UNIVERSITÀ DELLA CALABRIA

Facoltà di Ingegneria

Dipartimento di Meccanica

**DOTTORATO DI RICERCA IN INGEGNERIA MECCANICA**

XX Ciclo

S.S.D. ING-IND/15 DISEGNO E METODI DELL'INGEGNERIA INDUSTRIALE

# Innovative methodologies and tools for the interaction with Virtual Prototypes

Coordinatore
Prof. Maria Laura Luchi

Dottorando
Ing. Francesco Caruso

Tutor
Prof. Maurizio Muzzupappa

Anno Accademico 2006 - 2007

# Summary

*Motivations* ........ *4*

*Outline* ........ *7*

*Chapter 1. Background* ........ *9*

**1.1. The role of the virtual prototype in the product development process** ........ **9**

**1.2. Virtual prototypes employments in Virtual reality** ........ **11**

**1.3. Virtual prototypes employment in Augmented Reality** ........ **13**

*Chapter 2. Visualization of Engineering Data in Augmented Reality* ........ *22*

**2.1. State of the Art of the augmented reality application for scientific visualization** **23**

**2.2. Methodology** ........ **24**

**2.3. Hardware set-up** ........ **25**

**2.4. The VTK4AR framework** ........ **27**

**2.5. Implementation** ........ **28**

**2.6. New VTK Classes for AR** ........ **31**

**2.7. Case Studies** ........ **32**

**2.8. Conclusion and Future Developments** ........ **36**

*Chapter 3. An Integrated Environment For The Validation In Virtual Reality Of The Dynamical Behavior Of Functional Digital Mock-Ups* ........ *38*

**3.1. Implementation** ........ **40**

**3.2. Division MockUp** ........ **41**

**3.3. The SIMLib library** ........ **42**

**3.4. The Simulink S-Function for the communication** ........ **44**

**3.5. The Divison plug-in** ........ **46**

**3.6. Test Case** ........ **46**

**3.7. Summary** ........ **49**

*Chapter 4. Multi-domain co-simulation using Multi-Body and Electronic Control System solvers51*

**4.1. State of the Art** ........ **54**

**4.2. Methodology** ........ **55**

**4.3. Benefits and Drawbacks** ........ **58**

**4.4. Case study** ........ **59**

**4.5. Conclusions** ........ **67**

*Chapter 5. An experimental environment for the runtime communication among different solvers and visualization modules* ........ *68*

**5.1. Introduction** ........ **68**

**5.2. Implementation** ........ **73**

**5.3. The IFXOpenSG application** ........ **74**

**5.4. The solvers** ........ **75**
**5.5. The VisSim application** ........ **76**
**5.6. Test Case** ........ **78**
***Chapter 6. References*** ........ ***80***

## Motivations

After the Scientific Revolution, scientists have always tried to integrate and correct their knowledge by observing and investigating real phenomena. The scientific method in fact, is properly based on the observation and the experimentation. Due to the recent technologies, the scientific method has been revised, and a wide range of experimentations are now performed using computers. Nowadays in fact, using the computers, it is possible to virtually recreate an experiment, i.e. to reproduce the experiment in a computer synthesized environment. As it is easy to understand, this possibility has enormous advantages with respect to the standard experimentation methods. Therefore, we assisted to the birth and widespread diffusion of software products for virtual simulations, which actually represent one of the most powerful tools in the hands of the scientists.

However, as the scientific method teach, the observation of the phenomena is crucial for the enhancement of the knowledge. Therefore, the results of the simulations must be processed and visualized in the most effective way to get a better comprehension of the phenomena. The large amount of data produced by the numerical simulation must be interpreted and presented in the most effective way. For this reason, part of this PhD work has been spent in the study of innovative techniques of visualization for the numerical simulations results.

During last years, a process of high specialization of the knowledge occurred. The specialization of knowledge lead to the specialization of the “tools” needed by the scientists. In particular, as far as simulation software concerns, it became highly specialized too. In fact we can find many software for simulations in a specific physical domain, e.g. CFD simulators, FEM simulators, and so on.

On the other side, the complexity of the industrial products is rapidly increasing, and more and more often, several physical domains are involved in the development of a single product. This lead to the birth of the so called cross-domain engineering, in which several engineers of different fields are involved and engaged in the development of a complex product.

In order to apply the scientific method to the current complex projects it is necessary "to join" the results of each simulator to create a cross-domain simulation environment. In this way scientists will be closer to the original scientific method, because in the real experiments all the domains are naturally involved.

Obviously, as a mechanical engineer, the research work has been focused and fitted for the mechanical engineering tasks. The mechanical engineer uses the computers in order to simulate and observe the behavior of its work before its physical realization. This target is achieved by the use of the *Virtual Prototype* (VP). Therefore, the role of the virtual prototype has been central throughout this research. As it is easy to understand, in the engineering field, the experiments regard the prototype of the under development product. The object of the experimentation is therefore the virtual prototype. During the product development process in fact, the virtual prototype is incrementally enriched by the results provided by the numerical simulations. Virtual prototype should include the geometry, the mathematical models to describe the physical properties of the product and, if presents, the behavioral model provided by the control logic. The final goal should be a complete and fully functional virtual prototype, on which it should be possible to perform a cross-domain validation phase before the launch of the product on the market. At present instead, each of the aspect of the experimentation is almost always performed in a specific simulation environment, i.e. the dynamic validation is performed with a multi body solver; the fluid dynamic evaluation is performed within a CFD simulator and so on. It is possible to perform a cross-domain simulation using dedicated multi-physics software like COMSOL/Multi-Physics, or trying to use built-in software interfaces between two different commercial software, e.g. Matlab/Adams, Simulink/Visual Nastran. However, both solutions do not allow engineers to employ their own solver or to choose the best combination among the available simulation software. A framework, which allows one to use different solvers as modular blocks to build an integrated cross-domain simulation environment, could be, obviously, a better and more flexible solution. Further, in order to achieve an effective comprehension of numerical data, it should be possible to use high-end visualization techniques, like Virtual or Augmented reality and powerful post-processors. One of the targets of this PhD work in fact was to improve the effectiveness of the numerical simulations. Therefore, a co-simulation framework for multi-domain simulations, which allows one to inter-change solvers and visualization frameworks has been developed. The framework allows one to perform the visualization in a Virtual Reality environment in order to improve the interaction of the user with the experiment. The aim of this framework is therefore to consider all the physical domains and to reproduce the experiments in the most natural way for the user.

The whole research is addressed in the creation of a new environment to support designers. This environment mixes different kind of information (coming from the real world, the numerical simulations and so on) to provide

## Outline

In the first chapter, a background of the current technologies employed throughout the research, i.e. Virtual Prototyping, Virtual Reality and Augmented Reality is presented. Afterwards, the research work is presented. The research work can be divided into two main topics:

Advanced interaction techniques with numerical simulation

Multi-Domain simulation

Chapter 2 and chapter 3 regards the first research topic.

In the second chapter an Augmented Reality framework for scientific visualization is presented [1][2][3][4]. The target of this work has been the capability of visualize data from numerical simulations superimposed to the real object that they refer to. As previously said, the visualization is crucial for the understanding and the comprehension of the real phenomena. Therefore, in this chapter is presented an investigation on the capabilities provided by the augmented reality in scientific visualization tasks. A methodology that provides the tools and defines the steps needed to obtain the visualization in Augmented R eality of industrial engineering data has been developed.

In the third chapter, a possible improvement of the user interaction with numerical simulations is presented. In particular, an integrated environment for the validation in virtual reality of the dynamical behavior of functional digital mock-ups is described [5][8]. In this environment it is possible to achieve multi-body simulation in Virtual Reality. The multi-body solver animates the three-dimensional model visualized in a virtual reality environment, and the user can interact with the simulation using joysticks.

Chapter 4 and chapter 5 regard the second research topic, i.e. the multi-domain simulation.

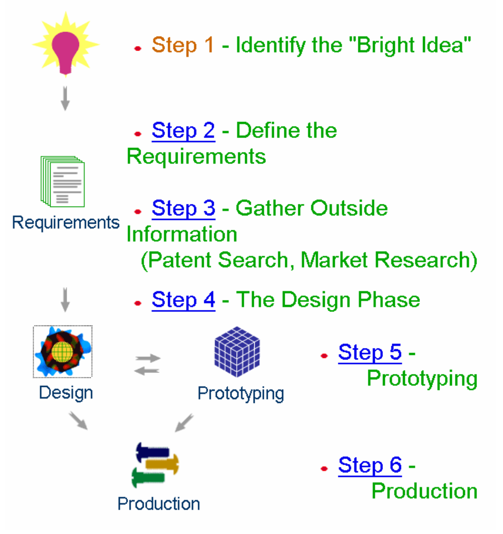
The fourth chapter describes a co-simulation environment in which multi-body solver and electronic control solver are used to simulate the behavior of a mechatronic product. With the help of the co-simulation, a new design methodology for complex product has been developed. This methodology, based on the top-down approach, starts from the conceptual model, and uses the simulation to validate each step of the design process. At the end of the design process it is possible to perform the validation of both the aesthetic and the functional part of an industrial product within an immersive virtual environment.

The fifth chapter describes an experimental environment for runtime communication among different solvers and visualization modules [7]. The target of this work is a visualization and simulation environment which allows the runtime communication among the visualization framework and different solvers, to obtain a multi-domain simulation and visualization environment. This work provides great flexibility and modularity for solvers and visualization environments, since it uses an external scheduler for triggering the different applications. It is a new approach compared to the standard simulation techniques.

# Chapter 1. Background

## 1.1. *The role of the virtual prototype in the product development process*

Today's product development is under a highly challenging environment, and companies are under increasing pressure to sustain their competitive advantages by reducing product development time and cost while maintaining a high level of quality. Normally, the Product development process is made up by different phases, resumed in the picture below.



Step 1 - Identify the "Bright Idea"
Step 2 - Define the Requirements
Requirements
Step 3 - Gather Outside Information (Patent Search, Market Research)
Step 4 - The Design Phase
Design
Prototyping
Step 5 - Prototyping
Production
Step 6 - Production

**Figure 1: Product Development Process**

Prototyping is the design verification phase of Product Development used to demonstrate or prove aspects of a design. Normally, prototyping is simply taking the design from the virtual and imaginary realm to the physical world. There are many levels of Prototypes. Some are simple duct-tape and bailing wire to visualize how something might work; Others are highly polished but fragile representations for show and tell; still others are functional representations that work. The kind of prototype used should fit the needs of the project especially since there is often a significant cost involved.

Typical prototyping methods include clay (or other) mock-up, fabrication and machining, and rapid prototyping. Mock-ups are typically done very early in the design for visualization, feel, and to allow adjustments or fiddling with shape and size. Fabricated prototypes are typically functional versions that may or may not look like the final product but give the opportunity to test function and prove something works.

The *virtual prototyping* is a new way to perform the validation phase. It aims to reduce (or avoid) the realization of the physical prototypes by creating a digital model of the product. The idea is to simulate and forecast the behavior of the real product before its realization by using a numerical model. All the tests will be therefore performed on the *Virtual Prototype* (VP) and not on a real one. In other words, the Virtual Prototype is a numerical model containing most of the information of the product needed for its validation and its evaluation.

The term "virtual prototype" (VP) has been widely used by various researchers and engineers. It is important to formally describe what constitutes a VP and, subsequently, provide a background on various types of virtual prototyping. Practically, it is a computer-based simulation of a system or subsystem with a degree of functional realism comparable to a physical prototype. In general, a VP is a 3-dimensional virtual reality (VR) based model which seeks to "mimic" a target (or "real-world") object, system, or environment. This model can be an object, system (or collection of objects), or a complex environment. The process of creating and using a VP in various applications can be referred to as virtual prototyping. A VP, has certain characteristics which differentiates it from other models or prototypes. These include:

1. Appearance characteristics: VPs must possess accurate geometry, topology, and appearance, reflecting characteristics of the target part, object, system, or environment
2. Simulation characteristics: VPs should be capable of simulating engineering or science-based characteristics, including behavior with real-time responses
3. Representation criteria: AVP is a digital or computer-based representation

4. Interface criteria: VPs must possess the ability to interface VR technology and graphics, including supporting semi-immersive or immersive applications

## 1.2. Virtual prototypes employments in Virtual reality

The term Virtual Reality (VR) was coined in the 1989 by Jaron Lanier, acknowledged as one of the fathers of the VR. Lanier in 1984 founded the VPL Inc. (Virtual Programming Language), which was one of first companies involved in Virtual Reality, producing first dedicated devices: the Eyephone helmet for the stereoscopic vision, the Dataglove gloves and the Datasuit suit. For several years research tried to find which were the best tools and instruments to provide the user the feeling of “immersion” in the virtual world. At the beginning nobody knew which were the possible employments of these new technologies, but it was clear that VR capabilities were enormous. The interest for the devices invented and developed by Lanier was very high, in fact also the Matsushita company invested in this field. However, times were not ripe to get concrete results to grant an adequate remunerations for the investments. The VPL in fact, was in loss and it was absorbed by Thomson. The excitement
Virtual Reality (VR) technologies are widely employed in the Product Development Process (PDP) in several industrial sectors like automotive, aerospace, defence, bioengineering, etc. [9][10]. Visualization has been the first application area in which VR has been appreciated, since the immersive visualization, employing large display and stereoscopic vision, allows the user to better understand the meaning of the data visualised or the aesthetics of new products. In the automotive field, for example, VR has been employed for the visualization, in real scale, of the virtual prototype of new cars [11] and for the visualization of complex data-sets like the results of the Finite Elements Method (FEM) or Computational Fluid Dynamics (CFD) analysis[12]. The visualization of data-sets generated by engineering simulation software does not present particular integration problems because the software for immersive visualization is usually able to read a wide range of file formats.

Although visualization has played an important role in the diffusion of VR inside industries, it does not exploit the natural interaction capabilities of VR. In a VR environment, in fact, it is possible to interact in a more natural way with the digital mock-up. As a matter of fact, thanks to special devices, the user finds him/herself in a simulated environment in which, through stereoscopic vision, he/she can perceive the depth of the scene, having -at the same time- the possibility of interacting with the virtual product by using his/her hands. Such interaction is possible thanks to the use of special gloves, which have sensors that are able to detect the bending of one's fingers. By supplying the gloves with tracking sensors, that are able to reveal position and orientation, it is possible to reconstruct the movements of the user's hands in a virtual environment; this way the user may carry out several types of simulations connected to the interaction with the product. The main advantages offered by VR technologies are more evident in all the applications in which the interaction with the virtual product is important. A widespread example in this sense is the virtual assembly that aims to simulate the entire operative context in which the human operator works, during the assembly/disassembly task. In this way, a designer can directly verify potential difficulties in component reachability, in posture and visibility [13][14][16][17][18][19][20][21].

Other examples in which the interaction capabilities of VR have been successfully employed in the product development process are Virtual Cabling[22][23] tools, ergonomics evaluation [24] and simulation of the product behaviour both for functional [25] and usability tests[26]. Most of the reported references show that VR is an effective tool to support the design process of industrial products, but it is not clear if the routine use of these tools is really efficient. In fact, the lack of integration between VR and other tools, employed in the design process (usually CAx software) requires a lot of manual work to prepare the virtual environment or to post-process the results obtained in VR. This problem is one of the main obstacles to a wider diffusion of the VR technologies in the PDP and, for this reason, it is important to create an efficient link between CAx and VR.

Many CAD vendors have approached this problem because they are widening their offer in IT solutions acquiring or developing the technologies needed to complete their product range with VR tools like visualization software for DMU or CAE analysis, in order to satisfy the manifold requirements of the product development process. The interoperability between CAD software and VR system has been gradually realised thanks to the development of common file formats for product data exchange and sharing. UGS [27] and Dassault Systemes [28], two of the foremost companies in the PLM systems market, have released some specific languages to support interoperability in the PLM system. UGS proposes the JT format, which is the one employed for the storage and visualization of DMU geometric data. UGS also proposes the PLM XML format, which is defined by a set of XML schemas and includes the representation of both the high-level metadata and the geometric representation of the product. Dassault proposes 3DXML, which is an open file format usable in all their software for product development. Both these XML based file formats offer the advantage of being customizable with user defined metadata that can be added to the standard schemas to enhance the model with all the information needed to perform a specific task.

This possibility can be very useful for VR application development. As a matter of fact, each specific VR tool requires different product data, and these can be stored using the previously mentioned XML based language. Besides, the availability of a powerful language for data storage does not resolve the integration problems at all, because it requires the development of software interfaces (in both the CAx and VR applications), able to read, write and manage the specific metadata stored in the file.

## *1.3. Virtual prototypes employment in Augmented Reality*

Augmented Reality (AR) is a variation of Virtual Environments (VE), or Virtual Reality as it is more commonly called. VE technologies completely immerse a user inside a synthetic environment. While immersed, the user cannot see the real world around him. In contrast, AR allows the user to see the real world, with virtual objects superimposed upon or composited with the real world. Therefore, AR supplements reality, rather than completely replacing it. Ideally, it would appear to the user that the virtual and real objects coexisted in the same space, similar to the effects achieved in the film "Who Framed Roger Rabbit?" shows an example of what this might look like. It shows a real desk with a real phone. Inside this room are also a virtual lamp and two virtual chairs. Note that the objects are combined in 3-D, so that the virtual lamp covers the real table, and the real table covers parts of the two virtual chairs. AR can be thought of as the "middle ground" between VE (completely synthetic) and telepresence (completely real) [29][30]

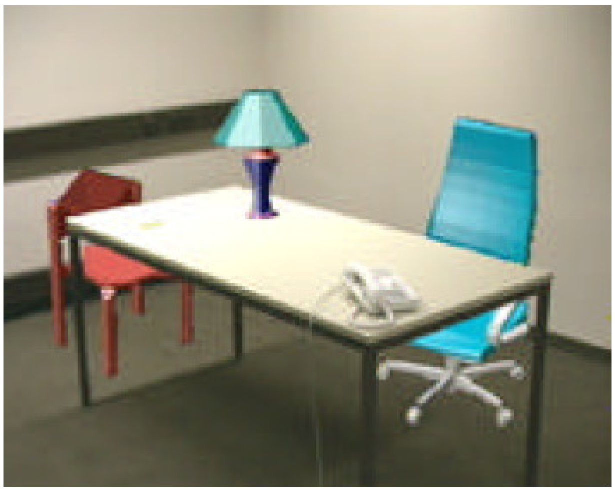


**Figure 2 Augmented Reality Scene**

An AR system supplements the real world with virtual (computer-generated) objects that appear to coexist in the same space as the real world. While many researchers broaden the definition of AR beyond this vision, we define an AR system to have the following properties:

- combines real and virtual objects in a real environment;
- runs interactively, and in real time; and
- registers (aligns) real and virtual objects with each other.

Certain AR applications also require removing real objects from the perceived environment, in addition to adding virtual objects. For example, an AR visualization of a building that stood at a certain location might remove the building that exists there today. Some researchers call the task of removing real objects mediated or diminished reality, but we consider it a subset of AR. Milgram defined a continuum of real-to-virtual environments, in which AR is one part of the general area of mixed reality (Figure 2). In both augmented virtuality, in which real objects are added to virtual ones, and virtual environments (or virtual reality), the surrounding environment is virtual, while in AR the surrounding environment is real. We focus on AR and don't cover augmented virtuality or virtual environments. The beginnings of AR, as we define it, date back to Sutherland's work in the 1960s, which used a seethrough HMD to present 3D graphics[31]. However, only over the past decade has there been enough work to refer to AR as a research field. In 1997, Azuma published a survey [32] that defined the field, described many problems, and summarized the developments up to that point. Since then, AR's growth and progress have been remarkable. In the late 1990s, several conferences on AR began, including the International Workshop and Symposium on Augmented Reality, the International Symposium on Mixed Reality, and the Designing Augmented Reality Environments workshop. Some well-funded organizations formed that focused on AR, notably the Mixed Reality Systems Lab in Japan and the Arvika consortium in Germany. A software toolkit (the ARToolkit) for rapidly building AR applications is now freely available on the internet.

