UNIVERSITÀ DELLA CALABRIA
Dipartimento di MECCANICA

DOTTORATO DI RICERCA IN INGEGNERIA MECCANICA

CICLO XXII (2006-2010)

DESIGN AND METHODS FOR INDUSTRIAL ENGINEERING: ING-IND/15

Visuo-Haptic displays for Interactive Mixed Prototyping

By Francesco Cosco

A dissertation submitted in partial fulfillment of the requirements for the

Doctoral Research degree in Mechanical Engineering

Doctoral Research Director

Prof. Sergio Rizzuti

Supervisors

Prof. Maurizio Muzzupappa

Prof. Miguel Otaduy

Candidate

Dr. Francesco Cosco

Acknowledgments

Just three years ago I could not imagine that today I would be here, completing my PhD. This had been possible thank to the big opportunity that all professors of the Department of Mechanical Engineering, Università della Calabria gave me, trusting in my potential. I would like to thank especially my advisors, prof. Maurizio Muzzupappa and prof. Fabio Bruno. They always gave me the right combination of guidance and freedom, let me learn from my errors and encouraging me to focus my attention to the main problems. Thanks also to the rest of the Design Group, in the persons of Chicco, Marco, Loris, Agostino, Stefano and Gianfranco. All them have made the work easier by letting me be their friend and by supporting me at all times. A special thank goes for Stefano for helping me with early drafts of all my publications, included this thesis, helping me to improve my English style.

This research would not have been possible without the precious help of the colleagues at the Department of Computer Science, Universidad Rey Juan Carlos, Madrid. In particular I would thank one of my advisor, Prof. Miguel Otaduy, for being understanding, for helping me identify research problems, for trusting me and for his good, essential, and continuous advice. I would also extend the achievements to Prof. Oscar D. Robles, and more in general to our group of "*los de siempre*". The time enjoyed with them gave me the fuel to work hard during my six-months period in Madrid. I want to thank also all the other colleagues and friends that I meet there: Carlos, Pily, Javier, Alberto, Caroline, Sofia, Marcos, Alvaro, Monica, David, Jorge, Angela, Pablo, Sara.

This research is dedicated to all my families. My original family, in the persons of my mother Pina, my grandmother Martina, my brother Cosimo and Alessandra and my little sister Gemma with her Fabrizio. My spiritual family, that the seraphic father St. Francis: through their faces, their voices, their hugs my Father had been next to me everyday, during the past two years. Last but not least, my actual family in the person of my wife Anna Lisa, and our little angel, too soon went back to heaven.

Contents

List of Figures

List of Tables

Acronyms

AR Augmented Reality 127

CAD Computer Aided Design 21

CACE Computer Aided Control Engineering 6

DMU Digital Mock Up 11

HMD Head Mounted Display 128

HMI Human-Machine Interfaces 127

IBR Image-Based Rendering 48

IMP Interactive Mixed Prototyping 127

MP Mixed Prototyping 30

MR Mixed Reality 127

PDP Product Design Process 1

R&D Research and Development 1

SAR Spatial Augmented Reality 15

VHMR Visuo-Haptic Mixed Reality 23

VR Virtual Reality 127

VP Virtual Prototype 26

To my wife Anna Lisa,

and our little angel, too soon went back to heaven.

"The Lord created the Wisdom;
He has seen her and taken note of her.
He has poured her forth upon all His works,
upon every living thing, according to His bounty;
He has lavished her to everyone loves Him".
Sirach 1,7-8

Chapter 1

Introduction

The Research and Development (R&D) activities are nowadays a crucial factor for the survival of an industrial company on the global market. Due to the increasing level of competition, industries requires to quickly and continuously revise both design and range of their products, as well as their product process and methods, in order to introduce new products on the market in advance to their competitors.

The design of a new product, as well as the re-design of an existing one, is usually accomplished through the *Product Design Process (PDP)*. The PDP can be decomposed into four main steps, as suggested in Fig. 1.1.

The first step regards the definition of design requirements. Starting from marketing analysis, the specialists of the R&D team try to maintain an eagle eye both on competitors, in order to keep pace with modern trends, and customers, analyzing their needs, demands and desires. The crucial purpose of this activity is to maintain a connection between the user expectations and the product during its design, in order to increase the chances of the product to survive on the market once its mass production begin. All these information are collected and concur to the definition of the technical requirements, made by designers.

The second step is the engineering of the product, that leads the R&D activities from the early sketched ideas to the definition of the first prototype. In facts, prototypes are often used as part of the PDP to allow engineers and designers to explore and validate design alternatives, to test theories and/or confirm performance before starting the production.

At this stage, the PDP starts to cycle: usually multiple iterations of prototypes are used to progressively refine the design, before obtaining the final product. However, manufacturing a physical product has high cost and requires time-consuming

production processes. Therefore, designers have to pay attention on keeping, as low as possible, the total number of prototypes used during the PDP, in order to gain, or at least to maintain competitiveness in term of *Time-to-Market* and *Cost-to-Market*. These considerations can be summarized by the following assumption:

> *Reducing the number of physical prototypes would shorten the Time-to-Market and reduce the Cost-to-Market.*

Thanks to the increase of computational power of the last decades, this approach has found a powerful tool in Computer Aided Design (CAD) software, evolved from the purely sketching environments of the beginning, to integrated design environments able to replicate and simulate not only the appearance (shape, color, texture, etc.), but also structural, mechanical, thermal and other physical advanced behaviors of the product. This evolution was inspired and guided by the need of shifting experimental tests from real domain (requiring the use of prototypes of the product) to virtual domain, thus reducing the need of using physical prototypes during the PDP.

Fig. 1.2 depicts the evolution of the semantic representation, focusing the amount of knowledge coded into the model: at each line along the down-up direction, the model acquires some structured information, i.e. the informations are embedded into the model representation. This results in reducing the amount of *background knowledge* needed by a general user to manipulate the product. The ultimate step in this evolution trend is the concept of Virtual Prototype (VP), where the unstructured knowledge is minimized.

A VP is the virtual replica of an industrial product, not yet in production, that appears and behaves as the real one. In other words,

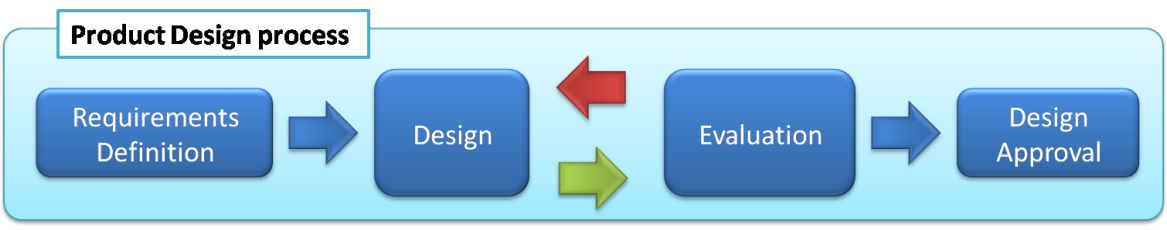


Figure 1.1: *Main tasks of the Product Design Process*

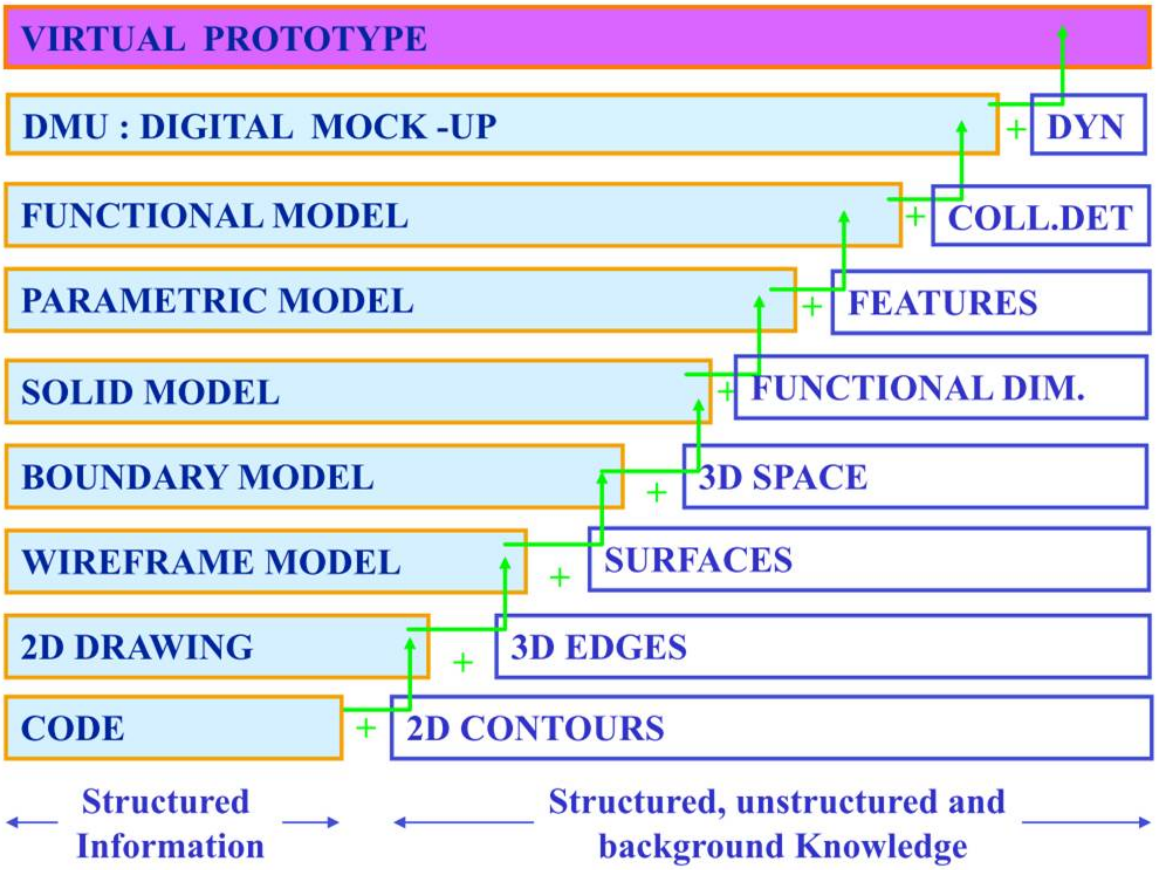


Figure 1.2: *From 2d model to Virtual Prototype [Cugini 2010]*

> *VP is a computer simulation of a physical product that can be* perceived *as it was real, through the use of Virtual Reality (VR) technologies.*

The given definition [Bordegoni 2010] reflects three critical aspects related to the use of a VP along the whole PDP, also referred as Virtual Prototyping.

The first one regards the fact that *the technological complexity of a VP is context-dependent*, where the context is intended as the type of perception that the VP is required to mimic, and the deriving technological challenges that are needed to build the VP. For example, if one requires to test the visual appeal of a new mobile phone, building a VP takes to define all the material properties (as texture, color, diffusion, reflection, etc..) for each material used into the product, then to define the shape and, finally, to apply some rendering methodology. The photo-realistic effect is crucial to mimic the visual aspect of a product, and is widely used during and after the PDP

for presenting the product, e.g. advertising campaigns usually feature some image rendering of VPs. For the visual appeal validation, the time-consumption is a secondary issue, that can be sacrificed to obtain high quality graphical results. In opposite, if one wants to conduct a virtual-assembly test on a new car, the VP should include an efficient multi-body physical simulator, able to solve dynamic and mutual collisions of each body-part, without breaking the real-time constraint. In this task the importance scale is completely inverted: visual appearance became irrelevant and time consumption guides the technological development.

The second aspect relies on the subjective nature of perception: *a VP must be perceptually scalable*. Following previous examples, a successful VP for visual appeal testing should elicit into a subject the same perceptions of a real physical prototype. The cultural background of the subject does not matter, nor his/her psychological aptitude towards the product. If a person likes the product, must like its VP. If another subject feels a sense of peace while looking at the product shape, s/he should feel the same sense of peace looking at the VP.

The last aspect deals with the *perceptual barriers* that are implicitly introduced by the use of VR technologies, as the communication interface between the *perceiver subject* and the product. This communication process can be broken if the receiver perceives the message differently than the intention. Barriers or obstacles to an effective communication may be physical, behavioral/emotional, linguistic and cultural. In the communication domain, perceptual barriers can lead to confusion, misunderstanding, false information and false beliefs, resulting in poor communication. In the domain of Virtual Prototyping, perceptual barriers are mainly due to technological deficits of the environment used to elicit into the subject the perception of the prototype. I consider that Mixed or Virtual Reality, used as an interface between the user and the VP, can limit, distort or even compromise the perception of the product, depending on the amount and the quality of sensations that the interface can transmit to the user.

Despite to the described challenges, a VP can represent a valid alternative to physical prototype for all kind of design evaluation tasks needed along the PDP, including also user test analysis. This is particularly useful in the user-centered design approach,

where common users are involved into the PDP since the early stage of the design. In these situations VPs could provide a common understandable language, different from the sketch based one familiar to engineers, but useful to grant communication of the design ideas between skilled and unskilled people.

The last promising advancement in the field of Virtual Prototyping is the Interactive Mixed Prototyping (IMP) approach, where Mixed Reality (MR) technologies are used instead of the VR interface, and a greater importance is given to the interaction aspects. The key feature of this promising evolution is that MR simplify the virtualization process, focusing the attention only to the simulation of the augmented object and its relations with the real environment. In facts, the derived Mixed Prototyping (MP) approach (Fig. 1.3) combines the advantages of the virtual prototyping (quickly evaluate various design alternatives) and the physical prototyping (involve the sense of touch).

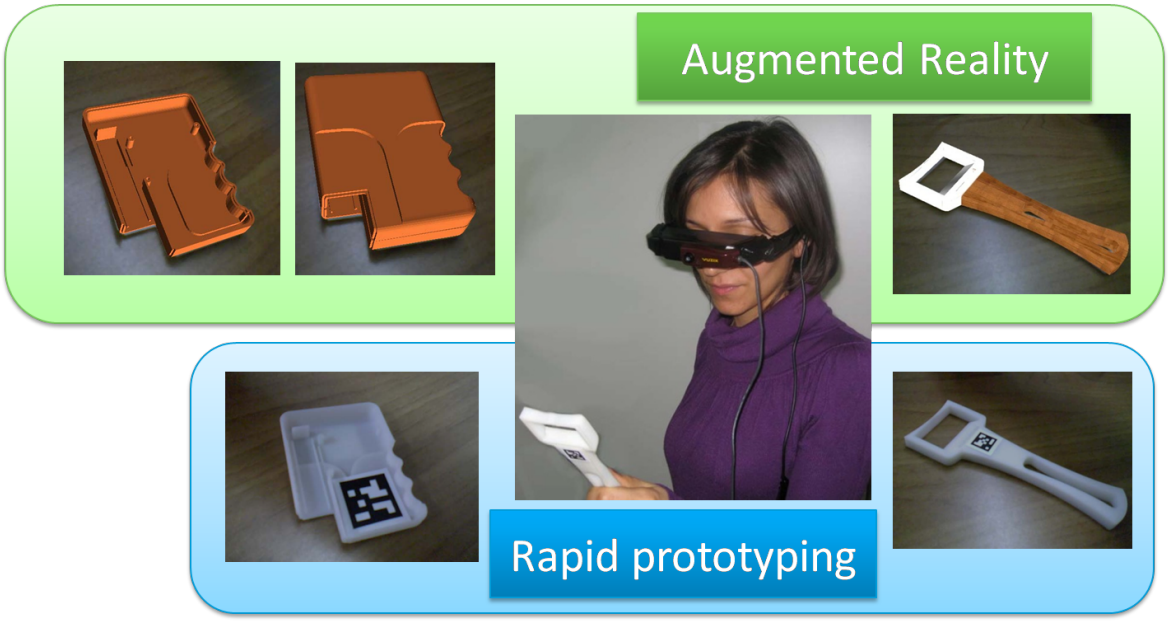


Figure 1.3: *Mixed Prototyping combines the advantages of Virtual Prototyping and Rapid Prototyping*

In the next future, IMP could lead an user to observe a microwave-oven before its production. A person could enter a car, explore its interiors, and drive it, even when it is still a concept. A common user could manipulate its virtual mobile phone, turn it on, then explore its functions, or even make a call.

1.1 Scope of the Dissertation

The final goal of this dissertation concerns *the enhancement of the naturalness of interaction of a person, in the domain of Interactive Mixed Prototyping applications, reducing the* perceptual barriers *introduced by Mixed Reality technologies.*

The given goal can be analyzed at three different levels. First, starting from the given goal, and considering the different typology of interactions of a person facing an industrial product, I recognize two main areas of interest: *direct interaction*, concerning the situations where the user can explore the product interface using his/her hands, and *tool-based interaction*, that refers to all those situations where the subject requires a tool to interact with the prototype.

Second, in order to reach the enhancement of the naturalness of interaction in both the above mentioned situations, I consider the use of some haptic methodology (able to provide the user a tactile feedback of his/her interactions) as a crucial factor. The haptic solutions can be classified into two distinct approaches: passive haptic, where the tactile feedback is obtained using physical prototypes, and active haptic, where the tactile feedback is simulated by special hardware devices, i.e. haptic devices.

Third, I consider the opportunity to improve the design of current IMP interfaces, in order to meet the requirements of their domain of use. Given a simplified representation of the PDP, according to what proposed in [Ulrich *et al.* 1995, Pahl *et al.* 1984], in this dissertation I consider two typical domains of IMP: the usability assessment of industrial products interfaces, where the main goal is to provide the user the ability to naturally interact without distorting its understanding of the object; and the concept design stage of the product, where the main goal is to maximize flexibility of the system, i.e. the environment should be able to automatically reflect any change made into the CAD/Computer Aided Control Engineering (CACE) environments by engineers, to give them the possibility to immediately test and evaluate different design options. In facts, an open issue in MP, closely regarding the concept of flexibility, is the reduction of time and efforts required to generate and update the behavior model of the product as detailed in Sec. 2.3.1.

Every above mentioned level features two operative choices, suggesting a total of eight combinations among the levels. However, as depicted in Fig. 1.4, I consider two of these combinations (both passive haptic solutions oriented to expert users) unpractical: they would rely on the use of some physical prototype to achieve the passive haptic effect. But in the early stage of the product design process, this could require the production of a lot of rapid prototypes. Instead, the expert users involved at this stage would tolerate, better than the final users, the use of some advanced instrumentation (haptic devices, non-lightweight HMD, sensored gloves) necessary to achieve more flexibility.

For these reasons, the dissertation deals with the development of only two of the six possible combinations. First, the hand-based system, developed in chapter 4, regards the direct interaction situations, relies on a passive haptic approach and is particularly oriented to the usability assessment domain (as described in chapter 5).
Second, the tool-based system, developed in chapter 6, is designed for the early design

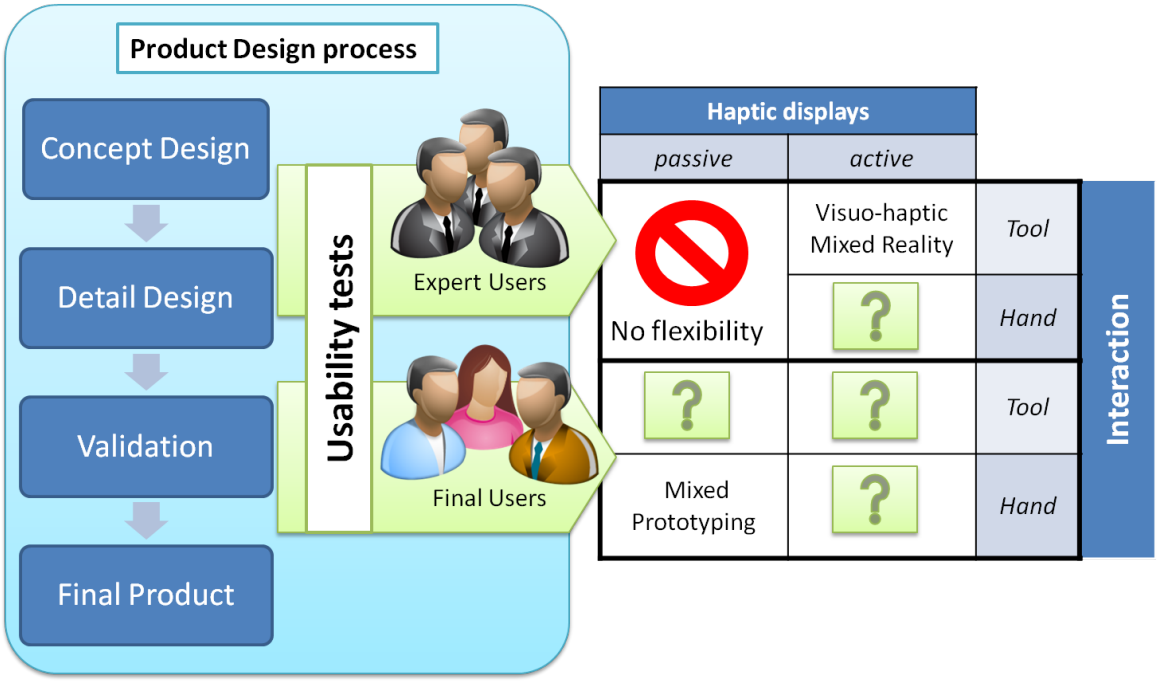


Figure 1.4: *An overview of the proposed Mixed Reality interfaces classified according to: domain of use (expert vs. final users), haptic approach (active vs. passive), interaction typology (by hand vs. by tool).*

stage. The flexibility is achieved using the active haptic approach. The described proof-of-concept implementation is particularly oriented to tool-based interaction situations. Moreover, the development of this system could be considered also as a preliminary study for a further extension to the hand-based case.

1.2 Summary of Results

As already stated in the introduction, the use of MR in the product development process is emerging as a promising solution that combines the advantages of virtual and rapid prototyping. A mixed prototype is usually based on a physical mock-up on which the visual appearance of the product is superimposed thanks to the augmented reality technologies (Fig. 1.3).

Chapter 4 presents a Mixed Reality Environment in which the product behavior is simulated using the same models and the same software employed by the engineers in the design phase. This approach guarantees the reliability of the simulation and allows a strong reduction of the time needed to develop the digital prototype.

Moreover, I presents an innovative technique, specifically studied for the simulation of electrical appliances, that aims to make the user able to naturally interact with the mixed prototype.

This MR environment proposes the following original features:

- the ability to properly manage the occlusion between user's hand and virtual objects;
- a more natural interaction metaphor, through the interpretation of the hand gestures that a user accomplishes during his/her interaction with the elements of the product interface;
- it does not require any specific device like data-gloves or tracking systems;
- the whole application works at interactive frame-rate.

The easiness of use of the proposed system was validated through some usability tests, conducted with users (as described in Chapter 5): during the test the users performed the required task on the mixed prototypes with completion time comparable to experiments using real products, especially for the task requiring only interactions with the buttons. Other kind of widgets (knob, slider, etc..) still present some distortion in the completion time.

Chapter 6 presents the proof-of-concept implementation of a novel Visuo-Haptic paradigm designed with two main intentions: reducing the perceptual impedance of the visuo-haptic system, and providing a more flexible interface oriented to expert users, i.e. engineers and designers.

The proposed systems performs a *perceptual consistent compositing* that simultaneously solves some typical problems of Visuo-Haptic Mixed Reality (VHMR): mutual occlusions between user's hands and virtual tool (better discussed at Sec. 2.1.3), the visual obtrusion of the haptic device (presented at Sec. 2.2.3), the spatial collocation of visual and tactile stimuli (Sec. 2.2.2). The presented VHMR system is expected to perform well in usability test accomplished by designers, and other experts users, during the concept design stage.

1.3 Outline

Chapter 2 gives a multi-disciplinary background of the state-of-the-art related to three different research domains: Mixed Reality, Visuo-Haptic Mixed Reality, and Interactive Mixed Prototyping. The original contributions of this dissertation are then organized as follow. Chapter 3 describes some Computer Vision component, part of the wider MR framework that is the basic software infrastructure upon which I developed two original and complementary environments for IMP: the Tool-Based one, described in Chapter 6, concerning the visuo-haptic simulation of human interaction, mediated by an hand held virtual tool, with both real and virtual objects; the Hand-Based one, described in Chapter 4, regarding the simulation of the interaction of a human being with the Human Machine Interface of a VP, using its own hands. Finally, Chapter

5 regards an experimental test case aiming to validate the usability assessment of an industrial product through Virtual Prototyping and *Hand-based* IMP.

Chapter 2

Related Works

This chapter gives a multi-disciplinary background of the state-of-the-art related to three different research domains: Mixed Reality, Visuo-Haptic Mixed Reality, and Interactive Mixed Prototyping. The primary goal of this review is to provide a general overview of the main technologies applied in this work, with a greater emphasis given to the topics for which this thesis propose original contributions.

Contents

2.1 Mixed Reality

Mixed Reality (MR) refers to the merging of real and virtual worlds to produce new environments and visualizations where physical and digital objects co-exist and interact

in real time. In 1994 Paul Milgram and Fumio Kishino [Milgram & Kishino 1994] introduced the concept of *Virtuality continuum* in order to clarify and distinguish MR from all other kinds of Virtual Reality (VR) related environments.

> "The conventionally held view of a VR environment is one in which the participant-observer is totally immersed in, and able to interact with, a completely synthetic world. Such a world may mimic the properties of some real-world environments, either existing or fictional; however, it can also exceed the bounds of physical reality by creating a world in which the physical laws ordinarily governing space, time, mechanics, material properties, etc. no longer hold. What may be overlooked in this view, however, is that the VR label is also frequently used in association with a variety of other environments, to which total immersion and complete synthesis do not necessarily pertain, but which fall somewhere along a virtuality continuum. In this paper we focus on a particular subclass of VR related technologies that involve the merging of real and virtual worlds, which we refer to generically as Mixed Reality."

According to the Milgram's Virtuality Continuum (Fig. 2.1), Augmented Reality (AR) is a subset of the MR, concerning the insertion of virtual objects into the real world. On the contrary, Augmented Virtuality refers to all those situations where the virtual context is dominant in respect with the real component.

Later on, Ronald Azuma clarified the concept asserting that "AR combines real and virtual, is interactive in real time and is registered in 3d"[Azuma 1997]. Given this commonly accepted definition [Zhou *et al.* 2008], and considering that the global

Figure 2.1: *Milgram's Reality-Virtuality Continuum*

scope of the thesis (1.1) deals with the interaction of a virtual prototype immersed in a real environment, in this dissertation I refer to AR and MR without making a clear distinction between them, assuming that:

> Both MR and AR systems enhances the perception of the real world by adding virtual objects within a real scene. An AR or MR scene, in fact, is created by combining the real scene viewed by the user and a virtual scene generated by computers, respectively augmenting or mixing the scene with additional information which can be useful to the user.

The first application of AR, as we define it, dates back to Sutherland's work in the 1960s, which introduced a see-through Head Mounted Display (HMD) to present 3D graphics [Sutherland 1968]. However, the term of AR was first introduced by Caudell in 1992 [Caudell & Mizell 1992].

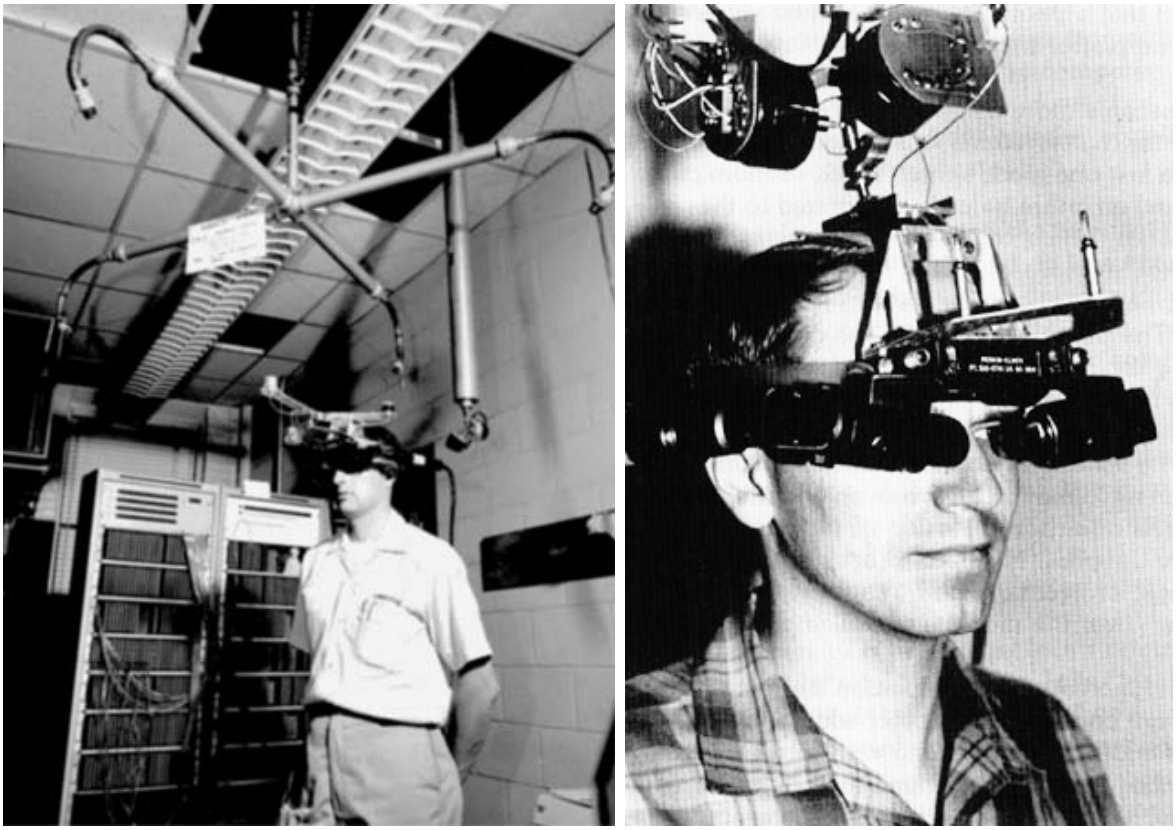


Figure 2.2: *The first Head Mounted Display proposed by [Sutherland 1968]*

A comprehensive overview of the first developments of AR could be found in [Azuma 1997, Azuma *et al.* 2001]. A more recent review of the specific literature is available in [Zhou *et al.* 2008]. All these works give a good starting point for understanding the main technological issues of this growing technology as well as its large scope, including the medical, manufacturing, visualization, path planning, entertainment, and military applications that have been developed.

In the following subsections I will focus on some challenging technological characteristics present in any AR system: (I) the blending approach: optical, video, and projector based; (II) the real time registration of the 3D virtual objects in respect of a 3D real environment, also know as tracking; (III) the real time occlusion handling between virtual graphics and real objects.

2.1.1 Display techniques for Augmented Reality

There are three major display techniques for Augmented or Mixed Reality: See-through Head Mounted Displays, Handheld Displays and Spatial Displays.

2.1.1.1 See-through Head Mounted Display

A see-through HMD places images of registered virtual graphics over the user's view of the world. There are two main categories: optical see-through and video see-through. Optical see-through HMDs allow the user to see the real world directly, i.e. users can see the physical world with their natural eyes. This is achieved using a holographic optical element, a half silvered mirror or similar technologies. On the contrary, with a Video see-through HMD the user observes the real world through the video-images coming from a camera, and the virtual graphics are overlaid on each image.

Compared to Optical see-trough displays, Video see-trough HMDs benefit the consistency between real and synthetic objects, can handle occlusion problem more easily, but they drastically distort the natural perception of the real world introducing problems like a limited field of view, the fixed parallax, the limited resolution and not lightweight structures.

2.1.1.2 Handheld Displays

A Handheld Display is a small computing device with a display, that fits in a user's hand. Several handheld devices (like Tablet PCs, mobile phones, smart phones, PDAs, etc.) are commercially available, and could be used for a mobile AR platform.

Compared with HMDs, handheld devices pose several limitation due to reduced computational power and memory. Moreover, they do not provide immersive visualization, but they are minimally intrusive and make the user able to explore the environment using the handheld display as a video see-trough mobile window.

2.1.1.3 Spatial Displays

Instead of wearing or carrying the display such as with HMD or handheld devices, the users of Spatial Augmented Reality (SAR) make use of digital projectors to display graphical information onto physical objects. The peculiarity of SAR is that the display is separated from the users of the system. As the displays are not associated with each user, SAR scales naturally up to groups of users, thus allowing for collocated collaboration between users.

SAR has several advantages over traditional head mounted displays and handheld devices. The user is not required to carry equipment or wear the display over their eyes. SAR does not suffer from the limited display resolution of current head mounted displays and portable devices. A projector based display system can simply incorporate more projectors to expand the display area. Where portable devices have a small window into the world for drawing, a SAR system can display on any number of surfaces of an indoor setting at once. The tangible nature of SAR would make this an ideal technology to support design, as it provides both a graphical visualization and a passive haptic sensation for the end users. However, one main disadvantage is that SAR devices usually lack mobility, due to the fixed set-up used, and also present a problem of self-occlusion.

2.1.2 Tracking for Augmented Reality

Correct tracking of the user position is crucial for an AR application. In order to obtain a perceptually correct AR scene visualization, a robust registration of the virtual objects within the real scene is needed. Without accurate registration, the illusion that the virtual objects exist in the real environment is severely compromised. The correct alignment of virtual objects is possible only after the estimation of the user pose within the scene. Virtual objects, in fact, must be rendered according with the user's point of view, and any mis-registration will prevent the user from seeing the virtual and real scenes as fused together [Bajura & Neumann 1995]

For this reason, tracking became a very important research subject in the field of AR. An overview of tracking systems is presented in [Rolland *et al.* 2001]. According to [Zhou *et al.* 2008], there are three main categories of tracking techniques: sensor-based, vision-based, and hybrid.

The sensor-based tracking techniques rely on using some sensors (magnetic, acoustic, inertial, optical, mechanical, etc..) to measure the position of the user. These methods were mainly developed in the field of Virtual Reality [Rolland *et al.* 2001]. However they never became popular for AR, where the video-stream (available from the some camera used for video-see-trough) can be used with vision-based methods, thus without requiring specific and more expensive hardware.
In facts, Vision-based tracking techniques can use image processing methods to compute the camera pose with respect to real world, which results in a dynamic error correction, typical of closed loop systems [Bajura & Neumann 1995].

Among all methods, vision-based tracking is the more commonly employed for AR [Park *et al.* 1999]. Its main advantage, especially for video-see-trough AR set-up, is that vision methods can estimate camera pose directly from the same imagery observed by the user, without using a separate sensor or emitter attached to the environment. This has several advantages [Neumann & Cho 1996]:

- tracking may occur relative to moving objects;
- tracking measurements made from the viewing position often minimize the visual

alignment error;

- tracking accuracy varies in proportion to the visual size (or range) of the objects in the image.

In general, vision based tracking consist of at least two steps: *detection* of features and *pose estimation*. The *detection* step consists of processing the video signal coming from the camera in order to find a set of visual *feautures*, i.e. some recognizable entities of the observed scene. This is fully executed in image-space, adopting some Computer Vision strategy. Moreover, all features need to be classified, in order to be robustly distinguished among each other. At the end, given the set of found features on image-space, and each corresponding real position, the *pose estimation* consists of computing the pose matrix that solves the described projective geometry problem.

For the pose estimation step, several vision-based tracking solutions are based on using artificial fiducial features like planar targets [Kato & Billinghurst 1999, Gerhard Shahzad *et al.* 2002, Kawano *et al.* 2003, Santos *et al.* 2006] or coplanar points [Lu *et al.* 2000]. Some have been extended to use points and lines [Ansar & Daniilidis 2003], and some others for arbitrary 3D target point configurations [Araujo *et al.* 1998]. Recent success has also been reported for live structure and motion estimation [Nistér *et al.* 2004].

Among all vision techniques, the marker-based optical tracking is, probably, the most common approach in AR applications. The pose estimation for this technique was widely investigated during the last decades. Abidi et al. [Abidi & Chandra 1995] presented a pose estimation algorithm based upon invariant properties of quadrangular markers. Quan et al. [Quan & Lan 1999] give a general solution for the N-point resection problem. Another widely employed method in computer vision has been presented by Zhang [Zhang 2000], in which both the camera calibration and resection problem are solved for the particular case of a planar marker. For the same problem, Kato et al. [Kato & Billinghurst 1999] presented a geometrical approach based upon the projection of the lines passing through marker sides. Schweighofer and Pinz [Schweighofer & Pinz 2005] solved the resection problem taking into account pose am-

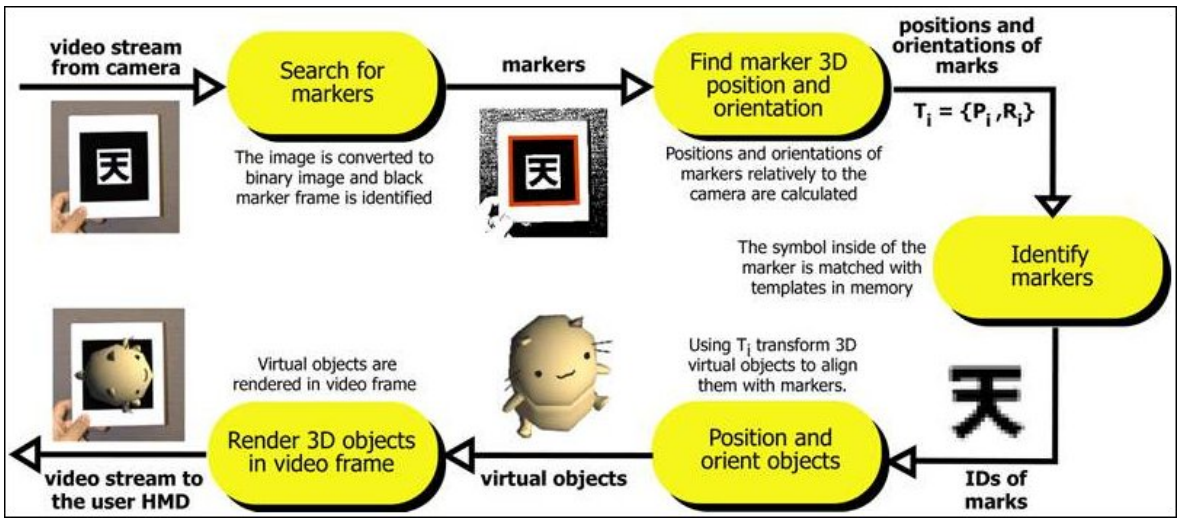
biguities due to perspective projection. Lucchese [Lucchese 2006] presented a closed solution suitable for real time application, using an approach similar to Zhang, i.e. based upon the homography relationship between marker plane and camera plane.

