Habilitationsschrift

Applications of Mixed Reality

ausgeführt zum Zwecke der Erlangung der venia docendi
im Habilitationsfach ”Angewandte Informatik”

eingereicht im Mai 2009 an der
Technischen Universität Wien
Fakultät für Informatik

von

Mag.rer.nat. Dr.techn. Hannes Kaufmann
Ospelgasse 1-9/1/39, 1200 Wien
kaufmann@construct3d.org

Wien, am 27. Mai 2009

Contents

Introduction

This thesis gives an overview of the author's scientific work in previous years. It reflects the author's ambition to develop applications of mixed reality which are beneficial to society as a whole or to specific groups of people. Providing and deploying high-end mixed reality hardware and software applications to multiple users and larger target groups finally raises questions of scalability, robustness, design and affordability of the technology involved. They trigger scientific questions and developments in return. All of these aspects will be touched in this work.

The first part of the introduction defines the scientific domain and various problems therein followed by a discussion of the author's contribution in this area. The individual publications that constitute the remainder of the thesis are briefly discussed and put in context.

Definitions

Since the area "Applications of Mixed Reality" – chosen as the title of this thesis – is very broad it is important to establish common terms in the beginning.

In order to classify virtual reality (VR) research Milgram and Kishino [1] published a taxonomy 15 years ago. Although the field widened and diversified over the years their work still provides a rough framework which helps to classify any work done in this area. We refer to the virtual continuum (Figure 1) as discussed in Milgram and Kishino's paper. The virtual continuum represents a continuous set of (infinite) possibilities between real environments and fully virtual environments (VEs). All environments within that range (except the extremes of fully real and virtual environments) are considered mixed realities.

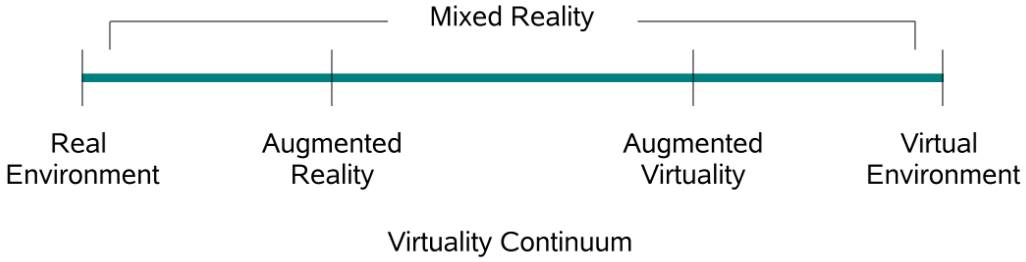


Figure 1: The Virtuality Continuum

Further on the authors specify and classify hardware and software environments within the virtual continuum and define six classes of "hybrid display environments". In a second paper [2] they add a seventh class. Nevertheless given the broad range of virtual reality hardware and setup variations available today it is neither always clear nor easy to identify into which category a specific setup falls.

Related to Milgram’s taxonomy many applications presented in this thesis belong to class 6 (defined in [1]) which states

> “6. Completely graphic but partially immersive environments (e.g. large screen displays) in which real physical objects in the user's environment play a role in (or interfere with) the computer generated scene, such as in reaching in and "grabbing" something with one's own hand […].”

They mention further

> “We note in addition that Class 6 displays go beyond Classes 1, 2, 4 and 5, in including directly viewed real-world objects also. As discussed below, the experience of viewing one's own *real* hand directly in front of one's self, for example, is quite distinct from viewing an image of the same real hand on a monitor, and the associated perceptual issues (not discussed in this paper) are also rather different. Finally, an interesting alternative solution to the terminology problem posed by Class 6 as well as composite Class 5 AR/AV displays might be the term*"Hybrid Reality"* (HR), as a way of encompassing the concept of blending many types of distinct display media.”

Milgram and Kishino described three additional dimensions that distinguish different mixed reality systems: Extent of world knowledge (i.e. degree of knowledge of the real world by the application), reproduction fidelity (visual quality) and extent of Presence metaphor. Presence in short can be defined as a subjective phenomenon of the sensation of being in a virtual environment [3, 4]. It is the most researched dimension of the three and of high importance when designing new applications [5]. Different concepts and interpretations of presence have been discussed [5]. Whereas some applications require full presence of users others might require shared and equal awareness of the real and virtual e.g. in educational applications where teachers are outside the VE guiding students. With different Mixed Reality (MR) setups these variations can be achieved while maintaining a high level of presence in all cases. Appropriate and corresponding examples of application areas and target groups will be mentioned later.

Our work fulfills as well Azuma’s definition of Augmented Reality (AR) [6], who defines AR as systems that have the following three characteristics:

1) Combine real and virtual
2) Interactive in real time
3) Registered in 3-D

The presented research covers a wide range of environments which are always interactive in real time but fulfill items 1 and 3 to varying degrees. The variety of systems is better encompassed by the term Mixed Reality or even Hybrid Reality – the latter term was not in use after Milgram and Kishino coined it.

System Architecture

A wide variety of MR hardware and software setups are imaginable and have been built in the past. However all share a common general system architecture. The five key elements of an MR system [7] comprise input and output devices whose spatial position and orientation might be tracked, a computing platform with a powerful graphics processor and a VR/MR software framework handling input, output and application behavior. The most important part is the user (or multiple, collaborating users) working on a certain task and interacting with the system.

Up to the end of the 20th century high-end graphics workstations dominated the VR/MR market. Due to the exponential performance increase of graphics hardware (due to the growing gaming market) starting in the mid 1990's PC graphics hardware reached an acceptable level of performance at the beginning of this decade. It is the main platform used in MR systems nowadays. High quality real-time rendering became state of the art in MR systems [8]. Diverse technologies for tracking are in use to determine the location of input, output devices, the user or specific body parts of the user up to full body motion capture [7]. Optical tracking evolved into the de-facto standard in recent years. In stationary indoor setups infrared-optical tracking based on retro-reflective markers is frequently used whereas outdoor applications utilize computer vision algorithms to perform natural-feature tracking (in combination with high-sensitive GPS or differential GPS, compasses, inertial and other sensors).

Input from (tracked) devices is typically handled by so called tracking middleware [9, 10] which supports a wide range of devices, pre-processes input events and passes them to the MR application. Most software frameworks are based on scene graph libraries for example open source toolkits such as Studierstube [11], VR Juggler [12], Avango [13] or commercial ones such as 3DVIA Virtools [14] and provide additional support for (stereo) output devices.

A comprehensive overview of VR technology including input, output devices and graphics architectures is given in [7]. Further details on hardware, software and application requirements with additional chapters on design and implementation approaches and evaluations are to be found in [15]. Looking at an early book on this topic [16] gives an insight on how technology changed over the years.

Within Augmented Reality a lot of research focused on mobile devices and applications for these devices in recent years. Stationary and mobile systems have a few opposing characteristics. While devices can be tracked at very high precision (sub-millimeter accuracy) within a stationary setup tracking data is usually imprecise (from centimeters to meters) in mobile setups. In general conditions in an indoor environment can be controlled well whereas outdoor controllability might be low. Full immersion is harder to achieve (or even unwanted) in a mobile setting compared to a stationary one. These and other characteristics reflect on the type of application scenario that can be realized. The author's work is limited to application areas which require stationary setups only, mainly because of high precision requirements. This means that all users interacting with the application are always inside a room or connected via a distributed setup in multiple rooms.

Little systematic work has been done on software design and implementation of virtual environments. Wilson et al. presented a structured approach called "Virtual Environment Development Structure" (VEDS) [17] which can be used as a methodology for MR software engineering. The author of this thesis followed a similar approach as described by Hix & Gabbard [18] namely usability engineering of virtual environments.

There are various approaches for application development. As Bimber [19] states

> "We believe that a rich pallet of different display technologies, mobile and non-

> mobile, must be considered and adapted to fit a given application so that one can choose the most efficient technology".

The author fully agrees with an application-centered and user-centered approach. User requirements have always been taken into account while designing and developing applications presented in this thesis. The whole range that the virtual continuum offers should be considered to find optimal solutions for specific end users with a specific task and goal in mind.

Application Areas of Interest

Ongoing is the search for so called killer applications of virtual reality, augmented reality and mixed realities in general. The defining criterion for a killer application is usually commercial success. For many reasons – a topic that filled many articles and continuously triggers discussions – such an application or area of application has not been found yet.

Nevertheless as Jaron Lanier, VR pioneer who coined the term "virtual reality", states in [20]:

> "[…] As used in industrial technology, there's no question that virtual reality has already been a success. You can't buy a car today that wasn't designed using it. And you can't put gas in that car that wasn't made out of oil that was discovered using virtual reality through an oil field simulation. Most new drugs are made in a process assisted by virtual reality. There are many other examples. […]"

Mixed reality applications in industrial areas of design, prototyping and marketing have been successful since the beginnings as well as applications in architecture and naturally data visualization. Applications in entertainment are mainly successful in theme parks. Due to high (hardware) costs and a lack of maturity of early devices (including some novel "VR devices" that target the gaming market nowadays) they did not find broad acceptance in the consumer gaming market yet.

In addition there are other very successful applications areas which are typically not considered mainstream. They are of primary interest in the context of the author's work:

- Psychology
- Medical Sciences
- Education
- and combinations of these.

MR technologies are nowadays used in all major directions of theoretical and practical work within psychology [21-23] – in research, education, therapy and rehabilitation, and in most of the psychological academic disciplines – cognitive, organizational, social, clinical, differential, instructional psychology, as well as in philosophical and neuropsychological studies of conscience. This relatively new field is being rapidly accelerated in universities, and partly in hospitals and rehabilitation centers, mainly in North America and the EU (particularly, in Spain, UK, Germany, the Netherlands and Italy), in somehow lesser extent – in Israel and several Pacific countries. The results of these projects, either finished or currently in progress, include for example pilot and/or professional MR systems which support psychological assessment and treatment of anxieties, phobias and post-traumatic stress disorder (exposure therapy) [24];

cognitive behavioral therapy in general; MR environments to conduct psychological tests in 3D of e.g. spatial abilities [25, 26]; instructional MR applications in experimental psychology – as diverse as for example psychology of visual perception and psychology of manipulative persuasion – and many more.

An area with higher success rates than traditional in vivo therapy is clinical neuropsychology where mixed reality applications are used as a therapeutic tool for e.g. treatment of anxieties [27] and also post traumatic stress disorder (PTSD), chronic pain and many more.

In some aspects these applications fall into and overlap with the medical domain which is another successful and growing MR application area. Examples are MR-based rehabilitation systems for e.g. physical rehabilitation or for rehabilitation of stroke patients; the assessment and treatment of impairments found in persons with central nervous system dysfunction including Alzheimer's Disease, Vascular Dementia, Parkinson's Disease [28]; assessment after traumatic brain injury; MR support for the disabled etc. In addition there is ongoing research regarding MR (specifically AR and VR) simulators for medical training e.g. to acquire specific surgical skills [29].

A large part of this thesis focuses on applications of mixed reality in education (overviews in [30], [31], [32](chapter 2)) and training. Therefore the problems and challenges related to this application domain will be discussed briefly (a summary is given in [33]).

It is interesting to note that nearly all of the educational projects reported in literature reached a certain point where trial studies and evaluations were conducted and then the projects ended e.g. [34, 35]. No reports about continuous progress, no iterative development process and ongoing tests can be found in literature which go a step further. Therefore no development process comparable to usability engineering [18] took place to optimize the application regarding usability and effectiveness for end users. There are exceptions e.g. work by Adamo-Villani [36, 37] but only few.

A major challenge of this young application area is that there are no studies proving the effectiveness of MR learning yet. In this context “effectiveness” measures the learning outcome achieved using an educational MR application in comparison to traditional teaching. The difficulty in comparing learning outcome is the comparability or rather incomparability of traditional and MR learning scenarios. Since a virtual learning environment is typically designed to provide added benefit to learners (compared to a traditional setting) it might introduce advanced or new learning contents. These might be hard or impossible to do in traditional environments. Therefore it is difficult to find learning tasks for evaluation purposes that do not penalize one scenario e.g. by over-simplifying an MR learning task to be solvable within reasonable time by traditional methods whereby eliminating the strength of the MR environment.

A body of work has been done on the theoretical pedagogical foundations of educational VEs [30, 38] whereas pedagogical guidelines about how to teach in VEs are rare. Evaluations of MR learning environments with a large number of users (>50) are difficult to find as well.

The author’s contribution can be found in the latter areas: psychology in the broad sense defined above – with recent work reaching into the medical domain – and education which includes training. A specific field opened up between psychology and education motivated by the work on the first application [39] namely the education and training of spatial abilities.

Motivation, Challenges and Context of the Conducted Research

In order to solve three dimensional mathematical but especially geometrical problems, spatial abilities are an important prerequisite [40-42]. During his studies of mathematics and geometry with the aim of becoming a teacher, the author gave countless private lessons to students of these subjects. A personal observation was that many students had difficulties solving tasks that required spatial visualization skills and spatial thinking. To get passing grades they used strategies such as learning construction steps by heart without fully understanding spatial problems and their solutions in 3D space.

Geometry Education

With the emergence of mixed reality technologies it became possible to immerse users in artificial worlds that are impossible or difficult to reproduce in reality. A number of training studies have shown the usefulness of VR in training spatial ability [25, 43]. However, little to no work has been done towards systematic development of VR applications for practical educational purposes in this field.

In our first paper we introduce Augmented Reality to mathematics and geometry education. The simultaneous sharing of real and virtual space in AR is an ideal match for computer-assisted collaborative educational settings. We have developed an application called Construct3D, a three dimensional geometric construction tool specifically designed for mathematics and geometry education. The main advantage of Construct3D to student learning is that students actually see three dimensional objects in 3D space which they until now had to calculate and construct with traditional methods. Augmented reality provides them with an almost tangible picture of complex three dimensional objects and scenes. It enhances, enriches and complements the mental images that students form when working with three dimensional objects.

However, there are a number of requirements and challenges for a mixed reality tool with the aim of effectively improving spatial skills. They motivated work on Construct3D and have not been addressed by existing systems, nor studied in an educational context before:

- No VR/AR application for actual use in high school or higher education has been developed with the main purpose of improving spatial skills.
- No VR/AR application existed prior to Construct3D that offered the flexibility to dynamically construct and modify 3D geometric content directly in 3D space. For a definition and explanation of “dynamic geometry” please refer to [44, 45].
- Hardly any evaluations could be found in literature giving hints to the actual learning transfer from a VR/AR learning environment to the real world.

The first four papers in this thesis summarize the author’s ongoing efforts towards filling these gaps. The first paper titled “Mathematics and geometry education with collaborative augmented reality” [39] is chronologically the earliest from these included. It introduces to the area and presents from a mainly technological point of view the first version of the application as well as multiple hardware setups for educational use.

Regarding the earlier mentioned short duration of related educational projects our work goes one step further. Construct3D is one of the longest developed educational applications so far. We studied how ongoing technological improvements (over a course of 8 years) can be integrated into an MR system and looked at pedagogical questions such as how to adapt contents of the current high-school curriculum to the new learning environment.

Construct3D was evaluated multiple times with over 500 users in total (students, teachers and experts) over the course of 5 years. A summary of three usability evaluations and findings is given in “Summary of Usability Evaluations of an Educational Augmented Reality Application” [46].

Based on the second evaluation Construct3D was redesigned with the help of a professional graphics designer to improve usability and effectiveness within the Lab@Future [47] EU FP5 IST project. In that context general design guidelines were formulated in “*Designing Immersive Virtual Reality for Geometry Education*” [48] which are partly applicable to other (educational) MR applications as well. This publication is not included herein but part of the appendix of the thesis. We describe improvements in the user interface and visual design and report on practical experiences with using our system for actual teaching of high school students, and present initial quantitative data on the educational value of such an approach.

The third paper is chronologically the last published paper in the collection. It closes the circle of work done on Construct3D and presents the end of an evolution from a merely technological focus to an application-centered focus. In “Dynamic Differential Geometry in Education” [45] educational dynamic geometry was introduced to the specific domain of differential geometry. Construct3D is the only available tool that can be used to study this application area. The focus lies on differential geometry which can be explored in a new way using three-dimensional dynamic geometry in MR.

Problems and Challenges

In general Construct3D was praised by teachers and students who had used it and questions arose about disseminating it to schools. During evaluations major hindrances became obvious which avoided the installation of mixed reality environments in schools. Some of these issues are not specific to the educational sector but apply to other end user groups as well. According to interviews with teachers three main reasons hinder the dissemination of mixed reality technology to schools:

1. Costs of the hardware and software environment.
2. The need for maintenance of all equipment which requires personnel and again generates costs (even if the application is very robust and mature).
3. The most effective and by users most preferred setup supports only a limited number of users (2-3) and questions arose about how to support larger groups of users.

The author was trying to work on solutions to these problems in order to be able to spread MR technology to a bigger community. Some of these problems triggered new scientific questions.

Regarding the support of multiple users (the third problem on the list) the following two papers “Long Distance Distribution of Educational Augmented Reality Applications” [49] and “Multiple Head Mounted Displays in Virtual and Augmented Reality Applications” [50] discuss different approaches to provide a virtual environment to larger groups of users.

A Matter of Costs

Tracking is a critical part of any MR system and typically the most expensive one therefore relates to the first problem on the list. Costs of tracking systems especially of optical tracking systems which provide the highest accuracy (as needed by an application such as Construct3D) have always been high. They are sold by a small number of companies worldwide and due to a small end user market for such systems prices were kept constantly high for over a decade. In a configuration tailored to a room-sized multi-user environment, all of the existing optical tracking systems have price tags in the range of tens of thousands of Euros. While corporate entities and well-funded research laboratories will not be deterred by such amounts, it is the author’s first-hand experience that many smaller educational institutions, especially secondary schools, operate on tightly constrained budgets that leave little, if any, room for an expense of this magnitude, even if third-party subsidies are available. The urging matter of costs finally led to the development of iotracker - a low-cost infrared optical tracking system - which the author initiated, conceptually designed and guided. The main goal was to reduce costs of high quality optical tracking systems without sacrificing quality i.e. speed or accuracy. Iotracker is commercially available (http://www.iotracker.com) and has already been a success in this respect. Since the introduction of iotracker which is available at a fraction of the price of other systems some vendors have already reduced their prices or introduced new products at lower prices.

Affordable tracking technology in return opens up new end user markets and new application areas. High quality tracking technology should not only be limited to members of the academic community, but also to artists, game designers, educators, small commercial application developers and all with an interest in Mixed Reality.

The publication about iotracker “*Affordable Infrared-Optical Pose-Tracking for Virtual and Augmented Reality*” [51] is not included in this collection but in the appendix that accompanies this work. In the paper we describe the hard- and software of a new low-cost infrared-optical pose-tracking system for room-sized virtual environments. The system consists of 4-8 shutter-synchronized 1394-cameras with an optical bandpass filter and infrared illuminator. All image-processing is done in software on an attached workstation. Preliminary results indicate low latency (20-40ms), minimal jitter (RMS less than 0.05mm/ 0.02°), sub-millimeter location resolution and an absolute accuracy of ±0.5cm. Up to twenty independent 6-DOF targets can be tracked in real-time with up to 60Hz.

Costs of a full MR system are still higher than those of computing equipment traditionally used in educational institutions and need to be justified well. If an MR system can be used in multiple courses and different subjects there will be a higher degree of utilization, it will be considered more useful than if there is only one application available - therefore it is more likely to be acquired.

In “Simulating Educational Physical Experiments in Augmented Reality” [52] we present an augmented reality application for physics, more specifically mechanics education. It is based on the same technological setup as used for Construct3D (including hardware and the software framework) and builds on experiences gathered during the development of the geometry application. PhysicsPlayground can be perfectly integrated into physics courses. It allows students to actively build own experiments and to study them in a three-dimensional virtual world.

Spatial Abilities

As mentioned in the beginning no mixed reality application for actual use in high school or higher education has ever been developed with the main purpose of improving spatial skills. The author's motivation was to help students develop correct mental models of three dimensional problems and to improve their spatial thinking - to enable them to find solutions to geometric problems themselves.

Based on psychological studies about mathematics and geometry education, national school authorities (e.g. Austria [53]) consider improving spatial abilities one of the main goals of geometry education. Spatial abilities present an important component of human intelligence. Spatial ability is a heterogeneous construct that comprises a number of sub-factors, such as mental rotation, visualization, and environmental orientation [54-57]. Many studies have shown that spatial abilities can be improved by well-designed training (e.g. [58]). Geometry education has proven to be a powerful means of improving these skills [59].

The hypothesis for our work on Construct3D was that if students see three-dimensional objects directly in 3D space and can interactively construct, touch and modify abstract geometric objects, they later build mental models of complex geometric situations more easily in real life. In order to verify this hypothesis the author initiated an interdisciplinary research project "Educating Spatial Intelligence with Augmented Reality" (FWF P16803-N12). We evaluated the effects of an AR-based geometry training on spatial abilities. A summary of this project is given in "*Virtual and Augmented Reality as Spatial Ability Training Tools" [60]* which is not part of the thesis but included in the appendix. In this paper we first review studies that used MR technologies to study different aspects of spatial ability. Then results and findings are presented from an MR large-scale study with 215 students that investigated the potential of an AR application (Construct3D) to train spatial ability. Further project results were presented in [61-63].

Our findings in the project indicate that augmented reality can be used to develop useful tools for spatial ability training. However within the training period of six weeks we were not able to measure significant improvements in traditional spatial ability tests neither by training with Construct3D nor within the control groups. Although two results were surprising and intriguing: (1) Classical paper-pencil spatial ability tests seemed to be not sensitive to some aspects of spatial performance, possibly due to their two-dimensional nature and limited difficulty range, and (2) in the control group (without any training) there were marked individual differences in performance increases between pre- and post-test. This suggests that individuals differ in their "learning potential" with respect to spatial abilities.

These findings led us to the idea of developing a new spatial ability test that (a) measures spatial abilities in three-dimensional space, and (b) includes a training phase, so that learning potential as well as performance status can be measured. We started a follow-up project "Development of an Augmented-Reality Dynamic Spatial Test" (FWF P19265) to develop the "dStar" test. Its ongoing development and first results of a pre-study are presented in "Design of a Virtual Reality Supported Test for Spatial Abilities" [26]. A large evaluation study (> 250 participants) using the new test is currently ongoing and will be finished later this year. In this project a complete low-cost MR system (based on iotracker) running the dStar application was installed at the psychology institute of our partner university. An achievement is the fact that the whole test is run by psychologists only and the system is maintained by them as well. Due to our experience we were able to build a robust, mature test environment which is in daily use (from 10am-6pm) and can be run and maintained by non-VR-experts during the whole project duration.

Interaction & Interfaces

The development of new interactive MR applications is very time-consuming and therefore costly. A survey in 1992 [64] showed that about 50% of application development code (and time) was used for the applications' user interfaces. From our experience with developing complex MR environments, we estimate that this may hold true for MR applications as well. Decoupling user interface code from application code could be a first step towards standardizing building blocks (widgets) for user interaction. These should be independent of the application and the higher level VR framework. Since most application developers are using tracking middleware [9, 10] in order to get hardware support for interaction devices (to avoid implementing it themselves), tracking middleware might serve as a common ground for the implementation of future user interaction standards.

Most code regarding menu system interaction can be decoupled from the application and handled by tracking middleware or an "interaction middleware". A standardized XML file, specifying the menu layout and widgets could be passed by the application onto the tracking/interaction middleware which then sends higher level commands back to the application in case of user interaction. Widget behavior itself could all be handled by the middleware if desired. A first approach towards this goal is described in "Towards a Universal Implementation of 3D User Interaction Techniques" [65].

Such an approach has further advantages not immediately obvious at first sight. In distributed MR applications a consistent application state is of primary importance (as mentioned i.e. [49]). An advantage of this approach is that consistent application behavior can be guaranteed in distributed environments with minimal network load. Interaction is handled by a central process which directly exchanges data with the tracking framework and is accessible by all distributed clients.

Furthermore there are advantages to pervasive MR scenarios where new interaction devices are added and removed on the fly during application runtime. In a pervasive setting interaction techniques must be accommodated to local hardware setups. In this context new devices might want to utilize additional interaction techniques other than the ones presently used in the application. Building repositories of interaction techniques is a good way to move on to the goal of a flexible and adaptive (tracking) system for that purpose. The scripted approach described in our paper is one possibility for choosing a suitable interaction technique for each setup and location in a dynamic adaptation process.

Ongoing and Future Developments

More work in the area of decoupling user interfaces and interaction techniques from the application is currently conducted within the VISION EU FP7 project “Immersive interface technologies for life-cycle human-oriented activities in interactive aircraft-related virtual products”. The above mentioned paper is a first step in that direction but we are working on a more general approach, introducing a new middleware layer on top of the tracking framework using an XML protocol to exchange object information with the application. This approach is independent of the VR framework used. First results are expected later this year.

The author is convinced that innovative software solutions alone will not suffice to attract a large base of end users to use MR systems. Innovations in hardware development will be required not only to reduce costs of current systems but also to introduce socially accepted (stereoscopic) 3D output devices. Auto-stereoscopic displays have been improved in recent years, smaller, light-weight HMDs and foldable, bendable displays are under development. The search for innovative hardware solutions is also why we contributed in [50] and [51] to this domain.

The development of our low cost optical tracking system opened up new application areas. Since we are building hardware and software ourselves the platform can flexibly be adapted to various application areas and needs.

In the running EU FP7-ICT project PLAYMANCER “PlayMancer: A European serious gaming 3D Environment“ [66] the tracking system is extended to perform full body motion capturing. This is done in the context of physical rehabilitation of chronic pain patients - mainly lower back and shoulder/neck pain. Full body motion capture will be used within a serious gaming environment as one input modality to first assess patients’ needs and to enable patients to interact with their body with the game in clinical therapy.

In [50] we noticed that HMDs’ VGA cables constitute a security risk if multiple users with HMDs collaborate. Currently we are working on a wireless modification of our HMDs using Amimon’s Wireless Uncompressed High-Definition (WHDI) video dongle which is the first to allow transmission of an uncompressed stereo VGA signal in sufficient resolution and update rate wirelessly. This would make the approach described in [50] more practicable.

As mentioned before a recent national research project allows us to conduct a comprehensive evaluation study using the dStar test in order to study general and differential effects of training on several components of spatial ability. Spatial ability research in MR is a growing field with only a small number of groups actively working in it yet. Within 6 years and two research projects we gathered and presented new findings to the spatial ability research community, developed a completely new spatial ability test in 3D which is currently being used with over 250 students. This is not the end of the road. Based on findings with the current test and new ideas we plan to continue our work in this field for years to come.

Overview of Selected Papers and Contributions of the Author

All ideas presented in the papers of this thesis were developed by Hannes Kaufmann. Furthermore all papers were written by the author and the evaluation work either performed by him or in case of large scale evaluations with the help of a team of psychologists mentioned as co-authors. The author has made a significant contribution to all these publications. The implementation work has either been done by the author alone or guided by him (in case of [49, 52, 65] where he acted in the role of both active researcher and master/PhD thesis supervisor.
In the following the major contributions of these publications are discussed.

Mathematics and geometry education with collaborative augmented reality [39]

This paper introduces Construct3D. Construct3D is a 3D geometric construction tool specifically designed for mathematics and geometry education. It is based on the mobile collaborative augmented reality system ‘‘Studierstube’’. We describe our efforts in developing a system for the improvement of spatial abilities and maximization of transfer of learning. In order to support various teacher–student interaction scenarios we implemented flexible methods for context and user dependent rendering of parts of the construction. Together with hybrid hardware setups they allow the use of Construct3D in today’s classrooms and provide a test-bed for future evaluations. Means of application and integration in mathematics and geometry education at high school as well as university level are being discussed. Anecdotal evidence supports our claim that Construct3D is easy to learn, encourages experimentation with geometric constructions and improves spatial skills.

Summary of Usability Evaluations of an Educational Augmented Reality Application [46]

We summarize three evaluations of Construct3D, which have been conducted in 2000, 2003 and 2005 respectively. Repeated formative evaluations with more than 100 students guided the redesign of the application and its user interface throughout the years. Results regarding usability and simulator sickness are discussed and guidelines provided on how to design augmented reality applications utilizing head-mounted displays.

Dynamic Differential Geometry in Education [45]

Dynamic geometry allows to study geometric properties under movement. This paper introduces differential geometry in educational dynamic geometry software. New functionality such as a Frenet frame, center and circle of curvature in arbitrary curve points, and others were implemented in Construct3D. We developed examples which enable teachers and learners to intuitively explore properties of interesting curves, to visualize contact of higher order between curves and surfaces, to construct Meusnier's sphere, Dupin's indicatrix and more.
This paper demonstrates the application of MR technology to higher geometry whereas the focus is not on technology but on extending the application area itself.

Long Distance Distribution of Educational Augmented Reality Applications [49]

For distance education utilizing shared Virtual or Augmented Reality (VR/AR) applications, reliable network distribution of educational content is of prime importance. In this paper we summarize the development of software components enabling stable and reliable distribution of an existing educational AR application for geometry education. Our efforts focus on three main areas: (1) For long distance distribution of Open Inventor scene graphs, throughout a wide area IP network, a TCP based network protocol was implemented in Distributed Open Inventor. (2) A tracking middleware was extended to support sending tracking data unicast instead or in addition to sending multicast messages. (3) Multiple adaptations in our geometry application were required to improve scalability, robustness and reliability. We present an early evaluation with high school students in a distant learning, distributed HMD setup and highlight final results.