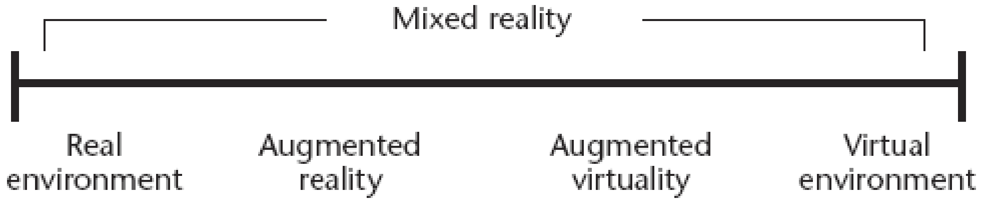


**Figure 3 Real to Virtual continuum**

Doctors could use Augmented Reality as a visualization and training aid for surgery. It may be possible to collect 3-D datasets of a patient in real time, using non-invasive sensors like Magnetic Resonance Imaging (MRI), Computed Tomography scans (CT), or ultrasound imaging. These datasets could then be rendered and combined in real time with a view of the real patient. In effect, this would give a doctor "X-ray vision" inside a patient. This would be very useful during minimally-invasive surgery, which reduces the trauma of an operation by using small incisions or no incisions at all. A problem with minimally-invasive techniques is that they reduce the doctor's ability to see inside the patient, making surgery more difficult. AR technology could provide an internal view without the need for larger incisions. AR might also be helpful for general medical visualization tasks in the surgical room. Surgeons can detect some features with the naked eye that they cannot see in MRI or CT scans, and vice-versa. AR would give surgeons access to both types of data simultaneously. This might also guide precision tasks, such as displaying where to drill a hole into the skull for brain surgery or where to perform a needle biopsy of a tiny tumor. The information from the non-invasive sensors would be directly displayed on the patient, showing exactly where to perform the operation. AR might also be useful for training purposes [33]. Virtual instructions could remind a novice surgeon of the required steps, without the need to look away from a patient to consult a manual. Virtual objects could also identify organs and specify locations to avoid disturbing [34]. Several projects are exploring this application area. At UNC Chapel Hill, a research group has conducted trial runs of scanning the womb of a pregnant woman with an ultrasound sensor, generating a 3-D representation of the fetus inside the womb and displaying that in a see-through HMD (Figure 3). The goal is to endow the doctor with the ability to see the moving, kicking fetus lying inside the womb, with the hope that this one day may become a "3-D stethoscope" [35][36]. More recent efforts have focused on a needle biopsy of a breast tumor. Figure 4 shows a mockup of a breast biopsy operation, where the virtual objects identify the location of the tumor and guide the needle to its target [37]. Other groups at the MIT AI Lab [38][39][40][41], General Electric [42], and elsewhere [43][44] [45] are investigating displaying MRI or CT data, directly registered onto the patient.

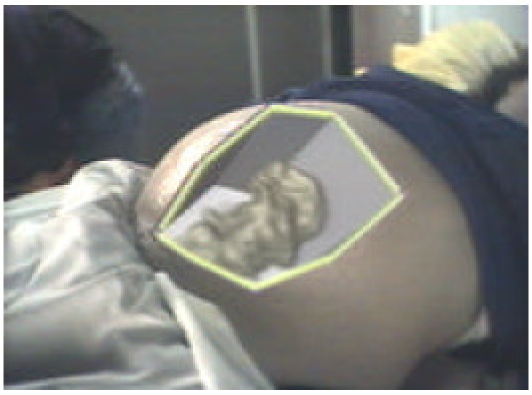


**Figure 4: Virtual fetus inside womb of pregnant patient. (Courtesy UNC Chapel
Hill Dept. of Computer Science)**

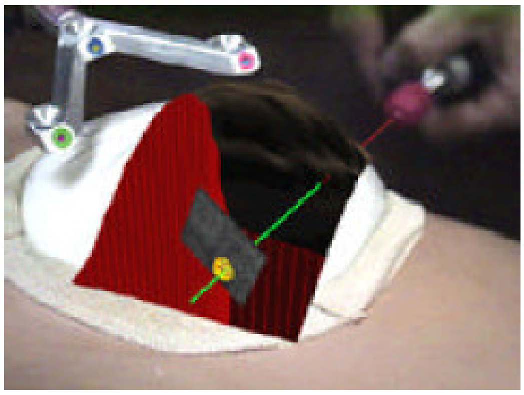


**Figure 5: Mockup of breast tumor biopsy. 3-D graphics guide needle insertion.
(Courtesy UNC Chapel Hill Dept. of Computer Science)**

Another category of Augmented Reality applications is the assembly, maintenance, and repair of complex machinery. Instructions might be easier to understand if they were available, not as manuals with text and pictures, but rather as 3-D drawings superimposed upon the actual equipment, showing step-by-step the tasks that need to be done and how to do them. These superimposed 3-D drawings can be animated, making the directions even more explicit. Several research projects have demonstrated prototypes in this area. Steve Feiner's group at Columbia built a laser printer maintenance application [46], shown in Figures 5 and 6. Figure 5 shows an external view, and Figure 6 shows the user's view, where the computer-generated wireframe is telling the user to remove the paper tray. A group at Boeing is developing AR technology to guide a technician in building a wiring harness that forms part of an airplane's electrical system. Storing these instructions in electronic form will save space and reduce costs. Currently, technicians use large physical layout boards to construct such harnesses, and Boeing requires several warehouses to store all these boards. Such space might be emptied for other use if this application proves successful [47][48][49]. Boeing is using a Technology Reinvestment Program (TRP) grant to investigate putting this technology onto the factory floor [50]. Figure 7 shows an external view of Adam Janin using a prototype AR system to build a wire bundle. Eventually, AR might be used for any complicated machinery, such as automobile engines [51].

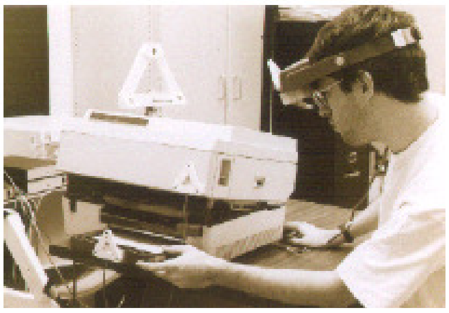


**Figure 6 External view of Columbia printer maintenance application. Note thatall objects must be tracked. (Courtesy Steve Feiner, Blair MacIntyre, and Dorée Seligmann, Columbia University.)**

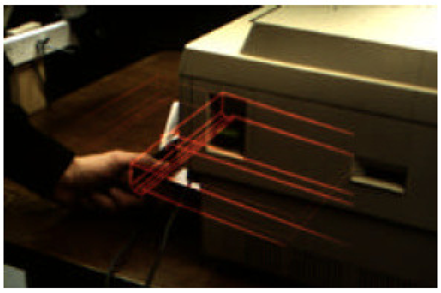


**Figure 7 Prototype laser printer maintenance application, displaying how to remove the paper tray. (Courtesy Steve Feiner, Blair MacIntyre, and Dorée Seligmann, Columbia University.)**

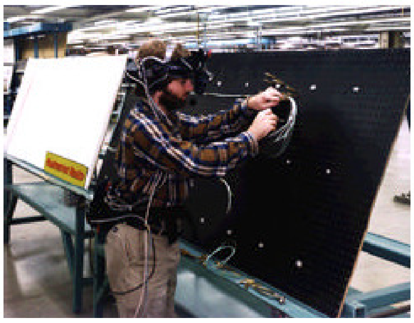


**Figure 8 Demonstration of Boeing's prototype wire bundle assembly (Courtesy David Mizell, Boeing)**

AR could be used to annotate objects and environments with public or private information. Applications using public information assume the availability of public databases to draw upon. For example, a hand-held display could provide information about the contents of library shelves as the user walks around the library [52][53][54]. At the European Computer-Industry Research Centre (ECRC), a user can point at parts of an engine model and the AR system displays the name of the part that is being pointed at [55]. Figure 9 shows this, where the user points at the exhaust manifold on an engine model and the label "exhaust manifold" appears.

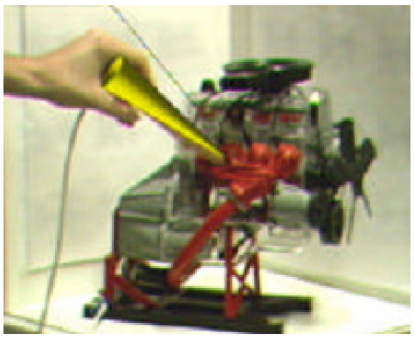


**Figure 9 Engine model part labels appear as user points at them. (Courtesy ECRC)**

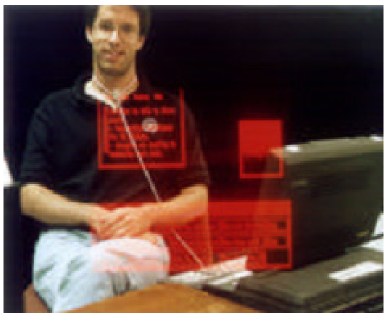


**Figure 10 Windows displayed on top of specific real-world objects. (Courtesy Steve Feiner, Blair MacIntyre, Marcus Haupt, and Eliot Solomon, Columbia University.)**

Alternately, these annotations might be private notes attached to specific objects. Researchers at Columbia demonstrated this with the notion of attaching windows from a standard user interface onto specific locations in the world, or attached to specific objects as reminders [56]. Figure 10 shows a window superimposed as a label upon a student. He wears a tracking device, so the computer knows his location. As the student moves around, the label follows his location, providing the AR user with a reminder of what he needs to talk to the student about.

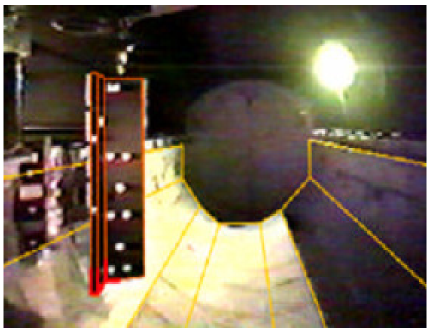


**Figure 11 Virtual lines help display geometry of shuttle bay, as seen in orbit.(Courtesy David Drascic and Paul Milgram, U. Toronto.)**

AR might aid general visualization tasks as well. An architect with a see-through HMD might be able to look out a window and see how a proposed new skyscraper would change her view. If a database containing information about a building's structure was available, AR might give architects "X-ray vision" inside a building, showing where the pipes, electric lines, and structural supports are inside the walls [56]. Researchers at the University of Toronto have built a system called Augmented Reality through Graphic Overlays on Stereovideo (ARGOS) [57], which among other things is used to make images easier to understand during difficult viewing conditions [58]. Figure 11 shows wireframe lines drawn on top of a space shuttle bay interior, while in orbit. The lines make it easier to see the geometry of the shuttle bay. Similarly, virtual lines and objects could aid navigation and scene understanding during poor visibility conditions, such as underwater or in fog.

# Chapter 2. Visualization of Engineering Data in Augmented Reality

During this PhD work a research about the capabilities of Augmented Reality in the field of industrial engineering has been carried out. The first idea was try to investigate the employment of AR techniques to improve the comprehension of numerical data in collaborative tasks. In order to validate this idea, an innovative application of Augmented Reality techniques in the field of industrial engineering has been developed. With the help of this application the user can explore data from numerical simulations or the results of measurements and experiments, superimposed to the real object that they refer to. The user observes the object through a tablet PC, used as a video see-through handheld display. Data are visualized superimposed to the real object that represents a spatial reference which the user can refer to, so the exploration is more natural compared to a traditional visualization software. Moreover a new framework, called VTK4AR, has been developed. This framework provides a set of useful software classes for the rapid development of AR applications for scientific visualization. VTK4AR is built on top of VTK (an open source API for scientific visualization), so it will be possible to employ a wide range of visualization techniques in many application fields, and moreover, it is possible to interactively manipulate data-sets in order to achieve a more effective way of visualization.

The AR visualization improves the collaboration among designers and technicians, since their analyses tasks are often correlated. Furthermore, it improves the handling and availability of the data itself and also simplify the explanation of complex data-sets to non-expert people. A methodology that provides the tools and defines the steps needed to obtain the visualization in AR of industrial engineering data has been developed. The methodology employs different engineering technologies, an AR system based on a tablet PC as a displaying device and a new all-in-one framework, called VTK4AR, that allows developers to easily and rapidly develop AR applications for scientific visualization. In accord with what is above mentioned, this work presents two main innovations:

- for the first time AR techniques are employed in data visualization in an industrial design context.
- A new AR software framework for the rapid development of AR applications for scientific visualization has been developed.

## *2.1. State of the Art of the augmented reality application for scientific visualization*

As previously mentioned, AR applications have been implemented in many application domains. Other applications of scientific visualization in AR have been developed upon Studierstube, a framework for collaborative AR [59][60][61][62]. These applications allow users to display and examine volumetric data in a collaborative AR environment, and to highlight or explore slices of the volume by manipulating an optical marker as a cutting plane interaction device. In these applications AR is a novel interface for scientific visualization for improving collaborations among the users, but it has not been employed to enrich real objects with useful information related to the objects itself. From the papers mentioned above, it can be inferred that AR visualization improves comprehension and interaction of the visualized data. However, none of the authors above employs this technique in the industrial engineering field, in which more and more effective ways to analyze data are required. From this perspective, in fact, the AR visualization improves the collaboration among designers and technicians, since their analyses tasks are often correlated. Furthermore, it improves the handling and availability of the data itself and also the need to explain complex data-sets to non-expert people is simplified.

## 2.2. Methodology

The introduction of computer aided systems (CAx systems) has made the use of virtual prototypes indispensable, thus radically modifying the design process of industrial products. On the other hand, despite the fact that such systems are nowadays widespread, design activities still strongly depend on physical mock-ups, which are necessary to validate numerical models.
The need to use both physical and virtual prototypes during the design activity has supported the development of technologies able to easily generate physical prototypes through virtual ones and vice versa. In particular, the Reverse Engineering (RE) is nowadays the most common technique used in the engineering fields to generate virtual models from physical models, through a 3D scanner; whereas the Rapid Prototyping (RP) is the technique which allows an easy and completely automated way of generating physical prototypes through virtual models.
The methodology, proposed in the present study, integrates the above-mentioned technologies with augmented reality, for the fist time. It stands out because it has two starting points according to the kind of prototype (virtual or physical) designers initially have at their disposal.
The methodology in question requires the presence of both prototypes and, therefore, the first step one has to take is the generation of the missing prototype through the above-mentioned techniques. Secondly, in keeping with project requirements, one has to decide whether to use the virtual model to make simulations or analysis (eg. structural, fluid-dynamic, thermal, etc.) or the physical model to carry out experiments and measurements (eg. assessing mistakes made during the forming process, tension and fatigue tests, wind tunnel analyses, etc.). The third step is the post processing of data obtained, both through CAE analysis and experiments and measurements. Thanks to this operation one may determine, for example, the magnitudes visualized, or eliminate any noises from the dataset. The last step is the visualization of AR, which is made through an ad hoc software: VTK4AR. This software is supposed to create an augmented reality

environment which uses a special hardware set-up and video-tracking technology in order to be able to visualize the data-set directly superimposed on the real model. Moreover, VTK4AR requires the contemporaneous use of both the physical and the virtual prototype. In particular, VTK4AR uses virtual models to correctly visualize the real model and the data-set solving any occurring occlusion problems. Furthermore, VTK4AR uses the physical model to define the spatial reference on which the results of the simulation are visualized.

In figure 12 the various steps of the methodology proposed, are shown.

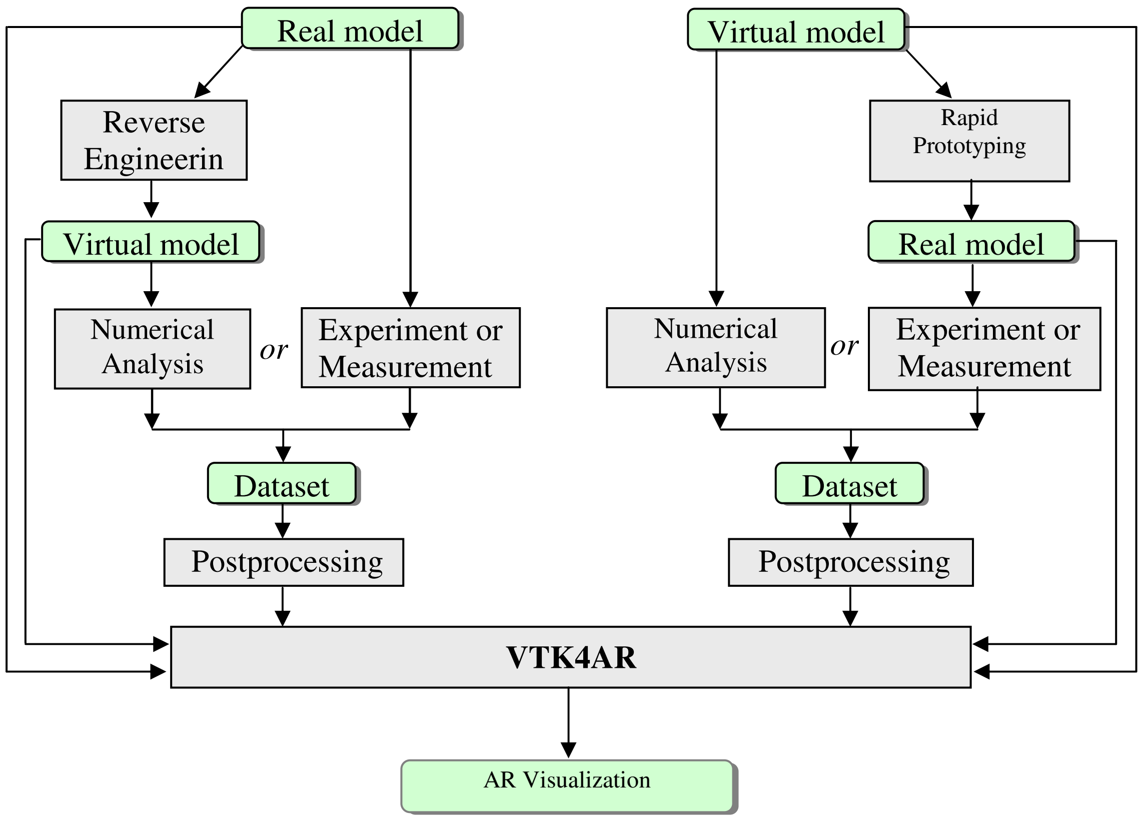


**Figure 12: Methodology Diagram**

## 2.3. *Hardware set-up*

An augmented reality system can be implemented using different hardware configurations. One of the most widespread solutions is the optical see-through Head Mounted Display (HMD). Another chance is the video see-through approach: one or two cameras can be employed to transform a normal HMD, a hand-held device [60] or a flat monitor [63], into a simulated see-through display.

Tablet PC and PDA has been widely employed in AR applications. In [65] the Tablet PC was employed to display detailed information about plant machines in an oil refinery (i.e. the inner parts of a pump), and, in the real in real project (www.contents4u.com), tablets and PDAs are employed in interior design applications to show virtual furniture in a real room.

A tablet PC has been employed as a display device because it offers some advantages compared to other possible configurations. Compared to the HMD, it is a low cost device, it is completely wireless and can be observed by two or three people at the same time, but, on the other hand, it cannot support stereoscopic vision. Compared to a handheld device, it offers a wider display and much better hardware performances, but it weighs more.

As described by Rekimoto [64], the tablet works like a magnifying glass, that can be used to perceive information about the objects that, otherwise, could not be seen. The tablet can be imagined as an augmenting window which augments the perception of the objects visualized through it. Navigation is then much more natural compared to a standard desktop environment. Furthermore, real objects on which the data are superimposed represent a spatial reference which the user can refer to. But the magnifying glass metaphor in industrial engineering data visualization has never been tested before.

The tablet PC is equipped with a video camera employed both to simulate the see-through functionality and to track the display position with a pattern recognition algorithm.

The hardware used in this study is a tablet PC with: Intel® Pentium® M processor 1.60 GHz, 1024MB PC2700 DDR333 SDRAM, Nvidia GeForce FX Go5200 32Mb graphics card, 12.1" Polysilicon SXGA+ Display, weight 2.0Kg. The webcam employed is a Creative LivePro. It has a VGA CCD sensor and a digital video capture speed of 30 fps.

The video camera has been mounted on the top of the tablet, so it is possible for the user to handle the tablet with one hand, while the other hand holds the pen used to interact with widgets displayed on the tablet monitor.

