novel 26-gesture dataset

To evaluate the different gesture distance estimation options on the novel 26-gestures dataset we first designed a classical retrieval task. For each for each example item taken from the dataset, we "retrieve" the list of the other items sorted by the chosen distance and check if they are relevant for the query (e.g. have the same label). Results are good if the relevant items are retrieved first. Different metrics have been developed for measuring retrieval performances; in our case, as each class has a fixed number of instances (13), we could use the set of retrieval scores typically used in 3D shape retrieval works (and often used in the evaluation of Eurographics SHape REtrieval Contests). These scores are Nearest Neighbor (NN), e.g the percentage of the queries with the first retrieved item of the same class of the query, First Tier (FT), e.g. the average over the queries of the percentage of the first $K-1$ retrieved items where K is the number of instances of each class in the database, and other indicators of the ability of the method of presenting correct retrievals with high rank for all the possible queries in the database, e.g. Second Tier (ST), e-measure (E), Discounted Cumulated Gain (DCG), Mean Average Precision (MAP). Their defini-

| **Dist.** | **Sm.** | **Res.** | **NN** | **FT** | **ST** | **E** | **DCG** | **MAP** |
|---|---|---|---|---|---|---|---|---|
| Eucl. | No | Length | 0,967 | 0,827 | 0,919 | 0,516 | 0,950 | 0,846 |
| Eucl. | No | Time | 0,964 | 0,809 | 0,903 | 0,507 | 0,942 | 0,828 |
| Eucl. | Yes | Length | **0,970** | 0,825 | 0,917 | 0,512 | 0,950 | 0,844 |
| Eucl. | Yes | Time | 0,941 | 0,620 | 0,745 | 0,432 | 0,866 | 0,661 |
| Cityb. | No | Length | 0,932 | 0,627 | 0,761 | 0,440 | 0,869 | 0,669 |
| Cityb. | No | Time | 0,932 | 0,615 | 0,747 | 0,434 | 0,863 | 0,658 |
| Cityb. | Yes | Length | **0,970** | **0,835** | **0,925** | **0,519** | **0,953** | **0,853** |
| Cityb. | Yes | Time | 0,964 | 0,816 | 0,908 | 0,510 | 0,946 | 0,837 |
| DTW | No | Length | **0,970** | 0,830 | 0,921 | 0,516 | 0,953 | 0,851 |
| DTW | No | Time | 0,941 | 0,623 | 0,748 | 0,434 | 0,868 | 0,664 |
| DTW | Yes | Length | 0,932 | 0,628 | 0,764 | 0,441 | 0,871 | 0,672 |
| DTW | Yes | Time | 0,932 | 0,618 | 0,749 | 0,434 | 0,864 | 0,660 |

Table 5.2: Retrieval scores obtained on the new 26-gesture dataset using the different options for distance function, smoothing and resampling of the trajectory. Bold fonts indicate best results.

tions can be found in [30]. Another visual indicator of the retrieval performances is the precision-recall plot representing the fraction of retrieved documents that are relevant to the query (precision), versus the number of correct results divided by the number of results that should have been returned (recall).

In our first retrieval test, we fixed the gesture resampling to 40 points equally spaced in the arc length parametrization and compared the trajectories using the different options related to the spatial alignment and scaling of the gestures.

Values reported in Table 5.1 and the Precision-Recall plots in 5.5.1 show that the best option for the task is to avoid trajectory rotation and scaling to a reference trajectory length. We called the method based on this option “3 cent” gesture recognizer, as it is actually simpler than the “3 dollar” method but much more suitable for this retrieval/recognition task. “3 cent” results are quite close to the perfect retrieval of all relevant gestures among the first 12 retrieved. Note that other spatial transforms of the gestures reported in the table are similar to those used in the “3 dollar” method and Protractor 3D (bounding box scaling and rotation) and provide relevantly lower scores. The bounding box-based scaling may work well for 2D touchscreens but introduces distortions that here make results bad. The use of orientation matching removes a relevant source of information, that may be recovered only adding different orientation-related features to the descriptor.

Fixed the “3 cent” spatial registration options, it is interesting to see if different preprocessing and different point sequence comparison metrics can improve the retrieval and classification accuracies. Table 5.2 shows the retrieval scores corresponding to the different choices proposed in Section 5.3.

A local regression based smoothing [24] applied to the time sequences seems not necessary for this dataset, as the acquisition noise is not too high.

The fact that gesture resampling at a constant arc length step instead of at constant time step provides better results indicates that the geometry of the shape is here the most discriminative feature, as expected.

Differences between Euclidean and city-block distances are actually negligible. The fact that Dynamic Time Warping (with Euclidean distance) does not give any advantage is probably the most important result, as it demonstrates that this kind of gestures is more characterized by its global behavior rather than on occurrence of local features and, more important, means that we can perform gesture comparison with negligible computational effort.

On the same data, we also tested a simple classification task, similar to that reported in [37], simulating the typical example-based recognizer training.

We selected randomly 5 users as training set and applied a simple Nearest Neighbor classifier to assign labels to the remaining test set including all the gestures of the other 8 subjects. Results are shown in Tables and 5.3 and 5.4. In this case as well, the “3 cent” options provide the best results, making the potential recognizer quite accurate, and sampling at constant arc length steps gives the better accuracy. In this experiment we presented also the average time required by the NN labeling, that is actually negligible for the “3 cent” options, even if performed with non-optimized Matlab code. This makes the method suitable also to be applied with a sliding window or multiscale approaches for online gesture detection and recognition.

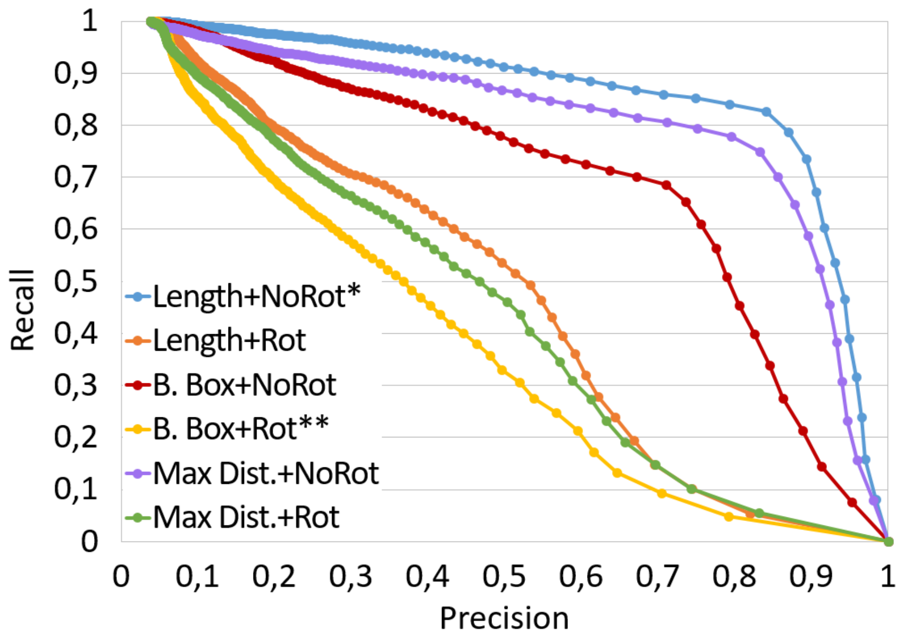


Figure 5.3: Precision vs. Recall plot for the gesture retrieval task on the 26-gestures dataset using selected combinations of spatial registration options of the gestures to be compared. (*) indicates the combination used in our “3 cent” method [18], while the combination labeled with (**) corresponds to the options used in the Protractor 3D method [37].