Multiple Head Mounted Displays in Virtual and Augmented Reality Applications [50]

With the introduction of low cost head mounted displays, prices of HMD-based virtual reality setups dropped considerably. In various application areas personal head mounted displays can be utilized for groups of users to deliver different context sensitive information to individual users. We present a hardware setup that allows attaching 12 or more HMDs to a single PC. Finally we demonstrate how a collaborative, educational, augmented reality application is used by six students wearing HMDs on a single PC simultaneously with interactive frame rates.

Simulating Educational Physical Experiments in Augmented Reality [52]

PhysicsPlayground is an augmented reality application for mechanics education. It utilizes a recent physics engine developed for the PC gaming market to simulate physical experiments in the domain of mechanics in real time. Students are enabled to actively build own experiments and study them in a three-dimensional virtual world. A variety of tools are provided to analyze forces, mass, paths and other properties of objects before, during and after experiments. Innovative teaching content is presented that exploits the strengths of our immersive virtual environment. PhysicsPlayground serves as an example of how current technologies can be combined to deliver a new quality in physics education.

Design of a Virtual Reality Supported Test for Spatial Abilities [26]

This paper focuses on the development of a new spatial ability test in virtual reality (VR). This test measures the ability to visualize and mentally manipulate three-dimensional objects directly in 3D space, and should thus have a higher ecological validity than previous spatial ability tests. Items are viewed through head mounted displays and manipulated by means of a wireless pen input device. As a dynamic tests consisting of a pretest, a training phase, and a posttest it does not only measure a person’s current status but also his or her learning potential. Monitoring user interactions in a VR environment allows to measure test performance in ways not possible with traditional means. We describe design and development of the test and will present results of a pre-study with 240 participants conducted in early 2008.

Towards a Universal Implementation of 3D User Interaction Techniques [65]

This paper presents a versatile - write once, use everywhere – approach of standardizing the development of three-dimensional user interaction techniques. In order to achieve a platform- and application-independent implementation of 3D interaction techniques (ITs), we propose to implement the related techniques directly in the tracking middleware. Therefore OpenTracker, a widely used tracking framework was extended by a Python binding to allow straight forward scripting of ITs. We cluster existing 3D ITs, into those which can be fully, partly or not implemented in the tracking middleware. A number of examples demonstrate how various interaction techniques can quickly and efficiently be implemented in the middleware and are therefore fully independent of the underlying application. We hint at how this approach can be used to decouple menu system control from the application with the final goal to help establishing standards for 3D interaction.

References

[1] P. Milgram and F. Kishino, "A Taxonomy of Mixed Reality Visual Displays," *IEICE Transactions on Information Systems,* vol. E77-D, Dec. 1994.

[2] P. Milgram, H. Takemura, A. Utsumi, and F. Kishino, "Augmented reality: a class of displays on the reality-virtuality continuum," *Proceedings of Telemanipulator and Telepresence Technologies,* pp. 282-292, 1994.

[3] C. Coelho, Tichon, J., Hine, T. J., Wallis, G., Riva, G., "Media Presence and Inner Presence: The Sense of Presence in Virtual Reality Technologies," in *From Communication to Presence: Cognition, Emotions and Culture towards the Ultimate Communicative Experience - Festschrift in honor of Luigi Anolli,* G. Riva, Anguera, M.T., Wiederhold, B.K., Mantovani, F., Ed. Amsterdam: IOS Press, 2006.

[4] M. J. Schuemie, P. van der Straaten, M. Krijn, and C. A. van der Mast, "Research on presence in virtual reality: A Survey," *Cyberpsychol Behavior,* vol. 4, pp. 183-201, Apr 2001.

[5] G. Riva, F. Davide, and W. A. Ijsselsteijn, *Being There: Concepts, Effects and Measurements of User Presence in Synthetic Environments*: IOS Press, 2003.

[6] R. Azuma, "A Survey of Augmented Reality," *Presence - Teleoperators and Virtual Environments,* vol. 6, pp. 355-385, 1997.

[7] G. C. Burdea and P. Coiffet, *Virtual Reality Technology,* 2nd ed.: Wiley-IEEE, 2003.

[8] T. Akenine-Möller, E. Haines, and N. Hoffman, *Real-Time Rendering,* 3rd ed. Natick, MA, USA: A.K. Peters Ltd., 2008.

[9] R. Taylor, T. C. Hudson, A. Seeger, H. Weber, J. Juliano, and A. T. Helser, "VRPN: A Device-Independent, Network-Transparent VR Peripheral System," in *ACM Symposium on Virtual Reality Software & Technology 2001 (VRST 2001)* Banff, Alberta, Canada, 2001.

[10] G. Reitmayr and D. Schmalstieg, "An Open Software Architecture for Virtual

Reality Interaction," in *ACM Symposium on Virtual Reality Software & Technology 2001 (VRST 2001)* Banff, Alberta, Canada, 2001.

[11] D. Schmalstieg, A. Fuhrmann, G. Hesina, Z. S. Szalavári, L. M. Encarnacao, M. Gervautz, and W. Purgathofer, "The Studierstube augmented reality project," *Presence - Teleoperators and Virtual Environments*, vol. 11, pp. 33-54, Feb 2002.

[12] C. Cruz-Neira, A. Bierbaum, P. Hartling, C. Just, and K. Meinert, "VR Juggler – An Open Source Platform for Virtual Reality Applications," in *40th AIAA Aerospace Sciences Meeting and Exhibit* Reno, Nevada, USA, 2002.

[13] R. Kuck, J. Wind, K. Riege, and M. Bogen, "Improving the AVANGO VR/AR Framework - Lessons Learned," in *5. Workshop Virtuelle und Erweiterte Realität der Fachgruppe VR/AR* VDTC Magdeburg, Germany, 2008.

[14] Virtools, "Virtools Dev User Guide," http://www.virtools.com, 2001.

[15] K. M. Stanney, *Handbook of Virtual Environments: Design, Implementation, and Applications*: Lawrence Erlbaum Associates, 2002.

[16] N. I. Durlach, A. S. Mavor, and N. R. C. (U.S.), *Virtual Reality: Scientific and Technological Challenges*: National Academies Press, 1995.

[17] J. R. Wilson, R. M. Eastgate, and M. D'Cruz, "Structured Development of Virtual Environments," in *Handbook of Virtual Environments: Design, Implementation, and Applications*, K. M. Stanney, Ed.: Lawrence Erlbaum Associates, 2002, p. 1232

[18] D. Hix and J. L. Gabbard, "Usability Engineering of Virtual Environments," in *Handbook of Virtual Environments - Design, Implementation, and Applications*, K. M. Stanney, Ed. Mahwah, New Jersey: Lawrence Erlbaum Associates, 2002, pp. 681-699.

[19] O. Bimber and R. Raskar, *Spatial Augmented Reality: Merging Real and Virtual Worlds*: A K Peters LTD, 2005.

[20] J. J. Heiss, "The Future of Virtual Reality: Part Two of a Conversation with Jaron Lanier," http://java.sun.com/features/2003/02/lanier_qa2.html ed: Sun Developer Network (SDN), 2003.

[21] A. A. Rizzo, J. G. Buckwalter, J. S. MCGee, T. Bowerly, C. van der Zaag, U. Neumann, M. Thiebaux, L. Kim, L. Pair, and C. Chua, "Virtual environments for assessing and rehabilitating cognitive/functional performance," *Presence - Teleoperators and Virtual Environments*, vol. 10, pp. 359-374, 2001.

[22] A. A. Rizzo, J. G. Buckwalter, and C. Van Der Zaag, "Virtual environment applications in clinical neuropsychology," in *Handbook of Virtual Environments - Design, Implementation, and Applications*, K. M. Stanney, Ed. Mahwah, New Jersey: Lawrence Erlbaum Associates, 2002, pp. 1027-1064.

[23] H. Hoffman, "Virtual reality: a new tool for interdisciplinary psychology research," *CyberPsychology and Behavior,* vol. 1, pp. 195-200, 1998.

[24] A. A. Rizzo, J. Pair, P. J. McNerney, E. Eastlund, B. Manson, J. Gratch, R. Hill, M. Roy , and B. Swartout, "Design and Development of a VR Therapy Application for Iraq War Veterans with PTSD," in *Technology and Informatics*. vol. 111, J. D. Westwood, et al., Ed. Amsterdam , NL: IOS Press, 2005, pp. 407-413.

[25] A. A. Rizzo, J. G. Buckwalter, U. Neumann, C. Kesselman, M. Thiebaux, P. Larson, and A. Van Rooyen, "The Virtual Reality Mental Rotation Spatial Skills Project," *CyberPsychology and Behavior,* vol. 1, pp. 113-120, 1998.

[26] H. Kaufmann, M. Csisinko, I. Strasser, S. Strauss, I. Koller, and J. Glück,

"Design of a Virtual Reality Supported Test for Spatial Abilities," in *Proceedings of the 13th International Conference on Geometry and Graphics (ICGG),* Dresden, Germany, 2008, pp. 122-123.

[27] M. B. Powers and P. M. G. Emmelkamp, "Virtual reality exposure therapy for anxiety disorders : A meta-analysis," *Journal of anxiety disorders,* vol. 22, pp. 561-569, 2008.

[28] J. S. McGee, C. van der Zaag, J. G. Buckwalter, M. Thiebaux, A. Van Rooyen, U. Neumann, D. Sisemore, and A. A. Rizzo, "Issues for the assessment of visuospatial skills in older adults using Virtual Environment technology," *Cyberpsychology & Behavior,* vol. 3, pp. 469-482, JUN 2000.

[29] F. Tendick, M. Downes, T. Goktekin, M. C. Cavusoglu, D. Feygin, X. Wu, R. Eyal, M. Hegarty, and L. W. Way, "A Virtual Environment Testbed for Training Laparoscopic Surgical Skills," *Presence,* vol. 9, pp. 236-255, June 2000 2000.

[30] F. Mantovani, "VR Learning: Potential and Challenges for the Use of 3D Environments in Education and Training," in *Towards CyberPsychology: Mind, Cognitions and Society in the Internet Age,* G. Riva and C. Galimberti, Eds. Amsterdam: IOS Press, 2001.

[31] C. Youngblut, "Educational uses of virtual technology," VA: Institute for Defense Analyses, Alexandria IDA Document D-2128, 1998.

[32] H. Kaufmann, "Geometry Education with Augmented Reality," Ph.D. Thesis, Vienna University of Technology, 2004, p. 179.

[33] M. Bricken, "Virtual reality learning environments: potentials and challenges," *SIGGRAPH Computer Graphics,* vol. 25, pp. 178-184, 1991.

[34] G. Taxén and A. Naeve, "CyberMath: Exploring Open Issues in VR-Based Learning," *SIGGRAPH 2001 Educators Program,* vol. SIGGRAPH 2001 Conference Abstracts and Applications, pp. 49-51, 2001.

[35] C. Dede, M. C. Salzman, and R. B. Loftin, "ScienceSpace: Virtual Realities for Learning Complex and Abstract Scientific Concepts," *Proceedings of IEEE VRAIS '96,* pp. 246-252, 1996.

[36] N. Adamo-Villani, E. Carpenter, and L. Arns, "An immersive virtual environment for learning sign language mathematics," in *ACM Proceedings of SIGGRAPH 2006 Educators Program,* Boston, MA, USA, 2006.

[37] N. Adamo-Villani and K. Wright, "SMILE: an immersive learning game for deaf and hearing children," in *ACM Proceedings of SIGGRAPH 2007 Educators Program* San Diego, USA: ACM Publications, 2007.

[38] W. Winn, "A Conceptual Basis for Educational Applications of Virtual Reality," Technical Report TR 93-9, 1993.

[39] H. Kaufmann and D. Schmalstieg, "Mathematics and geometry education with collaborative augmented reality," *Computers & Graphics,* vol. 27, pp. 339-345, Jun 2003.

[40] D. L. Brown and G. H. Wheatley, "Relationship between spatial knowledge," in *Proceedings of the 11th Annual Meeting, North American Chapter of the International Group for the Psychology of Mathematics Education,* C. A. Maher, G. A. Goldin, and R. B. Davis, Eds. Brunswick, NJ: Rutgers University, 1989, pp. 143-148.

[41] E. Fennema and J. A. Sherman, "Sex-related differences in mathematics achievement, spatial visualization, and affective factors," *Americal Educational Research Journal,* pp. 51-71, 1977.

[42] R. B. Guay and E. McDaniel, "The relationship between mathematics

achievement and spatial abilities among elementary school children," *Journal for Research in Mathematics Education*, pp. 211-215, 1977.

[43] N. Durlach, G. Allen, R. Darken, R. L. Garnett, J. Loomis, J. Templeman, and T. E. von Wiegand, "Virtual environments and the enhancement of spatial behavior: Towards a comprehensive research agenda," *Presence - Teleoperators and Virtual Environments*, vol. 9, pp. 593-615, Dec 2000.

[44] U. H. Kortenkamp, "Foundations of Dynamic Geometry," Swiss Federal Institute of Technology, Zürich, Switzerland, PhD Dissertation1999.

[45] H. Kaufmann, "Dynamic Differential Geometry in Education," *to appear in Journal for Geometry and Graphics*, vol. 13 (accepted for publication), 2009.

[46] H. Kaufmann and A. Dünser, "Summary of Usability Evaluations of an Educational Augmented Reality Application," in *HCI International Conference (HCII 2007)*. vol. 14, LNCS 4563, R. Shumaker, Ed. Beijing, China: Springer-Verlag Berlin Heidelberg, 2007, pp. 660-669.

[47] V. Baudin, M. Faust, H. Kaufmann, V. Litsa, D. Mwanza, A. Pierre, and A. Totter, "The Lab@Future Project: Moving towards the future of e-Learning," in *IFIP International Federation for Information Processing, Technology Enhanced Learning* Toulouse, France: Springer Boston, 2004, pp. 3 - 18.

[48] H. Kaufmann and D. Schmalstieg, "Designing Immersive Virtual Reality for Geometry Education," in *Proceedings of IEEE Virtual Reality Conference 2006*, Alexandria, Virginia, USA, 2006, pp. 51-58.

[49] H. Kaufmann, M. Csisinko, and A. Totter, "Long Distance Distribution of Educational Augmented Reality Applications," *Proceedings of Eurographics 2006 (Educational Papers)*, pp. 23-33, 2006.

[50] H. Kaufmann and M. Csisinko, "Multiple Head Mounted Displays in Virtual and Augmented Reality Applications," *International Journal of Virtual Reality*, vol. 6, pp. 43-50, 2007.

[51] T. Pintaric and H. Kaufmann, "Affordable Infrared-Optical Pose-Tracking for Virtual and Augmented Reality," in *Proceedings of "Trends and Issues in Tracking for Virtual Environments" Workshop, IEEE VR 2007*, Charlotte, NC, USA, 2007, pp. 44-51.

[52] H. Kaufmann and B. Meyer, "Simulating educational physical experiments in augmented reality," in *ACM SIGGRAPH ASIA 2008 educators programme* Singapore: ACM, 2008.

[53] "Lehrplan der AHS-Oberstufe," in *Österreichisches Bundesministerium für Bildung, Wissenschaft und Kultur*: http://www.bmbwk.gv.at/, 1989.

[54] D. F. Lohman, "Spatial ability: A review and reanalysis of the correlatonal literature," Stanford University School of Education, Aptitude Research Project, Stanford, CA Tech. Rep. No. 8, 1979.

[55] D. F. Lohman, "Spatial abilities as traits, processes, and knowledge," in *Advances in the psychology of human intelligence*. vol. 4, R. J. Sternberg, Ed. Hillsdale, NJ: Erlbaum, 1988, pp. 181-248.

[56] J. B. Carroll, *Human cognitive abilities. A survey of factor-analytic studies*. Cambridge, UK: Cambridge University Press, 1993.

[57] H. Stumpf and J. Eliot, "A structural analysis of visual spatial ability in academically talented students," *Learning and Individual Differences*, vol. 11, pp. 137-151, 1999.

[58] E. Souvignier, "Training räumlicher Fähigkeiten. [Training spatial abilities.]," in *Handbuch Kognitives Training*, K. J. Klauer, Ed. Göttingen: Hogrefe, 2001,

pp. 293-319.

[59] G. Gittler and J. Glück, "Differential Transfer of Learning: Effects of Instruction in Descriptive Geometry on Spatial Test Performance," *Journal of Geometry and Graphics,* vol. 2, pp. 71-84, 1998.

[60] A. Dünser, H. Kaufmann, K. Steinbügl, and J. Glück, "Virtual and Augmented Reality as Spatial Ability Training Tools," in *Proceedings of the CHINZ 2006,* University of Canterbury, Christchurch, New Zealand, 2006, pp. 125-132.

[61] A. Dünser, "Trainierbarkeit der Raumvorstellung mit Augmented Reality [Trainability of spatial ability with augmented reality] " in *Fakulty of Psychology* University of Vienna. Doctoral thesis: Vienna, 2005, p. 242.

[62] J. Glück, A. Dünser, K. Steinbügl, and H. Kaufmann, "Warning: Subtle aspects of strategy assessment may affect correlations among spatial tests," *Perceptual and Motor Skills,* pp. 123-140, 2007.

[63] H. Kaufmann, K. Steinbügl, A. Dünser, and J. Glück, "General Training of Spatial Abilities by Geometry Education in Augmented Reality," *Annual Review of CyberTherapy and Telemedicine: A Decade of VR,* vol. 3, pp. 65-76, 2005.

[64] B. A. Myers and M. B. Rosson, "Survey on user interface programming," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, New York, NY, USA, 1992, pp. 195–202.

[65] M. Csisinko and H. Kaufmann, "Towards a Universal Implementation of 3D User Interaction Techniques," in *Mixed Reality User Interfaces: Specification, Authoring, Adaptation (MRUI'07), Workshop Proceedings*, Charlotte, North Carolina, USA, 2007, pp. 17-25.

[66] E. Kalapanidas, C. Davarakis, F. Fernández-Aranda, S. Jiménez-Murcia, O. Kocsis, T. Ganchev, H. Kaufmann, T. Lam, and D. Konstantas, "PlayMancer: Games for Health with Accessibility in Mind," *Communications & Strategies no. 73,* pp. 105-120, 2009.

PERGAMON

Computers & Graphics 27 (2003) 339–345

COMPUTERS & GRAPHICS

www.elsevier.com/locate/cag

Mathematics and geometry education with collaborative augmented reality

Hannes Kaufmann*, Dieter Schmalstieg

Interactive Media Systems Group, Vienna University of Technology, Favoritenstrasse 9-11/188, Vienna A-1040, Austria

Abstract

Construct3D is a 3D geometric construction tool specifically designed for mathematics and geometry education. It is based on the mobile collaborative augmented reality system "Studierstube". We describe our efforts in developing a system for the improvement of spatial abilities and maximization of transfer of learning. In order to support various teacher–student interaction scenarios we implemented flexible methods for context and user dependent rendering of parts of the construction. Together with hybrid hardware setups they allow the use of Construct3D in today's classrooms and provide a testbed for future evaluations. Means of application and integration in mathematics and geometry education at high school as well as university level are being discussed. Anecdotal evidence supports our claim that Construct3D is easy to learn, encourages experimentation with geometric constructions and improves spatial skills.

Keywords: Augmented reality; Mathematics education; Geometry education; Spatial intelligence

1. Motivation

Spatial abilities present an important component of human intelligence. The term spatial abilities covers five components, spatial perception, spatial visualization, mental rotations, spatial relations and spatial orientation [1]. Generally, the main goal of geometry education is to improve these spatial skills. In a long-term study by Gittler and Glück [2], the positive effects of geometry education on the improvement of spatial intelligence have been verified. Various other studies [3,4] conclude that spatial abilities can also be improved by virtual reality (VR) technology. However, little to no work has been done towards systematic development of VR applications for practical education purposes in this field.

To fill the gap of next-generation VR interfaces for mathematics and geometry education we are developing a 3D geometric construction tool called Construct3D [5] that can be used in high school and university education (Fig. 1). Our system uses augmented reality (AR) [6] to provide a natural setting for face-to-face collaboration of teachers and students. The main advantage of using AR is that students actually see 3D objects which they until now had to calculate and construct with traditional (mostly pen and paper) methods. We speculate that by working directly in 3D space, complex spatial problems and spatial relationships can be comprehended better and faster than with traditional methods.

It is important to note that while geometry education software shares many aspects with conventional 3D computer-aided design (CAD) software at a first glance, its aims and goals are fundamentally different. Geometry education software is not intended for generating polished results, but puts an emphasis on the construction process itself. While relatively simple geometric primitives and operations will suffice for the intended audience of age 10–20, the user interface must be both intuitive and instructive in terms of the provided visualizations and tools. Commercial CAD software offers an overwhelming variety of complex features and often has a step learning curve. In contrast, geometry

---

*Corresponding author. Tel.: +43-1-58801-18860.

E-mail addresses: kaufmann@ims.tuwien.ac.at (H. Kaufmann), schmalstieg@ims.tuwien.ac.at (D. Schmalstieg).

0097-8493/03/$ - see front matter © 2003 Elsevier Science Ltd. All rights reserved.
doi:10.1016/S0097-8493(03)00028-1

Fig. 1. Students are working with Construct3D in our standard lab setup. In the left image they inscribe a sphere in a cone, the right image shows a simple example from vector algebra. Images generated as live video capture with computer overlays.

educators are interested in simple construction tools that expose the underlying process in a comprehensive way.

For productive use in the classroom, a number of circumstances must be accommodated: Support for a variety of social settings including students working alone and together, a teacher working with a student or teaching a whole class, a student or the whole class taking an exam, etc. Collaboration in these situations is largely determined by roles, and the teacher should be able to retain control over the activities. Moreover, it is not realistic to expect that schools can afford extensive installations of expensive equipment such as used in our lab, and therefore the software must run on a variety of immersive and non-immersive hardware platforms including heterogeneous and hybrid setups.

This paper presents our current prototype of such an AR based geometry education tool, including hard- and software, user interface design, and initial experiences.

2. Related work

Construct3D combines four research areas: geometry, pedagogy, psychology and AR. We will give a short overview of closely related work in these areas.

A large body of work has been done on 3D modeling in general. However, although 3D input devices with six degrees of freedom (6DOF) have been used to enhance modelers, little modeling has been done in immersive VR systems. A good overview of 3D modeling systems with 6DOF input devices can be found in the work of Mine [7]. Mine's CHIMP [8] was used to study user interaction techniques. Similar goals were driving the work on DesignSpace by Chapin [9], Bowman's Conceptual Design Space [10] and SeamlessDesign by Kiyokawa [11].

A large number of researchers have been working on VR applications for pure educational use ([12–15] and many others). A very good summary of existing educational applications is given by Mantovani [16]. We want to point out CyberMath [17] which is an avatar-based shared virtual environment aimed at improving mathematics education, combining the areas of VR, pedagogy and mathematics. Unlike our AR approach, CyberMath is intended for remote rather than face-to-face collaboration, and currently uses desktop VR with no support of immersive displays. In contrast, dedicated educational dynamic geometry desktop applications such as Geometer's Sketchpad [18], Cindarella [19], Euklid [20] and Cabri Geometry [21] support 2D geometry only.

Regarding spatial intelligence, a recent article by Durlach et al. [22] gives a very good overview of work that has already been done in the research area of enhancing spatial skills within virtual environments but mainly identifies the indispensable need for comprehensive future research in this area. Interesting work on gender differences has been done by Rizzo et al. [4] and Larson et al. [23]. Gittler and Glück [2] study how courses in descriptive geometry improve students' spatial intelligence. The main component of their results is that there is a significant positive effect of descriptive geometry instruction on performance in spatial ability tasks.

Finally, AR is a rapidly evolving area of computer graphics and user interface research. A good overview is given in the survey by Azuma [6] and its recent update by Azuma et al. [24]. However, despite its obvious appeal for face-to-face collaboration, we are unaware of any AR application directly concerned with geometry education.

3. Application design

3.1. Basic construction functions

Construct3D is based on the *Studierstube* system recently described by Schmalstieg et al. [25]. *Studierstube* uses AR to allow multiple users to share a virtual space. We use see-through HMDs capable of overlaying computer-generated images onto the real world, thereby achieving a combination of virtual and real world, allowing natural communication among users. The latest version of *Studierstube* allows the mix and match of heterogeneous output devices such as personal HMD, virtual workbench, conventional monitors, and input through a variety of tracking devices. All these devices appear to act as interfaces to a single distributed system.

The current version of Construct3D offers a basic set of functions for the construction of primitives such as points, lines, planes, cubes, spheres, cylinders and cones. Construction functions include intersections, normal lines and planes, symmetry operations, and taking measurements. Boolean operations based on the Open-Cascade tool [26] have been added which (for instance) enable learning about intersection curves of second-order surfaces. Currently new functions for constructing general and special curves (e.g. conic sections, B-Splines) and surfaces (e.g. quadrics, B-Spline surfaces) are being implemented.

Construct3D promotes and supports exploratory behavior through dynamic geometry, i.e., all geometric entities can be continuously modified by the user, and dependent entities retain their geometric relationships. For example, moving a point lying on a sphere results in the change of the sphere's radius.

All construction steps are carried out via direct manipulation in 3D using a stylus tracked with six degrees of freedom. AR affords that users see their own body and hand as well as the effects of their actions while working, so the construction process physically involves the students and resembles handcraft more than traditional computer operation. We believe that this is a key factor in the potential success of using AR for teaching geometry.

Necessary system operations such as selection of primitive type, load, delete, undo, etc. are mapped to a hand-held tracked panel, the personal interaction panel (PIP) [27]. The PIP allows the straightforward integration of conventional 2D interface elements like buttons, sliders, dials etc. as well as novel 3D interaction widgets. The haptic feedback from the physical props guides the user when interacting with the PIP, while the overlaid graphics allow the props to be used as multi-functional tools.

3.2. Layers

We are using a 3D-layer system very similar to the one used in current image editing programs. Our 3D-layers offer the possibility to arrange parts of a construction into overlapping sub-spaces that can be controlled independently. This feature is particularly powerful in conjunction with distributed multi-user operation, where every user has a personal display for which visibility of layers can be controlled independently.

The current implementation allows the arbitrary selection of visibility per user and per layer. We have implemented three basic modes:

- independent mode, i.e., every student can only see the elements constructed by himself,
- collaborative mode, i.e., everything is visible to everybody,
- teacher mode, i.e., a special user—the teacher—can set visibility with a user/layer matrix of controls on the PIP.

Consider a teacher working on a construction with students watching him and work on the model themselves by request. The whole construction is visible to all users. If later the teacher wants the students to practice on the same construction again, he switches to "independent" mode while the application is still running. Now each student can only see the immutable specification and the elements that he constructed himself without being influenced by the work of the teacher or fellow students. If needed, the teacher is able to switch his own construction or a reference solution on again so that some or all students can see it. The full solution to a construction task can be available from the beginning for reference purposes in a separate set of layers, and be progressively revealed by the teacher.

4. Hybrid hardware setups

To complement the diverse teacher–student interaction scenarios that are possible on the software side with practical hardware solutions for an educational environment we created various hybrid hardware setups. Realistically not all scenarios can be done in schools with equipment similar to our standard lab equipment of rather expensive tracking systems, head mounted displays and stereoscopic video projections. However, many components such as PC workstations with accelerated graphics and inexpensive projection systems are becoming feasible for classroom use. We are evaluating the following hardware setups.

4.1. The augmented classroom

The setup consists of two wearable AR kits composed of back pack computer, stereoscopic see-through head mounted display with camera, and custom pinch gloves for two-handed input. One kit can be worn by the teacher, the second one is available for use by students. Both users can move around freely, since the kits are equipped with battery power for all devices and wireless LAN cards for communication. Furthermore, there is a small table, serving as a place for collaboration between the two users (see Fig. 3, middle).

While this setup allows for first-class experiences on the students' side, the number of available AR sets significantly restricts the use in larger groups. This situation is somewhat analogous to the use of a blackboard in class: Either the teacher or a single selected student work on the blackboard, while the remainder of the class watches or works along on paper. During a lesson, students take turns at the blackboard. With the aid of an additional computer with video camera and video projection screen, we can mimic this classroom procedure by projecting a live (monoscopic) video of the users (teacher/student) augmented with their current construction on a projection screen next to the users for the remainder of the class to watch (Fig. 2).

Just like in conventional classrooms, students can take turns at using the HMD and working in front of the class. To enhance the classroom situation for students not wearing an AR kit, the overhead projection can also be used to view 3D content attached to markers. By moving a marker in front of the projection surface, its contents are shown on the projection (Fig. 3).

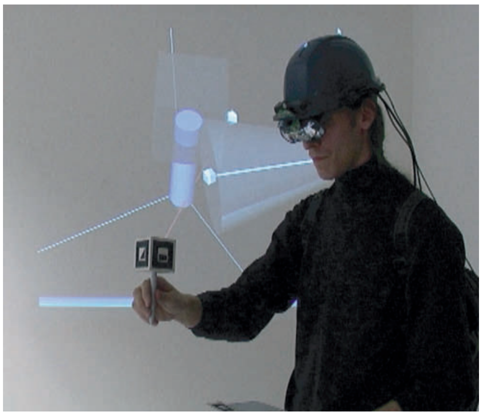


Fig. 2. A teacher is working in Construct3D with the mobile AR setup while a live monoscopic video of his current construction is projected onto a projection screen behind him.

It is intended to be used by high school students and teachers in an interactive, collaborative manner and to blend seamlessly into an everyday classroom situation. Tangible, tool-based interaction provides a simple and intuitive user interface. Support for multiple users and spontaneous collaboration encourages team work and simplifies supervision. The system integrates mobile AR, collaboration, and a tangible user interface.