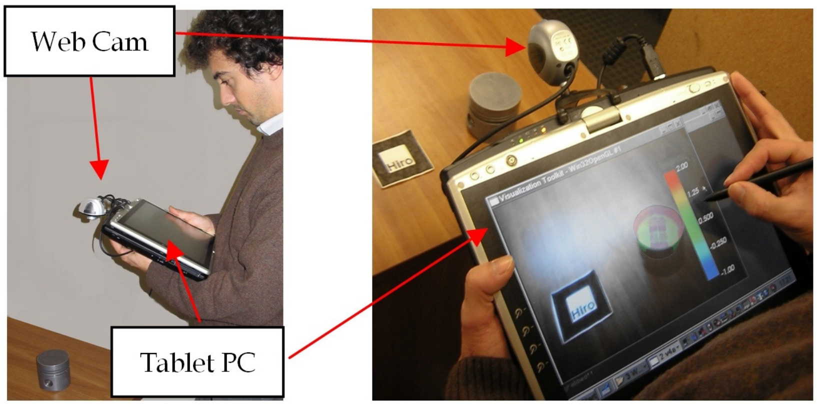


**Figure 13: Hardware set-up of vtk4ar**

The use of the tablet PC as a magnifying glass allows the user to immediately become familiar with the system, even if he/she does not have any experience in augmented reality. The exploration of the dataset is natural because the user only has to move the tablet where he/she would like to display data. In fact the magnifying glass metaphor gives the user the possibility to directly experience the 3D space, and this is the best way to understand the 3D space itself [66]. The real object on which data are superimposed, represents a spatial reference, which the user can refer to when interacting in 3D. In fact, as also reported in an experiment, made by Badler et al. [67], the presence of a real spatial reference can change the 3D interaction task from a “consciously calculated activity” to a “natural and effortless” action.

## 2.4. The VTK4AR framework

A video-tracking based AR application for scientific visualization needs to perform four fundamental operations: video capture, user position tracking, reading and managing data-set and a correct visualization of the AR scene. In vtk4ar the first two tasks are performed using the video-camera handling, and video tracking functionalities of the ARToolKit library suite (www.hitl.washington.edu/artoolkit), considered by now a standard de facto for Augmented Reality. Instead, as regards the management of data-sets and the visualization of the AR scene, VTK (www.vtk.org), a specific library for scientific visualization has been used.

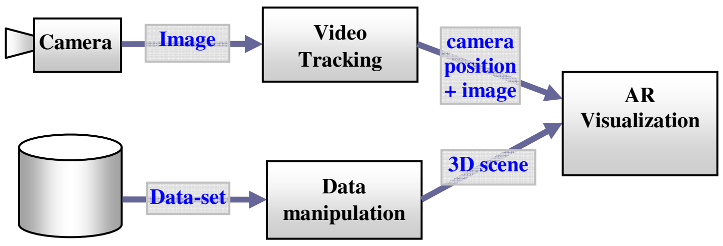


**Figure 14: Implementation schema of a video-tracking based AR application for scientific visualization**

There are many free libraries that perform such operations, but only open source software has been employed, both to keep license costs down and for the possibility to publish them achieving a large diffusion of the vtk4ar framework. As a matter of fact, it will soon be released under GNU/GPL license. Both libraries are cross platform, so all the porting problems have been bypassed (actually vtk4ar has been tested on Windows and Linux).

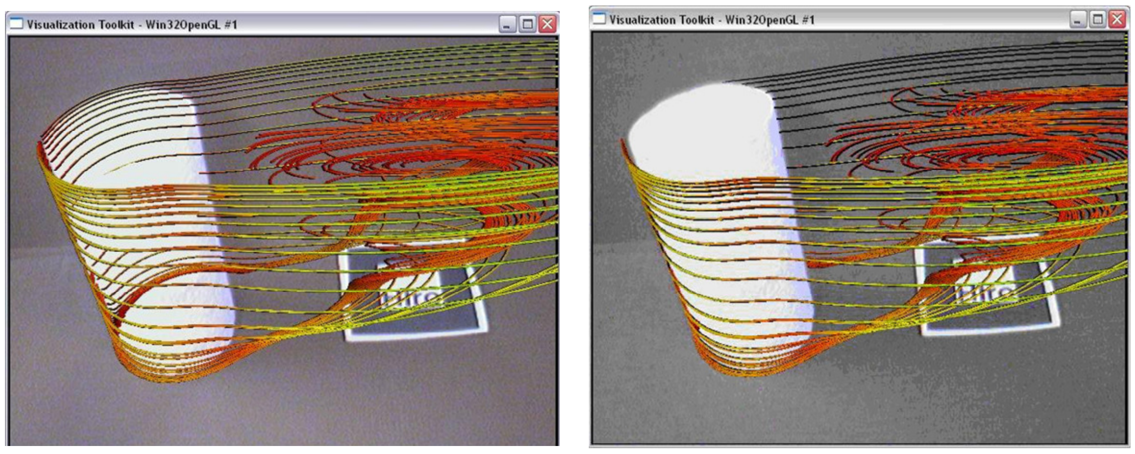
## 2.5. *Implementation*

VTK4AR is born to be a framework to visualize all kinds of datasets in AR with the availability of all the functionalities implemented in VTK. VTK4AR is made up by a set of high level software classes to simplify the development of new AR applications. It can be considered as an extension of VTK with the support for AR. In fact, VTK has been extended with some classes, specific for the scientific visualization in AR. Furthermore, a set of classes for an easy and object oriented handling of the video camera and tracking system has been coded.
Compared to the medical applications [68][37][69], in which the virtual data and the real images were mixed by the hardware, vtk4ar is able to completely handle the application, natively supporting the video see-through and the video-tracking functionalities. Also compared to the technology used in the Studierstube applications, vtk4ar represents an improvement. In fact, in the above-mentioned works, there was a total decoupling between the scientific visualization system and the AR user interface (Studierstube). The AR scene was obtained exporting

visualization data from the visualization system to Studierstube. As the authors also recognize, this approach has a bottleneck in the data conversion process that does not have to be repeated each time that visualization parameters change. Instead, vtk4ar allows a rapid implementation of all the interaction techniques for data manipulation, without any conversion.

## The occlusion problem

VTK4AR also manages the occlusion between real and virtual objects. As a matter of fact, one of the problems in AR visualization arises when real objects occlude 3D geometries, thus leading to a misunderstanding of the AR scene. In the case of CFD (Computational Fluid Dynamics) analysis of a cylinder such misunderstanding  is shown in figure 13.a. If the streamlines are drawn superimposed on the real scene, the result is not realistic, because the streamlines "do not wrap" the cylinder. In figure 13.b the correct occlusion of the cylinder has been calculated. The data-set, employed in the example represented in figure 4, is a demo included in the VTK package.


Visualization Toolkit - Win32OpenGL #1
Visualization Toolkit - Win32OpenGL #1

a) b)

**Figure 15: Streamlines wrapping a real cylinder without (a) and with (b) occlusion taken in account**

In previous works the occlusion problem has been tackled with different methods, but two general approaches can be identified. The first approach, defined as "model-based", implies that geometries of real objects are known. It means that the model of the real object, correctly positioned in the virtual world, can be used to produce occlusions by drawing it in black. These models have to be placed in the correct position in camera coordinates to produce an image identical to the live video image of the real objects. In other words, when the geometric model is rendered, it appears at the same location and orientation in the computer-generated image as the real object in the live video. In the system presented in [70] the video output of the graphics workstation is combined with the output of a video camera using luminance keying.

If the geometric models of the real objects are unknown, the depth-based approach can be employed [71][72][73][74]. Depth information has to be written in the Z-Buffer before the rendering of the virtual objects occurs. With this approach, the model of the real world is acquired reconstructing the depth of the real environment from the point of view of the camera.

In our methodology, the geometrical models of the real objects are available, so a model based approach has been used. Occlusion handling has been implemented thanks to a proper class, which writes in the Z-Buffer only. The matter will be analyzed in depth later.

## The ARToolKitHandle Library

In order to have an easy handling of the tracking and video libraries, and to have a full Object Oriented environment, the ARToolKitHandle, a C++ API for managing ARToolKit C libraries, has been developed.

The ARToolKitHandle library is made up of two main classes:

- ARVideoHandle class that provides an easy interface to the video devices;
- ARVideoTracker that is an abstract class, implemented in ARMonoMarkerTracker and ARMultiMarkerTracker classes, for managing mono-marker and multi-marker tracking.

Every ARVideoTracker object has associated an ARVideoHandle object, from which it acquires images that will be processed for tracking.

## 2.6. New VTK Classes for AR

The technique to obtain an AR scene is quite standard: virtual objects (i.e. the results of CAE analysis) must be rendered above the real scene obtained by the camera. To automate/mechanize the development of new applications, VTK4AR directly provides the developers with an environment made up by:

- a viewport to render virtual objects (vtkRenderer)
- a viewport to render the real image (vtkARImageRenderer)
- a window with two layers to contain the previously mentioned viewports (vtkARWindow).

The vtkRenderer class is a native class of VTK that renders actors (e.g.: 3D objects, data-set and images) in a window. Every vtkRenderer has its own camera, lights and actors.

The new class vtkARImageRenderer (directly derived from the vtkRenderer class) renders the camera grabbed images in the window. Every vtkARImageRenderer is linked by default to a ARVideoHandle object, which manages the camera, to generate the background image. Finally, the new class vtkARWindow contains a vtkARImageRenderer in the background and a vtkRenderer in the foreground.

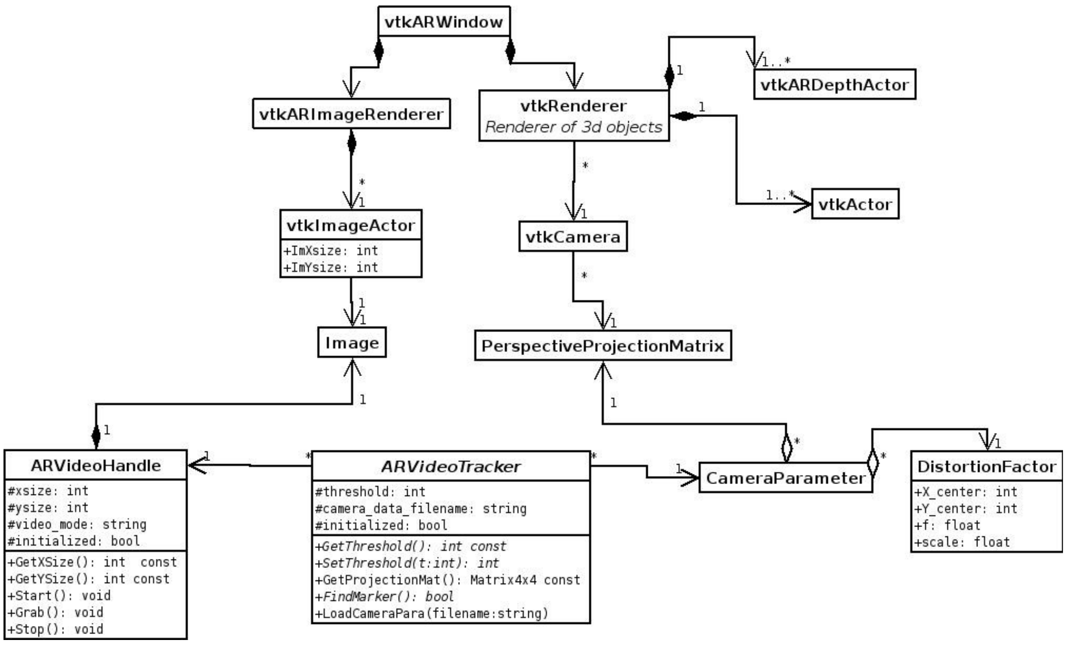


**Figure 16: UML class diagram of the framework**

In order to be able to display the occlusion between real and virtual objects correctly, a new class called vtkARDepthActor has been implemented. The main feature of this kind of actor is that it will be rendered in the Z-Buffer only. VtkARDepthActors must be rendered before the standard vtkActor. In this way, the depth-buffer is filled with the Z values of real objects, before virtual objects are rendered, and, therefore, only the visible parts of the virtual objects will be drawn.

## 2.7. Case Studies

In this section three case studies of our methodology are presented.

The first case (figure 6) shows the results of a CFD simulation of the flow past a helmet. The application of our methodology to this case has four steps:

1. the CAD model of the product is obtained by means of reverse engineering techniques, in particular a 3D laser scanner Minolta Vivid 300 (VI-300) has been employed;
2. a CFD analysis is performed on this model using COSMOSFloWorks (www.solidworks.com) and the results have been exported in ASCII format;
3. the results are imported, through a data reader developed ad-hoc, in Paraview (www.paraview.org). It is an open-source postprocessor, built on top of VTK and it has been employed to properly choose visualization parameters.
4. the processed data are visualized in AR with vtk4ar.

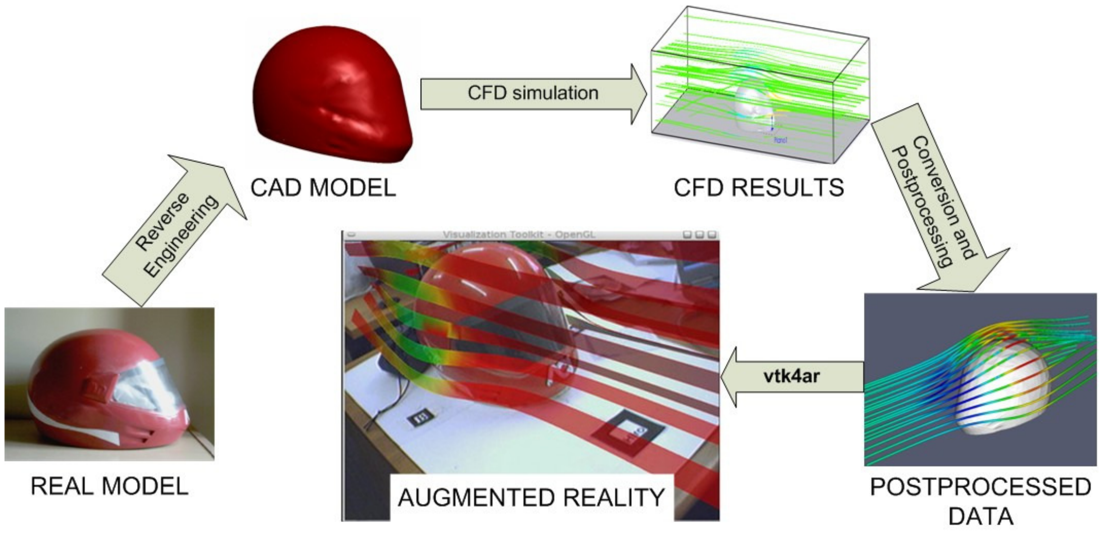


**Figure 17: Schema of the methodology employed**

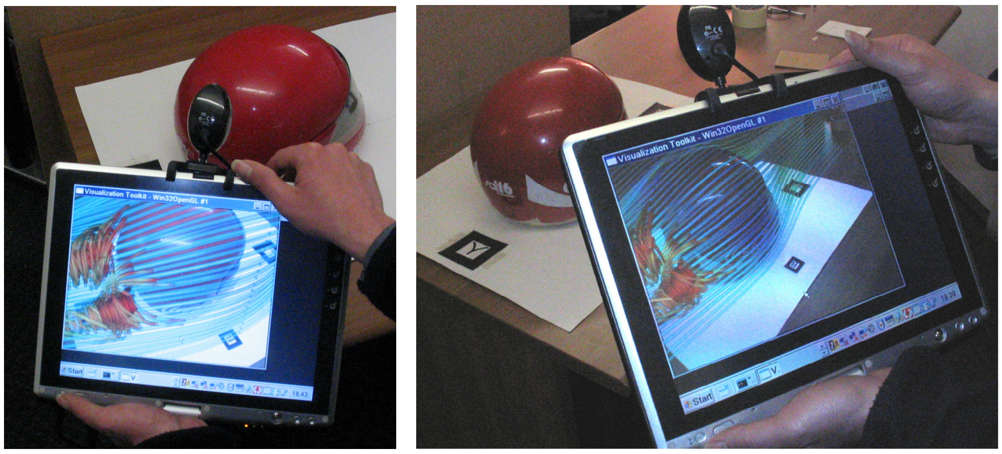


**Figure 18: Virtual streamlines around a real helmet**

The performance of the system, obtained in this test case, are reported in the table below. Two different visualization techniques have been tested: Stream surfaces (central image of figure 6) and stream lines (figure 7).

Table 1: Performances obtained with two different visualization techniques

| Model | Number of points | Number of Polygons | Frame Rate |
|---|---|---|---|
| Stream-surfaces model | 12270 | 532 | ~28 fps |
| Helmet model | 1723 | 3442 | |
| Stream-lines model | 29921 | 58828 | ~20 fps |
| Helmet model | 1723 | 3442 | |

The second test case focuses its attention on the manufacture of an ankle support, which is made starting from what we can roughly call the "patient geometry". As concerns production technology, Incremental Forming [75] has been utilized; in this kind of application, one of the main drawbacks is represented by the discrepancies between the desired shape and the obtained one. For this reason, a final evaluation of the manufactured product has been carried out at the end of the production step, using AR visualization. The methodology employed in this test case has five different steps:

5. the CAD model is generated starting from the patient ankle geometry, acquired by a 3D laser scanner;
6. the prototype of the ankle support is built deforming a metal sheet;

7. the geometry of the prototype is acquired by a 3D laser scanner and the cloud of points, which represents the real product, is compared, in Pro-Engineer environment (www.ptc.com), with the CAD model of the designed surface. This operation generates a map where, for each point, the discrepancy between the designed and the obtained shape is represented by a scalar value (discrepancies map);
8. the discrepancies map is visualized in AR superimposed to the prototype, so the user evaluates the accuracy of the forming process.

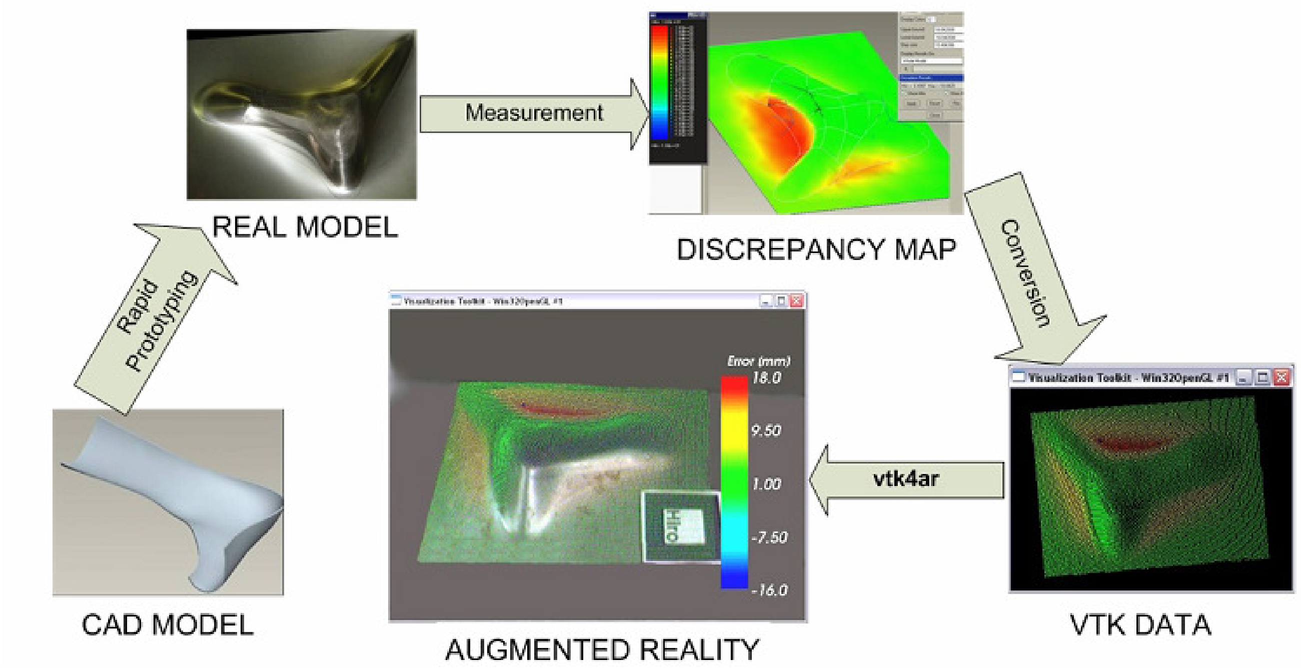
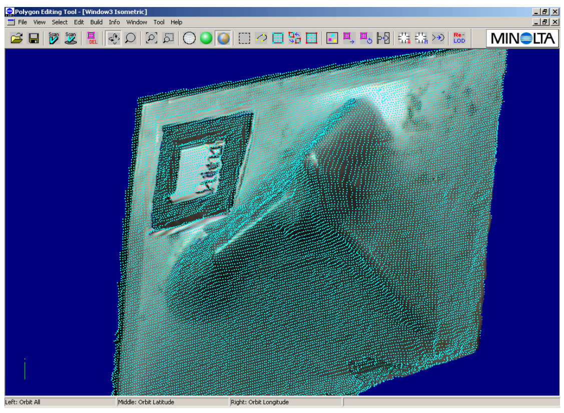


**Figure 19: Schema of the methodology employed in the second test case**

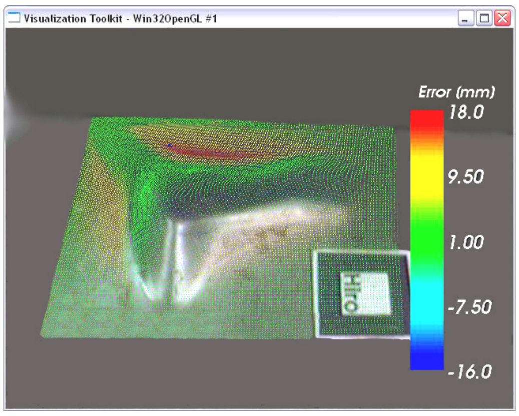
The marker needed for the optical tracking has been placed on the physical prototype of the ankle support before the scanning. The texture has been acquired using the camera integrated in the scanner, so the marker appears, drawn on the 3D model surface in the acquisition software. The user can easily determine the position and orientation of the marker in the CAD coordinate system, by retrieving the coordinates of some points belonging to the marker itself.