More generally the marker-based tracking approach requires the solution of three sub-problems:

1. camera(s) calibration in order to compensate for image distortion and retrieve the intrinsic property of the camera projection matrix;

2. pattern recognition of the marker within the image(s) and solution of the correspondence problem;

3. marker pose estimation with respect to the camera system coordinates (resection problem).



More information availables on the Artoolkit's webpage

Figure 2.3: *The tracking and rendering pipeline in ARToolkit.*

ARToolkit [Kato & Billinghurst 1999] is one of the most commonly used software library able to completely solve all of the above-listed problems. This C-Library was originally developed by Dr. Hirokazu Kato, and its ongoing development is being supported by the Human Interface Technology Laboratory (HIT Lab) at the University

of Washington, HIT Lab NZ at the University of Canterbury, New Zealand, and ARToolworks, Inc, Seattle. The great success of ARToolKit relied on the fact that, as a software library that can be used to calculate camera position and orientation relative to physical markers in real time, it enables the easy development of a wide range of Augmented Reality applications (see the Artoolkit's webpage for further informations).

The system relies on putting into the scene some square marker patterns mounted on a planar surface. The ARToolKit tracking works as follows:

- the camera captures video of the real world and sends it to the computer;
- the software searches through each video frame for any square shapes;
- for each square found, the software uses some mathematics to calculate the position of the camera relative to the black square.

Once the position of the camera is known, a computer graphics model is drawn according to the perspective view relative to that position: the model is drawn on top of the video of the real world and so appears stuck on the square marker. The final output is shown back in the handheld display, so when the user looks through the display s/he sees graphics overlaid on the real world.

Some years later, Mark Fiala presented an improved fiducial marker systems, ARTag. This system featured an increased robustness against non uniform illumination scenarios and accidental occlusion of the marker's borders [Fiala 2005a, Fiala 2010].

Partially inspired by ARTag, Daniel Wagner and Dieter Schmalstieg released ARToolkitPlus[Wagner & Schmalstieg 2007], the successor of Artoolkit. It came out by the effort to optimize and extend ARToolkit for the usage on handheld devices such as smartphones, PDAs, and Ultra Mobile PCs. However, they basically wrapped ARToolkit into a C++ class, and then developed a class-based API to integrate some new features and many improvements, mainly to mimic the tracking strategy adopted by ARTag.

A detailed explanation of the main differences between these three systems, and some analytical results were presented by [Fiala 2005b]. His experimental analysis

demonstrated that the edge-based approach of ARTag allows more unique features (quadrilateral contours) to be detected with respect the to binary morphology method based on greyscale thresholding used by ARToolkit Plus. Moreover, he has shown how ARTag is able to work properly despite lighting variations and partial occlusion.

A more recent project, aimed to provide a framework for Augmented Reality applications, was the Studierstube Augmented Reality project: Augmented Reality for collaborative and ubiquitous computing. It provide both marker-based tracking technology, as well as modern markerless methods. However, it is not open-source.

Despite all the development history of optical marker-based tracking, this vision-based technology still suffers from a notorious lack of robustness: end-to-end system delay, and high computational expense. Since vision sensors (cameras) nominally sample at video rates (30Hz), they are most appropriate for measuring low-frequency pose variations. Rapid or abrupt camera rotations or motions can cause vision tracking failures or instabilities. For this reason, users of such systems usually observe that vision-based pose is not very precise, which results in significant jitter, and not very robust suffering from pose jumps and gross pose outliers. Increasing accuracy, stability and precision is one of the most addressed challenges in the last years.

In many works, some researchers attempted to improve stability adding inertial sensors in order to compensate for rapid acceleration. In this cases a kalman-filter is used to fuse information of each tracking methodology.

The accuracy of the optical tracking can be improved using more than one camera. The system presented in [Kanbara *et al.* 2001] employs two cameras. At the beginning, the tracker uses fiducial markers to calculate the camera projection matrices, and afterwards uses natural feature points to determine the pose.

Very often, an AR system uses a video see-through HMD equipped with two cameras. Therefore, it is possible to use the information coming from both cameras for tracking, in order to improve tracking accuracy. According to this, I developed a methodology for real-time tracking (see Sec. 3.1), that exploits the use of two cameras for tracking, switching itself into an ordinary single camera method whenever a stereo pair is missing, i.e. every time the marker is visible only for one camera.

Despite the problems described above, several AR systems are nowadays implemented using the marker-based tracking approach. It relies on placing into the scene a set of fiducial markers, that are recognized and used to solve the tracking problem [Zhang *et al.* 2002]. However, it is not easy to use this approach in a realistic, dynamic and complex scenario (e.g. for a worker facing a car on the assembly line).

An answer to this problem may come from the last frontier of the tracking research: natural feature tracking [Neumann & You 2002, Yuan *et al.* 2006, Guan & Wang 2009, Wagner *et al.* 2009, Gruber *et al.* 2010, Chen & Li 2010]. The basic idea is to use naturally occurring features instead of the artificial markers, thus without altering the real scenario. In a few recent works, some researchers have proposed a set of descriptors, more robust and efficient if compared to geometrical fiducials such as points, lines, edges or textures: the Scale Invariant Feature Transform (SIFT)[Lowe 2004], Speeded Up Robust Features (SURF)[Bay *et al.* 2006], or *Ferns* [Ozuysal *et al.* 2007, Ozuysal *et al.* 2010].

Another suitable alternative is given by the model-based tracking methods, that explicitly use a model (such as a Computer Aided Design (CAD) one) in order to retrieve the object features to be tracked. A widespread solution is the use of Simultaneous Localization and Mapping (SLAM) methods [Neubert *et al.* 2008]: they construct edge-based models from image sequences and use them for tracking without needing any prior knowledge of the world. An alternative to SLAM was proposed by [Klein & Murray 2008]. Their Parallel Tracking and Mapping (PTAM) method is able to produce detailed maps, with thousands of features tracked at frame-rate. Recently it was successfully extended in [Newcombe & Davison 2010] where a complete 3D model of a cluttered indoor environment can be reconstructed from a live handheld camera in a few seconds, and then used as a structured reference for the tracking.

2.1.3 Occlusion handling

Realistic occlusion between real and virtual objects enhances users' perception that virtual objects truly exist in the real world, whereas an incorrect occlusion handling confuses viewers[Sekuler & Palmer 1992]. There are two large classes of methods for

handling occlusion in the visual compositing of an AR scene: those that exploit prior geometric models of the real objects, and those that do not assume previous knowledge about the geometry. In model-based techniques, model registration becomes the main computational task, while non-model-based techniques must rely on image processing and depth estimation techniques [Breen *et al.* 1996, Fischer *et al.* 2007].

Furhmann et al.[Fuhrmann *et al.* 1999] extended model-based occlusion handling to manage occlusions caused by the user's body parts. They utilized an avatar of the user, modeled as a kinematic chain of articulated solids. The avatar was continuously tracked to mimic user motions, and then used to compute occlusions. Unfortunately, its registration is a time-consuming task, not suitable for real-time applications. Lok et al.[Lok 2003] developed a method relying on image-based modeling techniques, capable of incorporating real objects into a virtual environment, in order to avoid the pre-definition of the geometric model of the real scene.

Non-model-based techniques typically aim at managing occlusion between real and virtual objects in dynamic scenes. The most investigated solution involves depth estimation of the scene, and due to the high computational cost of depth estimation, most approaches have targeted the optimization of performance with real-time applicability in mind. In [Kanbara *et al.* 2000] the processing time was optimized by restricting an edge-based stereo matching algorithm to the screen-region covered by virtual artifacts. Schmidt et al. [Schmidt *et al.* 2002] presented an efficient method for computing dense stereo matching. However, recently very promising solutions have been obtained by adopting special hardware and/or by designing highly parallelizable algorithms on graphics processors. Fischer et al. [Fischer *et al.* 2007] proposed a prototype system that integrates time-of-flight range data to compute depth maps. In [Gordon *et al.* 2002], the authors improved computational performance adopting a dense-stereo matching approach, executed on dedicated hardware. Lu and Smith [Lu & Smith 2009] developed a GPU-based dense stereo matching algorithm 20× faster than the equivalent CPU-based optimized algorithm.

Ventura and Hollerer [Ventura & Hollerer 2008] designed a technique aimed at the particular but very common case where a real occluding object, e.g., the user's hand,

lies between the user's viewpoint (the camera) and the virtual objects. Given this assumption, they described a method that combines stereo-matching depth reconstruction with a color-based statistical refinement for noise reduction. In this dissertation, I develop a similar strategy, presenting a skin-detection method (see Sec. 4.2.4) that correctly manages occlusions between virtual objects and user's hands without performing depth reconstruction, with the subsequent benefits for real-time performance.

In [Lee & Park 2005], Lee and Park presented a method for mixed prototyping applications that uses physical props of the virtual objects to determine depth relationships. They painted the physical prop with a special color, then applied a chroma-key filtering to the image, and composed the virtual object only in the chroma-key regions.

2.2 Visuo-Haptic Mixed Reality

MR has typically dealt with the visual addition of virtual objects to a real scene. But, in order to fully experience the mixed environment, the integration of virtual and real objects must be extended to the rest of the sensory modalities.

Visuo-Haptic Mixed Reality (VHMR) consists of adding to a real scene the ability to see and touch virtual objects. It requires the use of some MR compositing approach (Sec. 2.1.1), e.g. see-through display technology, for visually mixing real and virtual objects, and haptic devices for adding haptic interaction with the virtual objects. Among others, VHMR has been introduced in medical applications [Fornaro *et al.* 2008], virtual prototyping, e.g., for the automotive industry [Ortega & Coquillart 2005], or digital entertainment [Knoerlein *et al.* 2007].

This section discusses general research on visuo-haptic displays focusing on three different aspects: (i) the general haptic rendering theory used as basis to develop the visuo-haptic displays proposed in this dissertation (chapter 6), and the specific problems of (ii)visual obtrusion and (iii) spatial collocation for a correct multi-sensory perception in Visuo-Haptic Mixed Reality.

2.2.1 Haptic Rendering Background

In the context of visuo-haptic displays, the haptic rendering is a fundamental component in order to achieve the desired tactile and kinesthetic feedbacks, while the user interacts with virtual objects.

Its role is to monitor the contacts between objects and accordingly compute the reaction forces and torques that the haptic displays (i.e the haptic devices) transmit to the user.

A good overview of the more recent advance in this topics could be found in [Otaduy & Lin 2008], where authors clarify the need of a multirate rendering approach, in order to improve stability. Their solution exploit the virtual coupling concept (Fig. 2.4), previously presented in [Weir & Colgate 2008, Colgate *et al.* 1995].

This commonly accepted solution allows to provide a good quality of the haptic

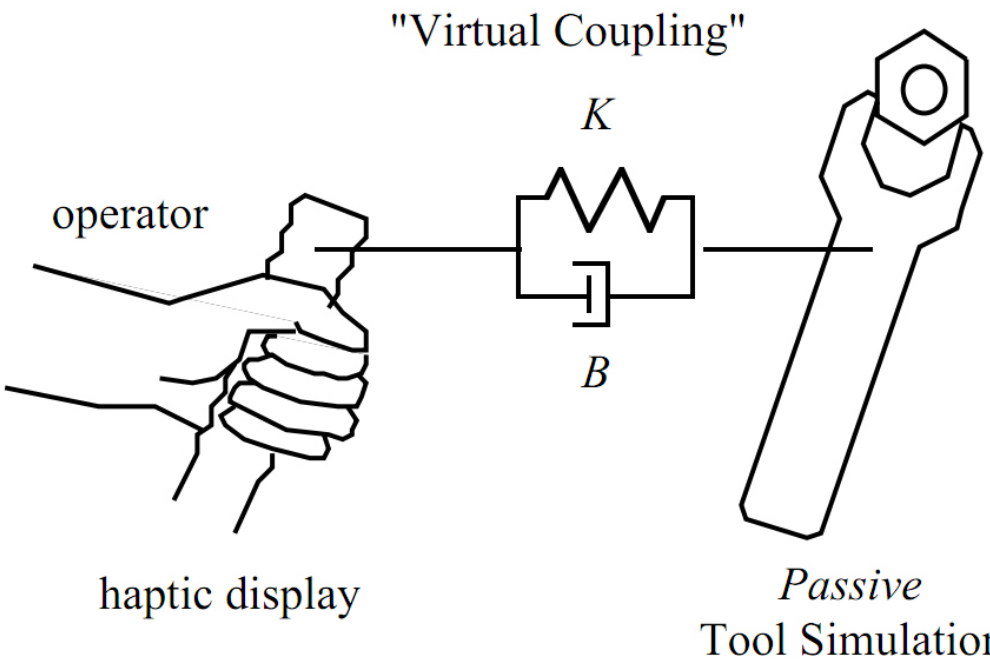


Figure 2.4: *The* virtual coupling *concept proposed in [Colgate et al. 1995]*

feedback, but in Visuo-Haptics environments it can results in some undesirable misalignments between the hand held tool and the visualized real hand (I propose an original solution to this problem in chapter 6).

The researches presented in this work are mainly based on two recently developed approaches: first, for the physical interaction between virtual object I adopt the constrained dynamics algorithm proposed in [Otaduy *et al.* 2009] able to solve deformation and contact of both rigid and deformable objects; and I use the multi-rate haptic rendering algorithm proposed by [Garre & Otaduy 2009], mainly designed for interactions through hand held rigid tools.

2.2.2 Haptic collocation

Several researchers have addressed the importance of collocating visual and haptic stimuli, as this allows virtual tasks to be carried out from a first-person point of view [Coquillart 2007]. Visual and haptic de-location is however quite common, because the construction of a de-located setup is far simpler. Congedo et al. [Congedo *et al.* 2006] emphasize that, in tasks where the contribution of touch is important, great effort should be undertaken to collocate vision and touch, so that the weight of the non-dominant modality, i.e., touch, is not penalized. Spence et al. [Spence *et al.* 2008] summarize crossmodal congruency effects involving vision and haptics.

Visuo-haptic collocation can be achieved in several ways, and the most popular ones include workbenches with stereo projection systems [Brederson *et al.* 2008, Tarrin *et al.* 2003], mirror-based projection systems where the virtual image occludes the real scene [Stevenson *et al.* 1999], or head-mounted displays with head and device tracking [Bianchi *et al.* 2006].

2.2.3 Haptic device camouflage

In VHMR, haptic devices tend to be bulky items that appear in the field of view of the user, disturbing or compromising the user's attention while interacting with a VP.

One possible solution to visual obtrusion is to use stringed haptic devices, such as the SPIDAR [Ishii & Sato 1994]. Stringed haptic devices place the actuators far from the region where manipulation and interaction are actually happening, and

transfer force and torque to the end effector using tensed strings. With sufficiently thin strings, the haptic device barely occludes the rest of the scene. Ortega and Coquillart [Ortega & Coquillart 2005] applied this visuo-haptic interaction paradigm in the context of an automotive virtual prototyping application. Moreover, they used as end-effector a geometric prop of the actual tool, and mounted a transparent structure around the prop for adequately attaching the strings. Stringed haptic devices have been integrated in a workbench that provides view-dependent stereo vision [Tarrin *et al.* 2003].

Another possible solution to visual obtrusion is optical camouflage [Inami *et al.* 2003], which consists of covering the obtrusive elements with retro-reflective paint, and use a projector to render on top of them the desired background image. This approach was proposed by Inami et al. [Inami *et al.* 2000] for solving the visual obtrusion produced by haptic devices in mixed reality scenes. The VHMR paradigm presented in chapter 6 can be perhaps interpreted as a computational approach to optical camouflage. In practice, optical camouflage is also closely related to diminished reality [Zokai *et al.* 2003].

2.3 Interactive Mixed Prototyping

As already discussed (1), the Virtual Prototype (VP) is the final stage of the evolution of model representation. The first section gives a deep overview about the last step of Fig. 1.2 that leads from DMU to VP. The second section focuses on the most recent advances in Virtual Prototyping, regarding the development of a more reliable interface between the user and the prototype, that leads to the concept of *Interactive Mixed Prototyping*.

2.3.1 From DMU to Virtual Prototype

VR technologies are widely employed for the aesthetic validation of industrial products. VR applications, in fact, provide a high quality, immersive visual representation of virtual prototypes, so that designers can easily evaluate aesthetic qualities and/or

discover any styling defect. Recently, VR has been used also for the simulation of virtual prototypes, because it allows engineers to enhance the analysis and validation of the digital product, before manufacturing any physical mock-up. Through VR it is also possible to conduct training sessions without any risk for the users (as it is in the case of particular products like vehicles, machine tools, etc.) or to carry out usability analyses [Jimeno & Puerta 2007]. In this second case, VR demonstrates to be a valid tool, because it is able to support and facilitate the participatory design of industrial products. VR allows designers to involve the final users of a product since the early design stages, without the need of a physical mock-up and with the advantage of being able to assess several design options [Bruno *et al.* 2007].

Among the various utilizations of VR in the industrial field, Virtual Prototyping seems to be one of the most promising and challenging [Jimeno & Puerta 2007]. There are several examples where VR is used to simulate the behavior of the product, a typical example is in training tasks. In [Acal & Lobera 2007] a training operator environment for a numerical control milling machine is presented. In [Bergamasco *et al.* 2005] an innovative fork-lift simulator, suited for training in industrial environments is presented. The numerical model is modeled and simulated using the Open Dynamics Engine library, hence any modification to the model implies the modification of the application code. Other interesting examples about the use of VR and simulation for training tasks are presented also in the medical field [Grantcharov 2006].

Unfortunately, VR software tools are not able to fully simulate the behavior of a virtual product, because they are mainly addressed to the aesthetical validation of the product. The functional simulation that one may obtain is limited to some basic behaviors like movements triggered by an event (e.g.: a door will open when clicking on a handle). Some of the most recent tools provide built-in physics simulators based on gaming technologies, in order to simulate physical phenomena like collisions and rigid bodies dynamics. Such simulators can provide fast results, but they are not as accurate and flexible as the simulation software used by engineers. As a consequence you donŠt obtain a simulation, but an animation which cannot be employed as a robust validation tool. Furthermore, when the product behavior has been sim-

ulated in VR [Barbieri *et al.* 2008, Aoyama & Kimishima 2009, Bordegoni *et al.* 2009, Park *et al.* 2008, Park *et al.* 2009], this has been done through the implementation of the code that replicates the behavior models of the product inside the VR software. This means that each change in the digital mock-up has to be manually reported in the code of the VR application, thus requiring a big effort for the VR operators.

In [Park *et al.* 2008] Park et al. present a study about the use of VR in product design evaluation. They focus their work on the simulation of the Human-Machine Interfacess (HMIs) that are analyzed through a state transition methodology to capture the functional behavior of the product, from which they construct a finite state machine for its functional simulation. The authors put in evidence that one of the limits of their approach is that *"it is very meaningful and challenging to devise an efficient way to reduce the time and effort required to generate the HMI behavior model of the product by reusing the information built in the embedded software".*

The previously mentioned works put in evidence that there is a lack of software tools able to support designers in the development of interactive and functional virtual prototypes. This lack is probably one of the main obstacles in the diffusion of VR techniques for the product behavior simulation. Engineers, in fact, use specialized simulation software to design the industrial product: at the moment it is not possible to evaluate the models developed in these simulation packages directly in VR. As pointed out in [Bruno *et al.* 2007] it is necessary to work out a specific solution for the several problems which occur during the integration between VR and the other applications of the product development process, including the simulation.

Some interesting works about the integration of simulation and VR have been realized in the Multi-Body Systems (MBS) field. Cuadrado et al. [Cuadrado *et al.* 2004] present an application for the virtual prototype of a car, where manoeuvrability evaluation in a VE is carried out by integrating 3D visual and haptic feedback. Eberhard and Li [Eberhard & Li 2006] also present MBS simulation and control applications using VR as a user interface. In both these cases, the MBS real-time simulations represent the main limitation. Antonya and Talaba [Antonya & Talaba 2007] present a novel approach to simplify MBS computations in the case of visual evaluation and modification

of virtual mechanical systems. The applications developed by the authors incorporate common interactive facilities offered by a virtual environment, e.g. stereoscopic visualisation in a CAVE-like immersive system, walk-through, interactive object handling. These interesting works demonstrate the added value of integrating VR with simulation in order to optimise a design.

Another interesting approach to the integration of simulation packages and VR has been presented by Sanchez et al. [Sànchez *et al.* 2005], who developed the Easy Java Simulation (EJS) a software tool designed to create interactive simulations in Java using models created with Simulink. EJS can also be used with Java 3D to create interactive 3D virtual products, but it has been conceived mainly for educational purposes and it cannot be efficiently integrated into a design process, because Java 3D is not suitable for the visualization of complex models.

In [Sinha *et al.* 2000], an environment for the behavioral simulation of CAD assemblies is presented. The global model is formed by several component with a behavior (simulation model) and a form (CAD model) connected through a port-based paradigm. The framework performs a numerical simulation to predict the behavior, but no visualization of the CAD model occurs.

In conclusion, the analysis of the state of the art puts in evidence that there is the need to create realistic product simulations in VR for several purposes like tests with users, training, functional validation, and others application. Previous researches have been focused on the devices that better allow the user to interact with the virtual or augmented mock-up. Some authors put in evidence that, in order to make this approach more effective, it is necessary to reduce the time required to implement the behavior simulation of the product.

The Mixed Prototyping approach described in chapter 4 relies on the simultaneous use of VR and numerical simulation environments. In fact, the simulation models of the product are often present in manufacturing companies, but they cannot be used to simulate the product in VR. Therefore, the approach followed, is based on the inter-process communication among different software modules, and relies on a middleware for the software communication. This approach allows to freely choose the software to

Figure 2.5: *Aesthetic evaluation of a Rapid Prototype [Fiorentino et al. 2002]*

be employed, both for simulation and visualization. In this way engineers can use their favorite software during the product development phase.

2.3.2 Form Virtual Prototype to Mixed Prototype

AR and MR have been widely used in industrial applications both in design and manufacturing [Lu *et al.* 1999, Regenbrecht *et al.* 2005, Ong *et al.* 2008].

The IMPROVE project [Santos *et al.* 2007] proposes several MR solutions in the area of collaborative design review: HMD development using unique OLED technology, markerless tracking, augmented reality rendering, image calibration for large tiled displays, collaborative tablet-based and projection wall oriented interaction and stereoscopic video streaming for mobile users.

Some researches have focused the attention on the concept of Mixed Prototyping (MP) [Fiorentino *et al.* 2002, Bordegoni *et al.* 2009, Verlinden & Horvath 2006, Verlinden & Horvath 2007, Verlinden & Horvath 2009, Aoyama & Kimishima 2009, Park *et al.* 2008, Park *et al.* 2009, Lee & Park 2005] which is intended as a design method that makes use of both physical and virtual components. The physical components give to the user the possibility to touch the object, while the virtual ones augment the perception of the real objects by superimposing missing parts, details, additional data, the appearance of the objects surface, etc.. The physical components are usually realized by means of Rapid Prototyping.

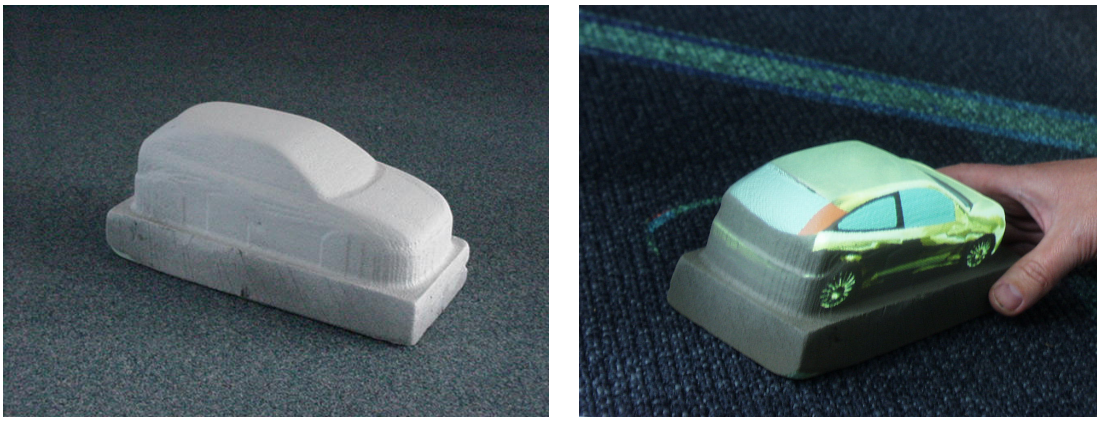


Figure 2.6: *Typical foam model of a car (left) augmented by projection (right) [Verlinden et al. 2003]*

SpaceDesign [Fiorentino *et al.* 2002] is a MR application devoted to the aesthetic design of free form curves and surfaces that incorporates also some design review functionalities based on a rapid prototype augmented with texture visualization (Fig. 2.5), for aesthetic evaluation purposes.

[Bordegoni *et al.* 2009] presented a reference framework for the MP practice. The authors report several examples where this practice has been effectively used for rapid design review of new products and in particular for information appliances. They mainly investigate the problem of positioning information appliances within systems, and for the evaluation of ergonomics aspects of interactive devices.

Verlinden and Horváth ([Verlinden & Horvath 2006, Verlinden & Horvath 2007, Verlinden & Horvath 2009]) describe a multiple case study addressed to assess usability and impact of AR based prototyping technologies. They put in evidence how the shortcomings in using physical prototypes and the bottlenecks in the design process technologies may be reduced using a mixed prototyping approach. They have also proposed a framework for testing and validating *Interactive Augmented Prototyping* that they define as "the combination of physical and virtual artifact models through Augmented Reality and Rapid Prototyping". They adopt the projection-based AR (Fig. 2.6), using projectors to cast computer imagery directly on physical objects [Verlinden *et al.* 2003].

In [Aoyama & Kimishima 2009] the authors present a MR set-up where a tangible

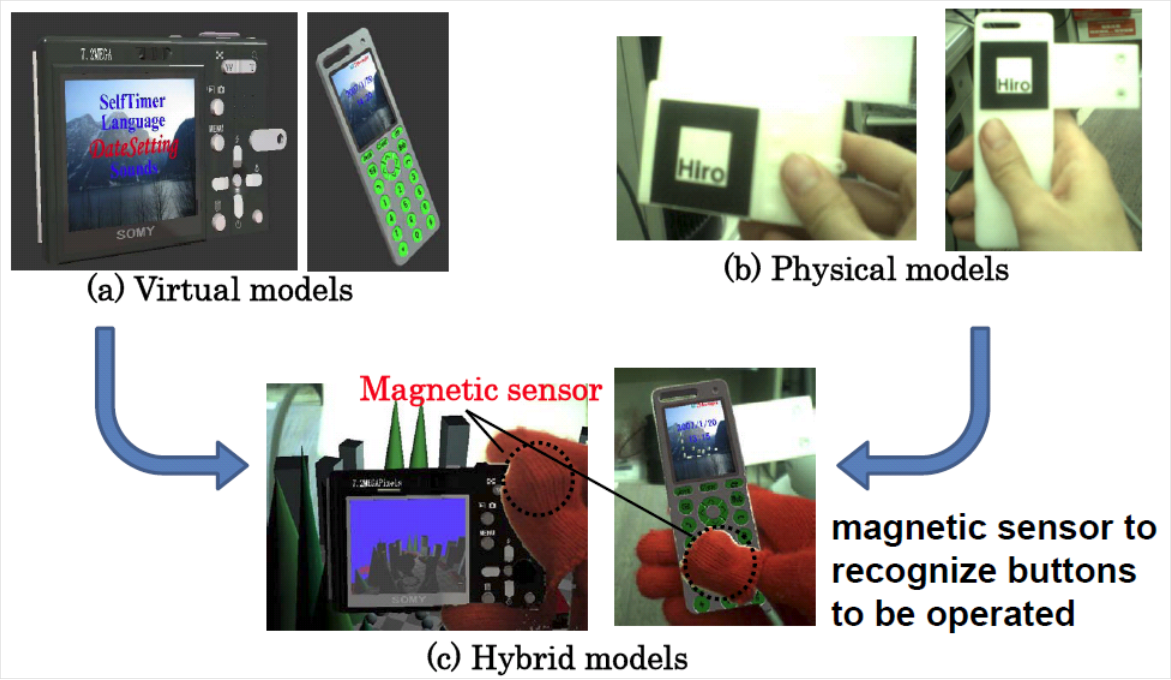


Figure 2.7: *Using a magnetic sensor to determine the intention to press a button [Aoyama & Kimishima 2009]*

dummy, realized in RP, is animated through the augmented reality visualization. The set-up integrates ARToolkit with an electromagnetic tracker in order to determine the position of both the dummy and the right thumb of the user. They focus the attention only on the hand-held products, and they assume that the user has to press the buttons of the product only by using the thumb. The tracking sensor placed on the right thumb (as showed in Fig. 2.7) allows the system to draw a virtual finger that is used both for resolving the occlusion problem and to determine the intention to press a button of the product. Moreover a *pseudo button* is mounted on the thumb surface in order to provide to the operator the feeling of pushing a real button. The tests have given good results, but the method is applicable only on hand held products and requires a tracking device to be implemented. In [Park *et al.* 2008] a study about the use of VR in product design evaluation is presented. The authors focus their work on the analysis of the HMI through a state transition methodology that is addressed to define a behavior model for the functional simulation.

In a recent progress the same authors propose an extension of their approach

adopting the use of tangible interfaces and AR visualization [Park *et al.* 2009]. They use rapid prototyping to create a physical mock-up, as also suggested in [Aoyama & Kimishima 2009], and they paste the markers needed for ARToolkit tracking [Kato & Billinghurst 1999] on the physical model. The interaction with this augmented tangible prototype (as showed in Fig. 2.8) is done through a paper pen on which some ARToolkit markers are placed. The adoption of this pen allows the authors to avoid problems related to the direct interaction through the fingers, but it also ends up to be a limit, because they cannot evaluate essential ergonomics aspects related, for example, to the reachability of the buttons.

In [Lee & Park 2005] the virtual object is superimposed to a blue foam mock-up.

Figure 2.8: *Tangible augmented prototyping [Park et al. 2009]*

The authors use a skin detection algorithm to correct hand visibility, but the employed technique sometimes causes negative visual effects. A similar application for the evaluation of car interiors is proposed in [Ohshima *et al.* 2003]. The authors put in evidence that the possibility for the user to see his/her own hands while touching the steering wheel and seeing the virtual images enhances the feeling of reality and the sense of presence.

Chapter 3

Computer Vision Solutions for Mixed Reality

Existing Mixed Reality (MR) technologies were the basic knowledge on which the researches of this dissertation have been conducted. The first challenge to cope with was the development of a Mixed Reality environment, i.e. the entire set of hardware and software tools used to build the first prototype of a simple MR application. Instead of describing the whole hardware and software architecture, necessary to implement a Mixed Reality environment, this chapter deals only with three original features, which have been developed and then integrated into the MR system:

1. *Marker-based stereo tracking;*
2. *Image-Based background rendering;*
3. *Segmentation based on color detection.*

Contents

3.1 Marker-based stereo tracking

The most common solution to implement an AR system is based on the use of an Head Mounted Display (HMD) and two cameras for the stereoscopic video see-through visualization (as described in 2.1.1). Almost all the systems presented in literature employ one camera to obtain a marker based video-tracking. This technique is widespread because it offers sufficient precision and accuracy.

This section describes an original tracking solution based on a stereo vision approach which aims to use the information coming from both cameras in order to improve tracking accuracy. The proposed solution is also able to switch itself into a standard single camera visual tracking when the stereo pair correspondence is missing (Fig. 3.1), i.e. whenever the marker is visible only by one camera at one time.

After recalling the Zhang's single camera approach [Zhang 2000], I describe first the n-camera mathematical framework and then the calibration methodology developed. Further, a comparison about computational time and accuracy of mono and stereo tracking is reported.

This tracking methodology is based upon a multi-camera generalization of the Zhang method [Zhang 2000], that directly computes the pose matrix of the marker given the projections of the four vertices on both left and right cameras. A new calibration strategy is also defined based on the use of this resection algorithm. Moreover, we implemented also a single camera tracking solution, which is executed when the marker is visible by only one camera.

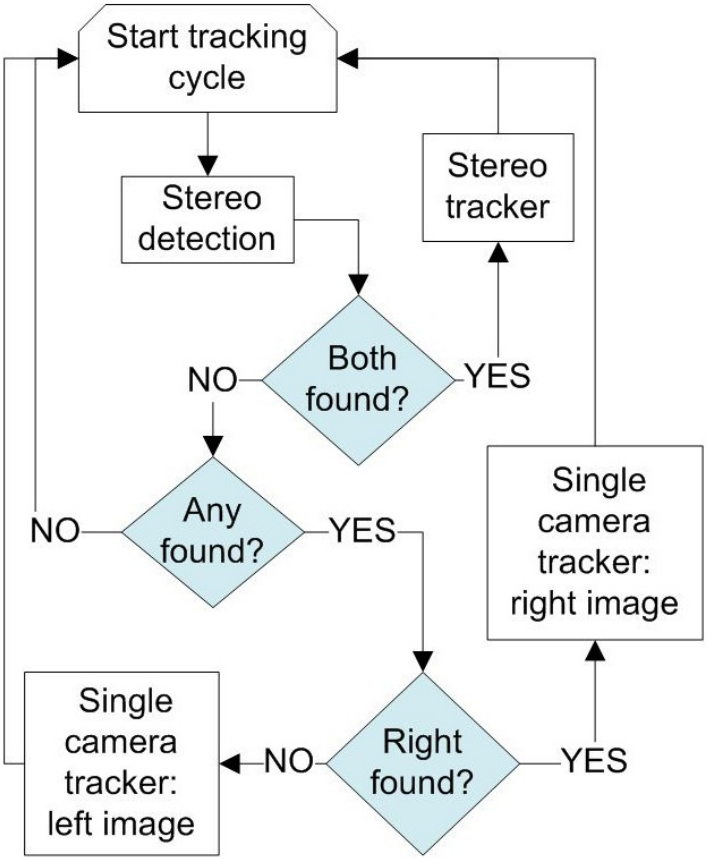


Figure 3.1: *Logic of the stereo tracking strategy.*

Figure 3.1 summarizes the logic of this methodology that basically relies on two different modules: the single-camera tracker and the stereo-camera tracker.

First of all, a pattern recognition algorithm, based upon the well known Artoolkit library [Kato & Billinghurst 1999], is performed in order to find the marker position in both left and right images. Whenever the marker is found in the image, the pixel coordinates of each vertex are stored and organized in a precise order. The conventional order permits to implicitly solve the well known problem of correspondences, and so to simplify the overall procedure.

The system decides which type of tracking to execute (stereo or mono) according to the logic illustrated in 3.1. Only when the marker is visible in both left and right image, the stereo tracking is executed; otherwise, if it is visible by just one camera, the single camera tracking is executed. Nothing is done when marker is not found at all.

The tracking stability, whenever a switching between stereo and single tracking modes occurs, is granted by the accurate knowledge of the relative pose between the cameras. As a consequence, the overall accuracy of our tracking approach, is strongly dependent by the correctness in estimating geometrical (the relative position of each camera in respect of the other) and optical (focal length, aspect ratio, skew ratio, distortion parameters, etc.) properties of the system, more generally indicated as intrinsic and extrinsic camera parameters. For this reason, also a calibration procedure (see 3.1.3), has been developed.

3.1.1 Single camera pose estimation

According to the general pinhole camera model depicted in fig. 3.2, the projection law can be expressed as

$$p = M_p \cdot P \tag{3.1}$$

where P is a generic 3d point vector in homogeneous coordinates

$$P = \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{3.2}$$

and p is its projection in image space coordinates,

$$p = h \cdot \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} h \cdot u \\ h \cdot v \\ h \end{pmatrix} \tag{3.3}$$

with h an arbitrary scale factor.