### 5.5.2 SHREC’ 2017 dataset

The dataset is designed for a classification task with training and test sets already separated and a number of examples per label that is not uniform. We, therefore, performed on it classification tests only, first extracting from the data only the subset of 9 “coarse” gestures, not characterized by finger movements, and then working on the full dataset.

In the first experiment, we selected test gestures from the “coarse” classes only and performed the classification tests labeling them with a Nearest Neighbor rule given the instances of the same classes in the provided training set.

As in the case presented in Section 5.5.1, we first evaluated the effect of different options for spatial transform options for gesture alignment, keeping fixed the trajectory resampling (40 equally spaced points) and trajectory distance estimation metric (Euclidean, without warping).

The results obtained (see Table 5.5) reveals, in this case as well, that the "3 cent" options are clearly those providing the best performances. Note that classification times are here higher than in the 26-gesture case due to the larger number of examples in the training set used in the nearest neighbor algorithm, but it is still quite small.

Looking at the resampling, smoothing and trajectory comparison metrics options, we found a different outcome with respect to the 26-classes dataset: in this case, in fact, resampling with fixed time step provided better results. The reason for this difference is probably the fact that, while in the first dataset users were instructed to draw a well-defined trajectory that was the only relevant information included in the gesture, in this case, some gestures (e.g. grab, pinch) are characterized by a sequence of stationary and moving parts that are better captured by a time-based resampling. The choice of the resampling method should be therefore adapted to the features of the gesture dictionary to be created.

### 5.5.3 Multi-finger gestures recognition

The fact that the "3 cent" comparison is actually effective in gesture difference characterization appears manifest by looking at its performances on the full SHREC'17 dataset (14 labels). The average classification accuracy is 73.6% using the palm trajectory only, not too far from those obtained with a complex encoding of multiple finger joints (see Table 5.7).

Of course, looking at the confusion matrix (Figure 5.4), it is possible to see that,

| **Center.** | **Scaling** | **Rot.** | **NN** | **Time (s.)** |
|---|---|---|---|---|
| Centroid | Length | No | **0,969** | 0,0007 |
| Centroid | Length | Yes | 0,638 | 0,0212 |
| Centroid | B. Box | No | 0,869 | 0,0006 |
| Centroid | B. Box | Yes | 0,515 | 0,0215 |
| Centroid | Max D. | No | 0,931 | 0,0006 |
| Centroid | Max D. | Yes | 0,600 | 0,0216 |
| Origin | Length | No | 0,962 | 0,0006 |
| Origin | Length | Yes | 0,638 | 0,0214 |
| Origin | B. Box | No | 0,869 | 0,0006 |
| Origin | B. Box | Yes | 0,515 | 0,0215 |
| Origin | Max D. | No | 0,962 | 0,0006 |
| Origin | Max D. | Yes | 0,600 | 0,0214 |

Table 5.3: Classification results on the 26 gestures dataset using the "3 cent" options and the other tested options for centering, scaling and rotation. Bold fonts indicate best results.

obviously, the palm trajectory is not effective for some "fine" gestures (classes 3-6), but the method actually outperforms the contest's winner for 5 gesture classes (swipe right, swipe left, swipe x, swipe +, shake), and works well also for a gesture considered "fine" as grab.

Adding the two-finger gestures characterization methods introduced in Section 5.4, e.g. orientation evolution distance (OED) and finger angles distance (FAD), we can obtain an average classification accuracy higher than those obtained by contest organizers and other participants, without using complex feature encoding, but still relying on simple trajectory distances (Table 5.7). The confusion matrix shows a good recognition accuracy for all the classes (Figure 5.5). The other methods compared in the contest were based on full hand skeleton encoding with Fisher Vectors plus directional information and linear Support Vector Machine based supervised labelling (SDG [26]), Joint Angles and HOG2 features using also depth map information (JA+HOG2[50]), CNN applied to depth sequences (Keyframes CNN, [58]), Histograms of oriented 4D normals (HON 4D, [51]) and another skeleton based method analyzing trajectories on the Riemannian Manifold (TRM, [27]). The fact that a straightforward method as the one we propose outperforms all these tech-

| **Distance** | **Smooth.** | **Res.** | **NN** | **Time (s.)** |
|---|---|---|---|---|
| Euclidean | No | Length | **0,969** | 0,0007 |
| Euclidean | No | Time | 0,938 | 0,0006 |
| Euclidean | Yes | Length | **0,969** | 0,0006 |
| Euclidean | Yes | Time | 0,938 | 0,0006 |
| City Block | No | Length | 0,962 | 0,0006 |
| City Block | No | Time | 0,915 | 0,0006 |
| City Block | Yes | Length | **0,969** | 0,0006 |
| City Block | Yes | Time | 0,915 | 0,0006 |
| DTW | No | Length | **0,969** | 0,0077 |
| DTW | No | Time | 0,931 | 0,0064 |
| DTW | Yes | Length | **0,969** | 0,0063 |
| DTW | Yes | Time | 0,931 | 0,0063 |

Table 5.4: Classification results on the 26 gestures dataset using "3 cent" spatial transform options and different choices for distance metric, smoothing and trajectory resampling. Bold fonts indicate best results.

niques, probably indicates that for short gestures of this kind it is not necessary to rely on advanced feature encoding methods that are successful on the long and complex skeletal paths processed in action recognition applications, and that the small amount of training data makes the use of neural networks not necessarily beneficial. This outcome can be exploited to create effective ways to design custom interfaces for different tasks without the need for training advanced classifiers, but just giving a few examples of the desired gestures.

The only weakness of the method appears the use of the DTW approach adding computational complexity, but, as we will show in Section 5.5.4, this issue is strongly reduced by trajectory subsampling making the method suitable for online recognition even without algorithms' optimization.

### 5.5.4 Subsampling and Recognition of Incomplete Gestures

As shown in the 2D case [63], the dollar recognizer family is rather robust against point subsampling. Figures 5.6 and 5.7 show that the classification accuracies on the 26-gestures and on the 9-gestures reduced SHREC'17 tests are practically unchanged

| **Center.** | **Scaling** | **Rot.** | **NN** | **Time (s.)** |
|---|---|---|---|---|
| Centroid | Length | No | **0,904** | 0,0075 |
| Centroid | Length | Yes | 0,522 | 0,2073 |
| Centroid | B. Box | No | 0,764 | 0,0077 |
| Centroid | B. Box | Yes | 0,419 | 0,2089 |
| Centroid | Max D. | No | 0,863 | 0,0079 |
| Centroid | Max D. | Yes | 0,467 | 0,2096 |
| Origin | Length | No | 0,876 | 0,0076 |
| Origin | Length | Yes | 0,522 | 0,2088 |
| Origin | B. Box | No | 0,764 | 0,0076 |
| Origin | B. Box | Yes | 0,419 | 0,2087 |
| Origin | Max D. | No | 0,840 | 0,0076 |
| Origin | Max D. | Yes | 0,467 | 0,2086 |

Table 5.5: Classification results on the 9 "coarse" gestures of the SHREC'17 dataset using "3 cent" normalization and different options for distance metric, smoothing and trajectory resampling. Bold fonts indicate best results.

reducing the sampling up to 10 equally spaced points and are still reasonable even using only three points.