The *Studierstube* platform on which the Augmented Classroom is built brings together advanced AR features in a unique way. A strong distributed shared scene graph infrastructure enables collaboration between independent mobile AR kits. Dynamic loading and sharing of multi-tasked AR applications between several hosts together with support for tool based interaction allow the users to load and share constructions by handling tangible markers, to print a snapshot of their work or to save it to file. The combination of these features into a single system allows the simple development and setup of a complex application like the Augmented Classroom.

4.2. Projection screen classroom

A popular semi-immersive technique is to use just a large screen projection shared by a group of users (in our case, the class), typically showing stereoscopic images using active or passive stereo glasses. The disadvantage is that since the screen is shared between the active user (e.g., teacher, demonstrator) and the observers, head-tracking is not useful, and consequently stereoscopic images are often severely distorted if rendered for an "averaged" viewpoint. In consequence, manipulation even with tracked input devices becomes indirect (comparable to screen and mouse manipulation) as objects do not appear aligned or superimposed with the users hands. Advantages of this approach include lower system complexity/cost, and the avoidance of cumbersome HMDs. Despite the shortcomings, projection walls are established techniques for semi-immersive group

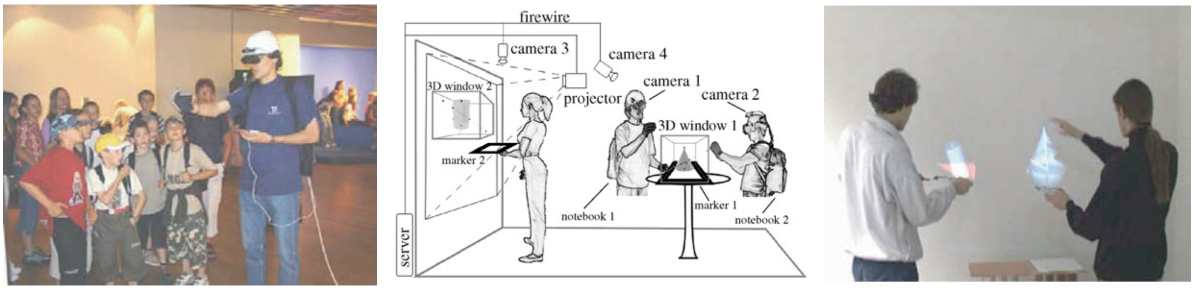


Fig. 3. Left: Demonstration of the mobile AR kit at a local science fair. Middle: Schema of the Augmented Classroom setup. Two mobile users interact with a construction while a third user inspects a finished model on a projection screen. Right: Interacting with models in front of a projection screen.

environments, and single-projector displays are affordable for classroom use.

4.3. Distributed hybrid classroom

Just like the hybrid AR classroom, this setup may use personal HMDs for realizing AR for the teacher and selected students. However, the students are all equipped with personal workstations displaying desktop VR watching the construction process on their screen. We built a desktop VR system using a FireWire camera for optical tracking and a standard consumer graphics card with shutter classes to get stereo rendering with optically tracked 6DOF input devices at a very low price (see Fig. 4). The advantage of this scenario lies in the relatively low price for a personalized semi-immersive display: Students can choose individual viewpoints, maybe even manipulate local copies of the constructed object. However, a teacher can also choose a guided mode, e.g., by locking the students' views to the teacher's viewpoint.

4.4. Remote collaboration

Although the advantages of co-located collaboration are lost, the same systems can be used for remote collaboration through a remotely shared 3D space. For example, a teacher can remotely advise a student at a homework problem by the same guided construction techniques as in the AR-classroom scenario, or multiple students can remotely work together. Each of the users has an individual choice of input and output facility, e.g., one user may wear a HMD, while another one uses a desktop VR setup. We are currently evaluating this possibility using our latest mobile AR hardware prototype [28] as a test platform.

Fig. 4. A user working with our desktop VR system. A FireWire camera (out of view) is used for optical tracking of the hand held props which are equipped with markers (see yellow ellipse). The image of the camera is used as a video background. Stereoscopic images are displayed on the monitor which give the user who wears shutter glasses the impression of working in 3D space. The virtual images of pen and PIP can be seen on the monitor (red ellipse) as an overlay over the video image.

5. Evaluations

The key hypothesis—that actually seeing things in 3D and interacting with them can enhance a student's understanding of 3D geometry—were supported by the anecdotal evidence we have gathered from trial runs with real students. In our first evaluation [5] with 14 students we got very positive and encouraging results and some problems were pointed out. During the evaluation it was gratifying for us to see users work with Construct3D in a very constructive manner. It was obvious that they did not need a long introduction to the system but applied their experience with 2D user interfaces to the 3D interface. The students' interactions with the system in our HMD-based lab setup were interesting to watch. After completing their task, some walked around the objects, viewing them from different sides or got down on their knees and looked at the scene from below. It was clear that they were proud of what they had "built". Half of the students felt that working with Construct3D for the first time is easier than a first experience with a desktop CAD package. All except one could imagine working with Construct3D without having worked with a traditional CAD package before.

Hand-eye coordination is very difficult when spotting a point accurately in 3D space without haptic feedback or constraints. All students reported problems with setting points at given coordinates. As a consequence, we implemented raster and grid functions. About constructing in VR, students especially liked walking around and inside objects, the "playful" way of constructing and that spatial relationships and complex 3D situations are directly visible. The clearness of Construct3D's menu system and the audio help system were mentioned positively.

Students mentioned the following possible application areas: interactive conic sections, vector analysis, enhancing spatial abilities, intersection problems, experiencing space (for very young students) and building 3D worlds from 2D views, elementary geometry, visualization of constructions, geometry didactics—learning by doing and training of spatial abilities by viewing objects from different sides.

At this stage Construct3D is not used by students on a regular basis in mathematics and geometry education but we plan to do extensive evaluations in current and upcoming research projects where students will actually learn by using our application. While developing Construct3D we are regularly visited by teachers, students, colleagues and friends who evaluate the system and give feedback on its quality. This helps to constantly improve the application and adopt it to the students' needs.

6. Conclusions and future work

In this paper, we present a fully functional educational AR application for mathematics and geometry education. We implemented flexible methods to support various teacher–student interaction scenarios. Hybrid hardware setups allow the use of Construct3D in today's classrooms and provide a testbed for these scenarios. Initial evaluations of the concept are encouraging, and our mid- to long-term plans are to integrate it in Austrian high school and higher education curricula. We have established partnerships with Austrian schools and scientific partners such as the Institute of Geometry at Vienna University of Technology and the Institute of Psychology at the University of Vienna.

Much work remains to be done. In particular, a comprehensive evaluation of the practical value of an education tool such as ours will require the development of substantial educational content that is put to real use in classroom. We are currently at the stage where we have working tools available, but now need to apply them to real educational work. For the beginning we plan to create tutorials for vector algebra, conic sections and Boolean operations. We believe that despite the exiting possibilities of the new media, educational content creation for an interactive system is at least as difficult as authoring good textbooks, and will require a substantial amount of time and work. Finally, the true value of the new tool in classroom use needs to be verified through controlled evaluations.

Acknowledgements

The authors would like to thank Gerhard Reitmayr for his ongoing support, Klaus Feiler and Reinhard Steiner for integrating Boolean operations, Florian Ledermann for his work on the Augmented Classroom and all of the Studierstube team. We also thank all participants in our pilot studies for their time. This work is sponsored in part by the Austrian Science Fund FWF under contract number P14470-INF and the EC under project number IST-2001-34204 (Lab@Future).

References

[1] Maier PH. Räumliches Vorstellungsvermögen. Peter Lang GmbH, Europäische Hochschulschriften: Reihe 6, Bd. 493, Frankfurt am Main, 1994.

[2] Gittler G, Glück J. Differential transfer of learning: effects of instruction in descriptive geometry on spatial test performance. Journal of Geometry and Graphics, 1998;2(1):71–84.

[3] Osberg K. Spatial cognition in the virtual environment, Technical R-97-18. Seattle: Human Interface Technology Lab, 1997.

[4] Rizzo AA, Buckwalter JG, Neumann U, Kesselman C, Thiebaux M, Larson P, van Rooyen A. The virtual reality mental rotation spatial skills project. CyberPsychology and Behavior, 1998;1(2):113–20.

[5] Kaufmann H, Schmalstieg D, Wagner M. Construct3D: a virtual reality application for mathematics and geometry education. Education and Information Technologies 2000;5(4):263–76.

[6] Azuma R. A survey of augmented reality. In: PRESENCE: Teleoperators and Virtual Environments, vol. 6(4). Cambridge, MA: MIT press, 1997. p. 355–85.

[7] Mine M. Working in a Virtual World: Interaction Techniques Used in the Chapel Hill Immersive Modeling Program. UNC Chapel Hill Computer Science Technical Report TR96-029, 1996. ftp://ftp.cs.unc.edu/pub/technical-reports/96-029.ps.z.

[8] Mine M. ISAAC: A Meta-CAD System for Virtual Environments. Computer-Aided Design. 1997;29(8): 547–53.

[9] Chapin WL, Lacey TA, Leifer L. DesignSpace: A manual interaction environment of computer aided design. In: Proceedings of the ACM's SIGCHI 1994 Conference: CHI'94 Human Factors In Computing Systems, 1994.

[10] Bowman DA. Conceptual design space—beyond walk-through to immersive design. In: Bertol D, editor. Designing digital space. New York: Wiley, 1996.

[11] Kiyokawa K, Takemura H, Yokoya N. SeamlessDesign for 3D Object Creation. IEEE MultiMedia Magazine, 2000;7(1):22–33.

[12] Bell JT, Fogler HS. The investigation and application of virtual reality as an educational tool. In: Proceedings of the American Society for Engineering Education 1995 Annual Conference, Session number 2513, Anaheim, CA, 1995.

[13] Bricken M, Byrne C. Summer students in virtual reality: a pilot study on educational applications of VR technology. In: Wexelblat A, editor. Virtual reality, applications and explorations. Cambridge, MA: Academic Press Professional, 1993.

[14] Byrne CM. Water on tap: the use of virtual reality as an educational tool. Ph.D. thesis, College of Engineering, University of Washington, 1996, Unpublished.

[15] Winn W. The impact of three-dimensional immersive virtual environments on modern pedagogy. HITL Technical Report R-97-15. Discussion paper for NSF Workshop. Human Interface Technology Laboratory, University of Washington, 1997.

[16] Mantovani F. VR Learning: potential and challenges for the use of 3D environments in education and training. In: Giuseppe R, Carlo G, editors. Towards CyberPsychology: Mind, Cognitions and Society in the Internet Age. Amsterdam: IOS Press, 2001.

[17] Taxén G, Naeve A. CyberMath: Exploring Open Issues in VR-Based Learning. SIGGRAPH 2001 Educators Program. In SIGGRAPH 2001 Conference Abstracts and Applications. 2001. p. 49–51.

[18] Jackiw N. The geometer's sketchpad version 3. Berkeley: Key Curriculum Press, 1995.

[19] Richter-Gebert J, Kortenkamp UH. The Interactive Geometry Software Cinderella: Version 1.2 (Interactive Geometry on Computers), 1999. URL: http://www.cinderella.de

[20] Mechling R. Euklid Dynageo, 2000. URL: http://www.dynageo.com

[21] Laborde JM, Bellemain F. Cabri-Geometry II. Texas Instruments. Copyright Texas Instruments and Université Joseph Fourier, CNRS, 1998. URL: http://www-cabri.imag.fr/index-e.html

[22] Durlach NI, Allen G, Darken R, Garnett RL, Loomis J, Templeman J, von Wiegand TE. Virtual environments and the enhancement of spatial behavior: towards a comprehensive research agenda. In: PRESENCE-Teleoperators and Virtual Environments, vol. 9(6), Cambridge, MA: MIT press, 2000. p. 593–615.

[23] Larson P, Rizzo AA, Buckwalter JG, van Rooyen A, Kratz K, Neumann U, Kesselman C, Thiebaux M, van der Zaag C. Gender Issues in the Use of Virtual Environments. CyberPsychology and Behavior 1999;2(2):113–23.

[24] Azuma R, Baillot Y, Behringer R, Feiner S, Julier S, MacIntyre B. Recent Advances in Augmented Reality. IEEE Computer Graphics & Applications 2001;21(6): 34–47.

[25] Schmalstieg D, Fuhrmann A, Hesina G, Szalavari Z, Encarnação M, Gervautz M, Purgathofer W. The Studierstube Augmented Reality Project. In: PRESENCE—Teleoperators and Virtual Environments, vol. 11(1), Cambridge, MA: MIT Press, 2002. p. 33–54.

[26] OpenCascade 4.0. Open-Source Toolkit for 3D modelling, 2001. URL: http://www.opencascade.com

[27] Szalavári Z, Gervautz M. The personal interaction panel—a two-handed interface for augmented reality. Computer Graphics Forum 1997;16(3):335–46.

[28] Reitmayr G, Schmalstieg D. Mobile Collaborative Augmented Reality. In: Proceedings of the second ACM/IEEE international symposium on augmented reality (ISAR'01), New York, 2001.

Summary of Usability Evaluations of an Educational Augmented Reality Application

Hannes Kaufmann[1] and Andreas Dünser[2]

[1] Interactive Media Systems Group, Vienna University of Technology
Favoritenstrasse 9-11/188/2, A-1040 Vienna, Austria
`kaufmann@ims.tuwien.ac.at`
[2] HIT Lab NZ, University of Canterbury, Private Bag 4800
8140 Christchurch NZ
`andreas.duenser@hitlabnz.org`

Abstract. We summarize three evaluations of an educational augmented reality application for geometry education, which have been conducted in 2000, 2003 and 2005 respectively. Repeated formative evaluations with more than 100 students guided the redesign of the application and its user interface throughout the years. We present and discuss the results regarding usability and simulator sickness providing guidelines on how to design augmented reality applications utilizing head-mounted displays.

Keywords: augmented reality, usability engineering, formative evaluation, geometry education.

1 Introduction

Our work is based on the educational Augmented Reality (AR) application Construct3D [1-3]. This system deploys AR to provide a natural setting for face-to-face collaboration of teachers and students. The main advantage of using AR is that students actually see virtual three dimensional objects. With traditional methods students have to rely on 2D sketching or calculating and constructing objects using pen and paper or CAD software. Direct manipulation and dynamic interaction with virtual 3D objects using tangible interaction devices are key features of Construct3D. In our standard setup users are wearing a see-through head-mounted-display; a pen and a panel are used for direct interaction in 3D space. Head, pen and panel are fully tracked in 3D which allows users to walk around objects and to view them from different perspectives (Fig. 1).

By working directly in 3D space, complex spatial problems and spatial relationships may be comprehended better and faster than with traditional methods. Our system utilizes collaborative AR as a medium for teaching, and uses 3D dynamic geometry to facilitate mathematics and geometry education.

Over the course of 6 years Construct3D has been developed, improved, tested and evaluated with more than 100 students in over 500 teaching lessons. Pedagogical theories such as constructivism and activity theory influenced the design of the

R. Shumaker (Ed.): Virtual Reality, HCII 2007, LNCS 4563, pp. 660–669, 2007.

collaborative educational AR hardware setup and content design. Technical details and pedagogical uses of Construct3D (including teaching content) have been published by the first author before [2-4].

Fig. 1. Students working with Construct3D

The development process of Construct3D resembles the usability engineering methods of virtual environments suggested by [5]. The first informal evaluation in 2000 helped to compile a detailed user task analysis whereas expert guideline-based evaluations occurred numerous times during the development process. Visiting teachers and researchers evaluated the system and provided useful feedback. Two formative evaluations in 2003 and 2005 had a big impact on the design and development of Construct3D. In this paper we summarize three usability evaluations conducted in 2000, 2003 and 2005 and will present the lessons learned.

2 Construct3D

Construct3D is based on the Studierstube AR system [6]. It promotes and supports exploratory behavior through dynamic 3D geometry. A fundamental property of dynamic geometry software is that dynamic behavior of a construction can be explored in real time by interactively moving individual defining elements such as corner points of a rigid body. Users can see which parts of a construction change and which remain the same. The histories of constructions as well as dependencies between geometric objects are maintained. Experiencing what happens under movement facilitates better comprehension of a particular construction and geometry in general.

The menu system is mapped to a hand-held tracked panel called the personal interaction panel (PIP) [7]. The PIP (Fig. 2) allows the straightforward integration of conventional 2D interface elements like buttons, sliders, dials etc. as well as novel 3D interaction widgets. Passive haptic feedback from the physical props guides the user when interacting with the PIP, while the overlaid graphics allows the props to be used as multi-functional tools. Students can for instance position

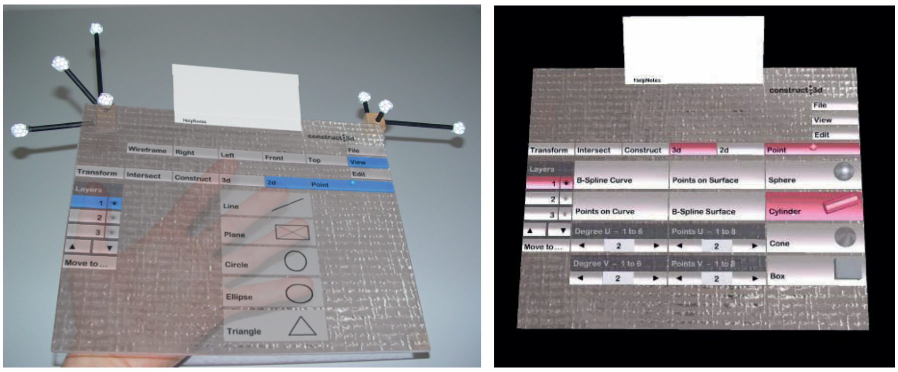


Fig. 2. Left: Menu system of Construct3D displayed on the PIP. In a help-box (on top) further details and help on application features are provided. Right: 3D submenu displayed for the user working with the red color scheme.

written notes onto the tablet which might help them during their work in the virtual environment.

All construction steps are carried out via direct manipulation in 3D using a stylus tracked with six degrees of freedom. In order to generate a new point the user clicks with his pen exactly at the location in 3D space where the point should appear. Users can easily switch between point mode (for setting new points) and selection mode (for selecting 3D objects).

Desktop CAD systems typically have a very steep learning curve and offer an abundance of features in deeply nested menus. For Construct3D we focused on a simpler menu system, which is easy to learn and intuitive to use. In addition we accommodated to the fact that menu widgets seen through a HMD need a certain size in order to be usable. Organizing the functions proved difficult under these conditions as the number of program functions increased over time. We finally organized the menu – according to a user task analysis, experts' guidelines and experience by logic grouping of functionality – into five submenus accessible via tabs (Fig. 2), with frequently used functions being visible all the time. This provides relatively quick access to all program functions. The menu concept is similar to that used in traditional desktop CAD menu systems known by many students, while avoiding excessive interface modes.

Hardware Setups. The standard immersive setup used for Construct3D supports two collaborating users wearing stereoscopic see-through head mounted displays (HMDs) (see Fig. 1) providing a shared virtual space. The users interact with the system using pen and pad props (Fig. 2). Both users see the same virtual objects as well as each others' pens and menu systems which provides a global shared space. In addition it allows users to help each other (i.e. with the menu system) if necessary. Position and orientation of head and hands are tracked using a 4-camera infrared-optical tracking system. In a co-located setup - such as the one used for our evaluations - one dedicated host with two graphic ports renders stereoscopic views for both users.

3 Usability Studies

We report and compare a first informal user study and formative usability studies completed in 2003 and 2005. Based on feedback from many trials with high school students and a first informal evaluation in 2000 [8] we continuously improved Construct3D over a course of 5 years.

All usability enhancements were conducted with the intention of improving collaborative learning and teaching. As usability can only be improved in accordance with users' needs and application specific strengths and weaknesses, the guidelines mentioned here cannot be applied directly to other applications without careful adaptation.

3.1 1st Informal Evaluation – 2000

In our first evaluation [8] with 14 students we observed the students' interaction with the system. We obtained very positive and encouraging feedback and a number of problems were pointed out. During the evaluation it was gratifying for us to see users work with Construct3D in a very constructive manner. They did not need a long introduction to the system but applied their experience with 2D user interfaces to the 3D interface. After completing the task, some walked around the objects, viewing them from different sides or got down on their knees and looked at the scene from below. Half of the students felt that working with Construct3D for the first time was easier than their first experience with a desktop CAD package.

Hand-eye coordination showed to be very difficult when spotting a point accurately in 3D space without haptic feedback or constraints. All students reported problems with setting points at given coordinates. As a consequence we implemented raster and grid functions. About constructing in VR, students especially liked walking around and inside objects, the "playful" way of constructing, and that spatial relationships and complex three dimensional designs are directly visible. The clear structure of Construct3D's menu system and the audio help system were mentioned positively.

At that time Construct3D was still a static modeling tool and did not provide dynamic features. Insights gained from the first evaluation (i.e. the difficulty for highly accurate 3D interaction) and the understanding that students would educationally benefit from 3D dynamic geometry encouraged us to change Construct3D into a dynamic 3D geometry application.

3.2 2nd Evaluation Study - 2003

In 2003 we conducted a study based on interviews and the standardized ISONORM 9241/10 usability questionnaire [9]. We designed a number of training exercises that fit the Austrian descriptive geometry curriculum of 11th and 12th grade [4]. Using Construct3D, 15 high school students (9 male, 6 female) worked on these exercises with the aid of their teachers. All students attended geometry classes (descriptive geometry) since the beginning of grade 11. Each of them participated in 5 training sessions lasting 6 hours. Our main objective was to assess the usability of our system and its potential as an educational tool for real high school work. At the end of all training sessions students had to answer an ISONORM usability questionnaire. Two questions regarding self-descriptiveness of the application had to be removed since they were related to desktop applications only. Afterwards students answered general questions regarding user acceptance, user behavior, technical requirements and organizational aspects.

Results. A closer look at the data (Figure 2) reveals that the categories "suitability for learning" and "suitability for task" received the highest rating which is very important in this context. In our opinion the highest priorities for an educational application that complies with pedagogic theories such as constructivism are that it (1) is easy to use and requires little time to learn, (2) encourages learners to try new functions and (3) can be used consistently and is designed in a way that things you learned once are memorized well. These are exactly the items that students rated very high. Almost all students reported that they could imagine using the current version of Construct3D in high school or university education.

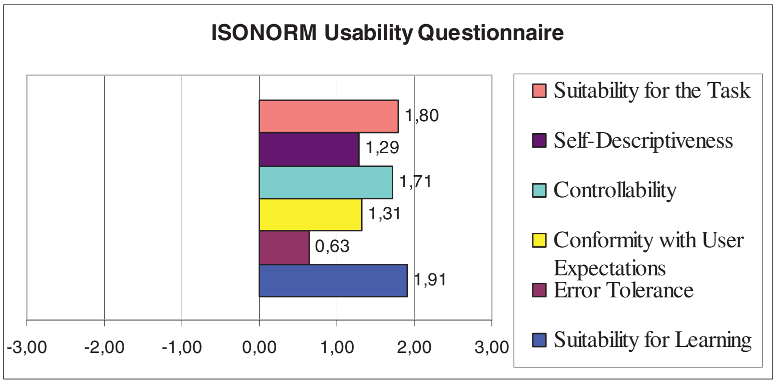


Fig. 3. Results of the ISONORM [9] usability questionnaire in 6 categories

The categories "self-descriptiveness" and "conformity with user expectations" got lower ratings than the rest. Self-descriptiveness of Construct3D was improved by adding better labeling and a help-box on the panel in order to explain all menu items.

As a result of this usability study the user interface was completely redesigned. The menu system was restructured (Fig. 2) to make features that are used most frequently easily accessible. In addition the visual design of geometric objects was enhanced considerably. The purpose of visual design of objects constructed by the user is to support the user's understanding of a construction. Unlike desktop visualization of the same content, using stereoscopic see-through HMDs requires to deal with limited contrast, resolution and viewing angle. Moreover, the system should present scenes of high depth complexity in a clear way, providing an improved insight into the construction. Among the techniques employed in Construct3D to support these goals are the use of transparencies for geometric objects to allow students to see inside objects (Fig. 1), consistent color coding to allow distinguishing between multiple users' contributions (which is especially important in distributed remote teaching scenarios), separation into layers to support semantic structuring of a construction, and automatic previewing of new objects. Details of the improvements are given in [3].

3.3 3rd Evaluation Study - 2005

In the 2005 evaluation 47 students were solving tasks with Construct3D in AR while another group of 44 students solved the same geometric problems with an educational desktop application called CAD3D [10] (which is used in Austrian high schools). Participants were Austrian high school students aged between 16 to 19 years (M = 17.49, SD = .79; 44 (48.4%) male and 47 (51.6%) female). Students attended 6 training sessions which lasted 45 minutes with one week pause in between. In both groups a tutor supervised two students working on the geometry tasks. The tutors explained the tasks to the students and supported them if they needed help.

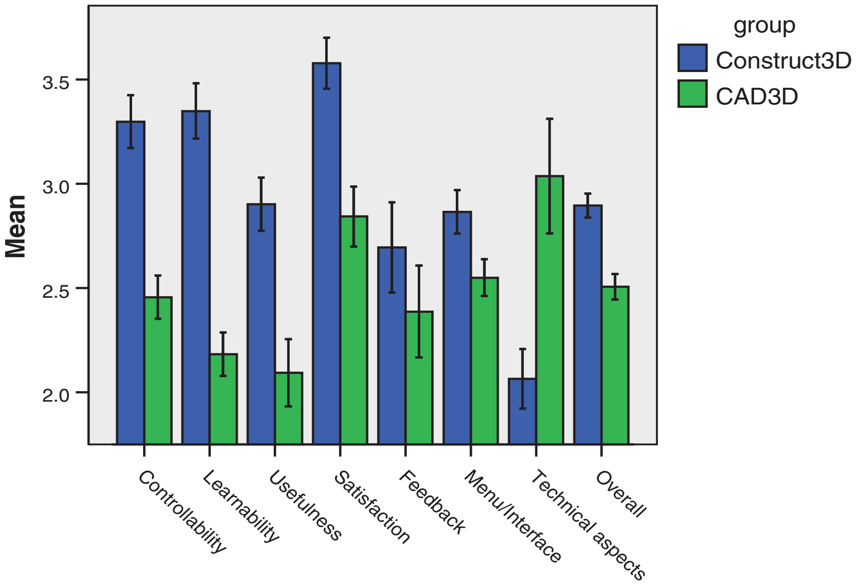


Fig. 4. Usability ratings of students working with Construct3D and CAD3D (4-point Likert scale; 1-min, 4-max = best; error bars ± 1.96 * standard error)

To assess usability we adapted questions of 8 established usability questionnaires to develop a questionnaire (7 scales (see Fig. 4); 28 questions in total) better suited for the range of applications tested. The questions were taken from the Questionnaire for User Interface Satisfaction, Perceived Usefulness and Ease of Use, Purdue Usability Testing Questionnaire, Computer System Usability Questionnaire, Practical Heuristics for Usability Evaluation (all at [11]), Software Usability Measurement Inventory [12], SUS [13] and the ISONORM [9] usability questionnaire.

Results. The analysis of the usability questionnaire showed that students using Construct3D gave higher ratings ($p < .01$) for all categories (Controllability, Learnability, Usefulness, Satisfaction, Feedback, and Menu/Interface) except technical aspects (e.g. robustness) than students using CAD3D. This indicates that the AR based geometry education application Construct3D is a highly usable system which - from a usability perspective - has several advantages over the traditional desktop based application. Especially user satisfaction, learnability and controllability got high ratings. However the low ratings for technical aspects suggest that there are still some issues regarding technical robustness that have to be addressed. Infrequent system crashes and minor technical problems can reduce motivation of participants and usability of the system.

Comparing the results of the 2003 and 2005 evaluations illustrates that conformity with user expectations (2003) / satisfaction (2005) was improved throughout the years. Suitability for the task got quite high ratings in the 2003 evaluation. In 2005 students rated usefulness, the equivalent scale, somewhat lower. In the 2005 evaluation a more extensive training setup was realized and thus students worked on a broader variety of geometric problems (e.g. problems used in standard school curriculum). Hence, this result may indicate for which kind of geometric problems Construct3D is a suitable educational tool. In both formative evaluations its strengths became obvious. Construct3D should mainly be used for teaching content which utilizes 3D dynamic geometry or requires the visualization of abstract problems. In addition these are areas that are hardly covered by other educational applications.

We also asked the students other questions in order to get more detailed feedback on the training task and setup. Analyzing the students' answers to these questions may help to refine our system setup and further adapt it to users needs. Table 1 shows the preferred training setup of students using Construct3D and CAD3D.

Table 1. How would you prefer to work with Construct3D / CAD3D

	Construct3D	*CAD3D*
2 students, one tutor (like in the training sessions)	80.95%	86.00%
1 student, one tutor	9.52%	4.65%
2 students, without tutor	4.76%	2.33%
alone	4.76%	4.65%

There were no significant differences regarding the preferred training setup between students working with Construct3D and CAD3D. Most of the students liked the setup we used for the trainings in our study: 2 students working with one tutor.

Regarding the potential use of Construct3D in educational institutions we asked the students if they would like to use Construct3D in school in a setting similar they worked with (1 to 2 students) given the technical equipment would be affordable for schools. The majority of students would like to use Construct3D in school (yes = 64.44%, rather yes = 26.67%); 8.89% would rather not like to use the system in schools. Students' comments on the potential problems of using Construct3D in schools were mainly concerned with lack of finances and the robustness of hardware and software.