M_p is the projection matrix, that can be expressed also by:

$$M_p = A\,[R_c T_c] \tag{3.4}$$

where R_c is the 3x3 rotation matrix and T_c is the translation vector that describe the extrinsic parameters of the camera, i.e the position of the camera's frame of reference

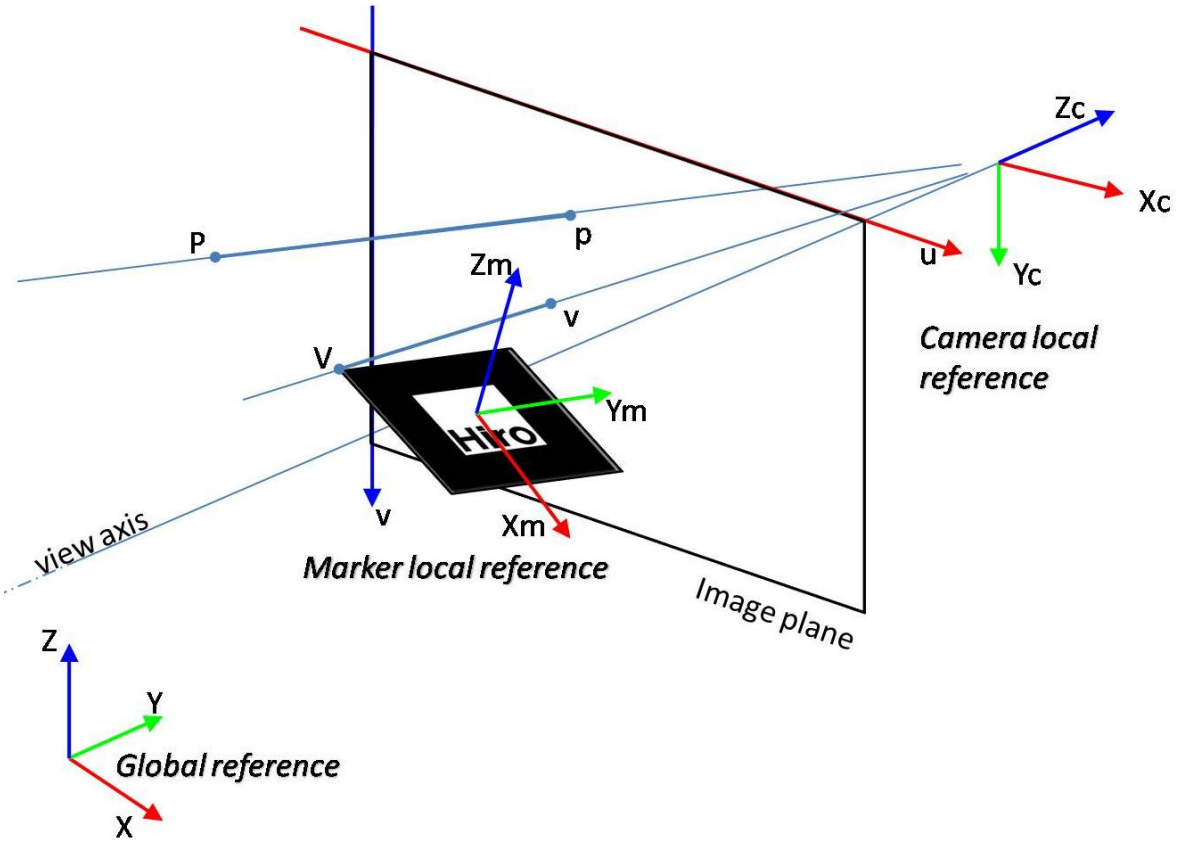


Figure 3.2: *Scheme of the pinhole camera projection model for marker based pose estimation.*

with respect to the world one, computed at each iteration. A is the camera intrinsic matrix, that does not changes on time, because it depends from intrinsic properties of the camera model: f_x and f_y are the horizontal and vertical focal length; s is a slanting parameter and u_o and v_o are the image coordinates of the principal point, that is the intersection between the view axis and the image plane.

$$A = \begin{bmatrix} f_x & s & u_o \\ 0 & fy & v_o \\ 0 & 0 & 1 \end{bmatrix} \tag{3.5}$$

Applying eq.3.1 for a generic marker position, each vertex of the marker, V,

$$V = \begin{pmatrix} V_x \\ V_y \\ V_z \end{pmatrix}, \tag{3.6}$$

expressed in the marker local reference, can be related to its projection, v,

$$v = \begin{pmatrix} v_x \\ v_y \end{pmatrix}, \tag{3.7}$$

expressed according to the camera local reference. After some substitution one obtains:

$$\begin{pmatrix} h \cdot v_x \\ h \cdot v_y \\ h \end{pmatrix} = A \begin{bmatrix} R_c & T_c \end{bmatrix} \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \begin{pmatrix} V_x \\ V_y \\ V_z \\ 1 \end{pmatrix} \tag{3.8}$$

in which R is the 3x3 rotation matrix and T is the translation vector that relates the marker coordinate system to the global coordinate system.

For a planar marker position, the corresponding homography, H, can be computed as described in [Zhang 2000]. This homography maps each point of the marker plane to a point on the image plane. Applying this for a vertex of the marker we get:

$$\begin{pmatrix} h \cdot v_x \\ h \cdot v_y \\ h \end{pmatrix} = H \begin{pmatrix} V_x \\ V_y \\ 1 \end{pmatrix} \tag{3.9}$$

Introducing an auxiliary transformation, with rotation matrix R^h and translation vector T^h, representing the relative 6 dof position between marker coordinates system and camera coordinates system, we can write

$$\begin{bmatrix} R_c & T_c \\ 0 & 1 \end{bmatrix} \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} R_h & T_h \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} r_{h,1} & r_{h,2} & r_{h,3} & T_h \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.10}$$

Considering the case of planar markers it can be assumed that $Vz = 0$. Then eq. 3.8 can be rewritten as a function of the auxiliary transformation, in the form of:

$$\begin{pmatrix} h \cdot v_x \\ h \cdot v_y \\ h \end{pmatrix} = A \begin{bmatrix} r_{h,1} & r_{h,2} & T_h \end{bmatrix} \begin{pmatrix} V_x \\ V_y \\ V_z \\ 1 \end{pmatrix} \tag{3.11}$$

where $r_{h,1}$ and $r_{h,2}$ are the first two column of the R_h 3x3 rotation matrix.

Comparing eq. 3.11 with eq. 3.9 we find a simple way to compute the auxiliary rotation matrix, R_h, in respect of the homography, H, and the intrinsic matrix A.

$$\begin{bmatrix} r_{h,1} & r_{h,2} & T_h \end{bmatrix} = A^{-1}H \tag{3.12}$$

Due to some numerical approximation in computing H, the final R^H is computed according to:

$$R^h = \begin{bmatrix} \frac{r_{h,1}}{\|r_{h,1}\|} & \frac{r_{h,2}}{\|r_{h,2}\|} & \frac{r_{h,1} \times r_{h,2}}{\|r_{h,1}\| \cdot \|r_{h,2}\|} \end{bmatrix} \tag{3.13}$$

where the normalization is needed to increase robustness of the overall computation.

Afterwards, to improve precision in translation computation [Kato & Billinghurst 1999], T_h is computed by solving, in a least square sense (see multi-camera tracking), the over-constrained linear system which can be written considering the couple of equations resulting from the division of the first two lines in eq. 3.11 by the last one, for each vertex of the marker. Then, the pose of the marker can now be computed by replacing R_h and T_h in eq. 3.10.

Moreover, an optimization problem has been defined to get the optimal pose estimation, where the above described methodology was implemented to compute the initial guess solution. In particular, the Rodrigues parameters of the rotation matrix [23] are utilized as optimization variables, and the auxiliary translation vector, T_h, is computed as a function of these parameters. The objective function to minimize is defined as the resulting sum of the squared differences between measured and computed coordinates of all the projected vertices of the marker: the optimization is executed applying a Levemberg-Marquardt standard algorithm for unconstrained non linear minimization. The optimization problem was formulated and then solved using the Optimization Toolbox within Matlab development environment.

3.1.2 Multi-camera tracking

This section describes the developed mathematical framework able to manage a multi-view solution for the tracking problem of a planar marker. For each view we can rewrite

eq. 3.1 as

$$p_{i,v} = Mp_v \begin{bmatrix} r_{1,1} & \dots & r_{1,3} & t_1 \\ \vdots & \ddots & \vdots & t_2 \\ r_{3,1} & \dots & r_{3,3} & t_3 \\ & 0 & & 1 \end{bmatrix} Pm_i \tag{3.14}$$

where i denotes the i-th point, and v denotes the v-th view; $p_{i,v}$ is the projection, in homogeneous coordinates, of the i-th point of the marker, Pm_i, on the v-th view.

Mp_v is the 3x4 projection matrix of the v-th camera, which can be written in the general form of

$$Mp_v = \begin{bmatrix} Mp_{v,1,1} & \dots & Mp_{v,1,4} \\ \vdots & \ddots & \vdots \\ Mp_{v,3,1} & \dots & Mp_{v,3,4} \end{bmatrix} \tag{3.15}$$

Substituting eq. 3.15 in eq. 3.14 and assuming

$$p_{i,v} = \begin{pmatrix} p_{i,v,1} \\ p_{i,v,2} \\ 1 \end{pmatrix}, Pm_i = \begin{pmatrix} Pm_{i,1} \\ Pm_{i,2} \\ 0 \\ 1 \end{pmatrix}, \tag{3.16}$$

the following generic linear equation is derived, where only $r_{k,h}$ and t_k are unknowns:

$$\sum_{k=1}^{3} \sum_{h=1}^{2} \left(Mp_{v,i,k} - p_{i,v,j} \cdot Mp_{v,3,k}\right) \left(Pm_{i,k} \cdot r_{k,h} + t_k\right) = p_{i,v,j} \cdot Mp_{v,3,4} - Mp_{v,j,4} \tag{3.17}$$

with

$$i = 1 \dots np \tag{3.18}$$

$$v = 1 \dots nv \tag{3.19}$$

$$j = 1 \dots nc \tag{3.20}$$

where np is the total number of points, nv is the total amount of views and nc is the total amount of camera.

Considering all the available equations, we obtain an over-constrained linear system, in the form

$$A_{[Nx9]} x_{[9x1]} = b_{[Nx1]} \tag{3.21}$$

with

$$N = nv \cdot np \cdot nc \tag{3.22}$$

N equations and nine unknowns, corresponding to six element of the first two column of the sub rotation matrix and three elements of the translational vector in eq. 3.14. The system in eq. 3.21 is solved in a least squares sense, i.e. computing a solution that minimizes $\|Ax - b\|$, that is the length of the vector $Ax - b$.

As remarked in the mono-camera algorithm at 3.13, a re-normalization is accomplished for increasing robustness. Also for the multi-view approach, a non-linear optimization-based version of the code was also implemented, that utilizes the pose just computed as a start guess solution.

3.1.3 Stereo-Calibration Procedure

The main purpose of the calibration is to retrieve a consistent set of parameters to correctly model the projective geometry of the system, i.e. compute the projection matrix for each camera.

Let P_{left} and P_{right} the projection matrices of left and right camera, referring to eq. 3.4 we can enforce, without loss of generality

$$P_{left} = A_{left} \begin{bmatrix} I & 0 \end{bmatrix} \tag{3.23}$$

$$P_{right} = A_{right} \begin{bmatrix} R & t \end{bmatrix} \tag{3.24}$$

that represent one of the couples of projective matrices definable up to a projective transformation [25]. In this way, we have parameterized the system with a total amount of 16 parameter (5 intrinsic parameters for each camera, and 6 parameters to completely define their relative position).

The calibration problem can be reformulated as the optimization problem of finding the set of parameters minimizing the cost function

$$Z = \sum_{p=1}^{N} \left(p^*_{i,j,p,c} - p_{i,j,p,c} \cdot s_{i,p,c} \right)^2 \tag{3.25}$$

where p and $p*$ are respectively the measured and computed j-th pixel coordinate of the projection of the i-th point of the marker, Pm_i, for the p-th position on the c-th

camera. More precisely, the optimization procedure iteratively computes p^* and p to satisfy the projection constraint

$$\begin{pmatrix} p^*_{i,1,p,c} \\ p^*_{i,2,p,c} \\ s_{i,p,c} \end{pmatrix} = P_c \cdot T \cdot Pm_i \tag{3.26}$$

where T is the pose matrix of the marker with respect to the left camera frame of reference. It is computed, as function of the measured projections and the optimization parameters, applying the multi-view algorithm previously described. The initial guess solution provided to the minimization algorithm can be defined considering the ideal geometry of the system.

3.1.4 Numerical Validation

Both the previously described tracking techniques, stereo (see sec. 3.1.2) and mono (see sec. 3.1.1), have been tested to highlight their benefits and drawbacks. The tests aim, above all, to evaluate tracking accuracy and computational performances of each technique.

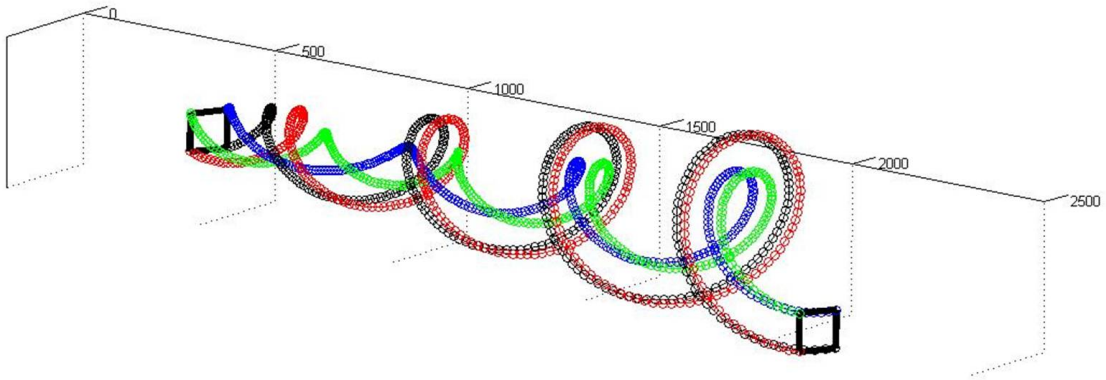


Figure 3.3: *Numerically simulated marker trajectory*

First, we have numerically simulated the movement of the marker. In Fig. 3.3, marker trajectory is showed. The initial and final positions are represented by the

black squares. We have simulated a spiral movement, filling almost the whole field of view of both cameras. In this way it is also possible to evaluate the influence of the distance on the overall accuracy.

Assuming a focal length of 15 mm, and a 6 mm diagonal sensor dimension with a full resolution of 1024 x 768 pixel, the resulting intrinsic matrix, K, has been defined to simulate the projection from real coordinates into image coordinates.

$$K = \begin{bmatrix} 3200 & 0 & 512 \\ 0 & 3200 & 376 \\ 0 & 0 & 1 \end{bmatrix}$$

Accordingly with the design dimensions of the system, the projection matrices have been defined as:

$$MP_{left} = K \begin{bmatrix} & & 30 \\ Ry(-0.06) & & 0 \\ & & 0 \end{bmatrix}$$

$$MP_{right} = K \begin{bmatrix} & & -30 \\ Ry(0.06) & & 0 \\ & & 0 \end{bmatrix}$$

where $Ry(\pm 0.06)$, identifies the sub-rotation matrix defined as a pure rotation of ± 0.06 radians (about ± 3.4 deg) around the y-axis. As a consequence, we obtain a global system of coordinates, placed in the middle of the cameras, representing the system of coordinate with respect to the pose of the marker is measured.

Once the numerical experiment has been performed, it was noted that, theoretically, both our tracking techniques give back an exact solution, up to some numeric computation error. Afterwards, we have introduced a random error in the vertices image coordinates of ś 5 pixels, to simulate a more realistic scenario, and analyze the robustness of the algorithms against noise.

In Fig.3.4-up is presented the translational error for the linear version of the algorithms. The error has been defined as the geometric distance between imposed and computed pose. The tracking accuracy decreases as the distance of the marker from

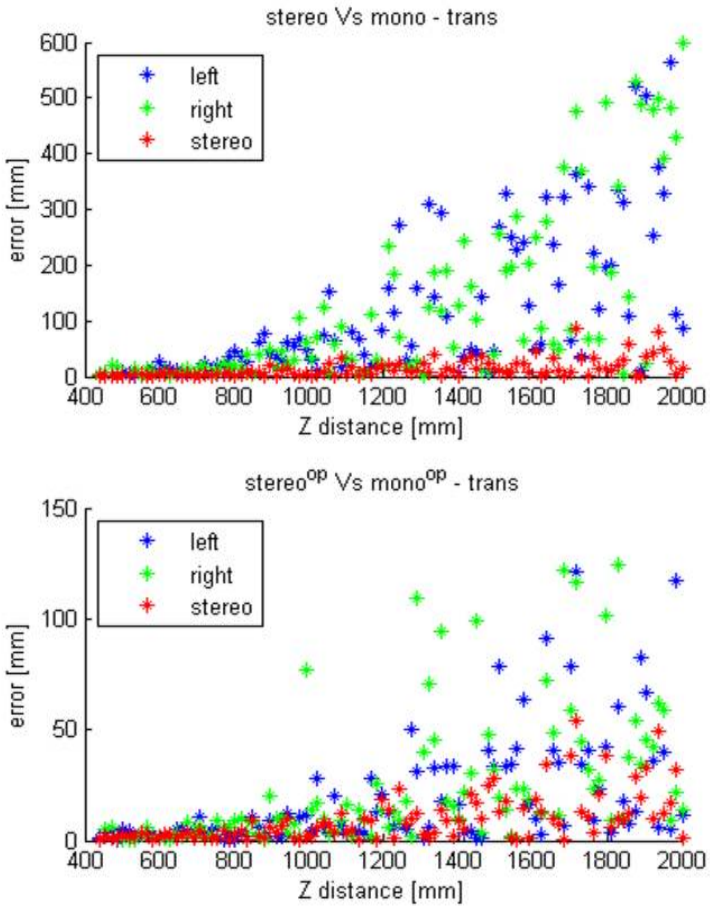


Figure 3.4: *Translation errors: (up) linear algorithm, (down) optimized algorithm*

the cameras increases, but the stereo tracking offers always a better performance (6 time better at 2m distance) than mono-tracking.

In Fig.3.4-down is depicted the translational error for the optimized version of the algorithms. The errors of both techniques are sensibly reduced, especially for mono-camera. However, comparing the linear and the optimized version, we can note that the linear stereo algorithm offers the same range of accuracy of the mono optimized one.

In Fig.3.5 the rotational error is presented. It has been defined as the magnitude of the Rodrigues vector corresponding to the relative rotation between imposed and computed pose. Differently than translational, the rotational accuracy of both techniques is comparable in both linear and optimized version of the algorithms. In the optimized

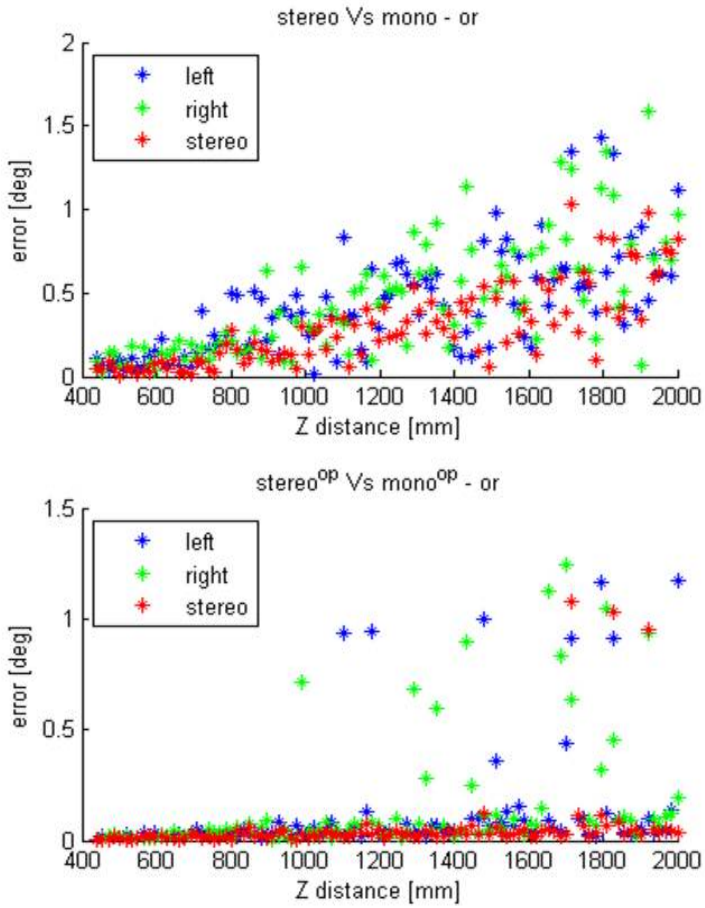


Figure 3.5: *Rotation errors: (up) linear algorithm, (down) optimized algorithm*

version we can put in evidence some sporadic loss of numeric stability, more evident for single camera technique.

Table 3.1 shows the average computational times required for each version of the algorithms. Stereo tracking is faster than mono in non-optimized algorithm and it is comparable in optimized algorithm. Moreover, the stereo tracking offers always better performances, and it provides a more stable solution against noise in image segmentation.

Therefore the presented tracking strategy could be a good alternative to the standard mono-camera tracking algorithms, especially for AR system employing two cameras.

In real-time applications, with no need of high precision, the linear version of stereo-

Table 3.1: *Average computational times*

Time (ms)	Linear	Optimized
Mono	0,83	11,05
Stereo	0,58	12,2

tracking could give almost the same accuracy of a standard single camera algorithm, but it is about 19 time faster.

In conclusion, for all the applications where an high tracking precision is desirable, it is possible to employ the optimized version of our tracking algorithm. The computation period of near 12 ms ($\approx$ 80 Hz) is comfortable compared to the refresh rate of commonly used camera (30 Hz).

3.2 Image-based background rendering

This section describes a technique able to reproduce a photo-realistic image of a scene from a generic point of view, using as input a subset of images of the same scene. In particular, this technique pertains to a collection of techniques and representations, referred in literature as Image-Based Rendering (IBR), that allows 3D scenes and objects to be visualized in a realistic way without performing a full 3D model reconstruction [Shum *et al.* 2007].

The main goal of the IBR methodology presented in this section is to build an implicit 3d model of a scene exploiting the usual set-up of a Mixed Reality environment. The current available alternatives consist in using some expensive devices for reverse engineering (laser scanner, time-of-flight scanner or other similar tools), or directly modeling a virtual replica using a CAD software. In opposite, the IBR solution requires only a commoditiy webcam, and a marker-based tracking algorithm.

The developed methodology can be used to implement some special effects in Mixed Reality: it can be applied to make real objects disappear from the scene by repainting the background on top of the undesired object (as in the new Visuo-Haptic paradigm proposed at Sec. 6.4); another trick could be the possibility to create an immersive

virtual environment, copying the scenario from some real place.

3.2.1 Related Works

The algorithm for image-based background rendering is strongly inspired to the unstructured lumigraph approach [Buehler *et al.* 2001]. The scene light field is considered to be known for a set of irregularly distributed rays, and the method uses view-dependent texture mapping [Debevec *et al.* 1996] to interpolate light field data from those rays. More precisely, the algorithm computes camera blend weights for a discrete set of vertexes on the occluded region, then meshes the vertexes and defines a camera blend field over the occluded region, and finally computes the final image by blending the results of view-dependent texture mapping. Two types of input data are used in a combined manner: a set of prerecorded images of the background scene together with their associated camera positions, and a very rough geometric approximation of the background.

I describe the full IBR algorithm in four steps: (i) description of the geometric proxy, (ii) sampling of the occluded region with known camera rays, (iii) meshing of the occluded region, and (iv) efficient view-dependent texture mapping.

Unlike in the Unstructured Lumigraph Rendering, the number of input cameras selected for each sampled vertex is reduced to one, in order to simplify implementation and avoid blur effects. This was a substantial simplification which resulted in a lowering of the graphical quality. However, this loss is compensated by the fact that the user is mainly focused on the virtual objects that are added into the scene, and a quasi-realistic background, instead of a photo-realistic one, may be sufficient to not cognitively disturb his/her perception.

3.2.2 Geometric Proxy of the Background

As a pre-process, I construct a very rough geometric proxy of the background, inspired by the examples given in [Buehler *et al.* 2001]. This proxy consists of planar desks, planar walls, and an average-depth plane that approximates other irregular surfaces.

For example, in my test setting in , I use a plane for the desk, another plane for the left wall, and a third plane that approximates the window area on the right.

Then I regularly sample the proxy planes. These samples, after being projected onto the image plane, will be used as vertexes for view-dependent texture mapping, as described in the next section. The geometric proxies serve as depth estimates for defining the homography transformation of view-dependent texture mapping.

3.2.3 Sampling the view-dependent texture map

Following the unstructured lumigraph rendering approach, a camera blend field for the view-dependent texture map would be sampled with two types of vertexes:

Geometry vertexes. They are the projections onto the image plane of the points sampled on the geometric proxies, used for defining a homography transformation between the current camera plane and the one of each prerecorded image.

Camera vertexes. They are the projections of camera centers of the prerecorded images onto the image plane. A camera vertex is discarded if the line of projection does not fit in its field of view. Camera vertexes exploit epipolar consistency [Buehler *et al.* 2001]. A camera vertex needs a depth estimate for computing homography transformations. One can obtain this depth estimate by interpolating the depths of nearby geometry vertexes in image-space.

3.2.4 Meshing and Camera Blend Field

Unlike what is proposed by the Unstructured Lumigraph, each vertex of map is associated to only one of the input cameras.

For each geometry vertex, we select the camera with smallest angle w.r.t. the line of projection.

For a camera vertex, the camera should be trivially selected as the one that produced the vertex itself. However, due to our simplification, the use of Camera vertexes is indirectly obtained by the effect of the geometry vertexes. In particular, I assume that whenever a Camera vertex is projected on the image plane, the geometry vertexes

in the nearby will select the same input camera corresponding to the Camera vertex. Therefore, for each geometry vertex, the blending weight of the associated camera is defined as one, and the weights of all other cameras as zero. The blending field for the whole map is defined by meshing the vertexes and interpolating the camera blending weights inside each triangle.

This allows to construct the mesh avoiding constrained Delaunay triangulation of the vertexes, because there is no need to add Camera vertexes to the original geometry mesh. In facts, the triangulation is defined once at the beginning through the proxy geometry model. In practice, only geometry vertexes need to be analyzed according to the current camera position.

3.2.5 View-Dependent Texture Mapping

Given a triangle of the mesh, with vertexes $\{a, b, c\}$, we compute the output image, O, inside the triangle by warping and blending three of the prerecorded background images, B_a, B_b, B_c. In particular, for each vertex we compute three homography transformations, H_a, H_b, H_c, based on the position of the vertex, the current pose of the camera, and each of the associated input camera pose. Given these transformations, the texture coordinates T_{v2B} on each input image, $B \in \{B_a, B_b, B_c\}$ is computed for each vertex, $v \in \{a, b, c\}$. This produces a total of nine texture coordinates pairs, $T_{a2B_a}, T_{a2B_b}, T_{a2B_c}, T_{a2B_a}, T_{a2B_b}, T_{a2B_c}, T_{a2B_a}, T_{a2B_b}, T_{a2B_c}$, used for warping each input image according to the output image plane. Finally, the blending weights are defined to make a linear interpolation of the input images.

To exploit the interpolation capabilities of graphics hardware the method was implemented with a fragment shader program that directly interpolates input parameters inside the triangle, once properly defined the vertex properties.

Therefore, the blending weights are trivially defined as three vectors: $W_a = \{1, 0, 0\}, W_b = \{0, 1, 0\}, W_c = \{0, 0, 1\}$. These are passed to the shader troguhFor each vertex

void CfHVH(float3 V1C : TEXCOORD1, float3 V2C : TEXCOORD2, float3 V3C : TEXCOORD3, float4 Alpha: COLOR,

out float4 color : COLOR,

uniform sampler2D dec1, uniform sampler2D dec2, uniform sampler2D dec3) float4 C1 = tex2D(dec1, V1C.xy); float4 C2 = tex2D(dec2, V2C.xy); float4 C3 = tex2D(dec3, V3C.xy); color = C1 * Alpha.x + C2 * Alpha.y + C3 * Alpha.z; color.w=Alpha.w;

Given the camera blend weights for the three vertexes of the triangle, we again exploit hardware interpolation to define blending weights for the pixels inside the triangle, and blend the warped input images defined by the three vertexes. I have implemented view-dependent texture mapping on the GPU, with a simple pixel-shader for blending. The shader has been implemented on NVIDIA's Cg shader language.

3.2.6 Results

In order to analyze the proposed methodology for background rendering, I used a simple test case, where I compared the actual scene image with the synthetic image of a simple scenario (Fig. 3.6-top-left). The scene geometrical proxy was defined considering three planes (see Fig. 3.6-top-right): one for the desk, where I place the multi-marker used for tracking; one for the book-side, and one for the pictures-side. All the proxy was modeled as a triangular 3D mesh, defined in respect with the world reference frame given by the AR application (see the augmented red cube on top of one of the markers). Given the position of the camera, the method computes the camera blending field choosing, for each vertex of the visible mesh, an input texture image from the pre-captured set of background images. In Fig. 3.6 the camera blending field is graphed, using a different color for each different input image. The absolute difference between the actual image of the scene (Fig. 3.6-top-left, directly coming from the camera) and the synthetic image (Fig. 3.6-center-left, made using the proposed method) is showed in Fig. 3.6-bottom-left. The showed image is obtained first computing the absolute difference on each color channel separately, and then considering the inverted resulting grey-level image where white means no difference. Finally, Fig. 3.6-bottom-right reports the same difference after a contrast-enhancement operation, and with the blending field added as a color overlay. For the reported frame, the view-dependent textured proxy is obtained using only three images as input (red in the middle part, blue in the lower part, and

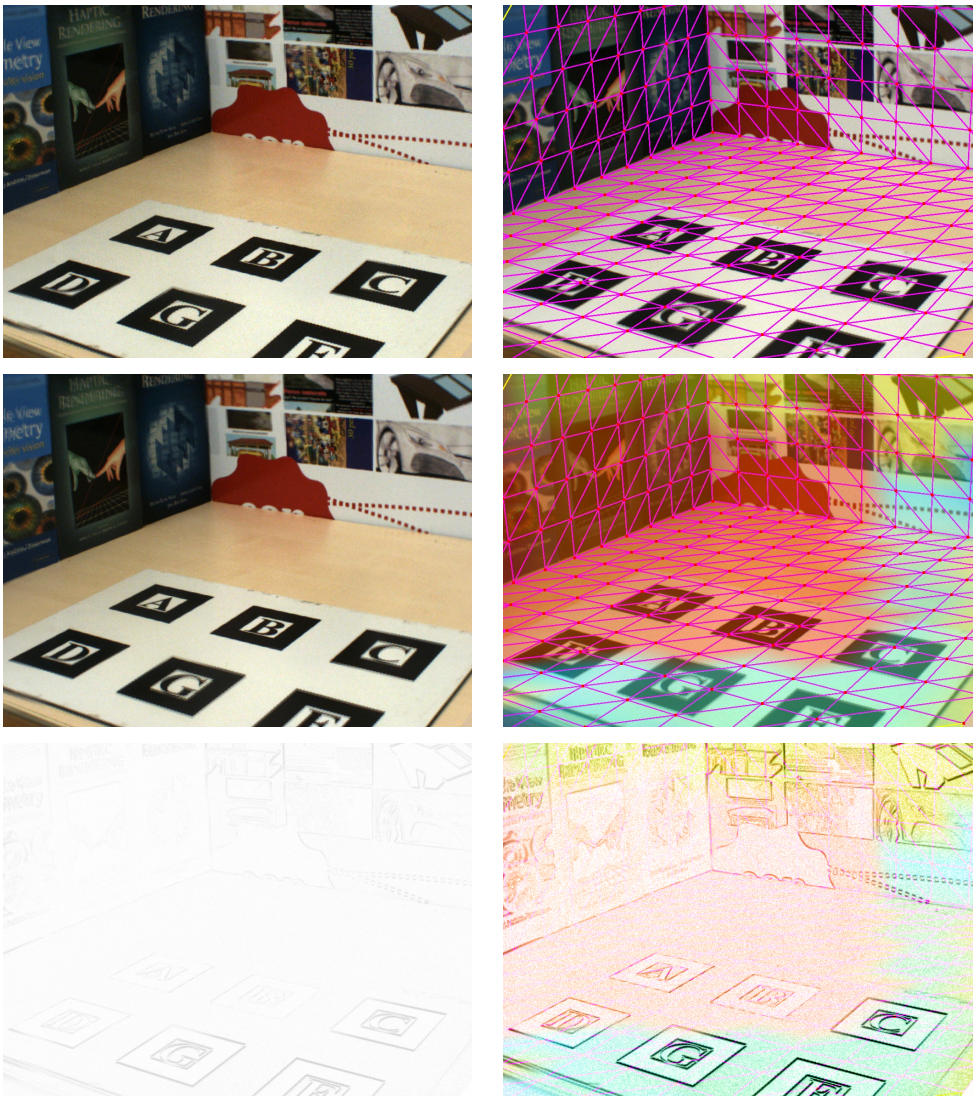


Figure 3.6: *Image-Based Background Rendering: frame 1. The 3D triangular mesh used as geometric proxy is projected on the current image plane (up-right), to obtain the camera blending field (center-right): each color means a different camera source. The difference between the true scene (up-left) and its synthetic image (center-left) is computed via absolute difference on each color channel (bottom-left). (bottom-right) is the same difference after some contrast enhancement, and a color overlay added to analyze relations between errors and the camera blend-field.*

yellow in the upper part). Analyzing the error distribution one can note that the errors, considered proportional to the thickness of the black lines, drastically increase out of the red region, especially in the blue region near the markers. Similar result are

shown in Fig. 3.7, where the same analytic approach was applied for a different position within the same scenario. From a qualitative point of view, the method suffers some evident ghosting effects (see the book titles) but is able to render a realistic replica of the background.

In conclusion, the discontinuity of the errors among different input images suggests some lack of accuracy due to the applied tracking methodology. A better result could be obtained using a more robust tracking method, or even adopting a bundle-adjustment approach over the input image set. Moreover, further improvements could be obtained adopting a more realistic geometric proxy, or even implementing some automated 3d model reconstruction method (as recently proposed in [Newcombe & Davison 2010]).

3.3 Segmentation based on color detection

Usually, the correct visualization of the user's hand in the MR environment requires a segmentation step, able to separate the hand from the background. This section describes a color-based detection algorithm, able to provide this segmentation step, that was later exploited for two different applications: first, it was used to develop an original method that correctly manages occlusions between virtual objects and the user's hands (Sec. 4.2.4); then the method was finally extended to the wider concept of the *perceptually consistent* compositing (Sec. 6.6).

Considering the simple segmentation challenge, tens of works have investigated this issue by analyzing the camera images in the different color spaces. Several techniques are proposed in literature and there are some comparisons [Vezhnevets *et al.* 2003, Shin *et al.* 2002, Kakumanu *et al.* 2007, Phung *et al.* 2005] that put in evidence pros and cons of each algorithm.

I adopted the HSI (Hue, Saturation, Intensity) color space to overcome illumination problems such as the high intensity at white lights, the influence of ambient lights or the surface orientation in relation to the light source [Kakumanu *et al.* 2007]. The first operation of the algorithm is a transformation from RGB to HSI [Gonzales & Woods 2008]. I have adapted the Single Gaussian model described in

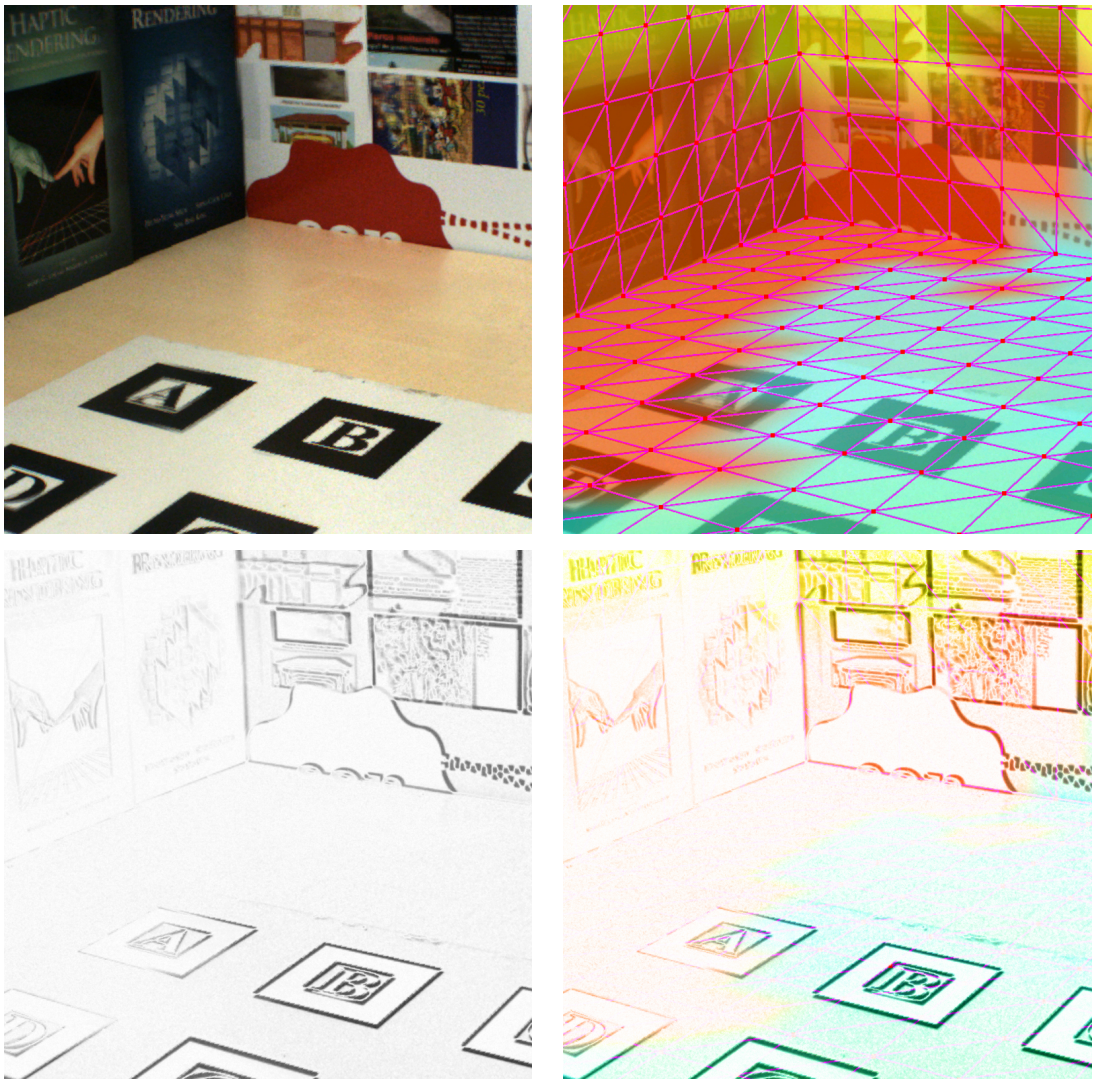


Figure 3.7: *Image-Based Background Rendering: frame2. The rendering of the background (up-left) is obtained after computing the camera blend field (up-rigth). I plot both the absolute difference with the true scene (bottom-left), and the contrast-enhanced version (bottom-right), with a colored overlay of the blend field. See Fig. 3.6 for more details.*

[Yang & Waibel 1996] to classify the hand region, simplifying the equation to keep the algorithm easily implementable in a shader program.