This behavior is maintained also for finger-based gestures, where the accuracy is practically unchanged and this fact is rather important as the subsampling does heavily reduce the recognition time, that would be, in this case, higher due to the use of the DTW distance. With 10 points the average Nearest Neighbor classification time is, in fact, 178 ms. This speed-up, that can also be combined with many other optimizations (use of an efficient compiled code, NN algorithm simplification, etc.) makes the method suitable for online applications.

An even more interesting fact is that, if we try to only match the first part of the performed gesture, the recognition performances are still quite close to the ones obtained with the full trajectories. This is demonstrated in Figure 5.8: the classification accuracy on the 26-gesture dataset is practically unchanged if the match is done on the first 60% of the gesture length (see Figure 5.6). The fact we can predict the gesture completion from a very partial execution can be exploited to improve the efficiency and decrease the latency of systems' feedback and may enable the

| **Distance** | **Smooth.** | **Res.** | **NN** | **Time (s.)** |
|---|---|---|---|---|
| Euclidean | No | Length | 0,904 | 0,0075 |
| Euclidean | No | Time | **0,925** | 0,0076 |
| Euclidean | Yes | Length | 0,915 | 0,0076 |
| Euclidean | Yes | Time | 0,909 | 0,0077 |
| City Block | No | Length | 0,902 | 0,0077 |
| City Block | No | Time | 0,922 | 0,0077 |
| City Block | Yes | Length | 0,911 | 0,0077 |
| City Block | Yes | Time | 0,899 | 0,0076 |
| DTW | No | Length | 0,901 | 0,0616 |
| DTW | No | Time | **0,925** | 0,0613 |
| DTW | Yes | Length | 0,904 | 0,0615 |
| DTW | Yes | Time | 0,904 | 0,0614 |

Table 5.6: Classification results on the 9 “coarse” gestures of the SHREC’17 dataset using “3 cent” normalization and different options for distance metric, smoothing and trajectory resampling. Bold fonts indicate best results.

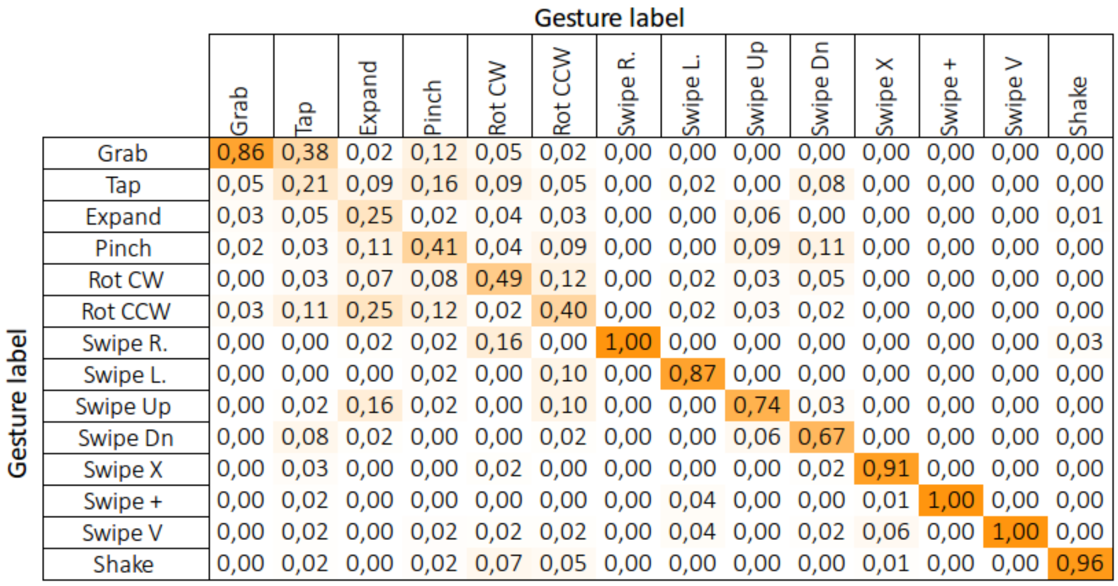
| | **Accuracy** |
|---|---|
| 3 cent | 73.6 |
| 3 cent + OED | 86.1 |
| 3 cent + FAD | 81.2 |
| 3 cent + OED+FAD | **89.52** |
| SDG [26] | 88.24 |
| JA+HOG2 [50] | 83.85 |
| Keyframes CNN [58] | 82.90 |
| TRM [27] | 79.61 |
| HON 4D [51] | 78.53 |

Table 5.7: Classification accuracy with 3 cent and 3 cent plus orientation and fingers descriptors compared with the state of the art on the SHREC’17 14-classes classification task. Bold fonts indicate best results.

design of visual feedback mechanisms suggesting gesture completion from already performed trajectories, as suggested in [29].

The robustness to partiality is lower in the case of finger gestures (Figure 5.9), even if, also in this case, we can reasonably predict the correct gesture with only $60-80\%$ of the trajectory.

This is reasonable as many gestures are characterized by different finger movements executed near the end and cannot be determined too early.


