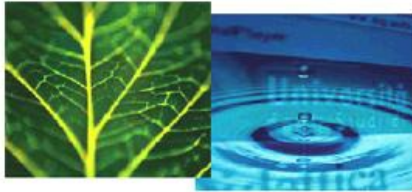


PhD Dissertation



**International Doctorate School in Information and
Communication Technologies**

DIT - University of Trento

GRAPH-BASED MULTIMODAL INTERACTION FOR DESIGN REVIEW IN VIRTUAL AND AUGMENTED REALITY

Martin Witzel

Advisors: Dr. Raffaele de Amicis
Dr. Giuseppe Conti
Fondazione Graphitech

December 2007

“Configure once, interact in any way.”

Giuseppe Conti

Acknowledgements

I would like to thank Dr. Raffaele de Amicis for giving me the opportunity and the incentives to pursue a PhD study in Italy. I would especially like to express my gratitude to my advisor Dr. Giuseppe Conti for his constant support, motivation and fruitful discussions during my PhD. I wish him and Giuliana all the best with the nicest and most important deliverable one could think of. Furthermore, I would like to thank Michele Andreolli for his dedicated and enjoyable collaboration with the communication backbone and the graphical design. Thanks also to my colleagues in Germany, Italy, Scotland, Portugal and Spain. It has been a great and formative experience for me. Many thanks also to Ben Discoe for helping with the porting of the Virtual Terrain Project to OpenSG.

Finally, I dedicate this work to my parents Norbert and Ursula, to my grandmother Anna and to my brother Stefan who have supported me throughout these years with exceptional energy and a great amount of understanding in all situations.

December 2007

Abstract

Much effort has been spent during the last years in providing environments dedicated for virtual design reviews. However, many approaches are very limited with respect to the range of used input devices and more importantly, in the way the users are enabled to interact with the product and the system itself. Specifically in the field of design reviews, the reviewing group consists usually of users with a very heterogeneous professional background having potentially as many different interaction preferences. This problem is further complicated when applications have to address multiple scenarios and hardware configurations. These issues were subject of research of this PhD and have led to the development of a methodology characterized by elevated usability based on the definition of a user-centered multimodal interface. This is done by introducing an interaction graph which can be modified by each user, through a user-friendly authoring tool, to create his/her favorite multi-modal discourse within the application he/she works with. Most notably this approach is both application and domain independent and it can be therefore generally applied to any context, far beyond the scope of virtual reality applications. However the very interactive nature of virtual and augmented reality applications has represented an ideal test bed to validate such an innovative approach. Further, for a thorough review of a product it is often required to examine it under real conditions in an outdoor setting. In the case of automotive design review, this could be placing the virtual prototype next to a physically existing design study while in an architectural scenario, the prototype of a construction could be examined in relation to the building surroundings and real or virtual terrain cultures.

Keywords:

Multimodal Interaction, Virtual and Augmented Reality, User-Centered Design, Distributed Systems, Geographic Information Systems

Contents

ABSTRACT..... VII

CONTENTS.....IX

LIST OF FIGURES XIII

1. INTRODUCTION..... 1

 1.1. MOTIVATION 1

 1.2. STRUCTURE OF THE THESIS 4

2. STATE OF THE ART 7

 2.1. DESIGN REVIEW REQUIREMENTS 7

 2.2. ANNOTATIONS..... 11

 2.3. MULTIMODAL INTERACTION 12

 2.4. INTERACTION WITH PORTABLE AR SYSTEMS AND SET-UPS 15

 2.5. GIS AND SENSORS..... 16

3. APPLICATION FRAMEWORK 19

 3.1. INTRODUCTION 19

 3.2. STANDARD AICI 20

 3.3. EXTENSION OF AICI..... 22

 3.4. STANDARD VIRTUAL TERRAIN PROJECT 29

 3.5. SCENEGRAPH ARCHITECTURE OF THE VTP AND INTEGRATION... 30

 3.6. EXTENSIONS TO THE VTP..... 31

 3.7. CONCLUSIONS 32

4. MULTIMODAL INTERACTION TECHNIQUES.....35

4.1. INTRODUCTION35

4.2. THE MULTIMODAL INTERACTION DIALOGUE35

4.3. DEFINITION OF COMMANDS36

4.4. USE OF MULTIMODAL COMMANDS38

4.5. THE INTERACTION GRAPH (IG).....39

4.6. SOME EXAMPLES47

4.7. MULTIMODAL INTEGRATION USING FEATURE SET51

4.8. SPEECH53

4.9. GESTURES55

4.10. DIALOGS AND HIERARCHICAL RING MENU56

4.11. THE NEED FOR APPLICATION FEEDBACK AND SUPPORT61

4.12. NAVIGATION TECHNIQUES65

4.13. CONCLUSIONS73

5. COLLABORATION AND DATA INTEGRATION75

5.1. INTRODUCTION75

5.2. SENSORS75

5.3. COLLABORATIVE ANNOTATION AND NAVIGATION86

5.4. DATA INTEGRATION91

5.5. CONCLUSIONS94

6. AUGMENTED REALITY FOR LARGE ENVIRONMENTS....95

6.1. INTRODUCTION95

6.2. LARGE AREA SURVEILLANCE96

6.3. ALIGNING VIRTUAL AND REAL WORLD.....98

6.4. CAMERA CALIBRATION100

6.5. HYBRID APPROACH (SUPERVISED AR).....	101
6.6. SUPERIMPOSITION OF TERRAIN CULTURES	102
6.7. AUTOMOTIVE REVIEW UNDER REAL CONDITIONS	105
6.8. CONCLUSIONS	106
7. ASSESSMENT AND VALIDATION	107
7.1. TESTERS	107
7.2. PROCESS.....	107
7.3. DATA ANALYSIS	111
7.4. EVALUATION METHODS	112
7.5. TEST RESULTS	114
7.6. QUESTIONNAIRE ANALYSIS	117
7.7. CONCLUSIONS	141
8. CONCLUSION AND PERSPECTIVE	143
8.1. CONCLUSIONS	143
8.2. PERSPECTIVE.....	146
BIBLIOGRAPHY	149
APPENDIX A. TEST-BED CONFIGURATION	163
APPENDIX B. HW CONFIGURATION(S).....	181
APPENDIX C. LIST OF PUBLICATIONS	185

List of figures

Figure 1. Product development process8

Figure 2. The actual process followed by Fiat styling designers. ...9

Figure 3. Virtual analysis9

Figure 4. AICI scenegraph architecture21

Figure 5. Event handling mechanism.....22

Figure 6. Services for data integration23

Figure 7. GPS receiver24

Figure 8. XSens MTi25

Figure 9. High level system architecture.....29

Figure 10. Tiled terrain with the VTP (OpenSG)30

Figure 11. Extension to the VTP.32

Figure 12. building, create-query sequence.....37

Figure 13. create-query, building sequence.....37

Figure 14. First set of normalized keywords39

Figure 15. Second set of normalized keywords39

Figure 16. Second normalization of keywords39

Figure 17. First graph.....40

Figure 18. Support for iterative commands41

Figure 19. A more elaborate interaction graph.....42

Figure 20. Combinations of modalities.....43

Figure 21. Edge attributes define modalities44

Figure 22. Grabbing a view.....47

List of figures

Figure 23. Taking a visual bookmark	48
Figure 24. Restoring a visual bookmark	48
Figure 25. Modality-based creation of a building.....	49
Figure 26. Input processing and command invocation	52
Figure 27. CFG for hierarchical navigation.	54
Figure 28. CFG for direct access.	54
Figure 29. Ring menu at top level	57
Figure 30. and at creation node	57
Figure 31. Tap-n-hold and Quick-drag activation	58
Figure 32. Consistent interfacing with the interaction graph	60
Figure 33. Gesture and speech helpers.....	62
Figure 34. Graph editing with GraphViz	63
Figure 35. Graph modification with Dynagraph.....	64
Figure 36. Editing of the ring menu inside CEGUI.....	65
Figure 37. World in miniature	66
Figure 38. Mapping to navigation schemes.....	67
Figure 39. Two handed input and change of navigation	68
Figure 40. Test of “Superhand” interaction technique.....	70
Figure 41. Ellipsoid navigation	71
Figure 42. Two-handed interaction with touchscreen and tablet.....	72
Figure 43. Sensor containers	76
Figure 44. Sensor class hierarchy	77
Figure 45. A view of a sensor generated from a web source.....	78
Figure 46. Update of values	78

Figure 47. Dataflow	79
Figure 48. Managed data pipeline.....	82
Figure 49. Sensor boundary creation..	83
Figure 50. SensorBuilder station and domain assignment	84
Figure 51. Managed sensor stations.....	86
Figure 52. The annotation tool.	88
Figure 53. A view of the embedded email client.....	89
Figure 54. Interaction graph for GIS scenario.....	92
Figure 55. An example of a GIS scene	93
Figure 56. Captured video stream and overlaid heightfield	99
Figure 57. Camera calibration tool	100
Figure 58. A user working with the AR setup.....	101
Figure 59. Superimposition of WFS features.....	104
Figure 60. Car in a real environment	105
Figure 61. An example of interaction graph.....	109
Figure 62. Staff from Graphitech showing functionalities	110
Figure 63. Users commenting on features.....	111
Figure 64. Assessment scale.....	114
Figure 66. The hardware configuration used for the test.....	115
Figure 67. The XSens Motiontracker setup.....	116
Figure 68. The XSens Motiontracker used for navigation	116
Figure 69. Automotive design review, system setup.	181
Figure 70. System setup	182
Figure 71. Augmented GIS setup	183

1. Introduction

1.1. Motivation

Virtual design reviews are currently extensively used within automotive industries. This is because in the initial phase, not only one design proposal is examined, but multiple variations are elaborated until one design is finally chosen. Virtual design reviews are specifically used in the development phase to discuss and scrutinize styling designs, the placement of components and their properties. For this reason, the reviewing of a prototype is characterized by a vivid discussion in a group of collaborating engineers and designers. Traditionally the reviewing process is performed by a panel of experts standing in front of a large display while only one user can actually interact with the virtual prototype itself. An efficient support for communication with the model and in turn with the team members is therefore needed. This outlines that multiple instances of the application and in fact distributed clients have to run simultaneously, giving each participant the possibility to interact with the model and the design review application.

Current industrial systems are often implemented exclusively for single user and single modality interaction. Offering multiple modalities to access embedded functionalities ameliorate the users effectiveness when working with the system. Although multimodal techniques try to activate larger parts of the conceptual bandwidth, they are limited in adapting to the way each user would like to access the application's functionality. Hence, the interaction requirements of users with different professional profiles (engineer, designer, etc.) in relation to the tasks have to be examined in detail.

Using multiple modalities further allows accessing functionality more efficiently when their respective strengths are exploited. For instance, speech enables fast and direct input while gestures allow specifying geometries and sketches with ease. Recognition-based interfaces such as speech and gesture subsystems require handling the continuous flow of input in such way that the users actually perform a natural dialogue with the system. Current multimodal interfaces lack the opportunity to align the user preferences properly to their needs and preferences, taking into account well acknowledged interaction paradigms or conventions.

Interactions are often associated by the users with a certain behavior of the application because they subliminally associate analogous (personal) paradigms with the technical workflow. Therefore a new approach to customize multimodal interaction should be developed which takes into account the clear separation of the application behavior from the interfacing user interactions.

The interaction metaphors should seamlessly integrate with the traditional tasks at hand. It is obvious that the users should be supported at all times when using new multimodal interaction schemes, because they are provided with a previously unknown degree of freedom. While the tasks at hand can be complex, interactions should be clear and simple to always keep the goal of the work in focus. This leads us to a further requirement: user interactions should be specifiable in patterns (building blocks) and thus made reusable in other scenarios. As we will show in the next chapters, the solution to this problem should be independent from the technical domain.

On the interaction design level, we propose a novel paradigm customizing user interaction for multiple modalities using a graph-based approach. The user is enabled to design interaction patterns using a graphical authoring tool. The so called interaction graph is persistent and thus it can be re-used and applied in other graph enabled applications. It should be noted that of course it is

necessary to find away how to activate specific functions within the application itself using the stored graph. We achieve this by attaching attributes to functionalities which are meant to be used by our approach. The functions are triggered when the attributes match the user's stage of dialogue with the system. To validate our proposed approach, we will show its applicability to two distinct design review scenarios: virtual automotive design review under real industrial conditions and hybrid outdoor large-area VR/AR environment.

Referring to the latter scenario, design review in an outdoor setting is usually performed on an area with several kilometers. A severe restriction of the user would be to limit his position to his GPS-determined location, since there might be areas which are occluded by objects or terrain topology and thus not accessible without moving physically close to the location of interest. It is often required that information is seen in a larger context. For this reason, we allow the user to temporarily leave the AR mode. He/she is then enabled to use specific terrain navigation metaphors such as flying on the terrain. Both VR and AR mode allow interacting with the scene equally. He/she might then decide if it is necessary to scrutinize the object of interest in AR by relocating to this location. It should be pointed out that the user experiences the product in a virtual reproduction of the scene, with only the related content shown, thus permitting an outdoor session with the purpose of reviewing a building which is planned to be constructed.

The requirements on the application level are therefore challenging in multiple ways. On the application level, it has to be examined how to extend an existing VR framework with video see through capabilities and location-based services (LBS). We intend to undertake a architectural design review on a large area scale in order to embed and superimpose virtual geo-referenced maps, objects and feature sets on the real landscape and estates.

Current optical tracking systems lack adequate means to embed dynamic data stemming from sources like sensors and static data from Web Feature Services (WFS) and Web Map Services (WMS). It has to be found a way in order to combine virtual and augmented terrain visualization and how to geo-reference the virtual with the real world. Until now, augmented terrain applications use widely optical tracking systems which deduct the user's position and orientation. Although recent systems use already Geographical Positioning Systems (GPS) and inertial motion sensor to locate the users, it still remains unclear how to keep a virtual copy of the real world in adequate alignment. As we will show in this work, we have achieved an interaction with the real world by superimposing a geo-referenced virtual topology on the real terrain, thus offering a new way to undertake large area surveillance which is in fact an Augmented Geographic Information System (AGIS). For this reason, GIS or design review content can be maintained with traditional GIS applications and can be visualized in the real world 3D context.

Further the application is required to support collaborative features and data exchange, thus allowing for example shared navigation and a physical distribution of the connected clients. For instance, a supervisor could follow the outdoor session performed on-site in-house.

1.2. Structure of the thesis

The thesis has been organized into eight chapters:

Chapter 1 is an introduction to our research work. It highlights the requirements for user-centered interaction and application design.

Chapter 2 “State of the Art” describes the requirements of an industrial virtual design review. It contains additionally a state of the art of multimodal interaction and interaction on portable

devices, in particular in AR outdoor scenarios. Further, it describes relevant collaborative concepts for GIS and dynamic data integration.

Chapter 3 “Application framework” explains the underlying concepts used for designing the application “IView” and in particular describes the methods of processing input stemming from various hardware devices. It is examined in detail how to integrate GIS functionality within the VR/AR application.

Chapter 4 “Multimodal interaction techniques” explains the newly developed multimodal interaction concept using a graph-based data structure for modality fusion. Special attention is paid to a high flexibility and configurability for a user-centered approach towards interaction during collaborative design reviews. It further describes efficient techniques for integration of common and more specific navigation schemes like the product centered ellipsoid navigation as well as two-handed multimodal interaction.

Chapter 5 “Collaboration and data integration” focuses on the development of information sharing amongst collaborative design review session using a distributed system setup. The chapter describes how static and dynamic data using the communication backbone and a geospatial database is inserted into the application at runtime, and how this data is accessed and visualized.

Chapter 6 “Augmented Reality for large environments” shows a novel approach to realize large area surveillance by superimposing a virtual topology on the real world video stream acquired from a motion-tracked video camera. It further demonstrates the usability of our multimodal interaction paradigm in an outdoor setting.

Chapter 7 “Assessment and validation” shows the relevance within an industrial setting in which automotive engineers and

designers evaluate the developed design review application in a real industrial situation and environment.

Chapter 8 “Conclusion and perspective” summarizes the contributions of this work to the state of the art and provides an outlook on future work.

Appendices A, B and C include various configuration files and illustrate realized hardware setups. At the end of this work related publications by the author are listed.

2. State of the art

2.1. Design review requirements

The application has been from start on intended to be used within industrial situations and environments. For this reason, it has been of highest importance to examine the requirements of real world scenarios and users in order to identify current issues and possible amendments to traditional design review interaction schemes. Before focusing on the actual users, it was required to scrutinize in which stages of the automotive development process virtual design reviews are undertaken. Then, it has to be examined which functionalities are needed. Figure 1 depicts the general phases in automotive car design.

The vehicle construction, in particular the chassis and suspensions are defined during the initial *concept stage*. A digital mock-up (DMU) of the underbody parts is created based on the automobile architecture found in this phase. Further activities include the verification of engine installation, transmission and other components. An intense benchmarking will then be performed in order to validate the target setting of the future automobile. It should be noted that during this stage, the main car performances are verified in a virtual way using simulations.

The following *product-* and *process design* phases are executed in parallel and in an integrated manner. Three or more design alternatives are created of which finally one is chosen. This stage is marked by the need for flexibility and transparency of process data. Specifically in the product design stage, the car designers develop multiple design alternatives according to the technical requirements set by the project. The group of stylists and engineers is extended by external designers to further inspire creativity. At this point it becomes obvious that a heterogeneous

group is working jointly on the given task, highlighting once more the collaborative character of the styling process. For this reason, means for model annotation, group discussion and style examination are needed. The design process is performed for each model until the choice for a final proposal is made. Of the final proposal a clay model will be built in a 1:1 scale (see schema in Figure 2).

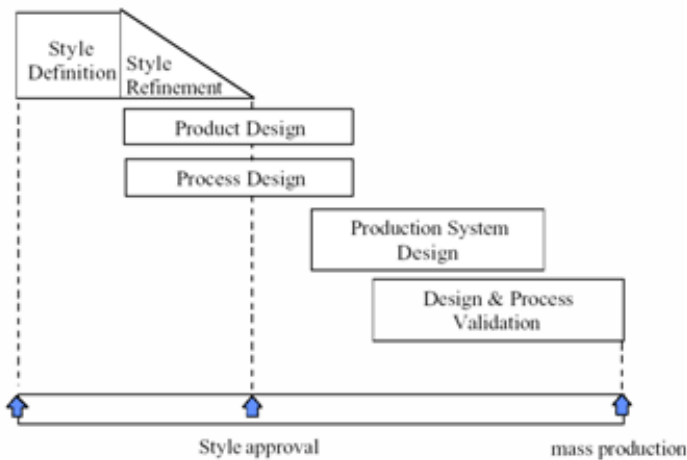


Figure 1. Product development process

The DMU is used additionally for other analyses, such as aesthetic analysis, geometrical and functional analysis and style presentations. These exchanges of information on the same model shows further the need for marking and annotating parts of the model with respect to user's comments.

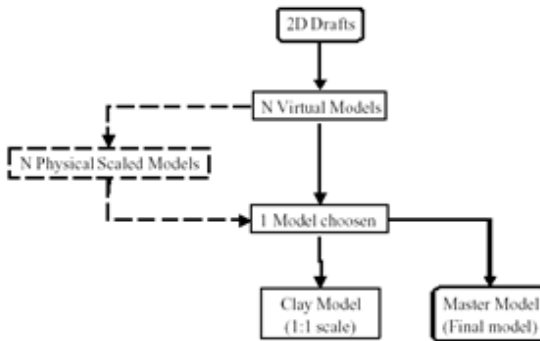


Figure 2. The actual process followed by Fiat styling designers

The following figure illustrates the entire process as specifically followed by the car designers at the Fiat Research Center Elasis and Centro Stile at Naples, Italy.

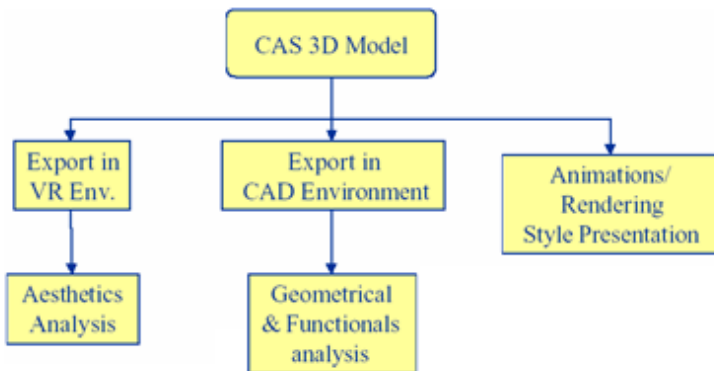


Figure 3. Virtual analysis

The different car projects are visualized on a Powerwall where designers can see the virtual car in 1:1 scale and explore the interiors according to their real point of view thanks to a tracking

system. However, this stage is entirely lacking the collaborative dimension of shared design reviews. As found out during a requirements analysis, the following improvements have been stated to be necessary in order to ameliorate the effectiveness a virtual design review in the automotive scenario.

- Fast method to calibrate physical and virtual environments.
- Fast method to calibrate user hand with the DataGloves.
- Accurate rendering (lights, materials, etc...)
- Improvement of the human- virtual components interaction.
- Delivery of frame rate of the virtual scene which can be suitable for the exploration with HMD.
- For the tracking system: accuracy, latency and sensor dimension.

The team of reviewers and thus potential users of our development consist of junior and senior designers, engineers, stylists. They are involved in activities related to underbody design, car body design and engine bay layout. They relate to CAE, Ergonomics and DMU showing once more the heterogeneity of the group and their respective professional background and domain. The tasks are performed in experimental laboratories specifically equipped with virtual reality tools. The users usually meet with other departments to share information to their colleagues. This highlights the need for a more efficient information exchange. The information is shared amongst the designer via talking, emailing and videoconferencing. It should be noted that no discrete data exchange exists using a dedicated collaborative design review framework and application.

The *final production* stage aims at planning the production process (logistics, machinery, equipment). Unlike a building construction, automobiles are mass products and built in batch production. For this purpose, a pre-series is built in a pilot plant to verify the production cycle.

2.2. Annotations

Several authors have stressed the importance of taking notes (1) which can be considered a by-product of the user's thoughts (2). This becomes particularly true during the design process where the sharing of annotations between designers is of key importance (3). The use of annotations in VR has been introduced by (4) who first introduced the post-it note concept. Similarly (3) present an annotation tool for 3D scenes which in turn is the evolution of the authors' early works described in (5) and (6). The notes are placed in the scene or through a placeholder. The work in (7) introduces a novel interaction technique based on the Boom Chameleon device (based on a screen mounted on a probe) where the user can annotate object in the scene by drawing directly onto the 3D object. Several annotation styles are made available and the user can navigate the scene by physically moving the screen in space. The possibility of annotating directly onto the 3D model has been proposed by several other authors such as (8) and (2). Annotation systems for Augmented Reality (AR) set-ups have been proposed in (9)(10). As far as interaction features are concerned the work in (11) presents a new widget called "tracking menus" which supports a menu moving with the pointer close at hand.(12) develops an interactive wall using a post-it metaphor and their manipulation via a pen. The work in (13) introduces a contextual control menu which combines the selection and at the same time the control of an operation.

2.3. Multimodal interaction

Most traditional applications adopt menu-based interactions to provide access to the functionalities available within the working environment. The interaction metaphors can vary from traditional 2D menus in standard programs, to more complex 3D widgets in more complex three-dimensional applications or in Virtual Reality (VR) system. Some VR applications make also use of hybrid approaches by using a further abstraction level with elements, such as a tablet or a pen (14) (15) (16), which in turn provide access to traditional menu-based commands. Major advances in the field of Human-Computer Interaction (HCI) have fostered the adoption of more natural forms of interaction. Recent developments have also brought to alternative forms of input which propose a brand new interaction paradigm more suited to the natural process of interaction to which humans are used to (17). In fact new interfaces have gone beyond the mere decoding of users' pointing actions by taking advantage of the information encoded through voice, gestures or gaze. This has led to multimodal VR interfaces where multiple communication channels (18) are used. The resulting interfaces make use of technologies which allow the user to interact through voice recognition and text-to-speech synthesis or gestures.

Such interfaces differ fundamentally from traditional GUIs since they adopt a novel, probabilistic approach rather than a simple event-driven command mechanism. In fact the "atomic" nature of the conventional event-based model used in GUI-based systems cannot handle the continuous flow of input streamed for instance by speech or gestures subsystems. In order to take advantage of the natural skills of the designer several authors have fostered the adoption of an integrated Multimodal Interfaces. In this context the idiom "modality" is used to refer to the syntactic and semantic properties of a signal; on the contrary the word "medium", is

adopted to focus on the production and transmission of signals (19).

Since the first system developed in 1980 (20) a number of researches have proved the efficiency of human-computer multimodal interactions (21). As cognitive scientists have proved, the design experience strongly benefits from the support of multi-sensorial, or multimodal, interactions (22). As stated by Forbus et al. (23), different modalities can be considered as complementary conceptual channels that can transmit information, not easily acquired spatially, regarding the spatial and semantic nature of the design. One of the main advantages of the integration of different modalities lies in the widened perceptual and conceptual bandwidth (23) available to the user to convey information regarding the object he is reviewing. Furthermore, such integrated approach is founded upon the effective support of human communication patterns (24) that can provide, if combined, spatial description and mutual interrelation hardly achievable through other means. Specifically it has been proved (24) that the raise in efficiency can be substantial in applications dealing with visual-spatial information. The very nature of multimodal interfaces has fostered a number of works which have adopted modular structure. Several authors have successfully promoted the division in different subsystems (25) where commercial recognizers were successfully integrated into customized applications. Most systems developed for engineering applications (26), for complex assembly and maintenance tasks (27) usually use off-the-shelf engines to recognize user's commands. The Studierstube (14) VR/AR platform introduces a new level of abstraction to the multimodal interaction. The system makes use of an open architecture for tracking devices, called OpenTracker (28), which provides high-level abstraction over different tracking devices and interaction modes.

Several approaches have been explored in order to assess the best merging of information coming from the different recognizer.

These include semantic fusion (29)(30), the MTC (Members-Teams-Committee) method (24) as well as other relevant statistical techniques. The adoption of Multimodal Interface in the mobile devices brings improved ergonomics through adoption of more natural interactions and it allows greater efficiency and naturalness in the way the user interface the machine through the adoption of human communication patterns (24). Research works have brought to the creation of portable multimodal VR/AR environments based on PDAs (31). Voice is used to navigate, annotate, and communicate (through voice-over-IP) with other users and a context sensitive interface shows the available speech commands.

Such a multimodal approach, although very promising, however lacks in standardized technologies, interaction paradigm and technologies. With respect to the latter issue some efforts are being made by industrial consortia. An example of this is the agreement made by Motorola, Opera Software ASA, and IBM (32) to submit a draft to the W3C® for a standard multimodal mark-up language X+V which stands for XHTML + Voice (33). The standard has evolved as part of W3C Speech Interface Framework (34). The main focus of the effort lies on multimodal Web applications. The authors in (35) developed an integration of gestures and speech by recognizing signals in parallel, yet unimodal recognizers were used to output lists of speech and 3D gesture hypotheses which were then routed to the time-aware multimodal integrator. The work of (36) proposed an open agent architecture to adapt to available input and output resources in order to provide distributed access to multimodal services. While the aforementioned approaches focused on command input, (37) has extended multimodal techniques to navigation in virtual environments (38). The author of (39) introduced graphs for binding modalities together. However this was done on a non-semantic level and provided means of customization only via a specification language.

2.4. Interaction with portable AR systems and set-ups

The use of advanced visualization technology and portable devices in product development has been a major goal in several projects of the last decade (40)(41). In fact the use of portable AR (Augmented Reality) applications represents a unique chance to improve the vision of the real world through by overlapping virtual information. The term Augmented Reality itself was coined in 1990 by Caudell and Mizell during the development of an industrial project at Boeing(40). In particular the work allowed revolutionizing the way wirings are assembled in modern airplanes. Similarly the ARVIKA project fostered the use of AR technologies for development, production and servicing of complex technical products following a user-centered approach. Recently the research project at the University of Technology Wien has specifically proposed the use of portable devices for 3D visualization/interaction for 3D mobile Virtual or Augmented Reality applications (42) (43). Another example is the Tinmith project at the Wearable Computer Lab at the School of Computer and Information Science, University of South Australia (44) (45) (46). Likewise the project MARS, Mobile Augmented Reality Systems (47) (48) carried on at the Computer Graphics and User Interfaces Lab, Columbia University, tries to solve the wide spectrum of issues emerging with the adoption of AR technologies in a mobile context. The project also brought to the delivery of an authoring tool to create dedicated content. The development of specific open-source 3D software library, targeted to PDAs and mobile phones libraries, such as Klimt (49) or OpenGL ES (OpenGL for Embedded Systems) (50), shows a growing interest from both the academy and from the industry towards the use of more advanced 3D visualization capabilities on mobile environments. The work in (51) emphasizes the importance of separating data models from specific applications. Further, they show how an inertial motion tracker can be used in conjunction with a GPS device to orient and position the user in

an outdoor scenario. However, their system lacks discrete matching and superimposition of georeferenced terrain features, and most importantly, the terrain topology remained without a virtual counterpart. It should be noted that (51) already mentioned the use of touchscreens as a suitable outdoor interaction device.

2.5. GIS and sensors

The integration of data stemming from sensors is currently an emerging field of interest in the GIS (Geographical Information System) domain for it allows the data to be visually analyzed in a 2D/3D terrain context in real-time. Sensor networks/grids form a new source of information if compared to the traditionally static integration of data into GIS systems. Current trends within the GIS domain currently are moving the research and industrial focus towards what are often referred to as LBS or Location-Based Services. LBS allows GPS-enabled mobile devices to georeferenced ad-hoc data for further processing in distributed systems, be it either immediate or a relayed communication, by using a client-server architecture for further interoperability with additional third party software. Recent improvements on the usability and the increasing computational as well as graphical power of mobile devices allows to create GIS systems capable to use a broad range of devices, ranging from cell phones, PDA's to laptops and tablet PC's. In this scenario the visualization of real-time data and in context dependent fashion, as in the case of sensors, poses a new challenge to distributed GIS architectures.

The authors in (52) presented a real-time immersive 3D system for visualizing geographical data and providing GIS functionality. Most importantly the work presented combined visualization with the management of the large, complex datasets common to geographical data. As a further step towards delivering an integrated interactive GIS, (53) have shown, that the concept of Virtual GIS can be extended towards the direct manipulation of

terrain features, creating a dynamic content. However, as illustrated by the authors, this took place in a standalone environment, where content creation and manipulation were separated. To remedy this shortcoming, Ohigashi et al. based their system on 2D legacy GIS applications and demonstrated the feasibility to link it to a 3D system via binding functional capabilities (shadow copy)(54). Traditionally, GIS systems work with 2D data which are organized into layers. To further bind 2D and 3D GIS systems, the authors in (55) developed a hybrid system which allows integration of 2D GIS data layers in a 3D view. The authors in (56) presented in their work capabilities to represent vector data for manipulation in a 3D environment by using visualizations directly mapped to the terrain textures. It should be noted that no translation to 3D representations according to the context was undertaken. Shumilov et al. developed a prototype of GIS system which allowed the construction of 3D and 4D models. Time-dependent VRML objects were integrated through the use of TimeSensors, which animated an object by switching among the time-mapped representations it (57). However it must be noted that the exchange of these data among multiple instances in the previously mentioned works however was not possible. This problem was efficiently resolved by Haist and Korte, who used a client-server based architecture in order to provide an adaptive delivery of 3D content over a public network (58). As support for workgroup collaboration is becoming increasingly important in the 3D GIS domain, (59) extended the concept of Geovisualization to support collaborative activities of users present within the system. Another step towards collaborative analysis of spatial data in a Virtual Reality environment was done by (60), who created a collaborative, interactive and location-independent working environment. Within this context, (61) showed the importance and feasibility of a scalable integration of sensor data and wireless sensor networks in a distributed system whilst maintaining their temporal relationship and thus achieving a

distributed, 4D enabled system. Steed used mobile devices in order to record sensor data like air pollution. The captured data were then integrated at a later time and mapped to overlay texture of the terrain (vector to raster) therefore not providing a real-time, distributed support (62). The authors of (63) emphasized further the need of a strong terrain and content visualization, particularly in the field of integrating data stemming from remote sensing sources. Complementary work of Walker et al. demonstrated the usage of a visual query language on conceptual and spatial relationships by forming an interface dedicated to non-expert users (64).