The resulting color detection procedure consists of computing the following quantities for each pixel:

$$\delta H = \frac{(H - H_s)^2}{K_h}, \delta S = \frac{(S - S_s)^2}{K_s}, \delta I = \frac{(I - I_s)^2}{K_i}$$

where H, S and I are the components of the color of the current pixel; H_s, S_s and I_s are the components of the skin color and k_h, k_s, k_i are the weights used to adjust the sensitivity of the respective component; finally the pixel is marked to be segmented if

$$\exp(-\delta H - \delta S - \delta I) \leq t$$

where t is a threshold value.

3.3.1 key-color calibration

The proposed method computes a binary segmentation of the image, distinguishing between the key-region, i.e. the region where color is similar to the key-color, and the non-key-region.

As detailed in the previous section, the procedure requires to define the key-color $K_C = (H_s, S_s, I_s)$, whose components reflect Chromaticity, Saturation and Intensity values, according to the HSI color space. Given the key-color, the algorithm also needs to know the weighting values, k_h, k_s, k_i and the threshold value t.

To perform this calibration task, I designed and implemented an interactive application. The user is requested to manually tune up the required parameters, while the application show him/her a preview of the current filtered video. At the end, the final state of the filter is stored for further use in any kind of application. This calibration approach permits to customize the environment for different users, different light conditions, different scenarios, thus resulting in a good flexibility.

3.3.2 Segmentation results

Figure 3.8: *Qualitative performance of the color-based segmentation method.*

Chapter 4

Hand-based Interactive Mixed Prototyping

This chapter presents a Mixed Reality environment in which the product behavior is simulated using the same models and the same software employed by the engineers in the design phase. This Mixed Reality Environment aims to make the user able to naturally interact with the mixed prototype, through two original features:

- *the ability to properly manage the occlusion between user's hand and virtual objects,*
- *the interpretation of the hand gestures that a user accomplishes during his/her interaction with the elements of the product interface.*

The final goal of the development of this system is to provide the user the possibility to naturally *interact with a digital prototype, using his/her own hands. ITs easiness of use was validated through some usability tests, conducted with users (as reported in Chapter 5).*

Contents

4.1 Introduction

Virtual Reality (VR) is widely considered a valuable tool for the simulation of virtual prototypes, because it allows engineers to enhance the analysis and validation of the digital product, before manufacturing any physical mock-up. In particular, VR has demonstrated to be able to support and facilitate the participatory design of industrial products, because it allows designers to involve the final users since the early design stages, without the need of a physical mock-up and with the advantage of being able to assess several design options [Bruno & Muzzupappa 2010]. Unfortunately VR still requires complex and expensive hardware like tracking systems, data-gloves, stereoscopic visualization systems. Moreover the user should perceive the tactile feedback while s/he is interacting with product, thus requiring the use of haptic devices.

In Sec. 2.3 some of the most interesting papers about mixed prototyping are presented, putting in evidence the main problems that usually arise in the application of this approach. Among all issues, I focused on three aspects: simulating the product behavior; managing the occlusion between the user's hands and the virtual object; tracking the movement of the user's fingers in order to make him/her able to interact with the mixed prototype.

The following section present some original solutions to all these problems, that are integrated in a Mixed Reality environment particularly designed for user interaction with a mixed prototype. Concerning the behavior simulation of the mixed prototype, I use a software tool for the run-time connection between the MR software and the simulators used for product design and analysis. This means that there is no need

to write code for describing the product behavior, and any modification done on the behavior models is immediately testable in MR.

The proposed environment has been applied to a real industrial case: all the implementation details are given to prove the applicability of the proposed Mixed Prototyping environment.

4.2 The Interactive Mixed Prototyping approach

The approach proposed in this chapter for the creation of an interactive mixed prototype is based on the idea that the most effective way to replicate the behavior of the product inside a virtual environment is by coupling a VR software with a simulation environment. This idea is mainly justified by the fact that, in many cases, the behavior model of the product is already available since the first steps of the product development process. In fact, control engineers usually realize a behavior model of the product using a Computer Aided Control Engineering (CACE) tool. This model is used not only to schematize the product behavior but also to analyze how it works with a proper simulator.

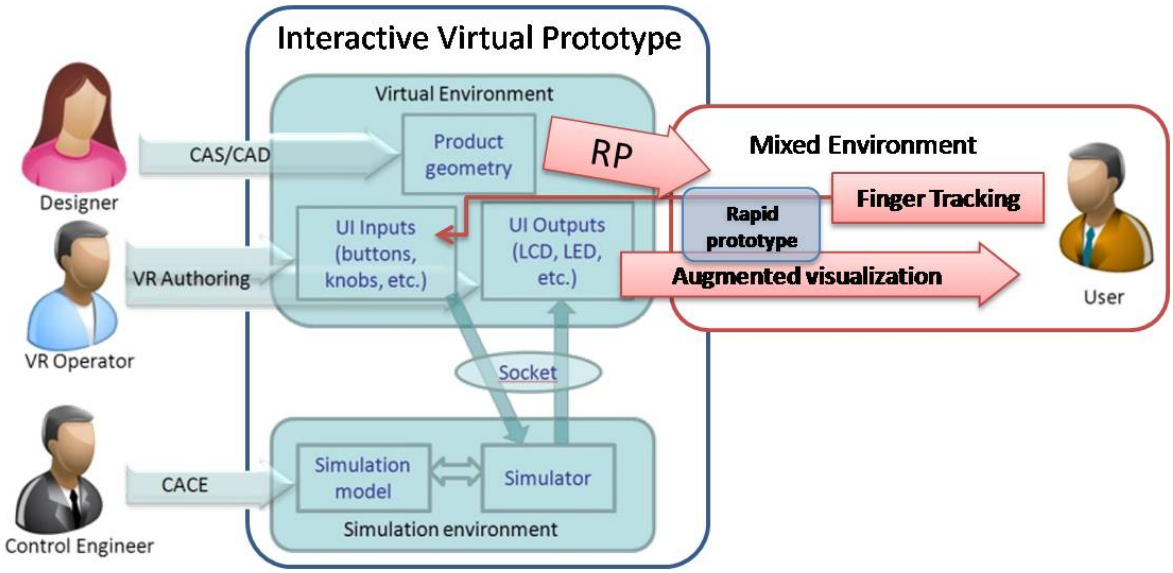


Figure 4.1: *Interactive Mixed Prototyping: a participatory approach*

Figure 4.1 schematizes the proposed approach putting in evidence the role of the

four actors involved in the process. The **Designer** defines the geometries of the product and in particular the external shape, that is essential from the aesthetical point of view, and the appearance of the interface.

The geometric model is used for the creation of the physical prototype, usually done through rapid prototyping, and for the creation of the virtual model that is superimposed to the physical one in order to improve the visual perception.

The **MR Operator**, , uses an authoring tool to prepare the virtual environment specifying which *input elements* of the interface (button, knob, etc.) may be controlled by the user, and which *output elements*(LED, LCD, etc.) may change their status during the simulation. On the other side the **Control Engineer** defines the behavior of the product , and connects the state transition of the model with the user generated events and the state variables with the *output elements* of the User Interface (UI) defined in the virtual environment. This connection, allows a data exchange between the two environments, so that the simulation of the virtual prototype is managed by the same model that the control engineer uses for his/her design and analysis activities.

The three actors that we have identified on the left part of Fig. 4.1 have to be intended as *roles played in the design process*. In some cases, e.g. in small enterprises, some of these roles may be played by the same person. For example, the MR environment may be prepared by the designer or by the control engineer. Extending this reasoning, we may assume that all the three roles may be played by only one actor with all the competences.

The final user is assumed to be one of the actors of the process because, as clearly evidenced in literature [Bruno *et al.* 2007, Barbieri *et al.* 2008, Aoyama & Kimishima 2009, Bordegoni *et al.* 2009, Park *et al.* 2008, Park *et al.* 2009], MR tools are particularly effective in virtual prototyping when the user is involved in the testing phase. In other terms, this approach allows the user to test what the engineers have designed by a direct interaction with the digital mock-up.

At the end of this chapter, a case study is presented to demonstrate how it is possible to realize an interactive virtual prototype connecting a simulation software with a MR environment. In particular I use the Matlab/Simulink environment to model

and simulate the product behavior and Virtools Dev to create the mixed environment.

4.2.1 The link between Behavior Simulation and MR

The link between Simulink and Virtools has been created by adapting an existing software library called SimLib that was previously developed for dynamic simulation in VR [Bruno *et al.* 2009]. This library provides an easy to use Inter-Process Communication (IPC) channel using TCP socket, therefore it is possible to run the simulator and the VR application on different machines. The SimLib library is quite easy to use. It has few functions, implementing the code for the TCP/IP communication and synchronization. Therefore, the developer must not take care of sockets and threads. The SimLib has been used to develop the customization for both Simulink and Virtools.

To integrate the SimLib inside Simulink I have developed two custom S-Functions. An S-Function can be used within a Simulink model as a conventional Simulink block, with a user defined behaviour and a set of actions. For my goals, the S-Functions are customized for the communication between Simulink and Virtools through the IPC channel provided by the SimLib. One S-Function receives the events generated by the user that interacts with the virtual product and switches different simulation parameters according to these events. The other S-Function sends the data that describing the state of the product and each change that occurs during the simulation. The data sending is based on an asynchronous channel: the SFunction sends data to Virtools, without stopping the simulation. In order to obtain a consistent visualization, in fact, it is necessary to simultaneously send all data at each time-step.

The Virtools socket connection for the IPC has been implemented through two user defined Building Blocks (BB). A BB is something like a black box with some parameters in input and some in output. By connecting several BBs, it is possible to create an interactive application in Virtools. The two BBs we have implemented to support the connection with Simulink are able to send and receive data through the IPC channel. The first BB sets the 3D model parameters in agreement with the data coming from Simulink. The second BB sends to Simulink the messages generated by the user interaction in the VR environment (e.g.: a button pressed, a knob rotated,

etc.).

4.2.2 Creation of the virtual prototype

I assume that the Simulink model can be realized following a top-down process where the product behaviour is progressively refined until each component of the product has been defined. For example, the model realized for the test case can be considered as the first stage of the design process. In the subsequent steps of the process the model could be refined transforming the behavior model (made of functions, formulas and equations) in a schema where each block represents a physical component (e.g.: electric motor, actuators, processors, signal converters, etc.).

The geometric model of the product is usually defined following a bottom-up approach. Using a CAD system, the engineers model each single component or import them from previous projects or from the suppliers catalogues. But in order to realize a virtual test with the product interface, only the external shape of the product is needed. All the components inside the object, that are not visible to the user, can be omitted in the virtual prototype. This allow engineers to start the test with the users as soon as the *skin* of the product has been defined by the designers.

When the shapes are ready, these are imported in the virtual environment and the properties of the interface elements are connected to the state variables of the simulation model. Then the events generated by the user are addressed to the simulator that considers them as triggers for the state transitions. In most cases the connection between the simulator and the VR software does not change during the process, because the exchanged data remain the same also when the behaviour model is updated and refined.

In the general case, at the end of the design process, several different models are defined for each engineering domain involved in the product. As an example, you may consider the test-case (see Sec. 4.3) , where you will find the Simulink model that simulates the electric and electronic behavior and the virtual prototype with all the CAD geometries of each part assembled in a unique digital mockup. The latter could also be used to realize multi-body and/or structural analyses using CAE software. Differ-

ent models can also be used together to realize a mechatronic co-simulation coupling different simulators [Bruno *et al.* 2007] or using an integrated simulation environment.

4.2.3 MR as a link between VR and user

The hardware set-up required for building the MR system is very easy to realize and it can be implemented with a very low budget. A video see-through HMD is used for augmented visualization.

The HMD (Fig. 4.2) is composed of an iWear VR920 and a CamAr, both from Vuzix. The CamAr has a resolution of 800x600 pixels at 30Hz. The camera is used both for the video see-through function and for marker-based tracking through the ARToolkitPlus library [Wagner & Schmalstieg 2007].

A colored glove (Fig. 4.3), equipped with a micro-switch placed at the index finger tip, is used both to solve the occlusion problem and to improve the interactivity. The micro-switch provides a force-feedback reaction when the user is pressing a virtual button, so s/he can feel the sensation of touching a real button on the physical prototype. The micro-switch signal is transmitted to the PC trough a wireless connection. Compared to the other solutions available in the state of the art (Sec. 2.3), this MR environment does not require any pen [Park *et al.* 2009] or tracking device [Aoyama & Kimishima 2009] to recognize the user actions while s/he is interacting with the digital product.

The rendering pipeline designed for the application rely on the segmentation component described in Chapter 3. In particular, it works in the following way:

Figure 4.2: *Vuzix HMD solution for AR: iWear CamAR bundled with iWear VR920*

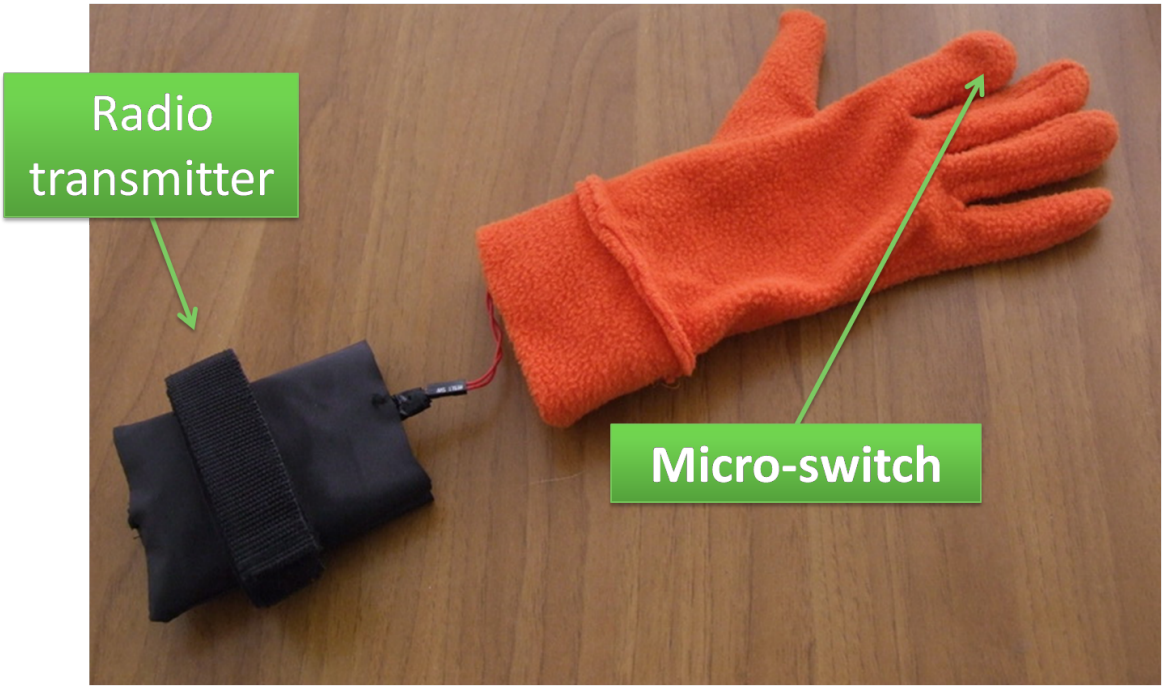


Figure 4.3: *Mixed Prototyping glove: a colored glove with a micro-switch placed at the index finger tip*

1. an image of the scene is acquired and displayed on the background;

2. the tracking identifies the marker and updates the position of the virtual prototype that is rendered in the proper position;

3. the hand-region is detected in the current frame and stored as a single bit hand-mask;

4. the hand-region is displayed on the foreground;

5. the index finger tip is tracked applying a blob detection procedure on the hand-mask;

6. the final image is then displayed on the HMD.

In particular, steps 3 and 4 are accomplished according to the occlusion handling strategy described in Sec. 4.2.4.

Figure 4.4: *Occlusion Handling Details of Mixed Prototyping: the raw image from the HMD-camera (left) is composed with (right) and without (center) an occlusion handling strategy*

4.2.4 Managing the occlusion of the user's hand

This section presents a skin-detection method that relies on the color-based segmentation approach previously described in Sec. 3.3.

Similarly to the work of [Ventura & Hollerer 2008], the method aims at the particular but very common case where a real occluding object, e.g., the user's hand, lies between the user's viewpoint (the camera) and the virtual objects. However, the method does not rely on depth reconstruction, to the benefit of real-time performance. In opposite to the methodology presented by Lee and Park [Lee & Park 2005], where a physical prop of the virtual object colored with a special hue is used, I use a skin-detection method that does not require the use of physical props, with a great benefit in term of flexibility.

The method correctly manages occlusions between virtual objects and the user's hands. In Fig. 4.4 the result of the adopted algorithm is shown. It is apparent that the segmentation is not perfect, but it would be acceptable for the interaction purpose. Moreover, a better results could be obtained exploiting a simple hole-filling algorithm,

that pertain to the wider morphological image processing set. However, at this moment a strong focus is given on the performance than on the accuracy of the segmentation. This is achieved trough a shader implementation, that needs less than 1ms to process and display the hand-region.

Note that the illustrated results (Fig. 4.4) were obtained in quite ideal conditions: a perfect illumination, high-saturated images, an high contrast background without any color chromatically similar to that of the skin. Although the technique is able to detect the skin region, a colored glove was used during the tests in order to improve the precision of the hand recognition without disturbing the actual purpose of the tests.

4.2.5 User's natural interaction: tracking the position of the user's index finger tip

As evidenced in the state of the art the interactive mixed prototyping requires the possibility to interact with the elements of the product interface. There are three methods currently investigated in literature: a tracking device mounted on a finger tip [Aoyama & Kimishima 2009]; a tracked pen [Park *et al.* 2009]; haptic-simulated elements placed on the physical prototype [Bordegoni *et al.* 2009]. In order to develop the finger tracking strategy, it was observed that when the user approaches the product interface to push a button, s/he tends to maintain the natural position of Fig. 4.5, with the index finger completely extended. Therefore, for the specific task of pushing a button, I assume that the index finger, used to interact with the Mixed Prototype, is visible, and therefore *detectable*, into the first-person view image.

The adopted tracking strategy determines the position of the user's index finger tip in the screen space coordinate system through a simple detection algorithm, assuming that it is localized to a left-up position in respect with the rest of the hand.

The segmented image, computed with the algorithm described in Sec. 3.3, is analyzed: starting from the up-left corner, the procedure searches, within each row of the image, for the first horizontal sequence of k pixels, where k is a parameter that is manually tuned according to the thickness of the finger, expressed in pixel. The tip of

the finger position is assumed to stay at the center of the first found sequence.

This 2D position is used to estimate where the user is pointing, through a ray casting method (Fig. 4.6).

The computed position is back-projected towards the mixed prototype, according to the current view projection matrix depending from the relative position of the user in respect with the object. Whenever this ray intersects an interactive object of the product interface (like button, knob, etc.), that object is enabled, but not yet activated, for interaction.

When the user attempts to interact with the prototype, and in particular to push a

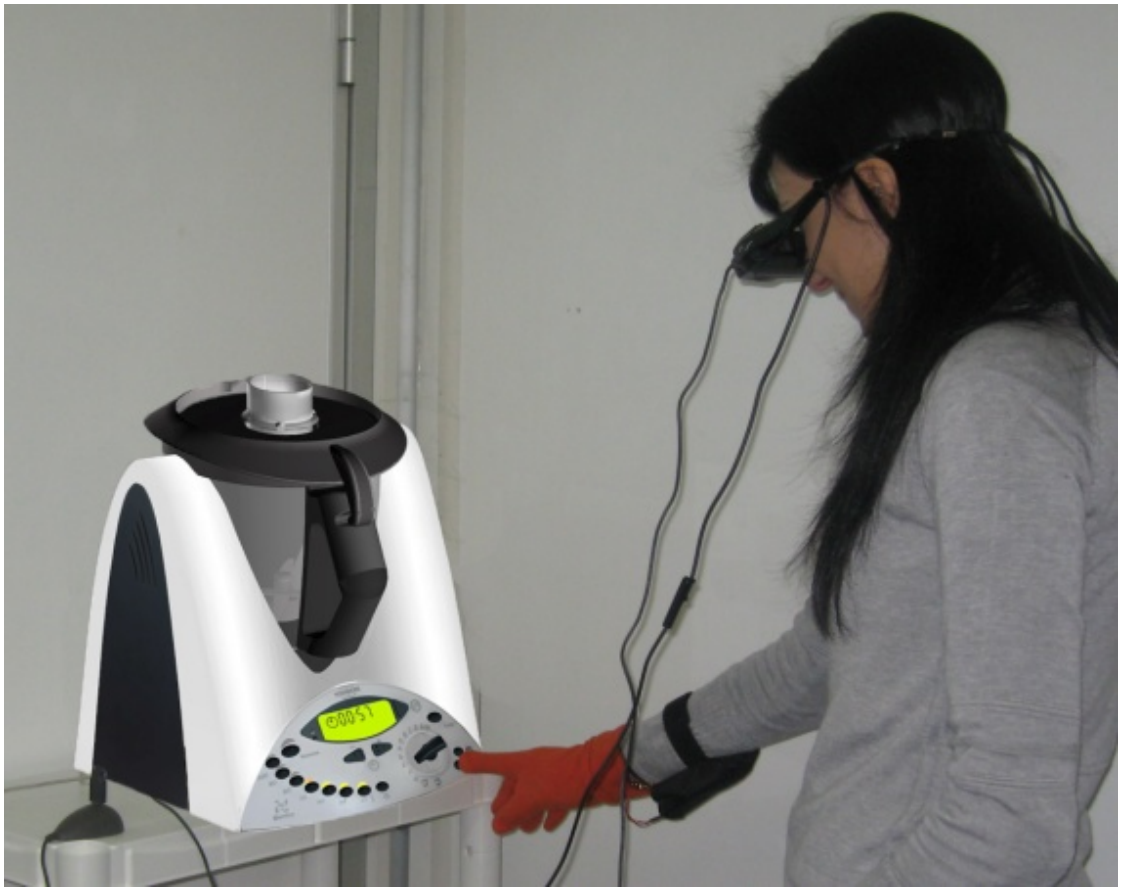


Figure 4.5: *When a user approaches the product interface to push a button, s/he naturally tends to maintain the index finger completely extended toward the button. The mixed prototype is added to the picture to simulate the scene perceived by the user.*

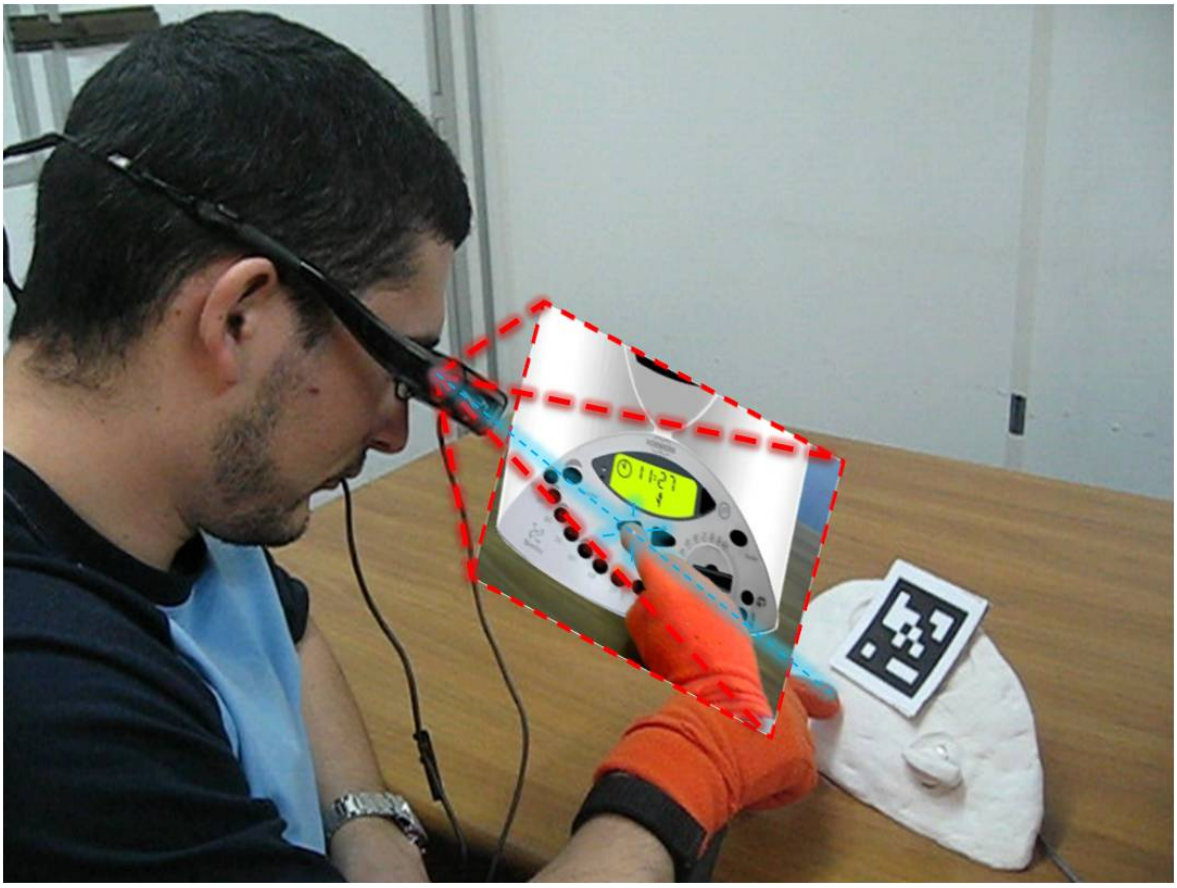


Figure 4.6: *Finger tip tracking for Mixed Prototyping*

button of the virtual product, his/her index finger impacts on the physical prototype, activating the micro-switch mounted on the tip. This activation triggers the activation of any button enabled by the ray casting, thus allowing the user to push the pointed button of the product.

In order to speed-up the finger tracking process, and assuming the hypothesis of slow motions, the algorithm attempts to reduce the search area to a square of 200 x 200 pixels, centered around the position of the blob in the previous frame. This considerably reduces the computational time: when the finger is found inside the square the computational time is reduced from 10 ms to 1.6 ms, thus increasing the frame rate of the application up to 60 fps.

4.3 The Test Case

The test-case analyzed to validate the *interactive mixed prototyping* approach is a hot mixer (Fig. 4.7). It is a quite complex appliance with several functions like weighing, blending, grinding, kneading, steaming and cooking.

The product presents a classic UI with a knob that allows the user to set the speed of the motor, eight buttons to set the temperature, a couple of buttons to set the timer, the stand-by button, and other three buttons for special functions: turbo, weighing, kneading and counter clockwise rotation. It has also a LCD that displays the state of the mixer (timer, weight, functions activated) and a set of eight LEDs, associated to the temperature buttons, that show the current and the desired temperature.

The geometric modeling phase has been done following a reverse engineering approach. A photogrammetric software has been used to draw the profile curves that have been exported to a CAD where the surfaces have been created. The model has been completed with materials and textures acquired with a digital camera, then imported in Virtools. To mimic the touchable user interface of the product, the resulting shape model was also used to build a rapid prototype. After putting a planar-marker on top of the rapid prototype (see Fig. 4.6), the VR operator was required to manually tune the reference of the model in order to spatially align the augmented virtual model in respect to its real partial shape.

4.3.1 Modeling the product behavior

The realization of an interactive virtual prototype implies the formalization of the product behavior in a simulator able to process the user inputs in order to determine how the status of the product changes.

The behavior model of the product has been realized in Simulink using some of the several toolboxes available in this software package. The Simulink model developed for the test case can be considered as the first one that the engineers may realize in a top-down design process. It schematizes only the functions of the product which have a direct correlation to the product interface.

The logics of the product behavior, for which concerns the user interface, could be schematized using a Finite State Machine (FSM) model. Simulink supports the creation of FSM through the StateFlow toolbox that combines hierarchical state-machine dia-

Figure 4.7: *The hot mixer used as test case to validate the* interactive mixed prototyping *approach*

grams (as the ones introduced by Statecharts [Harel 1987]) with traditional flowchart diagrams.

4.3.1.1 Behavior model in Simulink

In Fig. 4.8 the complete Simulink model of the product is shown. The FSM, labeled as *Logic Unit*, is located in the left part of the picture. The FSM has two kinds of input: the first one, on its left, is related to the knob that controls the speed of the motor. It consists in two arrays that specify the speed of the motor for each angle of the knob rotation. The other input, on the upper part of the FSM block, is a bus that transmits all the data received from Virtools through the socket connection managed by the orange block.

The user actions are processed by Virtools that collects all the user generated events (button pressed, knob rotated, cover opened/closed) and sends them to Simulink. It receives all these data through the *Receive from Virtools* block. Also the clock signal is transmitted because it is used to synchronize the two applications. In Fig. 4.8 the

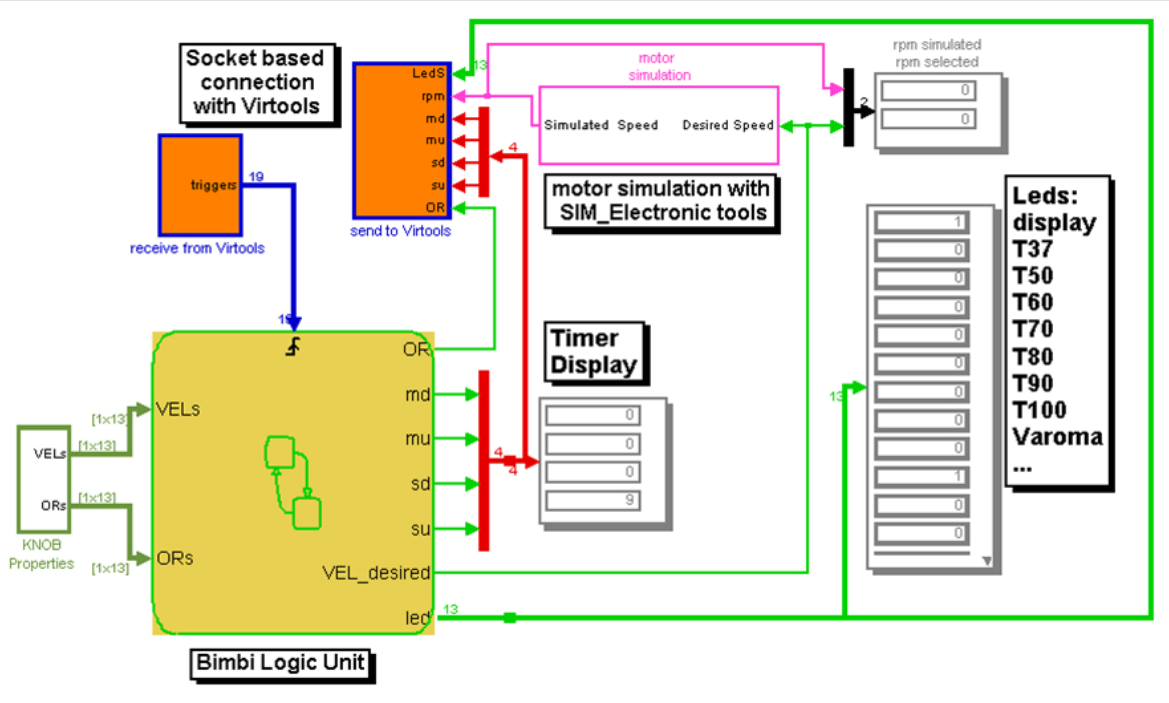


Figure 4.8: *Simulink Behavioral model of the product*

interaction between the user and the product has been explicated indicating where the user generated events are transmitted to the FMS.

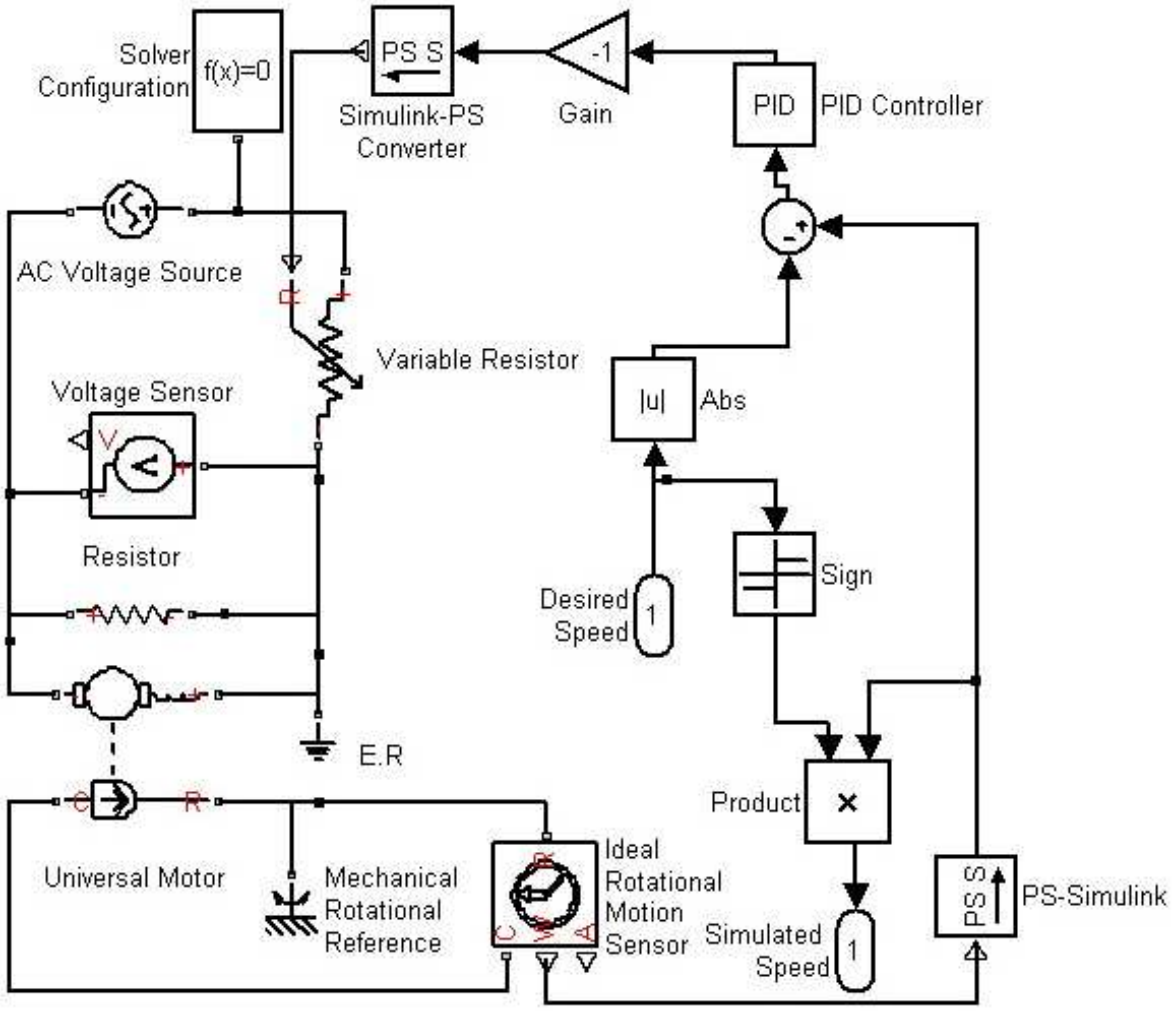


Figure 4.9: *Simulink Behavioral model of the product*

The *Logic Unit* continuously update the state of the simulated product, based on the received inputs and the current state. Then, the current configuration is iteratively sent back to Virtools (see the *send to Virtools* block in Fig. 4.8) in order to update the model and simulate the behaviour of the Interactive Virtual Prototype.

The FSM has seven outputs. The first value, OR, is the current angular orientation of the knob. The following four values, md, mu, sd, su, are the four digits on the LCD. $VEL\_desired$ is the desired motor speed set by the user with the knob, led is a vector that contains the state (on/off) of each temperature LED. The desired motor speed is

processed by a block that defines the physical behavior of the electric motor and the relative control circuit using the SimScape and the SimElectronics toolboxes.