4 Simulator Sickness

As described earlier, Construct3D requires users to wear a HMD. In the second evaluation study (2003) some of the students reported negative side effects after working in the virtual environment, a condition known as simulator sickness, which is similar to motion sickness [14]. One female student reported headache and eye strain after 20 minutes of work in the virtual environment but did not stop working and wanted to use Construct3D again. In retrospect we know that our evaluation sessions lasting one hour were simply too long for continuous work with a HMD. Since negative side effects are a general potential problem when working with HMDs and influence the user's subjective experience of a VR/AR environment considerably they are relevant to all VR/AR applications that use these displays. We identified some possible reasons of such negative side effects that may be relevant to our virtual environment such as accommodation problems, low frame rate, lag or bad fitting helmets. If not taken into account, symptoms experienced by users affected by simulator sickness can drastically diminish usability of a system [15].

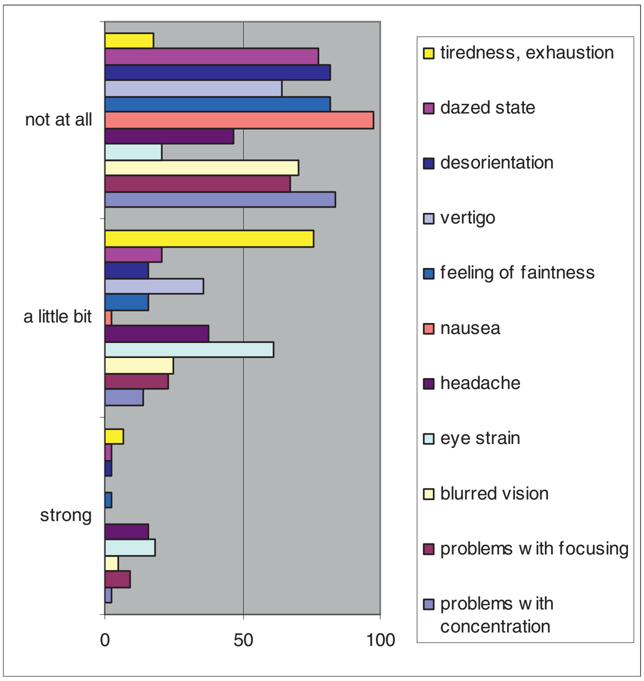


Fig. 5. Percentage of users reporting a specific symptom is shown (0% = reported by no user; 100% = reported by all users)

In order to minimize the chance of users suffering from symptoms of simulator sickness we limited training sessions to a maximum of 45 minutes in our third evaluation study (2005). Furthermore we replaced the hard plastic helmet (Fig. 1, left) which caused pressure on some students' forehead or even headache with a relatively lightweight bicycle crash helmet (Fig. 1, right). Students also were asked to take a rest when they felt the need to. After they finished the training sessions with Construct3D we asked them to which extent they actually did experience specific symptoms related to simulator sickness (questionnaire; 11 questions). Fig. 5 shows the percentage of participants having experienced a specific symptom 'not at all', 'a little bit' or 'strong' during or while having worked with Construct3D.

75.56% of the 47 participants felt a moderate amount of tiredness or exhaustion and 61.36% reported a little bit of eye strain. There were also some participants who reported having experienced some headache (37.78%) and vertigo (35.56%). Most of these symptoms may be related to the use of a HMD. Thus although we limited training time there still seem to persist issues with respect to some simulator sickness symptoms, especially exhaustion and eye strain. However in general most of the participants did not report having experienced severe problems.

In accordance with our observations and other studies we recommend limiting HMD usage to 20-30 minutes per session. Based on our experience image quality of HMDs but especially lag and quality of tracking data contribute most to the reported effects.

5 Conclusion and Future Work

In this summary of usability evaluations we describe how we managed to improve usability of Construct3D iteratively. We gradually adapted, reconfigured and redesigned hard- and software and integrated new interaction techniques and interfaces according to our observations and user feedback. A number of studies report that cognitive overhead in mastering the interface can hinder training and learning of the task [16]. Especially in educational applications it is of utmost importance to focus students' attention on the actual task and to reduce cognitive overhead needed to use the application. This motivated us to put a lot of time and effort into interaction and user interface design. We gained valuable results from the evaluations which helped us to create a more usable AR-based learning environment with improved user satisfaction.

In our latest evaluation we found that the usability of Construct3D was rated higher than the usability of a desktop based geometry education application. This may be due to the more intuitive workflow when working on 3D tasks. However there are still technical issues (e.g. software robustness) that have to be solved in order to improve usability even further. Especially problems related to the use of HMDs and tracking latency need careful thought. Thus at this stage we recommend to limit usage times of head mounted displays in immersive training setups. For an educational application such as Construct3D we envision its integration into courses; therefore temporally limited usage is very reasonable in this context.

Developers of AR-based applications face specific hard- and software related issues that are different from those of desktop based GUI or WIMP design. No set of common design guidelines exist yet that would facilitate or streamline the development of easy to use AR systems [15].

Regarding future work we plan to use Construct3D as a tool for evaluating various aspects of virtual learning environments in our future research including a comprehensive pedagogic evaluation, studying e.g. teaching styles/methodology or transfer of learning to tasks in the real world.

References

1. Kaufmann, H., Schmalstieg, D.: Mathematics and geometry education with collaborative augmented reality. Computers & Graphics 27, 339–345 (2003)
2. Kaufmann, H.: Geometry Education with Augmented Reality. Ph.D. Thesis. Vienna University of Technology (2004) http://www.ims.tuwien.ac.at/research/construct3d/videos.php
3. Kaufmann, H., Schmalstieg, D.: Designing Immersive Virtual Reality for Geometry Education. In: Proceedings of IEEE Virtual Reality Conference 2006, Alexandria, Virginia, USA 2006, pp. 51–58 (2006)
4. Kaufmann, H., Papp, M.: Learning Objects for Education with Augmented Reality. In: Proceedings of EDEN 2006 (European Distance and E-Learning Network) Conference, Vienna 2006, pp. 160–165 (2006)
5. Hix, D., Gabbard, J.L.: Usability Engineering of Virtual Environments. In: Stanney, K.M. (ed.) Handbook of Virtual Environments - Design, Implementation, and Applications, pp. 681–699. Lawrence Erlbaum Associates, Mahwah, New Jersey (2002)
6. Schmalstieg, D., Fuhrmann, A., Hesina, G., Szalavári, Z.S., Encarnacao, L.M., Gervautz, M., Purgathofer, W.: The Studierstube augmented reality project. Presence - Teleoperators and Virtual Environments 11, 33–54 (2002)
7. Szalavári, Z.S., Gervautz, M.: The Personal Interaction Panel - A Two-Handed Interface for Augmented Reality. Computer Graphics Forum 16, 335–346 (1997)
8. Kaufmann, H., Schmalstieg, D., Wagner, M.: Construct3D: a virtual reality application for mathematics and geometry education. Education and Information Technologies 5, 263–276 (2000)
9. Prümper, J.: Der Benutzungsfragebogen ISONORM 9241/10: Ergebnisse Zur Reliabilität und Validität. In: Liskowsky, R. (ed.): Software-Ergonomie ’97, Stuttgart (1997)
10. Stachel, H., Wallner, J., Pfeifer, M.: CAD-3D für Windows. (2003) http://www.geometrie.tuwien.ac.at/software/cad3d/
11. Perlman, G.: Web-Based User Interface Evaluation with Questionnaires. (accessed Feb. 15th 2006) http://www.acm.org/~perlman/question.html
12. Kirakowski, J., Corbett, M.: SUMI: the Software Usability Measurement Inventory. British Journal of Educational Technology 24, 210–212 (1993)
13. Brooke, J.: SUS: a quick and dirty usability scale. In: Jordan, P.W., Thomas, B., Weerdmeester, B.A., McClelland, I. (eds.): Usability Evaluation in Industry. Taylor and Francis, London, pp. 189–194 (1996)
14. LaViola Jr., J.J.: A discussion of cybersickness in virtual environments. ACM SIGCHI Bulletin 32, 47–56 (2000)
15. Dünser, A., Grasset, R., Seichter, H., Billinghurst, M.: Applying HCI principles to AR systems design. In: MRUI’07: 2nd International Workshop at the IEEE Virtual Reality 2007 Conference, Charlotte, North Carolina, USA (2007)
16. Dede, C., Salzman, M.C., Loftin, R.B.: ScienceSpace: Virtual Realities for Learning Complex and Abstract Scientific Concepts. In: Proceedings of IEEE VRAIS ’96, pp. 246–252 (1996)

Journal for Geometry and Graphics
Volume 13 (2009), No. 1, 1–13.

Dynamic Differential Geometry in Education

Hannes Kaufmann

Institute of Software Technology and Interactive Systems, Vienna University of Technology
Favoritenstr. 9-11/118/2, A-1040 Wien, Austria
email: kaufmann@ims.tuwien.ac.at

Abstract. We present an augmented reality application which introduces differential geometry in educational dynamic geometry software. New functionality such as a Frenet frame, center and circle of curvature in arbitrary curve points, and others were implemented. Dynamic geometry allows to study differential geometric properties under movement. Using this tool we developed examples which enable teachers and learners to intuitively explore properties of interesting curves, to visualize contact of higher order between curves and surfaces, to construct Meusnier's sphere, Dupin's indicatrix and more.

Key Words: Differential geometry, dynamic three-dimensional geometry, augmented reality, geometry education.

1. Introduction

Concepts and findings that originate from differential geometry are applicable in a many areas such as physics [6], economics, computer graphics (e.g. [5]), engineering in general and geology. To understand the potential of differential geometry and its areas of application the topic is taught as part of many higher education curriculums of technical studies worldwide. In this work we present a dynamic geometry software application to aid teaching the basics of differential geometry of space curves in a very visual and interactive way (Figure 1).

Therefore an augmented reality (AR) [1] application for dynamic geometry education has been extended to support operations such as the creation of Frenet frames in points on curves, the plane, center and circle of curvature and others. This software tool is designed to support teaching and learning of basic geometric principles and properties of different types of curves and surfaces. The main advantage of using AR is that students actually see three-dimensional objects which they until now had to calculate and construct with traditional methods. Due to evaluations and observations [12] we hypothesize that by working directly in 3D space, complex spatial problems and spatial relationships can be comprehended better and faster than with traditional methods.

After briefly summarizing related work in the area of dynamic geometry software and differential geometry we present our software tool Construct3D and its extension together with practical, intuitive examples for teaching differential geometry in higher education. They

ISSN 1433-8157/$ 2.50 © 2009 Heldermann Verlag, Berlin

Figure 1: A student working with Construct3D in our standard AR lab setup with a head mounted display.

demonstrate which flexibility and potential three-dimensional dynamic geometry holds in teaching differential geometry.

2. Related Work

In Austrian schools the use of commercial 3D computer-aided design (CAD) software such as AutoCAD, Pro/ENGINEER, MicroStation, CATIA and others is wide spread in modern geometry education for teaching principles of 3D modeling. In addition there are excellent educational 3D modeling programs such as CAD3D [18] or GAM [20] (developed by Austrian geometers specifically for students) which are frequently used.

In addition to classical educational CAD tools such as CAD3D and GAM a new category of educational geometry software emerged in recent years.

2.1. Dynamic 2D Geometry Software

Since a computer can record the way we construct geometric objects the software is able to quickly redo constructions after changing some parameters. A fundamental property of dynamic geometry software is that the dynamic behavior of a construction can be explored by interactively moving individual defining elements such as control points of a Bézier curve: pick a point, move it and see immediately how the construction changes. This dragging capability is a fundamental improvement compared to drawings on paper or static CAD models.

Comprehensive work on dynamic geometry was done by Kortenkamp in "Foundations of Dynamic Geometry" [15]. The first software packages for dynamic geometry were Geometer's Sketchpad [10], which appeared first in 1989, and Cabri Geometry [16], dating back to 1988. Since then dynamic geometry software has spread in education. Today, there are more than 40 packages for dynamic geometry. The most popular ones are Cinderella [21], Euklid [17], Geometer's Sketchpad or Cabri Geometry. All of them support two-dimensional geometry only.

2.2. Dynamic 3D Geometry Software

In late 2004 the first three-dimensional dynamic geometry desktop application Cabri 3D was presented [4]. The current version supports basic 2D and 3D objects and the intersection of lines and planes with these objects but lacks support for general intersection curves between surfaces, Boolean operations and more complex geometric primitives such as surfaces of revolution which are present in Construct3D. Lenghts, angles, areas and volumes can be measured and further calculations can be performed with these results. Animations can be used for modeling physical phenomena. A tool replays the user's previously performed construction steps. Unfolding of all polyhedra into a printable net is supported as well.

Archimedes Geo3D [7] is a cross-platform 3D dynamic geometry application which is under development by Andreas Goebel since 2006. Similar to 2D dynamic geometry software Archimedes Geo3D supports the creation of loci which are traces of points, i.e. curves. In addition it is also possible to create traces of curves in 3D, which are surfaces. Points and basic shapes can be used as input but curves and surfaces can also be defined using mathematical parametrizations. Further available features are texturing, animation creation and shadow generation. Macros can be used to record and replay multiple construction steps and can also be called recursively. Archimedes Geo3D supports stereoscopic output either by anaglyph images or by using shutter glasses [7].

2.3. Professional CAD Software

Similarities exist between variational or parametric CAD modelling and dynamic geometry software. In general small changes of parameters in a CAD construction do not cause stringent topological changes in the construction. This can be used for instance to customize a single prototype construction quickly or in case of data compression for storage of a large number of similar objects. The problems that occur in parametric CAD [8, 9] are similar to those of dynamic geometry. Parts of these problems are discussed and solved in [15].

Only few commercial 3D CAD software packages provide differential geometry functionality and are therefore related to this work. Rhino3D (www.rhino3d.com) and Pro/ENGINEER are two such examples. Professional CAD packages are usually not interactive in a sense that changes are applied in real time in comparison to dynamic geometry software which always provides immediate feedback to the learner. Because of their fields of application they are not necessarily optimized to deliver real time results. CAD modelling tools fulfill strigent accuracy requirements and are typically used for models of higher complexity compared to those used in education when learning about surface properties.

Rhino3D provides features for the analysis of curves on surfaces in order to visualize Gaussian curvature, mean curvature, and the minimum or maximum radius of curvature. Pro/Engineer and other CAD packages offer similar curve and surface analyzing tools. None of the above presented tools allows to study differential geometric properties of curves and surfaces in a real-time dynamic - in the sense of dynamic 3D geometry - way.

In the following we present Construct3D which is the first 3D dynamic geometry application that provides functions to explore curves and surfaces using dynamic differential geometry. In addition we demonstrate through a series of educational examples which 'added value' dynamic geometry provides to teaching differential geometry and how it enhances understanding of fundamental geometric knowledge.

3. Construct3D

Construct3D [13, 11] is a three-dimensional dynamic geometry construction tool which has been designed for educational use. Three usability studies with more than 100 students have been conducted since 2000 [12] and guidelines have been formulated regarding how to design

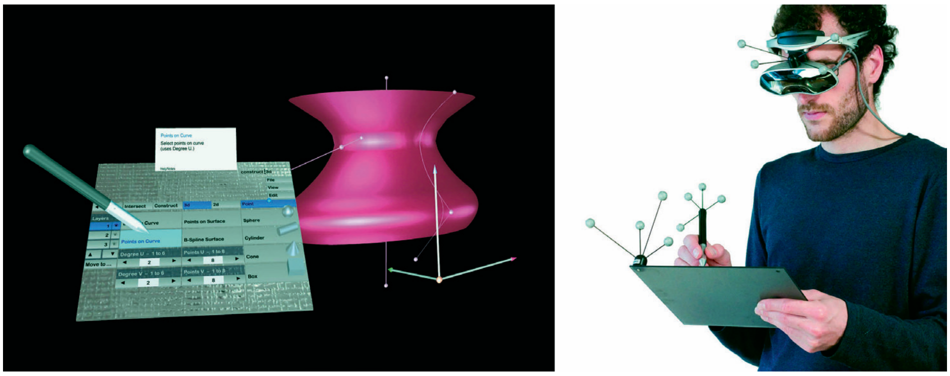


Figure 2: Right: A student working with Construct3D holds a wireless pen and a panel which are optically tracked using retro-reflective markers. Left: The current view of the student in the head-mounted display. The menu system on the panel is visible.

AR applications for (geometry) education [14]. A collaborative augmented reality (AR) setup is utilized with the main advantage that students actually see three dimensional objects in 3D. The setup supports two collaborating users wearing stereoscopic see-through head mounted displays (HMDs) (Sony Glasstron D100BE) providing a common, shared virtual construction space. One PC with two graphic ports renders stereoscopic views for both users. Head and hands are tracked with millimeter accuracy using an iotracker [19] optical tracking system. This allows students to walk around objects and to view them from different perspectives.

Construct3D's menu system is mapped to a hand-held pen and panel interface, the Personal Interaction Panel (PIP) [23] (Figure 2). The pen is used for operating the menu on the panel as well as for direct manipulation of the scene in 3D space. Augmented reality is used so that both users see the same virtual objects as well as each others' pens and menus, therefore a user can provide help to another user if desired. The face-to-face setting allows for traditional pedagogic communication between teacher and students. Other setups for educational use have been reported in [11].

Construct3D is based on the Studierstube software platform [22] as a runtime environment and for multi-user synchronization. The current version of Construct3D offers functions for the construction of 3D points and geometric objects. It provides planar and spatial geometric operations on objects, measurements, and structuring of elements into '3D layers'. It supports generation of and operation on these object types: Points (either freely positioned in space or fixed on curves and surfaces), lines, planes, circles, ellipses, cuboids, spheres, cylinders, cones, B-Splines curves, NURBS surfaces up to 8x8 control points and variable degree, and surfaces of revolution. To mention just a few, the following geometric operations are implemented: Boolean operations (union, difference, intersection) on 3D objects, intersections between all types of 2D and 3D objects resulting in intersection points and curves as first class objects, planar slicing of objects, rotational sweeps, helical sweeps, general sweeps along a path, surface normals, tangential planes, tangents and many more. The system features support for 3D dynamic geometry. All points can be picked and dragged at any given time. Experiencing what happens under movement allows better insight into a particular construction and geometry in general.

A comprehensive overview of Construct3D is given in [13, 11].

3.1. Geometry Kernel

Construct3D utilizes the ACIS geometry kernel for a wide range of calculations. The 3D ACIS Modeler [3] is Spatial's 3D modeling development technology used by developers worldwide, in industries such as CAD/CAM/CAE, AEC, animation, and shipbuilding. It is the geometry kernel of Autocad and many other well known CAD applications. ACIS is under development for more than 15 years and features an object-oriented C++ architecture that enables robust, 3D modeling capabilities. It integrates wireframe, surface, and solid modeling functionality with both manifold and non-manifold topology, and a rich set of geometric operations.

In Construct3D the ACIS geometry kernel has been integrated especially for calculating Boolean operations, intersections, tangents and tangential planes, sweep and helical surfaces as well as NURBS and B-Spline surfaces. ACIS uses mathematical boundary representations internally and provides methods to calculate derivatives of arbitrary order in a given point on curves and surfaces (as long as they are differentiable). The ACIS documentation states that "a certain number of derivatives are evaluated directly and accurately; higher derivatives are automatically calculated by finite differencing. The accuracy of these decreases with the order of the derivative, as the cost increases." This functionality allowed the straightforward extension of Construct3D to visualize basics of differential geometry in three-dimensional space.

3.2. Differential Geometry Functions

New features were implemented in Construct3D to support the creation of Frenet frames in points of curves, the plane, center and circle of curvature and the osculating sphere (sphere of curvature).

3.2.1. Frenet Frame

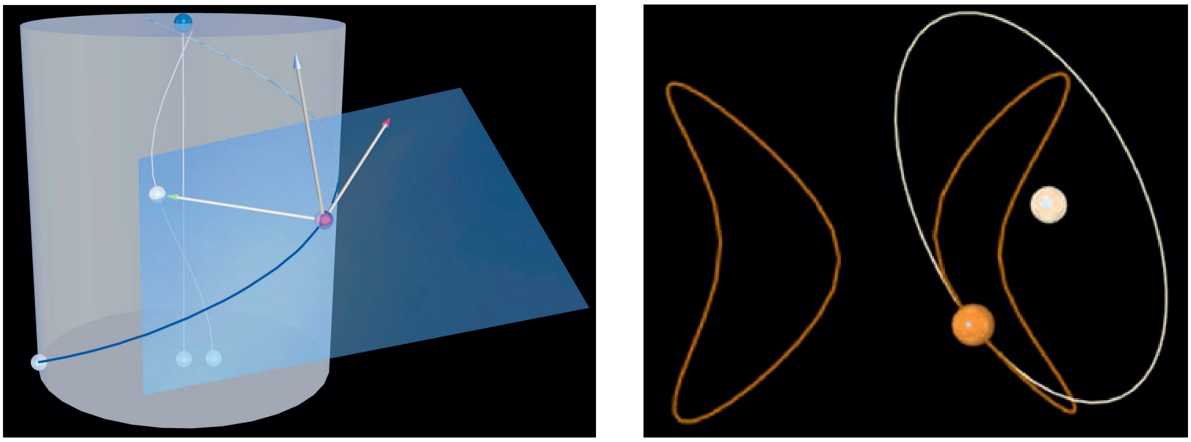
The Frenet frame or Frenet trihedron is a reference frame, a rectilinear coordinate system attached to a point of a space curve consisting of the tangent $\mathfrak{t}$, normal $\mathfrak{n}$, and the binormal vector $\mathfrak{b}$ which are defined as

$$\begin{array}{rcl} \mathfrak{t} & = & \mathfrak{x}'(s) \\ \mathfrak{n} & = & \dfrac{\mathfrak{x}''(s)}{|\mathfrak{x}''(s)|} \\ \mathfrak{b} & = & \mathfrak{t} \times \mathfrak{n} \end{array}$$

with a non-degenerate curve $\mathfrak{x}$, parametrized by its arclength s. In Construct3D the Frenet frame can be attached to as many points of a curve as desired. During movement of such points along the curve (or while changing the curve itself e.g. by moving control points of a B-spline curve) the trihedron travels along which allows to study tangent, normal and binormal of the curve in any point at any time (Figure 3(a)).

3.2.2. Plane, Center and Circle of Curvature

The plane of curvature, or osculating plane, in a curve point contains the circle of curvature of a space curve and is spanned by the normal vector $\mathfrak{n}$ and the tangent $\mathfrak{t}$ (Figure 3(a)). The circle of curvature osculates the curve. Its midpoint $\mathfrak{M}$ - the center of curvature - lies in direction of the normal vector of the curve. The distance between center of curvature and



(a) Frenet frame in a point P of a helix together with the plane of curvature in P. The center of curvature moves along a helix as well (section 4.1).

(b) Center and the corresponding circle of curvature (white) in a point on the intersection curve between two cylinders.

Figure 3: Frenet frame, plane, center and circle of curvature in Construct3D.

curve point is the radius r of the circle of curvature. In case of arc length parametrization of the curve $\mathfrak{x}$ the corresponding curvature is computed by $\kappa(s) = |\mathfrak{x}''(s)|$. With the help of the Frenet formulas the position of the center of curvature can be derived $\mathfrak{M}(s) = \mathfrak{x}(s) + \frac{1}{\kappa}\mathfrak{n}(s)$. The radius of curvature is therefore inversely proportional to the curvature $\rho(s) = \frac{1}{\kappa(s)}$.

3.2.3. Osculating Sphere

An osculating sphere, or sphere of curvature has contact of at least third order with a curve $\mathfrak{x}$. The osculating sphere in P can also be defined as the limit of a variable sphere passing through four points of $\mathfrak{x}$ as these points approach P - a property that is used in example 4.2.

The center $\mathfrak{M}$ of any sphere which has contact of (at least) second order with $\mathfrak{x}$ at point P, where the curvature $\kappa > 0$, lies on the axis of curvature (also called polar axis) which is parallel to the binormal passing through the center of curvature corresponding to P. The torsion of a curve point can be regarded a measure of the rotation of the corresponding plane of curvature around the tangent. The osculating sphere has center

$$\mathfrak{M}(s) = \mathfrak{x}(s) + \rho(s)\mathfrak{n}(s) + \frac{\rho'(s)}{\tau(s)}\mathfrak{b}(s).$$

All respective derivatives are calculated by the ACIS kernel in real time whenever the position of a curve point P changes in order to i.e. update the center of the osculating sphere while moving P.

4. Teaching Contents for Dynamic 3D Differential Geometry

To demonstrate Construct3D's potential in dynamic differential geometry we present teaching contents. Previous evaluation studies identified the main strengths of Construct3D as an augmented reality teaching aid: The biggest advantages compared to traditional software tools are obvious if using Construct3D for teaching content which utilizes three-dimensional dynamic geometry and requires the visualization of abstract problems. We noticed that students need to be challenged to use dynamic functionality. Otherwise some of them are satisfied with constructing static models and do not intend to explore on their own. Therefore examples are introduced which require to study constructions under movement to foster active exploration. Our approach of active, explorative learning is in accordance with pedagogic theories such as activity theory and constructivism.

The examples range in difficulty from higher grade high school to basic university mathematics and geometry education. For each example we provide brief background knowledge as a quick summary of the topic and highlight properties which are most relevant and most interesting in regard to dynamic geometry.

4.1. Tangent, Normal, Binormal

A good starting point is to study Frenet frames in various curve points. We construct a helix and display the corresponding cylinder that contains the helix. When moving a Frenet frame along a helix (Figure 3(a)) diverse curve properties can be studied. Students will soon notice that the slope of the tangent does not change when moving the point along the curve. A plane normal to the axis of the helix through the point, or a generator line (which moves with the point) helps to realize the constant slope of the tangent quickly.

The curvature and the torsion of a helix are constant. Conversely, any space curve with constant non-zero curvature and constant torsion is a helix. Constant curvature can again be observed: Since the curvature is constant the center of curvature moves on an offset curve to the original helix with the same axis which is a helix itself. This can easily be seen in dynamic geometry by moving a point along the helix and studying its center of curvature during movement (Figure 3(a)). Equivalentely the centres of the osculating spheres of a helix are on a helix which can be visualized with Construct3D as well.

In this context it might also be reasonable to discuss the geodesic property of a helix on a cylinder as well as the helix as a loxodrome of the cylinder and as a line of constant slope. This is just one example of how Frenet frames can be used in dynamic geometry to learn about properties of curves.

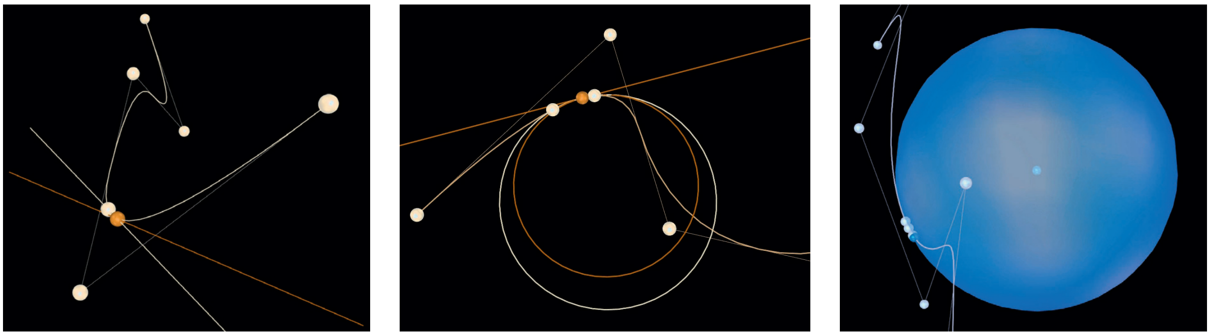
4.2. Tangency and Contact of m-th order

Introducing differential calculus in high school frequently starts by introducing the difference quotient. Graphically the slope of a tangent to a function graph is calculated by choosing two points on a curve. One is the point of interest T where the slope of the tangent needs to be calculated, the other one is an arbitrary point P on the curve. The slope of their secant is computed. When moving P closer to T the secant converges to the tangent and the difference quotient becomes the derivative in the limit case. We call the tangent to be in first order contact with the curve. This can be quickly visualized in Construct3D using an arbitrary curve e.g. a B-Spline curve such as in Figure 4(a).

In general a curve $\mathfrak{c}$ touches a surface $\Phi : F(x_1, x_s, x_3) = 0$ in point $\mathfrak{c}(s_0)$ in $(m+1)$ points (which is contact of order m) if the function $g := F \circ \mathfrak{c}$, i.e. $g(s) = F(x_1(s), x_2(s), x_3(s))$ possesses the root of multiplicity $(m+1)$ i.e.

$$g(s_0) = g'(s_0) = \ldots = g^{(m)}(s_0),\ g^{(m+1)}(s_0) \neq 0$$

Given the possibility of constructing the circle of curvature and the sphere of curvature in a curve point, together with the option of dynamically moving points on curves we can visualize the principle of higher order contact in Construct3D.



(a) The tangent (orange) as the limit case of the secant (white) of two converging points.

(b) Second order contact: A circle of curvature (orange) to a curve (orange) and the approximating circle (white) through three converging points.

(c) Third order contact: The approximation of an osculating sphere (blue) through four converging points (white) on a B-Spline curve is shown.

Figure 4: Visualizing contact of higher order in dynamic geometry.

Utilizing the 'circle of curvature'-feature the circle in a point P has been constructed in Figure 4(b) (orange) to the given curve. It serves as a reference and represents the limit case. In addition two points on the curve were chosen and moved close to P. The circle passing through all three points is displayed in white. Students can move these points and compare the circle to the limit case of second order contact. In the limit case all three points have identical position and the circles coincide.