3. Application framework

3.1. Introduction

When designing an application for design review for both indoor and outdoor usage, the main problem is to bridge the distinct requirements of both scenarios in multiple ways. First, the application needs to support capabilities not only meant for pure virtual reality but at the same time, augmented reality techniques need to be seamlessly integrated. Reviewing a product in a virtual environment might help to examine certain features explicitly, but for many products like cars and buildings it is of utmost importance to compare virtual prototypes with real surroundings, real structures and physical mock-up's. We do not only seek to allow dedicated virtual and augmented modes, but we strive for offering a hybrid setup in which VR and AR can be changed at runtime. A current limitation of outdoor AR applications is the limited ability to track large areas where optical tracking is difficult. This can be due to poor video quality, weather conditions such as with rain or clouds and poor lighting conditions, for instance at dusk or with fog. We have addressed this issue by integrating location-based services (LBS) into the application. A gyro-enhanced attitude and heading reference (AHRS) system has been used to align the virtual camera with the orientation of the real camera. Further, we have integrated a service for determining the user's location using the global positioning system (GPS). A problem consists of embedding feature sets and virtual objects in both the real and virtual scenario *at runtime*. We have used a web feature service and message based sensor service to create appropriate means for the inclusion of virtual objects and data into the application.

3.2. Standard AICI

The requirements analysis undertaken within IMPROVE has outlined the need for the framework to be used as basic development platform. The **Advanced Immersive Collaborative Interaction Framework** (65) is based on the open-source portable scenegraph system OpenSG (66). AICI has been designed in a user-centered way, and high flexibility in usage and extensibility has been considered of high importance. Using OpenSG as base scenegraph library, AICI takes advantage of features such as advanced multithreading and clustering support. Taking a look at the scene graph architecture as shown in Figure 4, it is possible to see how this is organized in straightforward fashion with root nodes provided for workspace (the scene content), artifacts geometries (the representation of devices in the scene) and a widget root (for 3D menus).

It should be considered noteworthy that the camera transformation and the headtracking node have been inserted separately from the scene, thus allowing an easy modification and adaption which eases the integration of further libraries. For instance, if we would like to use a Head Mounted Display (HMD) instead, it would be sufficient to redirect the camera transformation beacon link.

A crucial factor for a VR/AR framework is how interaction capabilities are offered to the user, and how easily they can be adopted. Not only the range of devices decides on the quality of the framework, but also the way incoming user data is processed and forwarded to the application. AICI uses the tracking library OpenTracker (28) for providing access to a wide range of tracking devices locally and in a distributed fashion when run as a server. In fact, it serves as a tracking device abstraction layer underlying the framework. In particular, the integration into AICI is performed in the following way (Figure 5): incoming data are transformed by an EventGenerator to internal events which are

accordingly propagated to the event handlers of the artifacts, the software-based counterpart of a physical device. Contrary to an *ArtifactOperation*, the *UserOperation* is linked directly to the acting user.

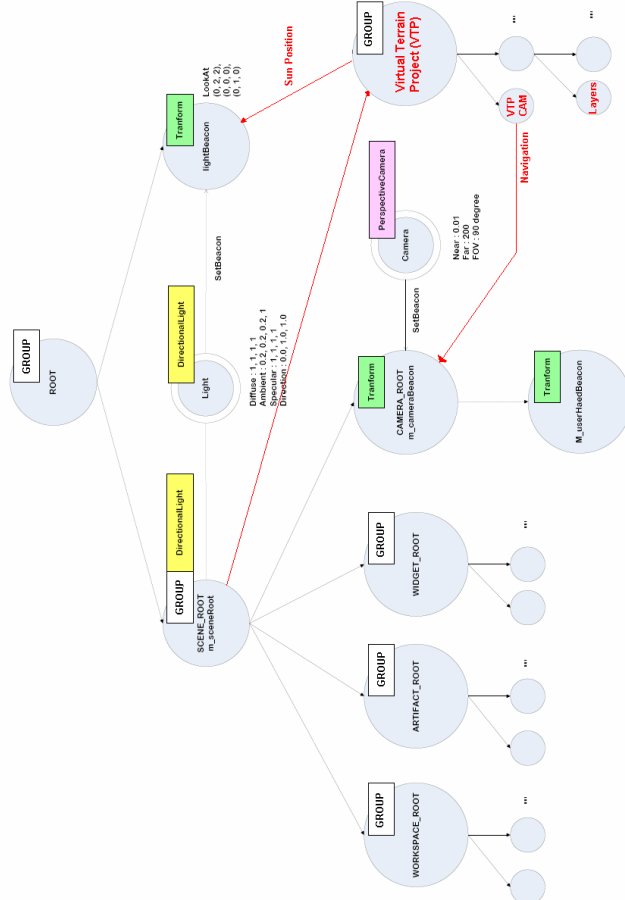


Figure 4. AICI scenegraph architecture

3.3. Extension of AICI

Since we want to support multimodal interaction schemes, we designate one artifact (the pen) for control and navigation input. This is done through a *CommandOperation* which runs in the background and redirects input according to the application's state to the *Sketch Observer* for gesture recognition, to the GUI handler, or to the navigation handler. We have created a special a Network Navigator artifact which manages networked input to allow the user to go a-/synchronous with design review sessions (section 5.3).

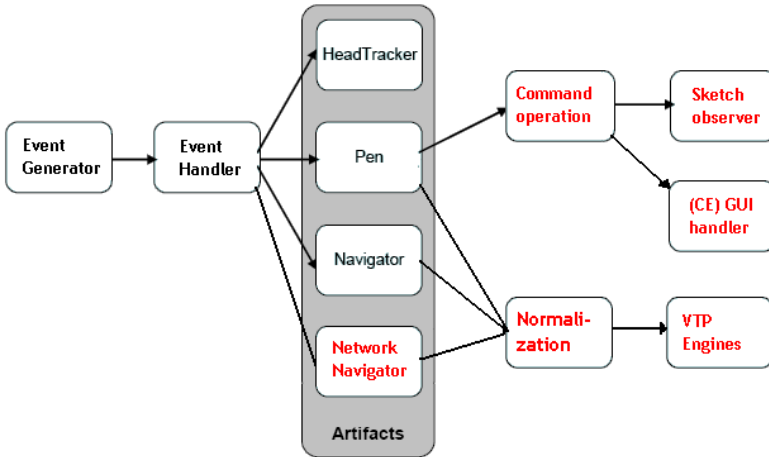


Figure 5. Event handling mechanism

It has to be highlighted that artifacts can be assigned multiple operations which can be turned on and switched off. Hence, we could as well have assigned the gesture recognition, the navigation and GUI handler as separate operations to the Pen artifact, but we have aimed for a more coherent input managing scheme. In any case, a command manager would have to decide

how which component has to process and interpret the inbound tracking data.

3.3.1 Integrated services

Several concurrent services have been integrated into the IView application/AICI framework to enable continuous location-based services and data integration. They can be separated in two types (Figure 6):

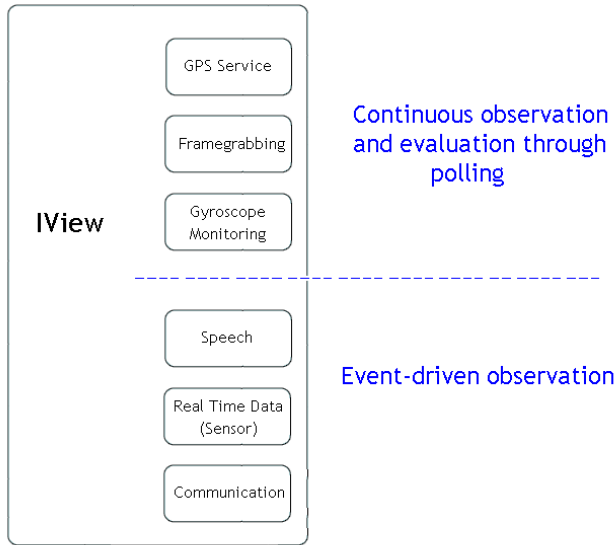


Figure 6. Services for data integration

3.3.2 Continuous observation

Especially location-based services (LBS) need to be designed to deliver data in near-to real-time to ensure at all times up-to-date nature of the applications' states. We have achieved this by

implementing a *polling* interface for each involved device as described below.

Global Positioning System (GPS)

We have used the Holux GPSlim 236 to retrieve the geo-location of the user to position him/her on the virtual terrain in the outdoor setup. The connection and communication with the device takes



Figure 7. GPS receiver

place via Bluetooth using the standard NMEA 0183 (National Marine Electronics Association) protocol. The data, including position, satellite id, no. of satellites, etc., are delivered in a unidirectional way via an ASCII string to our application. This has made it necessary to provide a parser which extracts only the geo-position used by the application. The time interval for polling has been set to one second as we have defined mobile AR for use in architectural design reviews as performing

from a fixed location with adequate precision. We have named the process ‘mobile’ due to the lightweight equipment and easy setup while on-site.

Video capturing

A crucial factor when enabling an application for augmented reality capabilities is the integration of video captured from a camera in terms of video quality. We have constrained ourselves to using a positional camera, that is we allow only rotational degrees of freedom. This cannot be considered a drawback due to our intention to provide augmented reality capabilities for large area surveillance. In fact, the movement of the user can be considered marginal since it would not add any noticeable

changes in the scene. Thus, the user is thought to perform a review from a fixed location. We do not access the camera device directly, but we have decided to take advantage of ARToolkit (67) which provides uncomplicated access to single frames. The video see-through capabilities are realized by defining a background viewport which places the captured image as a texture in a convenient way. The user can specify at system initialization which resolution should be used at which frame rate. Alternatively it is possible to pre-record videos from multiple locations for an area of interest (construction site) and having it played back in-house using a virtual camera software for indoor AR reviews.

Motion-tracking

As mentioned in the previous section, we need to track the rotation of the real camera and to measure the current magnetic northing in order to align the virtual and real camera. We have chosen the Xsens MTi Inertial Measurement Unit (68) because it provides drift-free 3D orientation, calibrated 3D acceleration, 3D rate of turn and 3D earth-magnetic field data, all of which provides an excellent measurement unit for stabilization and



Figure 8. Xsens MTi

control of cameras. The information is retrieved via querying the device constantly using a low-level protocol to achieve timeliness. We have further used the device to allow a new navigation metaphor, coined ‘SuperHand’ (section 4.12.1). The device is connected to the system via a USB cable of several meters length and thus is not limiting the user. The next generation model MTi-G, AHRS will ship with an integrated GPS receiver which will further ease the setup of the system on site.

3.3.3 Event-driven observation

Contrary to the previous section, some data are not available at all times thus rendering a continuous polling approach ineffective. For this reason a notification via a event/handler mechanism was chosen as illustrated in the following sections.

Speech recognition

Since the focus of this work is the innovative support of multiple modalities, an efficient way to react upon speech input needed to be found. We have chosen to use the Microsoft Speech SAPI 5.1 SDK(69). The recognition itself takes place using the Component Object Model (COM). When a phrase has been successfully recognized according to the generated Context Free Grammar (CFG), the command manager is notified and it then decides which action has to be invoked (section 4.7).

Gesture recognition

The events stemming from the configured interaction device (pen artifact) are processed by a command operation as depicted in Figure 5. This particular operation however is dependent on the application's mode. The input is redirected either to the GUI handler or to a sketch manager which further details the gestures. As 2D gestures are embedded in an interaction scheme which is reliant on the current dialog of the user with the application, the sketch manager acts a data provider. For this reason it does not react directly on the events in order to separate data acquisition from action invocation. However it signals to a command manager (section 4.7) that new gesture input is available. The data consists of the sketch geometries and derived data like bounding hull, which is beneficial during selection commands (Figure 23).

Communication

We have strived for support of collaboration amongst the users. The shared information consisted mainly of navigation and annotation. A continuous exchange of navigation and thus sharing of OpenTracker data has proven to be ineffective and error prone due to faulty transmission. Further, there is still the need to exchange categorized information amongst the distributed clients. As depicted in the high level architecture (Figure 9), the chosen architecture consists of five distributed, autonomous subsystems: communication backbone, interaction, tracking and rendering component and a central repository. Each subsystem, which can be physically located on a separate machine, communicates with the other modules through a high-level message exchange.

Each component is deployed entirely autonomously of the other system components. This gives the system a high degree of flexibility with respect to the physical distribution of the devices as well as robustness. By specifying a consistent well-defined protocol, the components are made independent and replaceable. The number of clients is in fact only restricted by the channel capacity of the communication backbone server. All messages are derived from a common structure which holds attributes such as author, system origin and its location. In section 5.2, we will describe in detail the data flow as taking place in the communication.

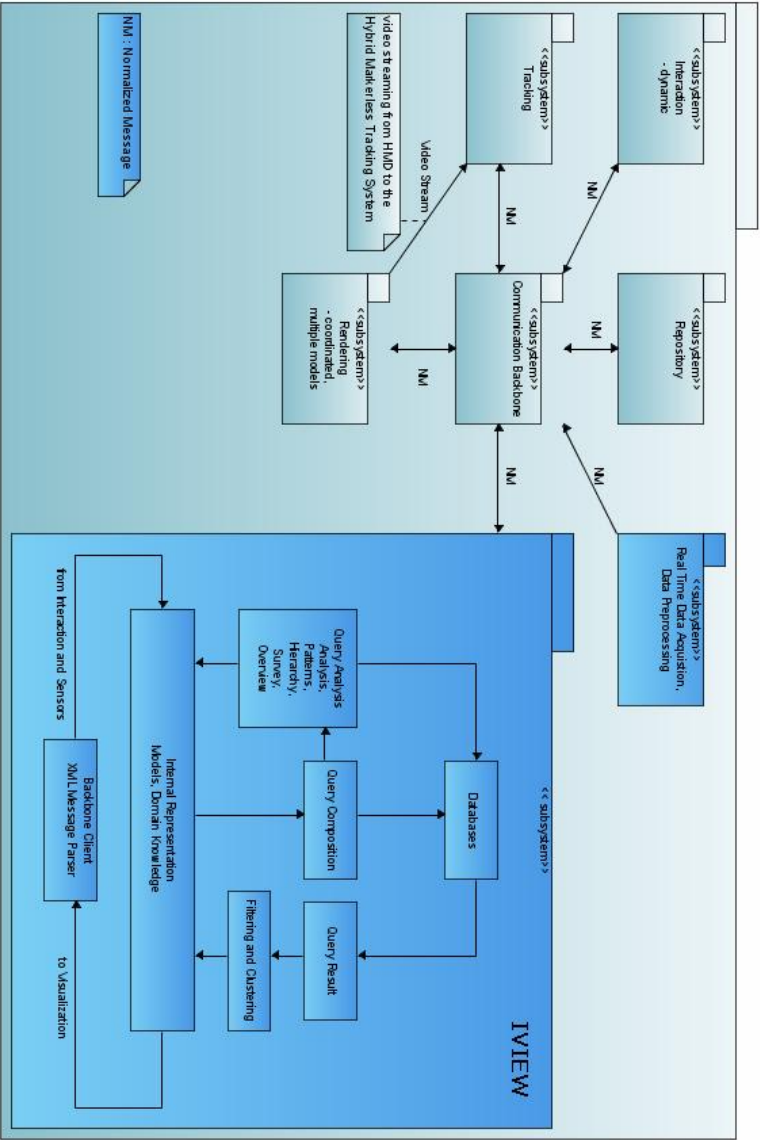


Figure 9. High level system architecture

Any given client does not need to know about the number of clients present within the working system. The information exchange is simply done by publishing and processing only the input data associated with a topic. This approach makes clients independent and it increases scalability, re-configurability, and reuse of components. However, this it emphasizes the need for a well defined communication protocol.

3.4. Standard Virtual Terrain Project

Unlike design reviews in an automotive setting, architectural design review sessions not only deal with the product itself, but it is highly required to view a building in its environmental context. This includes surroundings, present and planned terrain objects like building structures, vegetation and technical entities as well as related derived data like visualizing the estate's boundaries, verifying the land use designation be it either as polygons draped onto the terrain or as superimposed maps. Further, the influence of sun light and its effect on the reviewed building is of highest importance to an architect, enabling him/her to audit lighting conditions on-site. This makes is necessary to simulate the sun's position contingent on the time of the day and the geo-location of the examined structure.

For this reason, we have chosen to take advantage of the OpenSource Virtual Terrain Project (70) by integrating it into the AICI framework. The VTP consists of a set of platform-independent applications and libraries dedicated to offer support for geo-visualization and rendering techniques like several continuous level-of-detail (CLOD) algorithms, specific terrain navigation metaphors, and various import capabilities for geo-referenced data.



Figure 10. Tiled terrain with the VTP (OpenSG)

As shown in Figure 11, VTP applications are based on three main libraries. The terrain library *vtlib* handles terrain generation and rendering while the data library *vtdata* implements access to geo-data import stemming from a multitude of data domains (geo-referenced imagery, elevation data, vegetation data, etc.). *vtui* offers an API independent 2D dialog-based user interface used in the main applications. Further advantage of the layer-based approach has been taken which enabled a CAD-like data organization.

3.5. Scenegraph architecture of the VTP and integration

The VTP uses its own overlay scenegraph, which decorates native OpenSG nodes. This simplifies its integration as it is only required to insert the VTP root node into an appropriate location in the AICI scenegraph (Figure 4). As it uses view-dependent tessellation of the terrain heightfield, it is required that cameras in

both scenegraphs share the same position and orientation. OpenSG's approach to using single parented nodes and multi-parented node cores have eased this such that only the AICI camera node core needs to be shared with the terrain camera. This way, we have achieved a full integration and interfacing with the application from the programming point of view can now be realized just like any other VTP application using all supported features. A customized VTP SceneViewer has been developed to extend viewing capabilities by switching between VR and AR mode at run-time. Finally we have integrated a root node for embedding and interfacing geo-visualization and navigation functionalities via the Virtual Terrain Project.

3.6. Extensions to the VTP

Since AICI is based on OpenSG, it has been necessary to transform the rendering support of *vtlib* from the 3D graphics toolkit OpenSceneGraph (71) to OpenSG (72). This process has been made feasible due to the high level of abstraction present in the modular design of the VTP components. Figure 11 depicts the architecture of the VTP plus the developed extensions. To achieve support of an augmented reality system setting, we integrated *libAR* (67) which is responsible for acquiring access to captured frames of a video camera.

During outdoor design review and GIS sessions alike, access to has to been given to the user at runtime to geo-referenced data. For this reason, *vtdata* has been enabled to retrieve features query a Web Feature Service (73) and to translate the retrieved Geography Markup Language (GML) to native VTP entities like buildings, roads with conjunctions, railroads, trees and additionally forests.

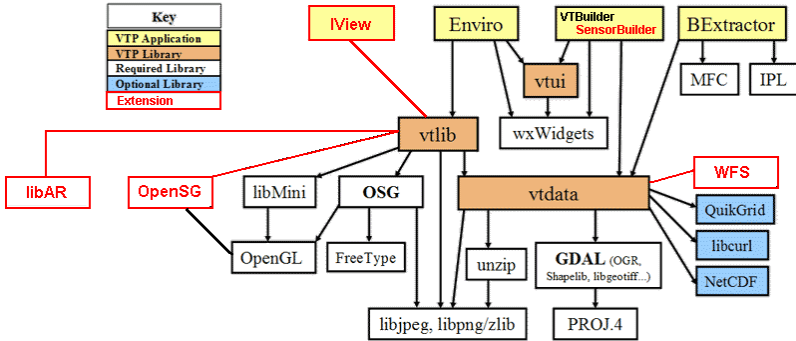


Figure 11. Extensions to the VTP

Further, we have developed an extension to VTBuilder (a tool for managing terrain elevation, imagery and cultures) called SensorBuilder (section 5.2.6). It allows being run as a client connected to the communication backbone and creates, observes and manages dynamic data.

3.7. Conclusions

In this chapter we have presented the architecture for a distributed VR/AR framework and application which establishes a communication to instances using a Message Passing Middleware (MPM). The information exchange uses XML messages in a channel topic/subscription method to deliver collaborative navigation and scene modification. It has been used to integrate and to visualize time-varying information inside the VR/AR application. This is especially important when working with distributed data providers such as sensors where the frequency of the information update is high.

In order to realize an outdoor scenario, we have extended the framework with a terrain visualization module. This module was enriched with video see-through capabilities to allow the alignment of virtual terrain with the real world. Further we have

integrated location-based services such as a GPS module and an electronic compass/motion tracker to geo-reference the user and his/her surroundings. A retrieval of geospatial data using Web Features and Web Map services has been successfully integrated. However it must be highlighted that this approach is specifically applicable when dealing with large distances where optical marker-based and marker-less tracking systems fail to achieve a correct camera mapping.

A major benefit of our system is the geo-referencing of virtual objects and the topology of the virtual terrain with the real world. This way it becomes possible to virtually interact with the real scene by placing virtual 3D content such as trees and houses directly or to drape derived data like 3D maps onto the real terrain.

The application can be configured to a wide range of input and display devices using mapping artifacts which normalize and redirect the incoming data. Within this scope several services to handle speech input and gesture recognition have been implemented allowing multimodal interaction schemes.

4. Multimodal interaction techniques

4.1. Introduction

Much effort has been spent during the last years in developing environments dedicated to virtual design review. However, many approaches are very limited with respect to the range of used input devices and, more importantly, in the way the users are enabled to interact with the product and the system itself. Specifically, in the field of design review, the reviewing group consists usually of users with a very heterogeneous professional background, potentially having different interaction preferences. This problem is further complicated when the application has to address multiple scenarios and hardware configurations. Additionally, such sessions naturally require a frequent switching between navigation and scene manipulation, for example to examine a particular part of the prototype while taking an annotation. The developed methodology builds onto these requirements and proposes an approach characterized by the elevated user-friendliness based on the development of a customized multimodal interaction dialogue.

4.2. The multimodal interaction dialogue

When designing interaction techniques for software systems, developers naturally look at an application from their particular point of view, typically biased by their background. For this reason they often implement interaction techniques that are appropriate for daily use by people with similar mind set. However during design reviews several people with different professional backgrounds are working collaboratively and each user has potentially his/her preferred way of working with software systems. Review panels are made by car engineers,

designers, architects and 3D modelers. An architect may want to use a sketch for invoking an action like taking a visual bookmark (his natural interacting way) while other users may prefer traditional dialog-based interaction. The greatest challenge of collaborative design reviews, in terms of interaction, is then to give users access to the same functionality through customized user-centered modalities.

For this reason it is essential to tackle the problem of distinct modality configurations according to each user's needs. This includes customization of gestures, voice commands as well as the graphical user interface (GUI). The whole interaction infrastructure in fact needs to be highly customizable according to the user's specific needs, to the design review scenario itself as well as his/her aesthetic taste to improve the users' efficiency and perception of the application. For example, the interface should allow an architect to define his personal interaction process to load a 3D model regardless of whether he wants to use a circular gesture, to select a specific icon on the GUI or to use a speech command in his native language.

The problem becomes more complex when applications need to address distinct industrial products and scenarios as it has been the case in of this work. For instance, during a car review session, the vehicle resides always in the focus of interest while in an architectural scenario, the surrounding environment, its structures and terrain, the lighting conditions, location as well as its social context, become crucial criteria for the final product.

4.3. Definition of commands

Starting from the requirement that the customization of the interaction dialogue must consider the fact that distinct users issue commands in different orders, we have defined provide a dialogue model capable to support the definition of different interaction sequences.

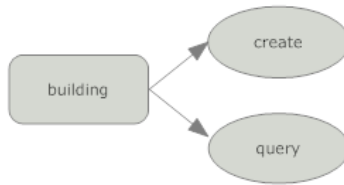


Figure 12. building, create-query sequence

or

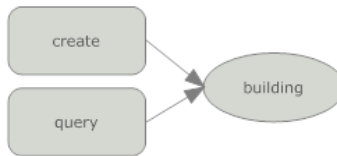


Figure 13. create-query, building sequence

For instance the first example (Figure 12) shows an “object oriented” command sequence while the second approach (Figure 13) focuses on the process itself. It is obvious that ideally both paths should be supported and both interactions should invoke the same set of commands. The availability of such configurability strongly influences the usability of the application. The first consequence of this is that users need to be advised constantly of which commands and modalities are available to them. This is essential for them not lose focus on the action they are performing. At the same time it is very important to define an approach that encourages user to explore new interactions different from their traditional approach.

Effectiveness plays a crucial role when dealing with industrial applications where the use of speech, gestures and other modalities must be supported in a way that minimizes the learning time, to maximize efficiency and reduce costs. However while in traditional desktop GUI's most functionality are

discovered by exploring menus, when using a multimodal approach the interacting schema become much more opaque.

4.4. Use of multimodal commands

Using one modality represents a major restriction in that it severely limits their conceptual bandwidth (30). Therefore it is of utmost importance that modalities are merged and made, whenever possible, exchangeable according to the users' preference and efficiency with respect to the intended response of the application. It is also essential that the interaction process is persistent and highly configurable by each user. Configurability does not only include specifying how user input is used and resolved by the application, but also how to use a particular modality for which purpose. A suitable interaction metaphor thus should offer:

- Support, when appropriate, of a single modality. Support for combination of multiple modalities during the dialogue with the system.
- Restriction to a sub-set of modalities when inappropriate for a application functionality.
- Definition of the application's functionalities according to the modalities available.
- Internal configuration of modalities.
- Easy-to-apply modification
- Persistency of the data structure.
- Establishment of a dialogue at multiple levels of resolution.
- Restriction to a subset of modalities for specific tasks and scenarios

- Possibility to freely change modality configuration at run-time.

For these reasons it is essential that users have a clear understanding of the functionality offered by the application while the data structure defining interactions is kept transparent to them.

4.5. The interaction graph (IG)

This points out that each element, or building block, of the interaction dialogue must be unique within the data structure adopted within the application. The different dialogues and their building blocks are then normalized and arranged (order-independent) in way that logically represents the way user would communicate with the application. For instance the command: *“create an annotation and pick a part”* or *“assign a material to a part”* can be normalized to the following keywords:



Figure 14. First set of normalized keywords

and



Figure 15. Second set of normalized keywords

The previous command sequences (Figure 14, Figure 15) can be combined into a more complex node structure:



Figure 16. Second normalization of keywords

This in fact can be seen as a second level of normalization. As a result the user only works with a predefined set of unique keywords, increasing the potential familiarity with the application and its adaption. Implementing a structure where nodes are interconnected by edges defining the “multimodal” commands has lead to the definition of a so-called interaction graph.

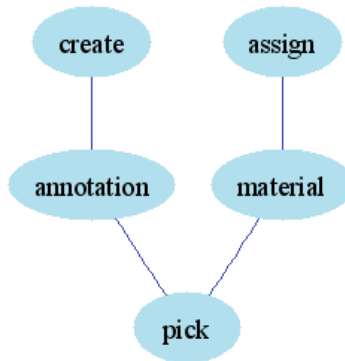


Figure 17. First graph

Within this graph structure nodes can be multi-parented and hence reused. Furthermore, it should be noted in the example in Figure 17 that there is the possibility for redundancy in the dialogue, which however should be excluded to enforce a dialogue free from misinterpretations. The graph nodes define actions while edges are used to define the interaction mechanism. Therefore actions become independent from the used modality. Further this provides the means for a straightforward extension to include new modalities and input metaphors (section 4.12.1).

During the dialogue with the system, support for repetitive tasks can be achieved by providing the possibility to go back in the history of interactions. For example, if the user wants to create an annotation and assign it to geometry, it would be tedious to repeat the complete dialogue. This can be remedied by forming a bidirectional graph where each node knows its predecessor. Thus

a repeated assignment of an annotation would be performed like so:

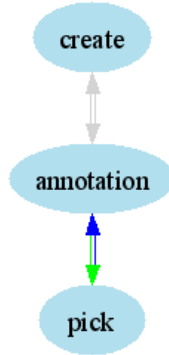


Figure 18. Support for iterative commands

The bidirectional graph approach was chosen to achieve a seamless integration of distinct modalities with their own natural advantages. It should be noted that the users' intention is clearly readable by reading the path of the user's interaction. The interactions path chosen by the user helps to find the correct action sequence thanks to the support for multiple parent nodes. Thus, create-annotation-pick can be separated from query-annotation-pick. This preserves the original intention of the user to create an annotation, and not to query it.

Figure 19 shows a complete example of an interaction graph for a design review application. It must be noticed that the existence of a root is required wherever the user is given access to the main functionalities. In the example discussed these top-level entries are: *navigate*, *create*, *query*, *assign*, *open* and *view*. These have been selected on the basis of a user requirement analysis undertaken in collaboration with the designers and engineers at Fiat-Elasis. The interaction dialogue is manifested by navigating through the graph. In fact connections between keywords (nodes) via abstract edges do not only show the keywords' semantics. The

edges in fact are the essential key to the integration of the various modalities. The edges of the graph are in fact assigned attributes which specify under which circumstances the current interaction node is shifted to the next level of interaction, i.e. child node.

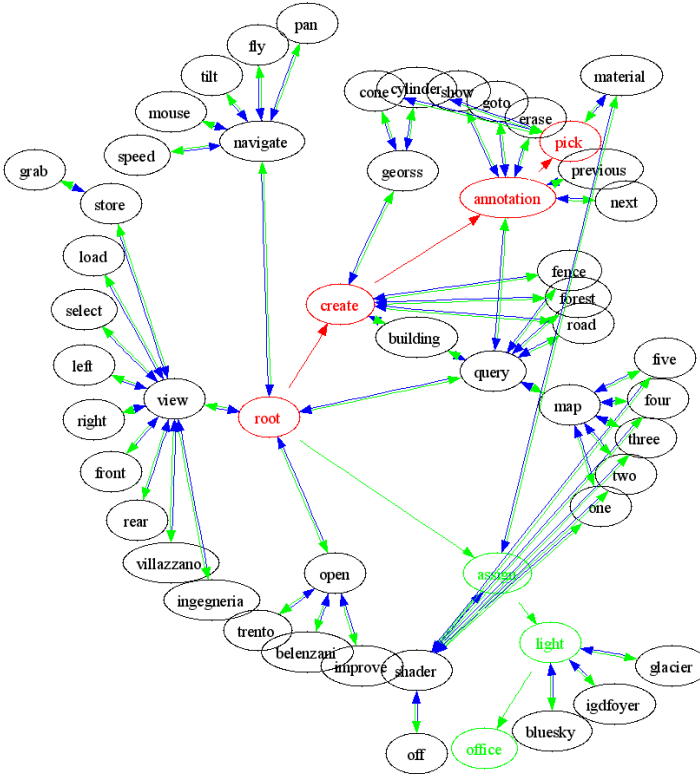


Figure 19. A more elaborate interaction graph

The provided example can therefore be represented, in a multimodal perspective, in the following form:

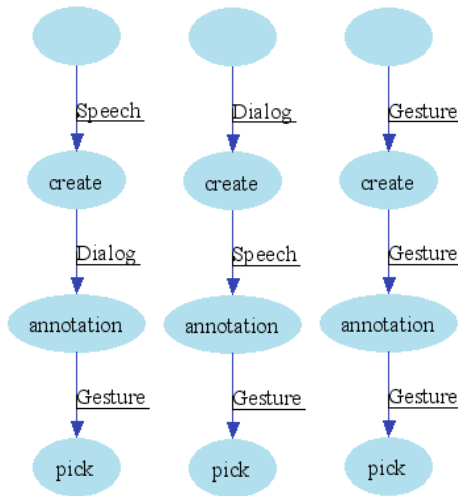


Figure 20. Combinations of modalities

Looking at the sequences of actions, this yields a structure like:

Root - (gesture, speech, dialog) - **Create** - (gesture, speech, dialog) - **Annotation** - (gesture) - **Pick**

where the type of modality is assigned as an attribute to the respective edges.