The desired motor speed is processed by a block that defines the behaviour of the electric motor and the relative control circuit using the SimScape and the SimElectronics toolboxes. The model contained in the motor simulation block is shown in Fig. 4.9. The Universal Motor block has been used to simulate the voltage controlled AC motor. The control is realized through a variable resistor set by a closed loop feedback circuit with a PID (proportionalÚintegralÚ derivative) controller block. The three constants in the PID controller (the gains of the proportional, integral and derivative terms) have been manually tuned to correct the error between the measured process variable (i.e.: the motor speed) and a desired setpoint that is, in this case, the speed set by the user through the knob.

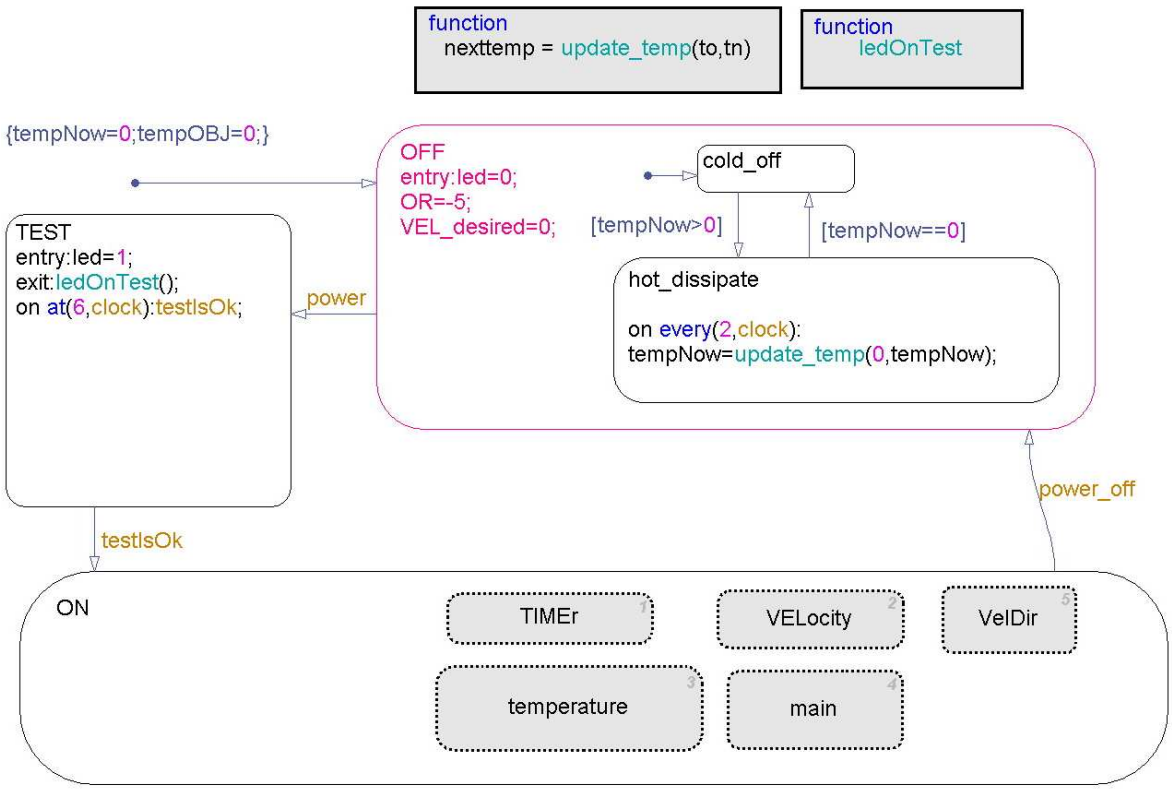


Figure 4.10: *The Finite State Machine that reproduces the behavior's logic of the product*

In Fig. 4.10 the main FSM model of the logic unit is shown. This FSM has three

states: OFF, ON, $TEST$. The default transition is characterized by the arrow present in the upper left corner of the image. It sets two environment variables to their initial states and activates the state OFF, that is the initial state of the product when the simulation starts. The **power** event, generated when the user presses the stand-by button, changes the state from OFF to $TEST$ and from ON to OFF. The $TEST$ state performs a test on the product. During the test, all the temperature LEDs are turned on and all the digits and the icons of the LCD appear, as shown in Figure Fig. 4.7. When the test is finished the current state switches to ON.

The OFF state is characterized by a subchart that contains two states: $cold\_off$ and $hot\_dissipate$: the former is activated when the machine is cold, the latter when the machine has been switched off but it has not completely dissipated the heat. This second state manages the status of the temperature LEDs and manages the cooling of the machine. When the machine is cold it activates the $cold\_off$ state.

The ON state is composed by four subcharts: *main*, *timer*, *velocity*, and *temperature*. The *main* subchart manages the completion of the job taking into account the time remaining before the job completion and any alteration of the motor speed or the timer.

The *timer* subchart catches and manages the user generated events related to the timer. In particular, when the user presses the + button associated to the clock icon, the timer is increased of a quantity that progressively changes: if the timer is between zero and one minute the increment is 1 second; if the timer is between one and ten minutes the increment is thirty seconds; after ten minutes the increment is one minute. The *timer* subchart controls also if the user reaches the maximum value for the timer (i.e.: sixty minutes) and calculates the value of each digit on the display.

The *velocity* subchart manages the events generated by the user when s/he rotates the knob to set the rotational speed of the tool inside the bowl. This chart reads the values contained in the two arrays, $VELs$ and ORs, that for each angle of the knob rotation specify the speed of the motor.

The temperature subchart, shown in Fig. 4.11, controls the part of the behaviour related to the hot working. In particular this chart defines how the temperature reaches

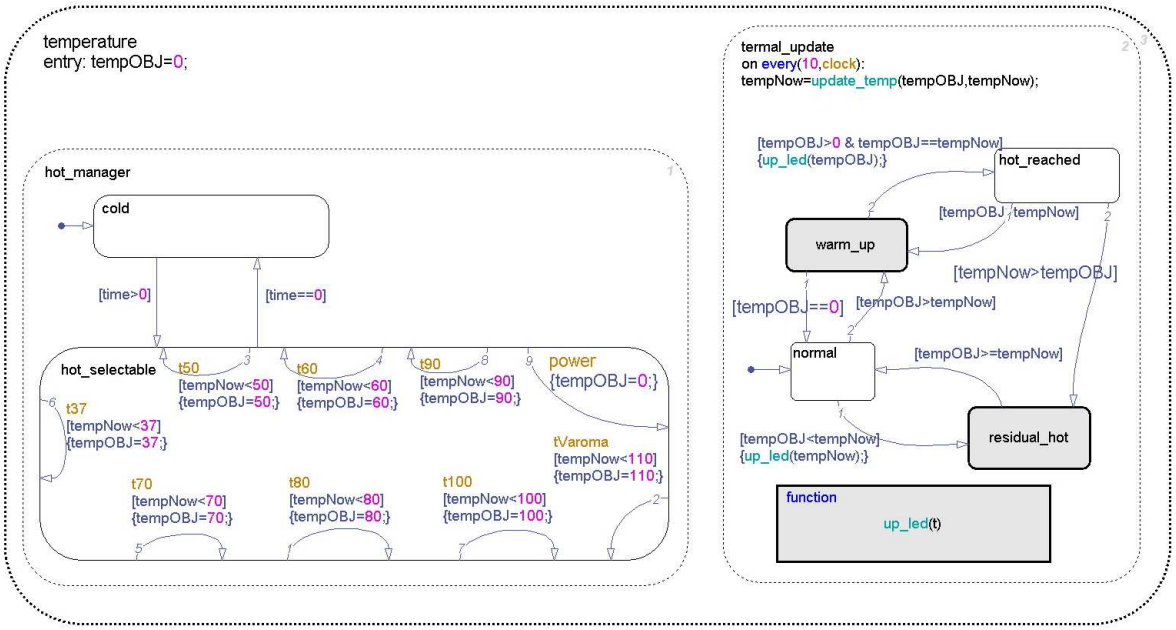


Figure 4.11: *The temperature subchart*

the value set by the user, controls if the user has set the timer (otherwise the temperature cannot be changed) and manages the state of the LED. When the user sets a temperature the corresponding LED starts to flash indicating the user choice. While the temperature is increasing, the LEDs are progressively turned on, indicating the temperature reached inside the bowl.

4.3.2 Coupling the behavioral simulation into VR

The design of the behavior model of the product, described in the previous section, usually represents one of the first steps in the design process of the control system for an electric appliance. With the proposed *Interactive Mixed Prototyping* approach, even this preliminary model could be used to evaluate the functions and the usability of the product interface. However, in order to test with the user the previously described behavior model, a digital mockup of the product is needed.

As previously mentioned, the geometries of the product have been reconstructed starting from an existing product. These geometries, completed with the materials and the textures, have been imported in Virtools. After that, the connection with the

Simulink model has to be created. This activity consists of some tasks:

1. Define the state variables that have to be connected to the Simulink model. The present testcase has six state variables that are the output of the FSM described in the previous section.

2. Create a *socket receive BB* able to process the data stream sent by the Simulink model.

3. Connect each output of the *socket receive BB* to its relative state variable.

4. Identify all the possible user generated events and catch them through a *Switch onMessage BB*. This BB sends a message to the Simulink model each time that the user generates an event.

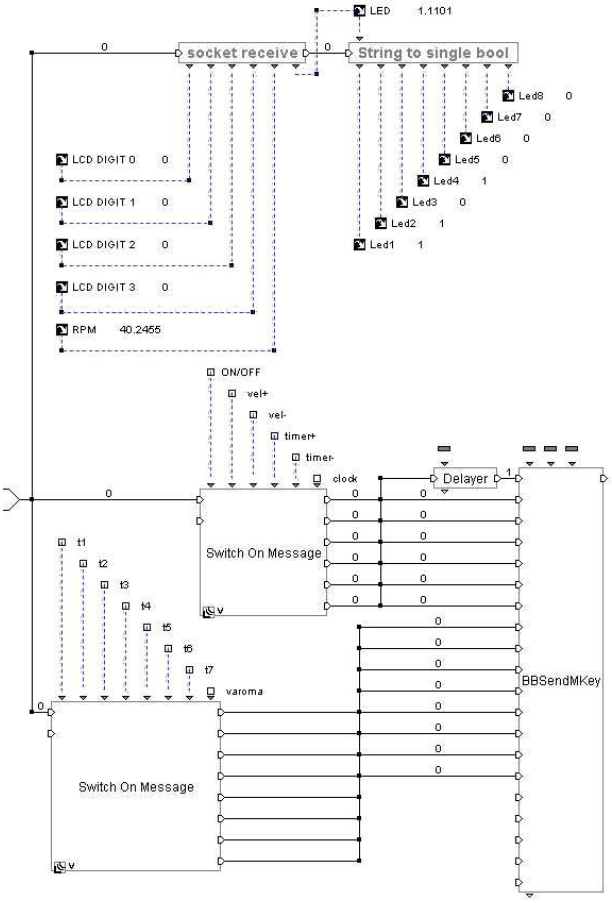


Figure 4.12: *The Virtools schematic that manages the connection with Simulink*

Figure 4.13: *The display of the virtual product is made up of nine rectangular patches with different textures that are changed at run time by the simulator*

Fig. 4.12 shows how the two BBs (*socketreceive* and *BBSendMKey*) are used to create the connection between the Virtools model and the Simulink model. The user generated events received by the *SwitchOnMessageBB* may be of two types: **button_pressed** and **knob_rotated**.

In the implementation of this test case the events related to the pressed buttons are managed with the micro-switch triggering approach already described in Sec. 4.2.5. In a more complicated set-up based on a data-glove [Bruno *et al.* 2007, Bruno *et al.* 2006] the button press may be identified as the collision between the geometry of the finger and the geometry of the button.

Regarding to the **knob_rotated** events, in [Bruno *et al.* 2006] a technique to recognize the user gesture for knob rotation is described. The technique is related to a VR set-up based on a data-glove, but it could be easily adapted to MR. However, a simpler solution was implemented by placing on the surface of the physical prototype a knob connected to an encoder that directly determines any rotation of the knob itself. This choice is motivated by the fact that it is a complex task for the user to rotate a virtual knob. The presence of the tangible knob strongly simplifies the interaction with the prototype. The display is simulated by defining, on the virtual model of the product, a series of rectangular patches on which different textures are mapped; such textures are changed at run-time according to the values of the different variables that

Figure 4.14: *First-person screen-shot of the Mixed Prototype while pressing a button*

define the state of the product (Fig. 4.13). The display is divided in nine parts: four are for the digits used to display time and temperature, one for the colon between the digits, four are used to show the status of some functions (timer, weighing, kneading, counter-clockwise rotation). The textures representing the ten digits are stored in a vector and assigned to the four slots according to the messages received by the Simulink connection.

4.3.3 Results

The presented study was carried out using a PC equipped with a Intel Core i7 CPU at 2,67 GHz, 3 GB of RAM and two GeForce GTX 260 video-cards with 896 MB RAM. The test (Fig. 4.14) has been done using the 800x600x30Hz acquisition modality of the camera, and we have measured an overall frame rate of about $40fps$. In detail, for each frame the application spends about $5ms$ to grab the image for the camera; $5ms$

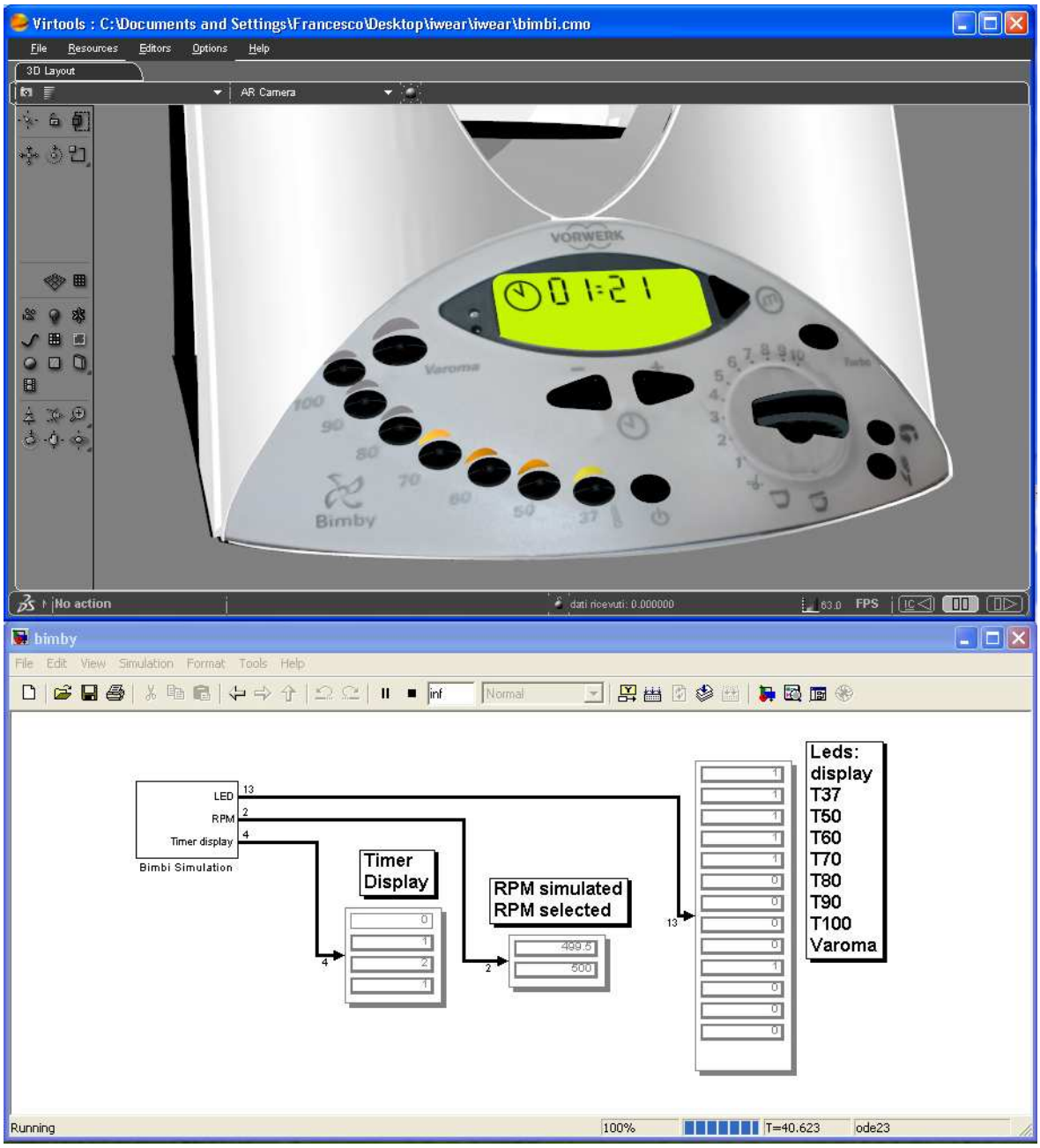


Figure 4.15: *Behavioral Simulation of the hot mixer running in Virtools (UP) and Simulink (Down)*

to complete the marker tracking task; less than $10ms$ to complete the hand recognition and the finger tracking; and other $5ms$ for rendering and executing the rest of the application. Figure 4.15 is a screenshot taken while the simulation is running. In the upper part of the image the Virtools window shows the interface with the timer

at 00 : 28; the temperature is currently at 50°C (yellow LED) and it has to reach 70°C (orange flashing LED); the speed is set to 5. In the lower part of the image the simulation running in Simulink is illustrated. As it is apparent, the state variables reflect the value of temperature, time and speed seen on the Virtools model.

4.4 Conclusions

The mixed prototyping approach presented in this chapter deals with three different issues usually present in the product behavior simulation in MR: the reliability and the development time of the behavior model, the occlusion between the user hands and the virtual objects, the interaction between the user and the product Human-Machine Interface.

Concerning the first point, the idea is that a CACE software, usually employed by control, electrical or electronics engineers, can be also used to simulate the product behavior in VR. This approach overcomes the classical problems related to the implementation of the code inside the VR software to replicate the product behavior.

For the other two points I have developed an interaction technique that, through the processing of the HMD video stream, is able to resolve the occlusion issue and recognize the user generated event. Moreover, the used set-up is implementable with a very low budget, and the use of a programmable shader for skin detection allows us to maintain a good frame rate, ranging from 40 to 60 fps.

The implementation of the test case puts in evidence that the proposed approach is immediately exploitable in the design of several products where the user interface is made by electromechanical elements like buttons, knobs, LEDs and displays.

The strength of the system is based on two different aspects: it does not require any specific device like data-gloves or tracking systems, and all the application work at interactive frame-rate.

A preliminary user study that should demonstrate the acceptability of the proposed set-up by the end-users has also been realized. A more rigorous usability test (see Chapter 5) was conducted to assess the validity of the tool and to compare it with

other systems like semi-immersive VR.

Chapter 5

Assessing Usability using Virtual and Mixed Prototypes

This chapter describes a user study addressed to establish whether and to what extent the augmented reality devices and the techniques proposed may distort the usability assessment of the product. The Mixed Prototyping approach proposed in chapter 4 was used during the user study. The results obtained were finally compared with those reported in a previous similar work, about the use of Virtual Reality in order to assess the usability of industrial products.

Contents

5.1 Introduction

Usability is considered one of the most important quality factors for several kinds of products and in particular for those that have an electronic interface. Mobile phones,

remote controllers, electrical appliances, car navigation systems and others may have a complex user interface that needs to be deeply analyzed by the designers. The ISO 9241 norm part 11 (ISO/DIS 9241-11) defines the elements through which one may evaluate, in qualitative terms, the usability of a generic product. These elements, that have to be detected through empirical tests, are: efficiency, effectiveness and satisfactory use of a product.

Usability evaluations can be conducted at different stages, both during and after the design and development process. The methods more frequently used are classified in analytic and empirical. Analytic methods allow to define the difficulties in using the product without involving users directly; such information is usually hypothesized by technicians after a long and accurate analytical study of the problems which may occur.

Empirical methods require the involvement of users during the usability evaluation, and are based on the observation of the ways in which users interact with a prototype or a product. Empirical methods allow one to carry out both quantitative rating and subjective rating. Quantitative rating refers to actions and attitudes, which may be estimated (time required to carry out a task, number of mistakes committed, etc.); whereas subjective rating refers to the experimentation subjectsŠ perceptions and opinions (scales which refer to the easiness of use, attitude scales, preferences for a product rather than another with the same functionalities, spontaneous comments, etc). The main empirical evaluation methods are usually based on the observation of a sample of users, interviews, questionnaires and interpretative evaluations.

Usually, a physical mock-up of the product concept or of the final product is needed to carry out a usability test with final users. The individuation of usability problems may take place in an advanced phase of the product development cycle. An assessment of usability during the conceptualization of the project would generate a great competitive advantage for companies, since it would allow a notable reduction of costs.

That's why in a previous research [Bruno & Muzzupappa 2010] it was proposed a methodology for the assessment of the usability of an industrial product, based on the realization of an usability test in Virtual Reality (VR). That approach allows designers

to involve final users of a product as from early design stages, without the need of a physical mock-up and with the advantage of being able to assess several design options.

The approach proposed in [Bruno & Muzzupappa 2010] has been tried out comparing two usability tests based on two different methods: the first one focusing on the interaction between final users and the real product, and the other one focusing on the interaction between users and a digital model in VR. The data obtained through this experimentation show that the types and the numbers of mistakes committed while carrying out the tasks, on the real product and in VR, are almost the same. This means that VR is a valid alternative to traditional methods for usability evaluation of industrial products, because the interaction through the VR devices does not invalidate the usability evaluation itself. The main limits of the methodology proposed in [Bruno & Muzzupappa 2010] are due to the lack of tactile feedback and to the inaccuracies of the tracking devices.

Mixed prototyping seems to be an interesting approach able to overcome these limits, because it combines the advantages of the virtual prototyping (quickly evaluate various design alternatives) and the physical prototyping (involve the sense of touch). A mixed prototype is usually based on a physical mock-up on which the visual appearance of the product is superimposed thanks to the augmented reality technologies.

In Chapter 4, I describe an original Mixed Realty set-up that solves two issues: managing the occlusion between the user's hands and the virtual object; tracking the movement of the user's fingers in order to make him/her able to interact with the mixed prototype.

This chapter describes a user study addressed to establish whether and to what extent the augmented reality devices and the techniques proposed may distort the usability assessment of the product. In other words, the Mixed Reality set-up has been tested with the users in order to determine if it can be employed in product usability tests. We have also compared it with the VR set-up reported in [Bruno & Muzzupappa 2010], based on electromagnetic tracking, data-glove and retro-projected stereoscopic screen.

5.2 A Methodology for Usability Evaluation in Mixed Reality

The mixed prototyping approach proposed in the previous Chapter is intended for all those products that have an electronic user interface (Fig. 5.1) as, for example, a car navigation system, a washing machine, a microwave oven, etc.

Figure 5.1: *Industrial Products with an electronic user interface*

In these cases the user controls the product acting on buttons, knobs and sliders and checks the status through LCDs, LEDs, etc... If we restrict the scope of the study to this kind of product, it is possible to define a methodology that synthesizes the tasks needed to conduct a usability test through a mixed prototype. The steps of the proposed methodology are the following:

1. Create the geometric model of the product.
2. Create a rapid prototype of the parts of the product that may be touched by the user during the tests.
3. Place on the rapid prototype optional tangible interaction elements like buttons, knobs or sliders.
4. Create a MR environment where the virtual model is correlated to the rapid prototypes.
5. Define the behavior of the product interface of the virtual model.
6. Conduct the usability tests with the users.

7. Analyze the collected data and decide which modification on the shape (and/or on the logics) may improve the usability of the product.

8. If any modification is needed on the interface layout, modify the geometry and iterate from step 2; if any modification is needed on the behavior, modify the logics of the interface and iterate from step 6.

The step 3 of the proposed methodology concerns the possibility to equip the rapid prototype with some interactive elements (buttons, knobs or sliders) as proposed in [Bordegoni *et al.* 2008, Caruso *et al.* 2007]. This idea offers several advantages because it makes easier the interaction of the user with the product interface, involves realistically the sense of touch and enhances the possibility of ergonomic evaluations. On the other hand, it requires an effort to create the interaction devices and to precisely place them on the physical prototype. This makes also more complicated to modify the layout of the product interface. Taking into account these considerations, one may decide how and when it is useful to enrich the rapid prototype with the interactive elements.

Step 5 basically relies on the idea to combine the MR environment with a simulation tool (as already described in the previous chapter). This idea is mainly justified by the fact that, in many cases, the behavior model of the product is already available since the first steps of the product development process. In fact, control engineers usually schematize and simulate a behavior model of the product using a Computer Aided Control Engineering (CACE) tool. One of the most spread techniques for the definition of the product interface behavior is the Finite State Machine (FSM).

5.3 Test With Users

A campaign of test with users was designed and accomplished in order to reach two different goals. The first one is to determine the influence of the MR environment in the assessment of the usability of virtual products. In other words the goal was to verify if the simulation of the product in MR may invalidate the result of the usability

evaluation. The second goal is to compare the usability test done in real and VR environment with the one done in MR. The comparison has regarded the benefits (in term of efficacy, efficiency and satisfaction of use) and the limitations related to the two set-ups giving particular attention on the presence of tangible elements in the MR environment.

The experiment illustrated in this section refers to the assessment of the usability of a product, currently present on the market (a microwave and electrical oven), thanks to usability tests based on three different approaches:

- the first on the interaction between end users and the real product,
- the second on the interaction between users and a model of the oven in VR,
- the third on the same virtual model in MR.

The results of the first two experiments have been already presented and discussed in [Bruno & Muzzupappa 2010]. In the next sections I will focus the attention on the new experiment that regards the interaction in MR.

5.3.1 Test case

The product chosen for the tests is a microwave and electrical oven already present on the market. The product presents a classic user interface with a knob that allows the user to set the timer, and 14 buttons to start/stop the oven and to set temperature, power, and other functions. All the information (temperature, timer, power, etc.) about the oven state are shown on a LCD.

Following the approach already described in Chapter 4, a digital model of the product has been created with a classical reverse engineering approach. The behavior model of the product has been schematized through a FSM using the Simulink StateFlow toolbox. The state transitions of the FSM are triggered by the events that the user generates while interacting with the digital product. The Virtools application recognizes these events and sends them to the Simulink model through an IPC channel.

In Fig. 5.2 the Simulink model of the product is shown. One S-Function receives the events generated by the user, which interacts with the virtual product and switches different simulation parameters according to these events. The other S-Function sends the data that describe the state of the product and each change that occurs during the simulation. The data sending is based on an asynchronous channel: the S-Function sends data to Virtools, without stopping the simulation. Virtools socket connection for the IPC has been implemented through two user defined Building Blocks (BB), able to send and receive data through the IPC channel.

The display is simulated by defining on the virtual model of the product a series of rectangular patches on which different textures are mapped; such textures are changed

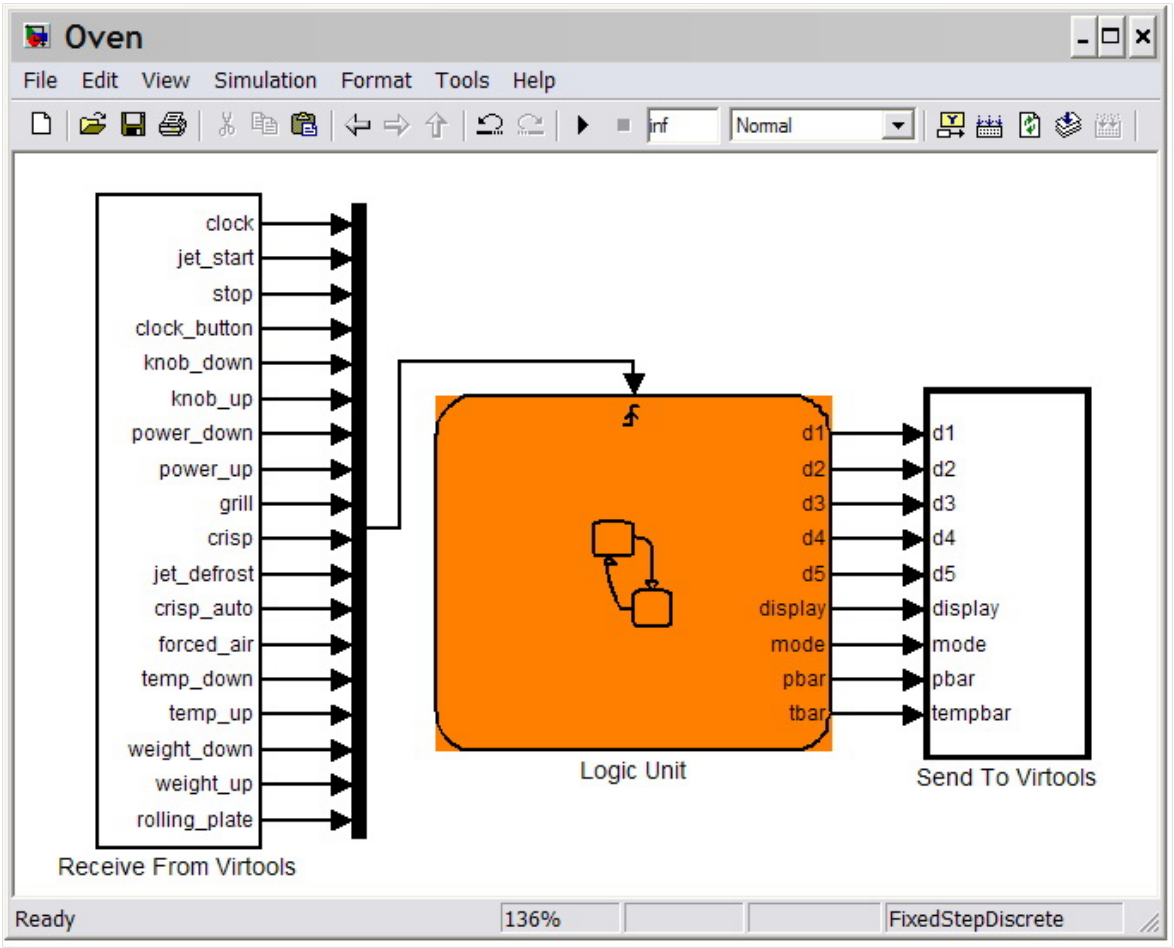


Figure 5.2: *The behavior model is defined in Simulink through a FSM connected to two s-functions that exchange data with the MR environment*

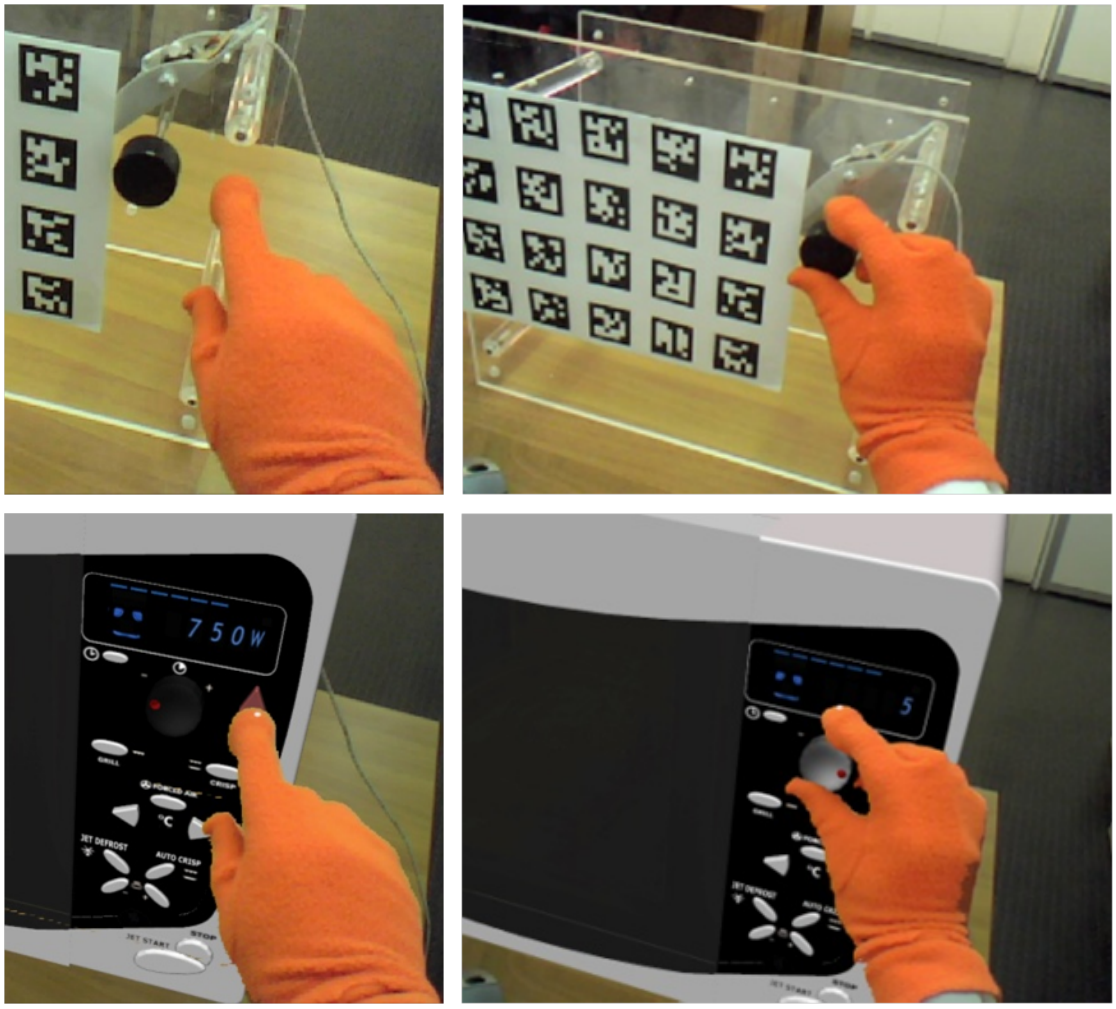


Figure 5.3: *Real (up) and Mixed (down) visualization while the user is pushing a button (left) or rotating the knob (right)*

at run-time according to the values of the different variables that define the state of the product.

The physical model of the oven is very simple and it has been approximated with a Plexiglas box. On the surface of the physical prototype we have placed a knob connected to an encoder that determines any rotation of the knob itself. This choice is motivated by the fact that it is a complex task for the user to rotate a virtual knob. The presence of the tangible knob strongly simplifies the interaction with the prototype.

After having put the multi-pattern marker, for tracking purposes, and having registered a virtual model to be properly aligned to the real one, the Mixed Prototype

model (Fig. 5.3) is ready to be used. More details on the preparation of the virtual prototype, as well as about the behavioral model, are available in Sec. 4.3, where the same approach is followed step by step for a different product.

5.3.2 Participants

In order to carry out the test (which took place in the Department of Mechanical Engineering, University of Calabria) 3 samples were chosen, each of 10 mechanical engineering students aged 23-26. All samples of users $A(A_1, \ldots, A_{10})$, $B(B_1, \ldots, B_{10})$ and $C(C_1, \ldots, C_{10})$ presented homogeneous features especially for which concerns the knowledge of the tested product (only 5 students per group owned and could use a microwave oven).

SampleA was asked to carry out the usability test with the real interface, *sampleB* was asked to assess the usability of the virtual interface whereas *sampleC* was asked to carry out the usability test in MR.

5.3.3 Experimental design and procedure

The usability testing was carried out by observing each group of users during their interaction with: a real object, a virtual model in VR, a virtual model in MR respectively (Fig. 5.4).

In concern to the object to be tested, we would like to point out that, since the testing had to focus on a comparison between data obtained through the user's interaction with a real object and data obtained through the user's interaction with the object in virtual and augmented environment, a common electrical appliance -a microwave oven- was chosen. The appliance included several functions, also in its VR reproduction. This allowed users to carry out several tasks of varied difficulty. The experiment was carried out mainly in three phases:

- an analytical phase during which we organized the entire carrying out of the test. We defined the procedures to be used during the test, the assessment tools (user-profile questionnaire, user-object interaction phase, satisfaction questionnaire,

virtual interface assessment questionnaire), the two kinds of users' tasks and choice criteria;

- an operational phase during which the sample of users, after having been told about the aim of the research, carried out the test with either the real product or the virtual product;
- an assessment phase during which we analyzed all the information we collected, thanks to questionnaires and the observation of users during the test.

During the operational phase each single user carried out three activities:

1. filling in the user-profile questionnaire;
2. the actual testing of the (real or virtual or augmented) object, carrying out assigned tasks;
3. filling in a questionnaire on the degree of satisfaction.

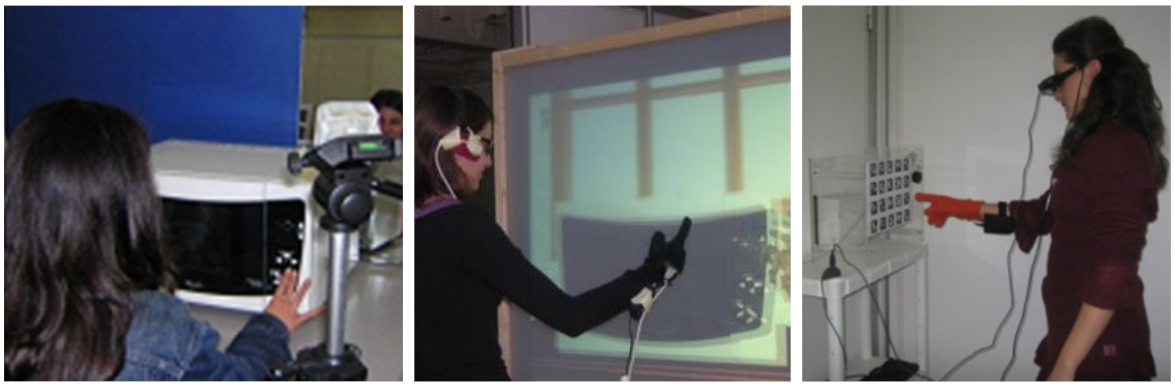


Figure 5.4: *Experiments with the real, virtual and mixed products*

Furthermore, the two samples B and C were trained, in order to be able to interact with the virtual systems. At the end of the test, the users also filled in a questionnaire concerning their approach with the virtual and augmented interfaces (difficulties in using interactive environment, perceptive level of the simulated reality, etc...).