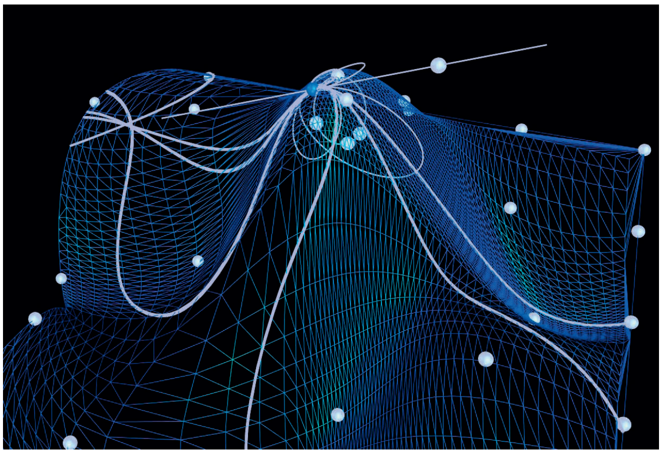
The osculating plane in P possesses second order contact with the curve as well. Likewise the plane of curvature can be approximated by a plane through three converging curve points [2].

Third order contact is established between an osculating sphere and a curve in point P. For a curve $\mathfrak{c}$, the limiting sphere is obtained by taking the sphere that passes through P (drawn blue in Figure 4(c)) and three other points on $\mathfrak{c}$ and then letting these three points converge towards P independently along $\mathfrak{c}$.

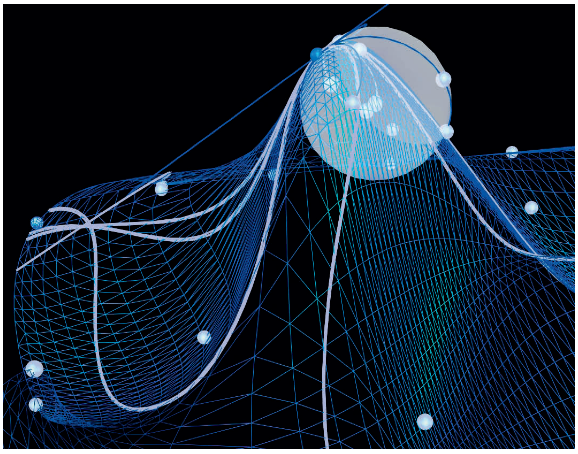
4.3. Meusnier Point

Jean Baptiste Meusnier's Theorem (1779) states that all curves lying on a surface Φ and having at a given point $P \in \Phi$ the same tangent t have the same normal curvature κ_n in this point P. Therefore the normal curvature κ_n is a property of the line element (P, t). Meusnier's Theorem further implies that the circles of curvature in (P, t) of all curves through P with tangent t lie on a common sphere called Meusnier sphere. The midpoint of the Meusnier sphere is the Meusnier point. The centers of curvature of all circles of curvature lie on a common circle.

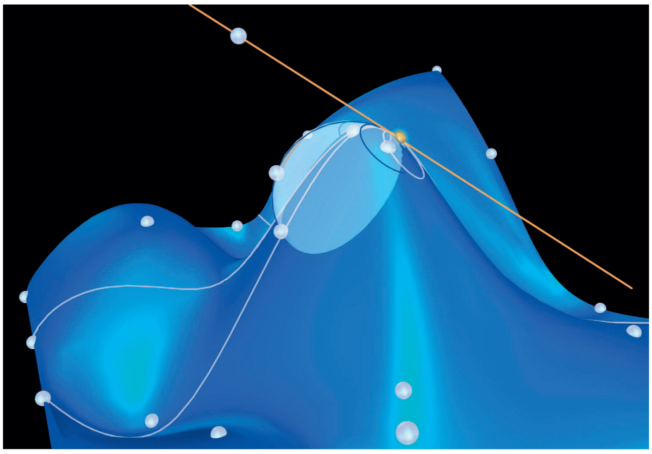
To construct Meusnier's point and sphere in Construct3D we take an arbitrary surface Φ - a NURBS surface with a 5x5 control patch was chosen in Figure 5. We pick an arbitrary point P on Φ, take an arbitrary tangent t through P in its tangential plane to Φ. For further constructions (P, t) is the line element of our choice. Three arbitrary planes through (P, t) are intersected with Φ resulting in three intersection curves. We construct the centers and circles of curvature to all these curves in P and get three centers and circles of curvature (Figure 5(a)). The circle containing all three centers of curvature can be seen in Figures 5(a) and 5(b) (a small white circle containing four points). We visually verified it by checking if a



(a) Intersection curves of four planes through (P, t) (P in blue, t white) with the NURBS surface Φ (blue wireframe). The circles of curvature to three curves in P are visible (white) as well.



(b) The circle containing all centers of curvature (white) to (P, t) also contains the Meusnier point. The Meusnier sphere is displayed transparent white.



(c) Meusnier sphere in (P, t) (orange) containing the circles of curvature (blue).

Figure 5: Meusnier point and Meusnier sphere.

fourth center of curvature coincides with it as well.

The Meusnier sphere contains all circles of curvature in (P, t) and therefore also the constructed ones. Four points were chosen on the circles and then the sphere passing through all of them was constructed (by intersecting their symmetry planes). This gives the center of the sphere, the Meusnier point to (P, t) (Figure 5(b)). Finally the Meusnier sphere in (P, t) is shown in Figure 5(c).

By moving P on Φ the Meusnier sphere, the circles of curvatures, the intersection curves and all other depending elements can be studied.

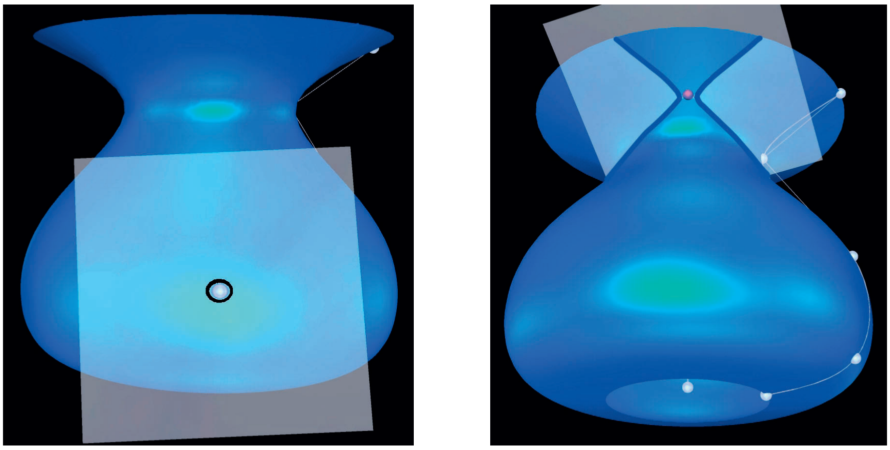
In a teaching lesson students can for example investigate in which cases the Meusnier point is identical to the center of the osculating sphere in a curve point. By dynamic exploration it is straighforward to find cases where the Meusnier sphere degenerates.

4.4. Classification of points on a surface

Curvature in curve points can be visualized well by drawing the circle of curvature. The curvature vector of a curve $\mathfrak{c}$ on a surface can be written as $\mathfrak{c}'' = \kappa_g \mathfrak{g} + \kappa_n \mathfrak{n}$. $\mathfrak{g}$ is a vector normal to the normal vector $\mathfrak{n}$ in the tangential plane, κ_g is called geodesic curvature and κ_n normal curvature.

Surface points can be classified regarding their curvature into elliptic, hyperbolic and parabolic points. The Dupin indicatrix visualizes and describes curvature properties at a point of a surface. It is named after Pierre Charles Francois Dupin (1813), who was the first to use this curve in the study of surfaces.

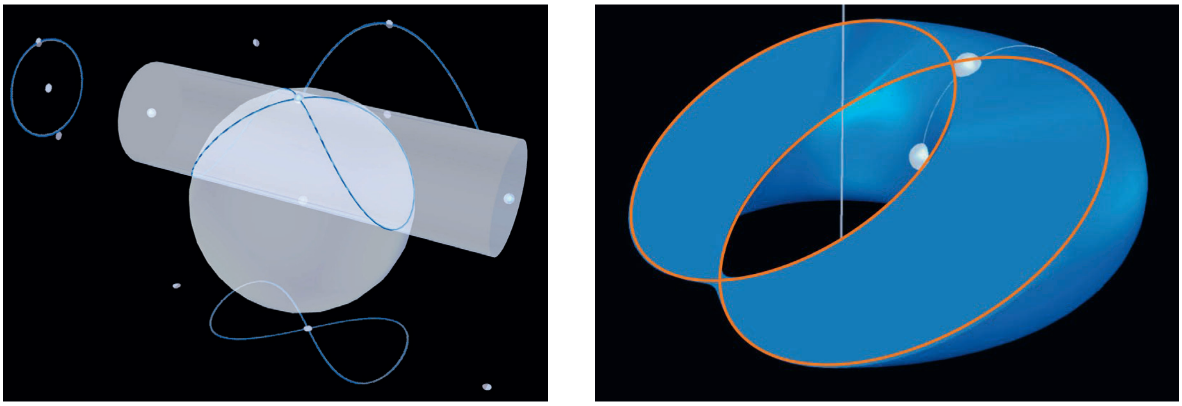
Dupin's indicatrix lies in the tangential plane to the surface Φ at point P. $\rho^N(t)$ is the radius of the normal curvature in direction t: $\rho^N(t) = 1/\kappa_n(t)$ and k is a positive number $k > 0$. If we take any tangent t in the tangential plane of point P and plot the length $\sqrt{k\rho^N(t)} > 0$ on both sides of P on t then we get a point set in the tangential plane - symmetric around P - called Dupin indicatrix $i(k)$ to the constant k. To each tangent direction the normal curvature can be read out of Dupin's indicatrix if the constant k is known.


(a) Clearly an elliptic point.

(b) Intersection of the offset tangential plane with Φ in a hyperbolic point.

Figure 6: Elliptic and hyperbolic points on a surface of revolution.

The Dupin indicatrix is an ellipse if P is an elliptic point, it degenerates into a circle if the point is an umbilical nonplanar point. For a hyperbolic point, the Dupin indicatrix is a



(a) Viviani's window with additional top, front and left side view.

(b) Villarceu circles.

Figure 7: Learning about properties of interesting curves.

pair of conjugate hyperbolas. For a parabolic point, the Dupin indicatrix degenerates into a pair of parallel lines.

Dupin's indicatrix can also be interpreted as an intersection of a plane, parallel to the tangential plane in P, which is offset by an infinitesimally small amount. In order to 'visualize' Dupin's indicatrix in Construct3D the tangential plane in a surface point was offset by an epsilon value in the direction of the surface normal vector. The intersection of this minimally offset plane with the surface is a visual indication to which type - according to the above mentioned classification - the point belongs to. The intersection resembles an ellipse in an elliptic point and resembles a hyperbola in a hyperbolic point. Figure 6 shows two examples of intersection curves in points of a surface of revolution. Parabolic points can be visualized easily if the surface of choice is a cylinder for instance. All points on a cylinder are parabolic and the intersection with the offset tangential plane are two parallel generator lines.

4.5. Studying Interesting Curves

Many interesting curves came into mind when considering educational applications of the presented work. A curve that is frequently studied is Viviani's Window (Figure 7(a)). Top (lemniscate), front (circle) and left side view (parabolic segment) are shown in Figure 7(a). There are multiple interesting properties of Viviani's Window that can be studied in this context.

Another example are the Villarceau circles (Figure 7(b)) produced by cutting a torus diagonally by a double tangential plane. The Villarceau cirlces are loxodromes of the torus [24]. In Construct3D this can be explored visually by moving a point along the Villarceu circles together with its Frenet frame. Observing the tangents' angle to the circles of longitude and latitude in that point shows that it stays constant during movement.

Without going into further detail it is obvious that there is a wide variety of content that can be studied in a dynamic geometry application such as Construct3D which provides differential geometric functionality.

5. Conclusion and Future Work

In this paper we introduced three-dimensional differential geometry in dynamic geometry software. We showed the applicability of Construct3D for dynamic differential geometry education in a wide range of examples. The content in section 4 has not been evaluated with students yet but previous evaluations with Construct3D have been comprehensive [12]. They provided constructive feedback that improved technological development as well as content design [14] and have been taken into account when developing the examples presented here.

Augmented or virtual reality is supposed to enrich traditional geometry education, not to substitute it. There are still major obstacles to overcome before these technologies may be used in schools which are mainly related to costs - costs of hardware but also of technical personnel to run and maintain technologically complex setups. In order to bring augmented reality to schools further technological developments are needed to lower prices of necessary hardware equipment and to develop alternative setups which enable larger groups of students to participate in the learning experience.

Regarding future work we are investigating the hypothesis that students' spatial abilities can be improved by training in augmented reality in an ongoing research project. Therefore an extensive psychological study with more than 250 students is currently under way.

6. Acknowledgements

The author thanks Marie-Theres Tschurlovits for her help with implementing some of the differential geometry features described in this paper and Boris Odehnal for inspiring ideas regarding interesting curves.

References

[1] R. Azuma, *A survey of augmented reality*, PRESENCE: Teleoperators and Virtual Environments, 6 (1997), pp. 355–385.

[2] G. Bol, *Zur definition von schmiegfiguren in der differentialgeometrie*, Archiv der Mathematik, 1 (1948), pp. 362–370.

[3] J. Corney and T. Lim, *3D Modeling with ACIS*, Saxe-Coburg Publications, 2002.

[4] P. R. d. Cotret, Sophie Cotret, *Cabri 3D User Manual*, Cabrilog, 2005.

[5] K. Engel, M. Hadwiger, J. Kniss, C. Rezk-Salama, and D. Weiskopf, *Real-Time Volume Graphics*, A K Peters, Ltd., 2006.

[6] G. Gibbons, *Part III: Applications of Differential Geometry to Physics (Course Material)*, Department of Applied Mathematics and Theoretical Physics, University of Cambridge, January 2006.

[7] A. Goebel, *Archimedes Geo3D User Manual*, http://raumgeometrie.de, 2008.

[8] C. M. Hoffmann, *How solid is solid modeling?*, in Applied Computational Geometry, M. Lin and D. Manocha, eds., vol. LNCS State-of-the-Art-Survey, Springer Verlag, 1996, pp. 1–8.

[9] C. M. Hoffmann, *Solid modeling*, in CRC Handbook on Discrete and Computational Geometry, J. E. Goodman and J. O'Rourke, eds., CRC Press, Boca Raton, FL, 1997, pp. 863–880.

[10] N. Jackiw and W. F. Finzer, *The geometer's sketchpad: Programming by geometry*, in Watch What I Do: Programming by Demonstration, Cambridge, MA: The MIT Press, 1993, pp. 293 – 308.

[11] H. Kaufmann, *Geometry Education with Augmented Reality.*, PhD thesis, Vienna University of Technology, 2004.

[12] H. Kaufmann and A. Dünser, *Summary of usability evaluations of an educational augmented reality application*, in Virtual Reality, Lecture Notes in Computer Science 4563, R. Shumaker, ed., vol. 14, Springer Berlin / Heidelberg, 2007, pp. 660–669.

[13] H. Kaufmann and D. Schmalstieg, *Mathematics and geometry education with collaborative augmented reality*, Computers & Graphics, 27 (2003), pp. 339–345.

[14] H. Kaufmann and D. Schmalstieg, *Designing immersive virtual reality for geometry education*, in VR '06: Proceedings of the IEEE conference on Virtual Reality, Washington, DC, USA, 2006, IEEE Computer Society, pp. 51–58.

[15] U. Kortenkamp, *Foundations of Dynamic Geometry*, phd thesis, Institut für Theoretische Informatik, ETH Zürich, 1999.

[16] C. Laborde, *The computer as part of the learning environment : the case of geometry*, in Learning from computers, mathematics education and technology; NATO ASI Series, C. Keitel and K. Ruthven, eds., vol. 121, Springer, Berlin, 1993, pp. 48–67.

[17] R. Mechling, *EUKLID DynaGeo von innen - ein Blick hinter die Kulissen*, http://www.dynageo.com, 1999.

[18] M. Pfeifer, W. Rath, and H. Stachel, *Handbuch zu CAD-3D für Windows*, Institute of Geometry, Vienna University of Technology, 2002.

[19] T. Pintaric and H. Kaufmann, *Affordable infrared-optical pose-tracking for virtual and augmented reality*, in Proceedings of Trends and Issues in Tracking for Virtual Environments Workshop, IEEE VR, Charlotte, NC, USA, March 2007, pp. 44–51.

[20] E. Podenstorfer, *Handbuch GAM: Generieren, Abbilden, Modellieren*, http://www.gam3d.at, 2009.

[21] J. Richter-Gebert and U. Kortenkamp, *The Interactive Geometry Software Cinderella*, Heidelberg: Springer, 1999.

[22] D. Schmalstieg, A. Fuhrmann, G. Hesina, Z. S. Szalavri, L. M. Encarnacao, M. Gervautz, and W. Purgathofer, *The studierstube augmented reality project*, Presence-Teleoperators and Virtual Environments, 11 (2002), pp. 33–54.

[23] Z. S. Szalavri and M. Gervautz, *The personal interaction panel - a two-handed interface for augmented reality*, Computer Graphics Forum, 16 (1997), pp. 335–346.

[24] W. Wunderlich, *über die torusloxodromen*, Monatshefte für Mathematik, 56 (1952), pp. 313–334.

EUROGRAPHICS 2006 / Werner Hansmann and Judy Brown
(Editors)

Long Distance Distribution of Educational Augmented Reality Applications

H. Kaufmann[1], M. Csisinko[1], A. Totter[2]

[1]Institute of Software Technology and Interactive Systems, Vienna University of Technology, Austria.
[2]Organisation, Work and Technology Group, Swiss Federal Institute of Technology, Switzerland.

Abstract

For distance education utilizing shared Virtual or Augmented Reality (VR/AR) applications, reliable network distribution of educational content is of prime importance. In this paper we summarize the development of software components enabling stable and reliable distribution of an existing educational AR application for geometry education. Our efforts focus on three main areas: (1) For long distance distribution of Open Inventor scene graphs, throughout a wide area IP network, a TCP based network protocol was implemented in Distributed Open Inventor. (2) A tracking middleware was extended to support sending tracking data unicast instead or in addition to sending multicast messages. (3) Multiple adaptations in our geometry application were required to improve scalability, robustness and reliability. We present an early evaluation with high school students in a distant learning, distributed HMD setup and highlight final results.

Categories and Subject Descriptors (according to ACM CCS): K.3.1 [Computer Uses in Education]: Distance learning, I.3.2 [Graphics Systems]: Distributed/network graphics, K.3.1 [Computer Uses in Education]: Collaborative learning, H.5.1 [Multimedia Information Systems]: Artificial, augmented, and virtual realities.

1. Introduction

In order to use Virtual or Augmented Reality applications in realistic, educational settings, a large group of students must be able to participate either actively or passively in the activities taught in VR/AR. In distance learning with VR/AR, reliable network distribution and replication of educational content is of prime importance.

Our work is based on the educational Augmented Reality application Construct3D [KS06, KS03]. This system deploys Augmented Reality (AR) to provide a natural setting for face-to-face collaboration of teachers and students. The main advantage of using AR is that students actually see three dimensional objects which they until now had to calculate and construct with traditional (mostly pen and paper) methods (Figure 1). By working directly in 3D space, complex spatial problems and spatial relationships may be comprehended better and faster than with traditional methods. Our system utilizes collaborative augmented reality as a medium for teaching, and uses 3D dynamic geometry to facilitate mathematics and geometry education. Pedagogical aspects influenced the design of collaborative AR hardware setups, user interface design and content design as reported in [Kau04]. This paper focuses on the technical development and recent advancements enabling distribution in order to serve groups of students and pedagogical findings when using Construct3D in a distributed setup for distance education.

Figure 1: *Collaborative co-located work in Construct3D.*

When sharing a virtual workspace for collaboration with people at distant locations, distribution and replication of data has to be taken into account.
Ideally, data transmission should be fast to achieve fast response times. Especially in long distance distribution this aspect is crucial, as the travelling time depends on the distance to cover. In addition data transmission has to be (in most cases) reliable. The amount of transmitted data should be low to achieve fast response times, to prevent network congestion, and to increase efficiency. We distinguish between different types of distributed data:

- Input data distribution:
  Tracked input device data to visualize the actions and movements of other participants, especially those working in distant locations. This type of data is typically sent by a tracker server in VR/AR environments.
- High level application state distribution:
  Shared application state, in the form of compacted metadata to reduce the amount of transmitted data. Ideally these metadata suffice to regenerate the correct application state without actually transmitting the whole application.
- Output data and application content distribution:
  (Educational) application content or additional application data that needs to be shared.

To allow collaboration between distant users immersed in a common shared space, a consistent application state is required throughout all participating sites. This implies that each participant perceives a similar virtual world, although slight differences might be possible. In a teaching scenario, for example, user roles (teacher - student) could be defined which result in displaying additional information - such as the solution of a given 3D construction - to a teacher, whereas the student does not see the solution.

1.1. Contribution

Hesina [Hes01] introduced Distributed Open Inventor (DIV) and distribution features in Studierstube [SFH*02], our Augmented Reality software framework, but a series of remaining shortcomings had to be resolved. Existing functionality was restricted to local networks, because the network implementation makes use of multicast UDP. This networking mode, though theoretically ideal for the task at hand, lacks of support for long distance distribution: As a major drawback multicast UDP packets are in general not sent through arbitrary routers on the internet (unless part of the MBONE network [Eri94]). Therefore immediate distribution between school networks, without setting up multicast tunnels in cooperation with local administrators is not possible. In our experience gathered in past e-learning projects, these obstacles (which require time and effort of school personnel) usually prevent usage of the technology in an educational setting.

Our work is supposed to fill this gap, enhancing distribution features, overcoming the rigid restriction in terms of networking and offering the possibility to truly distribute Studierstube applications such as Construct3D to remote places. Due to the practical usage of DIV by Hesina for the past 5 years, the shortcomings (as mentioned above) became obvious. Increased interest in Construct3D and collaborative projects with partners in other countries required a flexible network implementation, breaking free from standard lab setups in a LAN. Efficient mechanisms for distributing AR/VR applications over long distances had to be implemented. This was done on 3 levels:
(1) Tracking middleware (OpenTracker [RS01]) was extended to send data of tracked input devices in unicast UDP mode in addition or instead of multicast UDP.
(2) Distributed Open Inventor [Hes01] was extended to send data using (reliable) TCP instead of reliable multicast UDP. It enables long distance distribution but also leads to a considerable performance increases in small networks compared to the multicast UDP implementation.
(3) Construct3D [KS03] (section 4) was selected to make in-depth long distance distribution tests. In addition to extending distribution functionality, we enhanced the replication behavior of the application. Initially the whole application state - as a scene graph containing all geometric objects - was transmitted which resulted in a high amount of transmitted data. To minimize the network load only state data is being transmitted which enables clients to rebuild the whole application state themselves. Therefore distribution is basically restricted to meta information in the form of command lists containing essential application states. Executing a command list generates the whole geometric construction and application state.

Another huge amount of work was spent on massively increasing robustness of present features by bug-fixing and reimplementation as well as extending them and introducing new functionality to push the application further. They are of major importance for a stable educational application but are mainly omitted in this context.
Finally an early evaluation of a distributed educational setup is presented which shows the usefulness of utilizing AR/VR applications, in example Construct3D, in distance education.

2. Related work

For the development of any educational, distributed VR/AR application, technological, domain specific, pedagogical and psychological aspects are of importance. Accordingly, literature from different and diverse research areas relates to our work: Tracking frameworks, distributed scene graphs, collaborative AR/VR, distributed virtual environments, desktop and immersive 3D modeling, educational 2D/3D applications, dynamic geometry and pedagogic theories such as constructivism or activity theory. We will briefly mention work related to the core parts of our work. For a comprehensive overview of related work regarding Construct3D we refer to [Kau04].

2.1. Tracking frameworks

With the wealth of different tracking systems and input devices available, it is impossible for application developers to deal with the details necessary to support each and every technology natively in their applications. Instead, it is desirable to add another level of abstraction, and try to encapsulate the details of the necessary software support for various tracking technologies in a tracking middleware. The goal of tracking middleware is to serve tracking (and other input) data to the application, independent of the underlying hard- and software. Several middleware systems for tracking devices have been developed.
VRPN (Virtual-Reality Private Network) [THS*01] is a wide spread device-independent and network-transparent framework for peripheral devices used in Virtual and Augmented Reality applications written in C++. Networking is built upon UDP and TCP. Depending on the reliable delivery property of the tracking data type, the protocol is chosen on a per message basis.
With OpenTracker [RS01] it is also possible to support distinct tracking device types by abstraction, to perform various preprocessing tasks (such as filtering) and network transmissions within a single framework. An OpenTracker client is integrated into the Studierstube [SFH*02] toolkit for tracking device support.

2.2. Distributed scene graphs

Current high-level 3D graphics libraries are engineered around the concept of a scene graph, a hierarchical object oriented data structure of graphical objects. Such a scene graph gives the programmer an integrated view of graphical and application specific data, and allows for rapid development of arbitrary 3D applications. Although shared memory systems are capable of directly sharing data, they have additional hardware requirements. Distributing and replicating scene graphs among heterogeneous computer systems does not require additional hardware.

The blue-c Distributed Scene Graph (bcDSG) [NLSG03] is based on OpenGL Performer. Distribution features are added on top of the blue-c framework and are not integrated into Performer. The scene graph can be divided into a shared and local partition. Shared parts have to be created using custom nodes, as standard Performer nodes do not support distribution. Scene graph synchronization is performed in a traversal operation at each rendering. This mechanism includes consistency, locking and ownership features.
Data transfer is done using UDP, enabling multicast support for more than two participating sites: While scene graph synchronization messages are transmitted to any participating site, locking operations are of unicast nature. Relying on multicast UDP and its routing deficients, the system will experience afore mentioned problems when used for large distance distribution. Its synchronization features are based on nodes as atomic units: Changing a single field causes the whole node contents to be transferred. This can be problematic, when having huge amount of data belonging to a single node.

Avango [Tra99] is also based on Performer. Similar to the Inventor toolkit, its own scene graph nodes act as field containers, storing data in terms of fields. In addition field connection and streaming mechanisms are introduced similar to existing concepts in Inventor. Distribution features are based on so-called distribution groups. To build a shared object, a local object has to be created and migrated to a distribution group. On the receiving end all group members reverse this process by creating a local copy of the distributed object.

Distributed Open Inventor is based on Open Inventor (OIV), a popular scene graph toolkit. Several implementations of adding distribution features to OIV exist:
Distributed Open Inventor (DIV) by Hesina [Hes01] is a stand alone open source add-on to OIV, and has also been integrated into the Studierstube framework [SRH03]. It enables sharing of a scene graph or parts of it in a network, which is a fundamental prerequisite for (distant) collaboration in AR/VR environments. If encapsulating application and its graphical object state altogether in a scene graph, distribution of that scene graph avoids the dual database problem [MF98]. Since DIV provides the basis of our work, we will describe some of the concepts implemented in DIV in detail:
The implementation makes use of the notification mechanism in OIV and observes occurred scene graph changes by sensors. On an atomic level changes of field values are monitored. For convenience, a special group called DivGroup denotes a subtree for distribution, offering the possibility to share several independent parts of a scene graph.
Usually a single master hosts the original copy of the scene graph for replication to guarantee total ordering of messages. The master is responsible for transmission of scene graph changes to the network. In this transmission the node name (where the field value change occured) is used as unique identifier and naming lies in the responsibility of the master. Scene graph modification messages transmitted by the master typically contain the name of the node where the change occurred with additional information such as (a) appropriate field data, if a field update occured or (b) structural information, if the update is of structural nature (involving group node operations).
Slaves process received changes and modify the scene graph. Initially, slaves are also capable of sending polling packets to the network, requesting the scene graph from the master. The master reacts on this message appropriately by transmitting the scene graph in its actual state. This is actually the implementation of a late joining feature. Networking is based on the ACE toolkit.

A similar approach to distributed Open Inventor was implemented by Pečiva [Peč02], also based on a master-slave architecture. Similar to Hesina, he extended the Open

Inventor source directly. Therefore scene graphs can be set up for distribution without replacing standard nodes by a customized counterpart.

2.3. Educational VR applications

Since the early 1990th researchers have been working on virtual reality applications for purely educational use ([DSL96, WB92] and many others).
In the area of mathematics education the most advanced immersive VR project is CyberMath [TN01]. CyberMath is an avatar-based shared virtual environment aimed at improving mathematics education. It is suitable for exploring and teaching mathematics in situations where both teacher and students are co-present or physically separated. It has been presented in a CAVE and exists as a desktop VR application. The recent VRmath system [YN04] is an online application that utilises desktop VR combined with the power of a Logo-like programming language, hypermedia and the Internet to facilitate learning of 3D geometry concepts and processes. A very good summary of educational VR applications is given by Mantovani [Man01].

2.4. Pedagogic theory

Constructivist theory provides a valid and reliable basis for a theory of learning in virtual environments [Osb97, Win93]. As constructivism underlines, learning takes place when students build conceptual models that are both consistent with what they already understand and with the new content.
The core commitment of a constructivist position is that knowledge is not transmitted directly from one knower to another but is actively built up by the learner. Learning is considered to be an active process in which learners "construct" their own knowledge by testing ideas and approaches based on their prior knowledge and experience, applying these to a new situation, and integrating the new knowledge gained with pre-existing intellectual constructs. This is supported through relevant, engaging learning activities, which involve problem-solving and critical thinking. We used activity theory [Eng99, TGG04] as a conceptual framework to design constructivist learning tasks for our evaluation. Details are given in subsection 5.2.