A first filter can be seen on the edge (annotation, pick). It allows only gestures to access the pick functionality, excluding support for voice and dialog input. This is because the gesture modality is the only appropriate form of interaction for a “pick” action. Other gestures, beside a “tap”, suitable to performing a pick could include a “cross” gesture for marking, or a “rectangle” for selecting multiple objects.

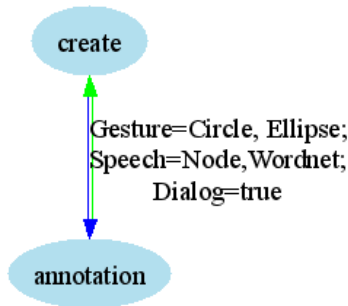


Figure 21. Edge attributes define modalities

Figure 21 shows how the user can create an annotation by drawing a circle or ellipse inside the application, or by speaking out the word ‘annotation’ as well as by selecting the ‘annotation’ functionality inside any interaction graph-enabled dialog.

A modality *accessor* can also be used as a means to provide input to the next interaction. For instance in **Annotation** - (gesture=Tap) - **Pick**, the tap gesture additionally functions as a locator for a picking interaction and therefore it reduces the necessary interactions to a bare minimum. This concept has been extended to generally use geometries of sketches as input to the next action, when appropriate and logically required by the application. Edges therefore can be separated into two types:

- (1) Selectors: Simple advances in the graph
- (2) Locators: Geometrical input AND advances

This way gestures are not only used for identifying the appropriate edges on the graph but they are also used to provide location-based interactions such as picking, selecting via their geometric properties.

The graph, as described in the previous paragraphs, forms an interaction mechanism where the user is enabled to tailor the

application to his personal interaction preferences. Besides this, it is possible to customize the application itself by restricting the graph to sub-graphs.

It has to be highlighted that we have defined the interactions independently from the application. This way we have freed ourselves from any hard-wiring modalities to the offered application functionalities. Patterns of user interactions can be specified and reused, potentially using them in other interaction graph enabled applications, too. A major benefit of this approach is that interaction mechanisms are defined only once for different modalities outside the application. To this extent, this technique could be called “anticipatory”, since only reasonable, context-aware commands are accessible.

Reducing the scope of the application, for instance by reducing the number of functionality the user has access to, becomes very easy. In fact it is sufficient to restrict the interaction graph to include only those nodes which are required, for instance to customize the application dialogue to a specific application scenario, be it automotive or architectural design review.

Finally configurations of the various modalities can be then generated and derived from the designed, user-tailored interaction graph. This is the case of the voice interaction whose corresponding CFG (Context Free Grammar) is automatically defined from the interaction graph.

Such a flexible approach, as the one introduced, based on the use of an interaction graph, it also requires proper design of the input handling, which must be dealt with in a flexible manner. Since the interaction graph is created and maintained from outside the application, the main problem lies in identifying the correct functionality handler at runtime. Because the action of the user is determined by the path (the sequence of nodes) in the graph, we assign a set of attributes (order-independent) to the action-handlers which are in fact the names of the nodes present within

the interaction graph. Actions can be distinguished on an abstract level if they can be performed in a standalone way, if they are embedded in a sequence of actions and if they require further input from the user.

- (1) *Standalone*. This is the simplest case. The action is directly invoked based on only one keyword.
- (2) *Relative*. These commands are based on a pair of nodes like e.g. (open, model) or (annotation, next). There is no need for knowing any previous interactions.
- (3) *Embedded in sequence of actions*.
- (4) *Parameters required*. Certain interactions and functions require geometric input which is taken from their geometries. This information is then passed to the interaction handler for processing. For instance, a Tap gesture for a picking interaction would provide 2D *screen* coordinates as input parameters for ray-intersection calculations. Restrictions on which objects are “pickable” and how screen coordinates are mapped are solely decided by the action handler according to the application’s state.
- (5) *Redirection to upper node after node has been visited*. Action handlers can manipulate the current active interaction node. This proves to be particularly useful after an action which could be potentially repeated multiple time like, as for instance when applying a material. The handler then moves the current node cursor to its parent, which is in fact retrieved from the interaction path.

The appropriate action in the application is found and invoked by matching these keywords with attributes assigned to the application-defined behavior.

In some cases it is required that actions act differently depending on the type of modality (speech, voice, dialog).

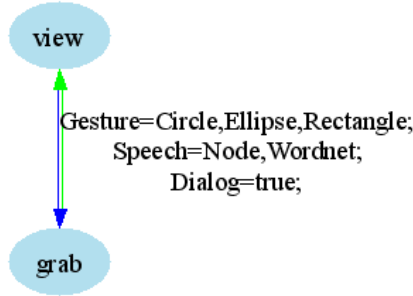


Figure 22. Grabbing a view

In Figure 22, a view can be stored by either sketching a circle, ellipse or rectangle or by speaking out the node name ‘grab’, as well as by selecting the graph-enabled button of the GUI.

4.6. Some examples

To better illustrate the approach in a real context it is useful illustrating the example of a user interacting with the system to create a visual bookmark of the scene. These are partial or global screenshot of the scene as currently seen by the user together with the users’ point of view at that time. The user is enabled to retrieve his/her previously saved locations through a visual selection of the preview, showing particular objects of the reviewed product.

In this very case the application will behave differently according to the specific interaction process used: in the case of using gestures for accessing the ‘grab’ functionality, the drawn geometry is used for specifying an area of the viewport as visual bookmark (Figure 23) whereas a speech and dialog interaction triggers a screenshot of the complete viewport due to the missing bounds.

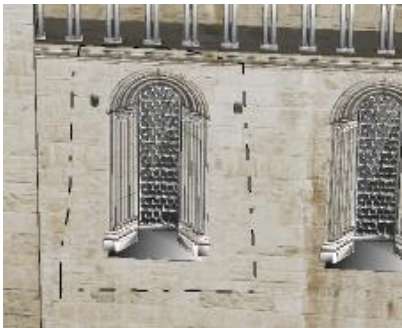


Figure 23. Taking a visual bookmark

As we will see in section 4.10.4, dialogs of the type as shown in Figure 24 are embedded in the dialogue of the user with the system to restore at any time the point of view saved with the bookmark.

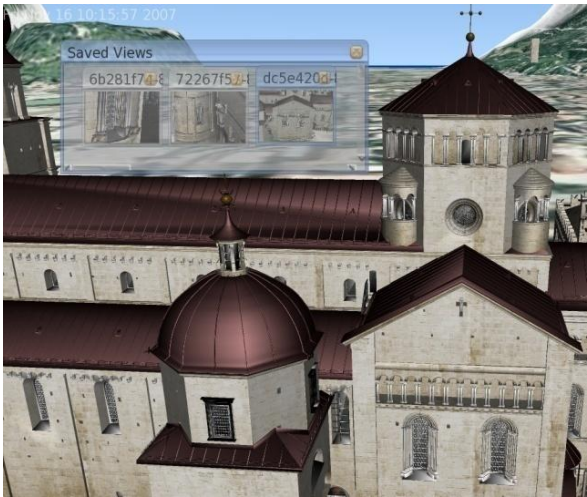


Figure 24. Restoring a visual bookmark

A further example is shown in Figure 25 where geometric input is now used for the creation of scene objects like fences or buildings, within the scene. The handlers react differently on each modality.

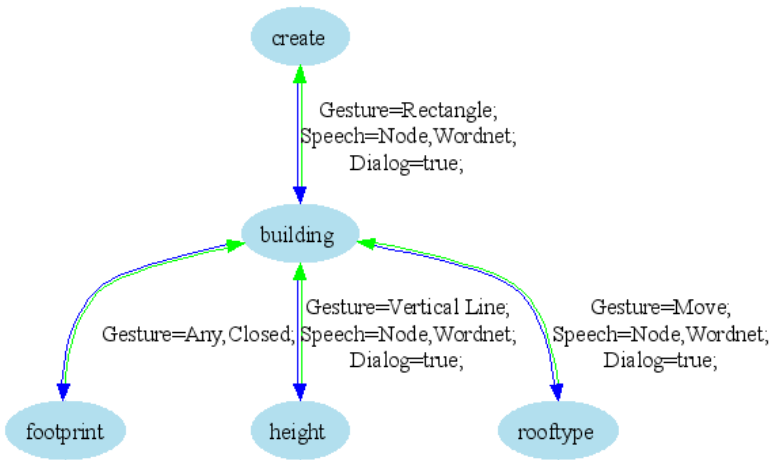







Figure 25. Modality-based creation of a building

In the following table it is possible to see how actions performed by the application are matched to the various interaction steps. It should be noticed that the order of interaction can be freely chosen. The building is visualized when all required geometric properties and the building location are present. The presence of all required properties is checked in the handlers for the nodes *footprint* and *height* which define the mandatory geometric definitions, a specification of the rooftopype remains obligatory. This shows that a certain amount of knowledge about the domain is encoded directly in the action handlers. In fact it is possible to state that the goal is to access domain knowledge through a predefined set of keywords which can be used in a configurable manner.

Graph-based multimodal interaction

	No action handler required. It is only present to provide a semantical context to the user.
	It creates a new building and it inserts it into the structure layer.
	It generates the footprint property of a building. For this step, an input gesture is mandatory as parameter to the action handler. This 2D gesture is projected from screen coordinates onto the actual heightfield of the terrain by a ray-intersection.
	It assigns height to newly created building. Dependent of the type of modality, it either takes the height from the <i>line stroke</i> , or it brings up a dialog for input via a spin control.
	It specifies the type of roof like Felkel, flat, etc. This action handler brings up a dialog for choosing the roof type.

4.7. Multimodal integration using feature set

An essential step of the multimodal process is multimodal fusion as illustrated in Figure 26. In fact application specific commands are directly invoked by the command manager with no knowledge about the active interaction graph. The main task of the command manager is to evaluate the various inputs and to find the correct node to advance to. Upon initialization of the application, an interaction graph is loaded from the file system. This can be specified at the command line, as described in detail in section A.5.4 “Framework configuration”. Consequently all dependent application objects are initialized, i.e. the speech module with the appropriate speech configuration (with the relevant CFG) and the ring menu with the correct icon resources.

As illustrated in Figure 26, a command manager forms the central component within a complex architecture with different components including uni-modal recognizers. The task of the command manager is to handle input from the available modality and to map it to the correct function handler of the application. By analyzing the set of keywords (nodes) present in the user interaction path of and by mapping these to the attributes of the application’s functionality handlers, the correct handler is determined and invoked.

The mapping of the interaction path to the applications functionalities is performed as follows:

In Figure 19 two interaction paths are depicted (rendered in red and yellow). The application creates the intersection set between each application functionality and the interaction path. It then invokes the function whose attributes are composed of the keywords of the intersection set.

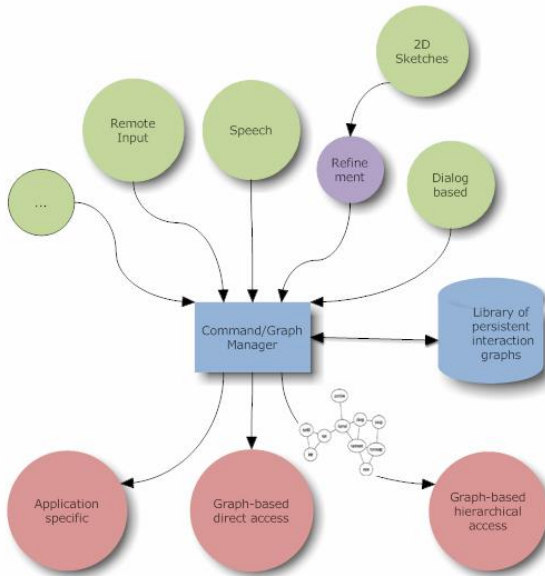


Figure 26. Input processing and command invocation

For example, if the user follows the path *[root, create, annotation]*, and the application offers a functionality with the attributes *[create, annotation]*, it will call the function to create an annotation. Further, if the user decides to place the previous annotation on the model, thus follows the path *[root, create, annotation, pick]*, the functionality *[create, annotation, pick]* to assign an annotation is invoked. It must be highlighted that still, the *create* attribute is necessary in the functionality attributes since it needs to be differentiated from for instance *[query, annotation, pick]* where an assignment of an annotation is not allowed.

4.8. Speech

The strength of using speech as input modality lies particularly in the fast and direct access of the nodes without the need to use additional pointing devices. An omnipresent issue when using voice has always been the peril of a false recognition which can easily bring the application to a non-valid state by activating unintended functionalities. However it must be noted how using the hierarchical approach, which reduces the available speech input only to semantically meaningful nouns and verbs, yields a significantly higher correctness, robustness and confidence.

Since the user adapts the interaction graph, it is necessary to create a speech configuration after each modification to the persistent interaction graph is introduced. This is an XML that is used by the speech recognition engine, in our case Microsoft SAPI, to filter only a specific subset of commands according to a set of rules. The configuration file which is referred to as CFG (Context Free Grammar), is automatically generated from the interaction graph and it can be validated by using the Grammar Compiler application bundled with the Microsoft SAPI SDK.

4.8.1 Hierarchical commands (defined in the IG)

Following the approach illustrated, enabling hierarchical navigation becomes straightforward. The configuration generator iterates over the nodes present within the actual interaction configuration, and it writes the node names or aliases as a top level ID and rule. To maintain a high configurability, we do not encode the actual verbs directly, but we encode keywords and node names. This allows exchanging recognized keywords with synonyms (like “*navigate*” with “*steer*” or “*plot a course*”) or their translations to other languages (like “*navigazione*” or “*Navigation*”).

The input handler within the voice command manager disassembles the recognized rule name into keywords, separated by a delimiter token which depicts the set of nodes necessary for invoking the assigned action. As in the simplest case for hierarchical advances this yields for the below example:

```
<GRAMMAR LANGID="409">

<DEFINE>
<ID NAME="VID_navigate" VAL="1"/>
</DEFINE>

<RULE NAME="navigate" ID="VID_navigate" TOPLEVEL="ACTIVE">
<P>navigate</P>
</RULE>
```

Figure 27. CFG for hierarchical navigation.

4.8.2 Direct speech access to any node

Similar to the previous paragraph, the node names have been encoded within the rule element in the name attribute:

```
<GRAMMAR LANGID="409">

<DEFINE>
<ID NAME="VID_navigate_fly" VAL="2"/>
</DEFINE>

<RULE
  NAME="navigate_fly" ID="VID_navigate_fly" TOPLEVEL="ACTIVE">
<P>navigate fly</P>
</RULE>
```

Figure 28. CFG for direct access.

When the handler assigned to the speech modality recognizes an uttered phrase, it checks the rule name and rule ID. Clearly visible is the benefit of using a Context Free Grammar (CFG) since it narrows the available voice commands to a predefined set, which

reduces recognition errors to a minimum. This is even truer when using multiple keywords within one single phrase element.

After this, the rule name is decomposed into keywords, which were concatenated via a delimiter “_”. These keywords, or node names, are inserted into a *set*, guaranteeing their uniqueness. It must be noted that there is no need to know about the modalities used so far. The set of keywords uniquely identifies a path to a node, which becomes the current active interaction node.

It should be highlighted that this procedure is distinct from the hierarchical speech interaction because the node needs to be redirected to a potentially completely different application context for consistency and further interaction. An example: When speaking ‘*create annotation*’, the node cursor is placed on the node ‘annotation’ characterized by the path “*root-create-annotation*”. This enables again all interaction edges to start from the annotation node, like using a horizontal line (rightwards), for instance to denote ‘next’ or a “wavy line” meaning ‘erase’.

4.9. Gestures

In order to decode user’s gestures we have chosen Cali(74), a software library for the development of Calligraphic Interfaces to support a 2D gesture/sketch modality. Cali supports only a basic set of primitives along with some additional properties like e.g. open/closed shapes or continuous/dashed. To enrich the set of 2D sketches to allow for a better usability, we have further examined the input through a post-processing step based on their shape. This way it is possible to identify if a line is horizontal or vertical, and on which side the sketch begins. Analogously, it is possible to identify whether an arrow points upwards, downwards, towards the right or left side of the screen.

A sketch manager observes the drawn geometries on the overlay. After a sketch is successfully identified, it notifies the command

manager via a message that a new gesture input is available for further dissemination by the event-handling mechanism. By assigning a list of gestures to the gesture attribute of the ‘modality’ edge, it is possible to reach a child vertex in multiple ways. Specifically this is helpful when assigning similar gestures, for instance *circle* and *ellipse*, to reduce the chance of faulty input.

Remote command invocation can be easily enabled by using the same approach as described in section 4.8.2 by issuing remote voice commands. This allows remote guidance of a user-in-the-field which is particularly helpful during collaborative sessions. This way, multiple users can control the same instance of the application.

4.10. Dialogs and hierarchical ring menu

4.10.1 Hierarchical ring menu

At the same time, context-dependent actions need to be offered and well-presented. Thus, a dialog system having a menu structure as its core consists of two parts, namely a static and a dynamic one. When designing the interaction with a GUI, one needs to know about the devices available for input. We have decided to use traditional touchScreen (single touch) displays, TabletPC, Wacom Tablet and 2D mice restricts on the common denominator to having only one button available for clicking an UI element. This outlines that providing separated context menus would be inconsistent with the interaction graph. As for the dynamic part of the menu, it is needed a way to (1) show the current interaction status and (2) present the options (which are in fact the vertices at the other end of the outgoing edges of the current node, having the ‘dialog’ attribute attached to them).

4.10.2 Layout

Further, close attention had to be paid on how to present the various commands graphically. As depicted in Figure 29, the static part is aligned on the left side, allowing immediate access to change application settings. The central button always reflects the current active interaction node. Additionally, it provides the means to go back in the history of actions. The right side of the menu shows the interaction nodes available for input by the *dialog* modality. All elements of the ring menu (buttons, background and ring) have been made transparent to not occlude unnecessarily any part of the scene.



Figure 29. Ring menu at top level



Figure 30. and at creation node

As we will explain in section 4.11.2, also the layout can be configured outside the application without the need to recompile it. As can be seen in Figure 29 and Figure 30, it is helpful if the icons of the buttons describe the node in an unambiguous way with as less detail as possible within a common look’n’feel.

Since the buttons are now overloaded with multiple meanings like “create annotation” or “query annotation”, we have implemented a transparent ring which shows the interaction context like ‘create’ or ‘query’ with an transparent colored ring. The color

itself is determined by looking at the first node in the interaction *path*.

4.10.3 Invocation and positioning

The ring menu is invoked via a gesture by drawing a triangle on the screen or by a speech command “menu” (section A.3), while the positioning is done via tapping on the screen. To further minimize the required UI interaction, we have not implemented a dedicated ‘close’ button. Instead, the ring menu fades out after a delay during which the user does not interact with the menu (keep alive). The ring menu is kept visible by either tapping and holding down the interaction device button anywhere within the menu or by hovering over the button controls in the menu. This keep-alive delay is configurable through the application settings dialog. As we will describe in section 5.3 we have adapted this technique when switching navigation metaphors via a menu of this type.

The interaction with the menu itself has been designed in an effective way where the amount of user “clicks” is reduced dramatically. The pointer’s traveling time required to invoke a command is brought to virtually zero, if compared with traditional interfaces where the pointer should travel back and forth from menus and bars placed on fixed position in the screen. This approach has been consequently followed throughout the dialog design, for instance the camera calibration tool (Figure 31).

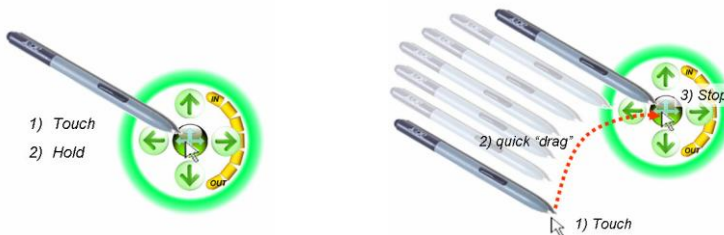


Figure 31. tap-n-hold keep alive (left), Quick drag activation (right)

Following this approach if the user releases the pen and slides it over the screen, yet maintaining the pen's tip at close distance from the screen, the menu will follow the pen's position and the relevant command is immediately invoked as soon the pen touches a button. It brings the major advantage of having the interface always close to the pen's tip with all the advantages already illustrated.

4.10.4 Enabling dialogs for interaction graph

The problem of embedding dialogs in the interaction dialogue lies in interfacing functionalities at the correct interaction stage and context. Reasons for using dialogs additionally to the ring menu lie in the necessity to specify detail which cannot be easily provided to the application otherwise. The application might require multiple choices or precise numerical input which otherwise could be stated only with difficulties. Thus, we have limited ourselves to using modal dialogs which are shown upon accessing specific nodes of the graph. The invocation of dialogs is not bound to the current interaction node itself, but it is encoded in the actual event handler. Figure 32 depicts the creation of an annotation, which is done in a dialog. An unacceptable constraint would be to allow only input via the buttons shown. To have a consistent interface, these functionalities have to be extended towards the use of speech and gesture being aware of the interaction graph. As shown in Figure 32, the annotation dialog is smoothly integrated into the interaction graph. The ring menu shows available nodes which can be reached following the outgoing edges of the annotation node. Clearly visible is the sharing of functionality between ring menu and annotation dialog through use of the same set of icons, which contributes further to a consistent look and feel of the application. However, not all functions have to be reachable through the ring menu. Instead, the annotation dialog offers additional functions such as emailing

annotations and changing the annotation types via the colored buttons to the right.



Figure 32. Consistent interfacing to the interaction graph with dialogs

It should be highlighted that still, all context dependent interaction of the graph is addressable. For instance, the user can navigate through the annotation using a horizontal line denoting *next* or *previous*, by the buttons of the ring menu or by speech.

Furthermore it might be more efficient to scroll through the list of annotations via a gesture to access the next or previous annotation as often used in Web Browsers.

Especially in the case of taking notes, the user needs to be able to sketch on the note as well as to transform the viewpoint in order to examine the object of interest. A straightforward solution to this problem of navigation on the one hand and interacting with

the scene on the other hand was integrated by offering two application modes which will be described in the following section.

4.10.5 Edit / navigation modes

To achieve a more complex behavior regarding the combination of navigation and the scene content, two modes have been integrated. When in *edit* mode the user is enabled to interact with the GUI and the scene itself while the dedicated *navigation* mode allows transforming the users view. It is possible to switch between the modes by either using the buttons in the upper right corner, or by voice. These application states assure that switching between the modes does not interrupt the workflow. For example, going from edit mode to navigation mode and back continues the current interaction context and it allows for navigation at each interaction step.

4.11. The need for application feedback and support

When working with multimodal applications, it is often a serious problem for the user to know about the available interactions. Because the user designs his/her interactions, it requires extended time for familiarization during which he/she needs to learn how to use fast and efficiently not only how to combine interactions like voice, gestures and dialog, etc., but also how to use the application itself. For this reason, the user should be supported at all times by providing means of feedback for (1) the current interaction path and (2) the available interactions, which are potentially predictive.

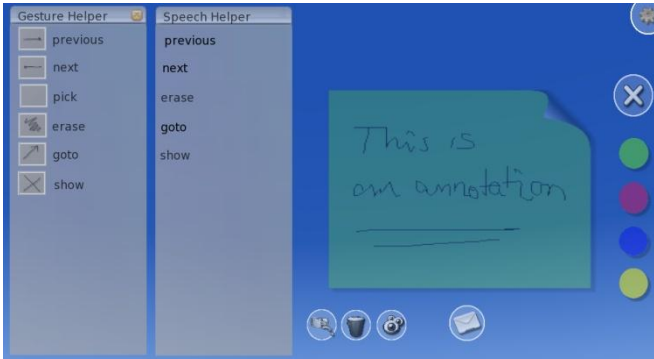


Figure 33. Gesture and speech helpers

The technique developed uses helper dialogs (Figure 33) which show the allowed gestures commands on each step of the interaction. This is done graphically by showing the available gestures and the corresponding commands. Similarly, available voice input is shown in a separate dialog. When issuing voice commands, the ring menu becomes visible for a brief moment of time and then fades out, showing the active interaction node in the large center button of the ring menu.

Through the use of Text-To-Speech (TTS) at configurable level of detail, audio feedback can be given to the user. This is particularly helpful to remind the user when data arrives in collaborative setup at the local application instance. For example, when a participant of a design review session creates an annotation, this event is announced to the local user via “*new annotation arrived.*”. Another example addresses the remote loading of a model: “*model [X] has been remotely loaded.*” Although sound events could be recorded in advance for playback, TTS has proven to be very helpful when dealing with placeholders. For instance, a remote material change is announced to the local user like “*Material of part *wind shield* on model *lotus* has been remotely changed to *glass*”.*

4. Multimodal interaction techniques

During the several user tests illustrated in chapter 7 “Assessment and validation”, it has been proved that visual feedback of gestures is of high importance. For this reason, all gestures and sketches are drawn as thin, dashed lines on a 2D overlay view-
viewport.

4.11.1 Multimodal interaction authoring

Editing interactions directly in the persistent graph text file (GraphViz *.bgl, (75)) yields satisfactory results for small graphs and with users with a deep insight into the application and its functionalities. However it is rather inconvenient to the targeted group of design reviewers. Interaction nodes and edges have to be creatable, editable and understandable with ease. For this we have decided to adopt the distribution to facilitate modifications of graphs like create and remove vertices, altering vertex names, etc.

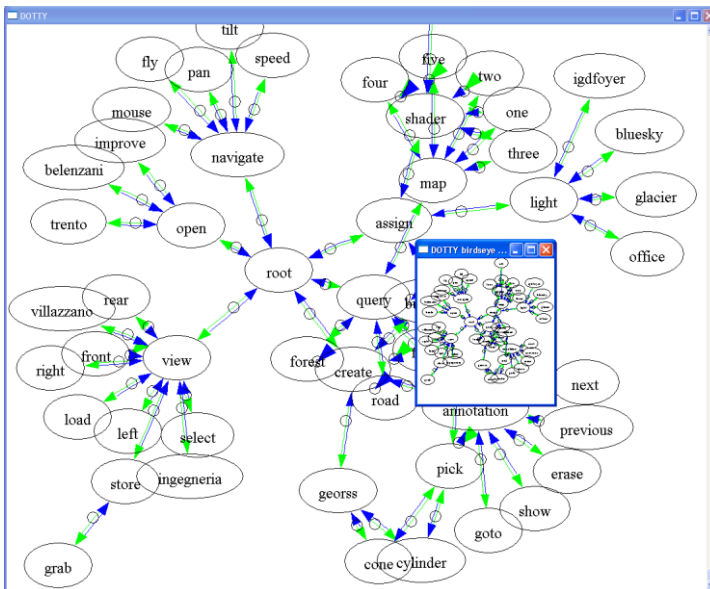


Figure 34. Graph editing with GraphViz

Another, more advanced tool with graph editing capabilities such as *DynaGraph* (76) can be used as well to insert a new node into the graph (see Figure 35, orange node). Here, the user inserts a new node into the graph, enabling him to access a mouse navigation scheme during a review session.

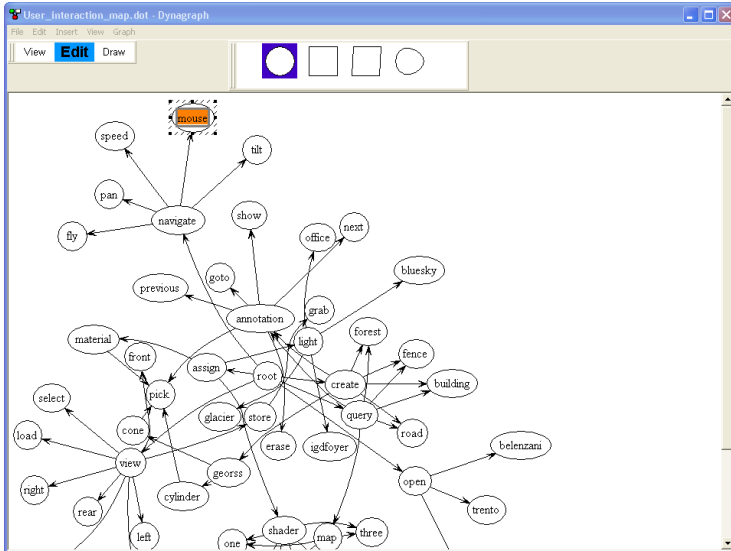


Figure 35. Graph modification with Dynagraph

4.11.2 GUI editing with CEGUI designer

One of the peculiarities of the technique developed is in that it allows for a fully customizable GUI from outside the application. Layouts can be easily re-arranged, assigned transparency, resized, etc. which contributes further to the customization of the application. We have used CEGUI (77) in order to provide a rich set of elements to be used on the OpenGL overlay viewport of the application. The Graphical User Interface (GUI) takes advantage of skins supported by CEGUI to improve the users perception and preferences. The various layouts are kept in *.xml files and are

4. Multimodal interaction techniques

loaded during the initialization phase of the application. This way, the application can be internationalized with ease, just replacing the layout file with localized versions without the need to recompile the application.

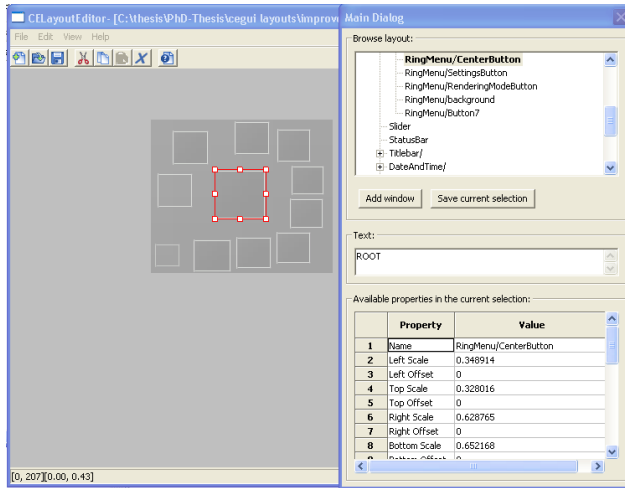


Figure 36. Editing of the ring menu inside CEGUI

Figure 36 depicts a session of the CEGUI Designer in order to change the visual appearance of the central ring menu. Buttons can be moved and rearranged and further properties such as transparency, their colors and element background are editable in the window to the right. For instance, if the user preferred to have a central menu in a linear form, this could be easily provided.