The questionnaire identifying the user's profile, made up of 12 questions and aiming to deepen our knowledge of the single user, allowed us to obtain information on his/her

level of technological knowledge (knowledge and familiarity of computer tools) and on his/her experience in using the product used during the test (knowledge and use of electrical and microwave ovens).

During the actual interaction with the product test, the user is asked to carry out four tasks while s/he was facing the object (real - virtual - mixed). The different tasks were presented as realistic scenarios (in order to make the user feel involved) with an increasing difficulty. We instructed the user about one task at a time in order to not disturb his/her concentration and avoid misunderstandings.

During the test, we carefully observed the users and timed them, in order to take note of the amount of time needed to fulfill each given task. We also wrote down any aspect, which could be useful for the usability evaluation (mistakes, comments, calls for help, expressions, etc.). Furthermore, the use of a video camera allowed us to re-examine each single test carefully.

At the end of the interaction with the product, we asked the users to fill in a satisfaction questionnaire, made up of 11 questions. The collected data allowed to evaluate the users' overall degree of satisfaction and their well-being or sense of unease perceived after having used the product, as well as detect problems related to specific aspects of the product itself (content, structure, graphic interface).

5.3.4 Results

As it was already pointed out, the main purpose of the experiments was the assessment of the reliability of a usability test in a MR environment, and the comparison with the same test done in real and VR environment. To this end, the first step to make is to establish whether and to what extent MR may distort the usability assessment of the product. Therefore, the analysis of the goals achieved through the test is focused on the most significant data regarding the augmented interface assessment, deliberately neglecting the usability assessment of the specific product. The analysis of variances (ANOVA) was used to analyze the number of errors, the degree of satisfaction and the task completion times.

Nevertheless, some considerations have to be done in order to better understand

Table 5.1: *One-way ANOVA tables for* number of errors

		N	Mean (s)	Std. Dev.	Std. Err.	Lower Bound	Upper Bound	Min	Max
TASK 1	Sample A	3	1,33	0,577	0,333	-0,10	2,77	1	2
	Sample B	3	1,00	0,000	0,000	1,00	1,00	1	1
	Sample C	5	1,00	0,000	0,000	1,00	1,00	1	1
	Total	11	1,09	0,302	0,091	0,89	1,29	1	2
TASK 2	Sample A	10	1,40	0,516	0,163	1,03	1,77	1	2
	Sample B	10	1,30	0,483	0,153	0,95	1,65	1	2
	Sample C	9	1,11	0,333	0,111	0,85	1,37	1	2
	Total	29	1,28	0,455	0,084	1,10	1,45	1	2
TASK 3	Sample A	9	1,00	0,000	0,000	1,00	1,00	1	1
	Sample B	4	1,25	0,500	0,250	0,45	2,05	1	2
	Sample C	7	1,14	0,378	0,143	0,79	1,49	1	2
	Total	20	1,10	0,308	0,069	0,96	1,24	1	2
TASK 4	Sample A	10	1,70	0,483	0,153	1,35	2,05	1	2
	Sample B	10	1,50	0,707	0,224	0,99	2,01	1	3
	Sample C	8	1,38	0,518	0,183	0,94	1,81	1	2
	Total	28	1,54	0,576	0,109	1,31	1,76	1	3

		Sum of squares	df	Mean square	F	Sig.
TASK 1	Between groups	0,242	2	0,121	1,455	0,29
	Within groups	0,667	8	0,083		
	Total	0,909	10			
TASK 2	Between groups	0,404	2	0,202	0,975	0,39
	Within groups	5,389	26	0,207		
	Total	5,793	28			
TASK 3	Between groups	0,193	2	0,096	1,020	0,38
	Within groups	1,607	17	0,095		
	Total	1,800	19			
TASK 4	Between groups	0,489	2	0,245	0,722	0,50
	Within groups	8,475	25	0,339		
	Total	8,964	27			

the results. Usability research is considered behavior-driven: you observe what people do, not what they say. In contrast, market research is largely opinion-driven: you ask people what they think. Behavior-driven research is more predictable. Different studies

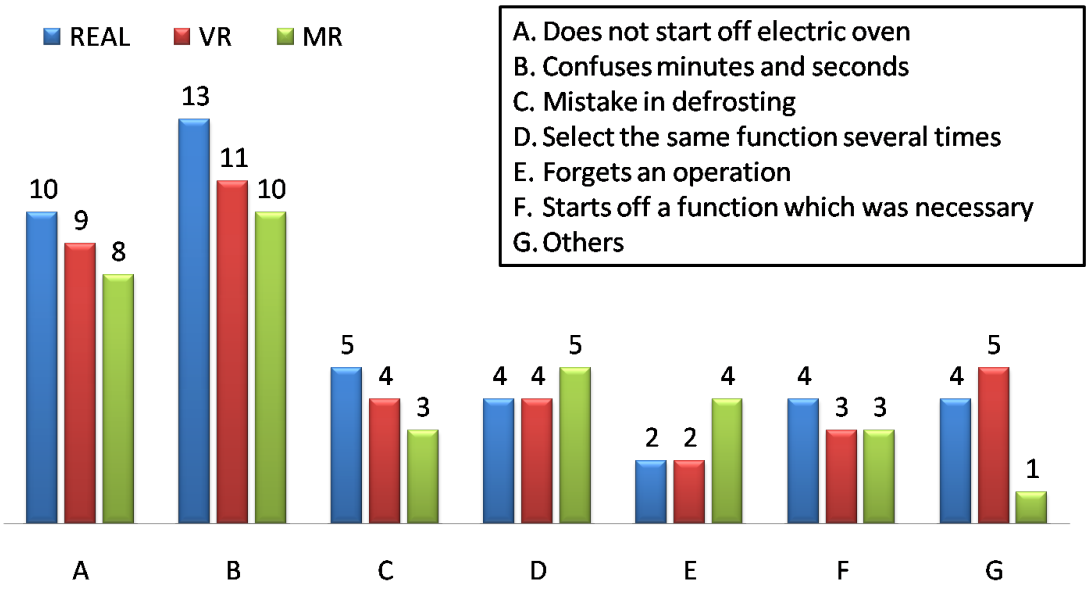


Figure 5.5: *Error typologies and their frequency during the test*

supporting the assumption argued that just 5 participants could reveal about 80% of all usability problems that exist in a product [Virzi 1992, Landauer & Nielsen 1993]. As a consequence, we have used samples with 10 users in our tests and an alpha level of 0.1 in the analysis of variances.

Table 5.1 compares the number of errors made by the three samples of users (A, B and C) while carrying out the tests. The difference between the mean values for the four tasks is not statistically significant ($sig. > 0.1$). On the other hand, after having carefully compared the most significant typologies of mistakes made by the three samples of users during the tests, we clearly noted that the number and the typologies of mistakes are the same for real, VR and MR experiments.

Starting from these results, we can state that the number of errors is not dependent by the type of experiment carried out by each sample. In other words, we can accept the null hypothesis since there are no significant differences among the samples (Fig. 5.5), or else that:

> **the VR and the MR interface do not increase difficulties in understanding while carrying out the tasks, and they do not distort the effectiveness of the system.**

Table 5.2: *One-way ANOVA tables for* degree of Satisfaction*: samples A and B*

		Mean	Std. Dev.	F	Sig
1. Easiness of task	Sample A	4,00	0,667	2,65	0,121
	Sample B	4,50	0,707		
	Total	4,25	0,716		
2. How agreeable the product was	Sample A	3,80	0,789	1,80	0,196
	Sample B	3,40	0,516		
	Total	3,60	0,681		
3. Frustration while using the product	Sample A	1,40	0,699	0,00	1,0
	Sample B	1,40	0,516		
	Total	1,40	0,598		
4. Clarity and simplicity of use	Sample A	4,50	0,527	3,46	0,079
	Sample B	4,00	0,667		
	Total	4,25	0,639		

The observation of users during the execution of tasks and the analysis of satisfaction questionnaires were very useful in determining the *degree of satisfaction* and in the evaluation of the unease felt while facing the product interface. Also in this case, the ANOVA analysis we carried out showed how the degree of satisfaction of the three samples of users was not statistically significant. The latter (ranging from $0 = low$ to $5 = high$) was not affected by the experiments. In particular, we have compared the results of four questions for sample A and B in Tab. 5.2 and for sample A and C in Tab. 5.3.

It is very interesting to note that in Tab. 5.2 only for question 4 (Clarity and simplicity of use) the difference between the two mean values (4,5 vs 4,0) is statistically significant due to the lack of haptic devices (as evidenced by video registrations and users feedback) that makes it more difficult to use the VR interface. The same question in Tab. 5.3 puts in evidence that this difficult can be considered overcome by MR setup (4,5 vs 4,4 and $sig > 0,1$).

Table 5.3: *One-way ANOVA tables for* degree of Satisfaction*: samples A and C*

		Mean	Std. Dev.	F	Sig
1. Easiness of task	Sample A	4,00	0,667	2,250	0,15
	Sample C	4,40	0,516		
	Total	4,20	0,616		
2. How agreeable the product was	Sample A	3,80	0,789	0,375	0,55
	Sample C	4,00	0,667		
	Total	3,90	0,718		
3. Frustration while using the product	Sample A	1,40	0,699	0,60	0,45
	Sample C	1,20	0,422		
	Total	1,30	0,571		
4. Clarity and simplicity of use	Sample A	4,50	0,527	0,184	0,68
	Sample C	4,40	0,516		
	Total	4,45	0,510		

As far as completion time is concerned, the experiment points out longer execution times for sample B (on average twice as much) than the average times registered by sample A; on the other hand, completion time for sample C is much closer to the results with the real interface (Tab. 5.4).

The main cause of delays in completing tasks in VR and MR environment with respect to the real product can be identified in the difficulty of using the knob to set the time on the oven. In particular, for sample B, the main limits of the VR system depend on the perception of the virtual product (typical limits in an immersive VE, since they are due to a lack of tactile feedback and to inaccuracies of the input devices). This problem is partly overcome by MR setup: in fact, when the user interacts only with buttons (Fig. 5.3-*left*), the average time are the same between sample A and C (see task 1 and 3). When the user uses also the knob (Fig. 5.3-*right*), the average time is growing albeit significantly less than in the virtual environment.

Table 5.4: *One-way ANOVA tables for* Task Completion Time

		N	Mean (s)	Std. Dev.	Std. Err.	Lower Bound	Upper Bound	Min	Max
TASK 1	Sample A	10	39,40	9,652	3,052	32,50	46,30	30	61
	Sample B	10	57,20	4,894	1,548	53,70	60,70	53	70
	Sample C	10	41,20	11,331	3,583	33,09	49,31	26	62
	Total	30	45,93	11,934	2,179	41,48	50,39	26	70
TASK 2	Sample A	10	51,60	3,627	1,147	49,01	54,19	46	56
	Sample B	10	72,60	13,858	4,382	62,69	82,51	56	95
	Sample C	10	58,40	9,582	3,030	51,55	65,25	45	80
	Total	30	60,87	13,090	2,390	55,98	65,75	45	95
TASK 3	Sample A	10	37,30	3,335	1,055	34,91	39,69	32	42
	Sample B	10	67,00	19,050	6,024	53,37	80,63	56	105
	Sample C	10	35,20	7,941	2,511	29,52	40,88	20	52
	Total	30	46,50	18,809	3,434	39,48	53,52	20	105
TASK 4	Sample A	10	52,10	2,283	0,722	50,47	53,73	49	56
	Sample B	10	92,40	17,709	5,600	79,73	105,07	58	104
	Sample C	10	74,60	15,848	5,012	63,26	85,94	56	98
	Total	30	73,03	21,405	3,908	65,04	81,03	49	104

		Sum of squares	df	Mean square	F	Sig.
TASK 1	Between groups	1920,267	2	960,133	11,732	0,0
	Within groups	2209,600	27	81,837		
	Total	4129,867	29			
TASK 2	Between groups	2296,267	2	1148,133	11,596	0,0
	Within groups	2673,200	27	99,007		
	Total	4969,467	29			
TASK 3	Between groups	6325,800	2	3162,900	21,709	0,0
	Within groups	3933,700	27	145,693		
	Total	10259,500	29			
TASK 4	Between groups	8157,267	2	4078,633	21,468	0,0
	Within groups	5129,700	27	189,989		
	Total	13286,967	29			

5.4 Conclusion

The proposed Mixed Prototyping methodology (Sec. 5.2) is immediately usable in the design of several products where the user interface is made of electronics elements like

buttons, sliders and knobs. The interaction techniques and, more generally, the Interactive Mixed Prototyping environment (Sec. 4) that we have adopted to recognize the user generated event, is realizable with a very low budget and a limited implementation effort.

But the most important contributions of the experimental work are the results obtained with the user tests that confirm that

> **both VR and MR, are valid alternatives to traditional methods for product interface usability evaluation**

and also that

> **the interaction with the virtual/mixed interface does not invalidate the usability evaluation itself**.

The proof of the truthfulness of the statements above is that, regardless of the type of interaction taken into consideration (whether real, virtual or augmented), the same difficulties of functional understanding of the product were noticed in all users. Finally we can state that the proposed MR set-up offers better performances with respect to VR, also when the user is interacting with knobs.

Chapter 6

Tool-based Mixed Prototyping without Visual Obtrusion

The first part of this chapter describes the challenges related to a Visuo-Haptic Mixed Reality (VHMR), where a user can see and touch *virtual objects in combination with real objects in the scene. In particular, I propose a novel mixed reality paradigm where it is possible to touch and see virtual objects in combination with a real scene, but without visual obtrusion produced by the haptic device. This mixed reality paradigm relies on the following four technical steps: tracking of the haptic device, visual deletion of the device from the real scene, background completion using image-based models, and consistent compositing of the user's hand based on a skin-color detection strategy. The final goal of the work presented in this chapter is to exploit the proposed VHMR paradigm for a new Mixed Prototyping approach, mainly intended for the concept design phase, where shape and topology of the product interface are continuously changed, thus making unprofitable the Mixed Prototype approach described in chapter 4.*

Contents

6.1 Introduction

In the typical desktop virtual-reality setup, the user looks at a screen, and visual and haptic stimuli are presented in a de-located manner. However, a MR setup allows the user to perceive visual and haptic stimuli in a collocated manner, i.e., the user can see and touch virtual objects at the same spatial location. Collocation improves the sensory integration of multimodal cues and makes the interaction more natural, but it also comes with technological challenges. In this chapter, I identify and address three of these challenges, with the common goal of improving the naturalness of the perceptual experience and the sense of presence.

First, the inclusion of haptic interaction in a MR scene requires the use of a haptic actuator, but most haptic actuators are bulky devices that occupy a large space in the visual region of interest, i.e., in the location where the interaction is actually taking place. Therefore, in a collocated VHMR setup, the haptic device becomes an obtrusive visual element. The importance of unobtrusive haptic interaction has been addressed in the past, and the proposed answers relied on mechanical solutions that placed the haptic actuators far from the region of interest using string-based haptic devices [Tarrin *et al.* 2003], or optical solutions based on retro-reflective paint and a head-mounted projector [Inami *et al.* 2000]. The proposed solutions aims to increase,

through a computational solution, the versatility of visuo-haptic interaction setups. It is based on visual removal of the haptic device from the context scene, together with the Image-Based Rendering(IBR) strategy for background completion, desscribed at Sec. 3.2.

Second, naïve visual compositing of real and virtual objects may produce scenes that do not fulfill depth consistency. In a MR system, correct handling of occlusion between virtual and real objects enhances the user's perception, by increasing the feeling that virtual objects truly exist in the real world[Sekuler & Palmer 1992]. In particular we focus on occlusion handling of the user's hands and virtual objects, even when grasping virtual tools. To answer this occlusion problem, I adopt the segmentation strategy described in Sec. 3.3. Used with the glove option, it provides a robust segmentation of the user's hand, and allows us to correctly compose the hand with the virtual objects.

Third, commodity haptic hardware suffers from mechanical limitations that may restrain the ability to render the virtual objects up to their full dynamic range. In particular, when a virtual tool is constrained by a wall, a commodity haptic device cannot prevent the user from pushing inside the wall, producing an undesired misalignment of the virtual tool and the user's hand. I propose a visual compositing technique that compensates for the virtual tool misalignment, redrawing the hand with an artificial displacement.

Given these challenges, my main contributions are combined in a novel VHMR paradigm, integrating the computer vision solutions described in Chapter 3. It provides augmented touch, collocated visual and haptic interaction, and occlusion handling between user's hand and virtual objects, but without visual obtrusion produced by the haptic device. It also visually enforces physical constraints between the user's hand and virtual parts.

In [Cosco *et al.* 2009], a first approximation to VHMR was already published, with a solution to the visual obtrusion produced by the haptic device and based on image-based rendering (IBR) of pre-captured background images. with a strong focus on view-dependent texture mapping [Debevec *et al.* 1996].

The solution described in this chapter extends the VHMR algorithm by adding

correct compositing of the user's hand, the virtual tool and objects, and the real background. I handle occlusion problems using a skin-detection approach, and I also address misalignment problems suffered with commodity haptic devices.

The proposed paradigm relies on two distinct phases: *system calibration*, that is a pre-processing phase, embedding all the tasks required to set-up the environment, and the run-time phase, that iteratively executes the designed *rendering pipeline*. The following sections reflect this distinction, describing a successful proof-of-concept implementation. I present also both qualitative and performance results of this proof-of-concept implementation. Finally I present also the ultimate goal of the presented approach describing the expected results in the field of Virtual Prototyping, but also a preliminary user study confirming the acceptability of the developed interface.

6.2 Overview of the Visuo-Haptic Display

As discussed in the introduction, the aim is to design a visuo-haptic display of mixed reality that allows to manipulate in a seamless manner both real and virtual objects. The focus is about manipulation metaphors where the user holds a virtual tool and interacts with the mixed reality content through this tool. For the interaction to be natural, the user should be able to see his/her own hand holding the virtual tool, and all objects, both virtual and real, should satisfy physical laws (e.g., gravity, non-penetration, action-reaction, etc.), both visually and haptically.

To solve this task, I employ state-of-the-art simulation techniques to compute the physical interaction between virtual objects and to haptically render the manipulation. In particular, it was adopted a constrained dynamics algorithm for solving deformation and contact of rigid and deformable objects [Otaduy *et al.* 2009], and a multi-rate haptic rendering algorithm for manipulating rigid handles [Garre & Otaduy 2009]. To enable contact between real and virtual objects, for each real object a virtual counterpart is modeled into the simulation algorithm . Currently, the interaction with real objects is limited to static ones.

For the visual rendering, I follow a vision-based tracking approach of the user's

head, together with video see-through display on an HMD. This solution allows view-dependent co-located display of visual and haptic feedback, and allows the user to see his/her own hand. These two features make manipulation and interaction more natural than looking at a monitor or not seeing the hand.

6.3 System Calibration

The first step of the calibration is to get a MR environment working properly: this requires at least to calibrate the camera and prepare the tracker. The intrinsic camera parameters were calibrated using Matlab's Calibration Toolbox [Bouguet 2008]. Then the procedure may vary according to the chosen tracking strategy. The used MR configuration relies on the use of ARToolkit's marker-based tracking system[Kato & Billinghurst 1999]. Therefore, a set of quadrangular markers have to be placed into the scene, taking care that at least one marker is visible during user's operations. This first step is fundamental because the following tasks are designed to exploit some MR feature (e.g. the real-time tracking).

In facts, once MR is enabled, we can proceed with the scene capture task. Its goal is to store a set made of a sufficient number of images to reproduce, by IBR, an image of the scene, from an arbitrary point of view. Note that the set is stored with no haptic hardware present in the scene, so that only the background appears in the image obtained with IBR. Each image of the subset is spatially referenced using tracking data, therefore its position is implicitly defined with respect to the global reference frame used by the tracker.

At the end we can put the haptic device in the scene, and then calibrate it for referencing its local system with respect to the global one.

6.3.1 Scene capture

Scene capture consist of building-up the set of data needed for the IBR algorithm to reproduce the image of the background during the last part of the run-time pipeline. Each record of the set represents an image of the scene, coupled with the camera

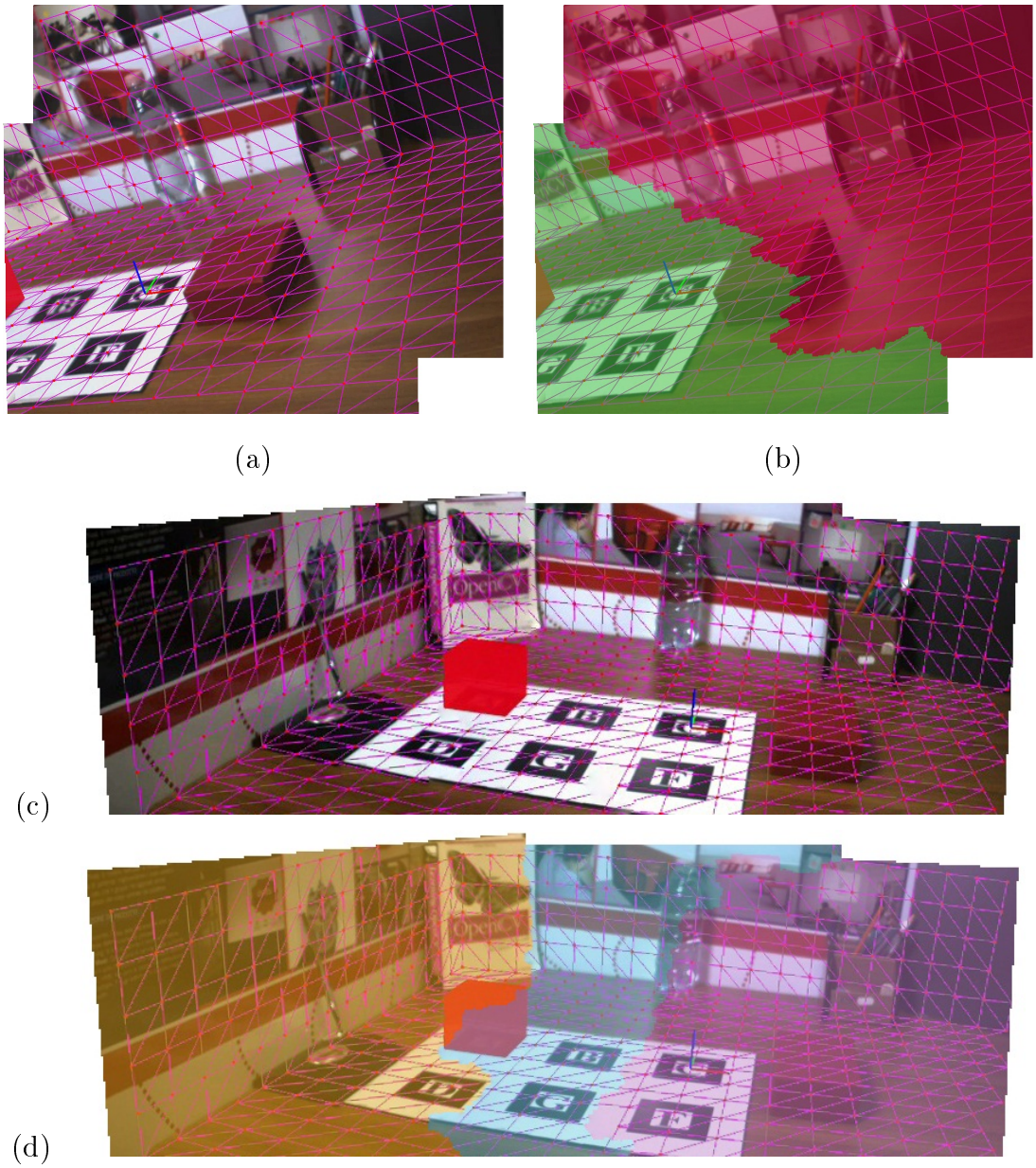


Figure 6.1: *An interactive scene capture application: the user qualitatively evaluates if the capture set is well-sampled, looking a preview of the current view-dependent texture mapped model. (a) and (c) report two of these previews, obtained for two distinct positions. (b) and (d) are the corresponding blending fields, where each color means a different input image, extracted from the set.*

position from which it was acquired.

To achieve this, we designed an interactive MR application that is helpful to iteratively build-up the dataset. At each iteration, the algorithm stores the image copying it

from the background AR-buffer, and retrieves its spatial-reference from current tracker state. This application is executed while the camera is moved into the scene, covering in particular the regions where the user is expected to move during the interactions.

One annoying issue related with the scene capture is to get a well-sampled set of images, i.e. collecting the sufficient amount of images needed to reproduce an IBR of the background from any needed viewpoint.

The IBR method can be thought as an interpolation technique of the *plenoptic* function, over the given dataset.

> "To measure the plenoptic function one can imagine placing an idealized eye at every possible (V_x, V_y, V_z) location and recording the intensity of the light rays passing through the center of the pupil at every possible angle (θ, ϕ), for every wavelength, λ, at every time t. [...] The resulting function takes the form: $P = P(\theta, \phi, \lambda, t, V_x, V_y, V_z)$"
>
> [Adelson & Bergen 1991]

Then, a dataset can be thought as a scattered set of points in the multi-dimensional space of the plenoptic function. A well-sampled dataset will assure a good density over the entire required region.

To overcome this issue, I implemented an interactive version of the scene capture application that presents a preview of the view-dependent texture mapped model of the background, obtained with the method described in Sec. 3.2.

This view-dependent scene model is updated and showed (as depicted in Fig. 6.1), while the scene capture is executed, allowing the user evaluating the current IBR model in order to fill eventual holes in the background map. When the IBR model is qualitatively sufficient, the subset is saved and stored for further use.

6.3.2 Haptic calibration

The integration of haptic devices in an AR context requires a calibration able to exactly align haptic and world coordinate systems as described by Harders et al. [Harders *et al.* 2009].

Keeping in mind that any error in this process would greatly diminish the usability of the system, and compromise user interaction with the VP [Summers *et al.* 1999], we have designed a simple calibration procedure inspired to the work of [Vallino & Brown 2002]. The procedure concerns the collocation of a Phantom Omni haptic device by Sensable into the marker-based MR environment.

The haptic calibration procedure was implemented as a MR application, in order to make it easier to use. It starts by asking the user to put the Phantom pen tip upon the corner highlighted in AR with a red flashing ring (Fig. 6.2). The procedure automatically asks for every available corner, computed upon the defined markers arrangement. After reaching the requested position, the user can press a button on the pen to store the current position, and eventually s/he may store different configurations of the same point to improve the quality of the result. When finished, or in the case of an unreachable position, the user moves to the next iteration by pressing another button.

When a point is selected, the system automatically stores both two vectors P_w and P_h, defining respectively the position in the world and the haptic reference frame. P_w and P_h are expressed in homogeneous coordinates. In particular, P_w is already known, because it is defined for the multi-marker AR set-up. The P_h vector can be retrieved by evaluating the direct kinematic function over the current angular encoders' configuration.

After the interactive calibration phase, an optimization algorithm computes the affine transformation T by minimizing the following objective function:

$$f_{obj}(x) = \sum_{i}^{n} P_h^i - T(x)P_w^i$$

The resulting transformation matrix T is then used to align the haptic reference in respect with the world reference system.

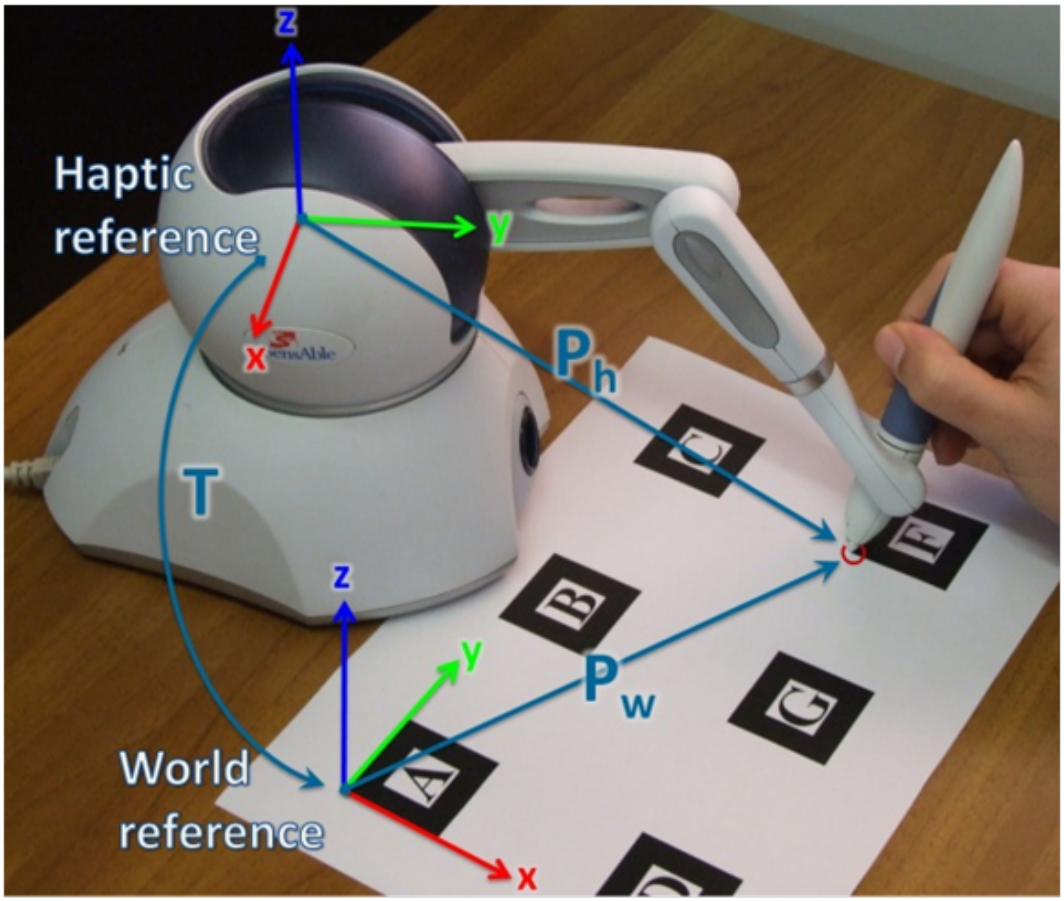


Figure 6.2: *Interactive AR calibration of an haptic device*

6.4 Overview of the Pipeline

The core of the contribution addresses problems induced by the combination of visual and haptic display in the mixed reality context, as VHMR introduces problems that are not present when visual or haptic display are used alone. As outlined in the introduction, the specific problems we address are (i) visual obtrusion of the mixed environment produced by the haptic device, (ii) occlusion handling of the user's hand and the virtual tool, and (iii) tool misalignment produced by mechanical limitations of commodity haptic devices.

The approach relies on using see-through head-mounted display technology, and the mixed reality paradigm is enriched with some *ad hoc* features.

As shown in Fig. 6.3, the visual display algorithm is based on three different phases: the identification of various semantically distinct regions in screen-space, the Image-

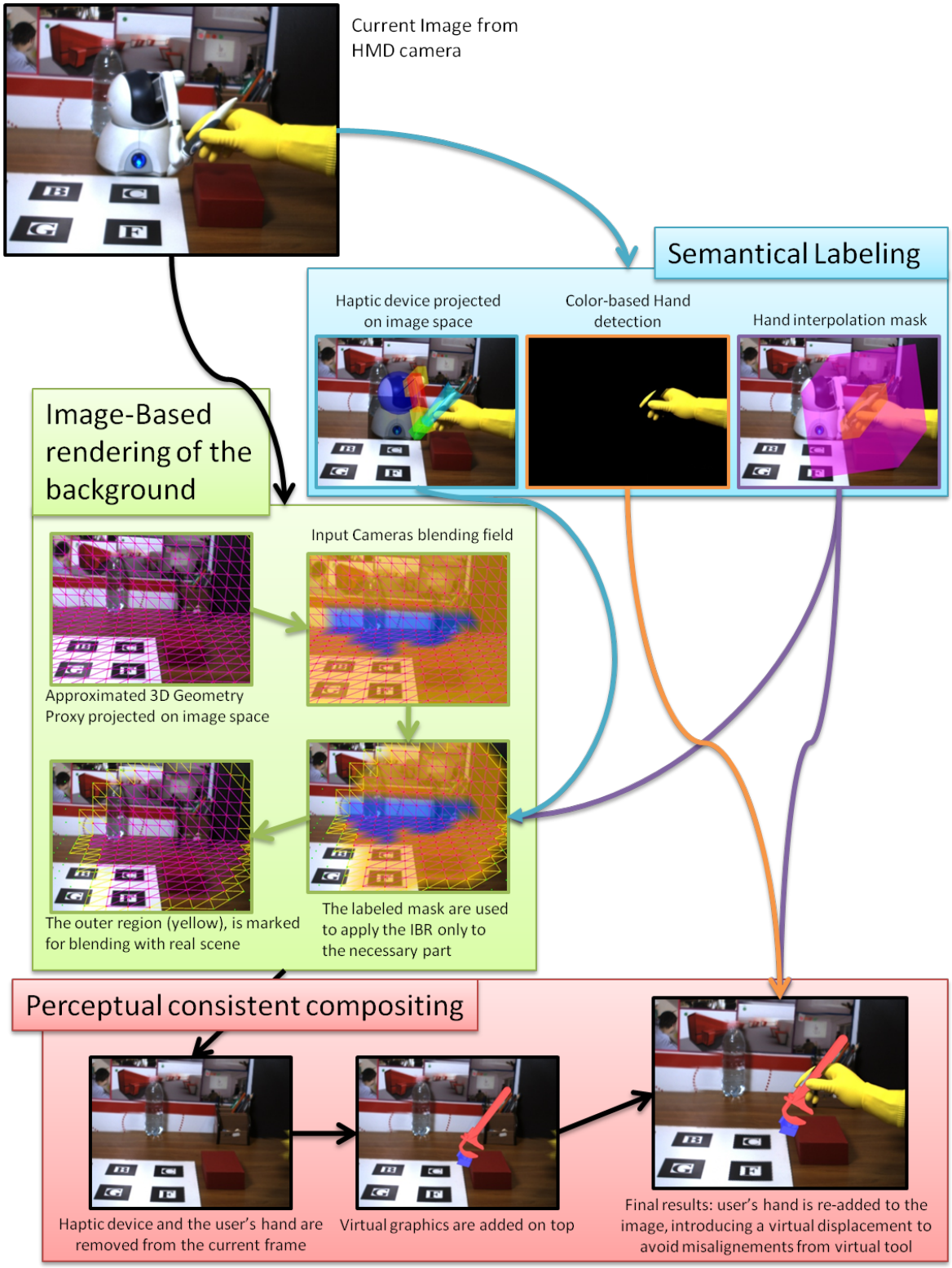


Figure 6.3: *A conceptual scheme of the proposed pipeline*

based rendering of the background (using the method described in Sec. 3.2, followed by the *perceptually consistent* visual compositing, that simultaneously solve all obtru-

sion, misalignments and occlusions issues. The whole run-time display pipeline can be outlined as follows:

1. Common Augmented Reality pre-compositing tasks
   (a) An image of the scene is captured with a camera
   (b) The camera pose, is computed following a marker-based approach
2. Semantic Labeling in screen-space
   (a) To address the problem of visual obtrusion, one has to identify and segment the region of the image occupied by the haptic device. This is achieved by monitoring the configuration of the haptic device, and protecting a virtual replica of the device onto screen-space.
   (b) To address the correct visualization of the user's hand, it is crucial to identify and segment the screen-space projection of the user's hand. This task is solved using the color detection approach described in Sec. 3.3.
3. Perceptually Consistent Compositing
   (a) The captured scene image is processed to remove both the haptic device and the user's hand.
   (b) Virtual objects are composed in the scene.
   (c) The user's hand is repainted in the foreground, accounting for a virtual displacement that removes possible misalignments.
   (d) Any eventual holes of the resulting image (produced by removing the haptic device or the misaligned hand) are re-filled with a view-dependent rendering of the background. This step is based on an IBR algorithm and a set of prerecorded images of the background.
4. The final result is displayed to the user.

6.4.1 Input Data

Our algorithm takes two types of data as input: *static* and *dynamic*. The static data is captured or computed only once during the System Calibration stage, before entering the run-time pipeline. The static data consists of:

- The intrinsic camera parameters.
- The arrangement of marker geometry in the world. This includes the definition of the global reference system of the environment.
- A set of images of the background scene (as many as required by the IBR model), acquired from a random set of viewpoints sampling the space where the user is expected to move during the actual interaction.
- For each image of the set, the extrinsic camera parameters, i.e., the camera pose, computed with respect to the global reference system.
- A geometric proxy of the background, i.e., several world points with an approximately known position.
- The transformation between the local reference system of the haptic device and the global reference system.
- An approximate geometric and kinematic model of the haptic device.
- The color histogram of the user's skin.

At run-time, the algorithm needs only the following *dynamic data*:

- An image of the scene coupled with the camera's pose from which it was captured, for each eye.
- The configuration of the haptic device in its local frame.

The VHMR display algorithm automatically computes all the information necessary to execute all the steps in the pipeline described above using as inputs the combination of static and dynamic data.