The data-set displayed in AR is composed by 15345 points and it is rendered as a point-set, which is a fast rendering technique. The frame-rate obtained is quite high (~29 fps) also because there is no need to consider the occlusion.

Due to the frame grabbing speed of the camera the frame rate of the application cannot exceed 30 fps.


Polygon Editing Tool - [Window3 Isometric]
File View Select Edit Build Info Window Tool Help
MINOLTA
Left: Orbit All
Middle: Orbit Latitude
Right: Orbit Longitude

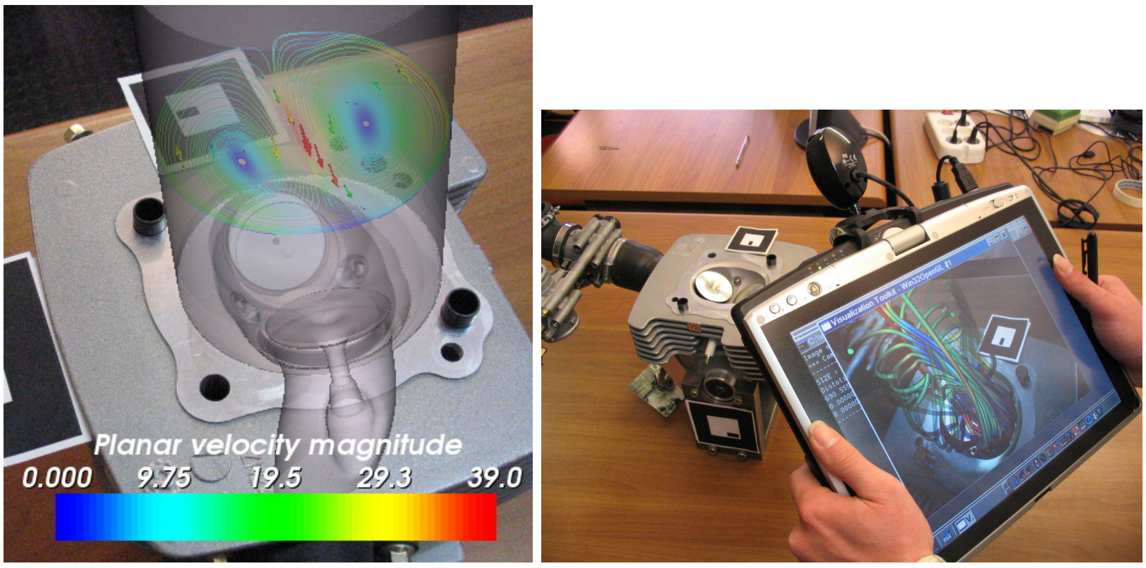
a)



b)

**Figure 20: The cloud of points of the ankle support obtained with a 3D laser scanner (a) and the visualization of the discrepancies obtained in the forming process (b).**

As it is easy to understand, it is possible to use the framework for general visualization purpose, without following the methodology. In the third case study therefore, the framework has been employed in order to display data of an analysis of the flow field in the intake system of a high performance production engine motorcycle. In particular, the results of both numerical and experimental studies were visualized and compared employing the AR visualization system. The numerical simulation has been performed using Fluent 6.0, while the experimental results have been obtained using Laser Doppler Anemometry (LDA) techniques.



a ) b)

**Figure 21: a) Experimental and Numerical comparison and b) stream-tubes of the flow field of the intake system**

## *2.8. Conclusion and Future Developments*

Industrial engineering deals with many kinds of data-sets from numerical simulations, measurements or physical experiments. Often these data-sets have to be presented to non-technical people like customers or managers, but this requires that the visualization techniques have to be as easy and clear as possible. This work presents an innovative methodology that integrates different engineering technologies with augmented reality and scientific visualization, in order to achieve an effective way to present and analyze industrial engineering data. Our methodology allows the data to be displayed directly superimposed on the physical objects, so the user can more easily understand the meaning of the data-sets and their relations with the studied objects. The software framework VTK4AR has been developed to allow the rapid implementation of AR applications, based on this methodology. Moreover, VTK4AR is able to visualize in AR any kind of data supported by VTK, so it could be employed not only in industrial engineering applications, but in every field dealing with scientific visualization.

As concerns future developments, it is possible to implement a formal usability test to quantify benefits of this methodology, which must be weighted against the cost of preparation time. In fact, visualizing data in vtk4ar requires data to be imported in VTK and some alignment of real and digital objects to be carried out as previously described. This task requires an effort that depends on the data complexity, on the scene extension in which data are presented and on the number and the shape of physical objects that can occlude data.

Even if in the industrial practice it is very important to weigh benefits against costs, in some visualization tasks the use of augmented reality, not only improves the human-computer interaction, but it could represent a new useful tool.

Two possible future applications have been identified:

1. explain physical phenomena to students from scientific and technical faculties
2. compare numerical and experimental data

The latter is the case of some experimental analyses in which the results could not be expressed with numerical values. For example, in aerodynamic studies in the wind tunnel, techniques like smoke injection, placement of tufts or streamers into the flow [76] and pressure-sensitive paint [77] are used to obtain essential information to be used in the prediction of aerodynamic performance. This data can be retrieved and represented in a virtual environment [78], but by using augmented reality techniques the results of the same experiment, realized both physically and numerically, could be visualized at the same time, superimposed on the subject of the experiment. Comparing the results of a numerical simulation and the corresponding physical experiment in augmented reality; as a matter of fact, the user can perceive possible discrepancies between them immediately, making the refinement of the simulation parameters easier. Moreover, the employment of AR visualization in physical experiments can be crucial when material defects change the expected behavior, when numerical models are not accurate enough in predicting the real behavior.

# Chapter 3. An Integrated Environment For The Validation In Virtual Reality Of The Dynamical Behavior Of Functional Digital Mock-Ups

In the previous chapter an innovative way to explore numerical data has been presented. The previously described methodology can be useful when numerical data regard finite element analyses. However, numerical simulations can also be addressed to the simulation of the behavior of a mechanical system. In this case, numerical data can be computed dynamically, in order to animate the virtual prototype and faithfully reproduce the behavior of the mechanical system. In order to improve the comprehension of such simulations, the visualization can be performed in a Virtual Reality environment.
Virtual Reality (VR) technologies in fact, are widely employed in the Product Development Process (PDP) in several industrial sectors like automotive, aerospace, defense, bioengineering, etc.[79][80].

The main target of the VR in industry is the preview/foresight and the discovering of any possible design flaw before the realization of physical prototypes. The aid provided by VR is noticeable, since the user can interact with the virtual prototype in a very natural way. The earliest industrial VR applications merely provided the visualization of the digital mock-up (DMU) in VR in order to perform an aesthetical validation of the product. The DMU is the geometrical representation of a more or less complex product. DMU visualization in VR is crucial to exploit the overall appearance of the product, but it can also be useful in order to exploit part reachability, cabling, etc.

More recent applications instead, try to simulate in VR the real behavior of the industrial products, i.e. try to simulate the functional virtual prototype. In this way it is possible to use VR not only for aesthetical purposes, but also for discovering any functional flaw.

An example of the use of functional virtual prototypes in the PDP is presented by Bruno et. al. who describe a methodology for the usability assessment of an industrial product, based on the creation of a usability test, carried out in a virtual environment. This approach allows designers to conduct the test also in the first stages of the PDP without worrying about the realization of physical prototypes needed to perform the test with the user. Authors point out that the behavior of the digital mock-up employed in virtual usability tests has been generated by implementing the logic of the product interface inside the code of the virtual reality application and this approach entails several problems that will be discussed in section 3.2.

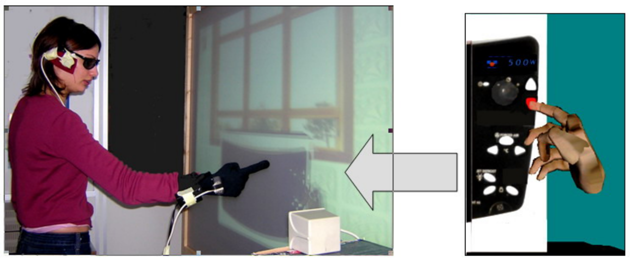


**Figure 22: The functional virtual prototype implemented for usability tests**

The lack of software tools able to support designers in the development of interactive and fully functional virtual prototypes is probably one of the main obstacles in the diffusion of VR techniques for the product behavior simulation. At the moment it is not possible to evaluate, in VR, models developed in simulation package. One of the few approaches devoted to the integration of simulation package and VR has been presented by Kirner et al. [81] who developed the VR-SIM, an object oriented C++ library, able to incorporate a Real-Time Systems (RTS) simulator with VR technologies. The use of VR-SIM involves the creation of the system to be validated and of a virtual environment related to this system. The case-study is a robot arm coupled to an automatic transport-belt, used in a factory for piling up boxes. This work demonstrates that VR technology is applicable and useful to support RTS simulations, as a form used to evaluate the correctness of such systems. But the VR-SIM is a tool addressed to software engineers responsible for the development of real-time, process control systems, it requires code development for the implementation of the virtual product and it is not suitable to be used by industrial or mechanical engineers in the PDP.

A similar work has been presented by Sánchez et al.[83] who developed the Easy Java Simulation (Ejs) [84], a software tool designed to create interactive simulations in Java using models created with Simulink. Basically, Ejs is able to communicate with the Simulink model and to tightly control its execution. The communication takes place through a DLL library written in C and some utility Java classes. The main advantage of this work is that Ejs creates Java applets that are independent, multi-platform, which can be visualised using any Web browser, read data across the net and be controlled using scripts from within html pages. Ejs can also be used with Java 3D to create interactive virtual products in 3D, but it has been conceived mainly for educational purposes and it cannot be efficiently integrated into a PDP because Java 3D is not suitable for complex models visualization.

The aim of this work is the development of an environment for the evaluation and the validation of both the aesthetic and the functional part of an industrial product. To achieve this target, it is important to consider both the geometrical model and the behavioral model of the product and let them interact each other. The behavioral model considers the dynamic features of the product, due to its mechanical and electronics properties. From a mathematical point of view, it is a differential equations system, but it can be modeled and simulated using a Computer Aided Control Engineering (CACE) software and/or a multi-body solver. We can summarize everything saying that with the visualization software we can validate the geometric model of the product, and with the multi-body solver we can validate the mechanics and the dynamic of the multi-body model. Our purpose is "linking" the geometrical and the multi-body model in order to visualize in VR the functional virtual prototype of the industrial product.

## 3.1. Implementation

In order to achieve our purpose, an high level software library for the inter-process communication (IPC) has been developed. This library let the simulation software communicate with the VR environment.

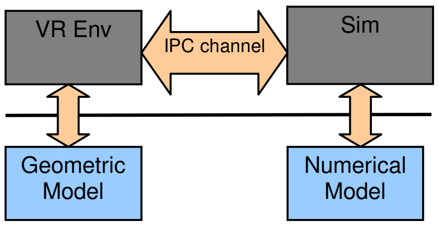


**Figure 23: Framework architecture**

We choose Matlab/Simulink as simulation software. This environment is almost a standard for the general purpose simulation. It is very diffused and versatile. Further, a lot of optional packages (called toolbox) are present. These packages provides further sets of high level operations for a specific task. In this work the SimMechanism toolbox has been used as multi-body solver. With the SimMechanism toolbox it is possible to model and simulate a mechanic assembly, specifying properties for bodies and joints.

The software used as virtual environment are Division MockUp by PTC and Virtools Dev by Dassault Systemes. They will be described in the next sections.

## 3.2. Division MockUp

Division MockUp provides several virtual environments visualization and exploration functionalities and, further, every application can be customized. Division MockUp in fact, provides an SDK for the plug-ins development, and a scripting language to create virtual environments.

Virtual environments are re-created using a VDI file, i.e. a plain ASCII file containing the description of the virtual scene. It contains both the hierarchy of the assembly and the attributes of each part of the assembly. Division is made up by two software module: a low-level core called dVS, and an high level interface, called dVISE. Further, it is possible to use the EC library, i.e. an API to dVISE, in order to easily achieve the following tasks:

- Define new action funtions to customize assemblies' behaviour.
- Create assemblies, cameras, reference points and annotations
- Execute functions on the assembly's hierarchy

The communication between Division and Matlab can be synchronous or asynchronous. The sender (i.e. MATLab/Simulink) sends data deriving from the simulation in asynchronous mode using non-blocking sockets, and does not take care about the reception of the visualization environment. In this way, the simulator compute the simulation results also for the next simulation cycles, without stopping the computation.

Division instead, adopts the synchronous mode communication, using the blocking sockets. It stops its execution until it gets the message from MATLAB/Simulink. Once received, it moves the 3D model using the obtained information, and it sends the request for the new data. As it is easy to understand, a new thread take care of the communication. In this way the user can still interact with the 3D scene.

Resuming, Matlab/Simulink does not send new data computed for the next simulation cycles until a DIVISION request occurs. Division instead, stops the execution and waits for the whole data from Matlab/Simulink, then sends a data request to the simulator.

## 3.3. The SIMLib library

In order to simplify the development of such applications, it is necessary the implementation of an high level communication library. This library should provide an easy to use IPC channel for the communication between the VR environment and the solver. Therefore, we implemented the SimLIB library. It is a versatile library that can be easily adapted to all the possible test cases. SIMLib uses TCP socket, therefore it is possible run the simulator and the VR application on different machines.

The SIMLib library is quite easy to use. It is made up by few functions, in which is implemented the code for the TCP/IP communication and synchronization. Therefore, the developer must not take care about sockets and thread.

The functions of the library are:

- SimLib_Channel* **SIMServerOpen**(unsigned short port, int connections, u_long non_blocking).
- SimLib_Channel* **SIMClientOpen**(const char* host, unsigned
- int **SIMClose**(SimLib_Channel* s)
- SimLib_Channel* **SimLib_Accept**(SimLib_Channel* server)
- int **SimLib_synchroSend**(SimLib_Channel* to, SimLib_Data* r)
- int **SimLib_synchroReceive**(SimLib_Channel* s, SimLib_Data* r)
- void **SimLib_SendNextData**(SimLib_Channel* s)
- void **SimLib_StartListener**(SimLib_Channel* server, void (*ptActionCB)(const SimLib_Data))
- void **SimLib_StopListener**()

The SIMServerOpen function creates a server communication channel listening at the specified port. It is possible to set it as a blocking or non-blocking communication channel.

The SIMClientOpen function creates a client communication channel, which attempts to connect the server specified by parameter. It is possible to set it as a blocking or non-blocking communication channel.

The SIMClose function closes and remove from memory a communication channel.

SimLib_Accept function establishes a connection between the server specified as parameter and the connecting client. It returns a new communication channel which is the one to be used by the client for the communication.

The SimLib_synchroSend and SimLib_synchroReceive functions send and receive data using a common protocol for the IPC. The developer must specify the SimLib_Data data structure used for the communication. Then, the struct is sent and received through the IPC channel. The data structure can be also very complex and huge, even if in this way the connection speed is slower.
The SimLib_StartListener and SimLib_StopListener functions respectively create and destroy the listener thread. The listener thread can be created by a server communication channel to handle the connection with a client. If there is a listener, the server communication channel must be in blocking mode. A listener thread calls another thread (called Answer thread) for each connecting client. The answer thread receives data from the client, and after calls a callback function (if specified) which can use data received from the client.

## 3.4. The Simulink S-Function for the communication

The Matlab/Simulink environment can be extended by the development of user defined S-Function. These ones can be used within a Simulink model as a conventional Simulink building block, with a user defined behaviour and set of actions. In our purpose, the S-Function is responsible of the communication between MATLab/Simulink and DIVISION. As it is easy to understand, the communication make use of the SIMLib library and rely on the IPC channel provided by this one. The main task of this S-Function is the asynchronous data sending to DIVISION. In other words, the S-Function sends simulation data to DIVISION, without stopping the simulation. In order to obtain a consistent visualization in fact, there is the need of sending all data of a single time-step. Since in the model there is more than one part governed by the simulator, it is important not to send data of different time-step to achieve a correct visualization. To achieve this target, we used the information from the Matlab ssGetT function, which returns the simulation time. As explained in the next image, once the OK message from DIVISION is obtained, all the blocks send data in a sequentially order.

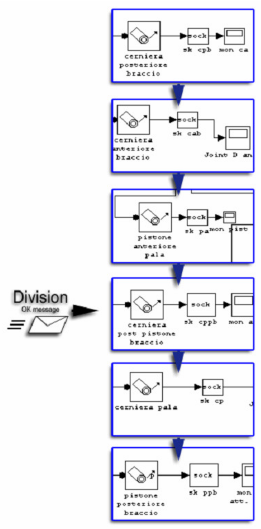


**Figure 24: Asynchronous receiving of the DIVISION OK message**

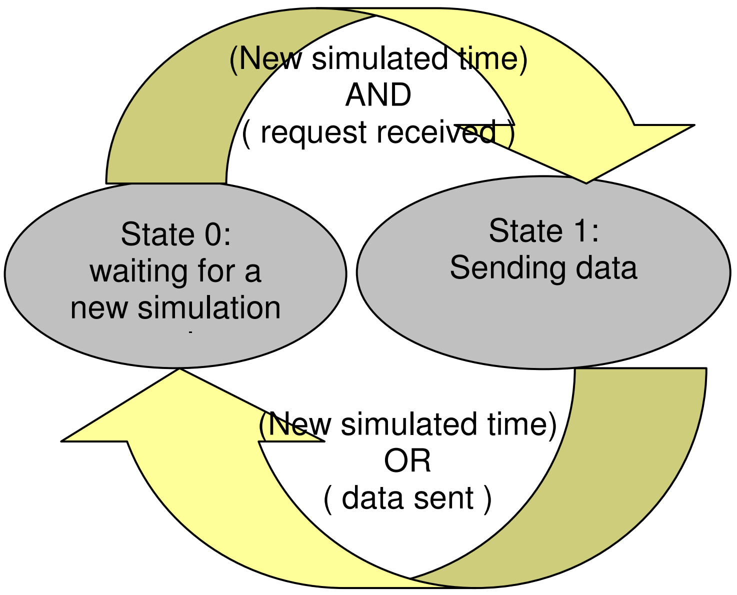
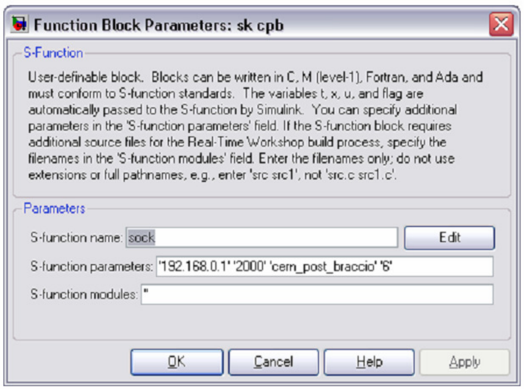


**Figure 25: Finite State diagram of the S-Function**

The configuration of each communication block is quite easy. It is possible in fact, set the network parameters and the part name via GUI as shown in next figure.



User-definable block. Blocks can be written in C, M (level-1), Fortran, and Ada and must conform to S-function standards. The variables t, x, u, and flag are automatically passed to the S-function by Simulink. You can specify additional parameters in the 'S-function parameters' field. If the S-function block requires additional source files for the Real-Time Workshop build process, specify the filenames in the 'S-function modules' field. Enter the filenames only; do not use extensions or full pathnames, e.g., enter 'src src1', not 'src.c src1.c'.

S-function name: sock

S-function parameters: '192.168.0.1' '2000' 'cern_post_braccio' '6'

S-function modules: ''

**Figure 26: GUI for the S-Function configuration**

The parameters needed by each block are:

- IP and port of the host to which data are sent
- the name of the part which is governed by the block
- number of the blocks in the Simulink model.