4.12. Navigation techniques

Providing means for proper navigation represents an essential requirement for the user to be able to explore the virtual and augmented scene. Since this is the base case for both the automotive and architectural scenario, the interface proposed provides both well know approaches like flying, walking and

examining as well as new ones, specifically though to make interaction easier. Flying and examining are dedicated to the navigation within the purely virtual space since the observer is detached from his/her physical location. This allows him/her to reach locations normally not accessible during the reviewing session. Walking is used in both virtual and augmented scenarios, allowing the user to explore the scene in a natural way. Fly-through and the possibility to zoom and examine building parts temporarily (camera navigation) have been adopted because there are no other means to reach distant locations. Exceptions occur when the user gets lost in the scene by losing his/her orientation. This case will most likely happen in the architectural scenario due to the model dimensions and the closed nature of the surroundings. A way to solve this problem is to provide an overview map which is overlaid with the display.

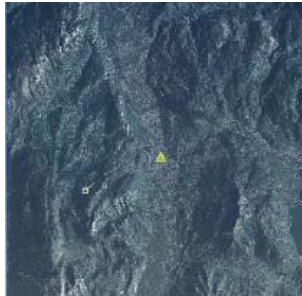


Figure 37. World in miniature

Figure 37 shows the top view over the reviewed scene. The triangle denotes the location of the reviewed product while the white rectangle and yellow vector represent the user's location and viewing direction.

It must be highlighted that an abstraction from input devices and mapping to common navigation schemes is of highest importance. Thus, a way needs to be found which allows for a broad and extensible range of input devices, but also for enabling various

product scenario-dependent navigation schemes. For this reason, the input is mapped to appropriate scenario-specific navigation metaphors as shown in Figure 38. The input of each distinct input device is mapped to normalized screen coordinates $(-1..1, -1..1)$ and then to viewport coordinates which forms an unified input to the offered navigation schemes of VTP. The peculiarities of each device, such as the definition of a dead-zone for blending out hand trembling, are paid attention to in the normalization step which defines a mapping scheme for each distinct device.

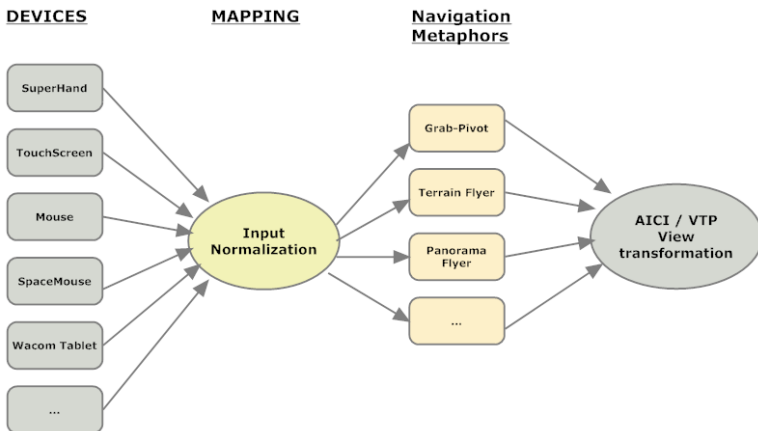


Figure 38. Mapping to navigation schemes

4.12.1 Superhand

Besides supporting traditional interaction techniques a new approach has been developed to form novel forms of interaction. Given the high priority of a non-fatigue navigation through the scene and around the reviewed car model, we have adopted a 3DOF gyroscope and accelerometer. This is XSens Motiontracker which has been adopted for controlling the virtual camera in a convenient and natural way. It delivers accelerations and orientation along the 3 axis of rotation of the tracker attached to the back of the hand via a glove.

In the approach proposed the user navigates the scene by bending his arm. To switch between navigation modes the user can speak a command, press the relevant button or simply shake gently his/her hands. A fast movement of the hand to the right or left triggers a change of the active navigation scheme in a cyclic manner whereas the natural orientation of the hand orients and controls the virtual camera (Figure 39). The system provides feedback by speaking out the new navigation mode through TTS to notify the user about the currently active navigation.

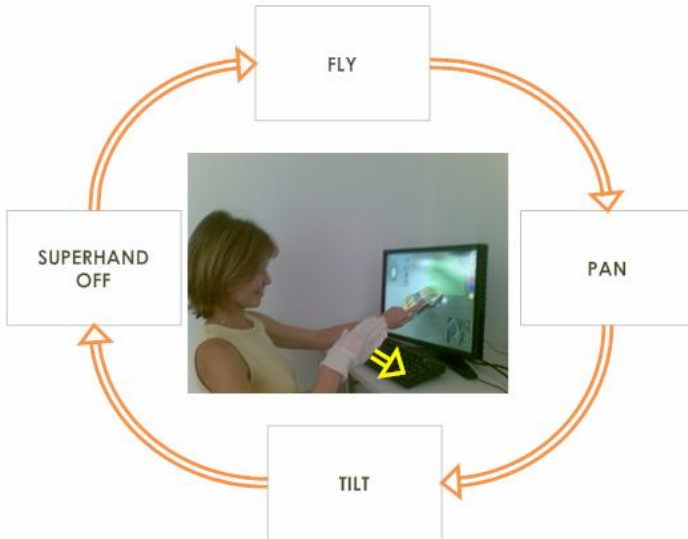




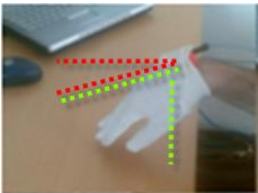
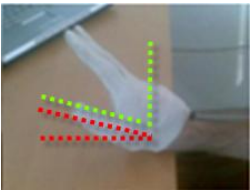
Figure 39. Two handed input and change of navigation

It should be noted that the orientation of the virtual camera is not hardwired to the orientation matrix received from the motion tracker. Instead, we map the inclination of the trackers' axes to screen coordinates, and use it as input for the configurable terrain navigators. A strong discomfort for the users would be an immediate response of the system due to hand trembling. For this reason, we have defined a dead zone of 15 degrees which blends

4. Multimodal interaction techniques

out marginal hand movements. Table 1 shows how the actual hand movements affect the user's navigation.

Table 1. Mapping of gestures to navigation

gesture	fly	pan	tilt
	turn left	move left	-/-
	turn right	move right	-/-
	accelerate	move down	look down
	decelerate	move up	look up

The effectiveness of the technique has been tested during a user test session. The users who undertook the final testing session at Elasis, Naples found it to be effective over their traditional ways of navigation and they have clearly stated that this required less fatigue since it only involved bending the hand without the need for holding their forearm up. This way, movements to achieve a desired view on the reviewed model have been reduced to a bare minimum.



Figure 40. Test of “Superhand” interaction technique during the final user test, Elasis, Naples

4.12.2 Ellipsoid navigation

During a design review session, it is of high importance that the exterior views on a product can be reached in an efficient, fast and continuous manner where the product is always in focus. Specifically during a discussion of a product, users tend to discuss an exterior part in the context of the complete model. This behavior cannot be achieved using a traditional flying, examining or walking navigation scheme. A solution would be to use a grab-pivot navigation, but then, the user would leave the global scope of the model. However repositioning the pivot many times would

make the interaction a tiresome one. For this reason, we developed an “ellipsoid navigation” where the user navigates on an ellipsoid which circumscribes the object of interest (Figure 41).

The positioning is effected via tapping or moving the cursor on a dedicated window; while the position is calculated based on the 2D projection of the ellipsoid. Thus, an easy access to any viewpoint and direction at a specific distance is given. Nevertheless, the most commonly predefined views (left, right, front, rear) can be reached through normal buttons at the right side of the window. As for all CEGUI elements, the window is transparent, moveable, and can be additionally collapsed and inflated which guarantees the possibility of having a maximum view on the scene and model. Similar to all other navigation schemes, the viewpoint (when in shared mode) is shared between the users.

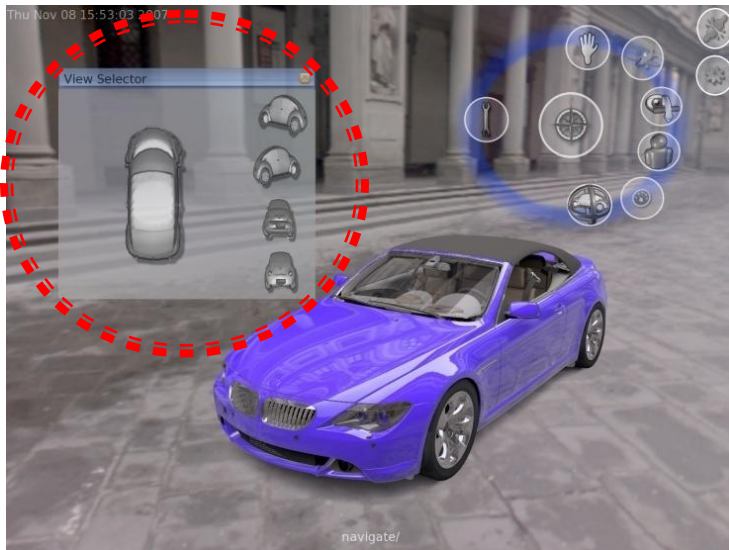


Figure 41. Ellipsoid navigation

4.12.3 Multimodal navigation

The ellipsoid navigation has not been designed to be exclusive. Instead, we have allowed the user to freely combine it with the (terrain-flyers) and the “Superhand” (section 4.12.1) navigation, thus allowing a two-handed, sequential interaction as can be seen in Figure 42. Here, an Elasis user uses the (dominant) ellipsoid view via a Wacom tablet to approximately position him within the scene and the touchscreen to refine his view.



Figure 42. Two-handed interaction with Touchscreen and Wacom tablet

In fact the system developed allows combining multiple modalities at the same time (multi-multimodal) and in distinct configurations. Especially in the field of design reviews, it is of high importance to access navigation schemes and command invocation functionalities in an alternating or simultaneous way. As can be seen in Figure 39, the user invokes a command by using a gesture on the touchscreen while still navigating using a motion tracked hand. It should be noted that at this step, interactions following the graph approach are still enabled.

4.13. Conclusions

The interaction promoted by the application introduces a novel approach to combine efficient navigation capabilities with highly customizable multimodal interaction techniques tailored to design reviews in a virtual and augmented reality environments. Due to the heterogeneity of design review participants, distinct modalities and in fact the interaction dialogue itself needed to be highly configurable according to each user's specific preferences and scenario definitions. This was achieved by separating the user interaction from the application's behavior. The interaction can be designed according to each user's needs using a visual editor.

Further we have offered the user multimodal sequential input for controlling the applications behavior. This has been extended to allowing parallel navigational input stemming particularly from the tracked hand navigation metaphor. The graphical user interface is customizable which is beneficial when changing the resolution of the application's viewport and for adapting the visibility of control elements to show or hide functionality from the user. It has to be highlighted that user profiles can be realized in configurations stored outside the application. For this reason we have been able to regionalize the program to using different languages with respect to speech input and dialog content.

5. Collaboration and data integration

5.1. Introduction

We have followed the peer reviewer paradigm where one master reviewer is enabled to lead the reviewing session with respect to navigation. A local and a shared mode allows the user both to follow the peer reviewing process, but also to leave the design review session temporarily for exploring the model separately from the group and to take for example annotations on the model. All information from the user is shared through propagating them (annotation/change of materials etc.) to the other users in the reviewing group via a communication backbone based on XmlBlaster (78) and OSGA (79). Since our application is intended for a range of distinct scenarios such as automotive design review, architectural design review and large area surveillance, a way to communicate and, more importantly, organize the data efficiently is crucial for a true distribution of the clients. On this problem we have defined a communication protocol based on XSD schemes which have allowed for the implementation of XML message hierarchies. A general interface to all messages includes information about the ownership and timestamp and it is implemented in all derived schemes.

5.2. Sensors

In order to bridge the gap between 2D GIS and a 3D interactive component, a set of spatial data described by a sensor domain has been integrated using an analogously layer-based approach, with layers providing thematic access to the actual dynamic 3D content.

A sensor manager forms the central management facility for the sensor data and their visualization. It holds an arbitrary number of layers accessible via a dictionary of domains and sensor layers.

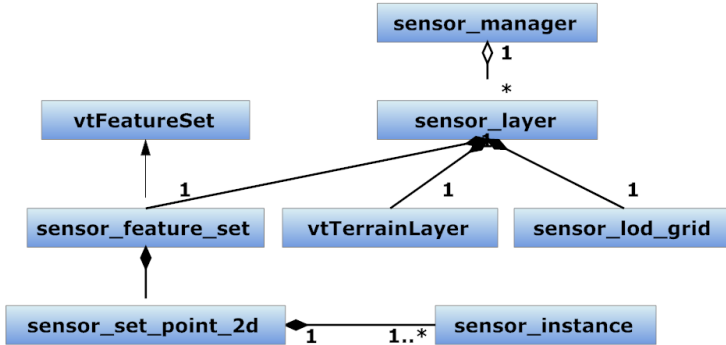


Figure 43. Sensor containers

The layers consist of a sensor feature set with all sensor data of a specific spatial dimension (Figure 43). Further, it provides basic query facilities like “*find the closest point to*” and “*find all points at location*”. The application provides the content interface to perform common VTP operations like tagging the underlying feature set with rendering and contextual attributes. Further, a level-of-detail grid gives access to the actual location within the scenegraph. Since VTP implements a level of detail grid (LOD grid) for its terrain positional 3D terrain cultures (like streets and buildings, and vegetation), similarly we have provided a LOD grid for the visualization of sensor data. The visual representations of dynamic data are inserted into a LOD grid which determines the according terrain LOD cell, where the sensor geometry is placed onto the terrain. Only content present within the current range of sight dependent on the viewpoint of the user is rendered, thus increasing performance and displaying only content within the current area of interest.

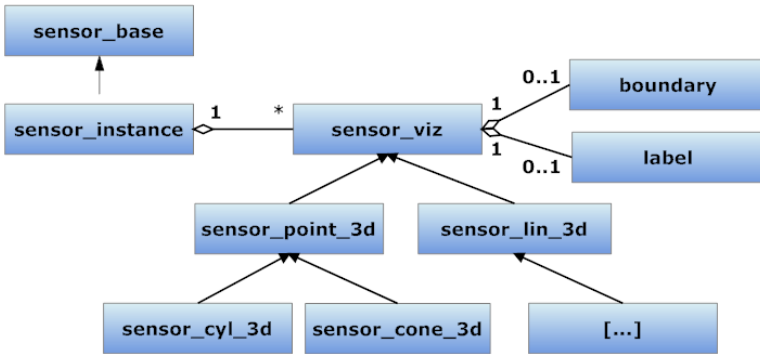


Figure 44. Sensor class hierarchy

Figure 44 defines the class hierarchy in order to separate data and their visualization. The class `sensor_instance` extends the basic sensor data-holding class by providing the means for the visualization and giving access to the sensor sub-scenegraph. Each visualization component has a label to be displayed on top of the sensor, a bounding geometry and a representation of the sensor value itself. A sensor is enabled to have multiple representations according to the context of the current examination. In Figure 46, a sensor is shown using a mapped color gradient and a height modification to represent a value. Further, the station's name is shown always residing on top of the cone.



Figure 45. A view of a sensor generated from a web data source

It should be noted that SensorBuilder acts mainly as a managing application. The sensor observers can be distributed as well, for they are simple applications that publish sensor messages to the sensor channel. To show this, an observer which retrieves geo-referenced attributes of streets from a website, has been implemented (Figure 45).



Figure 46. Update of values

5.2.1 From data generation to visualization

The exchange of dynamic, time varying data requires specific communication paradigms such as support for asynchronous initialization and release senders as well as the necessity to determine the systems which is the destination of the information. To tackle these issues, we have decided to use a message passing middleware (MPM) which follows a topic publish/subscribe

approach. In our system, the management of the dataflow coming from the sensor is performed as presented by Figure 47.

First, the SensorBuilder generates sensor messages with updated sensor data. These messages are then published to the communication backbone using a channel identified by a sensor topic which identifies the type of message being sent. Finally, the communication backbone server redirects the messages to all the clients interested to that specific type of sensor data, i.e. any system which has subscribed to that sensor specific topic.

5.2.2 Sensor message definition

The GIS message definition relies on plain XML syntax and it defines the sensor information exchanged by all client applications. All messages are derived from a common structure which holds attributes such as author, system origin and its location.

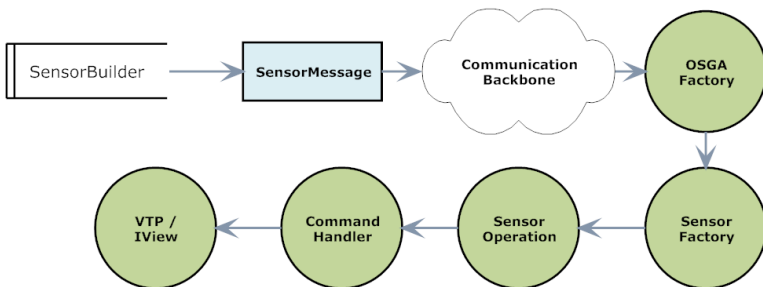


Figure 47. Dataflow

This enables session management for multiple users, since it identifies each user, place and system uniquely. This common structure, which is used by all the messages of our communication backbone, has been extended with the sensor-related information, associated sensor operation and a time-stamp.

Currently, a sensor is defined by having an unique identifier, the value domain, a boundary influence area, a station name, a sensor domain and most importantly the value itself.

However, the XML nature of the protocol will allow a simple extension to introduce new attributes. When messages are sent, each message gets a time-stamp assigned to it which is important to enable further dissemination of the message or to identify the correct sequence of the work-flow during a given virtual session. Finally the message operation defines the possible actions that our system can perform over sensor data. Specifically these are one of the following types: `CREATE_SENSOR` to support creation of a new sensor, `DELETE_SENSOR` to enable deleting sensors and `UPDATE_SENSOR` to change any attribute of the sensor.

5.2.3 Communication initialization

While navigating and interacting with the virtual scene, clients are able to send and receive individual messages containing sensor data to the communication server via a communication channel identified through the relevant topic '`SENSOR`'. Other topic-related visualization and interaction means are also supported by our network backbone in order to enrich the data exploration and the collaborative functionalities. Any a given client does not need to know about the number of clients present within the working system. The information exchange is simply done by publishing and processing only the input data associated with a topic. This approach makes clients independent and it increases scalability, re-configurability, and reuse of components. However, this approach emphasizes the need for a well-defined communication protocol according to which rendering clients only need to implement parts of the communication protocol which is relevant to them.

For the purpose of our work there are two types of topics available: a general communication channel for all sensorial data

'SENSOR' and a field-specific topic related to the sensor domain, which is provided inside the message content. This way, any client application, which is used to render a different instance of the environment, knows the presence of sensors through the arrival of data on the sensor communication channel. When a GIS message with a new domain arrives on the client, a new sensor layer is registered on the client system. Each sensor is identifiable through the usage of a Unique Universal Identifier (UUID) which is created each time a sensor is added to the data managing and manipulating application, SensorBuilder.

5.2.4 Handling sensor message exchange

The sending and the creation of sensor messages takes a straightforward approach: once sensor data is added, modified or deleted, a sensor message is created and it holds the description of a sensor and its attributes. During the XML serialization only the sensor object itself needs to be passed to the message object. In order to abstract future extensions of our sensor protocol, all the messaging data use XSD schema definition to automate message translation. Regarding the reception of sensor messages, the deserialized sensor information is delegated to a sensor factory singleton on the client side of the OSGA communication backbone.

For each particular message, a sensor operation is created and then queued on the application event loop as a command. The operation is processed during the idle time of the application. This ensures safe access to the local OpenSG context in order to allow smooth visualization. All operations are performed in a non-blocking manner. Hence, users do not experience interruptions neither in their interaction experience nor in their workflow, Allowing smooth navigation and interaction with the scene. All handling processes rely on no knowledge about the sensor data content besides the message topic and its correct operation type, which is extracted from the message.

As well as any other communication operation inherited from the original communication backbone, the sensor operation is determined during the serialization of the message and is bound to a specific application handler through a global reception callback. Multiple handlers, besides the default handlers, can be linked to the command using function binders, specifiable anywhere within the application, even in- place as stateful function objects. Thus, (un)binding of handlers at runtime enables the application to achieve a more complex behavior according to the applications state. This makes it possible for example to enable objects within the application to react individually to a received message. The several components described for the message handling are illustrated in Figure 47.

5.2.5 Sensor data creation and management

We use the concept of message exchange in order to allow two distinct approaches: the task of sensor observers is basically to retrieve observation values, which are published to the SENSOR channel. These data can be considered *raw*, as there is no way of manipulating fields of transmitted data. It is evident, that an additional tool for editing and redirecting sensor messages in near-to real time fashion is needed to control the data to be inserted into the viewing/interaction component IView.



Figure 48. Managed data pipeline

As shown in Figure 48, we allow an operator to be inserted into the flow of data which redirects manipulated messages to the SENSOR channel.

5.2.6 SensorBuilder: the graphical editor

The standard VTP graphical editor, used for viewing and processing geospatial data (VTBuilder, 2D), has been extended by the authors through the development of the so-called SensorBuilder. This tool allows the definition and graphical manipulation of sensor data, boundaries and attributes. In fact the original VTP editor, which is a standalone component to create and manipulate data and achieves, does not allow communicating at runtime with the client application. Because of the time-dependent nature of sensor data, a more flexible solution was required capable of managing the publishing of real-time information.

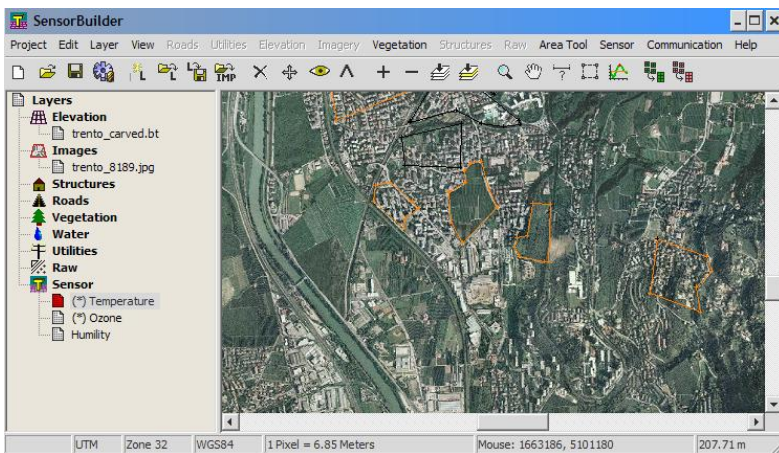


Figure 49. Sensor boundary creation. The sensor domains are automatically inserted as layers into SensorBuilder and data are updated in realtime

Following the architecture of the standard VTP graphical editor we use a layer-based architecture in which data are classified by the user into a specific context. Layers are thus assigned a context like water, structures, elevation and imagery. The possibility to create multiple layers and data sets for each domain gives an excellent opportunity to extend this concept to the administration

of sensor data. For each topic, a dedicated layer is created within the sensor context. It can draw its data from multiple sources. First, the location and boundaries of a geospatial sensor needed to be defined and visualized. This is currently possible either via a WFS request, an import via an ESRI *.shp data or a manual editing in the editor, as shown in Figure 49. Before proceeding to the graphical representation level, the sensor location or area has to be linked to a time-dependent or static field. Then the selected numerical attribute is continuously requested at specifiable intervals and in case of changes published to a sensor channel along with additional attributes like observation station, measurement unit and geographical position.

As mentioned earlier, this component is only aware of the sensor data and it does not need to know about other clients how have subscribed within the distributed virtual session. The strength of building on top of the standard VTP tool is in the support for import and export from/to common GIS formats and software. For example, the authors have showed how an ESRI shape file provided by a local authority can be used in order to use lake boundaries as boundaries for a sensor definition, linked to manually specified observation value. As depicted in Figure 50, the user can choose which observation value and station links to the boundaries of a sensor definition.

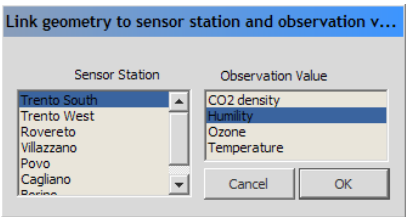


Figure 50. SensorBuilder station and domain assignment

5.2.7 Managed sensor stations

The concept of administrating sensor stations via the SensorBuilder relay application requires us to support two distinct types of messages:

1. Direct sensor messages (SM)
2. Managed sensor messages (SSM)

After the user logs in to the communication backbone, SensorBuilder listens on the SENSOR channel for the presence of sensor stations and sensor domains and it keeps track of changes by storing in a dictionary, for each station, available data sets. In case the user links an observation value, e.g. humidity, to a sensor boundary, SensorBuilder acts as a managing component for this particular station #1('Trento South') and the observation chosen value domain. The pair (station, value domain) is then marked as managed on the sensor observer. This is done (Figure 51) via publishing a 'marking message' on the sensor channel which carries the sensor identifier, the observation pair and the operation tag 'MANAGE'. Upon receiving, station #1 will then mark all future outgoing sensor messages related to this particular pair as marked (SMM). These marked, managed messages are then disregarded by all connected clients. However, unmarked observations of station #2 via unmarked messages (SM) are still integrated. The SensorBuilder thus is assigned the task to send out unmanaged messages, updated with the assigned boundary and/or sensor specification. These are then handled by rendering clients like the data coming from station #2, and integrated.

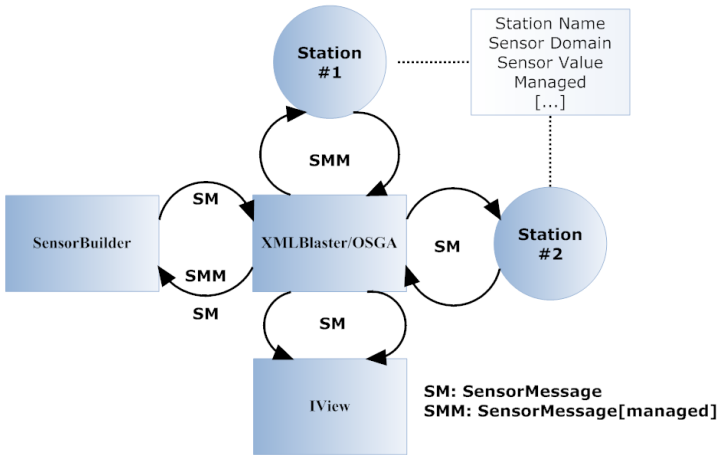


Figure 51. Managed sensor stations

This way, we allow both direct input from a sensor source and additionally supervised, modified input via an administration component.

5.3. Collaborative annotation and navigation

5.3.1 Introduction

The very nature of design reviews lies in a collaborative scenario where several users are able to interact and conduct design review using a variety of devices, such as a touchscreen and Tiled Display System (TDS), sharing annotation, navigation and minor model and scene modification abilities. Annotations allow the user to attach his/her comments and thoughts in several ways to a model entity. Thus, they possess the character of an addendum to define what cannot be expressed in any other way, by capturing design intentions, modification suggestions and by documenting the reasons for the alterations applied to the model. Annotations

are not specialized to the automotive or architectural scenarios for they are applicable in both industrial domains.

A group of evaluators and engineers is standing in front of a Tiled Display System (TDS) where an instance of the system is rendered at very high resolution (realtime 18 MPixel - stereo). Since the system is inherently distributed and collaborative the two set-ups are sharing the same virtual content through the communication backbone. Likewise each reviewer in the panel can interact with the very same virtual content, through an instance of the system running on his/her machine which is connected through wireless LAN with the units running the TDS (see section B.1, Indoor design review setup).

Our approach to collaborative design review is based on the assumption that a peer reviewer steers the reviewing session and for this reason receives privileges over common users. The master reviewer should exclusively control the shared navigation. The interface has been designed to support collaborative navigation and annotation of a 3D scene. It focuses on the interface designed to let reviewers independently interact with a personal device (i.e. TabletPC, touchscreen, etc.) during a shared review session.

As underlined by Hong et al. (2), annotations can be considered a by-product of the user's thought and the importance of sharing annotations between members of design teams has been highlighted by a number of researches (3). In our application the message-distributing infrastructure is used to generate notes during the design review by the members of the panel. The review takes place in front of a high-resolution Tiled Display System (TDS) which is used to share and comment the design product. Each reviewer has a machine running an instance of the system. The point of view of each user is synchronized with the TDS., our approach becomes highly scalable using the MPM based communication backbone.

The communication infrastructure broadcasts the information on the view transformation and it sends them to the TPCs so that synchronously each user's TPC can render the same point of view (at much lower resolution) of the tiled display. The synchronization between the TPC and the TDS' point of view is shown by the "shared mode" symbol at the bottom right of the window. The annotation itself is represented through a panel facing the user which represents the "post-it" note (4) attached to the geometry (Figure 52). For the geometry to be visible the note is placed over the current geometry and rendered in overlay to the scene's image.

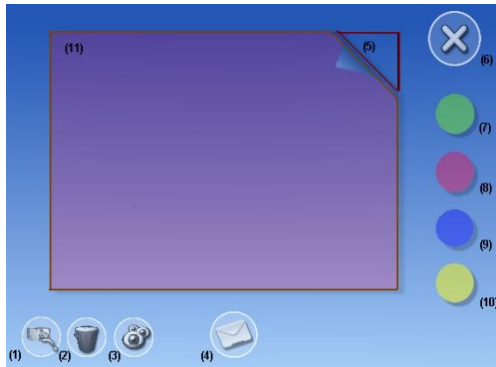


Figure 52. A screenshot of the annotation tool: (1-4) nodes of the interaction graph, (5) note dragger, (6) close note, (7-10) change of annotation type, (11) scribbling area

Every time a new note is added, modified or deleted its information is sent to the other users through the communication backbone. We made annotations entirely concurrent. For this reason all users can sketch, change annotation type or replace the annotation marker at any time. Next to the note there are buttons for performing annotation related operations like changing the annotation type (modification, problem, urgent, design) or going to the point of view at which the note was initially taken. An important element during reviews is the ability to discuss jointly

an issue. For this reason, the user is enabled to broadcast in a show-to-all manner to all connected participants. Further, the user can send the annotation plus a screenshot where it was taken via email to other, to non-participating colleagues.



Figure 53. A view of the embedded email client

5.3.2 Synchronous review

The application offers a synchronized mode for a conventional approach to design review: one peer reviewer broadcasts viewpoint and interaction events to the passive participants of the distributed review session. Thus, the viewpoint is shared on all display devices (clients) and this allows for a collaborative discussion by the participants. The viewpoint of the master is broadcasted via an underlying service continuously: no matter which navigation scheme is currently used as described in section 4.12 the camera transformation remains synchronized at all times.

5.3.3 Asynchronous review

At certain points during the review, a participant wants to scrutinize a certain model entity in-depth. Going out-of-sync from the main review process, the participant is enabled to temporarily

leave the group by changing his/her view to take annotations over the model of interest and to perform other local operations like applying materials or changing light conditions. While the group continues the normal review, the single user keeps on receiving all communication messages. In this “local mode” the user can perform all operations as he/she would do in a non-networked session. It is worth underlining that the absence of synchronization refers only to the point of view being rendered on the TDS and on the user’s machine because the content of the scene is always synchronized among each instance of the system. When done, the user goes back to synchronous mode and he/she ideally rejoins the group by getting in sync with the TDS’ point of view.