Gesture label

| Gesture label | Grab | Tap | Expand | Pinch | Rot CW | Rot CCW | Swipe R. | Swipe L. | Swipe Up | Swipe Dn | Swipe X | Swipe + | Swipe V | Shake |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Grab | 0,86 | 0,38 | 0,02 | 0,12 | 0,05 | 0,02 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| Tap | 0,05 | 0,21 | 0,09 | 0,16 | 0,09 | 0,05 | 0,00 | 0,02 | 0,00 | 0,08 | 0,00 | 0,00 | 0,00 | 0,00 |
| Expand | 0,03 | 0,05 | 0,25 | 0,02 | 0,04 | 0,03 | 0,00 | 0,00 | 0,06 | 0,00 | 0,00 | 0,00 | 0,00 | 0,01 |
| Pinch | 0,02 | 0,03 | 0,11 | 0,41 | 0,04 | 0,09 | 0,00 | 0,00 | 0,09 | 0,11 | 0,00 | 0,00 | 0,00 | 0,00 |
| Rot CW | 0,00 | 0,03 | 0,07 | 0,08 | 0,49 | 0,12 | 0,00 | 0,02 | 0,03 | 0,05 | 0,00 | 0,00 | 0,00 | 0,00 |
| Rot CCW | 0,03 | 0,11 | 0,25 | 0,12 | 0,02 | 0,40 | 0,00 | 0,02 | 0,03 | 0,02 | 0,00 | 0,00 | 0,00 | 0,00 |
| Swipe R. | 0,00 | 0,00 | 0,02 | 0,02 | 0,16 | 0,00 | 1,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,03 |
| Swipe L. | 0,00 | 0,00 | 0,00 | 0,02 | 0,00 | 0,10 | 0,00 | 0,87 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| Swipe Up | 0,00 | 0,02 | 0,16 | 0,02 | 0,00 | 0,10 | 0,00 | 0,00 | 0,74 | 0,03 | 0,00 | 0,00 | 0,00 | 0,00 |
| Swipe Dn | 0,00 | 0,08 | 0,02 | 0,00 | 0,00 | 0,02 | 0,00 | 0,00 | 0,06 | 0,67 | 0,00 | 0,00 | 0,00 | 0,00 |
| Swipe X | 0,00 | 0,03 | 0,00 | 0,00 | 0,02 | 0,00 | 0,00 | 0,00 | 0,00 | 0,02 | 0,91 | 0,00 | 0,00 | 0,00 |
| Swipe + | 0,00 | 0,02 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,04 | 0,00 | 0,00 | 0,01 | 1,00 | 0,00 | 0,00 |
| Swipe V | 0,00 | 0,02 | 0,00 | 0,02 | 0,02 | 0,02 | 0,00 | 0,04 | 0,00 | 0,02 | 0,06 | 0,00 | 1,00 | 0,00 |
| Shake | 0,00 | 0,02 | 0,00 | 0,02 | 0,07 | 0,05 | 0,00 | 0,00 | 0,00 | 0,00 | 0,01 | 0,00 | 0,00 | 0,96 |

Figure 5.4: Confusion matrix for the 14-classes SHREC'17 dataset using "3 cent" Nearest Neighbor classification.

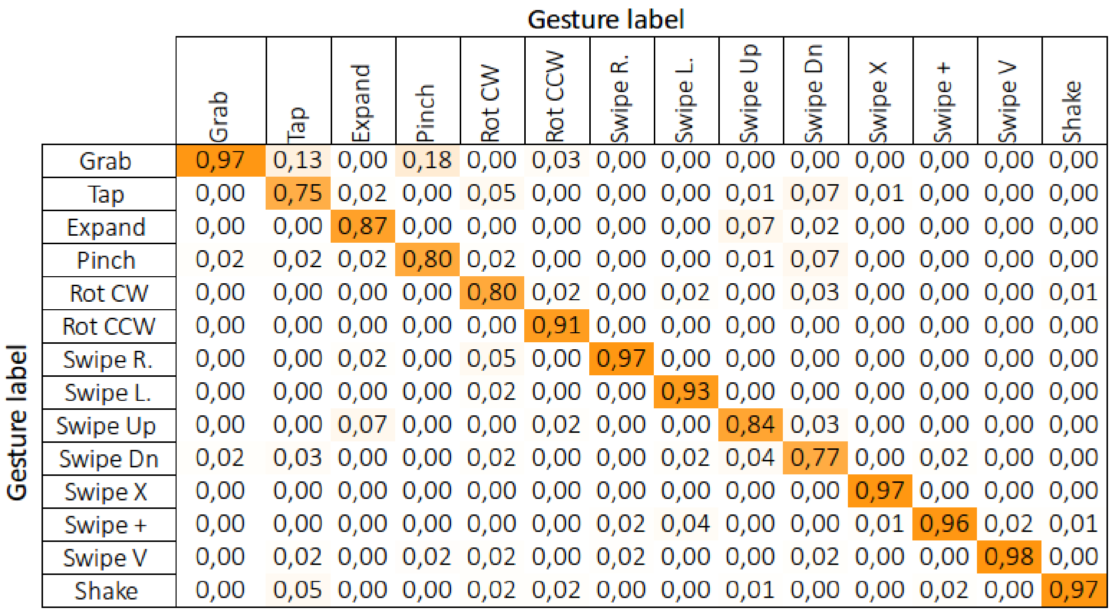
## 5.6 Discussion

Our experiments demonstrate that simple trajectory distances can be effectively applied for command gesture recognition tasks.

Avoiding orientation matching and using length based rescaling and Euclidean or city-block distances of sampled points, it is possible to obtain a reliable gesture distance measure that is effective for Nearest Neighbor based gesture trajectory classification. Using a similar approach to compare finger movement it is possible to obtain good performances in the classification of command gestures where both hand trajectory and finger movements determine the semantic labeling.

This simple approach can be extremely powerful for interface design, as it allows the creation of design tools where novel interface gestures can be added immediately giving a limited number of examples without the need of further training procedures.

Our experimental results provide also some guidelines for the design of these tools, e.g.:



| Gesture label | Grab | Tap | Expand | Pinch | Rot CW | Rot CCW | Swipe R. | Swipe L. | Swipe Up | Swipe Dn | Swipe X | Swipe + | Swipe V | Shake |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Grab | 0,97 | 0,13 | 0,00 | 0,18 | 0,00 | 0,03 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| Tap | 0,00 | 0,75 | 0,02 | 0,00 | 0,05 | 0,00 | 0,00 | 0,00 | 0,01 | 0,07 | 0,01 | 0,00 | 0,00 | 0,00 |
| Expand | 0,00 | 0,00 | 0,87 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,07 | 0,02 | 0,00 | 0,00 | 0,00 | 0,00 |
| Pinch | 0,02 | 0,02 | 0,02 | 0,80 | 0,02 | 0,00 | 0,00 | 0,00 | 0,01 | 0,07 | 0,00 | 0,00 | 0,00 | 0,00 |
| Rot CW | 0,00 | 0,00 | 0,00 | 0,00 | 0,80 | 0,02 | 0,00 | 0,02 | 0,00 | 0,03 | 0,00 | 0,00 | 0,00 | 0,01 |
| Rot CCW | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,91 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| Swipe R. | 0,00 | 0,00 | 0,02 | 0,00 | 0,05 | 0,00 | 0,97 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| Swipe L. | 0,00 | 0,00 | 0,00 | 0,00 | 0,02 | 0,00 | 0,00 | 0,93 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| Swipe Up | 0,00 | 0,00 | 0,07 | 0,00 | 0,00 | 0,02 | 0,00 | 0,00 | 0,84 | 0,03 | 0,00 | 0,00 | 0,00 | 0,00 |
| Swipe Dn | 0,02 | 0,03 | 0,00 | 0,00 | 0,02 | 0,00 | 0,00 | 0,02 | 0,04 | 0,77 | 0,00 | 0,02 | 0,00 | 0,00 |
| Swipe X | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,97 | 0,00 | 0,00 | 0,00 |
| Swipe + | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 | 0,02 | 0,04 | 0,00 | 0,00 | 0,01 | 0,96 | 0,02 | 0,01 |
| Swipe V | 0,00 | 0,02 | 0,00 | 0,02 | 0,02 | 0,00 | 0,02 | 0,00 | 0,00 | 0,02 | 0,00 | 0,00 | 0,98 | 0,00 |
| Shake | 0,00 | 0,05 | 0,00 | 0,00 | 0,02 | 0,02 | 0,00 | 0,00 | 0,01 | 0,00 | 0,00 | 0,02 | 0,00 | 0,97 |

Figure 5.5: Confusion matrix for the 14-classes SHREC'17 dataset using 3 cent + OED + FAD Nearest Neighbor classification.