Note that, compared with a classic VHMR display system, our display algorithm does not require additional hardware. The pose estimation can be computed at each

frame by the same tracking strategy used for typical MR pipelines, provided that the scene enjoys good visibility and illumination of some marker. The configuration of the haptic device is typically retrieved from the device's own driver library.

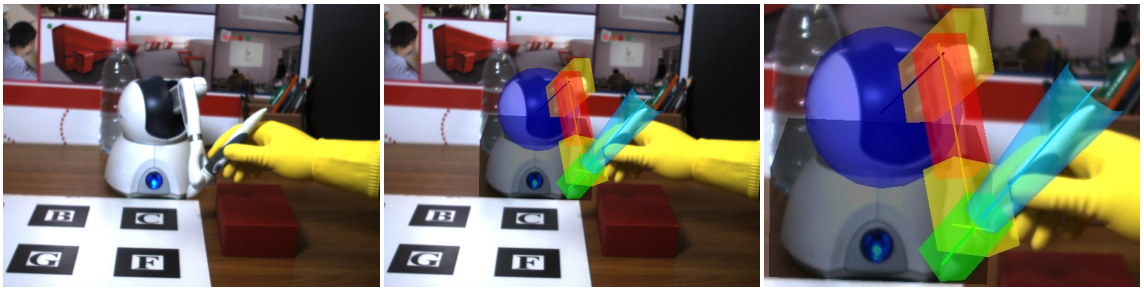
6.5 Semantic Labeling in screen-space

This step of the pipeline performs the computation of two different mask-regions required, for each frame, by the subsequent rendering steps. The first mask-region regards the part of the image obtruded by the haptic device. The mask is used to apply the *removing* algorithm only where it is actually needed, thus improving rendering performance. The second mask-region regards the user's hand. This mask is mainly used as a filter for occlusion handling purposes.

6.5.1 Visually obtruded region in screen-space

This section gives some details about the task of the pipeline (see. Sec. 6.4) that labels the region of the image obtruded by the haptic device.

I solved this crucial task with a mechanical approach, made of two basic steps: (i)



(a) (b) (c)

Figure 6.4: *Visually obtruded region in Screen-Space. Image (a) shows the raw image, coming from the video-see-through HMD. Image (b) shows the mask produced with the approach described in Sec. 6.5.1. Image (c) focuses on the colored mask overlayed upon the raw image: each color means a different body of the kinematic model.*

a 3d virtual replica of the haptic device is connected to the real one in order to follow its movement; (ii) the projection of each body-element of the virtual replica on the current image plane compose the desired label. Step (i) relies on a direct-kinematic model of the device, used to compute the kinematic configuration of the device in the local reference, and on the calibration results described in Sec. 6.3.2, used to properly align the local reference to the world reference.

Given the approximate geometric model of the haptic device, the transformation from the global to the local reference frame of the device, and the current configuration of the device (e.g., joint angles), we can place the approximate geometric model in the virtual world. Fig. 6.4-down shows the approximate geometric model blended on top of the actual haptic device in one of our demonstration examples.

The extrinsic camera parameters for the current frame complete the definition of model view and projection matrices, and the approximate geometry of the device can be rendered onto the screen-space. When performing this rendering step, a mask in the stencil buffer is activated for the rendered pixels. This mask defines the occluded region in screen-space (shown as a multi-color overlay in Fig. 6.4-right).

There are other possible options for identifying the occluded region in screen-space. One of them is to paint the device with a characteristic color and use it as a chroma. This approach would be more robust if calibration problems might occur, but it could also suffer from color and/or lighting issues. In my experiments, the tracking robustness of the haptic device appeared to be sufficient, and I found that the approach based on mask rendering was sufficiently accurate. Note that with our method is possible to manually tune the level of approximation of the geometric model, in order to compensate any eventual lack of accuracy due to the haptic hardware. In fact, in presence of accuracy problems, one can slightly increase the size of the virtual model, in order to completely remove the haptic from the scene.

6.5.2 Segmenting user's hand in screen-space

Implementation details of the color-based detection algorithm used for the segmentation are given in Sec. 4.2.4. To achieve an acceptable quality of the skin-detection

results, there are two different possibilities: one is to increase the saturation of the video camera, near to a level of 100%, thus altering the usual color perception; the second one is to use a colored glove, as shown in Fig. 6.6, that performed a more robust effect.

After the hand-mask is computed, the occlusion handling strategy can be applied segmenting the user's hand from the image and then repainting it in front of the virtual tool. Further details are given in the following section.

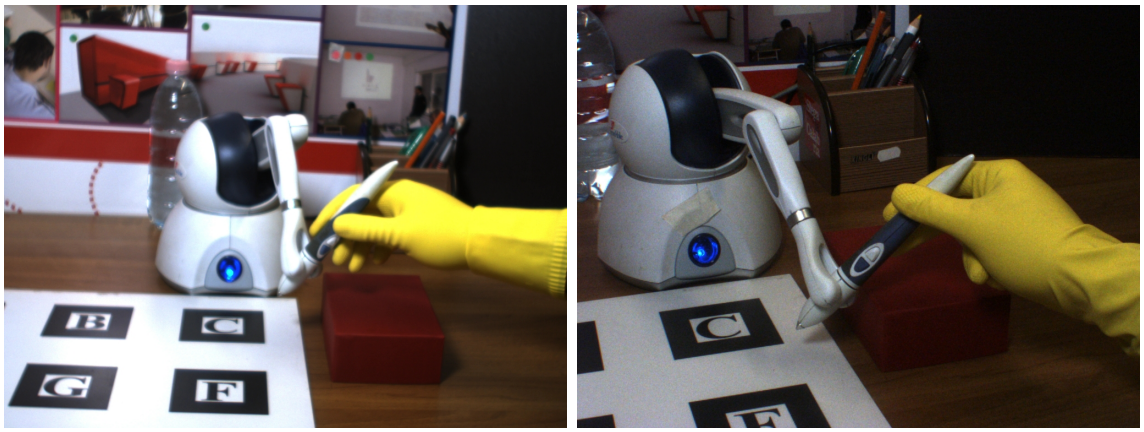
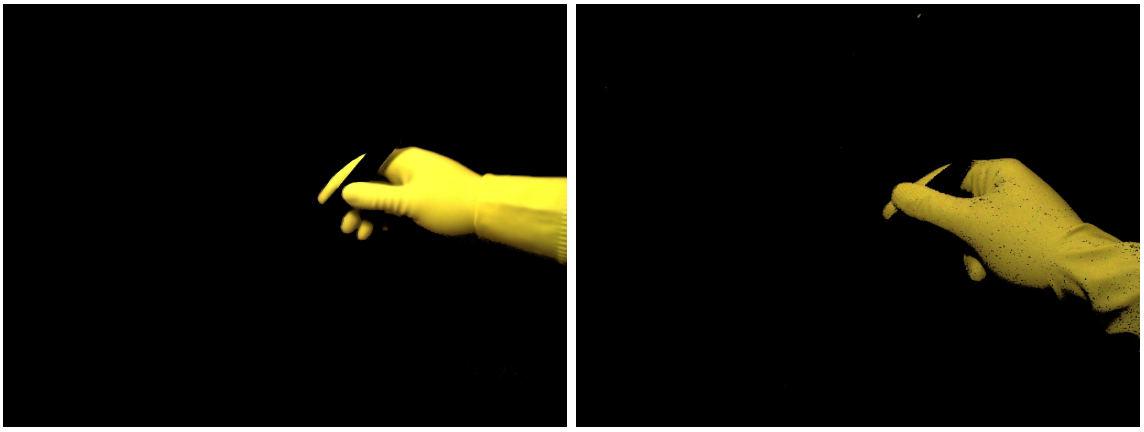


Figure 6.5: *Hand-Mask of the current frame.*



6.6 Perceptually Consistent Compositing

This section describes the final part of the pipeline, concerning the rendering tasks executed since the addition of the virtual graphics (point 6-8 of the pipeline, see Sec. 6.4), just before showing the rendered frame to the user. This part of the pipeline is com-

pletely designed to increase the sense of presence of the environment. The goal is to solve three "perceptual barriers" related to the current state-of-the-art in visuo-haptic display: *mutual occlusion*, *virtual tool misalignments* and *visual obtrusion*.

6.6.1 Mutual occlusion

The first problem is related to the mutual occlusions occurring between user's hand and the handled tool. When we grasp a pen, we are used to see our hand occluding the handled part of the pen. Therefore, the key concept of our occlusion handling strategy is to simulate the occlusion between the user's hand and the virtual tool, replicating the visual occlusion occurring between the hand and the handled tool of the haptic device. The hand is segmented (using the algorithm described in Sec. 3.3) and repainted in front of the virtual tool, only where the haptic tool is occluded and the hand is visible. This strategy is not conceived to manage the occlusion between the hand and a generally placed virtual object: we refer only to a scenario in which the user's hand and the virtual tool are placed between the user's point of view and the augmented scenario. This assumption is required in order to simplify the computation of the occlusion, thus maintaining the frame-rate adequate for real-time renderings. The main limit of this solution is related to the shape of the virtual tool used. To perform a well grasping visual effect, it would be desirable to adjust the shape of the haptic handle in order to be congruent to the virtual tool.

6.6.2 Virtual Tool misalignments

The second problem is given by the virtual tool misalignment, that may occur when the virtual tool is constrained by a wall. In Fig. 6.6, for example, the virtual tool is colliding with the desk, and the physics simulation is preventing that the tool penetrates the colliding surface.

In the meantime, haptic feedback is computed by the haptic rendering algorithm, and then displayed to the user through the device. However, commodity haptic hardware suffers of mechanical limitations that may restrain the ability to render the virtual

objects up to their full dynamic range. In the showed example, the device is not strong enough to prevent the user from pushing inside the table. This may produce a non-consistent perception, due to the conflicting information coming from visual and tactile stimuli. From the visual side, the user observes an undesirable and oscillating misalignment of the virtual tool in respect of his/her hand. From the haptic side, the user is grasping the tool, and tactile receptors give him/her the sensation that the tool is not moving in respect of the hand. I propose an original methodology able to delete the conflict between visual and haptic perception, by adding a compensating displacement to the rendered user's hand.

Note that, to avoid the misalignment between the hand and the handled tools, the algorithm needs to compute a displacement vector in screen-space. At each frame, the procedure retrieves the separation vector, $\overline{S}$, computed by the haptic rendering algorithm, that represents the rigid displacement between the physical tool and its virtual instance. This information is used to extract the motion component parallel to

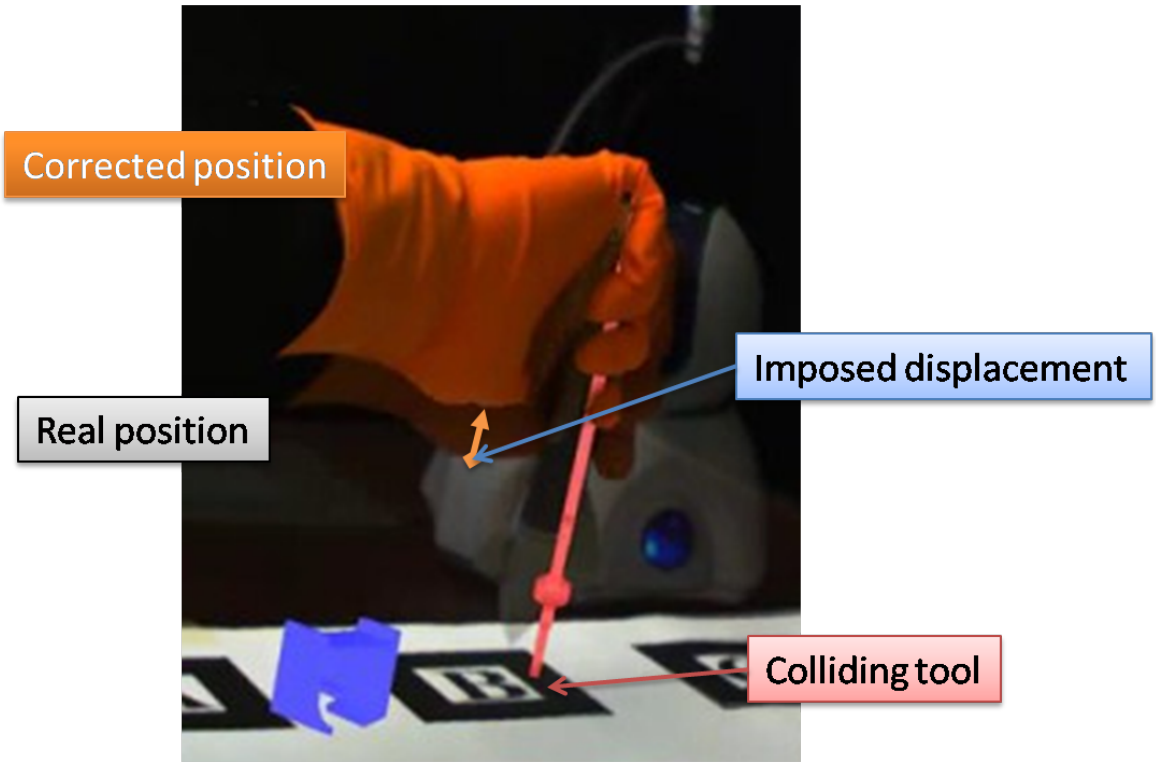


Figure 6.6: *Haptic tool misalignments*

the image plane, considering also the scale factor introduced by the projective geometry. This is done by projecting the origin and the tip of the vector on the image plane. The projected vector is then used to apply the compensating translation to the user'hand.

6.6.3 Visual Obtrusion

The last problem concerns the visual presence of the haptic device. In order to add haptic interaction to the Visuo-Haptic environment, we need to introduce the haptic device on the scene. However, the visualization of the device is, perceptually speaking, something that may distort the realistic understanding of the real scene. The solution of this bug relies in allowing the user to perceive only the meaningful stimulus provided through the haptic device, i.e. the kinesthetic feedback. This is mainly achieved using the IBR technique described at Sec. 3.2, as depicted in Fig. 6.7. The key idea is to store the scene when the haptic device is absent, and then use IBR methodology in order to refill the background where it is obtruded by the haptic device.

The algorithm is executed on the IBR-subset of the mesh defined considering all the triangles that have at least one vertex within the device-mask, marked in pink in Fig. 6.7-b. In the same figure, the subset marked in yellow depicts the region where the linear interpolation between the real image of the background and the synthesized one is performed.

In particular, final version of the algorithm was modified to enlarge the IBR-subset including a one-trianlge-ring. This is necessary to improve robustness of the method against several inaccuracy of the system, like low precision of the device, jittering and inaccuracy of the tracking, but especially to compensate the errors due to the IBR method (Sec. 3.2).

6.6.4 The semantical pipeline

The Perceptually Consistent Compositing task is made up of four distinct steps, as already discussed in Sec. 6.4. Step *a* regards the deletion of both the haptic device and the user's hand from the image. More exactly, in the current implementation of the

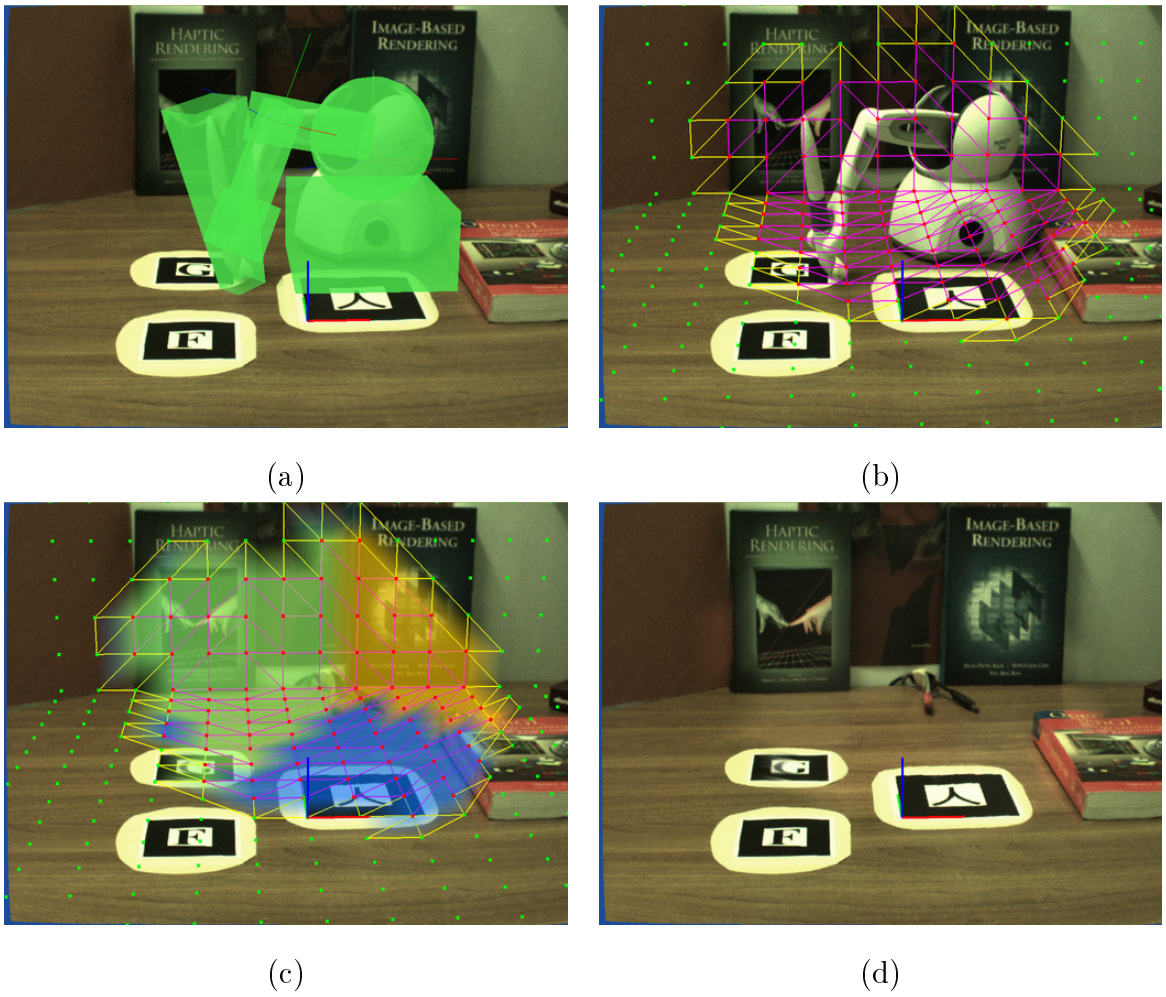


Figure 6.7: *Visual Obtrusion managed through IBR. At each iteration, the haptic device region is marked for removal (a), the proxy is projected to the image (b) and the camera blend field is used to refill the obtruded region.*

algorithm no deletion takes place at this stage.

The *Haptic Device*-mask, D, and the *hand*-mask, H, are used to semantically compute the *removing*-mask, R.

$$R = D \cup H$$

The definition of R is the output of step a.

Next two steps, b and c, are related respectively to the composition of virtual graphics and user's hand to current image. Using stencil buffer, other two mask are produced during the above-mentioned steps: the $VirtualGraphics$-mask, VG, and the *corrected_hand*-mask, CH. These two are combined to compute the *overlay*-mask,

O, relative to the region of the image upon which something else is displayed.

$$O = VG \cup CH$$

The last semantical operation regards the computation of the *repaint_background*-mask, B, that affects the region of the image for which is needed to synthetically compute the rendering of the background.

$$B = R \ominus O = R - (R \cap O)$$

R is computed as the semantical difference, $\ominus$, of R by O, meaning that B is obtained by removing from R its intersection with O.

Once B is defined, an image of the background relative to the current position of the user is synthesized with the IBR methodology described at Sec. 3.2, using the dataset stored during the scene capture (see. Sec. 6.3.1); this image is rendered on top of the current image, using the B mask as a filter.

6.7 Performance Results

We have evaluated the mixed reality paradigm on scenes with combined visuo-haptic feedback. The algorithm for visually removing the haptic device from the mixed scene is combined with state-of-the-art methods for haptic rendering of the interaction between a virtual tool and other deformable and rigid objects, both virtual and real. We have used the following framework for our tests: A PHANToM Omni haptic device by SensAble Technologies, a head-mounted display composed of a Z800 3D visor by eMagin with two external flea2-08S2C cameras by Point Grey, and a quad 2.83-GHz processor PC with 4-Gb of RAM. We recorded 121 images for the background model, and sampled its geometric proxy with 597 vertices. Under these settings, the execution of the visual subtraction of the haptic device does not remarkably affect the system performance: the rendering frame rate is 30 fps for the standard AR system, and remains 30 fps after the addition of our algorithm.

In details, an OpenGL application is commonly divided into two cyclic threads: *idle* and *display*. Typically, all the rendering processes are managed by the display function,

while the idle function is executed more intensively until the application requires to render a new frame (that happens at the video refresh rate).

The basic MR application, i.e. a trial application that retrieves and displays the image from the camera, executes the tracking and augment a virtual cube on top of the marker, showed a performance of 30 fps for the rendering and about 600 Khz for idle frequency. Note that, in this situation, the idle function only executes a non blocking call to the camera for checking if a new image is available. Therefore, when a new image is available, the display is invoked and a new frame is rendered: this means that the capturing rate of the camera, 30Hz, represents a physical upper-bound limit for the rendering rate.

Enabling both the IBR algorithm and the hand occlusion feature, the rendering performance maintains its upper-bound value of 30 fps, while the idle frequency slows down to 350-400 Khz. This is due to the fact that most of the pipeline was implemented using shading programming, thus allowing its direct execution on the GPU.

Next, we incorporated dynamic virtual objects to the scene. The physically-based simulation takes the frame rate down to 20 fps, that occasionally worses to 15 fps in very complex contact configurations, but we have used the multirate haptic rendering algorithm of [Garre & Otaduy 2009] to maintain stable interaction.

Our first experiments involved only visual deletion of the device from the scene, as shown in Fig. 6.7. Possible run-time tracking and device calibration errors required a slight scaling of the virtual model of the haptic device. However, this scaling was minimal, as shown in Fig. 6.4. The quality of the IBR implementation was not optimal, because the books were not part of the geometric proxies, leading to possible ghosting effects, and the illumination varied from the capture session to the time when some of the examples were computed.

Next, we incorporated dynamic virtual objects to the scene. In these examples we have used a wrench as the virtual tool, shown in Fig. 6.8, which visually replaces the handle of the haptic device. The physically-based model of the virtual environment also incorporates virtual models (not depicted in the figure) of the desk and of the red box, which allows the computation of contact with real objects. Although not included

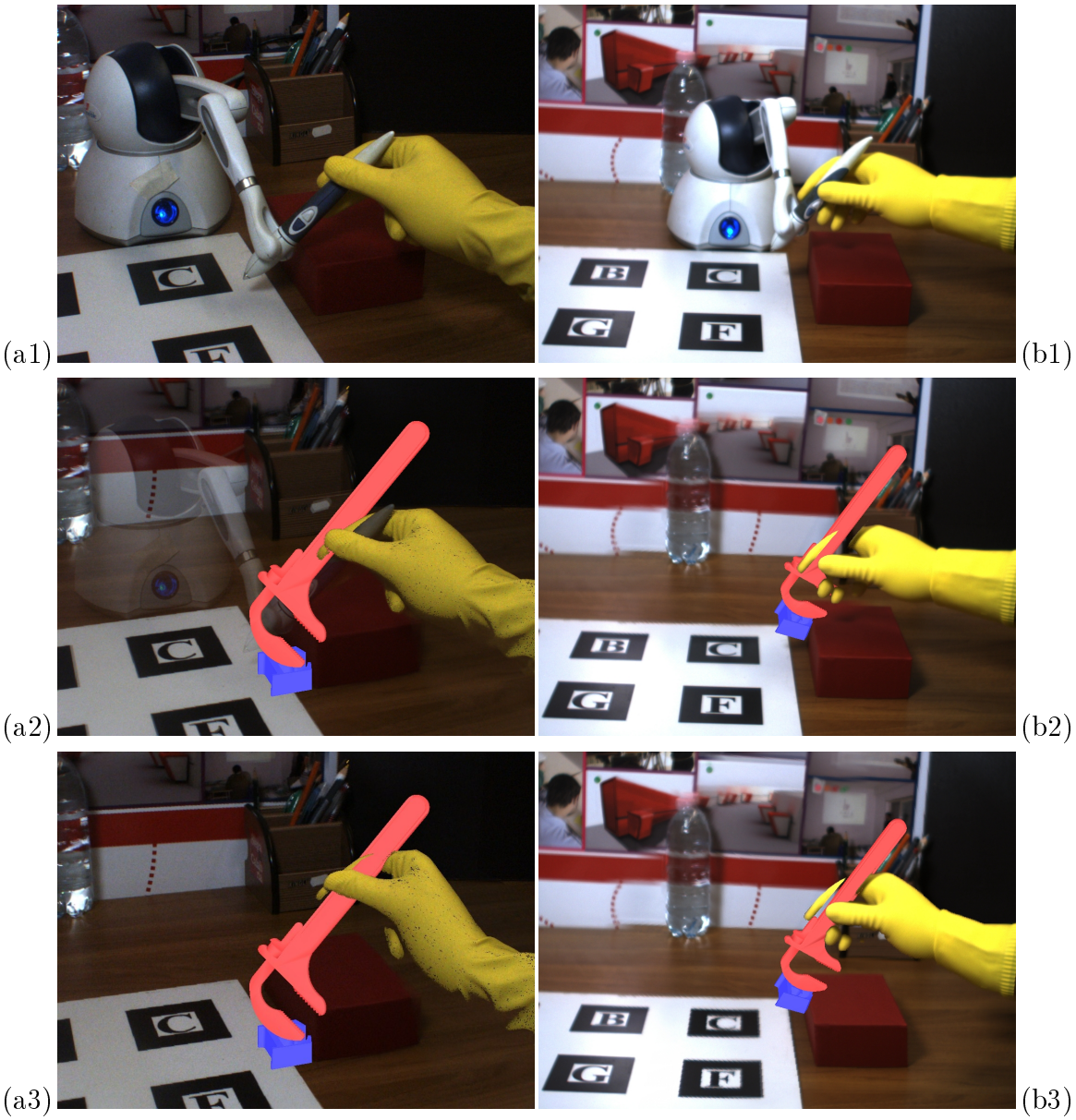


Figure 6.8: *Two different instant, (a) and (b), coming from the same demo: haptic interaction with both virtual (the letter) and real (the red box) objects. (a1) and (b1) images are the original frames. (a2) and (b2) images are obtained after the visual obtrusion removal, in particular in (a2) the device is only partially deleted (only for demonstrations purposes). (a3) and (b3) images show the final result, obtained after that the misalignments between the hand and the wrench is corrected.*

in our proof-of-concept implementation, these virtual models of real objects would also allow casting shadows of real objects on virtual objects and vice versa.

6.8 Conclusion

In this chapter, I have presented a novel mixed reality paradigm that enables compelling visuo-haptic augmented reality without the visual obtrusion introduced by haptic devices. The key element of the mixed reality paradigm is a computational approach for the camouflage of the haptic device, using an image-based model of the context scene. I believe that this mixed reality paradigm can increase versatility and outreach of visuo-haptic mixed reality.

Another great contribution of the proposed mixed reality paradigm is the correct display of the user's own hand. In the presented proof-of-concept implementation, I have realized a perceptually consistent compositing that not only solves the occlusion between the hand and the virtual handheld tool, but visually corrects the haptic device deficits in terms of strength, preventing the display of some disturbing misalignments between the hand and the tool.

I believe that the validation of the proposed VHMR paradigm through some test case will definitely confirm the improvements in terms of acceptability, sense of presence and cross-modal congruency. Moreover, the methodology could be easily extended

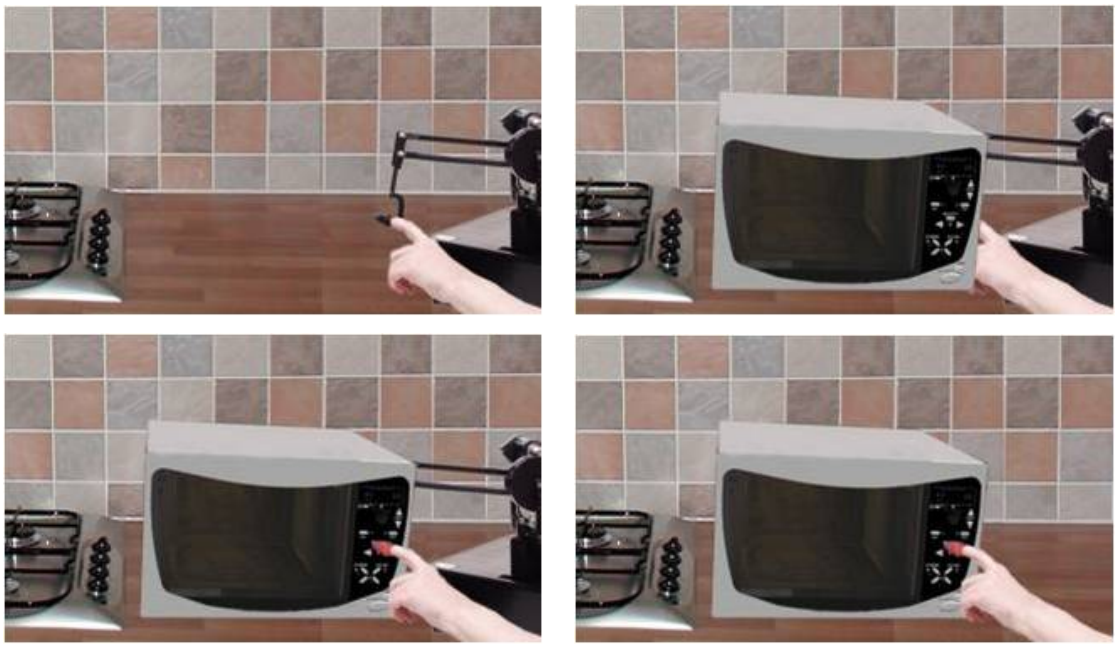


Figure 6.9: *Towards a more flexible Mixed Prototyping approach*

to the direct interaction scenarios (see Fig. 1.4) through the use of more general haptic devices (e.g. a haptic glove), and oriented to both experts and final users. Fig. 6.9 permits to qualitatively understand the visual impact of the proposed method, depicting a feasible usability application in which the final user interacts directly with the object through the use of a single-finger haptic interface, without being disturbed by occlusions and obtrusions.

Chapter 7

Conclusions

7.1 Summary of Results

Interactive Augmented Prototyping, also referred as Interactive Mixed Prototyping (IMP), was recently defined by Verlinden and Horwath' works [Verlinden & Horvath 2006, Verlinden & Horvath 2007, Verlinden & Horvath 2009]. It is a promising prototyping approach, that they define as "the combination of physical and virtual models through Augmented Reality (AR) and Rapid Prototyping". The IMP approach presented in this dissertation extends the original approach in at least three directions. First, I have successfully defined a methodology capable to connect the real-time execution of an industrial simulation environment, Simulink, into a Virtual Reality environment. This innovative approach was developed following the idea that the CACE software, usually employed by the control, electrics or electronics engineers, can also be used to simulate the behavior of the Human-Machine Interfaces (HMI) product in Virtual Reality (VR)/Mixed Reality (MR). The proposed approach overcomes the classical problems related to the implementation of the code inside the VR/MR software to replicate the product behavior. This methodology was applied for modeling the behavioral simulation of the two test-cases described in chapter 4 and chapter 5. The implemented test-case puts in evidence that the main advantages of our approach are:

- the time needed to implement an interactive digital prototype is reduced because there is no need of coding to model the HMI behavior;
- any modification done on the behavior models in the simulation environment is immediately testable in VR/MR;

- the reliability of the HMI simulation in VR/MR is ensured by the direct use of the same solver and the same behavior models that engineers use to design and manufacture the product.

Second, starting from behavioral simulation, I have developed the Hand-based IMP environment (chapter 4), that let the user interact in real time with a mixed prototype using only a common video-see-trough display for AR, and a sensored colored glove. The developed MR environment deals with two different issues usually present in the product behavior simulation in MR: the occlusion between the user's hands and the virtual objects and the interaction between the user and the product interface. Given these challenges, I developed some original Computer Vision components, that exploit just the real-time processing of the Head Mounted Display (HMD) video stream. The proposed technique

- is able to handle the occlusion issue and to recognize the user generated event;
- is implementable with a very low budget;
- performs a good frame rate, ranging from 40 to 60 fps.

The system was definitively validated along a user study aiming to perform usability assessment of a mixed prototype (chapter 5). The results of the study confirmed that the Hand-based IMP system provides a more natural interaction with respect to a Virtual Prototyping approach, especially for the specific task of pushing a button. This suggests that the proposed approach is immediately utilizable in the design of several products where the user interface is made by electromechanical elements like buttons, knobs, LEDs and displays.

Third, I developed the tool-based IMP (chapter 6), that can be thought as complimentary to the hand-based one, because it focuses mainly on applications where the user interactions with the Mixed Prototype are mediated by another object. This was achieved defining a new visuo-haptic display which major contributions were the definition and solution of three typical *perceptual barriers* of this kind of systems: visual obtrusion of the haptic devices, mutual occlusion and collision-related misalignments between the virtual tool and the user's hand.

Moreover, both the IMP environments were defined using a standard MR framework that I developed starting from ARToolkit solution [Kato & Billinghurst 1999], and then I extended it with some original features like color-based occlusion handling, the stereo tracking approach and the Image-based rendering (IBR) background modeling, as well as the hand gesture tracking. The IBR background modeling consist in a new methodology that is capable to implicitly reconstruct the 3d textured model of a scenario, starting from its raw shape and integrating a set of captured images obtained by moving a camera into it.

Both the hand-based and the tool-based environments offer the possibility to interactively evaluate the functioning of a digital prototype. The first approach makes use of RP in order to allow a direct interaction with the product, giving the possibility to directly touch the physical interface of the product with the fingers. This may give a better comfort for the user compared to the tool-based MR environment, but it requires the realization of a Rapid Prototype each time that the shape of the product interface is changed. Having this in mind, it is possible to state that the tool-based environment is suitable for the conceptual design phase when the shape of the product is continuously changed. After that the shape has been fixed, it is possible to employ the hand-based environment in order to make the last evaluations both on the shape and the behavior of the product interface.

7.2 Discussions and Future Works

The topology chart of the technology solutions for Mixed Prototyping depicted in Fig. 7.1 is developed on three main levels: domain of use (expert vs. final users), haptic approach (active vs. passive), interaction typology (by hand or direct vs. by tool). The given charts helps to distinguish between the Mixed Prototyping solutions proposed in this thesis (black text) and a few interesting MP approaches, that I believe of a great interest, that could be easily designed exploiting some of the many features developed in this work.

Keeping in mind that the final goal of the Mixed Prototyping is to accomplish

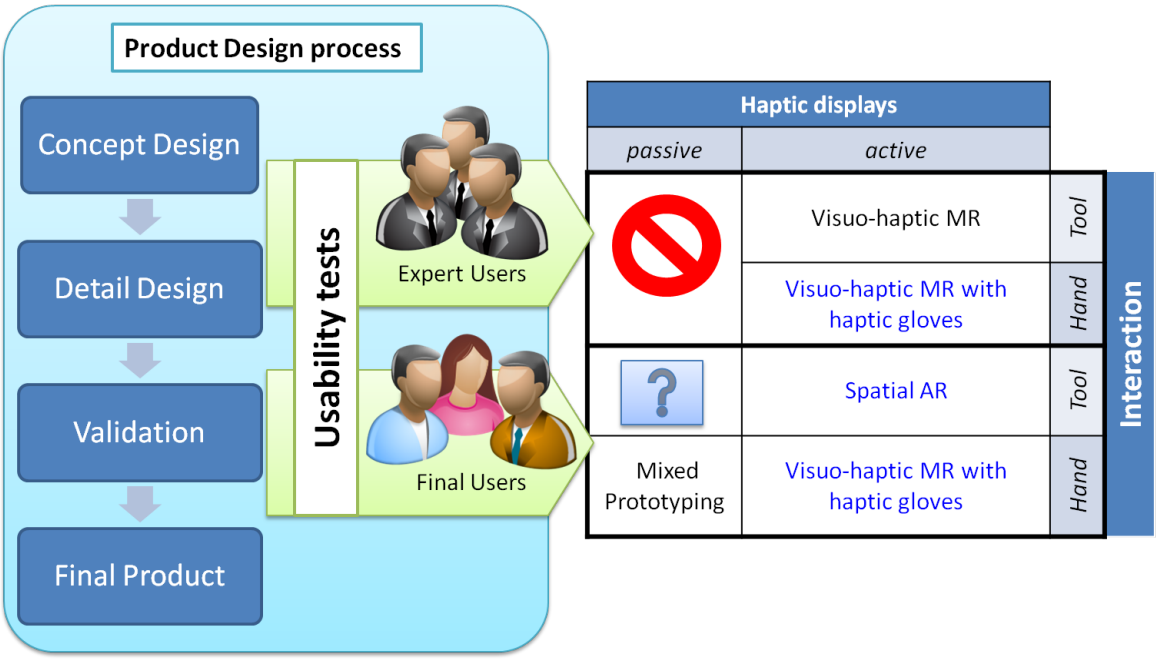


Figure 7.1: *An overview of the explored solutions (black text) and the possible investigations (blue text) in the field of Mixed Prototype following the same classification proposed in Fig. 1.4: domain of use (expert vs. final users), haptic approach (active vs. passive), interaction typology (by hand vs. by tool).*

some usability tests on the product interface using the digital prototype instead of the physical one, I distinguish two different domains for the application of the Mixed Prototyping approach: one is mainly oriented to the use during the first steps of the PDP, while the other is mainly intended for allowing final users to naturally interact with a digital prototype. The following sections reflect this distinction and focus on several limitations of the works proposed in this dissertations suggesting also a set of feasible improvements for each technique proposed.