## 3.5. The Divison plug-in

The plug-ins are software tools used to link Division to customized applications. Firstly, during the plug-in initialization, a server communication channel is created. Then a new thread for the communication is created using a SimLIB_listener. The callback functions of this one are *applyAction* and *received*. The *applyAction* function implements the modification of the assembly in accord with received data, while the *received* function sends the signal of end reception. It is necessary to save the references to the assemblies to be moved in order to animate the model. These references in fact, are used by the *applyAction* call-back to animate the model. The _saveRef function has been implemented in order to save the references to the assemblies into an hash table. It is important to save the reference to the assembly with the same name specified in the Matlab S-Function. The name of the assembly contained into the SimLib_Data sent by the S-Function in fact, is the key search within the hash table.

## 3.6. Test Case

The test case regards the simulation and relative visualization of a tractor. It is possible to simulate the movements of arm and bucket. It has been necessary to add six communication blocks in the Simulink model. These blocks can send the orientation of:

- Lower hinge of the arm
- Lower hinge of the support cylinder of the arm
- Lower hinge of the bucket
- Lower hinge of the support cylinder of the bucket

And the translations of:

- Piston of the arm
- Piston of the bucket

These communication blocks are highlighted in the figure below.

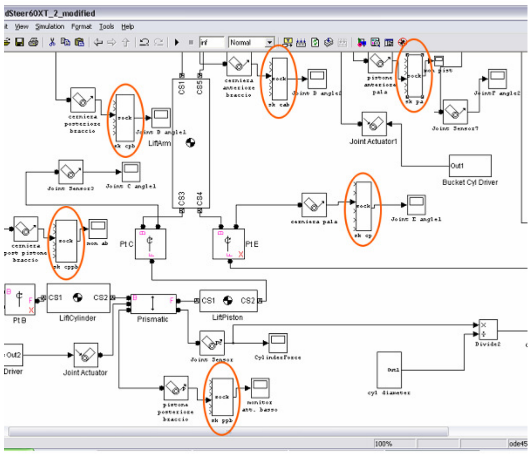
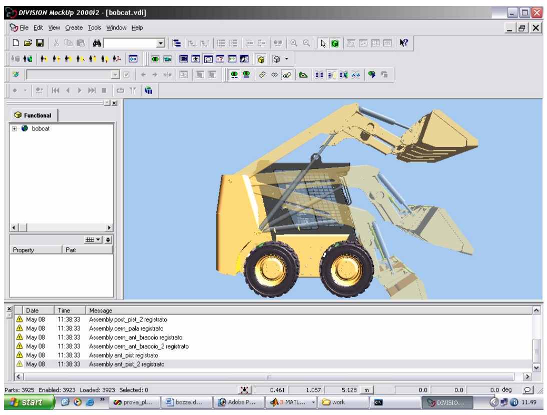


**Figure 27: Building block for data communication within the Simulink model.**

Each of these communication blocks receives data relatives to the position/orientation of a joint. The first three ports (starting from the upper side of the block) are used for the rotations in the three axes, and the last three ports are used for the translations in the three axes.

As regards the 3D model of the tractor, it is necessary to find and to registry the assembly corresponding to the Simulink blocks. Once the part has been found, it must be renamed using the same name of the parameter in the Simulink communication block, as previously described. In the 3D model, there is an hierarchy among the several parts which made up the assembly. In other words, it is possible to specify a parent/child relationship between two parts. Once this relationship has defined, the child will follow the same movements of its parent. Therefore, moving a part, all the children of this part will follow the same movements. This feature is useful for our task, because once defined relationship among the parts, it is sufficient move parent of a kinematic chain to obtain the movement of all its children. In the 3D model of the tractor e.g., a parent/child relationship between cylinder and piston of an hydraulic actuator.

With the use of a common joystick it is possible to govern the actuators, like in a real tractor. The Matlab/Simulink environment in fact, provides the interface with such devices via Virtual Reality toolbox. During the simulation, it is possible to notice a delay between the command and the response. This delay is due only to the complexity of the 3D model. DIVISION, in fact, has to complete the rendering for each position. After simplifying the 3D model deleting all the parts not interested by the simulation, the response of the system was really high.



```
Date    Time      Message
May 08  11:38:33  Assembly post_pist_2 registrato
May 08  11:38:33  Assembly cern_pala registrato
May 08  11:38:33  Assembly cern_ant_braccio registrato
May 08  11:38:33  Assembly cern_ant_braccio_2 registrato
May 08  11:38:33  Assembly ant_pist registrato
May 08  11:38:33  Assembly ant_pist_2 registrato
Parts: 3925  Enabled: 3923  Loaded: 3923  Selected: 0    0.461  1.057  5.128 m    0.0  0.0  0.0 deg
```

**Figure 28: The division environment during the simulation**

As a further investigation with this test case we have implemented an application for visibility analysis.

This application allows us to simultaneously run two executions of dv/Mockup. One execution is directly connected to the input devices and it runs the simulation in first person. The operator wears the head mounted display and controls the virtual tractor by a pair of joysticks.

The second execution allows engineers to view in third person what happens in the simulation. And in particular they may analyze also the viewing cone of the operator that controls the tractor.

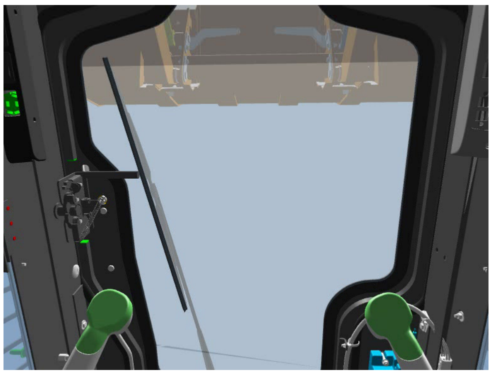


**Figure 29: The operator's view inside the cabin**

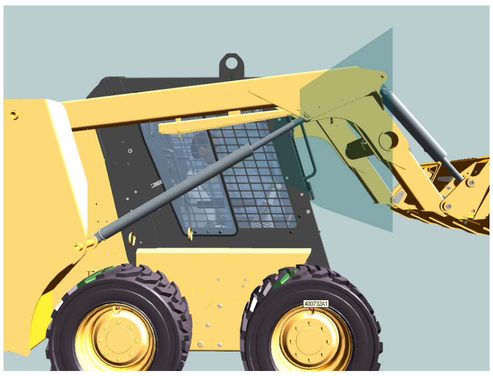


**Figure 30: Visualization of the viewing cone of the operator**

## 3.7. *Summary*

This work describes an experimental environment for the validation of functional digital mockups. It can be considered as one step towards the functional digital mockup. Two environments are needed in order to achieve this task: a virtual reality and a numerical simulation environment. The "glue" is an inter-process communication library developed for this purpose. In the test case we used Division as VR environment and Matlab/Simulink as numerical solver, but the employed methodology can be used with other software environments, even if the development of the interfaces between each software and the IPC library is needed. The effort for the interface development represents the main drawback of the proposed methodology.

Another drawback is represented by the 3D model modification in order to be animated by the numerical solver. This modification regards only the names assigned to the parts, but it can be a time-consuming and error-prone operation.

Further, in the test case a novel approach for the visibility analysis is proposed. Designers can easily discover all the non visible area during the validation.

In the provided test case, the user can interact with the model in a very natural way. The test case in fact, could be easily converted into a training operator application. To achieve this target, the application should be able to perform the collision detection, which represents another step forward towards the realism.

# Chapter 4. Multi-domain co-simulation using Multi-Body and Electronic Control System solvers

As highlighted in the introduction, the observation is crucial for the understanding of phenomena. This sentence, of course, is valid also in the engineering field. In fact, even if the experimentation is done in a virtual environment, the evaluation and the validation of the Digital Mock-up should be possible to evaluate the performance of the simulated Virtual Prototype. During the design phase, in fact, it should be possible to validate the prototype and its functionality. This is true above all for all the products having an integrated control system, like all the mechatronic systems.

In recent years, the interest of both industrial companies and universities in mechatronics was noticeable, and thus, mechatronics has undergone a deep development.

Mechatronic systems design is based on an interdisciplinary combination of domains: mechanical engineering, electrical engineering, electronics, and computer science. They are characterized by a tight coupling of different implementation technologies, e.g. hydraulics, mechanics, pneumatics, electromechanics, electronics and software [85].

For this reason, the development process for mechatronic products is different from others, in the sense that it spans over many closely coupled engineering domains. In recent years existing design methodologies of mechatronic systems have undergone a progressive advancement required by the need to manage a rising of design complexity, reduce time-to-market, integrate heterogeneous fields like electronics, mechanics, and software into a single product. This have stimulated a steadily enhancement of existing design environments, tools, and methods. Such development process typically applies a subsystem-based approach. With the term “subsystem based” is intended a product development strategy by which integrated systems are built from technology homogeneous subsystems (mechanics, electronics, control and software). The subsystems are developed in a concurrent manner with an important focus on subsystem interfaces. Once the interfaces are designed, each subsystem is designed in a fairly traditional way. The subsystem-based approach to mechatronics is still a drastic improvement from the early days when the mechanical engineers first designed the mechanical system, which was then handed over to the control engineers doing a control design [86].

It is very useful to validate the project of the entire system through simulation, before building the physical mock-up. The realization of the physical prototypes, in fact, is usually a complex and expensive task in which design errors and non-optimal solutions, related to the design phase, cause lateness in product launch and development costs increase.

In order to simulate a mechatronic system, a multi-domain simulation environment is required. Multi-domain simulation could be achieved in two different ways: the first and more traditional way is to use a general-purpose solver to simulate the entire system, while the other way, called co-simulation, is to use different solvers that communicate each others, to simulate each sub-system. Each of these multi-domain simulation techniques has its own advantages and disadvantages. One of the main advantages of the co-simulation is that each solver is optimized for its own specific domain.

In fact, from the software point of view, co-simulation is not only a simulation performed in two or more different physical domains, but it is a simulation in which several software run in parallel and continuously exchange data to verify the behavior of each different components of the product under the specific working conditions.

Further, co-simulation emphasizes the subsystem design approach, in which the interfaces among the sub-systems remain the same, but an inter-changeability among the possible models of a sub-system is allowed.

Using co-simulation, in fact, there is no necessity to import the several models of different domains into one multi-purpose simulation software, but each model is simulated in its own environment without any import operation. The main disadvantage of the co-simulation approach, instead, is represented by the non trivial effort to synchronize the different solvers, also if a lot of simulators can be interfaced each others.

In our opinion co-simulation techniques are not still exploited, because these are employed only at the end of the design process to validate the choices taken by engineers. Instead co-simulation can be intended as a technique able to support the designers also in the earliest stages of the design process.

This work describes a design methodology devoted to the development of mechatronics product supported by an integrated and interactive co-simulation. This methodology, in fact, allows designers from different domains (mechanical, electronic and software engineers) to follow the top-down approach in which, starting from the conceptual phase, they can check, at each milestone of the project, the functionality of the product at the different stages and for different levels of detail.

At the end of the design process the engineers can directly interact with the mechatronic model using the product interface implemented in the control system. The actions of the user are processed by the electronics simulation software that dialogs with actuators, motors and sensors placed in the multi-body model inside another specific simulator. So the 3D model in the multi-body software interactively responds to the actions performed by the user.

The case study, implemented to test the proposed methodology, is the complete design of a mobile elevated working platform. The machine has four driving and steering wheels, four outriggers provided to make the machine stable and an articulated arm with six degree of freedom. There are 22 actuators for the movement of the various parts and each of them has a sensor position for the feedback control. Thanks to the availability of the integrated behavioural model of the machine it has been possible to implement a functionality for the automatic reconfiguration of the outriggers that allows the user to move the platform employing only three outriggers at a time.

The case study presented puts in evidence that co-simulation can be an innovative technique to develop new design methodologies devoted to the mechatronic field.

A lot of hi-tech industries can benefit of this approach that allows designer from different domains to check their idea in an integrated model that includes the mechanics, electronics and software models.

## *4.1. State of the Art*

The key of a mechatronic design is concurrent and multi-disciplinary engineering. One of the earliest approach to the multi-disciplinary design for mechatronics was described in [87] and [88]. The methodology described in these papers was based on a software platform, called Schemebuilder, that is a design tool aimed at guiding designers through the several design options, and assists the designer in the conceptual and embodiment stages of design. It can be considered like a multi-disciplinary decision support system.

Another example of mechatronic design methodology is described in [89], in which a general model is derived to mathematically describe the concurrent design of a mechatronic system. Based on this model, a concurrent engineering approach is formally presented for mechatronic systems design.

Also companies like Bosch GmbH spent a lot of efforts in the formulation of mechatronics design methodologies [90]: Bosch started the initiative "Systems Engineering Mechatronic" to cope with the challenges of system design and complexity handling. Targets are high design efficiency (reduction of development time and cost) as well as high design quality (design correctness). New design methodologies and processes are currently established in different business units (automotive equipment, consumer goods and industrial equipment). Systems Engineering Mechatronic is based on the V-model for system design [91].

Artificial Intelligence (AI) and genetic algorithms are used in some design and optimization methodologies of frontier researches: in [92] the Niching genetic algorithm (described also in [93], [94] and [95]) is used to find local and global optimal design alternatives, while in [96] is presented a methodology for automatic generation of robots high-level control using a new evolutionary methodology [97].

In [98] the development of a brake by wire system is presented. The simulation is performed at the early design stages of the design process, to depict just from the beginning any possible design flaw. Authors put in evidence that analysing and simulating a mechatronic system at an early design stage is of great help to find the optimum design solution. Unfortunately, they present only the case study, without deriving a general methodology neither giving any suggestions to employ simulation techniques in the different stages of the design process.

Co-Simulation has been used also within several companies. Ford Motor Company used it to simulate a Vehicle Attitude Control (VAC) System [99]. The authors present a comparison between a simulation based on a single solver and another based on co-simulation technique. They report that co-simulation provides a more complete representation of control system and vehicle by selectively using the strengths of each application, but it requires more computational resources.

Visteon Corporation used co-simulation to simulate a vehicle stability control system. On the basis of mathematical models developed, a virtual prototyping vehicle to simulate winter test events was used to validate the applicability of driveline torque controls to maintain vehicle stability. These results could serve as the upfront analysis of the control strategies and the influence of torque biasing devices on vehicle responses before building the real hardware.

## 4.2. Methodology

The basic idea of the methodology presented here, is to enrich the classical top-down design approach with the use of co-simulation tools. In other words, the methodology establishes a cooperation among the various specialists in each engineering domain (mechanics, electronics, software, etc.) by defining some milestones in the product development process in which all the contributions from each domain are integrated and tested together.

The methodology is generally valid for a large set of mechatronic products, but it is particularly intended for the development of product characterized by a set of moving components that have to be studied in different possible configurations determined by the user and/or by a control logic. This kind of products are usual in different fields like robotics (e.g.: motorized arms), automotive (e.g.: windshield wipers), etc.

At each step of this top-down design methodology the DMU is progressively defined and its components are detailed. Each version of the DMU is validated by a co-simulation that involves both the mechanical and the control model. If the result of the co-simulation is the desired one, then the designers can improve and refine their models for the next step.

At the early stages of the methodology the designer defines some macro-components that are the main parts of the product used to sketch a preliminary kinematic model. These macro-components are initially identified with a simplified geometry made up by few primitives. Later, the macro-components are detailed with one or more components and the kinematic model is enriched with forces, torques, masses and other data to become a dynamic model. At the last step of the methodology the final geometries are modelled and inserted in the dynamic model. So it can be enriched with friction and the estimated masses are substituted with the real one calculated on the final geometries, knowing the density of the material.

The main steps and tasks of the methodology are the following.

Step 0: Conceptualization

0.1 Roughly dimension the product and define the main functions and macro-components.

0.2 Identify the main performances and constraints of the product.

Step 1: Kinematic model design

1.1 Define a rough multibody model composed by the overall kinematics, constraints and degrees of freedom for each macro-component.

1.2 Define a preliminary interface between the product and the user and/or the environment.

1.3 Design a first control logic.

1.4 Connect the control logic with variables (position and velocity of each macro-component) defined in the mechanical model.

1.5 Test the kinematic co-simulation model and evaluate performance and constraint satisfaction.

1.6 Perform the necessary modification on the models until the desired performances are reached and the constraints are satisfied, as defined in task 0.2.

<u>Step 2: Dynamic model design</u>

2.1 Detail the multibody model by defining the dynamics of each components and estimating physic properties like forces, inertia, mass.

2.2 Refine actuators and sensors in the mechanical model.

2.3 Refine the control logic and update it to manage the new connections with the mechanical model.

2.4 Test the dynamic co-simulation model and evaluate performance and constraint satisfaction.

2.5 Perform the necessary modification on the models until the desired performances are reached and the constraints are satisfied, as defined in task 0.2.

<u>Step 3: Detailed design</u>

3.1 Import macro-components in the CAD and use it as a reference for modelling in the next task.

3.2 Model the geometries of each component with the CAD.

3.3 Import the detailed geometries in the multibody simulator, substituting the macro-components previously defined.

3.4 Define friction and materials for each detailed geometry in the mechanical model.

3.5 Implement the control logic defined in task 2.3 with electronics components and software code.

3.6 Test the final co-simulation model and evaluate performance and constraint satisfaction.

3.7 Perform the necessary modification on the models until the desired performances are reached and the constraints are satisfied, as defined in task 0.2.

Modern Computer Aided Control Engineering (CACE) software also allow control engineers to extend the described methodology with an hardware-in-the-loop test. To do this, the engineers build the control hardware defined in task 3.6 and load the code on it. The hardware can be connected to a PC running the CACE software, so the real control hardware can be validated in conjunction with the virtual mechanical model.

The top-down diagram of the product obtained during the application of the methodology is shown in figure 1.

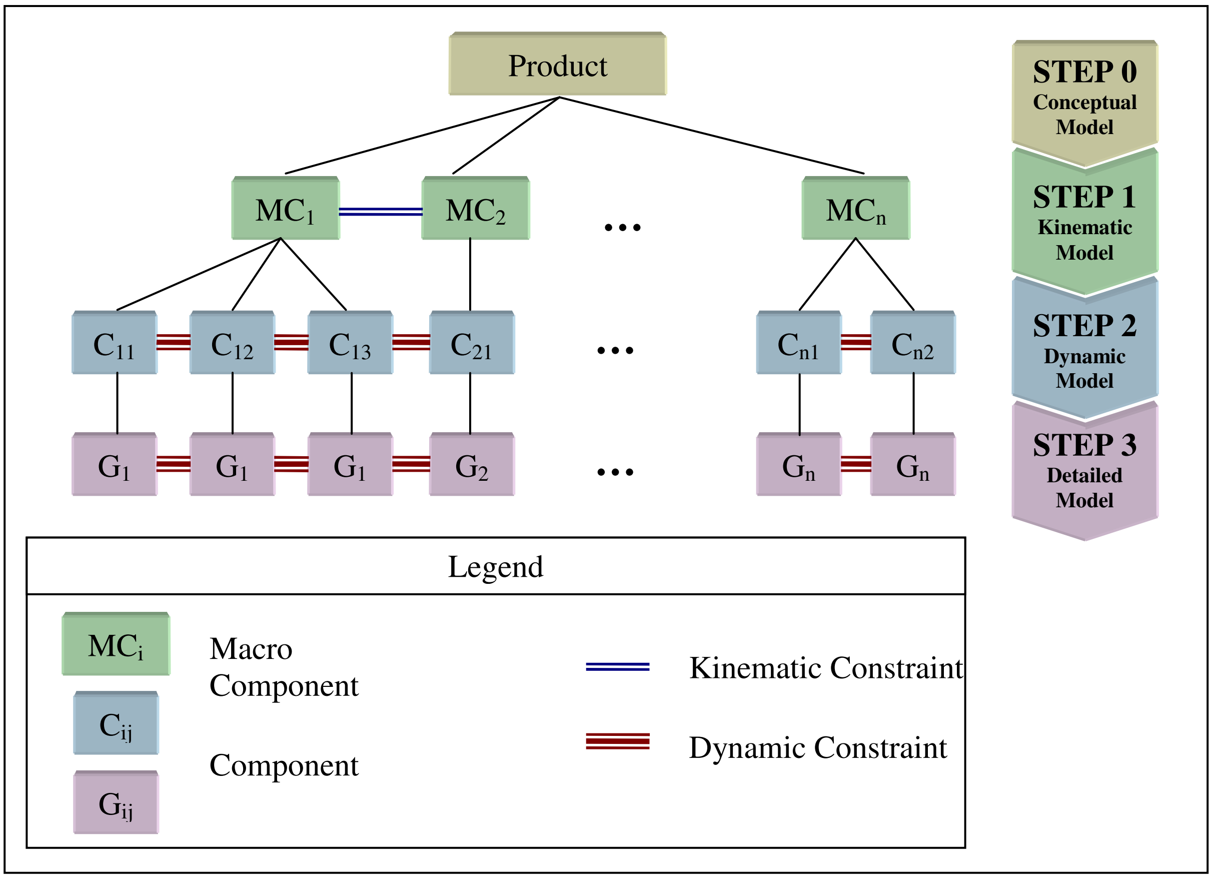


**Figure 31: Schema of top-down product structure obtained applying the methodology proposed**

### *4.3. Benefits and Drawbacks*

The main problems related to the application of this methodology are:

- The need to have a good estimation of the work required by the different specialists that collaborate in the design process. In fact, the realization of a complete functional digital mock-up (DMU) requires that both the mechanical and control models have to be ready for each milestone in which a test of the co-simulation model is needed (tasks: 1.5, 2.4 and 3.7). A delay in the mechanical or control design could create a bottleneck in the product development process.