3. Distribution - Technical Design

In this section we provide a brief overview of our design of the three components that were extended to support long-distance distribution: OpenTracker, Distributed Open Inventor and Studierstube. In section 4 Construct3D is described.

3.1. Tracking data distribution in OpenTracker

OpenTracker [RS01] contains components providing tracking data transmission between several OpenTracker instances on different hosts. Just like Distributed Open Inventor these capabilities are built upon multicast UDP, which causes the earlier mentioned multicast-related problems.

Following the data flow principle of OpenTracker, tracking data is inserted into the data flow graph by means of a so called NetworkSource, while a NetworkSink transmits data to the network. This implies that network traffic concerning tracking data is unidirectional and of multicast nature: Payload data is always transmitted by a single NetworkSink and received simultaneously by one or more NetworkSources. It was rather straightforward to add unicast UDP as an additional networking protocol. A NetworkSink generating tracking data packets has to deliver them simultaneously to associated receivers. To know all receivers, the NetworkSink has to maintain a list of counterparts (each of them usually a NetworkSource of an OpenTracker instance on the receiver side).

The network topology on the logical level is a star. This topology implies that tracking data of several devices can be distributed by a single network, as long as data occurs on the same central location. Of course building several independent networks (consuming more network resources) is the alternative and more general way, as this allows distribution of tracking data occurring at different places. Establishing a tracking data network is initiated by NetworkSinks, similar to the traditional server-client scenario: Each client must have knowledge in advance about the server providing desired tracking data in terms of socket information (host and port).

3.2. Distributed Open Inventor

In general Distributed Open Inventor is utilized to distribute parts of a scene graph. Since scene graph data usually contains important application information, network communication must be reliable. To overcome the borders of private local networks, TCP as a very widespread and reliable network protocol was chosen. This implies a lot of changes to Hesina's implementation.

In contrast to multicast UDP, TCP allows only point-to-point communication. On the other hand no reliability treatment is necessary in TCP as this is implicitly taken care of in the protocol. To allow data delivery to all network nodes, the TCP implementation, considering its unicast nature, has to emulate multicast data delivery to comply to the requirements of Distributed Open Inventor. As any node might act as a server, a many-to-many property has to be taken into account.

Multicast data delivery with multiple senders is done by building up a logical network of so called true mesh topology. This is a network, where each peer is logically connected to each other peer. The main challenge of the TCP implementation is to establish and ensure true mesh topology at any time. Sending and receiving is, as mentioned before,

fairly simple: Data is automatically transmitted to each connection simultaneously. On the receiving end nothing special has to be taken into account. Processing order of data received from different connections is uncritical as the next higher network layer implies that critical data in terms of processing order is sent from exactly one peer at any time.
Whenever a peer joins the distribution network it must know at least one peer of the existing network. Otherwise it will be the only participant of a new network.
A new peer initially contacts the network by sending a special message identifying itself just after connection establishment. This identification contains the server port of the peer as each peer contains server and client functionality. The arrival of a new peer must be forwarded to all other participating sites of the network. All other peers establish connections to the new peer identifying themselves.
On receiving any identification message, the peer has to check, if another connection to the counterpart currently exists. If this is the case, the connection is closed immediately. Since network redundancy involving more than a single peer is effectively prevented in advance, connection closing should only be done in case of cycles. Livetime information (time to live (TTL)) is embedded in the message of a joining peer. It allows only a certain low number of hops between peers. This increases efficiency and helps to avoid infinite cycles.

3.3. Studierstube

The enhancements of Distributed Open Inventor have to be reflected in Studierstube. Studierstube applications are distributed automatically by a DivGroup, implicitly created as a parent of each application. Configuration of the distribution is handled by the core library with the help of a tool called session manager. Implicit distribution is implemented in favor of ensuring distribution capabilities without further intervention by the application programmer.

The session manager assigns master property to the participant who originally hosts a certain application. All other participating sites (slaves) receive the application scene graph via network due to the node transfer feature. Terminating the master results in reassigning master property to another participant. This master-slave property assignment is conducted autonomously and cannot be influenced by Studierstube instances. Another task of the session manager is to create network resources according to the requested networking mode. To do this, a generator produces network configuration data. In multicast UDP mode, a single multicast group address and associated port number is generated per application. In TCP mode, each peer is given a unique port number to allow running several applications on a single machine.
As the assignment of network resources lies in the responsibility of the session manager, it simply creates a list containing proper contact information of all other participants for each peer and includes this in reconfiguration messages sent to each participating site. This strategy guarantees highest chances to contact any of the other peers to successfully build up a network.

Further details about design and implementation of our approach are described in [Csi06].

4. Construct3D

Construct3D is based on the Studierstube AR system [SFH*02] and uses augmented reality to provide a natural setting for face-to-face collaboration of teachers and students. Based on an underlying distribution mechanism, Studierstube extends its support to multiple users working with multiple different display techniques in a shared workspace that features multiple applications and management techniques similar to a common 2D desktop [SRH03]. Studierstube applications are custom nodes which are part of the scene graph. As Construct3D is just another Studierstube application, it inherits automatically its distribution features.

4.1. Software design

Construct3D offers functions for the construction of points, two-dimensional geometric primitives and three-dimensional geometric objects. It provides functionality for planar and spatial geometric operations on these objects, allows measurements, features structuring of elements into layers and offers basic system functions.
Construct3D promotes and supports exploratory behavior through dynamic geometry. A fundamental property of dynamic geometry software is that dynamic behavior of a construction can be explored by interactively moving individual defining elements such as corner points of a rigid body. For example, moving a point lying on a sphere results in the change of the sphere's radius. It can be seen what parts of a construction change and which remain the same. The histories of constructions as well as dependencies between geometric objects are maintained. Experiencing what happens under movement allows better insight into a particular construction and geometry in general.
At its start Construct3D initializes a 3D window and the user interface. The menu system is mapped to a hand-held tracked panel called the personal interaction panel (PIP) [SG97]. The PIP allows the straightforward integration of conventional 2D interface elements like buttons, sliders, dials etc. as well as novel 3D interaction widgets (Figure 2). Passive haptic feedback from the physical props guides the user when interacting with the PIP, while the overlaid graphics allows the props to be used as multi-functional tools. Students can position written notes onto the tablet for instance that might help them during their work in the virtual environment.
All construction steps are carried out via direct manipulation in 3D using a stylus tracked with six degrees of freedom. In order to generate a new point the user clicks with his pen

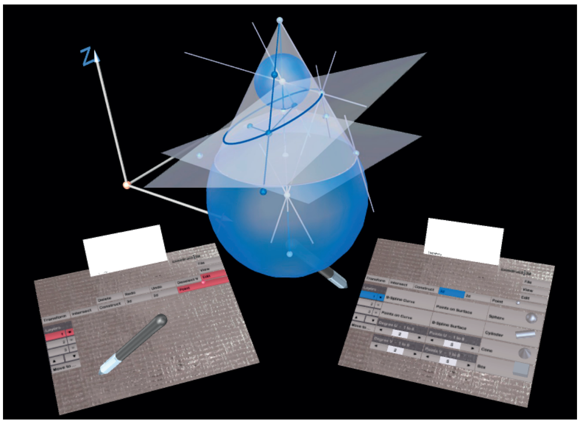


Figure 2: *Two users collaborate on a construction in Construct3D. To distinguish users' contribution each user is working within an own color scheme (note the differently colored menus).*

exactly at the location in 3D space where the point should appear. Users can easily switch between point mode (for setting new points) and selection mode (for selecting 3D objects). All 3D operations consistently support dynamic modifications of their input elements and re-evaluate the resulting elements accordingly. Necessary system operations such as selection and deselection of primitives, save, load, delete, undo, redo, export and import of VRML files are provided too. Details on the implementation, specifically the implementation of undo, redo and other features for multi-user environments are explained in detail in [Kau04].

The internal structure of the application's scene graph is rather simple. Avoiding the dual database problem [MF98], it encapsulates all of the application's data. Basically the scene graph hierarchy is composed by command lists storing all Construct3D operations. Geometric objects are the visible results of these operations. A command list and its interpretation (= the geometric elements) are represented by node kits. The command list represents the meta-state of the application, the node kits containing the geometric elements are rendered and represent the visual state.

These two distinct parts forming the application's scene graph are interrelated: Manipulation of geometry causes the generation of new commands in the command history list. On the other hand, the execution of commands generates deterministic results on visible geometry. Therefore the command history list is used for file operations (load/save) and for the undo/redo functionality.

An important property of the command history list is that it allows the complete regeneration of the application state at any specific time during the construction process. Therefore distributing the command history list alone is sufficient to rebuild the correct application state on any client computer. Other parts of the scene graph are excluded from distribution. As each action causes a modification of the command list position pointer, the latter is a key element in assisting the detection of changes caused by distribution. Whenever the shared pointer changes, actions have to be taken.

5. Evaluation

Being one of the longest actively developed educational AR applications, Construct3D has been used with teachers and students in more than 500 teaching lessons yet. Usability aspects of Construct3D and pedagogical content design have been evaluated in two previous evaluations as summarized in [Kau04] providing very good results and useful feedback. The current evaluation focuses on distributed, distant learning.

5.1. Collaborative, distributed hardware setup

The standard setup used for Construct3D supports two collaborating users wearing stereoscopic see-through head mounted displays (HMDs) (see Figure 1) providing a shared virtual space. The users interact with the system using pen and pad props. Both users see the same virtual objects as well as each others' pens and menu systems which allows a student or teacher to help the other user with the menu system for instance if necessary. The same is valid in a distance learning scenario since input device data is shared amongst remotely located users. Because of see-through head mounted displays they perceive their real bodies, gestures and actions and those of people outside the virtual space, i.e. a teacher, as well which is especially important for co-located work. Head and hands are tracked using an ARTTrack optical tracking system. In a co-located setup one dedicated host with 2 graphic ports renders stereoscopic views for both users. In distributed setups rendering as well as computation of the geometric objects is done locally on each participant's PC.

Our immersive setup that uses head mounted displays is most favored by teachers and students. The big advantage of this setup is that it allows users to actively "walk around" geometric objects which are fixed in space. Excited students sometimes lie down on the floor to view objects from below or step on a chair to look down from above. This is a unique feature of an HMD setup which cannot be provided by monitor or projection screen based hardware configurations. It actively involves students and therefore complies with constructivist learning theories. Geometric objects are not abstract anymore but in spatial relation to the learner's own body, they can be manipulated directly and are nearly tangible. We think these are key features to learning and to improving spatial abilities with Construct3D.

Other AR setups for educational use have been tested with Construct3D such as a basic desktop setup, semi-immersive, mobile and hybrid setups which are described in detail in [KS03].

5.2. The link to pedagogical theory

In the course of this study the technical requirements of Construct3D, the learning tasks as well as the evaluation methodology were aligned in accordance with pedagogical concepts and learning practices based on constructivism, combined with action-oriented learning such as real-problem solving, collaborative learning, exploratory learning and interdisciplinary learning, stemming from activity theory and the theory of expansive learning [Eng87, Eng99]. In particular, learning tasks had to

- be part of the actual curriculum in schools.
- represent a holistic real life problem. The description (instruction) of the task should be embedded in authentic (real life) context and not at the level of an abstract instruction.
- offer the possibility to be viewed from several perspectives. The focus on different perspectives should support the transfer of knowledge to other similar, but not identical problems.
- be available in multiple representations (different kind of visualisations of the task).
- meet the experience and interests of the students Ű which kinds of tasks they are already familiar with, what kind of problems might they be confronted with in the near future, etc.

The pedagogical theories also influenced the context in which the learning takes place. During the learning process, the following aspects were considered: Learning was an active process, and students collaboratively performed practical tasks to improve their procedural knowledge. Students structured and controlled the learning process. They chose the approach, and the methods for solving the task. The learning process should enable knowledge construction; students should develop their own ideas and approaches. They should be able to identify a contradiction or a conflict in the task. Students should investigate their learning with respect to methods used to organize their information and interpretation. Finally, they should analyze and evaluate their solution with respect to strengths and weaknesses.
The teacher acted as a coach, analysing students' strategies during the collaborative learning process, diagnosing mistakes and misunderstandings and supporting students.

5.3. Evaluation design

Early at the development phase we conducted a first evaluation with distributed Construct3D to investigate its usefulness for distributed collaborative learning and teaching. The evaluation was based on the methodological framework CIELT (Concept and Instruments for evaluating learning technologies [TGG04]). Within the evaluation design we discriminated three phases for the evaluation. The preparation-phase was the period before the actual evaluation sessions start. During the experiments, learning was observed and the assessment-phase concluded the experiments. The evaluation design for the experiments is shown in Figure 3. To be able to derive information about the effectiveness of distributed collaborative learning, we followed a quasi-experimental design for the geometry experiment, which involved splitting learner groups. Seven learner groups performed three geometry tasks in a face-to-face setting, working next to each other, whereas two learner groups performed these three tasks in a distributed setting (Figure 4). For the distributed setting each group member was located in a different room. A learner group consisted of two students and one teacher. In both settings both users were wearing HMDs, were tracked by an optical tracking system or a magnetic tracking system in the second lab, and a teacher was watching (in a different room) on a monitor.

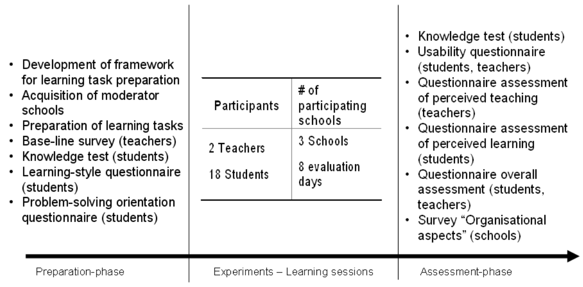


Figure 3: *Students working in a distributed HMD setup in two different labs.*

For this early evaluation all students were located in rooms in the same building, therefore issues of latency in long distance collaboration were not faced at that stage. Long distance collaboration was tested throughout the development process and with the final implementation at an informal level. Application specific adaptations for improved robustness had to be implemented in case of delayed messages arriving from different sources (e.g. tracking data coming from a different source than application data). Large scale testing is part of our future work (see 6.1).

Figure 4: *Students working in a distributed HMD setup in two different labs.*

5.4. Learning tasks

Based on the above mentioned characteristics, three learning tasks were developed. Each group had to accomplish these tasks during the evaluation.

- The first learning task dealt with the wheels of an airplane, which were rotated into a shell in the hull of an airplane after its take off. The wheel in its start and end position were given. Students had to construct the axis of rotation and the angle of rotation for the wheel from its original position into the position in the hull. A screenshot of the given elements that they will face later in the virtual world was given as well.
- In the second task, a satellite dish had to be adjusted to point to the TV-SAT2 satellite. Students had to translate this real life problem into a geometric problem to be able to identify two angles, which are needed to adjust the satellite dish. Web links were presented with additional information about geostationary satellites; images were also given to help understand and translate the problem. The virtual scene in Construct3D showed a small model of the earth where all continents and seas could be seen, to help pupils find the correct places on earth and to immerse them further into the problem.
- In task 3 a rope was redirected from one given position to a final given position. Two deflection sheaves, which were drawn as circles, redirected the rope. These had to be constructed by the students. Deflection sheaves can be found in skiing lifts, elevators and many other machines. A draft was given to lead the students to a correct solution.

5.5. Evaluation results

Overall 18 students and 2 teachers participated in the evaluation of the C3D system. Students were between 17-18 years old and attended grade 11 or 12 of Austrian senior high school. They all head average to good computer experience.

The usability of the C3D system was measured with the ISONORM 9241/10 questionnaire [Prü97]. The overall usability of C3D was rated good, with M=2.37 (SD=0.42; with values ranging from "1=Completely adequate" to "5=Completely inadequate"). When analysed with respect to the two settings, distributed and face-to-face learning, there was no significant difference between the two groups, except for the principle of suitability for learning. There, groups in the distributed setting rated the suitability for learning statistically significant better than groups in the face-to-face setting (p=.01). We want to point out that due to the small sample, results have to be interpreted carefully. In an additional interview, students of the distributed setting mentioned that due to the distributed setting they concentrated better on what the other person said or did.

Open ended questions from the participants were analysed in a qualitative way to summarize the difficulties experienced while working with Construct3D. Most frequently students complained about an instable distributed system that crashed during the experiments. These difficulties were related to the very early trial of our implementation. We got very useful feedback which helped to make distribution very robust and develop the final system as described in this paper.

Participants also rated the perceived usefulness of Construct3D for meeting (1) learning, (2) communication and (3) collaboration needs (1=very good, 5=not at all). The mean values for those 3 categories were between 1 and 2.14. There was no statistically difference between the distributed and the face-to-face group.

Furthermore participants rated the perceived collaborative awareness (ranging from "always = 1" to "never = 5") based on findings from Caroll et al. [CNI*03]. The authors stated that three aspects of awareness have to be taken into account measuring the effectiveness of the collaboration. Students rated their awareness (while working with Construct3D) of other *working* students with 1.17 (std.dev. $\sigma = 0.707$). Their awareness of *interacting* colleagues was rated with 1.50 ($\sigma = 0.632$) and their awareness that other users are *thinking and planning* was rated with 2.06 ($\sigma = 0.873$). The good ratings could be explained by our very specific application design with respect to supporting multiple users. A different color scheme is used for every user which allows teachers and students to clearly distinguish between each user's contribution [KS06]. In co-located setups collaboration is supported by Augmented Reality, specifically see-through head mounted displays which enable users to see the movements of others.

To investigate the learning outcome we differentiated between successful and non-successful groups. The learning outcome was measured in two ways. Firstly, for each task a specific time frame was defined. After that, groups had to stop working on the task, but the teacher explained the solution. The time frame for the three geometry tasks was set for 45 minutes each to complete the task. Learning sessions either stopped when students solved the task or after the set period of time was reached. Second, a fixed, quasi-experimental design was used, following the traditional pre-knowledge test - intervention - post-knowledge test design. Before the actual experiment started, students had to fill in a multiple choice test, trying to find the correct answers for 8 geometry content related questions. The correct answers of this pre-test constituted the individual base-line for each student, providing information concerning knowledge about the geometry topics that each student had before the actual topics were taught. After performing the three tasks students had to fill in another multiple choice test, again trying to find the correct answers for 8 content related questions. The results of the knowledge pre- and post test could then be compared providing information about how much knowledge had been increased during the experiment.

In the geometry experiment 9 groups (2 students and 1 teacher) from three schools in Austria participated, performing 3 tasks. Five out of the nine groups solved all 3 tasks within the given time frame (45 minutes for each task). Based on the results of the knowledge pre- and post tests we investigated changes in the domain specific knowledge. In the pre-test students were able to answer on average 4 out of the 8 questions (M=4.67), after the experiments they were able to answer almost 6 out of the 8 questions (M=5.86). This difference between the pre- and post-test was statistically significant (p=.003). A positive correlation between pre- and post-test was found (r=.66, p=.003). These results provide first hints that learning in a distributed Construct3D setup has positive effects on the knowledge increase of students. In both co-located and distributed learning groups the knowledge gain was high. No difference was found, therefore distance learning did not effect knowledge gain in any negative way.

We were extremely surprised to see that students collaborated in the distributed, distance learning setup without any problems. Four participants from a school for highly-gifted students were extremely skilled collaborating in the distributed setup. They said that it's even easier than co-located collaboration because they can fully move around geometric objects and can freely interact with them without being considerate of other people who physically share virtual and real space with them in a co-located setting.

6. Conclusion

Running DIV on TCP enables long distance distribution without the effort of tunneling or relying on special infrastructure (MBONE [Eri94]). Depending on the amount and complexity of scene graph data, initial node transfer takes some time. But after this initialization state, interaction has to be fast and responsive.

Comparing the multicast UDP and TCP implementation, it is easily observable that TCP performance is over-topping multicast UDP, especially in small networks: Generating huge amounts of DIV updates by heavily manipulating the scene graph contents, network data throughput in the TCP implementation seems to be much better. On multicast UDP, the send queue gets comparatively quickly full, causing rendering thread blocking of the master. Consequently, interactive manipulation is not possible while having the render thread waiting for dequeuing to take place. Maintaining the same conditions (queue size) while running these massive stress tests, this blocking phenomenon could not be achieved on TCP. We assume that this is related to a sub-optimal implementation of multicast UDP in the ACE network library that we use.

Construct3D is fully benefiting from all distribution features: Multi-user functionality raises the demand for tracking data distribution. Supporting master transfer ability is desired for more flexible use cases. Generally speaking, a very high degree of flexibility is ensured by three orthogonal aspects:

- User configuration and user resources such as output devices, panels and pens can be freely specified.
- The host of the application (DIV master) can be selected without restriction and is completely independent of associated users and their resources. Startup order is completely insignificant and the master automatically migrates by session management on termination.
- Finally, by configuring OpenTracker properly, tracking data distribution (usually done on a separate tracking server) is independent of all other aspects.

Each Construct3D instance can be configured in multiple ways by defining the number of users, its associated resources, specifying application retrieval method (by distribution as slave or by file input as master) and tracking data obtaining strategy.

Further on a central and persistent Construct3D service can be established as a background process without the need of directly associated users and rendering output. This allows joining and leaving a persistent Construct3D learning experiment at any time. In contrast to this, dynamically migrating Construct3D application hosts with directly associated users and rendering tasks is also easily possible without difficult configuration effort, as contacting the session manager performs all bootstrapping.

6.1. Future work

Regarding the technical aspects we omitted the fact that our implementation of DIV supports using multicast UDP and TCP connections simultaneously in a hybrid network configuration. For example a simple hybrid network setup could consist of two local networks with multicast support (e.g. two local school networks or two university networks) which are connected using a reliable TCP connection. Extensive tests with hybrid network configurations are planned since they allow more efficient distribution between multicast enabled subnets.

Since the early evaluation in 2005 no further user studies have been conducted with distributed Construct3D. In order to simulate real classroom conditions, large scale testing of our implementation with a large number of client PCs (> 15) needs to be done. It should ideally be coupled with a large scale evaluation with high-school students.

7. Acknowledgements

The authors thank all participants of evaluations as well as other teachers and students for testing and giving useful feedback. Part of this research was funded by the Austrian Science Fund (FWF) contract P16803 and by the EU IST project Lab@Future (IST-2001-34204).

References

[CNI*03] CARROLL J. M., NEALE D. C., ISENHOUR P. L., ROSSON M. B., MCCRICKARD D. S.: Notification and awareness: synchronizing task-oriented collaborative activity. *International Journal of Human-Computer Studies 58* (2003), 605–632.

[Csi06] CSISINKO M.: *Long Distance Distribution of VR and AR Applications*. Master's thesis, 2006.

[DSL96] DEDE C., SALZMAN M. C., LOFTIN R. B.: Sciencespace: Virtual realities for learning complex and abstract scientific concepts. *Proceedings of IEEE VRAIS '96* (1996), 246–252.

[Eng87] ENGESTRÖM Y.: *Learning by Expanding: An Activity-Theoretical Approach to Developmental Research*. Helsinki: Orienta-Konsultit Oy, Finland, 1987.

[Eng99] ENGESTRÖM Y.: Activity theory and individual and social transformation. In *Perspectives on Activity Theory*, Engeström Y., Miettinen R., Punamäki R.-L., (Eds.). Cambridge University Press, UK, 1999.

[Eri94] ERIKSSON H.: MBONE: the multicast backbone. *Commun. ACM 37*, 8 (1994), 54–60.

[Hes01] HESINA G.: *Distributed Collaborative Augmented Reality*. PhD thesis, Institute of Computer Graphics and Algorithms, Vienna University of Technology, Austria, 2001.

[Kau04] KAUFMANN H.: *Geometry Education with Augmented Reality*. PhD thesis, Vienna University of Technology, Vienna, 2004.

[KS03] KAUFMANN H., SCHMALSTIEG D.: Mathematics and geometry education with collaborative augmented reality. *Computers & Graphics 27*, 3 (2003), 339–345.

[KS06] KAUFMANN H., SCHMALSTIEG D.: Designing immersive virtual reality for geometry education. In *Proceedings of IEEE Virtual Reality Conference 2006* (Alexandria, Virginia, USA, 2006), pp. 51–58.

[Man01] MANTOVANI F.: Vr learning: Potential and challenges for the use of 3d environments in education and training. In *Towards CyberPsychology: Mind, Cognitions and Society in the Internet Age*, Riva G., Galimberti C., (Eds.). IOS Press, Amsterdam, 2001.

[MF98] MACINTYRE B., FEINER S.: A distributed 3D graphics library. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1998), ACM Press, pp. 361–370.

[NLSG03] NAEF M., LAMBORAY E., STAADT O., GROSS M.: The blue-c distributed scene graph. In *EGVE '03: Proceedings of the workshop on Virtual environments 2003* (New York, USA, 2003), ACM Press, pp. 125–133.

[Osb97] OSBERG K. M.: *Constructivism in practice: the case for meaning-making in the virtual world*. PhD thesis, University of Washington, 1997.

[Peč02] PEČIVA J.: *Development of a Distributed Scene Toolkit Based on Open Inventor*. Master's thesis, 2002.

[Prü97] PRÜMPER J.: Der Benutzungsfragebogen ISONORM 9241/10: Ergebnisse zur Reliabilität und Validität. In *Software-Ergonomie '97*, Liskowsky R., (Ed.). Stuttgart, 1997.

[RS01] REITMAYR G., SCHMALSTIEG D.: An open software architecture for virtual reality interaction. In *VRST '01: Proceedings of the ACM symposium on Virtual reality software and technology* (New York, USA, 2001), ACM Press, pp. 47–54.

[SFH*02] SCHMALSTIEG D., FUHRMANN A., HESINA G., SZALAVARI Z. S., ENCARNACAO L. M., GERVAUTZ M., PURGATHOFER W.: The studierstube augmented reality project. *Presence - Teleoperators and Virtual Environments 11*, 1 (Feb 2002), 33–54.

[SG97] SZALAVARI Z. S., GERVAUTZ M.: The personal interaction panel - a two-handed interface for augmented reality. *Computer Graphics Forum 16*, 3 (1997), 335–346.

[SRH03] SCHMALSTIEG D., REITMAYR G., HESINA G.: Distributed applications for collaborative three-dimensional workspaces. *Presence - Teleoperators and Virtual Environments 12*, 1 (Feb 2003), 52–67.

[TGG04] TOTTER A., GRUND S., GROTE G.: A methodological approach to the evaluation of e-learning technologies. *Proceedings of ED-MEDIA, World Conference on Educational Multimedia, Hypermedia & Telecommunications* (Lugano, Switzerland, 2004), 4392–4397.

[THS*01] TAYLOR R. M., HUDSON T. C., SEEGER A., WEBER H., JULIANO J., HELSER A. T.: VRPN: a device-independent, network-transparent VR peripheral system. In *VRST '01 Proceedings* (New York, NY, USA, 2001), ACM Press, pp. 55–61.

[TN01] TAXEN G., NAEVE A.: Cybermath: Exploring open issues in vr-based learning. *Proceedings SIGGRAPH 2001 Educators Program* (2001), 49–51.

[Tra99] TRAMBEREND H.: Avocado: A Distributed Virtual Reality Framework. In *VR '99: Proceedings of the IEEE Virtual Reality* (Washington, DC, USA, 1999), IEEE Computer Society, p. 14.

[WB92] WINN W., BRICKEN M.: Designing virtual worlds for use in mathematics education: The example of experiential algebra. *Educational Technology 32*, 12 (1992), 12–19.

[Win93] WINN W.: A conceptual basis for educational applications of virtual reality. *Technical Report 93-9* (1993).

[YN04] YEH A., NASON R.: Vrmath: A 3d microworld for learning 3d geometry. In *Proceedings World Conference on Educational Multimedia, Hypermedia & Telecommunications* (Lugano, Switzerland, 2004).

The International Journal of Virtual Reality, 2007, 6(2):43-50

Multiple Head Mounted Displays in Virtual and Augmented Reality Applications

H. Kaufmann and M. Csisinko

Abstract—**With the recent introduction of low cost head mounted displays (HMDs), prices of HMD-based virtual reality setups dropped considerably. In various application areas personal head mounted displays can be utilized for groups of users to deliver different context sensitive information to individual users. We present a hardware setup that allows attaching 12 or more HMDs to a single PC. Finally we demonstrate how a collaborative, educational, augmented reality application is used by six students wearing HMDs on a single PC simultaneously with interactive framerates.**

Index Terms —**Augmented reality, head mounted displays, multi-user applications, virtual reality.**

I. INTRODUCTION

In previous years Virtual or Augmented Reality (VR/AR) applications using head mounted displays (HMDs) supported only a low number of users. Very few applications [21] and setups are described in literature using more than two head mounted displays simultaneously. In a number of application areas multi-user setups with head mounted personal displays would be beneficial though. In contrast to stereoscopic VR multi-display setups used for large audiences (which are typically projection systems nowadays), personal displays allow to deliver specific context or user dependent information [16] to the individual.

HMDs support collaborative VR/AR and allow personalized viewing of data at the same time. In education and training applications [10], for instance, this is of specific interest. In training (e.g. medical training) groups of users with personal displays can study a dataset together with a tutor whereas each participant is still able to choose his own visualization mode of the data (or a subset), depending on the own knowledge and preferences. Context sensitive rendering is also useful in teaching scenarios where a teacher can be enabled to see a solution of a problem whereas (some) students in the same VR/AR environment cannot see it [8]. Entertainment is obviously another interesting application area for the usage of head mounted personal displays on a larger scale [21].