- If we want to recognize a set of gestures characterized by hand motion only, we can rely on arc length-based resampling and Euclidean or city-block distance.

- If we want to recognize gestures with coordination of hand motion and finger gestures, it is probably better to resample trajectories based on time and use the finger related features compared using DTW. In order to speed up DTW estimation, we can subsample the trajectories to a quite low number of samples still having good results.

The major issue left unsolved in order to obtain an efficient and reliable online gesture recognizer is related to gesture segmentation or detection in tracked sequences. In some potential real-world applications, it may be not necessary, as it is also possible that the user could signal the gesture start clicking on tracked devices or with a different procedure. In any case, the efficiency of the algorithm suggests that approaches similar to those already proposed for the task [64, 70] could allow the creation of a reliable online gesture recognizer.

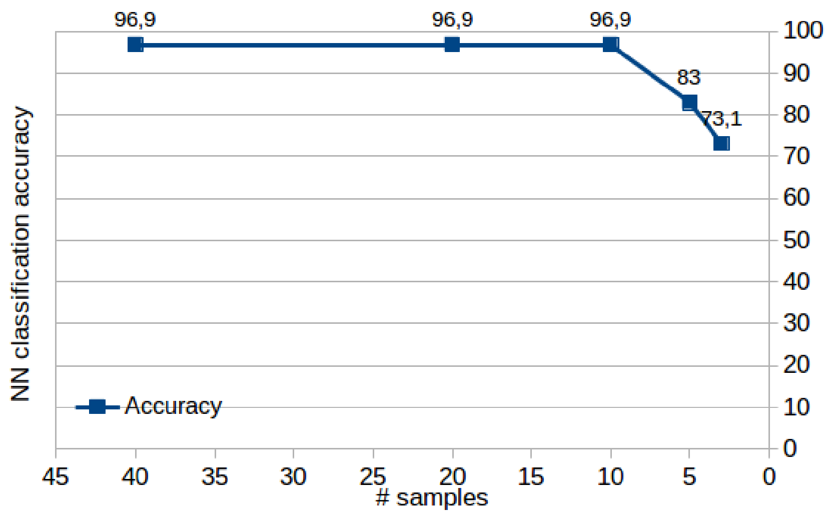


Figure 5.6: The accuracy of Nearest Neighbor classification on the 26 gestures dataset with the “3 cent” options is practically unchanged when trajectories are downsampled to 10 points only.

The main limitation of the method is related to the fact that the simple point-to-point trajectory comparison works well with short gestures with a small number of features, but would not probably scale well to long single or multiple trajectories, as required by different application domains (e.g. action recognition, sign language decoding).

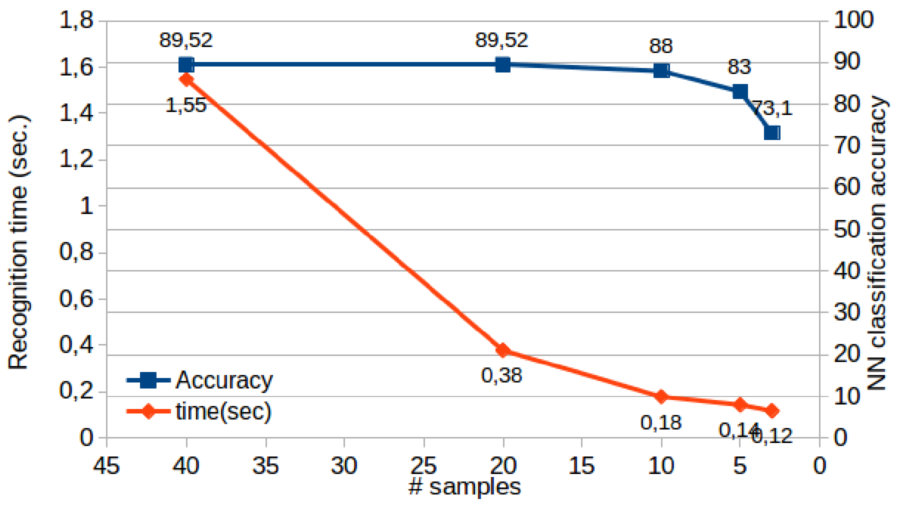


Figure 5.7: The accuracy of Nearest Neighbor classification of the 14 gestures from the SHREC'17 database with the "3 cent + OED + FAD" options is practically unchanged when trajectories are downsampled to 10 points only. The time required for the labeling is strongly decreased with the subsampling due to the use of the DTW approach for finger-derived trajectories' comparison (red line).

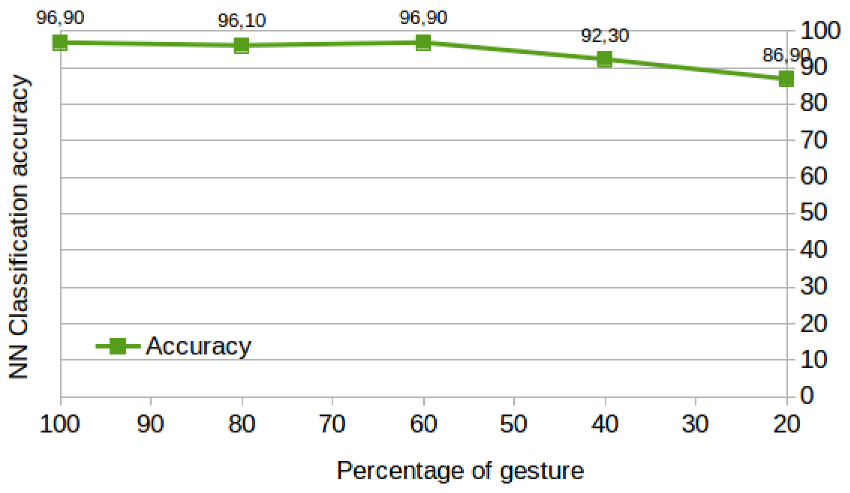


Figure 5.8: Accuracy in 3 cent/NN classification on the 26 gestures database is practically unchanged if gestures are truncated at 60% of the trajectory length and it's only 10 % lower, compared to the full gesture, after just 20%.