- The estimation of the physic properties for each macro-component (task 2.1) strongly depends on the information initially preventable and on the experience of the designers.
- Mechanical model must be designed in the CAD environment and then imported in the multi-body solver. During this process, all the constraints and loads will be lost, and must be redefined within the simulation environment. This is actually a time-consuming task.
- On the other hand the advantages offered by this methodology are:
- It is possible to execute a top-down design both on the mechanical and the control side, validating each step with the co-simulation.
- It is possible to reduce the occurrence of design errors in the final steps of the product development process. This is particularly true for those errors related to the misunderstandings between mechanical and control engineers.

When the product has an user interface, this can be sketched, starting from step 1, using the CACE software. So functional and usability tests can be conducted with the participation of the user also in the first stages of the product development process.

Using dedicated software allows the designer to achieve a very high detail in the mechatronic model, both from the mechanical and electronic point of view.

## *4.4. Case study*

In this section the application of the methodology on a case study will be described. Object of the case study is a mobile elevated working platform with articulated arm. The final CAD model of the product is shown in Figure 2.

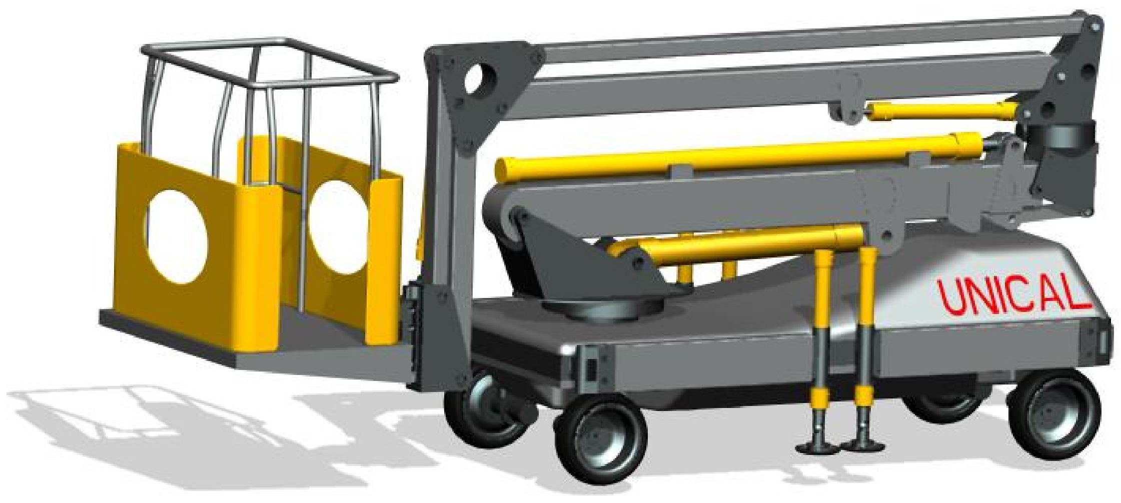


**Figure 32: The working platform analyzed in the case study**

The platform is provided with four driving and steering wheels. A great operational flexibility is guaranteed by four variable-length stabilizer legs. Every leg has 3 d.o.f. (degrees of freedom), that allow multiple support configurations. The articulated arm, conceived to easily cover a large work-volume, has 6 d.o.f.. There are 22 actuators for the movement of the various parts and each of them has a sensor position for the feedback control.

By the co-simulations results, it has been also possible to implement a control-logic that allow the dynamic reconfiguration of the support stabilizers, making use of equilibrium of the whole structure on three of the four contact points with the ground.

According to the methodology explained above, the design of the mechatronic model uses co-simulation like a test tool, in order to validate the design choices and to develop the model on levels of abstraction gradually decreasing. Interactive co-simulation with the controlled machine is realized drawing the higher performances by the integration of the software ADAMS and MATLAB/Simulink (Figure 3.)

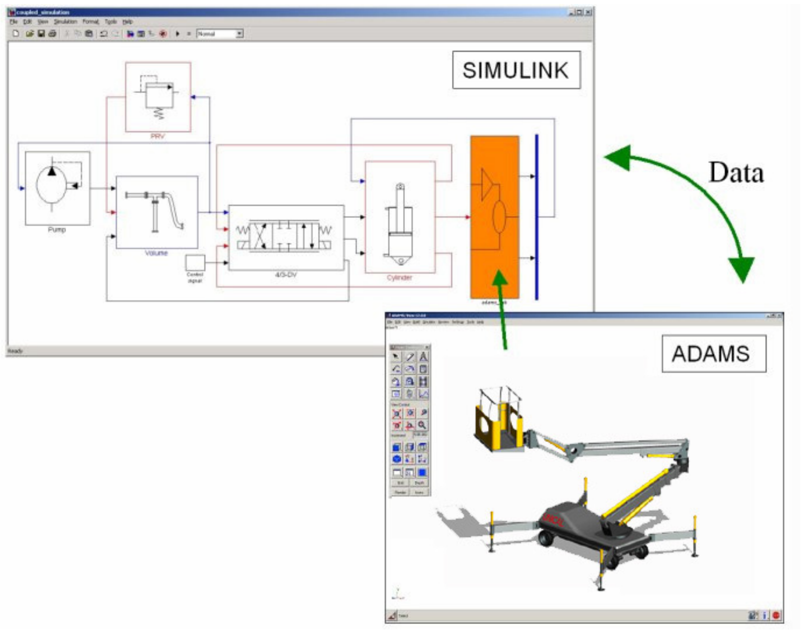


**Figure 33: Software integration for co-simulation**

The interface between the Matlab/Simulink environment and the Adams environment is the building block highlighted in orange. It is generated using Adams/Controls, specifying all the control variables (both input and output) of the mechanical system that the user wants. This variables can be the positions, velocities, pressures and so on. It is called Adams-SubBlock. Actually it is the link between the control electronics developed in Simulink and the mechanical system of the machine modelled in Adams. Simulink can import it as a special building block called S-Function. From the Adams point of view, this block receives forces in input and gives the position of the actuators in output. From the Matlab point of view, the Control Block receives positions in input and returns forces in output.

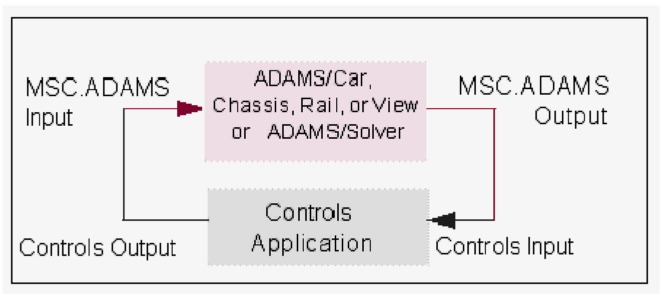


**Figure 34: Inputs and outputs of the interface**

Within this block, all the parameters regarding simulation and communication are defined. It is possible, in fact, specify the time-step of the simulation, communication

During the co-simulation, all the parts that made up the 3D model are moved according to the simulation results.

## Step 0

The design idea is initially conceptualized by the rough definition of the volumes and the functional geometry of the mechanical model, represented in ADAMS by elementary solid elements (parallelepipeds, cylinders, etc.) (Figure 7a). Three fundamental sub-systems can be recognized: basket, articulated arm and base.

## Step 1

The kinematics to match the functional targets of the machine is defined for each macro-component. The articulated arm is disassembled in two main groups. The first one can rotate on two axis, one vertical and the other horizontal, and it is constrained to the base. The second group is made up by two quadrilateral elements, and it is linked to the first group by a joint which allows vertical translation only. The base of the platform can be subdivided into two macro-components: the driving system and the stabilization system. The first is made up by four motors, acting on each wheel. A kinematic mechanism rotating simultaneously the four wheels has been developed for the steering system.

The stabilization system can lift the platform upon four stabilizing legs. In this step only the kinematics of the multibody model is tackled. Therefore, the building of the multi-body model (Figure 7b) is easy and fast because the system dynamics is not considered. At the same time, the user interface is defined. It is made up by two joysticks, by which the user can interact with the virtual model (Figure 4 on the left). In this case study it is not necessary an environment interface (point 1.2 of methodology) because all the inputs of the mechatronic system come only from the user interaction. In the MATLAB/Simulink environment, the control logic is implemented. It manages the use of the joysticks connected to the control variables (position and velocity of each macro-component). These are defined in ADAMS in correspondence of the kinematic constraints. The user can interact and drive the multibody model by the joysticks, according to the degrees of freedom allowed by the constraints.

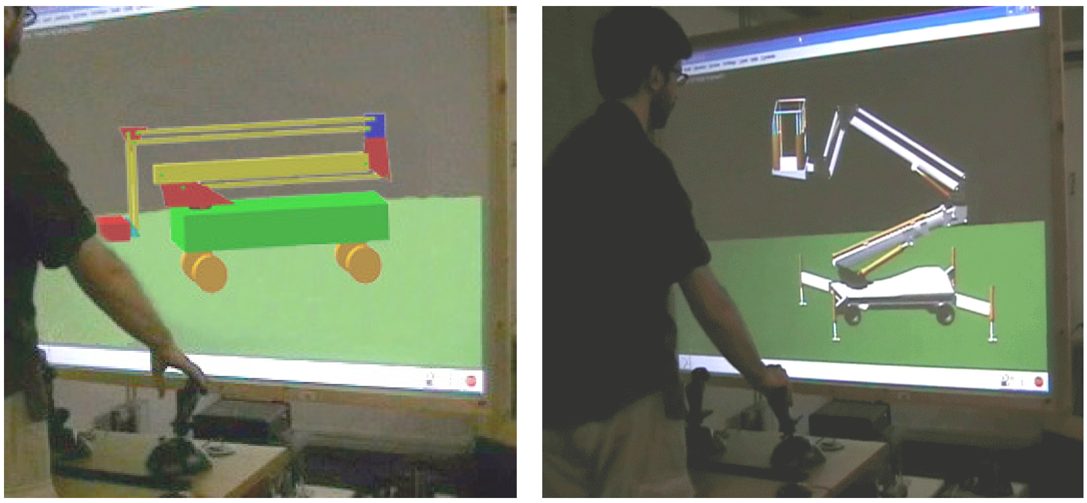


**Figure 35: The methodology proposed allows designers to validate the digital mock-up of the platform interacting with a pair of joystick, starting from step 1 with the kinematic model (left image), until step 3 with the detailed model (right image).**

Co-simulation make possible to test input devices functionality and the kinematics of the mechanical system, evaluating volumes, working-space and flexibility in positioning. One of the design targets is to maximize working-volume against idle volume and, during this phase, it is possible the evaluation of the volumes, of the working-area of the articulated arm and of the idle configuration of the machine. Supported by the visual information that comes from the co-simulation output, the designer can improve kinematic mechanisms finding alternatives to the original schema. Each leg can rotate about 180 degrees respect to the base, and telescopically lengthen itself to confer more stability to the platform. This make possible a flexible functioning of the system, because it is possible to reduce the room required by the platform when a wide action-radius of the arm is no required. Vice versa, when a wide action-radius of the arm is required, every leg can be extended independently. Further, vehicle driving is simulated evaluating the manoeuvrability of the platform.

## Step 2

In this phase the kinematic model is enriched with information and data to obtain a dynamic model (Figure 7c). To evaluate the dynamics of the mechatronic model, it is necessary to define every macro-components. The geometries of the hydraulic pistons used as the actuators for the articulated arm are sketched for every sub-system. Then, a weight estimation of each part occurs. In this way it is possible to evaluate the dynamic behaviour, and further, it is possible to improve the control logic. End-stroke sensors and pressure sensors are placed, therefore it is now possible to develop a coherent control logic, taking in account the stability of the system and the forces and the torques that each actuator should provide. It is easy to understand that the stability of the platform is linked with the dynamic and the position of the articulated arm, therefore the availability of the multi-body model in the co-simulation environment is very useful to achieve this target. The critical condition of the anti-reversal control logic starts when the pressure of one of the legs is zero. When the critical condition is verified, all the manual controls that could worsen the equilibrium are disabled. During this phase the power of our design methodology is highlighted, because it is possible to foresee the behaviour of the whole mechatronic system before the detailed modelling. The aim of the automatic safety control is to grant the platform stability under every load condition with every possible configuration of the legs. Two of the possible working volumes, for two different legs configuration, are showed in Figure 5.

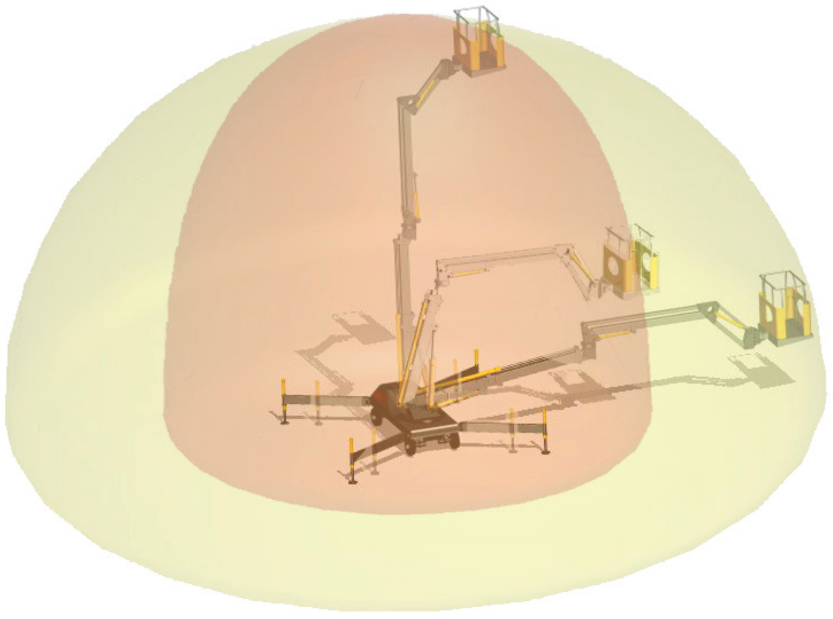


**Figure 36: Two possible working volumes for two different legs configurations.**

The flexibility of the legs allows the platform to adapt itself to several morphological configurations of the terrain. A control logic for a dynamic re-configuration of the legs has been implemented, it works maintaining the equilibrium of the whole system upon three contact points with the terrain. If the centre of gravity is within the area bounded by the three legs, the fourth one can be repositioned. For this reason the control logic make possible to move (only) the unloaded leg.

## Step 3

In this phase each component is defined with a detailed geometry (Figure 7d). Each part is modelled using a CAD software, using as skeleton all the references developed in the previous phases (Figure 6).

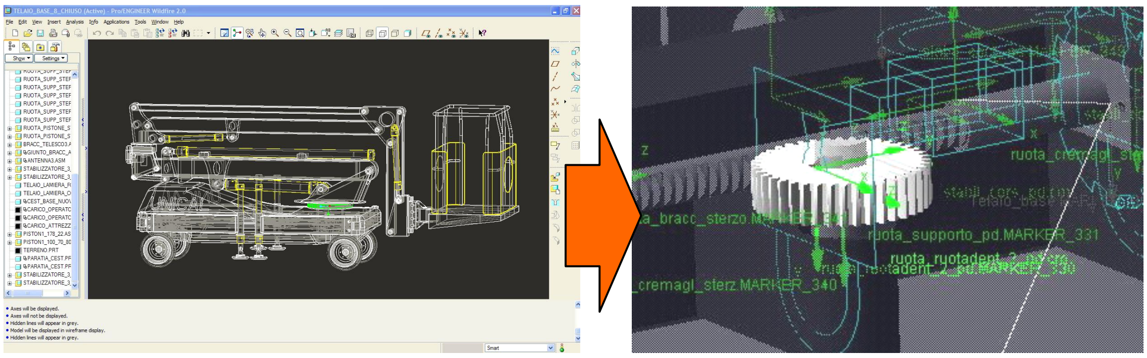
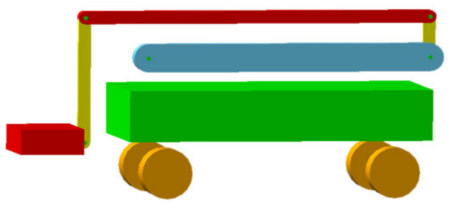
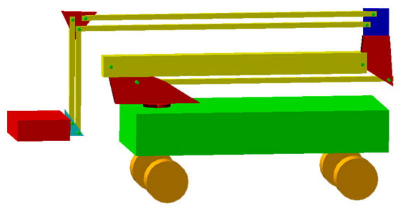
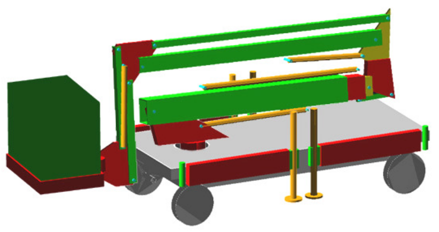


**Figure 37: The detailed geometries modeled in Pro/Engineering are used to build the final multi-body model of the platform in ADAMS.**

Once all the parts are modelled, these are imported in the simulator, and all the mass properties and friction coefficients are automatically calculated by the simulator once the material properties are specified. After the values of the parameters in the Simulink block relative to actuators are updated in accord with the new properties, the final simulation occurs. The complexity of the multi-body model is quite high, and the computational load is not affordable by a mainstream calculator. The several co-simulation performed, highlighted an overall slowness in simulation. Therefore, we chose to split the  functional digital mock-up in two co-simulation models. The first regards just the drive and the lift of the platform, while the second regards just the moving of the articulated arm.


a) Step 0


b) Step 1


c) Step 2

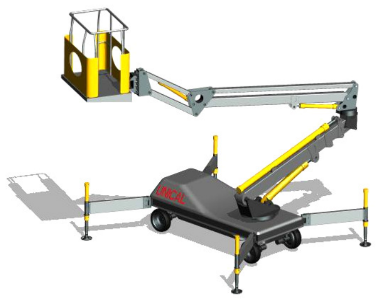

d) Step 3

**Figure 38: The working platform at different steps of the methodology.**

## 4.5. Conclusions

In the mechatronic field co-simulation is usually employed as a tool to improve efficacy and accuracy of the models employed in the validation of the functional digital mock-ups. We think that this approach limits the advantages that engineers can obtain from co-simulation, so we have proposed a methodology that exploits the powerful of co-simulation techniques since the earliest stages of the design process. The main idea is to apply co-simulation in conjunction with the classical top-down design approach. The result is that the mechanical and the control models can be jointly validated on each of the three steps of the methodology described, also when the models have not been completely detailed. In particular, the methodology proposes to abstract the model considering, at the beginning, only the kinematics and later refining it with the dynamics and the final geometries. Also the control model is progressively detailed on the basis of the results obtained on each step.

This approach improve the communication between mechanical and control engineers because they can jointly validate their ideas and assumptions at each milestone of the project, reducing the possibility that design errors occur in the final steps of the product development process.

Moreover, this methodology could be a good support for participatory design, because functional and usability tests can be conducted, with the participation of the user, also in the first stages of the product development process.

The proposed case-study puts in evidence that the methodology is applicable also in complex situations and that the maturity of the co-simulation tools allow designers to develop precise and complete virtual model of the products. The power of the hardware currently available allow also engineers to interact with the virtual model using, when it is possible, an user interface very similar to the real one.

# Chapter 5. An experimental environment for the runtime communication among different solvers and visualization modules

## 5.1. Introduction

With the arrival of mechatronics, the complexity of modern industrial products is increasing. Mechatronic products are the result of the synergy among mechanical engineering, electronics and computer science. As it is easy to understand, this integration among several engineering domains increased the complexity of the final product. The development of such products often takes place in large organizations and needs the involvement of many different disciplines

As the complexity of industrial products keeps increasing, people are seeking design methodologies with higher productivity and performances. Companies are devolving a lot of efforts in research for new design methodologies. To cope with the challenges of system design and complexity handling, Bosch started the initiative ''Systems Engineering Mechatronic''. Targets are high design efficiency (reduction of development time and cost) as well as high design quality (design correctness). New design methodologies and processes are currently established in different business units (automotive equipment, consumer goods and industrial equipment). Systems Engineering Mechatronic is based on the well known V-model for system design (VDI, 2004). The V-model incorporates different levels for customer, system and components.