II. BACKGROUND AND RELATED WORK

Considering the image generation process to provide multiple users in VR/AR with individual graphical output, three different types and approaches are known in literature:

- Private or individual screen(s): Each participating user is associated with an individual display and frame buffer(s), where the “personal” image is generated. One HMD per user is a common example of this.
- Frame interleaving: Users share the same set of screens, but a time-based multiplexing/slicing algorithm separates the images of individual users. Rendered images are displayed time-sequentially on the output device. Special hardware (e.g. glasses as in [3]) is needed to synchronize correctly for each user. With an increasing number of participants this method suffers from bad brightness caused by long time slices of blank images after de-multiplexing on a per-user basis [3].
- Screen partitioning: Rendered images are written into a single frame-buffer and displayed on the same output device. They have to be separated by additional optics to allow each user to see his specific portion of the screen only. Typical examples are the Virtual Showcase [2] or the IllusionHole [12].

With a growing number of users (more than four) the frame interleaving as well as the screen partitioning approaches both become impracticable. There are multiple reasons why applications utilizing large numbers of HMDs are rare:

High costs are a major factor why the acquisition of multiple HMDs has been infeasible in previous years. With the introduction of low cost HMDs such as the Emagin Z800 3DVisor or the Icuity DV920 non-see-through HMDs became affordable. Optical-see-through HMDs for AR applications are currently still not available at a low price. Only few large scale HMD setups can be found in literature. One example are the Virtual Theaters [21] built for edutainment purposes and medical training during the phase of the VR hype in the mid 1990’s until the end of the decade. The usage of multiple (up to four) HMDs has also been considered in neuroendoscopy [22] to allow the principal surgeon, as well as other members of the surgical team to view the surgical site concurrently.

In the following we focus on the use of stereo-capable HMDs only. Depending on the stereo mode supported by the HMD, one (field-sequential/quad-buffered) or two (dual-head) graphic card outputs are needed per HMD to generate a stereoscopic image on the device. Therefore the number of available VGA/DVI ports is an important factor when building

---

Manuscript Received on December 20, 2006.

Hannes Kaufmann was involved in projects on computational geometry, geometry education, augmented reality and psychological research in AR/VR. E-Mail: kaufmann@ims.tuwien.ac.at.

Mathis Csisinko is a research assistant and PhD candidate at the Interactive Media Systems group at Vienna University of Technology, Austria. His research interests include 3D user interaction techniques, augmented and virtual reality and especially distribution in VR/AR systems.

multi-HMD setups. On professional PC graphics cards with two output ports, two HMDs supporting quad-buffered stereo can be connected – one to each output. Alternatively expensive high-end workstation hardware which provides multiple outputs supports setups with multiple HMDs too.

Obviously distributed VR/AR frameworks can be used to build large scale HMD setups as well. Cluster-based approaches for multi-display rendering have been studied in the past [6, 15, 17, 20] (and by many more).

As mentioned in related work [17, 20] an advantage of cluster rendering to single workstation/PC solutions is higher computing power per graphics port. With the advent of multi-core processors being now widely available to the public rendering load can be distributed more efficiently within a single PC though (see section Ⅴ).

Maintenance costs and efforts of distributed VR/AR setups with multiple clients are considerably higher than those of single workstation solutions. In addition hardware costs of small cluster-PC solutions are nowadays higher than single PC solutions with multi-core processors assuming comparable performance. There is also no network traffic on physical media when running on a single workstation and in case of a single instance solution there is even no need for networking and data synchronization at all.

Multi-display setups in general have been studied by many research groups in previous years. For instance, Schmalstieg et al. [18] describe a distributed multi-user system combining various display technologies in an augmented reality environment: projectors, HMDs and desktop monitors. Nowadays typical multi-display setups serving large groups of users are based on passive stereo technology. For instance, in VR theaters and similar display environments multiple high resolution projectors are driven by workstations or high end PC hardware (e.g. NVIDIA Quadro Plex). These setups do not allow delivering specific content to a specific viewer in the audience.

Fig. 1. Six users equipped with HMDs working on our lab setup on an education AR application. See Color Plate 13.

We present a hardware setup that supports groups of users with personal head mounted displays (Fig. 1). Initially a single PC solution is proposed which is easy to maintain, though the setup can be extended to PC clusters.

In the next section we describe the hardware setup that was used in our lab to attach six HMDs to a single graphics card. Tracking all users was another challenge that will be mentioned briefly. Section Ⅳ focuses on software specific aspects and describes relevant features of the Studierstube VR/AR framework that we used. In section Ⅴ extensions and other feasible hardware scenarios which become possible with the proposed hardware are outlined. Finally results are presented in section Ⅵ. An educational AR application for geometry education, Construct3D, is used to demonstrate practicability of the proposed setup.

III. HARDWARE SETUP

High-end graphics cards such as those in the NVIDIA Quadro series are equipped with two 400Mhz digital-analog signal converters (RAMDAC chips) which allows them to output very high resolutions (such as 3840x2400) on both DVI/VGA outputs. For example, choosing horizontal span mode on an NVIDIA Quadro card permits selecting a resolution of 4800x600 (2400x600 on each output).

3.1 Matrox TripleHead2Go

A recently introduced hardware device, the Matrox TripleHead2Go plays a key role in our setup. It is somehow the successor of the Matrox DualHead2Go device, which is able to drive a pair of displays, arranged next to each other (in screen space) without the need for a second graphics card or VGA output. Consequently, the TripleHead2Go allows splitting one VGA input signal of large resolution (up to 3840x1024) into three individual VGA signals representing three screens, each with one third of the (horizontal) resolution of the original image. The technical functionality of the device is described in the specifications by Matrox [13] “Inside the TripleHead2Go, the monitor signal from the computer is first converted to digital data using various techniques to ensure the best possible conversion. These techniques include gain compensation to normalize the signal and phase adjustment to properly interpret the analog input signal. After the input is converted to digital data, TripleHead2Go divides the display information into 3 display outputs. The first third of the image is prepared to be sent to a left monitor, the middle third of the image is prepared to be sent to a middle monitor and the rightmost third of the image is prepared to be sent to a right monitor. Three separate CRTCs inside the TripleHead*2Go* are used to generate the three timings and then each one of these three separate images is converted into a separate analog output using Matrox's signature high quality analog output technology”.

Therefore it is possible to split a resolution of 2400x600 into three 800x600 signals. For example three monitors or three HMDs with a resolution of up to 1280x1024 (at 60Hz) can be attached to a TripleHead2Go box. By using two Matrox TripleHead2Go boxes, it is possible to attach 6 HMDs to a single graphics card with two VGA/DVI output ports. Fig. 2 shows a TripleHead2Go with three HMDs attached to it. By doubling the number of PCIe graphics cards in a PC (using two) we can drive up to 12 HMDs on a single workstation. For enthusiasts there is still the possibility to go even further by getting a mainboard with more fast PCIe slots (e.g. from Gainward featuring 4 PCIe slots) or to plug in additional PCI graphics cards in order to support more HMDs. For instance, for parallel computation on the GPU a PC has been equipped with 6 PCI graphic cards and 1 AGP card recently [5].

Fig. 2. One input and 3 output connectors. Three HMDs (two Emagin Z800 and one Sony Glasstron) are attached to the Matrox TripleHead2Go. In our 6-user setup three more Sony Glasstron HMDs were attached to another TripleHead2Go box.

In our setup we use two Emagin Z800 HMDs and four optical-see-through Sony Glasstron HMDs, all supporting a resolution of 800x600 at 60Hz (32bpp) and stereoscopic viewing (quad-buffered stereo). Two Matrox TripleHead2Go devices receive the input signal (2400x600 each) from an NVIDIA Quadro 4400. This graphic card supports quad-buffered stereo even at a high resolution of 2400x600 on both DVI outputs. In order to enhance immersion in the virtual world, stereoscopic viewing is of major importance. It was our goal from the beginning to enable stereo viewing for all HMDs in our setup. The Sony Glasstron HMD natively displays stereo when receiving the 800x600 input signal after the larger 2400x600 signal has been processed/split by the Matrox TripleHead2Go.

The stereo mode of the Emagin HMDs was originally tailored towards NVIDIA graphic cards featuring automatic and correct assignment of stereoscopic images to the left and right eye. Unfortunately, after the signal has been processed by the TripleHead2Go device, some information about this image-eye assignment is lost and the Emagin HMDs do not automatically switch to stereo mode anymore. Although we could not get clear insight into this issue, it seems that an undocumented signal property is responsible for this behaviour. Obviously, the TripleHead2Go is not able to deal with this VGA signal extension, which is probably of rather proprietary kind.

Help was provided by Emagin a short time ago, when they offered the possibility to circumvent the constraints regarding switching to stereoscopic mode. After releasing an updated version of the Z800 firmware and extending the programming interface by a method to manually switch into stereo mode it is easily possible to force the Emagin Z800 HMDs into stereoscopic mode. After carrying out the proper command, the Z800 displays perfect stereo too. We developed a simple, tiny software tool to enable and disable stereo mode manually. It utilizes the Emagin SDK and communicates with the Z800 control box via USB.

The large number of HMDs - given the fact that HMDs have to be connected by cables - requires proper mounting of the wiring. For example one possibility is to mount control boxes plus cables on the ceiling to omit users stumbling over cables lying on the floor and to minimize cable crossing occurrences. Considering typical group applications, where users usually do not move around to such an extent, the problem of cables is not crucial. In our applications no observable issues occurred.

3.2 *Tracking*

In order to provide groups of users with an immersive experience exactly tailored to their point of view, all users (respectively their HMDs) must be tracked in space. Tracking a large number of users in such an environment is a problem on its own. It is obvious that any wireless form of tracking is preferred over a wired alternative because of the number of cables, limited freedom of movement leading to a rather tethered experience and other practical problems that would arise due to cables. Because of occlusions optical tracking is usually not ideal in environments with many users. The big advantage in our setup is that we are mainly interested in tracking the displays of all users which are attached to their heads. Markers/targets that are used for optical tracking are easily visible by cameras mounted to the ceiling and hardly occluded by any other objects. We used our own low-cost optical tracking system [14] which is very similar to commercial system such as the ones from ART or VICON but available for 1/10th of the costs. In terms of tracking our system is cable-free and therefore the total amount of cables in the setup is equal to the number of HMDs.

In addition to tracking displays, it is possible to track supplementary interaction devices of 1 or 2 users in a centered interaction area by adding extra cameras directly above that hotspot. We provide user interfaces (a pen and tablet) for 1-2 users, who can collaborate simultaneously while others watch. In order to break a strict role assignment, actively collaborating users and passive watchers can exchange user interfaces on the fly. That way passive observers become active participants and vice-versa. For educational purposes (teacher-students), training or demonstrations, where it is not feasible to equip each user with the same full set of interface devices, this is an ideal collaboration scenario.

A side effect of AR/VR environments, which is usually seen as deficiency, turns out to be advantageous in multi-user environments, especially if the number of co-located users increases: Without further processing virtual scenes are always displayed in (see-through) HMDs “on top” of the real environment. As long as users are not rendered or represented by avatars in the virtual world there will be no occlusions of the virtual content. In other words: Users can never occlude a virtual scene. No matter how many users are located in a room, the whole VR scene, all virtual objects, can be seen by each user regardless where the colleagues are located. The focus of the user assembly can not get lost by line-of-sight issues. This also proves to be important in educational scenarios, during marketing presentations or demonstrations where people often suffer from not being able to fully see at each instant what is being shown.

IV. SOFTWARE IMPLEMENTATION

From a software perspective the problem remains of how to generate graphical output fast enough for such a large number of displays on a single PC. Thanks to the advances of multi-core processors, server mainboards with 2-4 CPUs, supporting quad-core processors are available these days. Even in simple VR environments, using the latest high-end graphics cards available, there is the absolute need to split the computational load. The classical approaches are either to do multi-threaded rendering or to run multiple instances of a distributed VR environment.

Our Studierstube [19] framework supports both modes. Studierstube is a collaborative multi-user VR/AR software development toolkit. It uses the scene graph library Open Inventor [24] with additional distribution capabilities (Distributed Open Inventor (DIV) [6]) and supports rendering on various VR/AR display devices as well as acquiring data input from 3D tracking devices. For computer graphics generation we utilize Coin, which is based on SGI's original Open Inventor implementation. The render traversal of the scene graph is single-threaded as in the original implementation; therefore it can not benefit from multi-core processor architectures directly. In section Ⅴ we will illustrate how load balancing can be achieved by running several instances simultaneously with enabled distribution features though.

With DIV features enabled, the whole or some parts of the scene graph can be shared among multiple instances running on different machines connected to a network. Network communication on standard Internet protocols (TCP/IP and multicast UDP) keeps the distributed scene graph on each participating host up-to-date. Studierstube applications are encapsulated in the scene graph (represented as nodes) and are therefore synchronized as well.

We are running multiple instances of the Studierstube application on the same PC and distribute data between single instances. Although it is possible to share the scene graph as a whole, it is often more efficient to distribute a more compact representation of high level data especially in long distance distribution as it was shown in [9]. Network transparency features also allow distribution on a single host. As a convenient side effect of this distribution strategy, a system composed of multiple rendering instances balances work load on modern multi-core processor architectures.

V. DISTRIBUTED MULTI-USER SCENARIOS

As already mentioned in section Ⅰ, there are several options of how to implement multi-user scenarios with multiple HMDs:

- *Single instance:* A single instance application used in conjunction with Matrox TripleHead2Go devices opens several windows, each of them being the render target of a HMD. With this configuration each graphics card output can drive up to 3 HMDs. Balancing work load between multiple processors has to be done manually in the single application instance.
- *Single host with multiple instances:* Multiple instances of an application allow implicit load balancing on multi-core CPUs. Each instance is able to render output for one or more HMDs, but distribution and synchronization with other instances has to take place. Again, using Matrox TripleHead2Go devices triples the number of possible HMDs.
- *Multiple hosts:* In true distributed systems multiple hosts are involved, but synchronization between these hosts has to be physically communicated over network lines. The distribution of our Studierstube application was demonstrated in [9]. In order to reduce the number of hosts, making use of TripleHead2Go devices typically allows running 6 HMDs per machine instead of just 2. Using TripleHead2Go splitting devices on multiple hosts, this configuration is able to run a huge number of HMDs, while keeping the number of machines low. Groups of HMDs (typically consisting of 6 display devices) share system resources on a particular host.

VI. EDUCATIONAL APPLICATION

In order to use VR/AR applications in realistic, educational settings, a large group of students must be able to participate either actively or passively in the activities taught in VR/AR.

6.1 Construct3D

Our work is based on the educational AR application Construct3D [8, 10, 11]. This system deploys AR to provide a natural setting for face-to-face collaboration of teachers and students. The main advantage of using AR is that students actually see three dimensional objects. With traditional methods students have to rely on 2D sketching or calculating and constructing objects using pen and paper or CAD software. By working directly in 3D space, complex spatial problems and spatial relationships may be comprehended better and faster than with traditional methods.

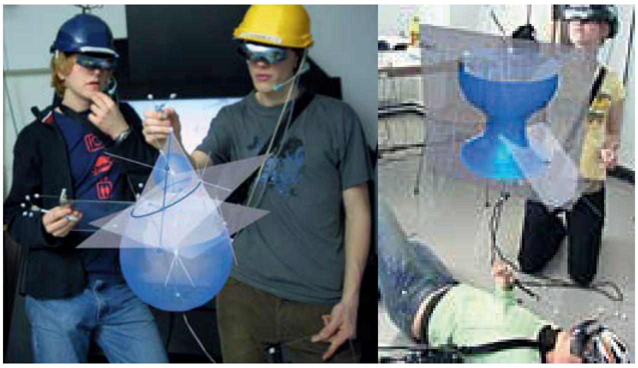


Fig. 3. Students working with Construct3D. See Color Plate 14.

According to pedagogical theories [23], collaboration is a fundamental social process that supports learners' development of capabilities. In a collaborative AR environment multiple users may access a shared space populated by virtual objects, while remaining grounded in the real world. This approach is particularly powerful for educational purposes when users are co-located and can use natural means of communication (speech, gestures etc.), but can also be mixed successfully with immersive VR [1] or remote collaboration [7]. Supporting natural collaboration in the mathematics domain opens new possibilities to the educational process.

Direct manipulation and dynamic interaction with virtual 3D objects using tangible interaction devices are key features of Construct3D. In our standard lab setup users are wearing an optical-see-through head mounted display; a pen and a panel are used for direct interaction in 3D space (Fig. 3). Head, pen and panel are fully tracked in 3D which allows users to walk around objects and to view them from different perspectives.

For students, teachers and spectators it must be possible to distinguish between the work done by each single user. This is especially important in distributed, remote teaching scenarios, but also in co-located setups. Therefore user information is encoded in object colors. Each user is working within a distinct color scheme: blue, orange, green and red. This color coding is consistently visible on the panel (representing the menu system) and on all geometric objects, indicating the user being responsible for object creation.

Complex constructions involving many objects and work steps can quickly lead to a loss of overview. Therefore, we enable additional structuring by introducing layers. A layer is a simple user-controlled grouping mechanism for geometric objects. Using layers, visual complexity can be managed when teaching with Construct3D. A teacher can switch off irrelevant parts of a construction, or prepare future steps and alternatives as invisible hidden layers to guide students through complicated steps. In conjunction with multi-user operation this feature is particularly powerful. Each user has a personal display for which visibility of layers can be controlled independently [10]. Context sensitive scene graph traversal [16] is used to generate private views of public shared data.

Construct3D has been evaluated in more than 500 teaching lessons with more than 100 high school students and is under development since 2000.

6.2 Teaching Scenarios

A typical application scenario may include several groups of students consisting of six persons. If the total number of students is low, all students can be co-located in a room - preferably in a special lab, permanently equipped with the required hardware and with professional wiring solutions. They collaborate and communicate directly and share a set of interaction devices.

In order to serve a larger number of students there is the possibility to locate smaller groups of users in different labs. Collaboration between these groups has to take place by making use of distribution features to share a common workspace. Within each group members are aware of each other, as the other's physical presence is guaranteed. Supplementary communication and coordination are possible in a very natural way. For communication between the distributed groups, voice chat is suggested. All interaction devices (e.g. pen and panel) have 3D representations in the virtual world and can therefore be shared as part of the distributed VR/AR environment. Their movements can be observed by all participants if desired to see and better understand what other groups are currently working on.

The latter grouping scenario prevents a room from getting too crowded by participants.

6.3 Teaching Content

In our lab setup with six users we used two different teaching examples to study the hardware and software issues that might arise in such a multi-user environment. In both cases a tutor presented the examples and asked the other users to participate, requesting feedback and interaction like in a teacher-student scenario. We will briefly describe the teaching content in the following.

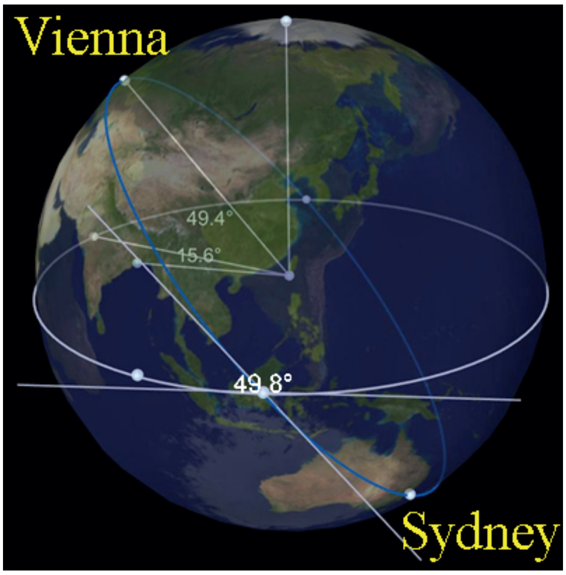


Fig. 4. Finding the shortest flight connection between two points on earth.

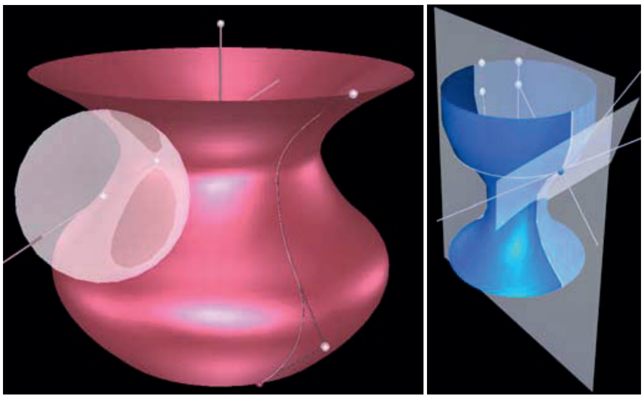


Fig. 5. Surfaces of Revolution. Left: Spherical cutter intersects the given surface and is inappropriate for milling. See Color Plate 15.

Flight Route: The task is to construct the shortest flight route from Vienna to Sydney, which in ideal case is an orthodrome. The given virtual scene in Construct3D (Fig. 4) shows a model of earth (with texture) to help pupils find the correct places on earth and to immerse them further into the problem. Fig. 6 shows this content being used in the six-user setup (image composition).

Milling Cutter: Surfaces of revolution are omnipresent in everyday life. This task should be an incitement to learn more about their geometric properties. The perpendicular axis of a surface of revolution is given. Students have to rotate an arbitrary B-Spline curve around this axis. The result is a surface of revolution Φ (Fig.5). Construct3D allows to modify the original B-Spline curve dynamically, which results in the modification of the surface of revolution in real time, too. This is a unique feature of Construct3D.

In addition a spherical milling cutter is given (transparent sphere in Fig.5 left) whose radius can be modified. The task is to find a spherical cutter which is suitable to mill the given surface. Students soon notice that if the radius of the spherical cutter is too big, it will intersect with the surface and therefore cut off too much. The example provides an excellent opportunity to discuss curvature of surfaces.

VII. RESULTS

Our results were generated on PCs running Microsoft Windows and various NVIDIA graphics cards were used with the latest drivers.

7.1 Low cost setup

A single-core AMD Athlon 64 FX-57 processor provided sufficient computing power in our setup. Using a publicly available free software tool (RivaTuner), we reset the PCI Device-ID of an ordinary Geforce 6800 GT graphics card in order to enable high end features of the card, a process which is publicly known and frequently used. Due to this change, the low cost consumer card becomes equivalent to an NVIDIA Quadro FX 4000 card, which allows stereoscopic rendering on both DVI outputs.

We used low cost HMDs - Emagin Z800 3DVisors - in our setup in addition to optical see-through Sony Glasstron HMDs that were already available in our lab. Stereoscopic rendering on both outputs was verified with this configuration and initially a dual user setup was built. Since the card is equipped with two RAMDACs, we set up a horizontal span resolution of 4800x600. A Matrox TripleHead2Go device was attached to each output, supporting six users with HMDs on a Geforce 6800 graphics card (Fig. 6). In dual view display mode two outputs with a resolution of 2400x600 each are also possible. This mode offers more flexibility in running different resolutions on each VGA/DVI port.

Our approach is not limited to a specific graphics card model or brand. Recent models, such as the Geforce 8800 support high resolutions too and are also equipped with two RAMDACs (two dual-link DVI outputs supporting two 2560x1600 resolution displays).

In case a newer Geforce model cannot be modified to support quad-buffered stereo, the NVIDIA stereo driver can be used to achieve stereo rendering on one output on low-cost consumer cards. Since the NVIDIA stereo driver was designed to offer stereo rendering in games, *only* applications running in full screen mode are supported yet. There is no windowed stereo support with the NVIDIA stereo driver. In case a higher screen resolution is split in various smaller parts (as done by the TripleHead2Go) to provide multiple users with stereo graphics, the software which renders the scene must be adapted to generate a full screen stereo window. With the driver constraint regarding stereoscopic mode it is not possible to render stereo images in smaller windows (e.g. three windows with 800x600 as we do in case of the TripleHead2Go) for each partition of the screen. Instead only one full screen window (e.g. 2400x600) can be used as stereo render target and all application output must be drawn accordingly for each user in his visible section of this large window. This can only be done if all sections of the output image are rendered by the same instance of the application; therefore no stereo setup “single host with multiple instances” (as described in section Ⅴ) is possible in such a low cost scenario with consumer graphics cards.

Fig. 6. Six students working on the flight route task in Construct3D. See Color Plate 16.

In our test we used a single instance setup with six users on a single-core Athlon 64 FX-57 PC. Even in this configuration it is possible to achieve interactive frame rates. During the flight route task (Fig. 6) the frame rate was about 15 fps which allows real time interaction. All HMDs were connected to one graphics card (exactly as described above) and stereo 3D graphics were generated for all six users.

VIII. CONCLUSION

Finally we got one step closer to our aim to provide a low cost setup for immersive VR/AR systems. Our recent work was focused on single or dual user educational settings, but in real classroom use groups of students must be able to follow the teacher. We demonstrated how to use low cost components such as the Emagin Z800 3DVisors, which is affordable for the masses. In addition an affordable optical tracking system was chosen. We used the Matrox TripleHead2Go devices in an uncommon, yet undocumented, but still creative and efficient manner: The ability to drive up to 6 HMDs with a single graphics card plugged into one computer helps saving money and reduces maintenance effort. Thinking of quad-core and 8-core PCs, such a configuration is most suitable for efficiently utilizing current high performance graphics and processing hardware without wasting too many resources, which might remain unused otherwise.

Apart from keeping the expenses low, this system configuration has other implications. As the limit is raised, where a distributed system with multiple hosts becomes practically indispensable, unnecessary system complexity can be effectively prevented.

IX. FUTURE WORK

Improvements are obviously possible by adapting scene-graph toolkits such as Open Inventor to multi-threaded rendering (this is already included in the commercial version of Mercury Open Inventor). Single instance applications without the need for inter-host synchronization should be optimized to take advantage over modern multi-core CPUs by running several processes or threads. Synchronization features of these single instance processes can be implemented without the need for networking overhead.

For future extensions, the TripleHead2Go device might be potentially interesting in a number of VR/AR applications. For example a low cost CAVE such as the DAVE [4] (Definitely Affordable Virtual Environment) could be built using a single PC. Three walls with passive stereo projection (2 projectors per wall) and a resolution of up to 1280x1024 could be served by the six outputs that can be generated with the help of the TripleHead2Go. Synchronization issues would be easier to handle, maintenance and hardware costs were lower. Even high-end notebooks with two VGA/DVI outputs (which are available too), could be used to drive such a projection environment.

ACKNOWLEDGMENT

This work was funded in part by the Austrian Science Fund FWF project P19265.

REFERENCES

[1] M. Billinghurst, H. Kato and I. Poupyrev. The MagicBook-Moving seamlessly between reality and virtuality, *IEEE Computer Graphics and Applications*, vol. 21, pp. 6-8, May-June 2001.

[2] O. Bimber, B. Fröhlich, D. Schmalstieg and L. M. Encarnacao. The Virtual Showcase, *IEEE Computer Graphics and Applications*, vol. 21, no. 6, pp. 48 - 55, 2001.

[3] J. Ehnes, C. Knopfle and M. Unbescheiden. The Pen and Paper Paradigm Supporting Multiple Users on the Virtual Table, Proceedings of *IEEE Virtual Reality Conference 2001 (VR'01)*, pp. 157-164, 2001.

[4] D. Fellner, S. Havemann and A. Hopp. DAVE – Eine neue Technologie zur preiswerten und hochqualitativen immersiven 3D-Darstellung, *Institute of Computer Graphics*, University of Technology Braunschweig, Technical Report TUBS-CG-2003-08, 2003.

[5] J. Fung, S. Mann and C. Aimone. OpenVIDIA: Parallel GPU Computer Vision, Proceedings of *the ACM Multimedia 2005*, pp. 849-852, 2005.

[6] G. Hesina. Distributed collaborative augmented reality, in PhD Thesis: Vienna University of Technology, 2001, pp. 102.

[7] T. Höllerer, S. Feiner, T. Terauchi, G. Rashid and D. Hallaway. Exploring MARS: Developing indoor and outdoor user interfaces to a mobile augmented reality system, *Computers & Graphics*, vol. 23, no. 6, pp. 779-785, 1999.

[8] H. Kaufmann. Geometry Education with Augmented Reality, in Ph.D. Thesis: Vienna University of Technology, 2004, pp. 179.

[9] H. Kaufmann, M. Csisinko and A. Totter. Long Distance Distribution of Educational Augmented Reality Applications, Proceedings of *Eurographics 2006 (Educational Papers)*, pp. 23-33, 2006.

[10] H. Kaufmann and D. Schmalstieg. Mathematics and geometry education with collaborative augmented reality, *Computers & Graphics*, vol. 27, pp. 339-345, Jun 2003.

[11] H. Kaufmann and D. Schmalstieg. Designing Immersive Virtual Reality for Geometry Education, in Proceedings of *IEEE Virtual Reality Conference 2006*, Alexandria, Virginia, USA, 2006, pp. 51-58.

[12] Y. Kitamura, T. Konishi, S. Yamamoto and F. Kishino. Interactive stereoscopic display for three or more users, Proceedings of *the 28th Annual Conference on Computer graphics and Interactive Techniques*, pp. 231 - 240, 2001.

[13] Matrox. TripleHead2Go technology brief, http: // www. matrox. com/ graphics / en / gxm / products / th2go / tech _ brief. php, last visited: Dec. 18th, 2006.