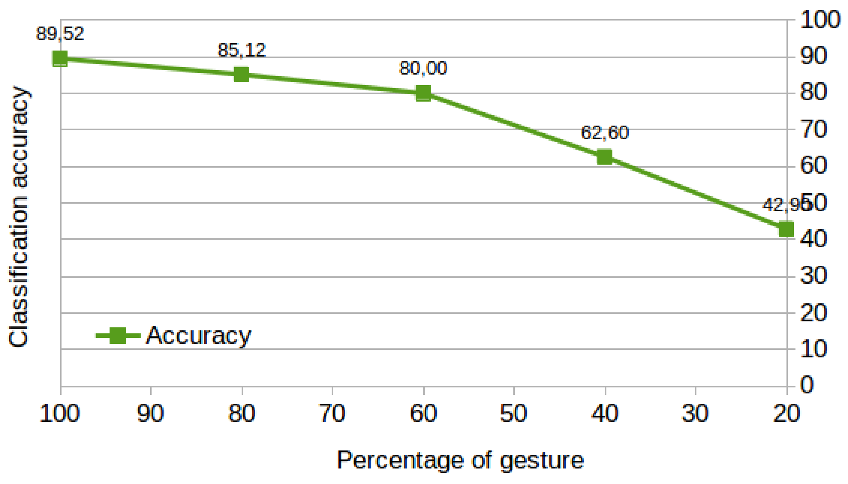


Figure 5.9: Accuracy in 3 cent/NN classification on the 14 gestures SHREC'17 database decreases slowly and it still high after 60%, but drops rapidly with less than half gesture.

# Chapter 6

# Conclusion

## 6.1 Overall Results Discussion

### 6.1.1 Manipulation Results

We started our research on manipulation techniques with the idea of identifying and tackling critical issues emerging from both our evaluation experiments and the literature. Our first evaluation results (Section 2.2) complied with our first two Hypothesis $H_1$ and $H_2$, in fact, the Handlebar and the Finger-Point techniques implemented in the test had the highest scores, both being the only ones in the experiment to exploit positional tracking to enable the rotation action and both differentiating from the others in the rotation sub-task performance. Surely the nature of the test wasn't completely unbiased considering the task. Users were in fact encouraged to split the docking operation in two separate actions (i.e. translation first and then rotation) due to the visual feedback confirming sub-task completion being provided in a specific order. By comparing the performances of the two sub-tasks, however, the difference between rotation and translation, for the techniques mapping 1:1 the hand orientation to the object orientation, was unquestionable.

Despite not being an exhaustive proof, the overall outcome of the experiment was, nonetheless, a strong enough hint to believe the Hypothesis $H_1$ and $H_2$ were valid. With the next works we tried to design techniques starting from $H_1$ and $H_2$ to confirm our previous results and aimed to explore the possibility of a viable single-handed manipulation technique and formulated our Hypothesis $H_3$. While the Finger-Point behaved really well in our tests it occurred to us that the technique could be limited in scenarios with multiple objects since the gesture-recognition based activation of the rotation (i.e. the pointing finger) doesn't have a reliable mechanism of selection. The Knob was designed to have a more direct manipulation feeling but the results were not exceptional in our experiment, overall performing worse than the reference technique (i.e. the Handlebar). Nonetheless, the work provided enough insights, on the one-handed manipulation issue and the goals we set when designing the Knob, to point us in the direction of a new technique: the Smart Pin. With this novel technique we introduced a widget with the idea of streamlining all the manipulation action with the same grab-release mechanism at different spots on the pin (i.e. the colored caps) while being consistent with all our previous findings that gave promising results. In this case, the $H_3$ was verified, in fact, while in terms of sheer performance the Smart Pin and the Handlebar were even, the UX was significantly positively leaning in favor of the Smart Pin. We do believe that the same results or even better ones can be achieved by using the design choices that we used to develop this technique as a basis for new ones.

### 6.1.2 Gesture Recognition Results

The idea behind this branch of our research was to eventually provide a viable addition to manipulation techniques to possibly enhance the interaction potential in IVEs and prove our Hypothesis $H_5$. Therefore we first formulated Hypothesis $H_4$ as a non-strict but certainly convenient requirement. The results of our experiments reported in our work described in 5 strongly proved that good accuracy results in gesture recognition can be achieved, computationally speaking, relatively easily. The 3cent method gave complying results on two datasets featuring different gestures and exceeded our expectation by performing at the same level even with partial information. While to this point we couldn't verify $H_5$, this last result definitely looks very promising. In fact, the concern about integrating such a system in a real-time application is the potential excessive time needed to recognize a gesture (e.g. the full length of the gesture motion) that could compromise the UX. By being able to feed the algorithm only partial information and still achieve a high score of accuracy we are confident that this issue can be overcome.

## 6.2 Future Work

About object manipulation, there is a number of things that we are looking forward to include in our future research. Given the results of the Smart Pin we are positive that a new manipulation technique could take the single-handed kind even further by surpassing successful bimanual techniques, such as the Handlebar, also in performance. Another thing that we are currently working on is the improvement of the "release accuracy" partially described in subsection 3.1. Besides possible visual feedbacks to help the user compensate for the displacement error we are looking into implementing correction algorithms with different strategies, ranging from simple heuristics to regression methods, in order to automate the position adjustment as seamlessly as possible.
As stated in 6.1.2 our hypothesis $H_5$ was not yet verified. We are looking forward to do so in our next works. One of the long-term goals of this research is to validate all our theoretical findings on an application featuring both single-handed mid-air manipulation techniques and gesture recognition methods to enable gestural commands. About this, we are currently developing a virtual environment editor

featuring manipulation and editing options on 3D models, menus and soon to be implemented navigation techniques. The application is being built to easily provide tools for performance comparisons between interaction techniques but also hardware setups.

About the gesture recognition, we are in the early stages of development of a new algorithm based on neural networks. Our goal is to build an efficient and robust recognition system to implement a reliable gestural interface, able to detect gestures in real time from 3D hands trajectories. This interface will be later implemented and tested in some real Augmented Reality applications.

# Bibliography

[1] Vamsi Kiran Adhikarla, Jaka Sodnik, Peter Szolgay, and Grega Jakus. Exploring direct 3d interaction for full horizontal parallax light field displays using leap motion controller. *Sensors*, 15(4):8642–8663, 2015.

[2] Roland Aigner, Daniel Wigdor, Hrvoje Benko, Michael Haller, David Lindbauer, Alexandra Ion, Shengdong Zhao, and JTKV Koh. Understanding mid-air hand gestures: A study of human preferences in usage of gesture types for hci. *Microsoft Research TechReport MSR-TR-2012-111*, 2012.

[3] Panagiotis Apostolellis, Brennon Bortz, Mi Peng, Nicholas Polys, and Andy Hoegh. Exploring the integrality and separability of the leap motion controller for direct manipulation 3d interaction. In *3D User Interfaces (3DUI), 2014 IEEE Symposium on*, pages 153–154. IEEE, 2014.

[4] Daniel Bachmann, Frank Weichert, and Gerhard Rinkenauer. Evaluation of the leap motion controller as a new contact-free pointing device. *Sensors*, 15(1):214–233, 2014.

[5] Fabio Bettio, Andrea Giachetti, Enrico Gobbetti, Fabio Marton, and Giovanni Pintore. A practical vision based approach to unencumbered direct spatial manipulation in virtual worlds. In *Eurographics Italian Chapter Conference*, pages 145–150, 2007.

[6] B. Bossavit, A. Marzo, O. Ardaiz, L. D. De Cerio, and A. Pina. Design choices and their implications for 3d mid-air manipulation techniques. *Presence*, 23(4):377–392, Nov 2014.

[7] Yves Boussemart, François Rioux, Frank Rudzicz, Michael Wozniewski, and Jeremy R Cooperstock. A framework for 3d visualisation and manipulation in an immersive space using an untethered bimanual gestural interface. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 162–165. ACM, 2004.