5.3.4 Collaborative navigation

As mentioned earlier a key feature of the application is its ability to share a view on all connected clients. This is achieved in the following way:

As illustrated in Figure 5 “Event handling mechanism” artifacts are used to manage events. Each assigned artifact operation performs actions based on these events. We therefore define a virtual artifact which does not have any physical device attached to it. Instead, the operation is executed continuously and publishes the transformation matrix of the camera during each execution. A direct advantage of this approach is the easy configuration within the framework. It is only necessary to include an `OverrideNavigatorOperation` in the configuration file to enable a broadcasting of navigation which is usually only done for the peer reviewer.

Further we have made ourselves independent on whatever navigation scheme physical device or in case terrain navigator the master reviewer is currently using: The same viewpoint is shared across all connected clients. We refrained from using multicasting

offered by OpenTracker due to lossy communication. Instead we have used, as for all communications, XML messages for the transformation propagation.

5.4. Data integration

Especially in the field of architectural and environmental design review, additional scene content becomes an important factor. We thus seek to integrate data at runtime from various sources. For this reason we have chosen a Web Feature Service as a primary data provider which draws features stemming from a PostGIS(80) relational spatial database as illustrated in Figure 69.

Although a dynamic retrieval of elevation from a Web Coverage Service (WCS) and image layers from a Web Map Service (WMS) have been implemented, we have constrained ourselves to sustain focus on the interaction scope of this work. Instead, we have used a pre-set arrangement of elevation and satellite imagery to have a discrete scene at hand, which is furthermore precisely replicable in following review sessions.

At various stages, it becomes necessary to review the construction or area of interest in its environmental or social context. For this reason we have integrated the ability to change the underlying terrain textures in order to superimpose maps onto the terrain.

5.4.1 Querying a WFS

The requesting of features is accessed through embedding nodes and edges within the interaction graph (Figure 54) analogously as described in (Figure 22).



create(voice) - georss(dialog) - pick(tap, gesture)

$$query(dialog) - map(voice) - forest(dialog) - pick(gesture, circle)$$

The retrieved GML polygonal data of the feature type is then interpreted as describing a 3D entity, for instance streets, railroads and houses as shown by Figure 55. It should be noted that the visualization as 3D entities remains optional. The features can also be draped in the form of polygons onto the real terrain.

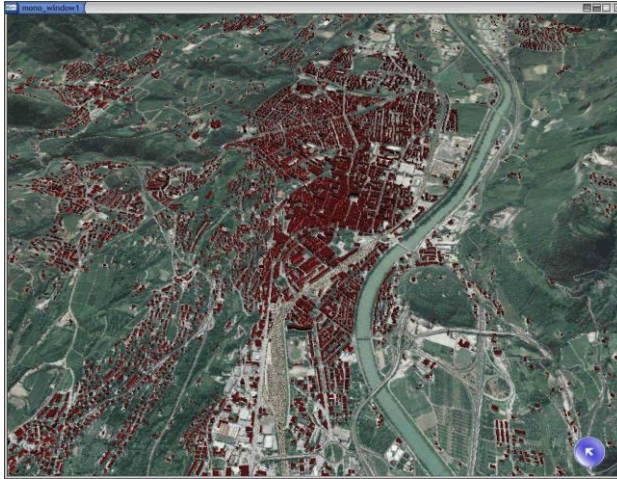


Figure 55. An example of a GIS scene where terrain cultures have been retrieved from a WFS at runtime

5.4.2 Communication with the WFS

Collaboration is enabled by the exchange of all actions executed by a user, such as e.g. performing queries to a Web Feature Service (WFS). Thus, we created the new communication channel `WFS_REQUEST` and new message definition `WFS_MESSAGE`. A WFS query message carries the base WFS URL, its version, the typename and the bounding box of the query. According to the typename, an appropriate visualization metaphor is chosen to represent e.g. a 2d polygonal area within the query box as a forest for the feature type `topp:forest`. This ensures synchronicity amongst all instances of the client application, without the need for transmitting particular geometries (that is, the result of the queries) over the communication backbone. The message exchange is simple yet effective: once a query is performed on a client, the according WFS query fields are embedded into a `WFS_query` message and published to the WFS channel topic. Except for the administrative component (SensorBuilder), the

repository and other instances of rendering client receive this message and query the WFS for the geometrical features and their attributes stemming from a PostGIS database. Since this puts heavy load on the WFS, future developments should implement a more efficient approach to include geometrical data in messages and to ensure that the feature data set is only retrieved once. Due to the similarity to sensor messages with respect to handling and exchange, we omitted here a detailed description.

5.5. Conclusions

This chapter proves the feasibility of a visual integration of distributed sensor data structure into VR- and AR-based client applications. The main features with respect to time-dependent data integration and visualization are:

- Distribution of sensor stations and graphical administration.
- Collaborative visualization of sensor data.
- Continuous integration of sensor data.
- 3D context-dependent visualization of WFS data.

In our application, a unique SENSOR channel was created and the types of sensors are only distinguished through an additional attribute. Future developments should focus on how to use dedicated channel topics to transmit data of various domains. Our system uses prototypically defined sensor stations, which are not described appropriately with respect to their dimensions and measurements. Because of this, a further improvement should use SensorML as a standard xml exchange format for sensor data. SensorML is part of the OpenGeospatial Consortium evolving standards for Sensor Web Enablement and allows for a detailed specification of sensors and processes.

6. Augmented Reality for large environments

6.1. Introduction

The last years have seen the birth of several 3D GIS applications which allow interactive access to geo-referenced data within virtual environments. However, these applications often lack important means of interaction, user navigation, collaboration and, most importantly, the integration of geo-referenced features and areal data. Numerous currently emerging 3D GIS applications such as Google Earth (81) and NASA World Wind (82) offer elevation, imagery and content using advanced streaming techniques. These desktop applications follow a strict client-server approach, which avoids a truly distributed system of largely independent users.

Collaboration of users is restricted to adding new content such as markers, 3D buildings and photos and can be changed only by the creator. Although there are attempts to provide multi-user capabilities such as Unype (83) by combining the viewing application with a VoiceIP telephone directory according to the users' location, a real interaction and embedded exchange of terrain data amongst the operators does not take place. Still, a shared navigation or a simple broadcast of the user's point of view is not possible. A support for specific terrain navigation schemes such as a panorama, terrain or a precise grab-pivot flyer has not been realized. Rather, the user navigates in a click-and-zoom, or pan-and-zoom style.

The content integration of these systems is rather marginal: there is no way to create terrain cultures within the applications. While Google Earth still provides means for importing geo-referenced data in a file format based on GML, NASA World Wind is limited to display only the 3D elevation and superimposed imagery. Also, Google Earth does not allow the integration of 3D cultures other than building structures. These applications can be only called viewers of geospatial data because of these shortcomings. Moreover, the provision of adequate geo-functionalities is a crucial point when developing GIS applications according to technical standards. An inherent severe limitation of all virtual 3D GIS applications is the abstraction away from real world conditions. The environment and surroundings are seen in an idealistic way not capturing present obstacles or unexpected conditions. For instance, when planning a new construction, outdated information, the presence of vegetation or a change of topology can severely affect the planning process. Also there is currently no way to integrate geometry which changes over time or to visualize volumetric data.

6.2. Large area surveillance

The use of Augmented Reality, whose scope is beyond traditional virtual environments, based on outdoor, large scale environments has added a new level of complexity to the interaction process and to the corresponding user's perception.

Unlike most of today's outdoor augmented reality applications, we do not use optical tracking to align the virtual with the real scene which would limit an outdoor GIS application in several ways. First, the demands on the image acquisition and tracking capabilities would be very high in order to track specific terrain features. This is excluded by the demand to remain mobile, which in our case means an easy and fast setup of the system. Second,

the viewing distance in an outdoor GIS application is extended by far over traditional augmented techniques (several kilometers). Further, there is the need for an underlying elevation heightfield to properly align and superimpose terrain cultures, features and to overlay map data onto the real terrain. Additionally, the precision of the underlying elevation grid has to be specified according to terrain topology and scenario requirements.

It has been made possible to navigate 3D scenes where the user is blended within geo-referenced context. The interaction has been designed to take advantage from the fact that the user can navigate with his/her body within the environment and the 3D geo-located maps. Interaction within 3D map-based collaborative environments becomes a crucial factor when several users, both indoor and outdoor have to collaborate on specific tasks and exchange information on the environment which surrounds them.

It is clear that interaction with geo-located 3D maps requires explicit interaction metaphors which can be adjusted to a wide range of specific issues. First, the interaction process must address the issue of navigating a 3D geo-referenced environment. It must also provide efficient access to further information available from it, such as thematic maps, features on a geo-database, geo-referenced files the file system, online data or dynamic data. This scenario becomes even more complex when addressing network-based collaboration with different mobile/on-site set-ups when providing an adequate sense of collaboration between users connected through the network becomes a crucial factor for success. In this case providing efficient access to information is of high complexity, since this requires using a wide range of modalities, that go beyond traditional mice and keyboards. A major problem of current augmented reality outdoor applications persists in the geo-referenced placement of cultures on-site in an efficient manner using adequate interaction techniques.

6.3. Aligning virtual and real world

The main problem in Augmented Reality applications can be identified by achieving the highest possible precision in matching the virtual and real world. In fact, two distinct cameras are used: the real camera captures the video images used as a texture for the background viewport while the virtual camera controls the view on the virtual scene. In order to achieve a credible augmented scene, their difference in their transformation needs to be minimized. Unlike traditional optical based tracking mechanism, we have used a GPS device to determine the position of the real camera in the scene and an inertial sensor/electronic compass mechanism to determine its orientation. These measurements however are prone to several imperfections.

The XSens MotionTracker (68) provided an excellent device to measure the camera's orientation with respect to the magnetic earth field. Although we retrieve the device's orientation with a very high precision ($< 0.1^\circ$ declination, alignment casing and internal sensors) we still need to take into account additional imperfections.

The virtual terrain is oriented according to the geographical (and not magnetic) north. For this reason, we need to calculate its declination from the magnetic north which can be done using the International Geomagnetic Reference Field Model (IGRF), version 10 (84). For this, we have calculated that at the time of writing this document the declination at Trento amounts to $1^\circ 41'$ E changing by $0^\circ 6'$ E/year by using an online tool provided by National Geophysical Data Center (85) which is based on the IGRF. This value is set as a correction parameter on the tracker itself and it requires only to be adjusted every one or two years. This information is used for calibrating all measurements stemming from the motion-tracking device. This imperfection contributes only marginally to the cumulated deviation. The major influence factor stems from the fixture of the motion-

tracking device on the camera. This has to be done in a very precise manner to prevent an extensive calibration on-site. It is obvious however that we are required to provide a rotational adjustment mechanism inside the AR application.

The determination of the operator's location on-site is determined using a GPS device, queried by the embedded GPS service. The positional accuracy is highly dependent on the used service. The public GPS signal provides only a precision in the range of 10m-20m. However, the accuracy can be enhanced to 10cm using differential GPS. For both cases, we have given the user the opportunity to fine-tune his/her position on the real site.

Figure 56 shows a typical scene with the virtual terrain overlaid in wireframe for demonstration purposes.

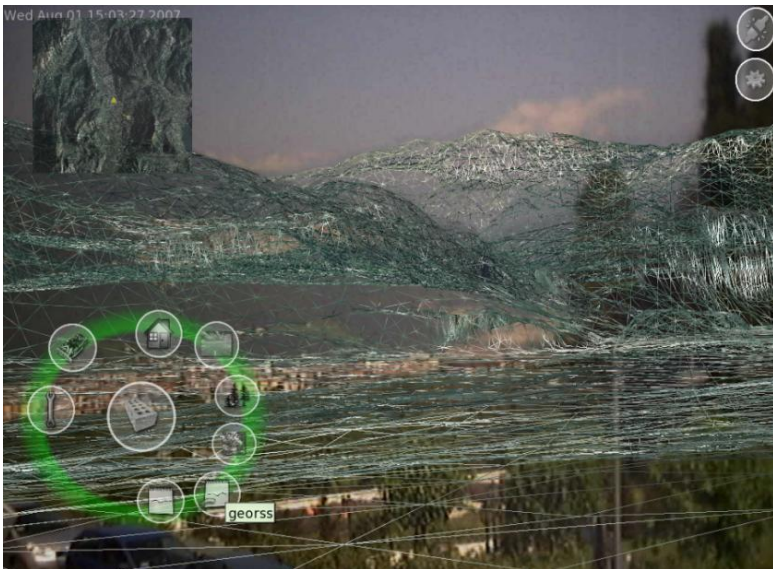


Figure 56. Captured video stream and overlaid heightfield

6.4. Camera calibration

The virtual camera calibration process has to include the perspective projection matrix (camera dependent) and the model transformation matrix (location dependent). We have directly used the GPS location and orientation from the tracking device as model matrix. The process of calibration itself has to be as efficient as possible to shorten the setup time of the system on-site. For this reason, we have developed a control element analogously to the ring menu. It allows for offsetting the virtual camera's translation and rotation to the real video camera as shown in Figure 57. We then add this correction matrix, which is determined by the user in case the real and virtual world do not match.



Figure 57. Camera calibration tool

The perspective adjustment is given by the perspective camera parameters of the real video camera, which are directly used as perspective parameters of the virtual camera. They can be conveniently determined by using the tools provided by ARToolkit(67) by manual line-fitting(86). The obtained configuration file is then used as a startup parameter during application initialization.

6. Augmented Reality for large environments

Figure 58 demonstrates a user working with the system. The TabletPC is connected to the motion tracker and the video camera. The hardware setup is illustrated in HW configuration(s, B.3).

It has to be highlighted that this scenario is collaborative through a wireless network connection. A supervisor in the office followed the actions of the user and shared the same view point on his local entirely virtual reality client.



Figure 58. A user working with the AR setup.

6.5. Hybrid approach (supervised AR)

Outdoor GIS is usually performed on an area extending for several square kilometers. A severe restriction of the user would be to limit his position to his GPS-determined location, since there might be areas which are occluded by objects or terrain topology and that would not result accessible without moving physically close to the location of interest. Moreover it is often required that information is seen in a larger context independently from the actual position of the user in the real world. For this reason, we allow the user to temporarily leave the Augmented Reality mode. He/she is then enabled to use specific terrain

navigation metaphors such as flying on the terrain. Both VR and AR mode allow interacting with the scene equally. He/she might then decide if it is necessary to analyze the area of interest in AR by relocating to this location. Additionally, the user is supported by a world-in-miniature which shows his position at all times on the real terrain and his viewing direction.

The need for a precise heightfield is most important when placing virtual content like trees, forests, roads and buildings on the virtual terrain and in fact, into the real captured scene. It should be highlighted that the video of the captured terrain scene is in 2D. The depth information however can be easily derived from the virtual heightfield (height, geo-coordinates, distances).

A direct benefit of this technique is to achieve a free user point of view while still maintaining a correct placement on the virtual and real terrain. Furthermore, it is not required having always several tracked terrain features in sight. The user is made independent of occluding terrain features and terrain topology.

6.6. Superimposition of terrain cultures

Augmented GIS is a currently emerging field in the field of environmental planning and territorial management. Within this scope, the embedding of environmental features such as forests, trees and roads in the scene is crucial when undertaking environmental surveillance. Moreover, augmented reality techniques help to place objects of interest in relation to their real environment and thus offer a much higher grade of detail than traditional virtual environments with mainly abstracted and idealized entities.

A major benefit of our approach is that it allows viewing geo-referenced objects in a context in dimensions of several kilometers. The mapping of the geo-database features to their 3D visualization is currently fixed within functionality handlers by

identifying the WFS feature types via a nomenclature. We refrained at this stage to allow a selection of available feature sets via a traditional GUI dialog selection. Instead, we have embedded nodes into the interaction graph to provide a consistent interface and to demonstrate the extensibility of our graph-based approach.

For instance, the paths *query-map-forest* or *query-map-road* are made accessible via the chosen modalities. Advantage is taken of the fact that gestural input can be used to specify boundaries for a WFS query analogously to taking visual bookmarks in section 4.6. We have stored terrain features in a spatial and relational database (PostGIS). The access to the geo-database is made through a WFS to allow a potential combination with Web Map Services (WMS) and Web Coverage Services (WCS). Further, we are able to query available data sets via a GetCapabilities request.

Table 2. Mapping of features to 3D representations

Feature type	GML geometry type	Visualization
Street, dirt road, railroad	Line, LineSet	Textured polygons draped on terrain
forest	Polygon, MultiPolygon	Distribution of bill-boarded trees within polygonal bounds
tree	Point	3D Tree

The feature types are retrieved from the WFS in GML format. However, the GML schema allows only for the specification of geometries either as points, lines or polygons. Thus we interpret these geometries as geo-referenced markers and create a 3D representation according to their feature type. This has been exemplary done for several geometries and domains as illustrated in Table 2.

It should be noted that the retrieved GML includes attributes as well. However we focused on the more substantial integration part. This is worth mentioning here because we could potentially visualize or even modify attached data that would then be stored in the database using a transactional WFS-T.



Figure 59. Superimposition of WFS features

Figure 59 depicts a view on Monte Bondone, Trento with superimposed forestry data and the village of Sardegna. Clearly there is no occlusion culling present in the scene (Figure 59, right

side). However, this can be also seen as an advantage since the operator sees the occluded terrain profile and the respective features within the scope of the terrain surveillance and thus is not hindered severely.

6.7. Automotive review under real conditions

Today's automotive virtual design reviews are widely undertaken in a virtual reality setting separated from real world conditions. For this reason, it is necessary to build a physical prototype in order to examine models at an early stage of the styling phase. We have used our approach to place a car model to view it under real lighting conditions and to put in relation to physically existing objects. Figure 60 shows a car model on Via Belenzani, Trento during a daylight situation.



Figure 60. Car in a real environment

6.8. Conclusions

In this chapter, we have presented a novel approach to an outdoor geo-referenced large area surveillance, where it is possible to locate virtual 3D objects and terrain cultures on the captured 2D video stream. This is done by aligning a virtual geo-referenced terrain with the real scene. The location of the user is determined by GPS whereas the rotational motion of the video capturing device is tracked using a motion tracker attached to a video camera. Further, we have shown the extensibility of the multimodal interaction graph towards an outdoor augmented reality scenario.

7. Assessment and validation

Specifically when dealing with user interaction, it is of utmost importance to a project to perform a validation of the achieved results. The development of our application was at all times marked by collaboration with the industrial user who provided us with inspiring and constructive feedback. We have undertaken two extensive test sessions during which advantages and shortcomings of our approaches became evident. The first user test at an early stage of the project realized a collaborative user interface, which was tightly wired to the application. Only the touchscreen interaction modality was allowed. Further there were no means available for any customization with respect to used devices and their scenario dependent use within the application. Following the first test and early feedback, we have striven for a high configurability in all areas of the application. As follows in the next sections, we have received a very positive feedback during the final user test that verifies our developed approach in an industrial environment.

7.1. Testers

The test session was run throughout a working day at ELASIS, Italy with users being asked to assess the system in groups of two during their working activities. None of the users had been previously informed of the event in order to avoid cross influencing.

7.2. Process

At the beginning of the test, users were invited to enter the VR lab and were given a short 2 minutes introduction to the system

and to the hardware configuration. During the following 5 minutes staff from Graphitech introduced the collaborative session and the concept of “master” application. Staff from Graphitech explained the concept of the interaction graph, being used by the system, and how it is possible via a simple configuration to change the entire interaction architecture following the motto “Configure once, interact in any way”.

For this it was shown to each group consisting of two users how to customize the interaction metaphor by creating the most appropriate combination of gestures/spoken commands/action. Specifically it was shown to each user how to use the interaction graph viewer and how to change the interaction dialogue. Specifically users were briefly taught that:

- Nodes of the graph define the actions.
- Connections define commands.
- Edges are used to define the interaction mechanism.
- Edges have properties.
- Actions/handlers are identified by path/order of user interactions.
- User interactions can be fully rearranged and customized according to the application context (architectural, automotive), and user-tailored.
- Separation between application and interaction definition.

An example of change in the configuration was carried on together with the user group. In particular, it was shown to users that it is possible to have seamless integrations of modalities by using nodes as definitions of actions/domains where edge attributes specify how to access them and gestures and speech are defined as attributes with a list of allowed items (how to advance

Users were also shown the final example of a configuration file (A.2).



Figure 62. Staff from Graphitech showing functionalities

The application was then started and users were also taught on how to use gestures and voice to drive the application. The staff was taught on how the system observes the drawn geometries on the overlay and how it checks if there are any current actions allowed according to the interaction graph. The helper dialogues for both speech and gestures were shown. Most importantly users were taught on how to activate the main ring menu and the navigation menu. An example of changing settings via voice was shown. Finally the *SuperHand* interaction was discussed, in particular how it is possible to navigate and how to change between navigation modes.

After such a short introduction to the systems (5 mins approx. – see Figure 62) users were asked to perform the tasks discussed in the following section. Users were invited to use the “think aloud” approach constantly commenting their feelings and impression during the system’s use (see Figure 63).



Figure 63. Users commenting on features and functionalities during the test session

The script which was handed out to the users can be found in appendix A.1.

Users were free to use the application and to swap hardware configuration between client no. 1 and client no. 2.

After the test session users a debriefing took place. Users were invited to comment on the system and on its features and usability. Eventually they were asked to fill in two different questionnaires (section 7.4.1 and 7.4.2).

7.3. Data analysis

Data was collected through the use of two questionnaires (see relevant section). Data was then collected within an excel spreadsheet and further processed. For each questionnaire the rating provided by the users was used to calculate mean values and standard deviations. Mean values of the ranking was also calculated. For the ISO questionnaire, which allowed users to

provide a “no opinion” comment, additionally we have calculated the distribution (in %) of agreement, disagreement, neutral opinion and “no opinion” for each sentence assessed. All this information was used to create graphs that could allow visual analysis of the results. Comparison with the corresponding values in the previous test session has provided direct criteria to assess the success of the new/improved functionalities. Recording of the users’ comment has also allowed the assessment of the development’s results.

7.4. Evaluation methods

7.4.1 First questionnaire

The goal of the first test, as in the previous test session, was to assess the functionalities and performances of the interfaces through a number of heuristics. The first questionnaire contained a set of 47 questions organized in 5 macro groups:

Ergonomics factors, assessing the comfort and ergonomics of the system and, more generally, of the set-up, e.g. exploring whether the use of the system was physically tiring or whether the user had experienced any discomfort (aching limb, headache, etc.).

Hardware and setup, assessing the hardware adopted. This included verifying if the user found it comfortable to use the tablet and the Powerwall at the same time and if it was difficult to adjust the eyesight back and forth from the screen to the Powerwall.

Scenario and test, assessing the validity of the scenario adopted in terms of similitude with the real-life scenario. This included assessing if the scenario of the test was representative of a normal operational task.

Human factors, assessing the HCI aspect of the system, including intuitiveness, responsiveness, sense of collaboration etc.

Users were asked to rate from 1 (unsatisfied) to 5 (satisfied) with a Severity ranking (low, med, high).

Comments were allowed at the end of the questionnaire in order to collect general remarks. A debrief took place after the first test session and users were asked to comment on the system and point out issues. Users were then asked to add these recommendations to the end of the document.

7.4.2 Second questionnaire

Similarly to the previous test session an ISOMETRIC questionnaire was prepared according to International Standard ISO 9241. Users were asked to fill in the ISOMETRIC questionnaire after completing the first (heuristics) questionnaire. According to ISO guidelines, and similarly to the previous test, the assessment has been done based on the following categories:

1. Suitability for the task.
2. Self descriptiveness.
3. Controllability.
4. Conformity with user expectations. (*omitted for brevity*)
5. Error tolerance.
6. Suitability for individualisation.
7. Suitability for learning. (*omitted for brevity*)

As in the previous test each of these categories included a number of different sentences which could receive a ranking from 1 to 5 (from strong disagreement to strong agreement) with a further “no opinion” option. The user was also asked to record the importance

provided by a set of loudspeakers. One laptop was connected to the touchscreen and to the motion tracker. The second laptop was connected to the Wacom tablet. Both laptops have integrated microphone and loudspeakers. The architecture of the hardware configuration used for the final user test can be found in appendix B.1.



Figure 65. The hardware configuration used for the test. The VR room

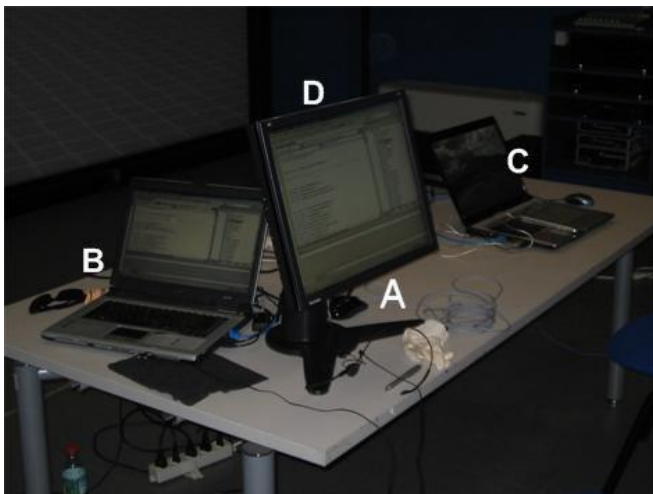


Figure 66. (A) The XSens Motiontracker, (B) and (C) the two laptops, (D) the touchscreen



Figure 67. The XSens MotionTracker used for navigation

7.5.1 Users

The test involved 9 users from ELASIS, 7 men 2 women, all from Italy, aged on average 25-34, with different profiles and with an average good experience with CAD/CAS (3.3 in a 1-5 range). Previous experience with VR system was also assessed; this showed that the majority of users had very good experience with VR while 30% of users had no experience at all. Four users had used IMPROVE during the first test session. Their feedback has been essential to assess the evolution of the system as it was perceived by the final users.

7.6. Questionnaire analysis

7.6.1 Initial questionnaire analysis

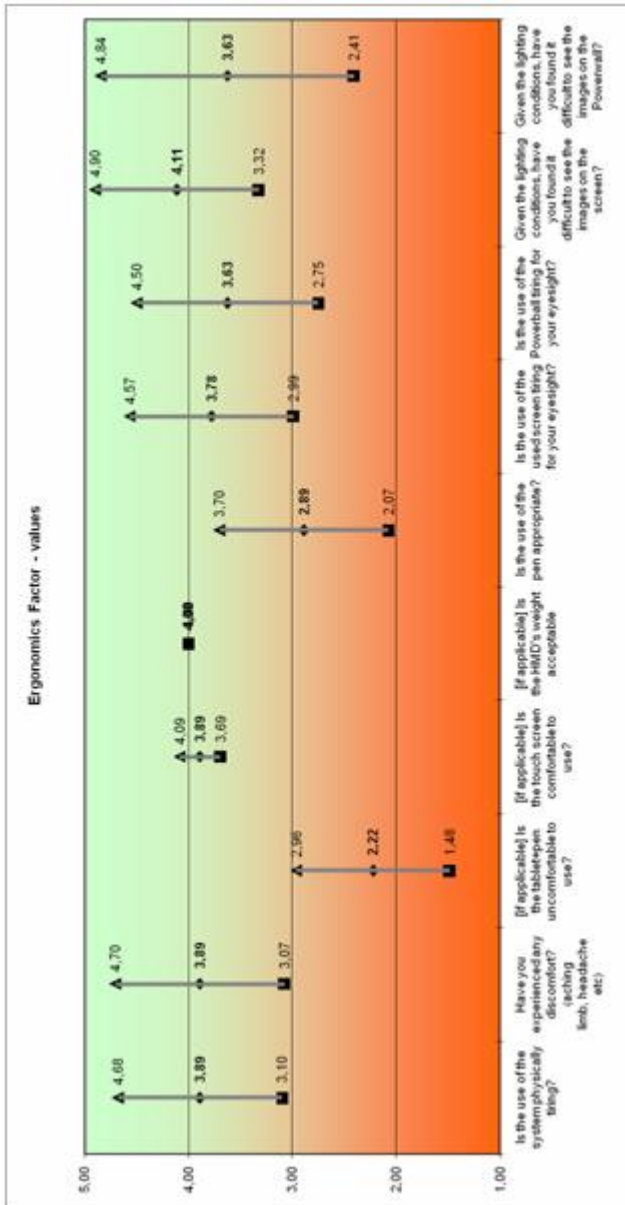
The overall assessment of the system is positive, with better ranking as reported in the following pages. As for the first test the analysis has been carried out by considering, for each heuristic class, two diagrams.

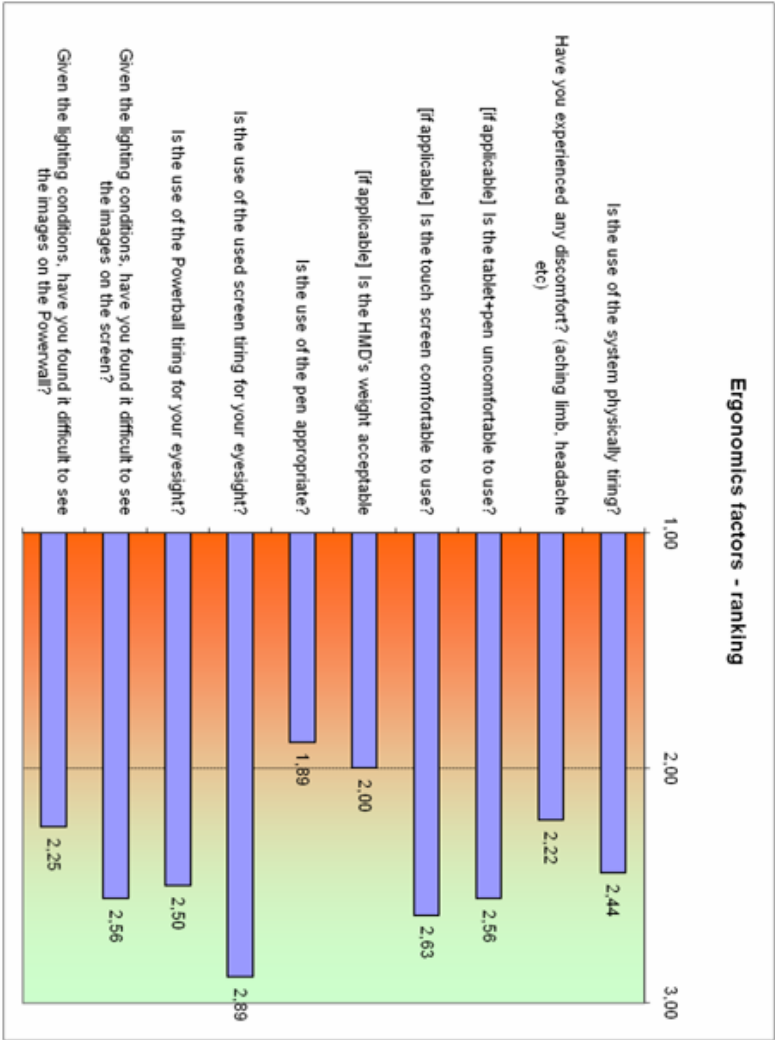
- 1) The first diagram collects the mean value (marked by a diamond) of the score recorded for each sentence. The higher the score the higher the level of satisfaction. The mean value is shown within a range defined by the average of the absolute deviations of data points from their mean. The closer interval indicates a general agreement on the mean value while wider interval shows a more diverse distribution of the feedback.
- 2) The second diagram shows the mean value of the ranking user has assigned to each item. This is defined in a scale from 1= low importance to 3= high importance. A higher value indicates high relevance of the issue and vice versa. This diagram should be considered an “amplifying” factor of the values shown in the first diagram.