7.2.1 Mixed Prototyping oriented to *Final* users

The Mixed Prototyping approach, proposed in chapter 4, was successfully validated for usability assessment applications of industrial products (chapter 5).

Its application is oriented only for the final steps of the PDP, where the realistic interaction provided through the passive haptic approach, i.e. with a physical replica

of the product interface, usually realized by rapid prototyping, is not yet achievable using more flexible haptic-based solutions. In some future works (Fig. 7.1), starting from the Visuo-Haptic MR paradigm presented in chapter 6, a similar approach for direct manual interaction could be developed, relying on a full hand haptic device (e.g. haptic glove).

Some other interesting Mixed Prototyping application oriented for final users could be developed exploiting the peculiarities of Spatial Augmented Reality: among all the ability to provide an immersive multi-person experience, without requiring the users to wear any uncomfortable apparatus.

The presented implementations of the test cases put in evidence that the proposed approach is immediately usable in the design of several products where the user interface is made by electromechanical elements like buttons, knobs, LEDs and displays. However, the presented technique is not yet adapt for simulating the interaction with hand held devices, e.g. a mobile phone, where the accuracy of the finger tracking methods is not adequate to distinguish between small buttons. A more flexible solution could be developed in the near future integrating a real-time depth reconstruction procedure, relying on specific hardware able to track hand gestures in real time, similarly to the recently launched *kinect* accessory for gaming applications.

7.2.2 Mixed Prototyping oriented to *Expert* users

As a Mixed Prototyping oriented to *Expert* users I consider the usability tests accomplished by expert users (designers, engineers or scientist) on the early stage of the Product Design Process, and more exactly among all the design steps ranging from Concept Design to Detail design, according to the commonly accepted scheme proposed by [Ulrich *et al.* 1995, Pahl *et al.* 1984].

In this situation I believe that the flexibility of the system, defined as the capability to automatically reflect any change made by designers from the CAD model to the Virtual Prototype, plays a primary goal compared to the reliability, intended as the capacity of the system, used as a communication medium, to not introduce perceptual barriers, i.e. to not distort or add false informations into the communicative process.

To achieve flexibility, the system interface should rely on an adaptive architecture: in this dissertation I propose the use of haptic devices combined with the Mixed Reality technology, also referred as Visuo-Haptic technology. Moreover, I developed a novel Visuo-Haptic Mixed Reality paradigm mainly oriented for tool-based interactions, where the user is able *to see and interact with* both real and virtual objects. Further analysis could deal with the adoption of the same VHMR environment, or similar Haptic-based solutions, for the case of direct interaction (hand row in Fig. 7.1) relying on haptic glove, or similar devices.

The presented proof-of-concept implementation still lacks several advances in mixed reality and image-based rendering that, although orthogonal to our work, could increase the quality and versatility of the results. Some of these advances include, but are not limited to:

- a more robust tracking technology, using one of the natural feature tracking methods mentioned in Sec. 2.1.2;
- image-based rendering with refined depth estimation and geometric proxies, more dense input acquisition together with advanced closest-ray-search data structures, or even on-the-fly modeling techniques (as the one recently proposed in [Newcombe & Davison 2010] and depicted in Fig. 7.2);
- full occlusion handling, i.e. managing not only the occlusion between user's hand and the virtual tool but, more in general, among virtual and real objects using,

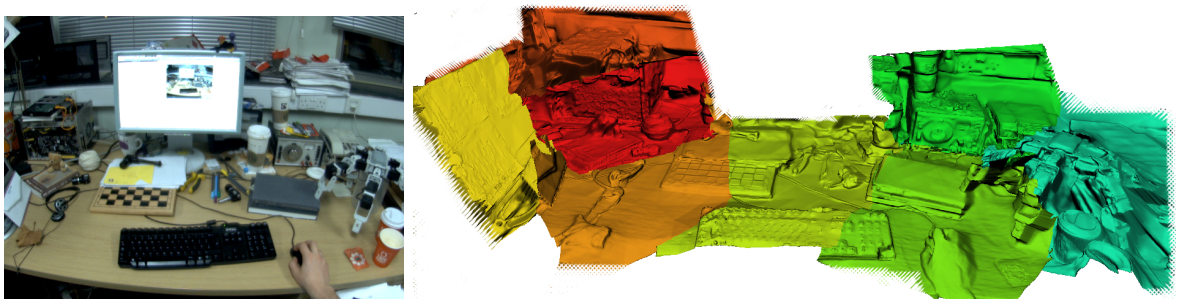


Figure 7.2: *An example of the Live Dense Reconstruction method recently proposed in [Newcombe & Davison 2010]: a detailed depth model (right) is extracted while moving a single camera into an indoor cluttered scenario (left)*

Figure 7.3: *Comparison between basic OpenGL rendering and the methods recently proposed by [Aittala 2010]*

e.g., foreground detection or interactive depth reconstruction;

- enhancing the composition of the augmented virtual graphics: in order to plausibly replicate the visual qualities (illumination, shadows, reflexions, etc.) of the real world scene [Aittala 2010], or to simulate the degradation effects typical of a low-cost camera (softness, chromatic aberration, chroma resolution, and bayer effects) [Klein & Murray 2010], as depicted respectively in Fig. 7.3 and Fig. 7.4.

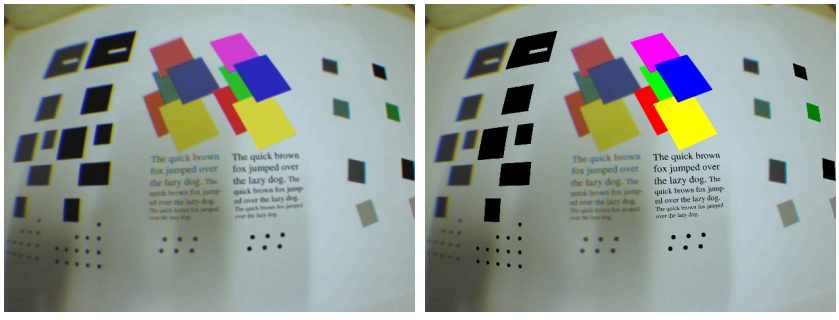


Figure 7.4: *Comparison between basic OpenGL rendering and the methods recently proposed by [Klein & Murray 2010]*

The mixed reality paradigm could also be enhanced to support dynamic backgrounds. This would require run-time acquisition of the background, e.g., with additional cameras. In principle the inclusion of a passive dynamic background appears

feasible, as long as the capture includes a rough depth estimation. It is more difficult, however, to include an active dynamic background that interacts with the virtual objects, as this requires virtual replicas of the dynamic background objects in the physically-based model of the environment.

Bibliography

[Abidi & Chandra 1995] M.A. Abidi and T. Chandra. *A new efficient and direct solution for pose estimation using quadrangular targets: algorithm and evaluation.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 17, no. 5, pages 534–538, 1995. 17

[Acal & Lobera 2007] A. Acal and A. Lobera. *Virtual reality simulation applied to a numerical control milling machine.* International Journal on Interactive Design and Manufacturing, vol. 1, pages 143–154, 2007. 10.1007/s12008-007-0016-2. 27

[Adelson & Bergen 1991] Edward H. Adelson and James R. Bergen. *The Plenoptic Function and the Elements of Early Vision.* In M. Landy and J.A. Movshon, editeurs, Computational Models of Visual Processing, chapitre 1. MIT Press, 1991. 109

[Aittala 2010] M. Aittala. *Inverse lighting and photorealistic rendering for augmented reality.* The Visual Computer, vol. 26, no. 6, pages 669–678, 2010. x, 133

[Ansar & Daniilidis 2003] Adnan Ansar and Kostas Daniilidis. *Linear Pose Estimation from Points or Lines.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, pages 578–589, 2003. 17

[Antonya & Talaba 2007] Csaba Antonya and Doru Talaba. *Design evaluation and modification of mechanical systems in virtual environments.* Virtual Reality, vol. 11, no. 4, pages 275–285, 2007. 28

[Aoyama & Kimishima 2009] H. Aoyama and Y. Kimishima. *Mixed reality system for evaluating designability and operability of information appliances.* International Journal of Interactive Design and Manufacturing, vol. 3, pages 157–164, 2009. vii, 28, 30, 31, 32, 33, 62, 65, 68

[Araujo *et al.* 1998] H. Araujo, R. L. Carceroni and C. M. Brown. *A fully projective formulation to improve the accuracy of Lowe's pose-estimation algorithm.* Computer Vision and Image Understanding, vol. 70, no. 2, pages 227–238, 1998. 17

[Azuma *et al.* 2001] R. Azuma, Y. Baillot, R. Behringer, S. Feiner, S. Julier and B. MacIntyre. *Recent advances in augmented reality.* Ieee Computer Graphics and Applications, vol. 21, no. 6, pages 34–47, 2001. 14

[Azuma 1997] R. T. Azuma. *A survey of augmented reality.* Presence-Teleoperators and Virtual Environments, vol. 6, no. 4, pages 355–385, 1997. 12, 14

[Bajura & Neumann 1995] M. Bajura and U. Neumann. *Dynamic registration correction in video-based Augmented Reality Systems.* Ieee Computer Graphics and Applications, vol. 15, no. 5, pages 52–60, 1995. 16

[Barbieri *et al.* 2008] L. Barbieri, F. Bruno, F. Caruso and M. Muzzupappa. *Innovative integration techniques between Virtual Reality systems and CAx tools.* The International Journal of Advanced Manufacturing Technology, vol. 38, no. 11/12, pages 1085–1097, 2008. 28, 62

[Bay *et al.* 2006] H. Bay, T. Tuytelaars and L. Van Gool. *Surf: Speeded up robust features.* Computer Vision–ECCV 2006, pages 404–417, 2006. 21

[Bergamasco *et al.* 2005] M. Bergamasco, S. Perotti, C.A. Avizzano, M. Angerilli, M. Carrozzino and E. Ruffaldi. *Fork-lift truck simulator for training in industrial environment.* In Emerging Technologies and Factory Automation, 2005. ETFA 2005. 10th IEEE Conference on, volume 1, pages 5 pp. –693, sept. 2005. 27

[Bianchi *et al.* 2006] G. Bianchi, C. Jung, B. Knoerlein, G. Szekely and M. Harders. *High-fidelity visuo-haptic interaction with virtual objects in multi-modal AR systems.* Proc. of ISMAR, 2006. 25

[Bordegoni *et al.* 2008] Monica Bordegoni, Giandomenico Caruso and Francesco Ferrise. *Mixed-Reality Environment based on Haptic Control System for a tractor cabin design review.* In Proceeding of CIRP Design Conference, 2008. 89

[Bordegoni *et al.* 2009] M. Bordegoni, U. Cugini, G. Caruso and S. Polistena. *Mixed prototyping for product assessment: a reference framework.* International Journal on Interactive Design and Manufacturing, vol. 3, pages 177–187, 2009. 28, 30, 31, 62, 68

[Bordegoni 2010] Monica Bordegoni. *Virtual Prototyping to anticipate products shapes and functions.* Seminar at the Virtual Prototyping Summer School, Milano, 21-24 September, 2010. 3

[Bouguet 2008] Jean-Yves Bouguet. *Camera Calibration Toolbox for Matlab*, June 2008. `http://www.vision.caltech.edu/bouguetj/calib_doc/`. 107

[Brederson *et al.* 2008] J. D. Brederson, M. Iktis, C. R. Johnson and C. D. Hansen. *The Visual Haptic Workbench.* Proc. of PHANToM User Group Workshop, 2008. 25

[Breen *et al.* 1996] David E. Breen, Ross T. Whitaker, Eric Rose and Mihran Tuceryan. *Interactive Occlusion and Automatic Object Placement for Augmented Reality.* Comput. Graph. Forum, vol. 15, no. 3, pages 11–22, 1996. 22

[Bruno & Muzzupappa 2010] Fabio Bruno and Maurizio Muzzupappa. *Product interface design: A participatory approach based on virtual reality.* International Journal of Human Computer Studies, 2010. 60, 86, 87, 90

[Bruno *et al.* 2006] F. Bruno, R.M. Mattanò, M. Muzzupappa and M. Pina. *A new approach to participatory design: usability tests in virtual environment.* In X. Fischer and D. Coutellier, editeurs, Research in Interactive Design, pages 80–90. Springer-Verlag, New York, NY, 2006. 79

[Bruno *et al.* 2007] F. Bruno, R.M. Mattanò, M. Muzzupappa and M. Pina. *Design for usability in virtual environment.* In Proceedings of ICED - International Conference on Engineering Design, 2007. 27, 28, 62, 65, 79

[Bruno *et al.* 2009] F. Bruno, F. Caruso, K. Li, A. Milite and M. Muzzupappa. *Dynamic simulation of virtual prototypes in immersive environment.* The International Journal of Advanced Manufacturing Technology, vol. 43, no. 5/6, pages 620–630, 2009. 63

[Buehler *et al.* 2001] C. Buehler, M. Bosse, L. McMillan, S. J. Gortler and M. F. Cohen. *Unstructured Lumigraph Rendering.* Proc. of ACM SIGGRAPH, 2001. 49, 50

[Caruso *et al.* 2007] Giandomenico Caruso, U. Giraudo, Francesco Ferrise and Monica Bordegoni. *Ergonomic interactive testing in a mixed-reality environment.* In Proceeding of HCI International, pages 431–440, 2007. 89

[Caudell & Mizell 1992] T.P. Caudell and D.W. Mizell. *Augmented reality: an application of heads-up display technology to manual manufacturing processes.* In System Sciences, 1992. Proceedings of the Twenty-Fifth Hawaii International Conference on, volume ii, pages 659 –669 vol.2, jan. 1992. 13

[Chen & Li 2010] Zhuo Chen and Xinyu Li. *Markless tracking based on natural feature for Augmented Reality.* In Educational and Information Technology (ICEIT), 2010 International Conference on, volume 2, pages V2–126 –V2–129, sept. 2010. 21

[Colgate *et al.* 1995] J.E. Colgate, M.C. Stanley and J.M. Brown. *Issues in the haptic display of tool use.* In IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 140–145, 1995. vii, 24

[Congedo *et al.* 2006] M. Congedo, A. Lécuyer and E. Gentaz. *The Influence of Spatial De-location on Perceptual Integration of Vision and Touch.* Presence: Teleoperators and Virtual Environments, vol. 15, no. 3, 2006. 25

[Coquillart 2007] S. Coquillart. *A First-Person Visuo-Haptic Environment.* Proc. of HCII, 2007. 25

[Cosco *et al.* 2009] F.I. Cosco, C. Garre, F. Bruno, M. Muzzupappa and M.A. Otaduy. *Augmented Touch without Visual Obtrusion.* In Proc. of ISMAR, pages 99–102, October 2009. 105

[Cuadrado *et al.* 2004] J. Cuadrado, M. Gonzalez, R. Gutierrez and M.A. Naya. *Real time MBS formulations: towards virtual engineering.* In Product engineering: eco-design, technologies and green energies, pages 253–272. Springer, Heidelberg, 2004. 28

[Cugini 2010] Umberto Cugini. *Why AR technologies will bring prototypes into real world?* Seminar at the Virtual Prototyping Summer School, Milano, 21-24 September, 2010. vii, 3

[Debevec *et al.* 1996] P. Debevec, C. Taylor and J. Malik. *Modeling and Rendering Architecture from Photographs.* Proc. of ACM SIGGRAPH, 1996. 49, 105

[Eberhard & Li 2006] P. Eberhard and Z. Li. *Virtual reality simulation of multibody systems.* In Proceedings of EUROMECH colloquium 476, 2006. 28

[Fiala 2005a] M. Fiala. *ARTag, a fiducial marker system using digital techniques.* In Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), volume 2, pages 590 – 596 vol. 2, jun. 2005. 19

[Fiala 2005b] M. Fiala. *Comparing ARTag and ARToolkit Plus fiducial marker systems.* In Proceedings of the IEEE International Workshop on Haptic Audio Visual Environments and their Applications, pages 148 – 153, 2005. 19

[Fiala 2010] M. Fiala. *Designing Highly Reliable Fiducial Markers.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 32, no. 7, pages 1317 –1324, jul. 2010. 19

[Fiorentino *et al.* 2002] M. Fiorentino, R. de Amicis, G. Monno and A. Stork. *Spacedesign: A Mixed Reality Workspace for Aesthetic Industrial Design.* In Proceedings of the International Symposium on Mixed and Augmented Reality - ISMAR02, 2002. vii, 30, 31

[Fischer *et al.* 2007] J. Fischer, B. Huhle and A. Schilling. *Using Time-of-Flight Range Data for Occlusion Handling in Augmented Reality.* In Eurographics Symposium on Virtual Environments (EGVE), pages 109–116, 2007. 22

[Fornaro *et al.* 2008] J. Fornaro, M. Harders, M. Keel, B. Marincek, O. Trentz, G. Szekely and T. Frauenfelder. *Interactive visuo-haptic surgical planning tool for pelvic and acetabular fractures.* Proc. of MMVR, 2008. 23

[Fuhrmann *et al.* 1999] Anton Fuhrmann, Gerd Hesina, François Faure and Michael Gervautz. *Occlusion in collaborative augmented environments.* Computers and Graphics, vol. 23, no. 6, pages 809 – 819, 1999. 22

[Garre & Otaduy 2009] Carlos Garre and Miguel A. Otaduy. *Haptic Rendering of Complex Deformations through Handle-Space Force Linearization.* In Proc. of World Haptics Conference, mar 2009. 25, 106, 123

[Gerhard Shahzad *et al.* 2002] Malik Gerhard Shahzad, Gerhard Roth and Chris Mcdonald. *Robust 2D Tracking for Real-Time Augmented Reality.* In Proceedings of Conference on Vision Interface, 2002. 17

[Gonzales & Woods 2008] R. C. Gonzales and R. E. Woods. Digital image processing. Pearson Education, 2008. 54

[Gordon *et al.* 2002] G. Gordon, M. Billinghurst, M. Bell, J. Woodfill, B. Kowalik, A. Erendi and J. Tilander. *The use of dense stereo range data in augmented reality.* In Mixed and Augmented Reality, 2002. ISMAR 2002. Proceedings. International Symposium on, pages 14 – 23, 2002. 22

[Grantcharov 2006] Teodor Grantcharov. *Virtual reality simulation in training and assessment of laparoscopic skills.* European Clinics in Obstetrics and Gynaecology, vol. 2, pages 197–200, 2006. 10.1007/s11296-006-0054-5. 27

[Gruber *et al.* 2010] L. Gruber, S. Zollman, D. Wagner, D. Schmalstieg and T. H "ollerer. *Optimization of Target Objects for Natural Feature Tracking.* In 20th International Conference on Pattern Recognition, ICPR2010, pages 3607–3610, 2010. 21

[Guan & Wang 2009] T. Guan and C. Wang. *Registration based on scene recognition and natural features tracking techniques for wide-area augmented reality systems.* Multimedia, IEEE Transactions on, vol. 11, no. 8, pages 1393–1406, 2009. 21

[Harders *et al.* 2009] Matthias Harders, Gérald Bianchi, Benjamin Knoerlein and Gábor Székely. *Calibration, Registration, and Synchronization for High Precision Augmented Reality Haptics.* IEEE Transactions on Visualization and Computer Graphics, vol. 15, no. 1, pages 138–149, 2009. 109

[Harel 1987] D. Harel. *Statecharts: a visual formalism for complex system.* Science of Computer Programming, vol. 8, pages 231–274, 1987. 73

[Inami *et al.* 2000] M. Inami, N. Kawakami, D. Sekiguchi, Y. Yanagida, T. Maeda and S. Tachi. *Visuo-haptic display using head-mounted projector.* Proc. of IEEE Virtual Reality Conference, 2000. 26, 104

[Inami *et al.* 2003] M. Inami, N. Kawakami and S. Tachi. *Optical Camouflage Using Retro-Reflective Projection Technology.* Proc. of ISMAR, 2003. 26

[Ishii & Sato 1994] M. Ishii and M. Sato. *A 3D Spatial Interface Device Using Tensed Strings.* Presence, vol. 3, no. 1, 1994. 25

[Jimeno & Puerta 2007] Antonio Jimeno and Alberto Puerta. *State of the art of the virtual reality applied to design and manufacturing processes.* The International

Journal of Advanced Manufacturing Technology, vol. 33, pages 866–874, 2007. 10.1007/s00170-006-0534-2. 27

[Kakumanu *et al.* 2007] P. Kakumanu, S. Makrogiannis and N. Bourbakis. *A survey of skin-color modeling and detection methods.* Pattern Recognition, vol. 40, no. 3, pages 1106–1122, 2007. 54

[Kanbara *et al.* 2000] M. Kanbara, T. Okuma, H. Takemura and N. Yokoya. *A stereoscopic video see-through augmented reality system based on real-time vision-based registration.* In Virtual Reality, 2000. Proceedings. IEEE, pages 255 –262, 2000. 22

[Kanbara *et al.* 2001] Masayuki Kanbara, Naokazu Yokoya and Haruo Takemura. *A Stereo Vision-Based Augmented Reality System with Marker and Natural Feature Tracking.* Virtual Systems and MultiMedia, International Conference on, 2001. 20

[Kato & Billinghurst 1999] Hirokazu Kato and Mark Billinghurst. *Marker Tracking and HMD Calibration for a Video-Based Augmented Reality Conferencing System.* In IWAR '99: Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality, pages 85–94, Washington, DC, USA, 1999. IEEE Computer Society. 17, 18, 33, 37, 41, 107, 129

[Kawano *et al.* 2003] T. Kawano, Y. Ban and K. Uehara. *A coded visual marker for video tracking system based on structured image analysis.* In Mixed and Augmented Reality, 2003. Proceedings. The Second IEEE and ACM International Symposium on, pages 262 – 263, oct. 2003. 17

[Klein & Murray 2008] G. Klein and D. Murray. *Parallel tracking and mapping for small AR workspaces.* In Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on, pages 225–234. IEEE, 2008. 21

[Klein & Murray 2010] G. Klein and D.W. Murray. *Simulating Low-Cost Cameras for Augmented Reality Compositing.* Visualization and Computer Graphics, IEEE Transactions on, vol. 16, no. 3, pages 369–380, 2010. xi, 133

[Knoerlein *et al.* 2007] B. Knoerlein, G. Szekely and M. Harders. *Visuo-Haptic Collaborative Augmented Reality Ping-Pong.* Proc. of Conference on Advances in Computer Entertainment Technology, 2007. 23

[Landauer & Nielsen 1993] T. K. Landauer and J. Nielsen. *A mathematical model of the finding of usability problems.* In Proceeding of ACM INTERCHI'93 Conference, pages 206–213, 1993. 97

[Lee & Park 2005] Woohun Lee and Jun Park. *Augmented Foam: A Tangible Augmented Reality for Product Design.* Mixed and Augmented Reality, IEEE / ACM International Symposium on, vol. 0, pages 106–109, 2005. 23, 30, 33, 67

[Lok 2003] Benjamin Lok. *Incorporating Dynamic Real Objects into Immersive Virtual Environments.* In Symposium on Interactive 3D Graphics, pages 31–40, 2003. 22

[Lowe 2004] D.G. Lowe. *Distinctive image features from scale-invariant keypoints.* International journal of computer vision, vol. 60, no. 2, pages 91–110, 2004. 21

[Lu & Smith 2009] Yuzhu Lu and Shana Smith. *GPU-Based Real-Time Occlusion in an Immersive Augmented Reality Environment.* Journal of Computing and Information Science in Engineering, vol. 9, no. 2, 2009. 22

[Lu *et al.* 1999] S.C.Y. Lu, M. Shpitalni and Gadh R. *Virtual and augmented reality technologies for product realization.* CIRP Annals - Manufacturing Technology, vol. 48, no. 2, pages 471–495, 1999. 30

[Lu *et al.* 2000] C. P. Lu, G. D. Hager and E. Mjolsness. *Fast and globally convergent pose estimation from video images.* Ieee Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 6, pages 610–622, 2000. 17

[Lucchese 2006] L Lucchese. *Closed-form pose estimation from metric rectification of coplanar points.* IEE Proceedings on Vision, Image and Signal Processing, vol. 153, no. 3, pages 364 – 378, 2006. 18

[Milgram & Kishino 1994] Paul Milgram and Fumio Kishino. *A Taxonomy of Mixed Reality Visual Displays.* IEICE Transactions on Information Systems, vol. E77-D, no. 12, December 1994. 12

[Neubert *et al.* 2008] J. Neubert, J. Pretlove and T. Drummond. *Semi-autonomous generation of appearance-based edge models from image sequences.* In Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on, pages 79–89. IEEE, 2008. 21

[Neumann & Cho 1996] Ulrich Neumann and Youngkwan Cho. *A Self-Tracking Augmented Reality System.* In Proceedings of ACM Virtual Reality Software and Technology, pages 109–115, 1996. 16

[Neumann & You 2002] U. Neumann and S. You. *Natural feature tracking for augmented reality.* Multimedia, IEEE Transactions on, vol. 1, no. 1, pages 53–64, 2002. 21

[Newcombe & Davison 2010] R.A. Newcombe and A.J. Davison. *Live dense reconstruction with a single moving camera.* In Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, pages 1498–1505. IEEE, 2010. x, 21, 54, 132

[Nistér *et al.* 2004] David Nistér, Oleg Naroditsky and James Bergen. *Visual Odometry.* In Proc. ofăIEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2004), volume 1, pages ă652–659, 2004. 17

[Ohshima *et al.* 2003] T. Ohshima, T. Kuroki, H. Yamamoto and H. Tamura. *A Mixed Reality System with Visual and Tangible Interaction Capability.* In Proceedings of the Second IEEE and ACM International Symposium on Mixed and Augmented Reality, pages 284–285, 2003. 34

[Ong *et al.* 2008] S. K. Ong, M. L. Yuan and A. Y. C. Nee. *Augmented reality applications in manufacturing: a survey.* International Journal of Production Research, vol. 46, no. 10, page 2707Ű2742, 2008. 30

[Ortega & Coquillart 2005] M. Ortega and S. Coquillart. *Prop-Based Haptic Interaction with Co-Location and Immersion: An Automotive Application.* Workshop HAVE, 2005. 23, 26

[Otaduy & Lin 2008] M. A. Otaduy and M. C. Lin. *Introduction to Haptics Rendering Algorithms.* In M. C. Lin and M. A. Otaduy, editeurs, Haptic Rendering: Foundations, Algorithms and Applications, chapitre 8. AK Peters, 2008. 24

[Otaduy *et al.* 2009] Miguel A. Otaduy, Rasmus Tamstorf, Denis Steinemann and Markus Gross. *Implicit Contact Handling for Deformable Objects.* Computer Graphics Forum (Proc. of Eurographics), vol. 28, no. 2, apr 2009. 25, 106

[Ozuysal *et al.* 2007] M. Ozuysal, P. Fua and V. Lepetit. *Fast keypoint recognition in ten lines of code.* In Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on, pages 1–8. IEEE, 2007. 21

[Ozuysal *et al.* 2010] M. Ozuysal, M. Calonder, V. Lepetit and P. Fua. *Fast keypoint recognition using random ferns.* IEEE transactions on pattern analysis and machine intelligence, vol. 32, no. 3, pages 448 –461, 2010. 21

[Pahl *et al.* 1984] G. Pahl, W. Beitz and K. Wallace. Engineering design. Design Council, 1984. 6, 131

[Park *et al.* 1999] Jun Park, Bolan Jiang and Ulrich Neumann. *Vision-Based Pose Computation: Robust and Accurate Augmented Reality Tracking.* In IWAR '99: Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality, pages 3–12, Washington, DC, USA, 1999. IEEE Computer Society. 16

[Park *et al.* 2008] H. Park, J.S. Son and K.H. Lee. *Design evaluation of digital consumer products using virtual reality-based functional behaviour simulationTan-*

gible augmented prototyping of digital handheld products. Journal of Engineering Design, vol. 19, pages 359–375, 2008. 28, 30, 32, 62

[Park *et al.* 2009] H. Park, H.C. Moon and J.Y. Lee. *Tangible augmented prototyping of digital handheld products.* Computers in Industry, vol. 60, pages 114–125, 2009. vii, 28, 30, 33, 62, 65, 68

[Phung *et al.* 2005] S.L. Phung, A. Bouzerdoum and D. Chai. *Skin segmentation using color pixel classification: Analysis and comparison.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 1, pages 148–154, 2005. 54

[Quan & Lan 1999] Long Quan and Zhongdan Lan. *Linear N-point camera pose determination.* IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 21, no. 8, pages 774–778, 1999. 17

[Regenbrecht *et al.* 2005] H. Regenbrecht, G. Baratoff and W. Wilke. *Augmented reality projects in the automotive and aerospace industries.* IEEE Computer Graphics and Applications, vol. 25, no. 6, pages 48–56, 2005. 30

[Rolland *et al.* 2001] Jannick P. Rolland, Larry D. Davis and Yohan Baillot. *A survey of tracking technology for virtual environments.* In Barfield and Caudell, editeurs, Fundamentals of Wearable Computers and Augmented Reality, pages 67–112. L. Erlbaum Associates Inc., Mahwah, NJ, USA, 2001. 16

[Santos *et al.* 2006] Pedro Santos, Andre Stork, Alexandre Buaes and Joaquim Jorge. *Innovative geometric pose reconstruction for marker-based single camera tracking.* In VRCIA '06: Proceedings of the 2006 ACM international conference on Virtual reality continuum and its applications, pages 237–244, New York, NY, USA, 2006. ACM. 17

[Santos *et al.* 2007] Pedro Santos, André Stork, Thomas Gierlinger, Alain Pagani, Céline Paloc, Iñigo Barandarian, Giuseppe Conti, Raffaele de Amicis, Martin Witzel, Oliver Machui, José Jiménez, Bruno Araujo, Joaquim Jorge and

Georg Bodammer. *IMPROVE: An innovative application for collaborative mobile mixed reality design review.* International Journal on Interactive Design and Manufacturing, vol. 1, pages 115–126, 2007. 10.1007/s12008-007-0010-8. 30

[Schmidt *et al.* 2002] Jochen Schmidt, Heinrich Niemann and Sebastian Vogt. *Dense Disparity Maps in Real-Time with an Application to . . .* In In WACV 02, pages 225–230, 2002. 22

[Schweighofer & Pinz 2005] G. Schweighofer and A. Pinz. *Robust Pose Estimation from a Planar Target.* IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 28, no. 12, pages 2024 – 2030, 2005. 17

[Sekuler & Palmer 1992] A. B. Sekuler and S. E. Palmer. *Perception of Partly Occluded Objects:A Microgenetic Analysis.* J. Exp. Psychol. Gen., vol. 121, page 95Ű111, 1992. 21, 105

[Shin *et al.* 2002] M.C. Shin, K.I. Chang and L.V. Tsap. *Does Colorspace Transformation Make Any Difference on Skin Detection?* In Proceedings of IEEE Workshop on Applications of Computer Vision, pages 275–279, 2002. 54

[Shum *et al.* 2007] H.Y. Shum, S.C. Chan and S.B. Kang. Image-based rendering. Springer-Verlag New York Inc, 2007. 48

[Sinha *et al.* 2000] Rajarishi Sinha, Christiaan J. J. Paredis and Pradeep K. Khosla. *Integration of Mechanical CAD and Behavioral Modeling.* In BMAS '00: Proceedings of the 2000 IEEE/ACM international workshop on Behavioral modeling and simulation, 2000. 29

[Sànchez *et al.* 2005] J. Sànchez, F. Esquembre, C. Martìn, S. Dormido, S. Dormido-Canto, R.D. Canto, R. Pastor and A. Urquìa. *Easy Java Simulations: an Open-Source Tool to Develop Interactive Virtual Laboratories Using MATLAB/Simulink*.* International Journal of Engineering Education, vol. 21, no. 5, pages 798–813, 2005. 29

[Spence *et al.* 2008] C. Spence, F. Pavani, A. Maravita and N. P. Holmes. *Multi-Sensory Interactions.* In M. C. Lin and M. A. Otaduy, editeurs, Haptic Rendering: Foundations, Algorithms and Applications, chapitre 2. AK Peters, 2008. 25

[Stevenson *et al.* 1999] D. Stevenson, K. Smith, J. Mclaughlin, C. Gunn, J. Veldkamp and M. Dixon. *Haptic workbench: A multisensory virtual environment.* Proc. of SPIE, 1999. 25

[Summers *et al.* 1999] Valerie A. Summers, Kellogg S. Booth, Tom Calvert, Evan Graham and Christine L. MacKenzie. *Calibration for augmented reality experimental testbeds.* In I3D '99: Proceedings of the 1999 symposium on Interactive 3D graphics, pages 155–162, New York, NY, USA, 1999. ACM. 110

[Sutherland 1968] Ivan E. Sutherland. *A head-mounted three dimensional display.* In AFIPS '68 (Fall, part I): Proceedings of the December 9-11, 1968, fall joint computer conference, part I, pages 757–764, New York, NY, USA, 1968. ACM. vii, 13

[Tarrin *et al.* 2003] N. Tarrin, S. Coquillart, S. Hasegawa, L. Bouguila and M. Sato. *The Stringed Haptic Workbench: A New Haptic Workbench Solution.* Proc. of Eurographics, 2003. 25, 26, 104

[Ulrich *et al.* 1995] K.T. Ulrich, S.D. Eppinger*et al.* Product design and development, volume 384. McGraw-Hill New York, 1995. 6, 131

[Vallino & Brown 2002] J. Vallino and C. Brown. *Haptics in augmented reality.* In Multimedia Computing and Systems, 1999. IEEE International Conference on, volume 1, pages 195–200. IEEE, 2002. 110

[Ventura & Hollerer 2008] Jonathan Ventura and Tobias Hollerer. *Depth compositing for augmented reality.* In SIGGRAPH '08: ACM SIGGRAPH 2008 posters, New York, NY, USA, 2008. ACM. 22, 67

[Verlinden & Horvath 2006] J. Verlinden and I. Horvath. *Framework for testing and validating Interactive Augmented Prototyping as a Design Means in Industrial Practice*. In Proceedings of Virtual Concept 2006, 2006. 30, 31, 127

[Verlinden & Horvath 2007] J. Verlinden and I. Horvath. *A Critical Systems Position on Augmented Prototyping Systems for Industrial Design*. In Proceedings of the International Design Engineering Technical Conferences, ASME-CIE/DETC, 2007. 30, 31, 127

[Verlinden & Horvath 2009] J. Verlinden and I. Horvath. *Analyzing opportunities for using interactive augmented prototyping in design practice*. Artificial Intelligence for Engineering Design, Analysis and Manufacturing, vol. 23, pages 289–303, 2009. 30, 31, 127

[Verlinden *et al.* 2003] J.C. Verlinden, A. De Smit, AWJ Peeters and MH Van Gelderen. *Development of a flexible augmented prototyping system*. Journal of WSCG, vol. 11, no. 3, pages 496–503, 2003. vii, 31

[Vezhnevets *et al.* 2003] V. Vezhnevets, V. Sazonov and A Andreeva. *A Survey on Pixel-Based Skin Color Detection Techniques*. In Proceedings of Graphicon 2003, Moscow, Russia, pages 85–92, 2003. 54

[Virzi 1992] R. A. Virzi. *Refining the test phase of usability evaluation: How many subjects is enough?* Human Factors, vol. 34, pages 457–468, 1992. 97

[Wagner & Schmalstieg 2007] Daniel Wagner and Dieter Schmalstieg. *ARToolKitPlus for Pose Tracking on Mobile Devices*, 2007. 19, 65

[Wagner *et al.* 2009] D. Wagner, G. Reitmayr, A. Mulloni, T. Drummond and D. Schmalstieg. *Real Time Detection and Tracking for Augmented Reality on Mobile Phones*. IEEE Transactions on Visualization and Computer Graphics, 2009. 21

[Weir & Colgate 2008] D. W Weir and J.E. Colgate. *Stability of Haptic Displays.* In M. C. Lin and M. A. Otaduy, editeurs, Haptic Rendering: Foundations, Algorithms and Applications, chapitre 7. AK Peters, 2008. 24

[Yang & Waibel 1996] J. Yang and A. Waibel. *A real-time face tracker.* In Proceedings of the third IEEE Workshop on Applications of Computer Vision, WACV96, 1996. 55

[Yuan *et al.* 2006] ML Yuan, SK Ong and AYC Nee. *Registration using natural features for augmented reality systems.* IEEE Transactions on Visualization and Computer Graphics, pages 569–580, 2006. 21

[Zhang *et al.* 2002] X. Zhang, S. Fronz and N. Navab. *Visual marker detection and decoding in AR systems: A comparative study.* In Proceedings of the 1st International Symposium on Mixed and Augmented Reality, page 97. IEEE Computer Society, 2002. 21

[Zhang 2000] Z. Zhang. *A flexible new technique for Camera Calibration.* IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 22, no. 11, 2000. 17, 36, 40

[Zhou *et al.* 2008] Feng Zhou, Henry Been-Lirn Duh and Mark Billinghurst. *Trends in augmented reality tracking, interaction and display: A review of ten years of ISMAR.* Mixed and Augmented Reality, IEEE / ACM International Symposium on, vol. 0, pages 193–202, 2008. 12, 14, 16

[Zokai *et al.* 2003] S. Zokai, J. Esteve, Y. Genc and N. Navab. *Multiview Paraperspective Projection Model for Diminished Reality.* Proc. of ISMAR, 2003. 26