[8] Doug A Bowman and Larry F Hodges. An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In *Proceedings of the 1997 symposium on Interactive 3D graphics*, pages 35–ff. ACM, 1997.

[9] Doug A Bowman, Ryan P McMahan, and Eric D Ragan. Questioning naturalism in 3d user interfaces. *Communications of the ACM*, 55(9):78–88, 2012.

[10] Marcio C Cabral, Carlos H Morimoto, and Marcelo K Zuffo. On the usability of gesture interfaces in virtual reality environments. In *Proceedings of the 2005 Latin American conference on Human-computer interaction*, pages 100–108. ACM, 2005.

[11] Fabio M Caputo, Marco Emporio, and Andrea Giachetti. The smart pin: an effective tool for object manipulation in immersive virtual reality environments. *Computers & Graphics*, 2018.

[12] Fabio M Caputo, Daniel Mendes, Alessia Bonetti, Giacomo Saletti, and Andrea Giachetti. Smart choices for deviceless and device-based manipulation in immersive virtual reality. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 1–2. IEEE, 2018.

[13] Fabio M Caputo, Pietro Prebianca, Alessandro Carcangiu, Lucio D Spano, and Andrea Giachetti. Comparing 3d trajectories for simple mid-air gesture recognition. *Computers & Graphics*, 73:17–25, 2018.

[14] Fabio Marco Caputo, Irina Mihaela Ciortan, Davide Corsi, Marco De Stefani, and Andrea Giachetti. Gestural interaction and navigation techniques for virtual museum experiences. In *AVI* CH*, pages 32–35, 2016.

[15] Fabio Marco Caputo, Marco Emporio, and Andrea Giachetti. The smart pin: a novel object manipulation technique for immersive virtual environments. In *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology*, page 89. ACM, 2017.

[16] Fabio Marco Caputo and Andrea Giachetti. Evaluation of basic object manipulation modes for low-cost immersive virtual reality. In *Proceedings of the 11th Biannual Conference on Italian SIGCHI Chapter*, pages 74–77. ACM, 2015.

[17] Fabio Marco Caputo, Pietro Prebianca, Alessandro Carcangiu, Lucio D Spano, and Andrea Giachetti. A 3 cent recognizer: Simple and effective retrieval and classification of mid-air gestures from single 3d traces. *Smart Tools and Apps for Graphics. Eurographics Association*, 2017.

[18] Fabio Marco Caputo, Pietro Prebianca, Alessandro Carcangiu, Lucio D. Spano, and Andrea Giachetti. A 3 Cent Recognizer: Simple and Effective Retrieval and Classification of Mid-air Gestures from Single 3D Traces. In Andrea Giachetti, Paolo Pingi, and Filippo Stanco, editors, *Smart Tools and Apps for Graphics - Eurographics Italian Chapter Conference*. The Eurographics Association, 2017.

[19] FM Caputo, M Emporio, and A Giachetti. Single-handed vs. two handed manipulation in virtual reality: A novel metaphor and experimental comparisons. *Smart Tools and Apps for Graphics. Eurographics Association*, 2017.

[20] Emmanuelle Chapoulie. *Gestures and direct manipulation for immersive virtual reality*. PhD thesis, Université Nice Sophia Antipolis, 2014.

[21] Michael Chen, S Joy Mountford, and Abigail Sellen. A study in interactive 3-d rotation using 2-d control devices. *ACM SIGGRAPH Computer Graphics*, 22(4):121–129, 1988.

[22] Hong Cheng, Lu Yang, and Zicheng Liu. Survey on 3d hand gesture recognition. *IEEE Trans. Cir. and Sys. for Video Technol.*, 26(9):1659–1673, September 2016.

[23] Isabelle D Cherney. Mom, let me play more computer games: They improve my mental rotation skills. *Sex Roles*, 59(11-12):776–786, 2008.

[24] William S Cleveland and Susan J Devlin. Locally weighted regression: an approach to regression analysis by local fitting. *Journal of the American statistical association*, 83(403):596–610, 1988.

[25] Clement Creusot, Nick Pears, and Jim Austin. A machine-learning approach to keypoint detection and landmarking on 3d meshes. *International journal of computer vision*, 102(1-3):146–179, 2013.

[26] Quentin De Smedt, Hazem Wannous, and Jean-Philippe Vandeborre. Skeleton-based dynamic hand gesture recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–9, 2016.

[27] Maxime Devanne, Hazem Wannous, Stefano Berretti, Pietro Pala, Mohamed Daoudi, and Alberto Del Bimbo. 3-d human action recognition by shape analysis of motion trajectories on riemannian manifold. *IEEE transactions on cybernetics*, 45(7):1340–1352, 2015.

[28] A Giachetti, FM Caputo, A Carcangiu, R Scateni, and LD Spano. Shape retrieval and 3d gestural interaction: position paper. In *Proceedings of the*

*Eurographics 2016 Workshop on 3D Object Retrieval*, pages 1–4. Eurographics Association, 2016.

[29] Andrea Giachetti, Fabio Marco Caputo, Alessandro Carcangiu, Riccardo Scateni, and Lucio Davide Spano. Shape Retrieval and 3D Gestural Interaction. In A. Ferreira, A. Giachetti, and D. Giorgi, editors, *Eurographics Workshop on 3D Object Retrieval*. The Eurographics Association, 2016.

[30] Afzal Godil, Zhouhui Lian, Helin Dutagaci, Rui Fang, TP Vanamali, and Chun Pan Cheung. Benchmarks, performance evaluation and contests for 3d shape retrieval. In *Proceedings of the 10th Performance Metrics for Intelligent Systems Workshop*, pages 42–47. ACM, 2010.

[31] Georg Hackenberg, Rod McCall, and Wolfgang Broll. Lightweight palm and finger tracking for real-time 3d gesture control. In *Virtual Reality Conference (VR), 2011 IEEE*, pages 19–26. IEEE, 2011.

[32] Fei Han, Brian Reily, William Hoff, and Hao Zhang. Space-time representation of people based on 3d skeletal data: A review. *Computer Vision and Image Understanding*, 158:85–105, 2017.

[33] Chris Hand. A survey of 3d interaction techniques. In *Computer graphics forum*, volume 16, pages 269–281. Wiley Online Library, 1997.

[34] Jacek Jankowski and Martin Hachet. Advances in interaction with 3d environments. In *Computer Graphics Forum*, volume 34, pages 152–190. Wiley Online Library, 2015.

[35] Taeho Kim and Jinah Park. 3d object manipulation using virtual handle with grabbing metaphor. 2014.

[36] Sven Kratz and Michael Rohs. A $ 3 gesture recognizer: simple gesture recognition for devices equipped with 3d acceleration sensors. In *Proceedings of*

*the 15th international conference on Intelligent user interfaces*, pages 341–344. ACM, 2010.

[37] Sven Kratz and Michael Rohs. Protractor3d: a closed-form solution to rotation-invariant 3d gestures. In *Proceedings of the 16th international conference on Intelligent user interfaces*, pages 371–374. ACM, 2011.