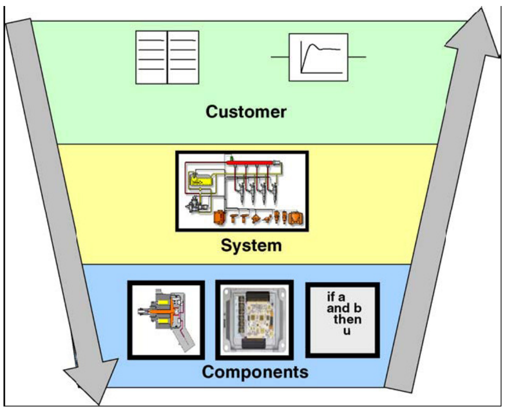


**Figure 39: V-model with customer, system and component level (W. Dieterle, 2005)**

Another study of design methodologies for mechatronic products can be found in Roos and Wikander. In Roos and Wikander's work an overview of the complexity of the design of mechatronic modules is presented and also the authors' approach to develop design and optimization methods for automotive mechatronic modules is described. The presented idea is developing a model based design and optimization methodology for self-contained and physically integrated mechatronic modules. New optimization methods for multi criteria optimization across engineering domains are a necessity to find the best sub-system design for a given set of requirements. The methodology will focus on module design, but it will also be useful for comparing different design concepts. Some requirements engineering also need to be incorporated into the methodology since it is very important that the sub-systems are specified to such a level of detail that multi criteria optimization can be performed.

In most of these new design methodologies there is a decoupling between the specification and the implementation of each sub-system, which lead to an independent development of the different components which finally will be "merged" into the final product.

As it is easy to understand, one of the main targets for the industries is the simulation and the verification of such complex products before the physic prototype realization. Actually there are no many software tools which allows the designer to simulate the whole product before its realisation. The simulation environment, in fact, should be able to simulate more than one sub-system at the same time. Therefore there is the need of a co-simulation environment rather than a standard simulation one. This need is particularly felt in the development of embedded systems. In this field in fact a lot of research in co-simulation techniques has been done, even if focused on the hardware/software co-simulation.

Amory et al. presented a tool for the implementation and evaluation of a hardware and software co-simulation. The co-simulation environment is made up by different simulators, which can be geographically distributed. The communication between simulators is done using a co-simulation backplane. The co-simulation backplane reads a file describing how the modules are connected, automatically launches the simulators and controls the simulation process.

Yang et al. presented a communication infrastructure for an integrated design framework that enables co-design and co-simulation of heterogeneous design components specified at different abstraction levels and in different languages. The core of the approach is to abstract different communication interfaces or protocols to a common high level communication semantics. Designers only need to specify the interfaces of the design components using extended regular expressions; communication adapters can then be automatically generated for the co-simulation or other co-design and co-verification purposes.

None of the papers mentioned above take care about the co-simulation of mechatronic products, but they regards above all the HW/SW co-simulation. However, they are very useful to understand the possibilities and the methods to achieve a correct co-simulation. Most of the efforts in mechatronic co-simulation are focused on the use of existent software simulators and try simulate the overall behaviour using the already available interfaces. For example, in Villec is described how ADAMS/Controls and Xmath are used to simulate a Vehicle Attitude Control (VAC) system requiring 300 states to represent the vehicle model in ADAMS and 30 states to represent the control system in Xmath. Xmath users make changes to the control system and initiate simulations from the familiar

Systembuild environment. Accurate modeling of a fixed frame controller in the discrete time domain is made possible in this cosimulation environment. Interface bandwidth issues between ADAMS and Xmath applications are explored.
Sánchez et al. developed the Easy Java Simulation (Ejs), a software tool designed to create interactive simulations in Java using models created with Simulink. Basically, Ejs is able to communicate with the Simulink model and to tightly control its execution. The communication takes place through a DLL library written in C and some utility Java classes. The main advantage of this work is that Ejs creates Java applets that are independent, multi-platform, which can be visualised using any Web browser, read data across the net and be controlled using scripts from within html pages. Ejs can also be used with Java 3D to create interactive virtual products in 3D, but it has been conceived mainly for educational purposes and it cannot be efficiently integrated into a PDP because Java 3D is not suitable for complex models visualization.
One of the best techniques to exploit the overall appearance and behaviour of the models is Virtual Reality (VR). At the moment it is not possible to evaluate, in VR, models developed in simulation package. One of the few approaches devoted to the integration of simulation package and VR has been presented by Kirner et al. [81] who developed the VR-SIM, an object oriented C++ library, able to incorporate a Real-Time Systems (RTS) simulator with VR technologies. The use of VR-SIM involves the creation of the system to be validated and of a virtual environment related to this system. The case-study is a robot arm coupled to an automatic transport-belt, used in a factory for piling up boxes. This work demonstrates that VR technology is applicable and useful to support RTS simulations, as a form used to evaluate the correctness of such systems. But the VR-SIM is a tool addressed to software engineers responsible for the development of real-time, process control systems, it requires code development for the implementation of the virtual product and it is not suitable to be used by industrial or mechanical engineers in the PDP.

Amor et al. presented an application under the virtual environment OpenMASK where the behavior of a rotary pneumatic jack is simulated. In this work the behavior of the jack has been modeled using an hybrid automata formalism, and the geometry retrieved from CAD files and exported in INVENTOR files. In this work the authors developed their own solvers to obtain the simulation.
During this PhD thesis, the research of an approach or a methodology to achieve a mechatronics co-simulation using several heterogeneous solvers has been carried out. This paper describes the development of an experimental middleware that supports the communication among different synchronously running simulations, solving interrelated problems and integrating a graphical interactive environment to support the interdisciplinary team in the design review and decision taking. The developed framework takes care about the synchronization and the communication of the visualization environment and different numerical solvers. Each solver is a stand-alone application generated by a Matlab/Simulink model. Their execution is scheduled and triggered by the framework. Subsequently to the simulation of the numerical models, the 3D visualization is updated in accord to the results of the different simulated objects. We used the framework to compute the dynamic behaviour of a mechanical system and then animate the 3D model. But the possibilities provided by the framework are still unexploited, because its architecture allows one to perform a wide range of simulations. This framework is well fitted to the simulation of mechatronic systems. Mechatronics system in fact are characterised by a tight coupling of different implementation technologies, e.g. hydraulics, mechanics, pneumatics, electromechanics, electronics and software. With our approach it is quite easy achieve the correct simulation and visualization of a mechatronic system, since the framework supports several simulations at once. It is possible in fact, to “link” the numerical results of a particular simulator to a sub-part of the 3D assembly.

## 5.2. Implementation

The target of this work is a visualization and simulation environment which allows the runtime communication among the visualization framework and different solvers. In this way it is quite simple to obtain a multi-domain simulation. Each solver in fact, can be the specialized solver for a specific domain. This approach, known in literature as *heterogeneous co-simulation*, has the advantage of a faster integration of the simulation model within the overall framework, since no conversion to a common solver is needed. The disadvantage is that it implies the communication among each solver, since it is possible that the output of a solver can be the input for another solver. Further, another issue of this approach, is the need of a scheduler to synchronize the several solvers. To achieve both these tasks we used an open source middleware called OpenMASK. It simplify the creation of a simulation tree, in which each solver acts as a simulated object, i.e. a node of the simulation tree. The simulation tree ends with a simulated object called VisSimulator. This one is the responsible of sending simulated data to the visualization environment.

The visualization framework is IFXOpenSG. This framework has been developed at Fraunhofer IGD, and it is a powerful and extendable post-processor, using OpenSG as scene-graph for visualization. The idea of using a post-processor allows one to visualise data from several numerical simulator: structural, CFD and whatever. It is therefore a general purpose environment, which has both graphics performances and scientific visualization tools.

The overall architecture of the system is represented in the figure below.

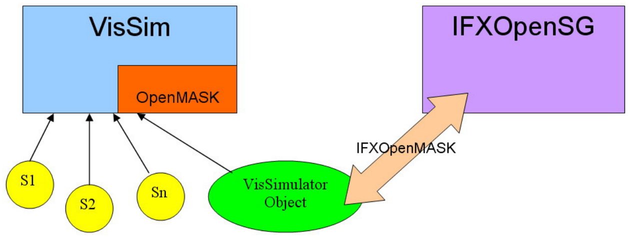


**Figure 40: Overall architecture of the framework**

The actors of this scenario are:

- The VisSim OpenMASK application
- The solvers
- The IFXOpenSG application

They are better explained in the following sections.

## 5.3. The IFXOpenSG application

The IFX framework is a powerful postprocessor developed at Fraunhofer IGD. It uses the data-flow paradigm, i.e. in order to obtain the desired visualization, a visualization pipeline should be created. As it is easy to understand, numerical datasets flow within this pipeline. All the post-processing operations (cross-section, iso-surface etc.), are available as modules. These modules made up the visualization pipeline. Each module execute its function on a dataset in input and puts the results in output. Therefore, each module has its own input and output ports.

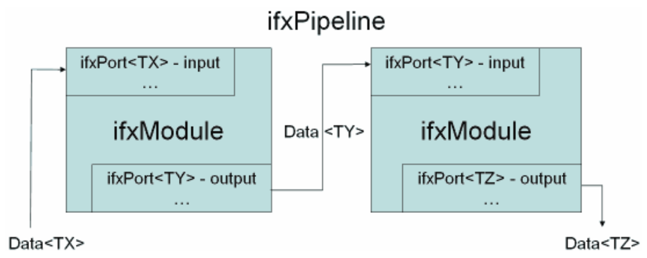


**Figure 41: schematic representation of the IFX modules**

In order to achieve our task we extended the IFX framework with a new module called ifxOpenMASK. This module takes in input port an unstructured mesh and puts in output port the mesh with the transformed vertex. Using a separate thread, in fact, the ifxOpenMASK module receives data from the VisSim application via TCP/IP socket. These data are in the form <*part, operation, value*>, where *part* is the part of the assembly which has to be moved, *operation* is the type of transformation and value is the amount of the transformation. The command < Window, ytrans, -1> means translate of one unit along the negative y axes the part named "Window", i.e. all the vertex belonging to the part named "Window". The

vertex transformation is executed only on the vertex belonging to the part specified in the command. Therefore, to achieve the animation, it is necessary that the 3D assembly has vertex and elements grouped into parts. Usually this is not a problem, since assemblies created using a CAD software are normally subdivided in different parts. Further, in a CAD file quite often a hierarchy is present. It is necessary that the transformation is executed also to all the children parts of the specified part.
For this reason, the IFX framework has been extended to handle the hierarchy, developing the ifxGroupHierarchy class. All the vertex transformation within the ifxUnstructuredMesh3D dataset are made by this object.

The visualization pipeline of the application is showed in the figure below.

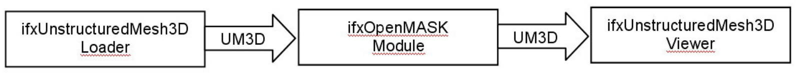


**Figure 42: The IFX visualization pipeline**

The pipeline is read by text-file by the IFXOpenSG application. The user must create a text file in which the pipeline is mapped and all the parameters of the modules are specified. Then,

## *5.4. The solvers*

The basic idea is to use different solvers for the co-simulation. In this work we used several stand alone simulators generated with the Matlab RealTime Workshop toolbox. Real-Time Workshop is an extension of capabilities of Simulink and MATLAB that automatically generates, packages, and compiles source code from Simulink models to create real-time software applications on a variety of systems. By providing a code generation environment for rapid prototyping and deployment, Real-Time Workshop is the foundation for production code generation capabilities. Along with other tools and components from The MathWorks, Real-Time Workshop provides

- Automatic code generation tailored for a variety of target
- Platforms

- A rapid and direct path from system design to implementation
- Seamless integration with MATLAB and Simulink
- A simple graphical user interface
- An open architecture and extensible make process

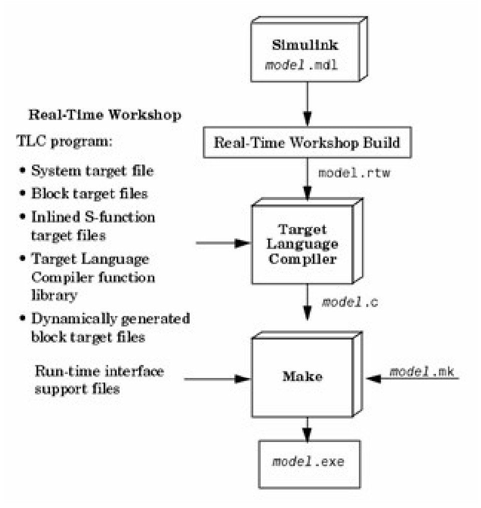


**Figure 43: The process of generating source code from Simulink models using Real-Time Workshop**

In other words, with this powerful tool it is possible to generate the C++ code for a Matlab/Simulink model and run it as a stand alone executable.
In order to achieve the communication between the solver and the VisSim application it has been necessary a modification in the RTW generated code. It is necessary in fact to create a TCP IP communication channel and send data through it whenever a new value has been computed.

## 5.5. *The VisSim application*

The VisSim application is an OpenMASK based application which act as a scheduler among the several solvers and the visualization framework. In the VisSim application the OpenMASK simulation tree is specified. The simulation tree is made up by different simulated objects. They are the basic building block of an OpenMASK application. It is inside a simulated object that all code for the evolution of the object and the communication with other objects is located. The

OpenMASK middleware executes each simulated object at the scheduled time and makes the simulated object communicate each other, in accord with the simulation tree.

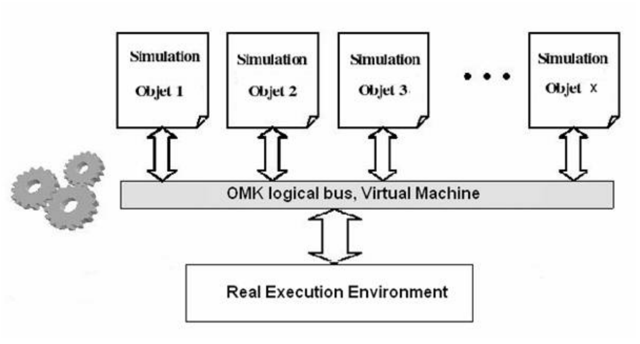


**Figure 44: OpenMASK Execution environment**

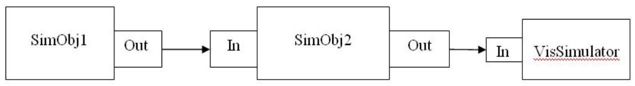


**Figure 45: VisSim simulation tree**

An OpenMASK module has a predefined architecture. Indeed, it consists of a C++ class which can contain one of the following generic methods [4]:

- Init (): This method contains all the operations which must be executed by the controller during the instantiation of the object of simulation (equivalent to the builder in C++).
- Compute(): This method contains all the calculations which will be made with the object of simulation. One can compare this method with a transfer function. In fact, the controller calls this method in every simulation step.
- ProcessEvent() This method contains all the operations necessary for the treatment of events.

In the Init() method all the data connections among the simulated object are specified. In other words, it is specified how simulated flows through the pipeline, i.e. the topology of the simulation tree. In the Compute() method instead, there is a data exchange with the real numerical solver. The simulated object in fact does not perform any calculation, but receives the results via TCP/IP by the application created using Real Time Workshop previously described.

The VisSimulator instead, sends via TCP/IP the numerical results to the IFXOpenSG application. Its “simulation” consists in sending data to the IFXOpenSG application and waiting for the finished visualization message. It is important to notice that in the VisSim application, a simulation tree always ends with a VisSimulator object. Since the VisSimulator object is the last of the chain, the visualization occurs only after the end of the simulation of all the simulated objects.

## 5.6. Test Case

As test case, the simulation and visualization of a power-window is reported. The dynamical properties of the window-door system have been modeled in Matlab/Simulink. They have been modeled in a simple fashion, because an accurate simulation of a power-window is out of the scope of this work. The dynamical properties are the glass acceleration due to the electric motor and the friction between the glass and the door. Moreover, a simple end stroke sensor stops the simulation when the end stroke is reached. In the test case the electric motor and the end stroke sensor are considered as independent solvers, therefore they will be both executed as stand-alone application. Once the end-stroke is reached, the *endstroke* event is sent, and then the simulation is stopped.

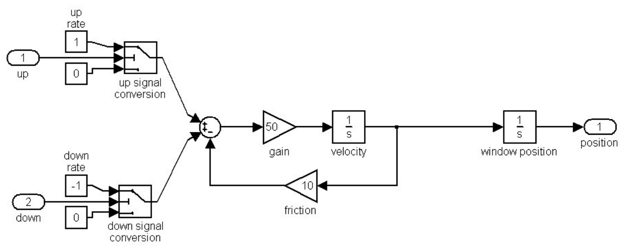


**Figure 46: Dynamical properties of the window-door system**

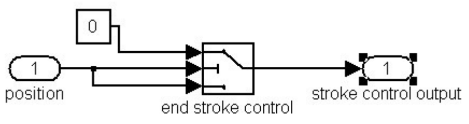


**Figure 47: End stroke sensor control**

The 3D model used for the visualization has been the front left door of a BMW 7er. The simulation pipeline is quite simple, and it is represented in the figure below.

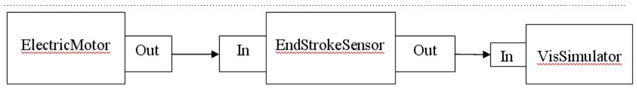


**Figure 48: Simulation Pipeline of the Power Window**

The window rise until the end-stroke sensor stops the simulation.