7.6.2 Ergonomics factors

The general level of satisfaction is fairly high, as shown by the average values. Specifically users have clearly indicated that the system is not physically tiring with limited discomfort being caused by its use (high satisfaction). The exception to this trend is the use of the standard tablet that, if compared with the use of the touch screen, shows a significant level of dissatisfaction (2,22). This was also recorded during debriefing when users clearly indicated that use of the touch screen is much more intuitive than the traditional tablet. The use of the pen is considered fairly appropriate (2,89) although not important with a ranking of 1,89). In fact, as recorded during the test, most users preferred, during the test, the use of their fingers to interact with the touch screen. This according to their comment was much more direct. Other fields show that the use of that screens where appropriate, little fatigue in was caused to the sight by prolonged use of the system. This last issue was considered very important with a severity ranking of 2,89. Interestingly, compared with the previous test using the TabletPC, the use of the touch screen has proved more pleasant, specifically in terms of brightness as demonstrated by the value 4,11 if compared with 3,3 previously scored by the TabletPC. Quite surprisingly this time the use of the power wall conditions have generated a lower level of content (3,63) than before (4,2) although the conditions were identical.

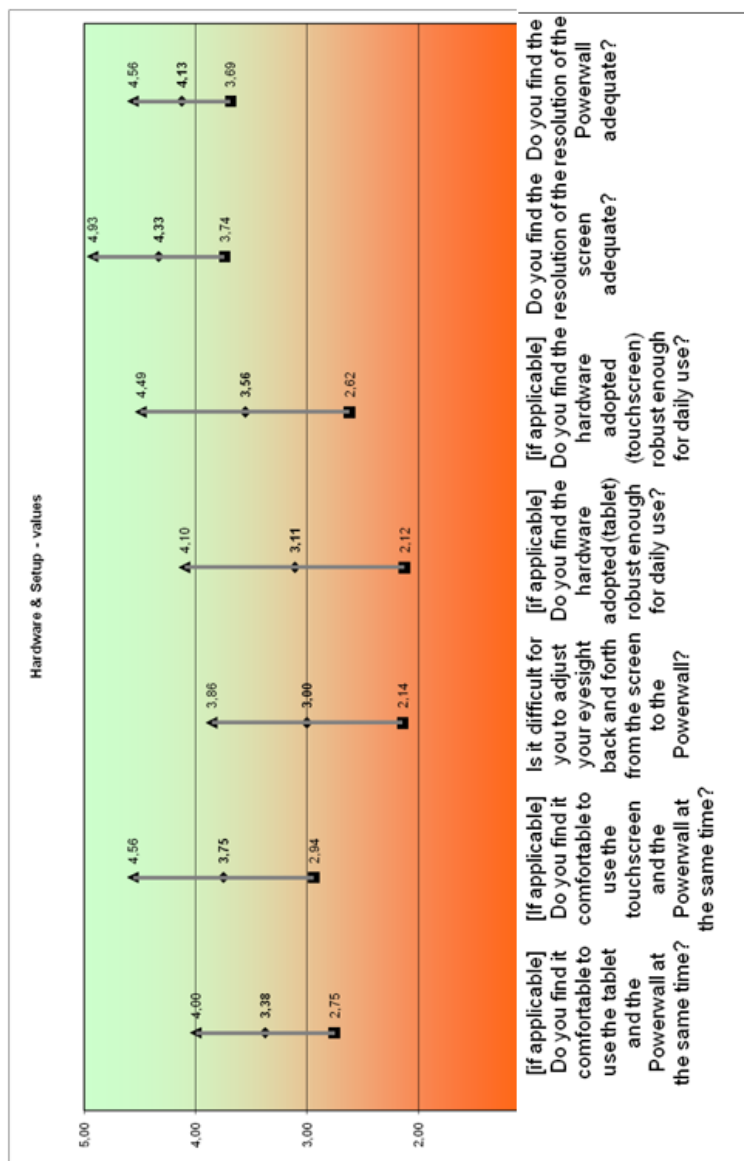
7. Assessment and validation

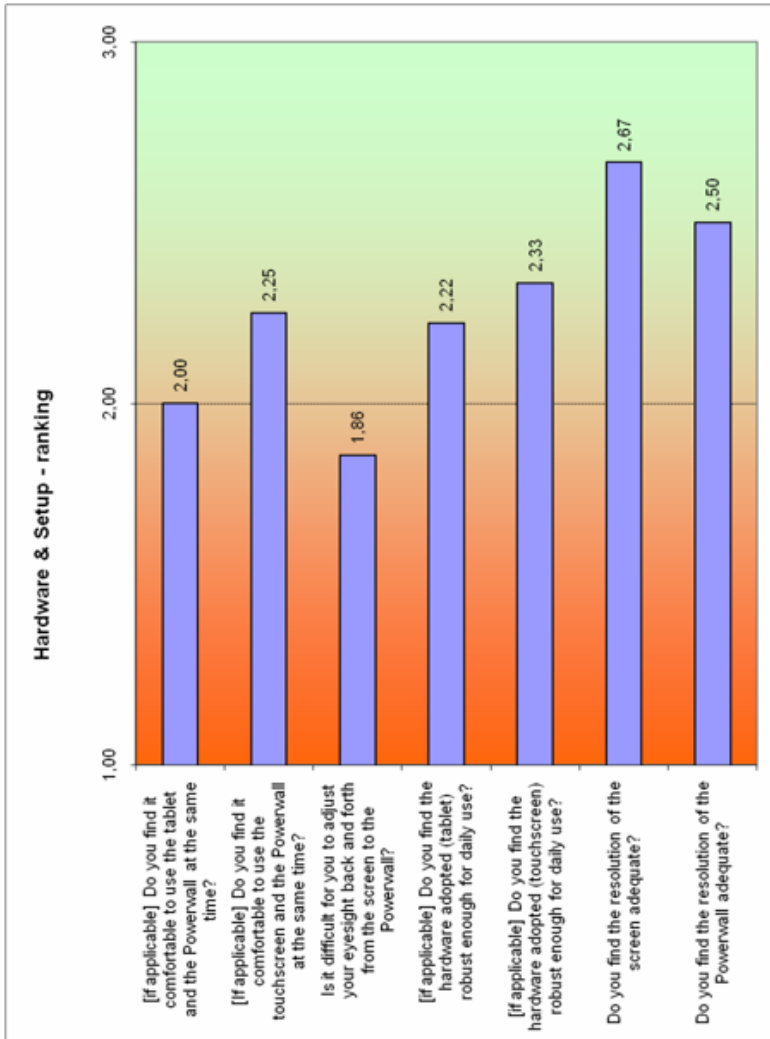




7.6.3 Hardware and setup

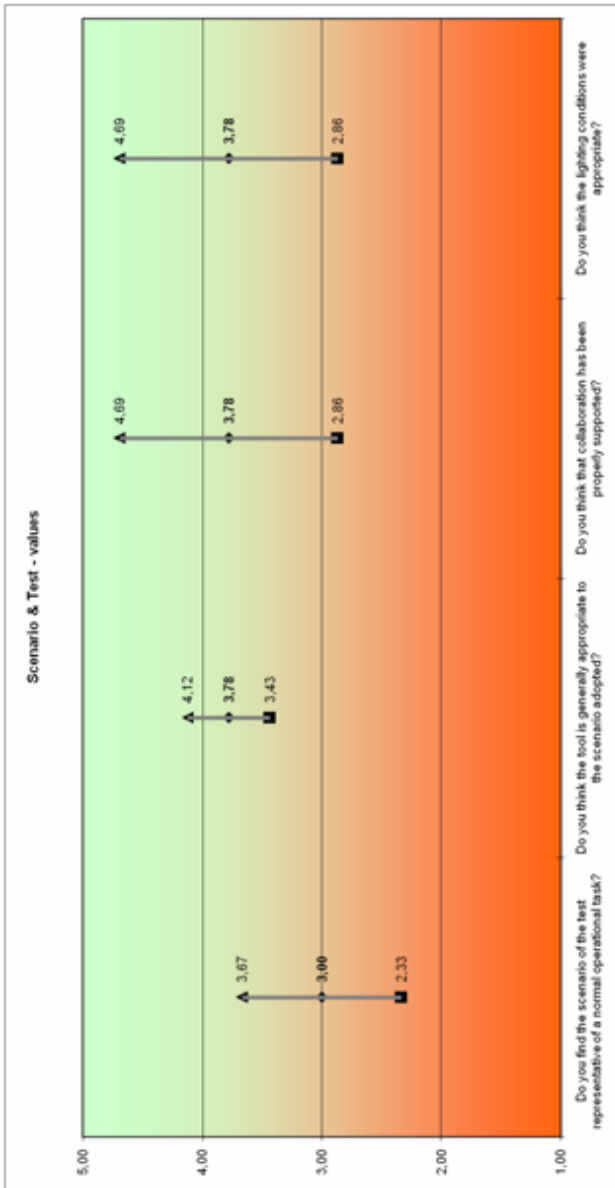
The combined use of tablet + Powerwall was considered appropriate although, as recoded by the assessment of the ergonomics, the touchscreen + Powerwall configuration was preferred (3,75 vs. 3,38). Adjusting the eyesight from the screen back and forth to the Powerwall was not considered acceptable (3,0) although in terms of importance this was given very little importance with a severity ranking of 1,44. Positively (3,11) the hardware configuration was considered robust enough (the touchscreen even more positive 3,56) scoring a higher value than the previous use of the touch screen (2,89). Also the resolution of the screen was considered very positively (4,33) with a very high ranking of 2,67 which is the highest in this group of heuristics. Finally the resolution of the Powerwall was considered positively (4,13) although, surprisingly, scoring a lower value than in the previous test (4,4).

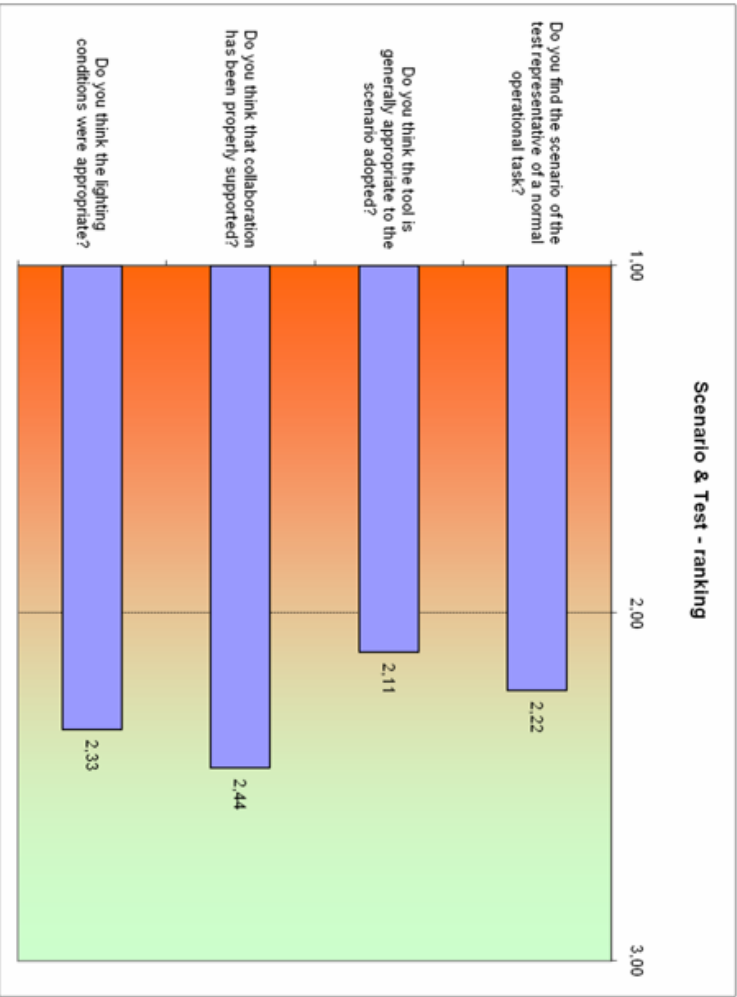




7.6.4 Scenario and test

The scenario was considered representative of standard operational task with a very high score (3,78) in terms of appropriateness of the tool, of sense of collaboration and lighting conditions. If compared with the results from the previous test the sense of satisfaction has generally increased with higher mean values being reported. The last indicator, referring to lighting conditions, has not shown any substantial difference with the previous test (3,78 vs. 3,70) reflecting the fact that the lighting conditions of the two test were identical.





7.6.5 Human factors

This is one of the most important sections of the test. At a glance, if compared with the previous test, the graph clearly indicates an improvement in the level of satisfaction, with average value higher (often substantially) than those recorded during the first test session.

The use of the pen has been considered very intuitive (4,0). An extremely positive result has been recorded in terms of satisfaction in the responsiveness of the system which has jumped from a very poor value of 2,2 of the first test to a very positive 3,89 (severity ranking 2,67). This is the result of the re-engineering of the interface which has taken place at the software level. Specifically the process that manages the creation of annotation has been re-designed after the result of the first test since it made the system very slow and un-precise to use when drawing notes. The result is extremely positive as visible from the clear improvement in the relevant heuristic.

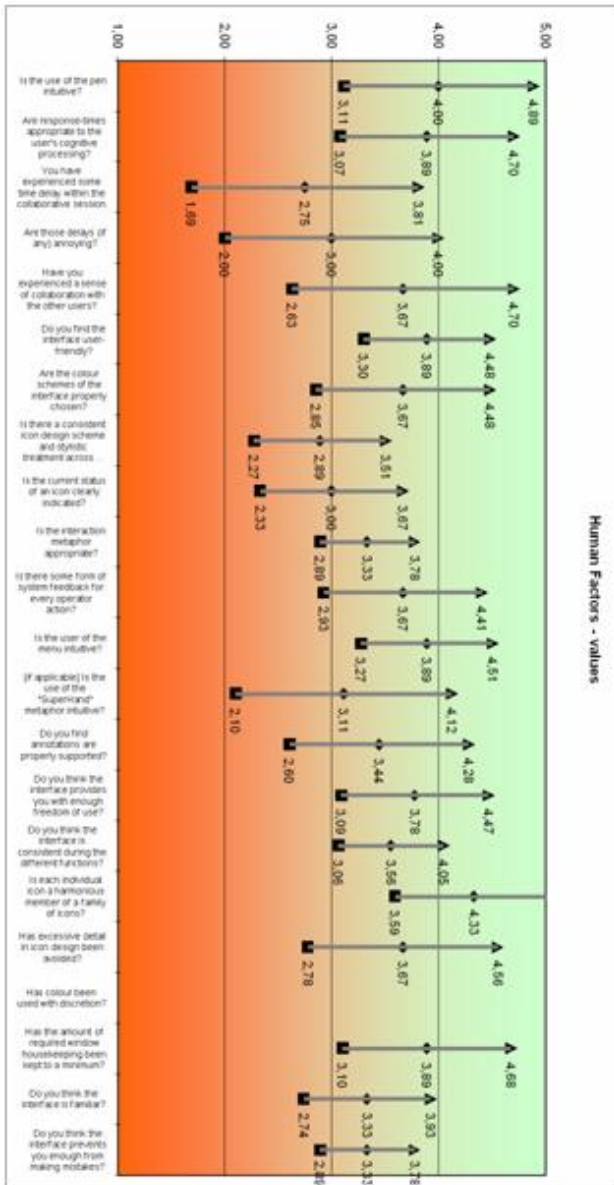
Positively the sense of collaboration has also improved from 2,7 of the previous test to 3,67 as result of the general improvement of the interface, in terms of responsiveness and support for collaboration. On average the system was considered very user friendly (3,67) with a consistent color scheme. The consistency of the icon design was rated less positively than in the previous test with a lowering of the appreciation in the indication of the current status.

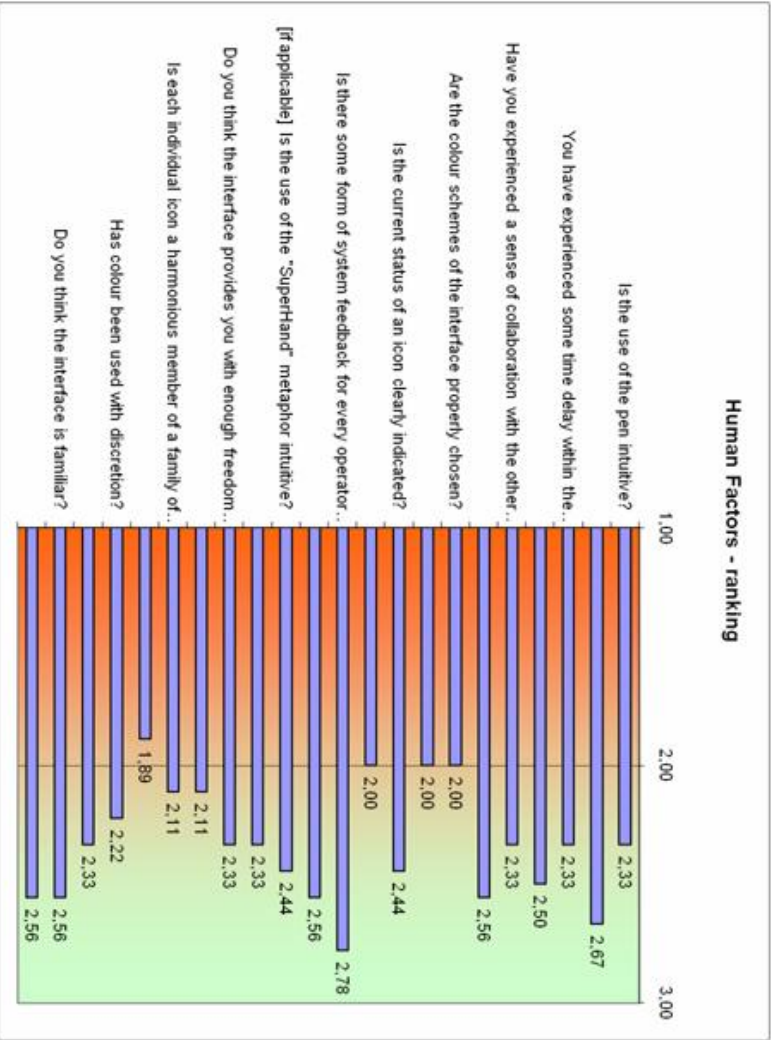
The system feedback has significantly improved (3,67 vs. 2,56) compared to the previous test. This is a fundamental improvement since this represents one of the most important element of the human-computer interaction dialogue, as highlighted by a severity ranking of 2,78, the highest in this group of heuristics. The use of ring menus was considered very intuitive (3,89) as well as the acceptance of the “SuperHand” approach which has scored a noteworthy 3,11.

Use of annotation was much better rated (3,44) than in the previous test when it scored only 2,3. This has been caused by the re-design, as mentioned, of the process managing the drawing of the notes that now results much more fluent and responsive. Further functionalities provided in this latest version (e.g. definition of type of notes etc) have also contributed to the improvement in the ranking.

The sense of freedom provided by the interface is quite good (3,78). The design of icon has been considered very harmonious (4,33) if compared with a much poorer 2,7 a clear result of the GUI graphical re-design.

7. Assessment and validation





7.6.6 Final questionnaire analysis

As for the previous test, the second questionnaire has been compiled in accordance to the ISOMETRIC test, based on the ISO standards. Two diagrams for each category show the results emerging from the questionnaires:

1. The first diagram shows level of user's agreement, disagreement or neutral behavior. Additionally, if compared with the heuristic test, the user can also express no opinion.
2. The second, more complex, diagram shows:
 - a. (diamond) The mean value of the score for each sentence, through a score from 1 = I disagree to 5 = I agree.
 - b. (triangle and square) The range defining the average of the absolute deviations of data points from their mean.
 - c. (grey bar) The mean value of the ranking in a scale from 1 = unimportant to 5 = important

Suitability for the task

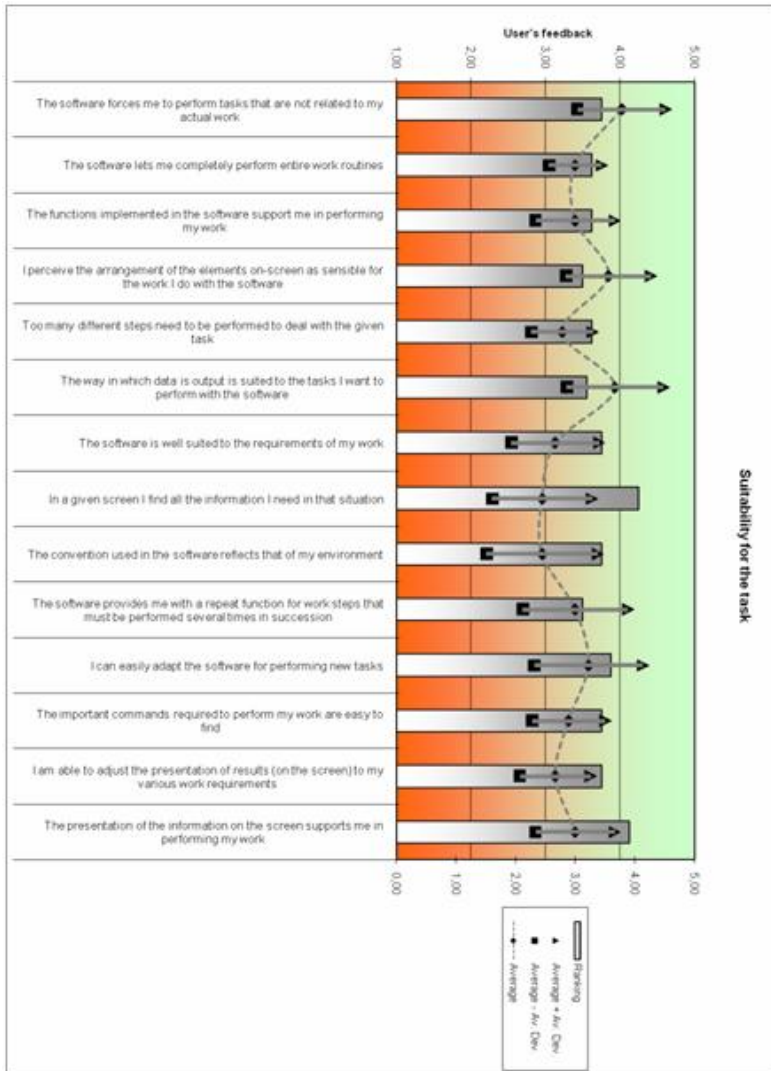
The assessment shows, if compared with the result from the previous test session, users have a more positive result and, interestingly, a much clearer view on the "suitability for the task" group as no user expressed a "no opinion". If compared with the previous test the amount of neutral judgment has grown.

At a glance, looking at the second diagram, the result never diverges too much from the average (3) value. The largest

difference is to be found in the fact that now users feel that the software forces to perform tasks that are not related to the actual work. Positively the users perceive that the software let them completely perform their tasks, that the arrangement of the elements is sensible for the work to be carried out and that the way data is output reflects the task the users want to perform with the software.

Positively a higher share of users considers that it is easy to adapt the software to perform new tasks and that the majority of the commands required to perform the work are easy to find.

7. Assessment and validation

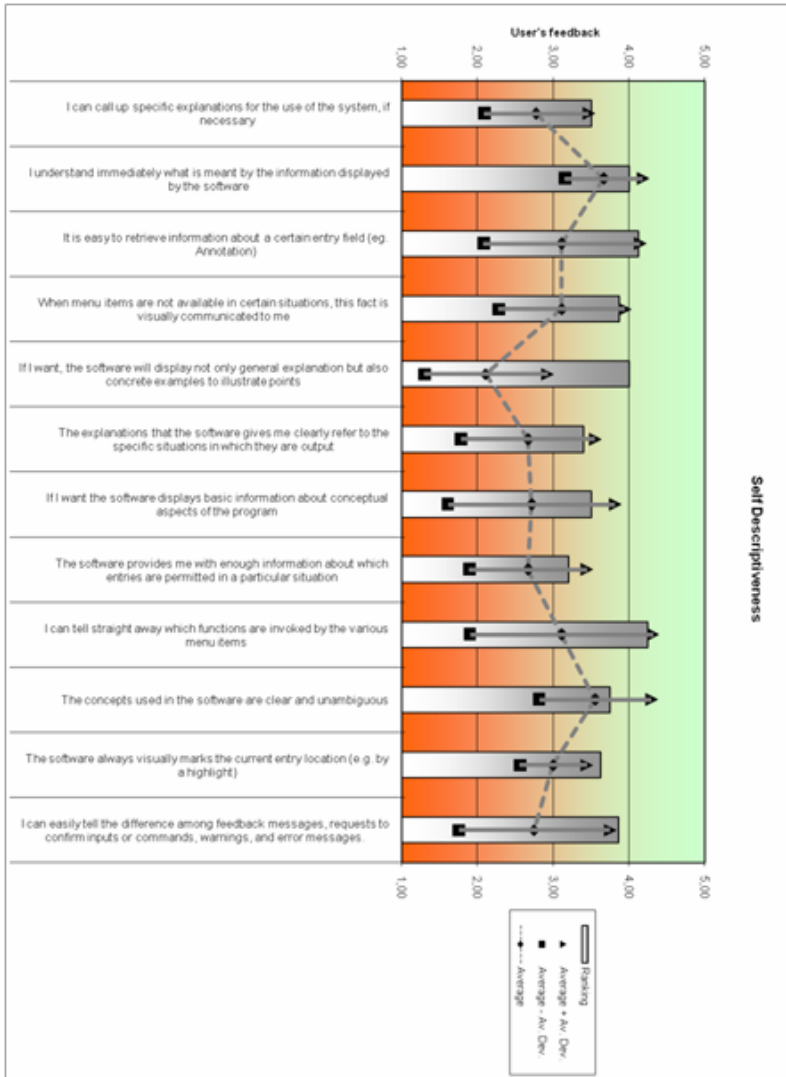


Self descriptiveness

The results in this group reflect show a better tendency if compared with results emerged in the previous test. Results, albeit always close to average (3) and much better than in previous test, never get very high ranking. According to the users' feedback during the de-briefing, this was due to the fact that, all communication between the interface and the users, entirely lies on the graphical language with virtually no text. Although this makes the entire interface much more compact, and once accustomed, much faster to use, at the beginning this requires interpretation by the user.

A positive point is that the user feels that it is possible to understand immediately what is meant by the information display by the software however, poor results are scored in terms of support for concrete examples, rather than using general explanations. This is not surprising as there is currently no way to invoke example in the use of the interface.

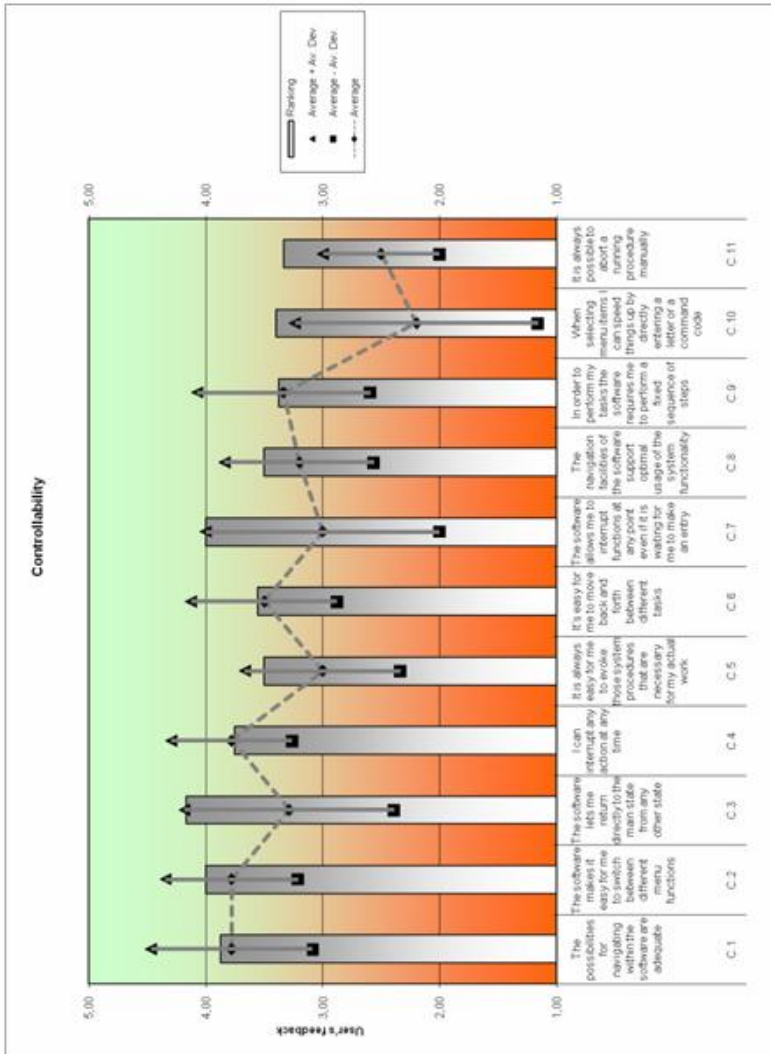
7. Assessment and validation



Controllability

The results in this group reflect show a better tendency if compared with results emerged in the previous test. Results, albeit always close to average (3) and much better than in previous test, never get very high ranking. According to the users' feedback during the de-briefing, this was due to the fact that, all communication between the interface and the users, entirely lies on the graphical language with virtually no text. Although this makes the entire interface much more compact, and once accustomed, much faster to use, at the beginning this requires interpretation by the user.

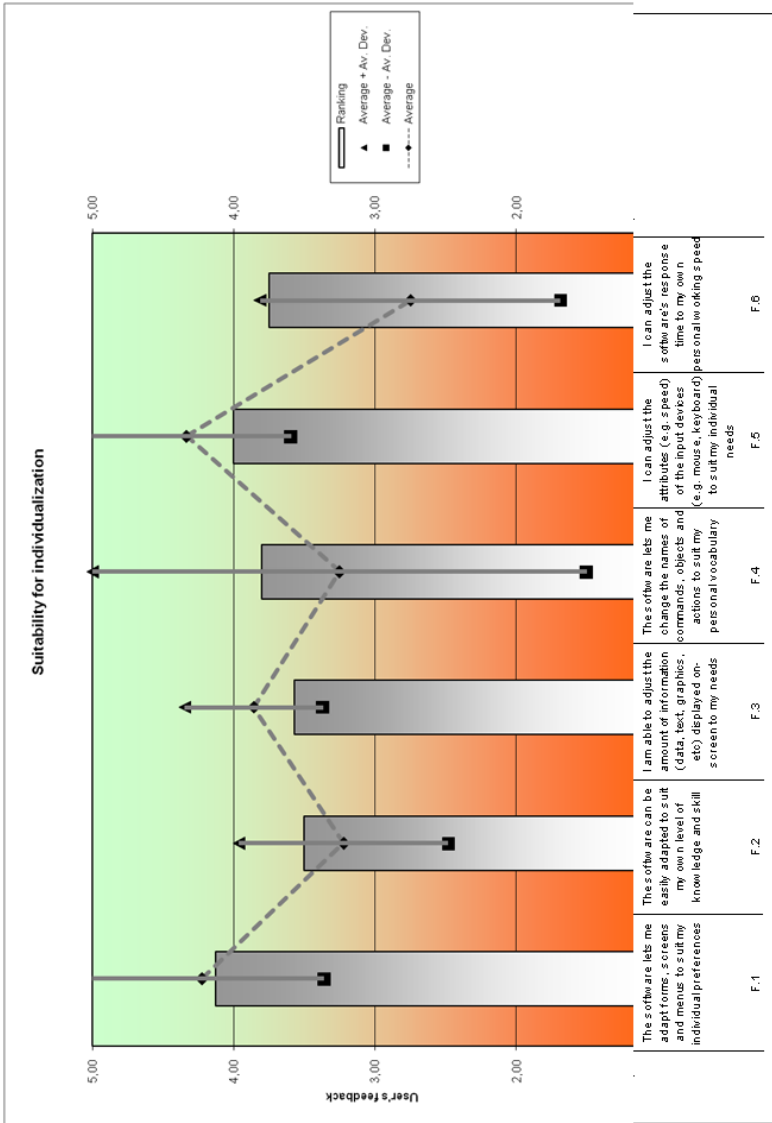
A positive point is that the user feels that it is possible to understand immediately what is meant by the information display by the software however, poor results are scored in terms of support for concrete examples, rather than using general explanations. This is not surprising as there is currently no way to invoke example in the use of the interface. A similarly low value is scored (2,5) by the possibility to abort a running procedure manually although the importance of this has decreased, if compared to the previous test, from 4 to 3,4.