[38] Steven M. LaValle. *Virtual Reality.* Cambridge University Press, 2017.

[39] Joseph J LaViola Jr, Ernst Kruijff, Ryan P McMahan, Doug Bowman, and Ivan P Poupyrev. *3D user interfaces: Theory and practice.* Addison-Wesley Professional, 2017.

[40] Guillaume Lavoué. Combination of bag-of-words descriptors for robust partial shape retrieval. *The Visual Computer*, 28(9):931–942, 2012.

[41] Rung-Huei Liang and Ming Ouhyoung. A real-time continuous gesture recognition system for sign language. In *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on*, pages 558–567. IEEE, 1998.

[42] D Mendes, FM Caputo, A Giachetti, A Ferreira, and J Jorge. A survey on 3d virtual object manipulation: From the desktop to immersive virtual environments. In *Computer Graphics Forum.* Wiley Online Library, 2018.

[43] Daniel Mendes, Fernando Fonseca, Bilza Araujo, Andre Ferreira, and Joaquim Jorge. Mid-air interactions above stereoscopic interactive tables. In *3D User Interfaces (3DUI), 2014 IEEE Symposium on*, pages 3–10. IEEE, 2014.

[44] Daniel Mendes, Maurício Sousa, Rodrigo Lorena, Alfredo Ferreira, and Joaquim Jorge. Using custom transformation axes for mid-air manipulation of 3d virtual objects. In *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology*, VRST '17, pages 27:1–27:8, New York, NY, USA, 2017. ACM.

[45] Mark R Mine, Frederick P Brooks Jr, and Carlo H Sequin. Moving objects in space: exploiting proprioception in virtual-environment interaction. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 19–26. ACM Press/Addison-Wesley Publishing Co., 1997.

[46] S. Mitra and T. Acharya. Gesture recognition: A survey. *IEEE Trans. Systems, Man, and Cybernetics, Part C*, 37(3):311–324, 2007.

[47] Mahdi Nabiyouni, Bireswar Laha, and Doug A Bowman. Poster: Designing effective travel techniques with bare-hand interaction. In *3D User Interfaces (3DUI), 2014 IEEE Symposium on*, pages 139–140. IEEE, 2014.

[48] Michael Nielsen, Moritz Störring, Thomas B Moeslund, and Erik Granum. A procedure for developing intuitive and ergonomic gesture interfaces for hci. In *Gesture-Based Communication in Human-Computer Interaction*, pages 409–420. Springer, 2003.

[49] Chris North, Tim Dwyer, Bongshin Lee, Danyel Fisher, Petra Isenberg, George Robertson, and Kori Inkpen. Understanding multi-touch manipulation for surface computing. In *IFIP Conference on Human-Computer Interaction*, pages 236–249. Springer, 2009.

[50] Eshed Ohn-Bar and Mohan Trivedi. Joint angles similarities and hog2 for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 465–470, 2013.

[51] Omar Oreifej and Zicheng Liu. Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 716–723, 2013.

[52] Vladimir I Pavlovic, Rajeev Sharma, and Thomas S. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Transactions on pattern analysis and machine intelligence*, 19(7):677–695, 1997.

[53] Siddharth S. Rautaray and Anupam Agrawal. Vision based hand gesture recognition for human computer interaction: a survey. *Artif. Intell. Rev.*, 43(1):1–54, 2015.

[54] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1):43–49, 1978.

[55] Donald J Schuirmann. A comparison of the two one-sided tests procedure and the power approach for assessing the equivalence of average bioavailability. *Journal of pharmacokinetics and biopharmaceutics*, 15(6):657–680, 1987.

[56] Zhanpeng Shao and Youfu Li. Integral invariants for space motion trajectory matching and recognition. *Pattern Recognition*, 48(8):2418–2432, 2015.

[57] Ken Shoemake. Arcball: a user interface for specifying three-dimensional orientation using a mouse. In *Graphics Interface*, volume 92, pages 151–156, 1992.

[58] Quentin De Smedt, Hazem Wannous, Jean-Philippe Vandeborre, J. Guerry, B. Le Saux, and D. Filliat. 3D Hand Gesture Recognition Using a Depth and Skeletal Dataset. In Ioannis Pratikakis, Florent Dupont, and Maks Ovsjanikov, editors, *Eurographics Workshop on 3D Object Retrieval*. The Eurographics Association, 2017.

[59] Peng Song, Wooi Boon Goh, William Hutama, Chi-Wing Fu, and Xiaopei Liu. A handle bar metaphor for virtual object manipulation with mid-air interaction. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, CHI '12, pages 1297–1306, New York, NY, USA, 2012. ACM.

[60] Peng Song, Wooi Boon Goh, William Hutama, Chi-Wing Fu, and Xiaopei Liu. A handle bar metaphor for virtual object manipulation with mid-air interaction.

In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1297–1306. ACM, 2012.

[61] Alessandro Soro, Riccardo Scateni, et al. Natural exploration of 3d models. In *Proceedings of the 9th ACM SIGCHI Italian Chapter International Conference on Computer-Human Interaction: Facing Complexity*, pages 118–121. ACM, 2011.

[62] Richard Stoakley, Matthew J Conway, and Randy Pausch. Virtual reality on a wim: interactive worlds in miniature. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 265–272. ACM Press/Addison-Wesley Publishing Co., 1995.

[63] Radu-Daniel Vatavu. The effect of sampling rate on the performance of template-based gesture recognizers. In *Proceedings of the 13th international conference on multimodal interfaces*, pages 271–278. ACM, 2011.

[64] Radu-Daniel Vatavu, Laurent Grisoni, and Stefan Gheorghe Pentiuc. Multiscale detection of gesture patterns in continuous motion trajectories. In *Gesture Workshop*, pages 85–97. Springer, 2009.

[65] Vanessa Vuibert, Wolfgang Stuerzlinger, and Jeremy R Cooperstock. Evaluation of docking task performance using mid-air interaction techniques. In *Proceedings of the 3rd ACM Symposium on Spatial User Interaction*, pages 44–52. ACM, 2015.

[66] Xinggang Wang, Bin Feng, Xiang Bai, Wenyu Liu, and Longin Jan Latecki. Bag of contour fragments for robust shape classification. *Pattern Recognition*, 47(6):2116–2125, 2014.

[67] Jacob O Wobbrock, Andrew D Wilson, and Yang Li. Gestures without libraries, toolkits or training: a $ 1 recognizer for user interface prototypes. In *Proceedings*

*of the 20th annual ACM symposium on User interface software and technology*, pages 159–168. ACM, 2007.

[68] Ying Wu and Thomas S Huang. Vision-based gesture recognition: A review. In *Gesture Workshop*, volume 1739, pages 103–115. Springer, 1999.

[69] Jianyu Yang, Junsong Yuan, and Youfu Li. Parsing 3d motion trajectory for gesture recognition. *Journal of Visual Communication and Image Representation*, 38:627–640, 2016.

[70] Yina Ye and Petteri Nurmi. Gestimator: Shape and stroke similarity based gesture recognition. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, pages 219–226. ACM, 2015.