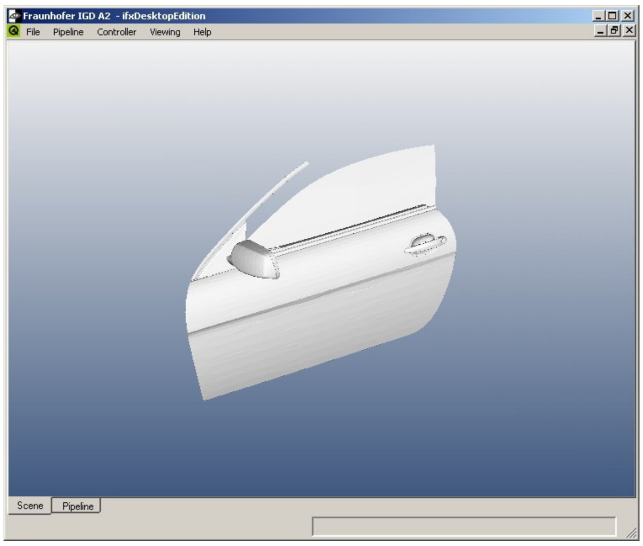


**Figure 49: IFXOpenSG during the simulation**

# Chapter 6. References

| | |
|---|---|
| 1. Bruno F., Caruso F., De Napoli L., Muzzupappa M. | CFD data visualization in augmented reality. In Proceedings of XVII Ingegraf – XV ADM, Sevilla - España, June, 2005 |
| 2. Bruno F., Caruso F., Ferrise F., Muzzupappa M. | VTK4AR: An Object Oriented Framework for Scientific Visualization of CAE Data in Augmented Reality. In Proceedings of Eurographics Italian Chapter, Catania, February, 2006, ISBN 3-905673-58-4 |
| 3. Bova S., Bruno F., Caruso F., Nigro A | Augmented Reality techniques to support CFD data visualization and validation in Proceedings of Virtual Product Development in Automotive Engineering, Munich, Germany, March 2006 |
| 4. Bruno F., Caruso F., De Napoli L., Muzzupappa M. | Visualization of Industrial Engineering Data in Augmented Reality. In Journal of Visualization, 2006, Vol. 9, n. 3, pp. 319-330 (Impact Factor 0.806) ISSN 1343-8875 |
| 5. Bruno F., Caruso F., Falbo L., Muzzupappa M. | A co-simulation based design methodology for mechatronic products. In proceedings of ICED07- International conference on Engineering Design, Paris, August 2007 |
| 6. Barbieri L., Bruno F., Caruso F., Muzzupappa M. | Innovative integration techniques between Virtual Reality systems and CAx tools. To appear in The International Journal of Advanced Manufacturing Technology (Impact Factor 0.418) |
| 7. Bruno F., Caruso F., Muzzupappa M., Stork A. | An experimental environment for the runtime communication among different solvers and visualization modules. To appear in Proceedings of 19th European Modeling and Simulation Symposium, Genova, October 2007 |
| 8. Bruno F., Caruso F., Falbo L., Muzzupappa M | Co-Simulazione meccatronica di una piattaforma di lavoro elevabile con braccio articolato, In Proceedings of XVI ADM - XIX INGEGRAF 2007, Perugia, 6-9 giugno 2007, ISBN 978-884671832-7. |
| 9. Jimeno A, Puerta A (2006) | State of the art of the virtual reality applied to design and manufacturing processes. The International Journal of Advanced Manufacturing Technology, DOI - 10.1007/s00170-006-0534-2 |
| 10. Brooks, FP (1999) | What's real about virtual reality?. Computer Graphics and Applications, IEEE, 19(6):16 – 27 |
| 11. Zhou T, Wu Y, Wang Z (2001) | Implementation of virtual reality techniques in car styling. Automotive Engineering, 23(1): 18-20 |
| 12. Bryson S (1993) | The Distributed Virtual Windtunnel. In SIGGRAPH'93 Course Notes 43: 3.1-3.10 |
| 13. Jayaram S, Connacher HI, Lyons KW (1997) | Virtual assembly using virtual reality techniques. Computer Aided Design 29(8):575-584 |
| 14. Jayaram S, Jayaram U, Wang Y, Tirumali H, Lyons K, Hart P (1999) | VADE: a Virtual Assembly Design Environment. Computer Graphics and Applications, IEEE, 19(6):44 – 50 |
| 15. Liu J., Ning R., Yao J. Wan B. (2006) | Product lifecycle-oriented virtual assembly technology. Frontiers of Mechanical Engineering in China Journal. 1(4):388-395 |
| 16. Gomes de Sá A, Zachmann G (1999) | Virtual reality as a tool for verification of assembly and maintenance processes. Computer & Graphics 23:389-403 |
| 17. Ye N, Banerjee P, Banerjee A, Dech F (1999) | A Comparative Study of Assembly Planning in Traditional and Virtual Environments. Trans. on systems, man, and cybernetics, Part C, IEEE, 29:546–555 |
| 18. Sun H, Hujun B (2002) | Two-handed Assembly with Immersive Task Planning in Virtual Reality. Virtual Reality, 6(4):11–20 |
| 19. Choi ACK, Chan DSK, Yuen AMF (2002) | Application of virtual assembly tools for improving product design. Int. J. Adv. Manuf. Technol., 19:377–383 |

| | |
|---|---|
| 20. Bao J.S., Jin Y., Gu M.Q., Yan J.Q., Ma D.Z | Immersive virtual product development, Journal of Materials Processing Technology 129, pp.592–596, 2002 |
| 21. Bruno F, Luchi ML, Milite A, Monacelli G, Pina M, Sessa F (2006) | Serviceability Analyses in Virtual Environment for the Automotive Industry. In Proceedings of International Design Conference – Design 2006, 15-18 May, Dubrovnik, Croatia |
| 22. Ng FM, Ritchie JM, Simmons JEL, Dewar RG (2000) | Design cable harness assemblies in virtual environments. Journal of Material Processing Technology 107:37-43 |
| 23. Bruno F, Luchi ML, Milite A, Monacelli G, Pina M, Sessa F (2006) | Cabling Design in Virtual Environment. In Proceedings of XVIII Ingegraf, 31 May – 2 June, Barcelona, Spain |
| 24. Monacelli G, Milite A, Sessa F (2004) | An Integrated Approach to Evaluate Engineering Simulations and Ergonomics Aspects of a New Vehicle in a Virtual Environment: Physical and Virtual Correlation Methods. Proceedings of FISITA 2004, 23-27 May, Barcelona, Spain |
| 25. Kirner TG, Kirner C (2005) | Simulation of real-time systems: an object-oriented approach supported by a virtual reality-based tool. In Proceedings of 38th Annual Simulation Symposium, 4-6 April, San Diego, California |
| 26. Bruno F, Mattanò RM, Muzzupappa M, Pina M (2005) | A new approach to participatory design: usability tests in virtual environment. Proceedings of Virtual Concept 2005, 8-10 November, Biarritz, France. |
| 27. | www.ugs.com |
| 28. | www.3ds.com |
| 29. Milgram P., Fumio K.. | A Taxonomy of Mixed Reality Virtual Displays. *IEICE Transactions on Information andSystems E77-D*, 9 (September 1994), 1321-1329 |
| 30. Milgram P., Haruo T., Akira U., Fumio K. | Augmented Reality: A Class of Displays on the Reality-Virtuality Continuum. *SPIE Proceedings volume 2351:Telemanipulator and Telepresence Technologies* (Boston, MA, 31October - 4 November 1994), 282-292 |
| 31. Sutherland I. | "A Head-Mounted Three-Dimensional Display,"Fall Joint Computer Conf., Am. Federation of Information Processing Soc. (AFIPS) Conf. Proc. 33, Thompson Books, Washington, D.C., 1968, pp. 757-764 |
| 32. Azuma R. | A Survey of Augmented Reality," Presence: Teleoperators and Virtual Environments. vol. 6, no. 4, Aug. 1997, pp. 355-385 |
| 33. Kancherla, Anantha R., Jannick P. Rolland, Donna L. Wright, andGrigore Burdea | A Novel Virtual Reality Tool for Teaching Dynamic 3D Anatomy. Proceedings of Computer Vision, VirtualReality, and Robotics in Medicine '95 (CVRMed '95) (Nice, France, 3-6 April 1995), 163-169 |
| 34. Durlach, Nathaniel I. and Anne S. Mavor (editors). | VirtualReality: Scientific and Technological Challenges. (Report of theCommittee on Virtual Reality Research and Development to theNational Research Council) National Academy Press (1995). ISBN 0-309-05135-5 |
| 35. Bajura M, Henry Fuchs, and Ryutarou Ohbuchi. | MergingVirtual Reality with the Real World: Seeing Ultrasound Imagery Within the Patient. Proceedings of SIGGRAPH '92 (Chicago, IL, 26-31 July 1992). In Computer Graphics 26, 2 (July 1992), 203-210 |
| 36. State A., T. Chen, David Chris Tector, Andrew Brandt, HongChen, Ryutarou Ohbuchi, M. Bajura, H Fuchs. | Case Study: Observing a Volume Rendered Fetus within a Pregnant Patient. Proceedings of IEEE Visualization '94 (Washington D.C., 17-21 October 1994), 364-368 |
| 37. State, Andrei, Gentaro Hirota, | Superior Augmented Reality Registration by Integrating Landmark Tracking and Magnetic Tracking.Proceedings of SIGGRAPH '96 |

| | |
|---|---|
| David T. Chen, Bill Garrett, and Mark Livingston. | (New Orleans, LA, 4-9 August 1996), 429-438 |
| 38. Grimson, W., T. Lozano-Pérez, W. Wells, G. Ettinger, S. Whiteand R. Kikinis | . An Automatic Registration Method for Frameless Stereotaxy, Image Guided Surgery, and Enhanced Reality Visualization. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (Los Alamitos, CA, June 1994), 430-436. |
| 39. Grimson, W.E.L., G.J. Ettinger, S.J. White, P.L. Gleason, T.Lozano-Pérez, W.M. Wells III, and R. Kikinis | Evaluating and Validating an Automated Registration System for Enhanced Reality Visualization in Surgery. Proceedings of Computer Vision, Virtual Reality, and Robotics in Medicine '95 (CVRMed '95)(Nice, France, 3-6 April 1995), 3-12 |
| 40. Mellor, J. P. | Enhanced Reality Visualization in a SurgicalEnvironment. MS Thesis, Department of Electrical Engineering, MIT (13 January 1995) |
| 41. Mellor, J.P. | Realtime Camera Calibration for Enhanced RealityVisualization. Proceedings of Computer Vision, Virtual Reality,and Robotics in Medicine '95 (CVRMed '95) (Nice, France, 3-6 April 1995), 471-475 |
| 42. Lorensen W., Cline H., Nafis C., Kikinis R., Altobelli D., and Gleason L. | . Enhancing Reality in the Operating Room. Proceedings of Visualization '93 (Los Alamitos, CA, October 1993), 410-415 |
| 43. Betting F., Feldmar J., Ayache N., Devernay F.. | A New Framework for Fusing Stereo Images with Volumetric Medical Images. Proceedings of Computer Vision, Virtual Reality, and Robotics in Medicine '95 (CVRMed '95) (Nice, France, 3-6 April 1995), 30-39. |
| 44. Edwards, P.J., D.L.G. Hill, D.J. Hawkes, R. Spink, A.C.F.Colchester, A. Strong, Gleeson M.. | Neurosurgical Guidance Using the Stereo Microscope. Proceedings of Computer Vision, Virtual Reality, and Robotics in Medicine '95 (CVRMed '95) (Nice, France, 3-6 April 1995), 555-564 |
| 45. Taubes, G. | Surgery in Cyberspace. Discover 15, 12 (December 1994), 84-94. |
| 46. Feiner, Steven, Blair MacIntyre, and Dorée Seligmann | Knowledge-based Augmented Reality. Communications of the ACM 36, 7 (July 1993), 52-62 |
| 47. Caudell, Thomas P. and David W. Mizell | Augmented Reality: An Application of Heads-Up Display Technology to Manual Manufacturing Processes. Proceedings of Hawaii InternationalConference on System Sciences (January 1992), 659-669. |
| 48. Janin, Adam L., David W. Mizell, and Thomas P. Caudell. | Calibration of Head-Mounted Displays for Augmented Reality Applications. Proceedings of IEEE VRAIS '93 (Seattle, WA, 18-22 September 1993), 246-255 |
| 49. Sims, Dave | New Realities in Aircraft Design and Manufacture. IEEE Computer Graphics and Applications 14, 2 (March 1994), 91 |
| 50. | http://esto.sysplan.com/ESTO/Displays/HMDTDS/Factsheets/Boeing.html (July 1994). |
| 51. Tuceryan, Mihran, Douglas S. Greer, Ross T. Whitaker, David Breen, Chris Crampton, Eric Rose, and Klaus H. Ahlers | Calibration Requirements and Procedures for Augmented Reality. IEEE Transactions on Visualization and Computer Graphics 1, 3 (September 1995), 255-273 |
| 52. Fitzmaurice, G. | Situated Information Spaces: Spatially Aware Palmtop Computers. CACM 36, 7 (July 1993), 38-49 |

| | |
|---|---|
| 53. Rekimoto, Jun, and Katashi Nagao | The World Through the Computer: Computer Augmented Interaction with Real World Environments. Proceedings of UIST '95 (Pittsburgh, PA, 14-17 November 1995), 29-36 |
| 54. Rekimoto, Jun. | The Magnifying Glass Approach to Augmented Reality Systems. Proceedings of ICAT '95 (Makuhari Messe, Chiba, Japan, 20-22 November 1995) |
| 55. Rose, Eric, David Breen, Klaus Ahlers, Chris Crampton, MihranTuceryan, Ross Whitaker, and Douglas Greer | . Annotating Real- World Objects Using Augmented Reality. Proceedings of Computer Graphics International '95 (Leeds, UK, 25-30 June 1995), 357-370 |
| 56. Feiner, Steven, Blair MacIntyre, Marcus Haupt, and Eliot Solomon | . Windows on the World: 2D Windows for 3D Augmented Reality. Proceedings of UIST '93 (Atlanta, GA, 3-5 November 1993), 145-155 |
| 57. Milgram, Paul, David Drascic, Julius J. Grodski, Anu Restogi, Shumin Zhai, and Chin Zhou. | Merging Real and Virtual Worlds. Proceedings of IMAGINA '95 (Monte Carlo, 1-3 February 1995), 218-230 |
| 58. Drascic, D., J.J. Grodski, P. Milgram, K. Ruffo, P. Wong, and S.Zhai. | ARGOS: A Display System for Augmenting Reality. VideoProceedings of INTERCHI '93: Human Factors in ComputingSystems (Amsterdam, the Netherlands, 24-29 April 1993). Also inACM SIGGRAPH Technical Video Review, Volume 88.Extended abstract in Proceedings of INTERCHI '93, 521 |
| 59. Szalavari, Zs., Schmalstieg, D., Fuhrmann, A., Gervautz, M. | Studierstube - An Environment for Collaboration in Augmented Reality, Virtual Reality - Systems, Development and Applications, 3-1, (1998), 37-49. |
| 60. Schmalstieg, D., Fuhrmann, A., Hesina, G., Szalavari, Zs., Encarnação, L. M., Gervautz, M., Purgathofer, W | . The Studierstube Augmented Reality Project PRESENCE - Teleoperators and Virtual Environments, 11- 1, (2002-2) 32-54. |
| 61. Fuhrmann, A., Löffelmann, H., Schmalstieg, D., Gervautz, M. | Collaborative Visualization in Augmented Reality, IEEE Computer Graphics & Applications, 18-4, (1998) 54-59. |
| 62. Barakonyi, I., Fahmy, T., Schmalstieg, D., Kosina, K. | Collaborative Work with Volumetric Data Using Augmented Reality Videoconferencing, Proc. IEEE and ACM ISMAR, (Tokyo), (2003-10). |
| 63. Rekimoto, J. | NaviCam: A Magnifying Glass Approach to Augmented Reality, Presence: Teleoperators and Virtual Environments, Vol. 6-4, (1997-8) 399-412. |
| 64. Rekimoto, J. | International Conference on Artificial Reality and Tele-Existence '95 / Conference on Virtual Reality Software and Technology (ICAT/VRST '95), (1995-11) |
| 65. Abawi, F, Dörner, R, Haller, M., Zauner, J. | Efficient Mixed Reality Application Development, CVMP, (2004-3) , 289-294 |
| 66. Hinckley, K., Pausch, R., Goble, J. C., Kassell, N.F. | A Survey of Design Issues in Spatial Input, Proc. ACM Symposium on User Interface Software and Technology, (1994), 213-222. |

| | |
|---|---|
| 67. Badler, N., Manoochehri, K., Baraff, D. | Multi-Dimensional Input Techniques and Articulated Figure Positioning by Multiple Constraints, Proc. ACM Workshop on Interactive 3D Graphics, (1986), 151-170. |
| 68. Grimson, W., Ettinger, G., Kapur, T., Leventon, M., Wells, W., Kikinis, R. | Utilizing segmented MRI data in image guided surgery, International Journal of Pattern Recognition and Artificial Intelligence, 11-8, (1998), 1367-1397. |
| 69. Sauer, F., Khamene, A., Bascle, B., Schimmang, L., Wenzel, F. and Vogt, S. | Augmented reality visualization of ultrasound images: System description, calibration and features, Proc. IEEE and ACM International Symposium on Augmented Reality, New York, USA, (2001-10). |
| 70. Breen, D.E., Whitaker, R.T., Rose, E., Tuceryan, M. | Interactive Occlusion and Automatic Object Placement for Augmented Reality, Computer Graphics Forum, 15-3, (1996). |
| 71. Fuhrmann, A., Hesian, G., Faure, F., Gervautz, M. | Occlusion in collaborative augmented environments, Computers and graphics 23-6, (1999-12), 809-819. |
| 72. Berger, M.O. | Resolving occlusion in augmented reality: a contour based approach without 3D reconstruction, Proc. IEEE Conference on Computer Vision and Pattern Recognition, (San Juan), Puerto Rico, (1997-6). |
| 73. Lepetit, V., Berger, M.O. | Handling Occlusion in Augmented Reality Systems: A Semi-Automatic Method, Proc. IEEE and ACM International Symposium on Augmented Reality, Munich, Germany, (2000-10). |
| 74. Wloka, M. M., Anderson, B. G. | Resolving occlusion in augmented reality, Proc. Symposium on Interactive 3D graphics, Monterey, California, 1995. |
| 75. Filice L., Ambrogio G., Muzzupappa M., De Napoli L., Fratini L., Costantino I. | Influence of some relevant process parameters on the dimensional accuracy in incremental forming: a numerical and experimental investigation. Journal of Materials Processing Technology, 153-1, (2004), 501-507. |
| 76. Yang, W.J., | Handbook of Flow Visualization, Hemisphere Publishing. New York, (1989). |
| 77. Mébarki Y., Cooper K.R., | Reichert T.M., Automotive Testing Using Pressure-Sensitive Paint, Journal of Visualization, 6-4, (2003) 381-393. |
| 78. Machacek, M., Rosgenm, T | . Development of a quantitative flow visualization tool for application in industrial wind tunnels. Proc. International Congress on Instrumentation in Aerospace Simulation Facilities, Cleveland, OH, USA, (2001-8). |
| 79. Jimeno A, Puerta A(2006) | State of the art of the virtual reality applied to design and manufacturing processes. The International Journal of Advanced Manufacturing Technology, DOI - 10.1007/s00170-006-0534-2 |
| 80. Brooks, FP (1999) | What's real about virtual reality?. Computer Graphics and Applications, IEEE, 19(6):16 – 27 |
| 81. Kirner TG, Kirner C (2005) | Simulation of real-time systems: an object-oriented approach supported by a virtual reality-based tool. In Proceedings of 38th Annual Simulation Symposium, 4-6 April, San Diego, California |
| 82. Bruno F, Mattanò RM, Muzzupappa M, Pina M (2005) | A new approach to participatory design: usability tests in virtual environment. Proceedings of Virtual Concept 2005, 8-10 November, Biarritz, France. |
| 83. Sánchez J, Esquembre F, Martín C, Dormido S, Dormido-Canto S, Canto RD, Pastor R, Urquía A (2005) | Easy java simulations: An open-source tool to develop interactive virtual laboratories using MATLAB/Simulink. International Journal of Engineering Education, 21(5):798-813. |

| | |
|---|---|
| 84. | http://fem.um.es/Ejs |
| 85. Isermann R., | Mechatronic systems: Fundamentals. 2003 (Springer-Verlag) |
| 86. Wikander J., Törngren M. , Hanson M., | Science and education of mechatronic engineering. IEEE Robotics & Automation Magazine. |
| 87. Bracewell R.H. and Sharpe. J.E.E. | Functional descriptions used in computer support for qualitative scheme generation - "Schemebuilder" - AI EDAM Journal. Special Issue: Representing Functionality in Design, volume 10, 1996 pages 333-346. |
| 88. Bracewell R.H., Bradley D.A., Chaplin R. V., Langdon P.M., Sharpe and J.E.E. | Schemebuilder, a design aid for the conceptual stages of product design. International conference on engineering design. Iced '93. The hague, august 17-19, 1993. |
| 89. Li Q., Zhang J., Chen L., | Design for Control IEEE/ASME TRANSACTIONS ON MECHATRONICS, Vol. 6, NO. 2, June 2001 |
| 90. Dieterle W. | Mechatronic systems: Automotive applications and modern design methodologies. Annual Reviews in Control. Vol. 29, 2005, page 273-277 |
| 91. Boehm B. W. | .A Spiral Model of Software Development and Enhancement, Computer, Vol. 21, No. 5, pp. 61-72, 1988. |
| 92. Behbahani S., de Silva C. W., | A New Multi-Criteria Mechatronic Design Methodology Using Niching Genetic Algorithm. IEEE Congress on Evolutionary Computation. Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada July 16-21, 2006 |
| 93. Cho D. H., Jung H. K., Lee C. G., | Induction motor design for electric vehicle using a niching enetic algorithm IEEE Transaction on Industry Applications, Vol. 37, No. 4, pp. 994-999, July/August 2001. |
| 94. Kim J. K., Cho D. H., Jung H. K., C. G. Lee, | Niching genetic algorithm adopting restricted competition selection combined with pattern search method. IEEE Transaction on Magnetics, Vol. 38, No. 2, pp. 1001-1004, Mar. 2002. |
| 95. Himeno M., Himeno R. | The niching method for obtaining global optima and local optima in multimodal functions. Systems and Computers in Japan, Vol. 34, No. 11, pp. 30-42, 2003. |
| 96. Sakka S., Coiffet P., | Contribution to Design of Complex Mechatronic Systems. An Approach through Evolutionary Optimization. Journal of Intelligent and Robotic Systems. Volume 42 , Issue 1 Pages: 1 - 25 January 2005 |
| 97. Ban, A. and Mayer, H. | PS Natural Computation Experimental analysis of evolutionary algorithms using chromosomes of variable length 2003. |
| 98. Roos F., Wikander J., | Mechatronics design and optimisation methodology Mekatronikmöte 2003, Gothenburg, Sweden, August 2003 |
| 99. Villec G. N., | Cosimulation of an Automotive Control Systemusing ADAMS and Xmath. International ADAMS User Conference 1998 |
| 100.Liu C.S., Monkaba V. , Tan H., McKenzie C., Lee H., SuoSae S., | Driveline Torque-Bias Management Modeling for Vehicle Stability Control. SAE Automotive Dynamics and Stability Conference and Exhibition Detroit, Michigan May 7-9, 2002 |