Suitability for individualization

Extremely good results have been collected in terms of suitability for individualization. The user perceive that the software lets them adapt forms, screens and menus to suit their individual preferences (4,22), that the software can be easily adapted to suit their own level of knowledge and skill (3,22), that they are able to adjust the amount of information (data, text, graphics, etc) displayed on-screen to their needs (3,86), the software lets them change the names of commands, objects and actions to suit my personal vocabulary (3,25), they can adjust the attributes (e.g. speed) of the input devices (e.g. mouse, keyboard) to suit their individual needs (4,33).

These very high values are the result for the high degree of customization provided by the authoring of the interface, which allows every user, if required, to define a customized interaction multimodal dialogue.



Video Analysis

During the analysis of the video a few issue emerged. The most important benefit highlighted by the users was the responsiveness of the system. The previous version was very slow for instance when rendering notes. This caused a sense of unease since the entire interaction process became slower. The current version now allows a much more fluent interaction.

As far as the hardware is concerned the use of the tablet was not positively consider as this does not allow the having direct feedback and obliges the user to interact with a device (the tablet) while receiving visual feedback from another (the screen). The touchscreen instead was very well accepted. Most users had noticed how the resolution was adequate and furthermore it allowed the use of fingers, in place of the pen.

The support for two handed interaction was also considered a plus by some users. In fact some of the users (see following figures) used the motion tracker connected to the (non dominant) hand to navigate while interacting with the other (dominant) hand. This increased the response and interactivity of the system and it was very well considered. A user had reported that further control could be provided by supporting pedals to set the navigation speed additionally to the two hand interaction provided.



Users who had experience the previous version of the system had also underlined that the touchscreen was preferred to the TabletPC also for the heat the latter produced, which made it very unpleasant to use after a few minutes. A major improvement, according to users, would be the implementation of a proper “undo-redo” mechanism. This in fact is considered a limit and it generates a general sense of anxiety to the user who is aware that no potentially bad action can be recovered.

7.7. Conclusions

The result of the work has been implemented in a very complex prototype which allows concurrent users to interact with a collaborative virtual environment specifically thought for design review sessions. This has been done in the context of the EU project IMPROVE. The resulting prototype has been successfully validated within a real-life scenario. Specifically two extensive test sessions have been run at ELASIS (Fiat research center) in Naples to assess the usability and user-friendliness of the final interface. The effectiveness of the prototype was assessed through specific ISO-compliant questionnaires that all final users of the

system were asked to fill in. The results have been very positive and show how the approach pursued has lead to a sharp increase in usability. As demonstrated during the tests the application can be configured to a wide range of input and display devices using mapping artifacts which normalize and redirect the incoming data. Further, we have proven that our approach for two-handed interactions by using a lightweight tracking system can be integrated within the multimodal dialogue with the system. As stated by the testing users, this is essential in a daily workflow with the high priority of non-fatigue navigation through the scene and around the reviewed car model.

8. Conclusion and perspective

8.1. Conclusions

Customizable multimodal interaction

This work has presented a novel approach to user-centered customizable multimodal interactions. The system integrates voice commands, gestures and traditional dialog elements which can be tailored to the users' interaction preferences and scenario requirements. This is achieved by binding modalities together via a bidirectional graph. The nodes and edges of the graph represent the dialogue of the user with the system. This way the users are enabled to specify precise modalities for interactions through navigation through the graphs. The dialogue of the user with the application and in fact the behavior and functionalities of the application are designable using a graphical authoring tool. For this reason we have developed a command interpreter which provides the end user with intuitive but powerful means to control the application. Our two-tier model decouples the application's functionalities from the actual user interactions and offers an exceptional degree of freedom for customization with respect to gestural and voice input. Moreover, the overlaid user interface is customizable through extended skinning with respect to language, appearance and offered elements.

Motion-tracked hand

We have demonstrated a novel navigation metaphor based on a motion tracked glove. These gestures are mapped to appropriate navigation schemes to allow for a natural, scenario specific

navigation. The tracked-hand modality (“Superhand”) can be optionally combined with the sequential multimodal interaction metaphor. This extends our work to using two modalities in parallel: one modality exclusively controls the navigation and the choice of navigation metaphor while the other provides the means for interacting with the scene using the aforementioned approach.

Distributed VR/AR application

The information exchange uses XML messages in a channel topic/subscription method to deliver collaborative navigation and scene modification. It has been used to integrate and to visualize time-varying information inside the VR/AR application. The distribution of the VR/AR application was achieved through establishing a communication to instances using a Message Passing Middleware (MPM). The information exchange uses XML messages in a channel topic/subscription method to deliver collaborative navigation and scene modification. It has been used to integrate and to visualize time-varying information inside the VR/AR application which is crucial when working with distributed data providers such as sensors where the frequency of the information update is high.

Collaboration

Using the distributed system architecture, we have shown how data and joint navigation can be effectively shared amongst the users. This is especially important when performing design reviews where a panel of reviewers jointly examines products. The collaboration consists of annotating model parts and modifying material properties and lighting conditions.

Terrain integration

In order to realize an outdoor scenario, we have extended a VR/AR framework with a terrain visualization module. This module was enhanced with video see-through capabilities to allow the alignment of virtual terrain with the real world. Further we have integrated location-based services such as a GPS module and an electronic compass/motion tracker to geo-reference the user and his/her surroundings. A retrieval of geospatial data using Web Features and Web Map services has been successfully integrated. This approach is specifically applicable when dealing with large distances where optical marker-based and marker-less tracking systems fail to achieve a correct camera mapping. A major benefit of our system is the geo-referencing of virtual objects and the topology of the virtual terrain with the real world. This way it becomes possible to virtually interact with the real scene by placing virtual 3D content such as trees and houses directly or to drape derived data like 3D maps onto the real terrain.

Hardware configurability

The application can be configured to a wide range of input and display devices using mapping artifacts that normalize and redirect the incoming data. Within this scope, several services to handle speech input and gesture recognition have been implemented to allow for the multimodal interaction schemes mentioned earlier.

Validation through industrial user tests

The very interactive nature of virtual reality applications has represented an ideal test bed to validate such an innovative approach. The result of the work has been implemented in a very complex prototype that allows concurrent users to interact with a

collaborative virtual environment specifically thought for design review sessions.

The resulting prototype has been successfully validated within a real-life scenario. Specifically two extensive test sessions have been run at ELASIS (Fiat research center) in Naples to assess the usability and user-friendliness of the final interface. The effectiveness of the prototype was assessed through specific ISO-compliant questionnaires that all final users of the system were asked to fill in. The results have been very positive and show how the approach pursued has lead to a sharp increase in terms of usability and effectiveness.

8.2. Perspective

Although the interaction graph provides an exceptional degree of freedom for the creation of customized interaction schemes, it is still limited in several ways. Until now there is no way to include dynamical component in the dialogue with the system which could for instance help to include a list of available WFS feature types as nodes. Further, nodes and the according application functions could be resolved using the semantic lexical database WordNet(87) to find synonyms of the embedded node names. This could offer a greater flexibility specifically when using speech input.

The general-purpose graphical editing tool is still limited in its capabilities. A dedicated tool for allowing only valid nodes and edges should be offered to the user, thus offering a graph validation. This scenario-aware editor could be based on open source authoring and validation tools (75)(76). This was not implemented because of time constraints, but it is highly recommendable and thus mentioned here.

Our camera tracking approach is specifically tailored to large area surveillance. GPS and Differential GPS are currently to imprecise

to locate the user in the field with an appropriate precision. The imprecision of GPS has made it necessary to embed a calibration of the virtual camera at each relocation of the system. In the advent of the European Galileo Positioning System (88), a much higher precision in the range of few centimeters through hyperbolic lateration will be achieved. Thus it will be possible to achieve a continuously mobile augmented reality application where the user is enabled to move in the field wearing a near-to-eye display (head mounted display). Alternatively an optical marker-based or marker-less tracking mechanism could be used for small distances where high precision is required. The combination with our approach could provide means to undertake effectively augmented reality terrain surveillance in near and far dimensions for viewing the area of interest in the large scope and in detail.

Further, the integration of data stemming from a Web Map Service and Web Feature Service was hardwired to the application and allowed only for one pre-set image to be overlaid onto the real terrain. An improvement would be to use multi-texturing or shader techniques to analyze 3D maps in relation to each other. The overlaid features are currently not rendered in a realistic quality yet this can be considered out of scope of this work since we have concentrated on the interaction part.

Bibliography

1. *Observing professionals taking notes on screen.* **Melenhorst, M.** IEEE Computer Society, 2005. Proc. of Int. Professional Communication. pp. 540 – 545.
2. *Annotating 3D electronic books.* **Hong, L., Chi, E. H. and Card, S. K.** ACM Press, 2005. Proc. of CHI - Conference on Human Factors in Computing Systems. pp. 1463 – 1466.
3. *Sketching annotations in a 3D web environment.* **Jung T., Gross, M. D. and Yi-Luen Do, E.** ACM Press, 2002. CHI '02 on Human factors in computing systems. pp. 618 – 619.
4. *An Annotation System for 3D Fluid Flow Visualisation.* **Loughlin, M. M. and Hughes, J. F.** IEEE, 1994. Proc. of the Conference on Visualisation '94. pp. 273-279.
5. *Constrained 3D Navigation with 2D Controllers.* **Hanson, A. J. and Wernert, E. A.** 1997. Proc. of the Conference on Visualisation '97. pp. 175-183.
6. *The Virtual Annotation System.* **Harmon, R., Patterson, W., Ribarsky, W. and Bolter, J.** IEEE, 1996. Proc. of IEEE Virtual Reality Annual International Symposium. pp. 239-245.
7. *Boom chameleon: simultaneous capture of 3D viewpoint, voice and gesture annotations on a spatially-aware display.* **Tsang, M., Fitzmaurice, G. W., Kurtenbach, G. and Khan, A.** ACM Press, 2002. Proc. of Symposium on User Interface Software and Technology. pp. 111 – 120.
8. *Drawing for Illustration and Annotation in 3D.* **Bourguignon, D., Cani, M-P. and Drettakis, G.** 2001. Proc. of EUROGRAPHICS'01. pp. 114-122.
9. *Intuitive Annotation of User-Viewed Objects for Wearable AR Systems.* **Tenmoku, R., Kanbara, M. and Yokoya, N.** ACM

Press, 2005. Proc. of Ninth IEEE International Symposium on Wearable Computers. pp. 183 – 184.

10. *Improvement of panorama-based annotation overlay using omnidirectional vision and inertial sensors.* **Kourogi, M., Kurata, T., Sakaue, K. and Muraoka, Y.** ACM Press, 2000. Proc. of the 4th Int. Symposium on Wearable Computers. pp. 183 – 184.

11. *Tracking Menus.* **Fitzmaurice, G., Khan, K., Pieké, R., Buxton, B. and Kurtenbach, G.** ACM Press, 2003. Proc. of the 16th annual ACM symposium on User interface software and technology. pp. 71 - 79.

12. *Fluid Interaction with High-resolution Wall-size Displays.* **Giumbretiere, F., Stone, M. and Winograd, T.** ACM Press, 2001. Proc. of the 14th ACM symposium on User interface software and technology. pp. 21-30.

13. *Control menus: execution and control in a single interactor.* **Pook, S., Lecolinet, E., Vaysseix, G. and Barillot, E.** ACM Press, 2000. CHI'00. pp. 263-264.

14. *Studierstube - an environment for collaboration in Augmented Reality.* **Szalavári, Z., et al.** 1998, Virtual Reality: Research, Development & Applications, 3, pp. 37-48.

15. *Using Transparent Props for Interacting with the Virtual Table.* **Schmalstieg, D., Encarnação, L. M. and Szalavári, Z.** ACM Press, 1999. Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics (I3DG'99).

16. *The Personal Interaction Panel - A Two Handed Interface for Augmented Reality.* **Szalavári, Z. and Gervautz, M.** 1997. Proceedings of EUROGRAPHICS'97. pp. 335-346.

17. *Spacedesign: conceptual styling and design review in augmented reality.* **Fiorentino, M. et al.** IEEE Computer Computer Society, 2002. ISMAR 2002 IEEE and ACM

Interaction Symposium on Mixed and Augmented Reality. pp. 86-94.

18. *A gesture processing framework for multimodal interaction in virtual reality.* **Latoschik, M. E.** ACM Press, 2001. Proc. of the 1st international conference on computer graphics, virtual reality and visualization.

19. *The Role of Natural Language in a Multimodal Interface.* **Cohen, P. R.** 1991. Proc. of UIST'92 conference. pp. 143-149.

20. *Put-that-there: voice and gesture at the graphic interface.* **Bolt, R. A.** 1980, ACM Computer Graphics, 14(3), pp. 262-270.

21. *Unification-based multimodal integration.* **Johnston, M. et al.** ACM Press, 1997. Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics ACL. pp. 281-288.

22. **Reeves, B. and Nass, C.** Perceptual user interfaces: perceptual bandwidth. *Communications of the ACM, Volume 43, Issue 3.* March 2000, pp. 65-70.

23. *Towards a Computational Model of Sketching.* **Forbus, K. D., Ferguson, R. W. and Usher, J. M.** 2001. Proc. of the 6th international conference on intelligent user interfaces.

24. **Oviatt, S. and Cohen, P. R.** Multimodal Interfaces that process what comes naturally. *Communications of the ACM, 43(3).* March 2000, pp. 45-54.

25. *The efficiency of multimodal interaction: a case study.* **Cohen, P. R. et al.** 1998. Proc. of the International Conference on Spoken Language.

26. *Augmented Reality for Construction Tasks: Doorlock Assembly.* **Reiners, D. et al.** 1998. Proceedings of the international workshop on Augmented Reality.

27. *An Augmented Reality System for Training and Assistance to Maintenance in the Industrial Context.* **Schwald, B. and de Laval, B.** 2003, Journal of WSCG, 11(3), pp. 425-432.
28. **Schmalstieg, D.** Studierstube project: Open Tracker. [Online] 2007. [Cited: October 23, 2007.] <http://studierstube.icg.tu-graz.ac.at/opentracker/>.
29. *Multimodal integration: a statistical view.* **Wu, L., Oviatt, S. and Cohen, P. R.** 4, 1999, IEEE Transactions on Multimedia, Vol. 1, pp. 334-342.
30. *Unification-based multimodal integration.* **Johnston, M. et al.** Madrid, Spain : Association for Computational Linguistics, 1997. Proc. of the eighth conference on European chapter of the Association for Computational Linguistics. pp. 281-288.
31. *Augmented Relity in the Palm of your Hand: A PDA-Based Framework Offering a Location-based, 3D and Speech-Driven User Interface.* **Goose, S. and Schneider, G.** 2003. Proc. of Workshop on "Wearable Computing".
32. **Opera Software ASA.** Multimodal Browser. [Online] 2007. [Cited: October 25, 2007.] <http://www.opera.com/products/devices/multimodal/>.
33. **W3C.** W3C Speech Interface Framework: XHTML+Voice Profile 1.2. [Online] 2007. [Cited: October 25, 2007.] <http://www.voicexml.org/specs/multimodal/x+v/12/>.
34. **W3C.** Introduction and Overview of W3C Speech Interface Framework. [Online] 2007. [Cited: October 25, 2007.] <http://www.w3.org/TR/voice-intro/>.
35. *Mutual disambiguation of 3D Multimodal Interaction in Augmented and Virtual Reality.* **Kaiser, E. et al.** Vancouver, Canada:, 2003. 5th Internation Conference on Multimodal Interfaces (ICMI 2003). pp. 12-19.

36. *An open agent architecture*. **Cohen, P. et al.** [ed.] M. Huhns and M. Singh. 1994. AAAI Spring Symposium. pp. 1-8.
37. *Multimodal Interaction for 2D and 3D Environments*. **Cohen, P. R. et al.** 1999, IEEE Computer Graphics and Applications, pp. 10-13.
38. *A Multimodal Interface Framework For Using Hand Gestures and Speech in Virtual Environment Applications*. **Laviola, J.** 1999. Lecture Notes in Artificial Intelligence #1739, Gesture-Based Communication in Human-Computer Interaction. pp. 303-314.
39. *TYCOON: Theoretical Framework and Software Tools for Multimodal Interfaces*. **Martin, J. C.** [ed.] J. Lee. AAAI Press, 1998, Intelligence and Multimodality in Multimedia Interfaces.
40. **Nash, J.** Wiring the Jet Set. *Wired*. October 1997.
41. **ARVIKA**. Augmented Reality for Development, Production and Servicing. [Online]
<http://www.arvika.de/www/e/home/home.htm>.
42. *A Wearable 3D Augmented Reality Workspace*. **Reitmayr, G. and Schmalstieg, D.** 2001. Proc. of ISCW 2001 conference.
43. **Reitmayr, G. and Schmalstieg, D.** Mobile Collaborative Augmented Reality Project. [Online] 2007. [Cited: October 25, 2007.] <http://www.ims.tuwien.ac.at/research/mobile/ocar/>
44. *Tinmith-Metro: New Outdoor Techniques for Creating City Models with an Augmented Reality Wearable Computer*. **Piekarski, W. and Thomas, B. H.** 2001. 5th International Symposium on Wearable Computers. pp. 31-38.
45. *Using ARToolKit for 3D Hand Position Tracking in Mobile Outdoor Environments*. **Piekarski, W. and Thomas, B. H.** 2002. 1st International Augmented Reality Toolkit Workshop.

46. *The Tinmith System - Demonstrating New Techniques for Mobile Augmented Reality Modelling*. **Piekarski, W. and Thomas, B. H.** 2002. 3rd Australasian User Interfaces Conference.
47. *Exploring MARS: Developing Indoor and Outdoor User Interfaces to a Mobile Augmented Reality System*. **Hoellerer, T. et al.** 1999, Computers and Graphics, 23(6), pp. 779-785.
48. **Computer Graphics and User Interfaces Lab, Columbia University.** MARS Project: Mobile Augmented Reality Systems. [Online]
<http://www1.cs.columbia.edu/graphics/projects/mars/mars.html>.
49. **Klimt.** The Open Source 3D Graphics Library for Mobile Devices. [Online] 2004. [Cited: October 25, 2007.]
<http://studierstube.icg.tu-graz.ac.at/klimt/>.
50. **OpenGL ES.** The Standard for Embedded Accelerated 3D Graphics. [Online] <http://www.khronos.org/opengles/>.
51. *The World as a User Interface: Augmented Reality for Ubiquitous Computing*. **Schmalstieg, D. and Reitmayr, G.** 2005. Proceedings of the Central European Multimedia and Virtual Reality Conference 2005 (CEMVR 2005).
52. *Virtual gis: A real-time 3d geographic information system*. **Koller, D. et al.** IEEE, 1995. VIS '95: Proc. of the 6th conf. on Visualization '95. p. 94.
53. *Visual terrain editor: an interactive editor for real terrains*. **Clark, R. W. and Maurer, R.** 2006, J. Comput. Small Coll. 22, 2, pp. 12-19.
54. *Integration of a 2D Legacy GIS, Legacy Simulations, and Legacy Databases into a 3D Geographic Simulation*. **Ohigashi, M., Guo, Z. S. and Tanaka, Y.** ACM Press, 2006. SIGDOC '06: Proc. of the 24th annual conf. on Design of communication. pp. 149-156.

55. *A 2d/3d hybrid geographical information system.* **Brooks, S. and Whalley, J. L.** ACM Press, 2005. GRAPHITE '05: Proc. of the 3rd international conf. on Computer graphics and interactive techniques in Australasia and South East Asia. pp. 323–330.
56. *Interactive 3d visualization of vector data in gis.* **Kersting, O. and Doellner, J.** 2002. GIS '02: Proceedings of the 10th ACM international symposium on Advances in geographic information systems. pp. 107–112.
57. *Managment and visualization of large, complex and time-dependent 3d objects in distributed gis.* **Shumilov, S., Thomsen, A., Cremers, A. B. and Koos, B.** ACM Press, 2002. GIS'02: Proceedings of the 10th ACM international symposium on Advances in geographic information systems. pp. 113–118.
58. *Adaptive streaming of 3d GIS geometries and textures for interactive visualisation of 3d city models.* **Haist, J. and Korte, P.** 2006. 9th AGILE International conf. on Geographic Information Science.
59. **MacEachren, A. M.** Moving geovisualization toward support for group work. *Exploring GeoVisualization.* Elsevier, 2005, pp. 445–461.
60. *A collaborative analysis tool for visualisation and interaction with spatial data.* **Manoharan, T., Taylor, H. and Gardiner, P.** 2002. Web3D '02: Proceedings of the seventh international conference on 3D Web technology. pp. 75–83.
61. *Dimensions: why do we need a new data handling architecture for sensor networks?* **Ganesan, D., Estrin, D. and Heidemann, J.** 2003, SIGCOMM Comput. Commun. Rev. 33, 1, pp. 143–148.
62. *Data visualization within urban models.* **Steed, A.** IEEE, 2004. TPCG '04: Proc. of the Theory and Practice of Computer Graphics 2004 (TPCG'04). pp. 9–16.

63. *Landscape visualisation*. **Scheepers, F.** ACM Press, 2001. In AFRIGRAPH '01: Proc. of the 1st international conf. on Computer graphics, virtual reality and visualisation. pp. 49–52.
64. *A framework for a dynamic interactive 3d gis for non-expert users*. **Walker, A. R., Pham, B. and Maeder, A.** 2003. GRAPHITE '03: Proc. of the 1st international conf. on Computer graphics and interactive techniques in Australasia and South East Asia. pp. 283–284.
65. *AICI-Advanced immersive collaborative interaction framework*. **Hur, H. et al.** Pyungchang, Korea, 2006. SCCE CAD/CAM Conference 2006. pp. 20-25.
66. OpenSG. **Reiners, D.** [Online] 1999-2007. [Cited: October 23, 2007.] <http://opensg.vrsource.org/trac>.
67. **ARToolkit**. The Augmented Reality Tool Kit . [Online] 2007. [Cited: October 2007, 23.] <http://sourceforge.net/projects/artoolkit>
68. **XSens motion technologies**. MTi. *Minitature Attitude and Heading Reference System*. [Online] 2007. [Cited: October 23, 2007.] <http://www.xsens.com/>.
69. **Microsoft Corporation**. SAPI 5.1 SDK. *Microsoft Speech Technologies*. [Online] 2007. [Cited: October 23, 2007.] <http://www.microsoft.com/speech/speech2007/downloads.mspix>.
70. *The Virtual Terrain Project: VTP*. **Discoe, B.** [Online] 1999-2007. [Cited: 8 8, 2007.] <http://vterrain.org>.
71. **OpenSceneGraph**. [Online] 2007. [Cited: October 23, 2007.] <http://www.openscenegraph.org/projects/osg>.
72. *Virtual Terrain Project: Port to OpenSG*. **Witzel, M.** OpenSG Application Gallery. [Online] 2007. [Cited: October 2007, 23.] <http://opensg.vrsource.org/trac/wiki/Gallery/VirtualTerrain>.
73. **GeoServer**. Open Source Web Feature Server. [Online] [Cited: October 23, 2007.] <http://geoserver.org/>.

74. **da Fonseca, M. and Jorge, J. J.** CALI: A Software Library for Calligraphic Interfaces. [Online] 2000. [Cited: October 23, 2007.] <http://immi.inesc-id.pt/cali/>.
75. **GraphViz.** Graph Visualization Software. [Online] 2007. [Cited: October 23, 2007.] <http://www.graphviz.org/>.
76. **Dynagraph.** Dynagraph for Windows. [Online] 2007. [Cited: October 23, 2007.] <http://www.dynagraph.org/dgwin/>.
77. **CEGUI.** Crazy Eddie's GUI System. [Online] 2006. [Cited: October 23, 2007.] <http://www.cegui.org.uk/phpBB2/index.php>.
78. **XMLBlaster.** [Online] 2007. [Cited: 8 8, 2007.] <http://www.xmlblaster.org/>.
79. **OSGA.** OpenSource Groupware Architecture. [Online] 2007. [Cited: October 23, 2007.] <http://www.osga.net/>.
80. **PostGIS.** Open Source Spatial Database. [Online] 2007. [Cited: October 23, 2007.] <http://postgis.refractory.net/>.
81. **Google.** Google Earth. [Online] 2007. [Cited: May 6, 2007.] <http://earth.google.com/>.
82. **National Aeronautics and Space Administration.** NASA World Wind. [Online] 2007. [Cited: May 6, 2007.] <http://worldwind.arc.nasa.gov/>.
83. **Holoscape Inc.** Unype: Multi-user Google Earth. [Online] 2005-2007. [Cited: November 1, 2007.] <http://www.unype.com/>.
84. International Geomagnetic Reference Field. [Online] **International Association of Geomagnetism and Aeronomy (IAGA)**, 2007. [Cited: June 11, 2007.] <http://www.ngdc.noaa.gov/IAGA/vmod/igrf.html>.
85. *Estimated Value of Magnetic Declination Tool.* **National Geophysical Data Center** USA. [Online] 2007. [Cited: June 12, 2007.] <http://www.ngdc.noaa.gov/seg/geomag/jsp/Declination.jsp>.

86. **Kato, H.** Inside ARToolkit. [Online] [Cited: November 6, 2007.] <http://www.hitl.washington.edu/artoolkit/Papers/ART02-Tutorial.pdf>.
87. **Cognitive Science Laboratory, Princeton University.** Wordnet. *A lexical database for the English language*. [Online] 2006. [Cited: November 4, 2007.] <http://wordnet.princeton.edu/>.
88. The GEO6 project. *Science with GNSS*. [Online] 2007. [Cited: November 1, 2007.] <http://www.gnss-geo6.org/>.
89. The Improve Project. [Online] 2003-2007. [Cited: 8 7, 2007.] <http://www.improve-eu.info/>.
90. *Drag-and-pop and drag-and-pick: Techniques for accessing remote screen content on touch- and penoperated systems.* **Baudisch, P. et al.** 2003. Proc. of INTERACT '03. pp. 57-64.
91. *Visual attention based information culling for distributed virtual environments.* **Beeharee, A. K., West, A. J. and Hubbard, R.** ACM Press, 2003. VRST '03: Proc. of the ACM symposium on Virtual reality software and technology. pp. 213–222.
92. *Multimodal interaction for distributed collaboration.* **Bolelli, L. et al.** ACM Press, 2004. ICMI '04: Proc. of the 6th international conference on Multimodal interfaces. pp. 327-328.
93. *Designing and deploying an information awareness interface* . **Cadiz, J. J., Venolia, G., Jancke, G. and Gupta, A.** ACM Press, 2002. CSCW '02: Proc. of the 2002 ACM conference on Computer supported cooperative work. pp. 314–323.
94. *Enabling geocollaborative crisis management through advanced geoinformation technologies.* **Cai, G., MacEachren, A. M., Sharma, R., Brewer, I., Fuhrmann, S., and McNeese, M.** Digital Government Research Center, 2005. dg.o2005: Proc. of the 2005 national conference on Digital government research. pp. 227–228.

95. **Card, S. K., Mackinlay, J. D. and Shneiderman, B.** *Information visualization*. 1999. pp. 1–34,57–61.
96. **da Rocha Barreto Pinto G., Medeiros, S. P. J., de Souza, J. M., Strauch, J. C. M. and Marques, C. R. F.** Spatial data integration in a collaborative design framework. *Commun. ACM*, 46(3). 2003, pp. 86–90.
97. *Previews and overviews in digital libraries: designing surrogates to support visual information seeking.* **Greene, S., Marchionini, G., Plaisant, C. and Shneiderman, B.** 2000, *Am. Soc. Inf. Sci.*, 51(4), pp. 380–393.
98. *Analysis experiences using information visualization.* **Hetzler, E. G. and Turner, A.** 2004, *IEEE Computer Graphics and Applications*, 24(5), pp. 22–26.
99. **Johansen, R.** *GroupWare: Computer Support for Business Teams*. New York, USA : The Free Press, 1988.
100. *Geotime information visualization.* **Kapler, T. and Wright, W.** 2004, *INFOVIS*, pp. 25–32.
101. *Geovisualization for knowledge construction and decision support.* **MacEachren, A. M., Gahegan, M., Pike, W., Brewer, I., Cai, G., Lengerich, E., Hardisty, F.** 2004, *IEEE Computer Graphics and Applications*, 24(1), pp. 13–17.
102. *Creating tangible interfaces by augmenting physical objects with multimodal language.* **McGee, D. R. and Cohen, P. R.** 2001. *International Conference on Intelligent User Interfaces*. pp. 113–119.
103. *Sensemaking processes of intelligence analysts and possible leverage points as identified through cognitive task analysis.* **Pirolli, P. and Card, S.** 2005. *Proceedings of the 2005 International Conference on Intelligence Analysis*.

104. *An adaptive visual analytics platform for mobile devices.* **Sanfilippo, A., May, R., Danielson, G., Baddeley, B., Riensche, R., Kirby, S., Collins, S., Thornton, S., Washington, K., Schrager, M., Randwyk, J. V., Borchers, B. and Gatchell, D.** IEEE, 2005. SC '05: Proceedings of the 2005 ACM/IEEE conference on Supercomputing. p. 74.
105. *Integrating 2d and 3d views for spatial collaboration.* **Schafer, W. A. and Bowman, D. A.** ACM Press, 2005. GROUP '05: Proceedings of the 2005 international ACM SIGGROUP conference on Supporting group work. pp. 41–50.
106. *Personalized peripheral information awareness through information art.* **Stasko, J. T., Miller, T., Pousman, Z., Plaue, C. and Ullah, O.** 2004, Ubicomp, pp. 18–25.
107. *Polaris: A system for query, analysis and visualization of multidimensional relational databases.* **Stolte, C., Tang, D. and Hanrahan, P.** 2002, IEEE Transactions on Visualization and Computer Graphics, 8(1), pp. 52–65.
108. *Interactive information visualization of a million items.* **Fekete, J. and Plaisant, C.** IEEE, 2002. Proc. of IEEE Symposium on Information Visualization. pp. 117–124.
109. *Improve: An innovative application for collaborative mobile mixed reality design review.* **Stork, A. et al.** 2006. Virtual Concept, Mexico.
110. *Deictic roles of external representations in face-to-face and online collaboration: Designing for change in networked learning environments.* **Suthers, D., Girardeau, L. and Hundhausen, C.** 2003. Proceedings of the International Conference on Computer Support for Context-Sensitive Aiding. pp. 173–182.
111. *Multimodal interface platform for geographical information systems (GeoMIP) in crisis management.* **Agrawal, P., Rauschert, I., Inochanon, K., Bolelli, L., Fuhrmann, S.,**

Brewer, I., Cai, G., MacEachren, A., and Sharma, R. ACM Press, 2004. ICMI '04: Proc. of the 6th international conference on Multimodal interfaces. pp. 339–340.

112. *The eyes have it: A task by data type taxonomy for information visualizations.* **Shneiderman, B.** IEEE, 1996. VL '96: Proc. of the 1996 IEEE Symposium on Visual Languages. p. 336.

113. **Leebmann, J.** *An Augmented Reality System for Earthquake Disaster Response.* [Volume XXXV, Part B] IAPRSIS, 2004.

114. *Interactive visual tools to explore spatio-temporal variation.* **Andrienko, N. and Andrienko, G.** ACM Press, 2004. AVI '04: Proceedings of the working conference on Advanced visual interfaces. pp. 417–420.

115. **Discoe, B.** The Virtual Terrain Project. *Terrain LOD: Runtime Regular-Grid Approaches.* [Online] 1999-2007. [Cited: Nov. 1, 2007.] <http://www.vterrain.org/LOD/Implementations/>

116. *Designing a human-centered multimodal GIS interface to support emergency management.* **Rauschert, I., Agrawal, P., Sharma, R., Fuhrmann, S., Brewer, I. and MacEachren, A.** ACM Press, 2002. GIS '02: Proc. of the 10th ACM international symposium on Advances in geographic information systems. pp. 119–124.

Appendix A. Test-bed configuration

A.1. Instruction script for the final user test session

Start-up and familiarization with the GUI (5 minutes)

1. The Powerwall application is started by the staff
2. User no. 1 starts the application
3. User no. 2 starts the application
4. Both users familiarize with the menu, with its commands and with the status bar
5. Both users open the control panel (settings) and set their preferences:
 - Activate TTS
 - Speech recognition
 - Set gesture as stippled line
 - Mini-map
6. Users move their main menu across the screen
7. Users are asked to show/hide the helpers (speech and gesture)

Navigation (5 minutes)

8. User no. 1 enter *shared* mode
9. The viewpoint is shared among the users and the power-wall
10. User no. 1 navigates through the GUI (different navigation modes have to be used)
11. User no. 1 goes into *local* mode

12. User no. 2 loads the model of a vehicle (lotus.aici)
13. User no. 2 enters *shared* mode and navigates the scene using Ellipsoid nav. Mode
14. User no. 2 activates the *SuperHand* mode from the setting panel
15. User no. 2 navigates through the *SuperHand* metaphor
16. User no. 2 goes into *local* mode
17. User no. 1 and 2 navigate through gestures and voice (different navigation modes have to be used)

Annotation (5 minutes)

18. User no. 1 creates an annotation via the GUI
19. User no. 1 modifies the type (color) of the annotation
20. User no. 2 picks the same annotation
21. User no. 2 queries the system to modify/clear/change the color of the same annotation
22. User no. 2 creates a new annotation
23. User no. 1 browses between annotations then opens one and select the “show to all” function
24. User no. 1 deletes a annotations via voice/gesture
25. User no. 1 associate an existing note to another part of the vehicle via voice/gesture
26. User no. 1 sends a mail with the note to another user
27. User no. 2 opens an annotation and it stores the original point of view corresponding to the annotation
28. Users freely interact with annotations

Materials (2 minutes)

29. User no. 1 selects a part of the vehicle and changes the material via GUI
30. User no. 1 selects a part of the vehicle and changes the material via voice/gesture
31. User no. 2 repeats the two previous tasks

View (3 minutes)

1. User no. 1 is asked to create a number of visual bookmark
2. User no. 1 is asked to recall previously stored visual bookmarks
3. User no. 1 enters shared
4. User no. 2 becomes the master
5. User no. 2 selects one of the visual bookmarks

Free running session (5 minutes)

A.2. Sample interaction graph in ASCII file

An example as stored in a text file can be seen below:

```
digraph G {  
0[label="root"];  
1[label="navigate"];  
2[label="pan"];  
3[label="fly"];  
4[label="tilt"];  
5[label="mouse"];  
6[label="speed"];  
7[label="view"];
```

```
8[label="store"];
9[label="load"];
10[label="select"];
11[label="left"];
12[label="right"];
13[label="front"];
14[label="rear"];
15[label="open"];
16[label="trento"];
17[label="belenzani"];
18[label="improve"];
19[label="create"];
20[label="query"];
21[label="building"];
22[label="road"];
23[label="forest"];
24[label="fence"];
25[label="annotation"];
26[label="next"];
27[label="previous"];
28[label="pick"];
29[label="erase"];
30[label="goto"];
31[label="show"];
32[label="assign"];
33[label="material"];
34[label="shader"];
35[label="light"];
36[label="office"];
```

```
37[label="bluesky"];
38[label="igdfoyer"];
39[label="glacier"];
40[label="one"];
41[label="two"];
42[label="three"];
43[label="four"];
44[label="five"];
45[label="off"];
46[label="grab"];
47[label="map"];
48[label="georss"];
49[label="cylinder"];
50[label="cone"];
51[label="villazzano"];
52[label="ingegneria"];
0->1 [label="gesture=Move;speech;dialog"];
1->2 [label="gesture=HLine;HLineLeft,HLineRight;dialog;speech=node,wordnet"];
1->3 [label="gesture=Ellipse;dialog;Circle;speech=node,wordnet"];
1->4 [label="gesture=VLine;VLineUp;VLineDown;dialog;speech=node,wordnet"];
1->5 [label="gesture=Rectangle;speech=node,wordnet;dialog"];
1->6 [label="gesture=Rectangle;dialog;"];
0->7 [label="gesture=Rectangle;dialog;speech=node,wordnet"];
7->8 [label="gesture=Cross;dialog;speech=node,wordnet"];
7->9 [label="gesture=Ellipse;speech=node,wordnet;dialog"];
8->46 [label="gesture=Rectangle,Ellipse,Circle;dialog;"];
7->10 [label="gesture=Ellipse;speech=node,wordnet;dialog"];
7->11 [label="gesture=Rectangle;speech=node,wordnet"];
7->12 [label="gesture=Rectangle;speech=node,wordnet"];
```

Graph-based multimodal interaction

7->14 [label="gesture=Rectangle;speech=node,wordnet"];
7->13 [label="gesture=Rectangle;speech=node,wordnet"];
7->51 [label="gesture=Rectangle;speech=node,wordnet;dialog"];
7->52 [label="gesture=Rectangle;speech=node,wordnet;dialog"];
0->15 [label="gesture=Ellipse;dialog;speech=node,wordnet"];
15->16 [label="gesture=Rectangle;dialog;speech=node,wordnet"];
15->17 [label="gesture=Rectangle;dialog;speech=node,wordnet"];
15->18 [label="gesture=Rectangle;speech=node,wordnet;dialog"];
0->19 [label="gesture=Rectangle;speech,dialog"];
19->21 [label="gesture=Rectangle;speech,dialog"];
19->22 [label="gesture=Rectangle;speech,dialog"];
19->23 [label="gesture=Rectangle;speech,dialog"];
19->24 [label="gesture=Rectangle;speech,dialog"];
19->48 [label="gesture=Circle;speech=node,wordnet,dialog"];
48->49 [label="gesture=Ellipse,Circle;speech=node,wordnet,dialog"];
48->50 [label="gesture=Cross;speech=node,wordnet,dialog"];
49->28 [label="gesture=Tap,dialog"];
50->28 [label="gesture=Tap,dialog"];
19->25 [label="gesture=Rectangle;dialog"];
25->27 [label="gesture=HLineLeft;dialog;speech=node,wordnet"];
25->26 [label="gesture=HLineRight;dialog;speech=node,wordnet"];
25->28 [label="gesture=Tap;"];
25->29 [label="gesture=WavyLine;Delete;Diamond;dialog;speech=node,wordnet"];
25->30 [label="gesture=Arrow;dialog;speech=node,wordnet"];
25->31 [label="gesture=Cross;dialog;speech=node,wordnet"];
0->20 [label="gesture=Rectangle;dialog;speech=node,wordnet"];
20->21 [label="gesture=Rectangle;dialog;speech=node,wordnet"];
20->22 [label="gesture=Rectangle;dialog;speech=node,wordnet"];
20->23 [label="gesture=Rectangle;dialog;speech=node,wordnet"];

Appendix A. Test-bed configuration

```
20->24 [label="gesture=Rectangle;dialog;speech=node,wordnet"];
20->47 [label="gesture=Rectangle;dialog;speech=node,wordnet"];
47->40 [label="gesture=Rectangle;dialog;speech=node,wordnet"];
47->41 [label="gesture=Rectangle;dialog;speech=node,wordnet"];
47->42 [label="gesture=Rectangle;dialog;speech=node,wordnet"];
47->43 [label="gesture=Rectangle;dialog;speech=node,wordnet"];
47->44 [label="gesture=Rectangle;dialog;speech=node,wordnet"];
20->25 [label="gesture=Rectangle;dialog"];
0->32 [label="gesture=Cross;speech;dialog"];
32->33 [label="gesture=Cross;speech;dialog"];
33->28 [label="gesture=Tap;dialog"];
32->34 [label="gesture=Rectangle;speech;dialog"];
34->40 [label="gesture=Tap;dialog"];
34->41 [label="gesture=Tap;dialog"];
34->42 [label="gesture=Tap;dialog"];
34->43 [label="gesture=Tap;dialog"];
34->44 [label="gesture=Tap;dialog"];
34->45 [label="gesture=Tap;dialog"];
32->35 [label="gesture=Cross;speech;dialog"];
35->36 [label="gesture=Cross;speech;dialog"];
35->37 [label="gesture=Cross;speech;dialog"];
35->38 [label="gesture=Cross;speech;dialog"];
35->39 [label="gesture=Cross;speech;dialog"];
}
```

A.3. Application specific voice commands

Application commands can only be defined from within the application, and thus, need to be integrated independently from the interaction graph. The configuration generator knows about application commands for insertion into the grammar.

[*“Settings”*] brings up a tab control to change globe app. settings

[*“Shared Mode”*] navigation is shared between users.

[*“Local Mode”*] local user is enabled to interact with sc/change of viewpoints.

[*“Navigation Mode”*] input is handled as navigation command.

[*“menu”*] brings up the ring menu (navigation).

[*“fly”*] fly navigation.

[*“pan”*] pan navigation.

[*“tilt”*] tilt navigation.

[*“speed”*] brings up the slider for speed adjustments.

[*“faster”*] increases the navigation speed by 50 per cent.

[*“slower”*] decreases the navigation speed by 50 per cent.

[*“stop”*] immediately stops positional navigation. However rotations are still possible.

[*“start”*] restarts the navigation when stopped previously

[*“superhand on/off”*] de/activates the motion tracked hand

[*“Edit Mode”*] input as seen as interacting with scene content.







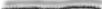









[*“menu”*] brings up the ring menu (interaction graph).

[*“back”*] goes to parent of current interaction node.

[*“help”*] brings up the helpers for speech and gesture

A.4. Assignable gestures

The following table depicts available gestures which can be assigned to the *gesture* attribute in the interaction graph. A further refinement (not shown) is the usage of their dashed version. The list of gestures which are allowed as input sketches is generated from the gestures present within the interaction graph.

			
Circle	Arrow	Cross	Wavy Line
			
Up Arrow	Ellipse	Horizontal Line	Line Left
			
Line Right	Arrow Left	Move	Rectangle
			
Point	Down Arrow	Vertical Line	Line Down



Line Up



Line



Arrow Right

A.5. Framework configuration

A.5.1. AICI

Main configuration

The main configuration file is mandatory for starting the system.

```
<?xml version="1.0" encoding="UTF-8"?> <AICIConfig>
<WindowsConfig filename="config/mono_window.xml"/>
<TrackerConfig filename="config/gt_MouseTracker.xml"/>
<WidgetConfig filename="map/cascade-menu.xml"/>
<Log4CPlusConfig filename="config/log4cplus.properties"/>
<ArtifactConfig>
<Artifact deviceId="0" userId="0" type="VIEWPOINT"/>
<Artifact deviceId="1" userId="0" type=
"OVERRIDE_NAVIGATOR"/>
<Artifact deviceId="2" userId="0" type="PEN"
filename="artifact/pen.wrl" />
<Artifact deviceId="3" userId="0" type="TOOL"/>
</ArtifactConfig>
</AICIConfig>
```

Window configuration

We provide two configurations for window settings. The first one is dedicated to start the application in a windows mode("mono_window.xml"), recognizable by `Screen fullscreen="false"`. It further allows to specify the window dimensions and origin on the used display via `origin="0 0"` `size="1024 768"`. For reasons of brevity we omit to include the full screen configuration.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Create Background/Pipe/Windows/Viewport -->
<WindowConfig>
  <Camera fov="90"/>
  <Background gradient="false"/>
  <Window id="0" name="mono_window1">
    <Screen fullscreen="false" origin="0 0" size="1024
768" screensize="-0.85 -0.64 0 0.85 -0.64 0 0.85 0.64
0 -0.85 0.64 0"/>
    <StereoVision type="none" headtracking="false"
interOcularDistance="0.06" eyeAngle="2"/>
    <Viewport id="0" viewportsize="0 1 0 1"/>
  </Window>
</WindowConfig>
```

Artifact configuration

Artifacts in IView are configured as follows:

```
<!DOCTYPE OpenTracker SYSTEM "opentracker.dtd">
<OpenTracker>
  <configuration>
</configuration>

  <!-- Device 0 -->
  <AiaSink station="0">
    <EventVirtualTransform>
      <AiaKeyboardSource number="0"/>
    </EventVirtualTransform>
  </AiaSink>

  <!-- Device 1 -->
  <AiaSink station="1">
    <EventTransform scale=".2 .2 .2">
      <NetworkSource port="6666" multicast-
address="192.168.253.42" number="1"/>
    </EventTransform>
  </AiaSink>

  <!-- Device 2 -->
  <AiaSink station="2">
    <AiaKeyboardSource number="2"/>
  </AiaSink>
```

```
<!-- Device 3 touchscreen -->
<AiaSink station="3">
    <AiaMouseSource mode="absolute" window="0"/>
</AiaSink>
</OpenTracker>
```

It should be noted that the specification of AiaSink station="3" is mandatory for running the system since it maps the input to the appropriate interaction with respect to the gesture and dialog modality.

A.5.2. VTP

```
<?xml version="1.0" encoding="utf-8"?>
<Terrain_Parameters>
<Name>Trento</Name>
<Filename>trento_plain_10.bt</Filename>
<Vertical_Exag>1.000000</Vertical_Exag>
<Min_Height>1</Min_Height>
<Nav_Style>0</Nav_Style>
<Nav_Speed>50.000000</Nav_Speed>
<Locations_File></Locations_File>
<Init_Location></Init_Location>
<Hither_Distance>0.000000</Hither_Distance>
<Accel>false</Accel>
<LOD_Method>0</LOD_Method>
<Pixel_Error>2.000000</Pixel_Error>
<Tri_Count>15000</Tri_Count>
<Tristrips>true</Tristrips>
```

Graph-based multimodal interaction

```
<Is_TIN>false</Is_TIN>
<Time_On>true</Time_On>
<Init_Time>104 4 8 7 0 0</Init_Time>
<Time_Speed>1.000000</Time_Speed>
<Texture>1</Texture>
<Num_Tiles>1</Num_Tiles>
<Tile_Size>2048</Tile_Size>
<Single_Texture>trento_single_precise-
0000.jpg</Single_Texture>
<Base_Texture>trento_</Base_Texture>
<Texture_Format>1</Texture_Format>
<MIP_Map>true</MIP_Map>
<Request_16_Bit>false</Request_16_Bit>
<Pre-Light>true</Pre-Light>
<PreLight_Factor>1.000000</PreLight_Factor>
<Texture_Gradual>false</Texture_Gradual>
<Allow_Sculpting>true</Allow_Sculpting>
<Tile_Threading>false</Tile_Threading>
<Tile_Cache_Size>20</Tile_Cache_Size>
<Cast_Shadows>false</Cast_Shadows>
<Color_Map>absolute_lower_elevation.cmt</Color_Map>
<Detail_Texture>false</Detail_Texture>
<DTexture_Name></DTexture_Name>
<DTexture_Scale>1.000000</DTexture_Scale>
<DTexture_Distance>1000.000000</DTexture_Distance>
<Roads>true</Roads>
<Road_File></Road_File>
<Highway>false</Highway>
```


Appendix A. Test-bed configuration

```
<Paved>false</Paved>
<Dirt>false</Dirt>
<Road_Height>1.000000</Road_Height>
<Road_Distance>6.000000</Road_Distance>
<Road_Texture>true</Road_Texture>
<Road_Culture>true</Road_Culture>
<Trees>false</Trees>
<Tree_File></Tree_File>
<Tree_Distance>5000</Tree_Distance>
<Fog>false</Fog>
<Fog_Distance>35.000000</Fog_Distance>
<Fog_Color>.8 .8 .8</Fog_Color>
<Structure_Distance>5000</Structure_Distance>
<Structure_Shadows>false</Structure_Shadows>
<Shadow_Resolution>1024</Shadow_Resolution>
<Content_File></Content_File>
<Trans_Towers>0</Trans_Towers>
<Tower_File></Tower_File>
<Vehicles>true</Vehicles>
<Vehicle_Size>1</Vehicle_Size>
<Vehicle_Speed>1</Vehicle_Speed>
<Number_of_Cars>50</Number_of_Cars>
<Sky>true</Sky>
<Sky_Texture></Sky_Texture>
<Ocean_Plane>false</Ocean_Plane>
<Ocean_Plane_Level>-20.000000</Ocean_Plane_Level>
<Depress_Ocean>false</Depress_Ocean>
```

Graph-based multimodal interaction

```
<Depress_Ocean_Level>-40.000000</Depress_Ocean_Level>
<Horizon>false</Horizon>
<Background_Color>0 50 155</Background_Color>
<Route_Enable>false</Route_Enable>
<Route_File></Route_File>
<Distance_Tool_Height>5</Distance_Tool_Height>
<HUD_Overlay>      ,0,0</HUD_Overlay>
<AR>true</AR>
<GPS>true</GPS>
</Terrain_Parameters>
```

A.5.3. Environment Parameters

Setting up the environment for starting the system is straightforward:

It is only necessary to map the root folder of the folder hierarchy to the virtual drive y:/ via subst y: [path to root folder]. All paths in the configuration files are dependent on the drive y:/. To adjust the system locally, several command line parameters are available to the user, which are order-dependent. These are in detail:

```
--config config/gt_tabletpc_mono.xml
```

This option specifies which main configuration file is to be used by the system. It bundles windows, artifacts and tracking information together. It is mandatory for startup.

```
--master 0
```

This option determines if the user is planned to be the peer reviewer who is enabled to guide a collaborative session, specifically for the shared navigation. 0 equals normal user, 1 makes him/her the peer reviewer. (optional, default value for this option is 0).

--datapath y:/iview/data,y:/iview/

This option specifies in which folders IView will look for datafiles. It is recommended not to be changed (mandatory).

--vtpconfig trento_hires_ar_texture.xml

This option loads a configuration of the underlying virtual terrain software, which allows to review automotives in distinct scenarios. Several configurations are provided in the VTP root directory (mandatory). In this configuration file, the user can further detail settings like default navigation speed, time of the day, etc.

--xmlblaster 192.168.253.54

With this option, the user specifies the IP-address of the machine where the central XMLBlaster server is running (optional).

--tabletpc 1

This option enables TabletPC specific functionalities such as dedicated navigation and interaction schemes (mandatory). 0 equals OFF, 1 equals ON.

--AR 1

This option lets the user determine if the system should support an Augmented Reality setup (optional). 0 equals OFF, 1 equals ON.

--speechconfig speech_user.xml

Lets the user specify which Speech configuration is used for the IMPROVE session. This is normally generated automatically dependent on the interaction graph (Optional).

--motiontracking 0

Enables the use of the XSens Motion Tracker for using the Superhand navigation metaphor (Optional), 0 equals OFF, 1 equals ON.

--mt-port 11

This is the serial port to which the XSens MotionTracker is connected (mandatory when connected).

A.5.4. Execution Batch

In the root folder of the installed system, “IView”, we provide two batch files with which it is possible to start the system.

Start_demo.bat starts IView in windowed mode.

Start_demo_full.bat starts IView in full screen mode.

The path variables are set automatically without the necessity to adjust system wide environment variables. To maintain multiple startup configurations, it is only necessary to copy one of the provided scripts and to change the startup parameters according to the previous paragraph in the last line of the batch file:

```
IViewrel --config config/gt_tabletpc_mono.xml --master 0
--datapath y:/iview/data,y:/iview/ --vtpconfig
trento_lores_AR_texture.xml -xmlblaster 192.168.253.58
--sensors 0 --tabletpc 1 --AR 1 --speechconfig
speech_user.xml --motiontracking 1 --mt-port 11
```

Appendix B. HW configuration(s)

B.1. Indoor design review setup

The setup of a sample indoor setup can be seen in Figure 68. The reviewing panel is located in front of a Large display (Powerwall). Each key reviewer interacts with the application in a multimodal way using microphones for speech input. Devices used for gesture input range from a Wacom Tablet to a single-touch touchscreen and a traditional mouse. A motion tracked hand serves as a new input metaphor used for effective navigation. As shown, the distributed, autonomous clients are interconnected via wireless network using a message oriented middleware which allows for the necessary collaboration amongst the reviewers.

**Final User Test at Elasis, Naples
-Hardware Setup-**

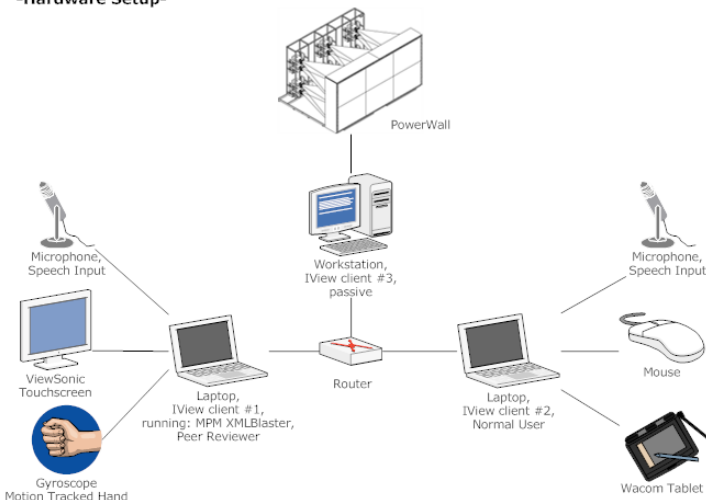


Figure 68. Automotive design review, system setup.

B.2. Integration of distributed data setup

Figure 69 shows the architecture for a distributed integration of data stemming from a Web Features Service using the communication protocol. A repository is used to store all data messages together with author, timestamp and message content in a relational database. Thus, the reviewing session becomes entirely recordable and can be examined for a later dissemination. The SensorBuilder is an application for managing incoming data from sensors for manipulation and assignment to data domains.

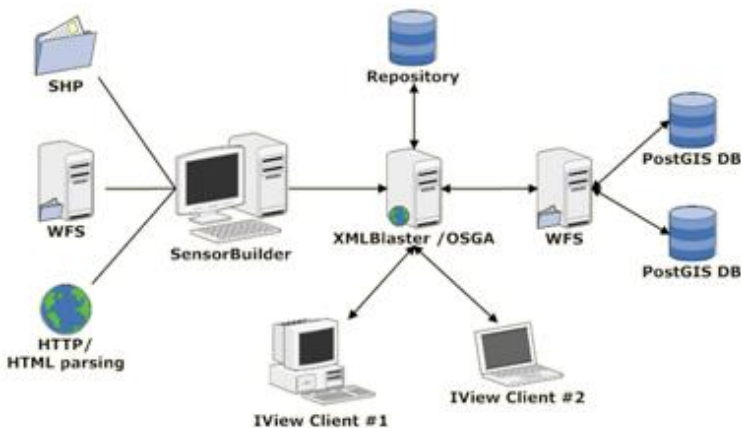


Figure 69. System setup

B.3. Augmented outdoor design review

The system is setup as depicted in Figure 70. The system components are interconnected via a Wireless connection. Access to geo-data is realized via a combination of WFS with a PostGIS Database holding the geo data.

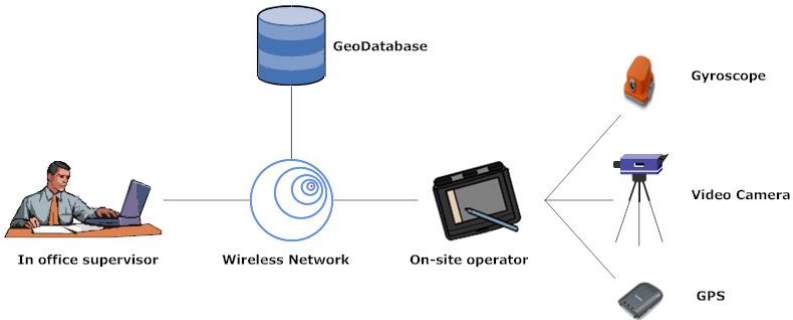


Figure 70. Augmented GIS setup

The user on-site is continuously positioned using a GPS device which is connected via a Bluetooth connection to a TabletPC. A video camera (or Webcam) with a sufficient resolution of at least 800 by 640 captures the scene in real-time, connected via USB. The orientation of the camera is tracked via an XSens MotionTracker which measures its local orientation. A TabletPC as input and display device was chosen over an HMD centered setup due to several reasons: First, interaction with the scene becomes much easier to handle when following the “*paper and pencil*” metaphor, and in fact we support multimodal interaction via gestures, voice and dialogs. Furthermore it is of high importance to have a reasonably large display size. The Web Feature Service GeoServer (73) is connected to a POSTGIS spatial database where environmental features like forests, streets are kept. The imprecision of using GPS and the imprecise fixture

of the MotionTracker on the camera have made it necessary to perform a calibration preceding each session to properly align the virtual and real camera. This is supported by a mechanism within the application which allows to fine tune the virtual camera. The concept of this approach to augmented GIS is to enrich a captured video with a properly aligned, location dependent virtual terrain, whose heightfield is superimposed at all times, in real-time over the real scene.

Appendix C. List of publications

C.1. Conferences

- [1] G. Conti, M. Witzel and R. de Amicis. A user-driven experience in the design of a multi-modal interface for industrial design review. (Under review) In *Proceedings of Human Factors in Telecommunication (HFT 2008)*, Kuala Lumpur, Malaysia, March 17-20, 2008.
- [2] M. Witzel, G. Conti and R. de Amicis. User-Centered Multimodal Interaction Graph for Design Reviews. (To appear) In *Proceedings of IEEE Virtual Reality 2008 conference*, Reno, USA, March 8-12, 2008.
- [3] M. Witzel, G. Conti and R. de Amicis. MIRAGE – sistema Mobile Interattivo in Realtà Aumentata per dati Gis ambiEntali. In *Conferenza Nazionale delle Associazioni Scientifiche per le Informazioni Territoriali e Ambientali (ASITA)*, Torino, Italy, October 6-9, 2007.
- [4] R. De Amicis, M. Witzel and G. Conti. Interoperable Networked Service-based infrastructure for Interactive. In *Proceedings of the NATO-OTAN Workshop on Development of a Prototype System for Sharing Information related to Acts of Terrorism to the Environment, Agriculture and Water systems (Ecoterrorism)*, Venice, July 1-3, 2007.
- [5] M. Witzel, M. Andreolli, G. Conti, R. De Amicis, B. De Araújo, R. Jota and J. Jorge. Collaborative Visualization of Sensor Data Through a Subscription-based Architecture. In *Proceedings of 5th Eurographics Italian Chapter*, Trento, Italy, Eurographics Association, Switzerland, pp. 173-179, February 14-17, 2007

- [6] P. Santos, A. Stork, T. Gierlinger, A. Pagani, B. Araújo, R. Jota, L. Bruno, J. Jorge, J. Madeiras Pereira, M. Witzel, G. Conti, R. DeAmicis, I. Barandarian, C. Paloc, O. Machui, G. Bodammer, J. M. Jiménez, D. McIntyre. IMPROVE: Collaborative Design Review in Mobile Mixed Reality. In *Proceedings of HCI 2007 International*, Beijing, P.R.China, July 22-27, 2007
- [7] P. Santos, A. Stork, T. Gierlinger, A. Pagani, B. Araújo, R. Jota, L. Bruno, J. Jorge, J. Madeiras Pereira, M. Witzel, G. Conti, R. DeAmicis, I. Barandarian, C. Paloc, M. Hafner, D. McIntyre. IMPROVE: Designing Effective Interaction for Virtual and Mixed Reality Environments. In *Proceedings of HCI 2007 International*, Beijing, P.R.China, July 22-27, 2007.
- [8] P.Santos, A. Stork, T. Gierlinger, A. Pagani, C. Paloc, I. Barandarian, M. Witzel, G. Conti., R. De Amicis, O. Machui, J. M. Jimenez, B. Araújo, J. Jorge, G. Bodammer. IMPROVE: an innovative application for collaborative mobile mixed reality design review. In *Proceedings of Virtual Concept 2006*, Playa del Carmen, Mexico, 2006.
- [9] P. Santos, A. Stork, T. Gierlinger, A. Pagani, B. Araújo, R. Jota, W. K. Müller-Wittig, J. Jorge, J. Madeiras, M. Witzel, G. Conti, R. De Amicis, I. Barandiaran, C. Paloc, M. Hafner, D. McIntyre. IMPROVE: Advanced Displays and Interaction Techniques for Collaborative Design Review Virtual reality. *Second international conference, ICVR 2007*: Held as part of HCI International 2007, Beijing, China, Proceedings (Lecture Notes in Computer Science 4563), July 22-27, 2007.
- [10] M. Witzel, G. Conti, R. De Amicis. A Message-Based Annotation System for Collaborative Design Review. In *Proceedings of the Fourth Eurographics (European*

Association of Computer Graphics) – Italian Chapter conference, Catania, February 22-24, 2006.

- [11] G. Conti, M. Witzel, R. De Amicis. ICAM - Interface for Collaborative Asynchronous Design Review on Mobile Device. In *Proceedings of Topics in Automatic 3D Modelling and Processing Workshop*, Verona, March 17, 2006.
- [12] I.T. Hernádvölgyi, G. Ucelli, M. Witzel, O. Symonova, L. Delpero, R. De Amicis. Shape Semantics from Shape Context. *Workshop on Modeling and Retrieval of Context in KI 2004*, University of Ulm, Germany, ISSN 1613-0073, September 20-21, 2004.

C.2. Journals

- [13] P. Santos, A. Stork, T. Gierlinger, A. Pagani, C. Paloc, Barandarian, G. Conti, R. de Amicis, M. Witzel, O. Machui, J. M. Jiménez, B. Araujo, J. Jorge and G. Bodammer. IMPROVE: An innovative application for collaborative mobile mixed reality design review. In *International Journal on Interactive Design and Manufacturing*, Springer Paris. ISSN: 1955-2513, 2007.

C.3. Short papers in journals (without review)

- [14] M. Witzel, M. Einhoff, G. Ucelli. ELIN – The Electronic Newspaper Initiative. In *topics, issue 16*, Reports of the INI-GraphicsNet, 2004.
- [15] G. Conti, M. Witzel, R. de Amicis. IMPROVE: a configurable multimodal interaction dialogue for design review. In *topics, issue 3*, Reports of the INI-GraphicsNet, 2007.