[14] M. Mehling. Implementation of a Low Cost Marker Based Infrared Optical Tracking System, in *Master Thesis: Fachhochschule Stuttgart & Vienna University of Technology*, 2006, pp. 101.

[15] E. Olson. Cluster Juggler: PC Cluster Virtual Reality, in *Master Thesis: Iowa State University*, 2002.

[16] G. Reitmayr and D. Schmalstieg. Flexible Parametrization of Scene Graphs, Proceedings of *IEEE Virtual Reality Conference 2005 (VR'05)*, pp. 51-58, 2005.

[17] B. Schaeffer and C. Goudeseune. Syzygy: Native PC Cluster VR, Proceedings of *IEEE Virtual Reality 2003*, pp. 15-22, 2003.

[18] D. Schmalstieg, A. Fuhrmann and G. Hesina. Bridging Multiple User Interface Dimensions with Augmented Reality, Proceedings of *ISAR 2000*, no. 5-6, pp. 159-164, October 2000.

[19] D. Schmalstieg, A. Fuhrmann, G. Hesina, Z. S. Szalavári, L. M. Encarnacao, M. Gervautz and W. Purgathofer. The Studierstube augmented reality project, *Presence-Teleoperators and Virtual Environments*, vol. 11, no. 1, pp. 33-54, February 2002.

[20] O. Staadt, J. Walker, C. Nuber and B. Hamann. A Survey and Performance Analysis of Software Platforms for Interactive Cluster-Based Multi-Screen Rendering, Proceedings of *the Workshop on Virtual Environments 2003*, pp. 261-270, 2003.

[21] Corp. StrayLight. 'Incredible Voyage' At International Stroke Conference, in http://www.strayvr.com/ISC2004.htm, 2004. Last visited Dec. 19th, 2006.

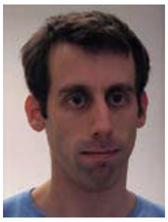
[22] E. J. van Lindert, J. A. Grotenhuis and T. Beems. The use of a head-mounted display for visualization in neuroendoscopy, *Computer Aided Surgery*, vol. 9, no. 6, pp. 251-256, 2004.

[23] L. S. Vygotsky, Thought and language. (Original work published 1934). Cambridge, MA: MIT Press, 1962.

[24] J. Wernecke. The Inventor Mentor: Programming Object-Oriented 3D Graphics with Open Inventor, 2nd ed.: Addison-Wesley, 1993.

Hannes Kaufmann finished his PhD thesis in 2004 on "Geometry Education with Augmented Reality". After his postdoc work in the EU-IST project Lab@Future he got an assistant professor position at the Interactive Media Systems Group, Institute of Software Technology and Interactive Systems at Vienna University of Technology in 2005. He is now leading the VR/AR group at that institute at Vienna University of Technology.

He is the main developer of Construct3D which is being used in a number of national and international research projects and initiated development of the optical tracking system at the Interactive Media Systems Group. He was involved in projects on computational geometry, geometry education, augmented reality and psychological research in AR/VR. His research interests include augmented and virtual reality, education in AR/VR, optical tracking, psychological topics in AR/VR (Cyberpsychology), user interface design for collaborative AR and educational applications, AR/VR and CAD Integration, mobile and ubiquitous computing and computational geometry.



Mathis Csisinko is a research assistant and PhD candidate at the Interactive Media Systems group at Vienna University of Technology, Austria. His research interests include 3D user interaction techniques, augmented and virtual reality and especially distribution in VR/AR systems. After finishing his master thesis on “Long Distance Distribution of Virtual and Augmented Reality Applications” where he also worked on Construct3D he received his MS in computer science in 2006 from Vienna University of Technology, Austria.

Simulating Educational Physical Experiments in Augmented Reality

Hannes Kaufmann* Bernd Meyer†

Interactive Media Systems Group
Institute of Software Technology and Interactive Systems
Vienna University of Technology

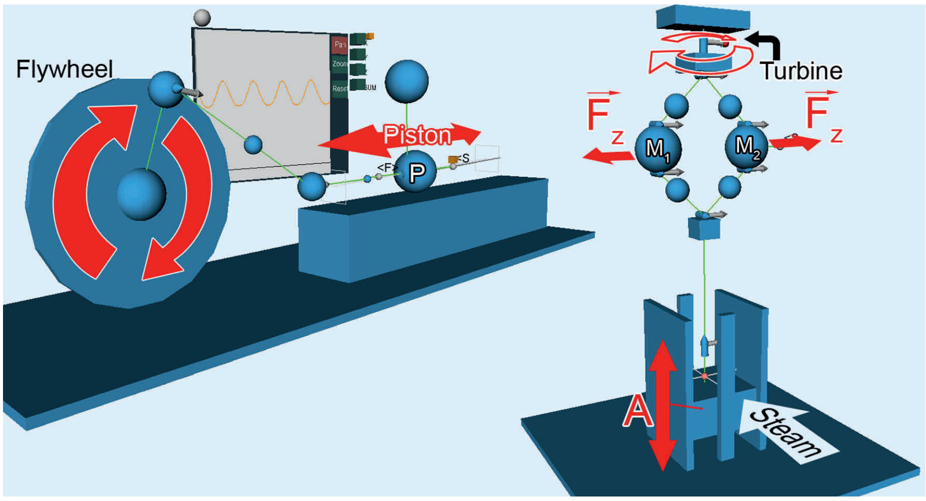


Figure 1: *Left: An experiment simulating the motion of a car crankshaft. Right: A centrifugal regulator in PhysicsPlayground.*

Abstract

We present an augmented reality application for mechanics education. It utilizes a recent physics engine developed for the PC gaming market to simulate physical experiments in the domain of mechanics in real time. Students are enabled to actively build own experiments and study them in a three-dimensional virtual world. A variety of tools are provided to analyze forces, mass, paths and other properties of objects before, during and after experiments. Innovative teaching content is presented that exploits the strengths of our immersive virtual environment. PhysicsPlayground serves as an example of how current technologies can be combined to deliver a new quality in physics education.

CR Categories: K.3.1 [Computers and Education]: Computer Uses in Education—Collaborative learning; H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, augmented, and virtual realities

Keywords: physics education, mechanics, augmented reality, virtual reality

1 Motivation

Classical mechanics [Goldstein et al. 2001; Lifshitz and Landau 1982] is the oldest discipline in the field of physics. It describes the common motion of objects that humans perceive in everyday life. The three fundamental laws of motion which were formulated by Isaac Newton (1642 - 1727) are still of high importance and concepts such as force, velocity and acceleration are traditionally taught in schools. Our knowledge about physics changed throughout centuries but most books on classical physics begin with a chapter on mechanics [Gerthsen and Vogel 1993; Körner et al. 1988; Pohl 1931].

Sometimes students have problems understanding the physical concepts of mechanics. It might be that physics in the traditional sense is sometimes taught in an abstract, jejune way and is therefore not very motivating for students. The result is that theoretical models behind physical phenomena are often misunderstood. It is not necessary to stress that conveying these concepts correctly is of utmost importance since they are fundamental to physics. Many theoretical models are based on Newton's laws of motion.

Therefore the authors developed an educational augmented reality (AR) application called PhysicsPlayground [Meyer 2007] that is supposed to support students in studying and finally understanding the concepts of mechanics (Figure 2). In this three-dimensional virtual environment learners and educators are able to freely create physical experiments that can be simulated in real time. The hardware setup is described in section 3. Features of the application are explained in section 4. We consider the analyzing functionality (section 4.3) an important strength of a virtual laboratory like PhysicsPlayground. It offers possibilities that are far superior to what can be done in a real physics lab. A direct connection between simulated reality and physical data is supposed to help students grasp the theoretical basics of mechanics. To establish a direct link to students pre-knowledge, physical data - that can be acquired through the application - is presented in a way so that it closely relates to formulas and equations of school mechanics. In section 5 we present teaching content that demonstrates the power and added benefit of our educational AR/VR software. We will also elaborate on correctness and robustness of physical simulations for educational purposes.

---

*e-mail: kaufmann@ims.tuwien.ac.at
†e-mail: meyer@ims.tuwien.ac.at

SIGGRAPH Asia 2008, Singapore, December 10–13, 2008.
ISBN 978-1-60558-388-4/08/0012

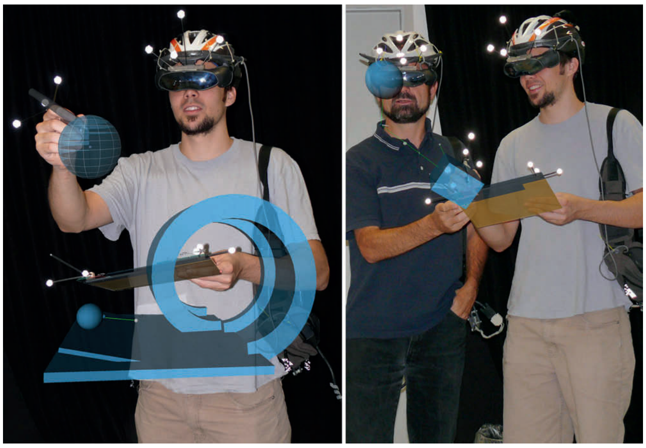


Figure 2: *Left: A student working with PhysicsPlayground. In the right hand he holds a wireless pen (used as input device), in the left hand the PIP. Right: Collaborative work in PhysicsPlayground.*

2 Related Work

In the following a few immersive virtual environments for mathematics and science education will be presented briefly. They all demonstrate unique advantages of using AR/VR for education and give insights to what the technology can offer.

2.1 Mathematics and Science Education in AR/VR

Water on Tap [Byrne 1996] is one of the earliest immersive virtual environments for chemistry education. It is a chemistry world which allows to build molecules. Therefore electrons have to be placed in orbits around the kernel of an atom. The spin of the electrons and other properties can be selected. ScienceSpace [Dede et al. 1996] is a collection of immersive virtual worlds consisting of Newtonworld, MaxwellWorld and PaulingWorld. They have been developed to study the strengths and limits of virtual reality for science education. NewtonWorld provides an environment for investigating kinematics and dynamics of one-dimensional motion. MaxwellWorld supports the exploration of electrostatics, up to the concept of Gauss' Law, and PaulingWorld enables the study of molecular structures via a variety of representations. Formative evaluation studies of these virtual worlds have been conducted with respect to usability and learn-ability. These studies report on learners' engagement, surprise and understanding. Limitations and discomfort caused by the head-mounted displays hindered usability and learning.

A technically advanced project for mathematics education is CyberMath [Taxen and Naeve 2001]. CyberMath is an avatar-based shared virtual environment aimed at improving mathematics education. It is suitable for exploring and teaching mathematics in situations where both teacher and students are co-present or physically separated. CyberMath is built like a museum with a virtual lecture hall in its center. Special care has been taken to design the environment as inviting as possible. Virtual mathematical objects can be manipulated and discussed in a realistic way. CyberMath has been tested for distributed learning in CAVEs but is also running as a desktop VR application with no support of immersive displays. Two usability studies of the DIVE version of CyberMath have been performed with 15 participants in total. Teacher and students worked in two separate locations. The studies provided useful feedback for further improvement of the application and for increasing robustness of the distributed environment. The developers believe that CyberMath in a networked CAVE environment holds the potential to provide a high-tech front end which is interesting enough to create public interest and contribute to a more positive attitude towards mathematics - especially among young people. It could also provide a useful platform for developing various forms of interactive problem solving games with an emphasis on cooperative problem solving skills.

Construct3D [Kaufmann and Schmalstieg 2003] is a three-dimensional dynamic geometry construction tool that can be used in high school and university education. It uses augmented reality to provide a natural setting for face-to-face collaboration of teachers and students. The main advantage of using VR and AR is that students actually see three dimensional objects which they until now had to calculate and construct with traditional (mostly pen and paper) methods. By working directly in 3D space, complex spatial problems and spatial relationships may be comprehended better and faster than with traditional methods. Three usability studies with more than 100 students have been conducted since 2000 [Kaufmann and Dünser 2007] and guidelines have been formulated regarding how to design AR applications for (geometry) education [Kaufmann and Schmalstieg 2006]. Although usability of Construct3D is high and teachers as well as students are highly motivated to use the application, practical usage in schools is hindered by hardware costs, support of a low number of users and technical complexity of the whole setup (requiring dedicated personnel for maintenance).

The SMILE project [Adamo-Villani et al. 2006; Adamo-Villani and Wright 2007] is an immersive learning game that employs a fantasy 3D virtual environment to engage deaf and hearing children in math and science-based educational tasks. SMILE is one of the first bilingual immersive virtual learning environments for deaf and hearing students combining key elements of successful computer games, emotionally appealing graphics, and realistic real-time 3D signing, with goal-oriented, standards-based learning activities that are grounded in research on effective pedagogy.

2.2 Pedagogic Background

Constructivist theory provides a valid and reliable basis for a theory of learning in virtual environments [Winn 1993]. Constructionism is based on constructivism and promotes that learning takes place when students can construct things. As Mantovani [2003] points out, the basic assumption that the learning process will take place naturally through the simple exploration and discovery of the virtual environment should be reviewed. Despite the value of exploratory learning, when the knowledge context is too unstructured, the learning process can become difficult. The learning process should support building of conceptual models that are both consistent with what students already understand and with new content. In order to ensure successful adaptation of old knowledge to new experience, flexible learning environments should be provided. One possibility is to integrate known types of information other than a 3D representation (such as audio and text annotations, images etc.). In our case we included the analyzer (section 4.3) as a tool similar to an oscilloscope. In addition our environment supports collaboration and therefore learning as an active, social process. Finally, VR environments can be tailored to individual learning and performance styles. Our examples in section 5 allow experimentation and support constructionism.

3 Working Environment

The implementation of PhysicsPlayground is based on the Studierstube AR framework [Schmalstieg et al. 2002]. The standard hardware setup consist of an head-mounted display (HMD), a wireless pen and the so called personal interaction panel (PIP) [Szalavari and Gervautz 1997]. Pen and PIP are used to fully control the application in 3D space (Figure 3). The overall hardware setup is equivalent to the one used in Construct3D [Kaufmann and Schmalstieg 2003], an augmented reality application for geometry education, since both applications were developed in the same lab. The setup supports two users, allows direct manipulation, free roaming around virtual objects and is favored by students. Since we're using Sony Glasstron see-through head mounted displays students can see each other and their interaction devices. Each of the hardware components is tracked by an iotracker [Pintaric and Kaufmann 2007] infrared-optical tracking system in six degrees of freedom. This gives the user freedom in motion some students lie down on the floor to look at objects from below or step on a chair - and simultaneously enhances the feeling of immersion and fun. PhysicsPlayground runs on a standard desktop setup as well but is more intuitive and more impressive to use in an immersive environment.

Figure 3: *Each user is equipped with input devices pen and PIP and a head mounted display when interacting with PhysicsPlayground.*

A wireless pen with one clickable button is used to select and drag objects and to control the application. Most of the application functionality can be invoked through the PIP, a touchable plexiglass sheet providing haptic feedback. When looking through the head mounted display control elements are displayed on the PIP which the user can click in order to trigger different actions (Figure 4). We intentionally designed it to be similar to a 2D GUI in order to provide students with a familiar type of interface. Studying usability of educational AR applications using Construct3D, the main author showed [Kaufmann and Dünser 2007] that a 2D type of menu interface on the PIP is perceived as highly usable by students, provided that some basic guidelines are followed [Kaufmann and Schmalstieg 2006].

In Figure 4 several menu widgets are shown. Many of them have 3D icons placed on top which are animated when moving the pen over them. This is self-explanatory and clarifies their specific functionality. The GUI is a standalone component which can flexibly be used by other VR applications. An additional layout manager system like it is standard for 2D GUIs is not available at present.

4 Features and Design

In [Seugling and Rölin 2006] a number of widely used physics engines have been evaluated in detail. Amongst them was the commercial engine PhysX formerly known as NovodeX/AGEIA, now NVIDIA [AGEIA 2008] and the open source engines Newton Game Dynamics and Open Dynamics Engine. In summary the evaluation showed that PhysX is the most stable, feature rich, precise and fastest engine available at the moment. Therefore the PhysX API was chosen as the base of PhysicsPlayground. PhysX (now owned by NVIDIA) is available for free for commercial and non-commercial purposes, supports Windows and Linux and the PS3 platform and is written in C++. Objects which can be used by PhysX in simulations are rigid bodies, soft bodies, deformable objects, fluids and cloth. Related to PhysicsPlayground robust simulation of rigid bodies is of highest importance. A rigid body in its original definition is an object with fixed geometrical characteristics [Hecker 1996]. At each point in time points within the rigid body stay fixed to each other. Additional dynamics provide the rigid body with the ability of translational and rotational motion. In PhysX rigid bodies and rigid body dynamics are referred to as shapes and actors. In the following the features of PhysicsPlayground will be described briefly.

4.1 3D Shapes

Because the application is intended to simulate physical school experiments it must be possible to integrate virtual models of real life objects. Therefore PhysicsPlayground allows users to create, destroy, modify and interact with different kinds of shapes. Each shape can be either static or dynamic and represents a solid object, enabled for collision detection during simulation. Static shapes stay in place while the physical simulation is running. They have an infinite mass. Dynamic shapes behave like real world objects. They have an adjustable mass, a center of mass, a surface friction and are affected by force during simulation.

The appearance of shapes can have various forms. Shapes can be simple objects like boxes, spheres, cylinders or more complex ones such as a looping (Figure 2 left) or a car. A number of primitive shapes are integrated into PhysicsPlayground by default. More complex objects can be defined by the user or loaded on demand. Position, appearance and parameters of all shapes, for example the width of a box, can be configured by the user after object creation. Modification takes place through the PIP or by direct manipulation of the shapes with the pen. Finally shapes can be grouped into larger shapes making it possible to create advanced objects.

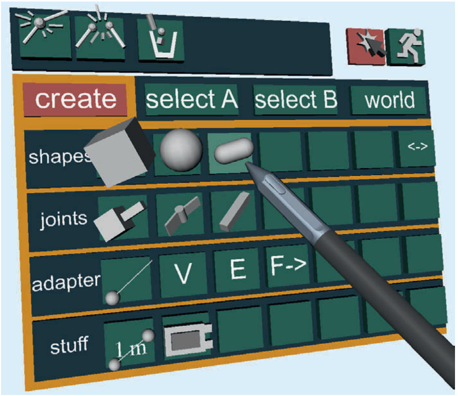


Figure 4: *An example menu of PhysicsPlayground on the PIP. The virtual representation of the pen is shown as well.*

4.2 Joints

Rather then simply grouping shapes our application allows the definition of mechanical linkages between different shapes (Figure 4). Currently implemented are a revolute joint, a prismatic joint and a stiff joint connection. In case of a revolute joint a motor can be added. It causes its attached shape to spin with constant radial velocity. With such connections at hand it is possible to create more complex physical scenes. A centrifugal regulator for example can be built and simulated in PhysicsPlayground this way (e.g. Figure 1 right). Modification and positioning of each joint is either done utilizing the PIP or directly by selecting a joint in 3D space.

4.3 The Analyzer

During every physical experiment certain magnitudes and properties of the simulated objects might be of interest. To be able to compare a virtual simulation with a real one it is important to offer possibilities to extract physical properties of objects before, during and after run time of the simulation. For educational purposes these data can be used to learn about the theoretical background or to confirm results in a traditional way for instance by using appropriate formulas. Physical magnitudes with a high interest of analyzability are speed, acceleration, force, friction, energy and path. Two mechanisms are provided to output such data:

- A simple measurement tool to measure distances and
- A powerful analyzer which can extract and visualize all interesting, relevant physical magnitudes from shapes (Figure 5 right).

As an archetype for the analyzer we used an oscilloscope. An oscilloscope has two axes, a time axis and a voltage axis. Both axes are equivalently used in our application. On the value axis we can visualize components of 3D vectors in addition to absolute values, like voltage for example. This is needed because magnitudes like velocity or path have three dimensions plus a value. Therefore our analyzer has four input fields - x, y, z and an absolute value.

The input fields can be connected to different shape attributes, like speed or kinetic energy. The attributes are represented through so called *adapters*. Velocity, energy and a path adapter are supported. Adapters have to be added to each shape manually. An added adapter interfaces a specific physical magnitude and via the adapter data can be plugged into the analyzer. During simulation the adapter will continually send its values to the analyzer and the accordant time-value function is drawn in real time. This allows students to study all data during the running experiment. Moreover adapters do not only act as interfaces to the analyzer but can visualize their values. For example a path-adapter (which is typically attached to an object) records and subsequently draws its trajectory whereas a velocity-adapter splits and visualizes the absolute velocity vector in its x, y and z direction.

4.4 Force Adapter

Besides all the adapters with analyzing capabilities we added a force adapter. With the force adapter a directed and dynamically changeable force can be put onto a shape to affect its motion. This enables many new physical experiments: For example imagine a simulation of precisely accelerating and slowing down a car. Configuration of the force adapter is done through the PIP. When selecting the adapter a force-time function can be defined using a built-in key frame editor. The function is finally processed during simulation. This functionality is demonstrated in the next section in the experiment Teaching Lesson: Speed and Velocity.

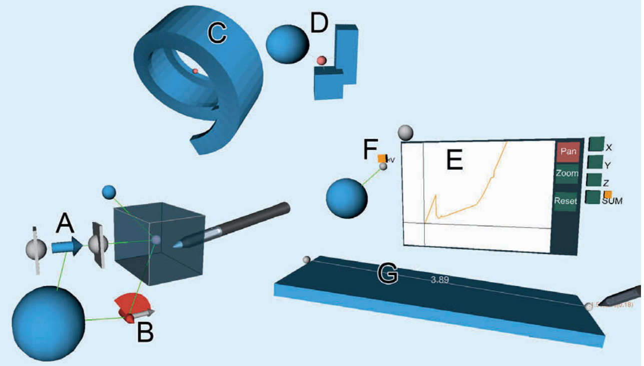


Figure 5: *The PhysicsPlayground elements and their visualizations in VR.*

In Figure 5 the above mentioned building blocks of PhysicsPlayground are shown. A and B represent mechanical linkages. Both connect the same shapes, a box and a sphere. The revolute joint B restricts the rotational motion between both bodies along the axis in which the red arrow indicates. The prismatic joint A restricts the translational motion between both bodies along the axis of the blue arrow. C is a more complex shape, a loop-the-loop. D visualizes grouped shapes which act together during simulation. E denotes the analyzer which is connected to the velocity adapter F of a nearby sphere. During simulation the velocity value of F is recorded. The orange curve on the analyzer results from an earlier run of the experiment where the sphere bounced onto the underlying plane and continued to roll down. The recorded time-velocity values can be read exactly by just moving the pen over the orange line. They are output in textual form next to the tip of the virtual pen. G demonstrates the measurement tool. Currently the length of a box is measured.

5 Educational Use Cases

In the following the applicability of PhysicsPlayground for school experiments and school exercises will be demonstrated by two selected examples that exploit the strengths of our immersive virtual simulation tool. These examples are targeted to high school students aged 12-18. Depending on the curriculum of the specific physics course Newton's laws are taught at different grade levels. We think that PhysicsPlayground can also be utilized for students at a younger age as well as for basic university/college courses.

5.1 Force and Counterforce

The effect of force and counterforce is described by the third law of motion. It states that every force which is invoked by a body A on a body B leads to a counterforce into the opposite direction with the same absolute value. An experiment that we use to demonstrate force and counterforce is described in [Pohl 1931]. Thereby two persons stand face to face each on their own frictionless moving carriage (Figure 5). At the same time both persons hold a rope which connects them. With this setup a series of tests can be conducted:

1. Both participants pull the rope at the same time.
2. Only the left person pulls the rope whereas the right person holds the rope.
3. The same as before but now only the right person pulls on the rope.

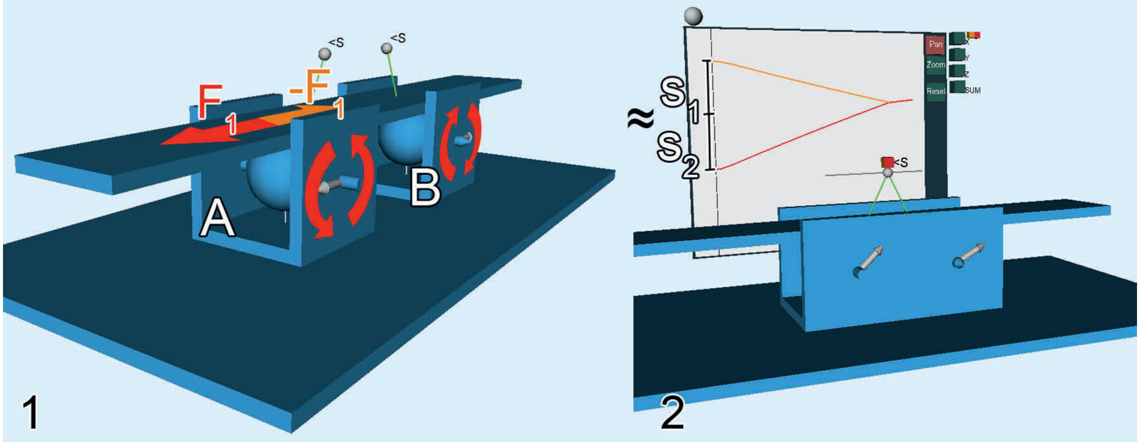


Figure 6: *Demonstration of force and counterforce by using two frictionless slidable boxes.*

Independent of which participant pulls the rope both wagons will move the same distance from start to the point where both collide in the middle. This is a result of force and counterforce.

In our simulation of the experiment the wagons (where the persons stand on) are replaced by two identical, frictionless slidable box frames A and B. Each of the boxes contains a rotatable sphere in the middle which is mounted on the box frame via two revolute joints. These spheres represent the two persons. The rope is then realized through a bar which lies on top of both spheres. Additionally bar and sphere must have a maximized friction coefficient of 1 so that everything works out right. Next we want the participants to pull the rope during simulation. Therefore a motor has to be attached to one revolute joint of each box frame. On body A this motor has to spin counter-clockwise whereas on body B it has to turn clockwise. When one of the motors is activated during simulation, it will put a predefined torque along the rotational axis of its attached sphere. This will put the sphere into a rotational motion. The sphere again will try to transfer this motion to the bar at the point where the bar touches the sphere. This leads to a force F_1 which affects the sphere tangentially opposed to its rotational direction. In the other direction it invokes a counterforce $-F_1$ on the bar which has the same absolute value as force F_1. No matter if one or both motors are activated, the person who watches the simulation will notice that both box frames will always meet in the middle (the mid position in between their starting positions). Additionally this can be proven in PhysicsPlayground by analyzing the path of each box frame through a trajectory adapter. During simulation the analyzer will record a similar curve like shown in Figure 6 right. Students can read the distances s_1 and s_2 from the analyzer. Regardless of which person pulls the rope both carriages always move equivalently.

5.2 Teaching Lesson: Speed and Velocity

This example demonstrates how PhysicsPlayground can be integrated into a traditional physics lesson. An appropriate physical exercise from [Stark 2002] was chosen. It was part of an actual high school final physics exam. In this exercise two bodies A and B slide on top of a plane along the global x-axis with different speeds. At some point in time both bodies collide. A number of physical scenarios can be studied with this setup. We will recreate the scenario which relates most to the school task [Stark 2002] and extend it in the following. The starting setup in PhysicsPlayground contains a plane and two bodies A and B as shown in step 1 of Figure 7.

The task description for students is the following: A toy car which is represented by body A is accelerated along the positive x-axis. Starting at its standing position it is accelerated by a time dependent force $F_A(t)$ which is given in Figure 8. The friction of all shapes is zero. Use PhysicsPlayground to acquire the final speed v_{end} of the toy car A and check your result by calculating it manually.

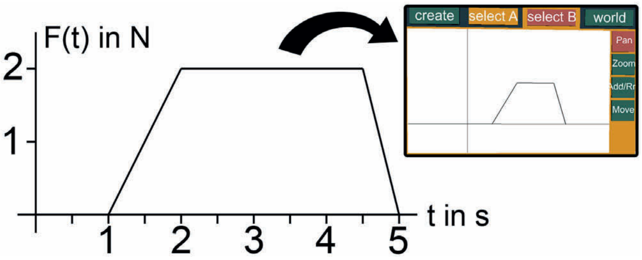


Figure 8: *The given time force-function (left) and force adapter conversation (right).*

In order to solve this task with the help of PhysicsPlayground, students have to transfer the time-force function to PhysicsPlayground first. Basically this is done by adding a force adapter which points towards the positive x-axis to the toy car. The adapter is displayed in Figure 7, attached to object A (orange dot denoted <F>). In the next step the key-frames of the adapter have to be adjusted so that they represent the time-force function as can be seen in Figure 8. If done correctly the body should start to move along the x-axis after starting the simulation. In order to check the movement students may create an analyzer and connect the force adapter to the sum-input of the analyzer. If the recording of the analyzer is equal to the time-force function in Figure 8 the previous steps were performed correctly. The next step is to investigate the velocity of the toy car. This is straight forward as we only have to attach a velocity adapter to body A. Afterwards the velocity adapter has to be connected with the x-input of the analyzer to get a recording of the final speed of the toy car on the analyzer. The print out of the analyzer is finally shown in Figure 9.

The only relevant graphs are $F_A(t)$ and $v_A(t)$. In figure 9 there are additional printouts of body A and B just to demonstrate what can be done with the analyzer (they are not relevant for this task). The graphs labelled E stand for kinetic energy. The graph of $F_A(t)$ represents the time force function which is applied to the toy car during simulation. It is obvious that its analyzer graph is pretty

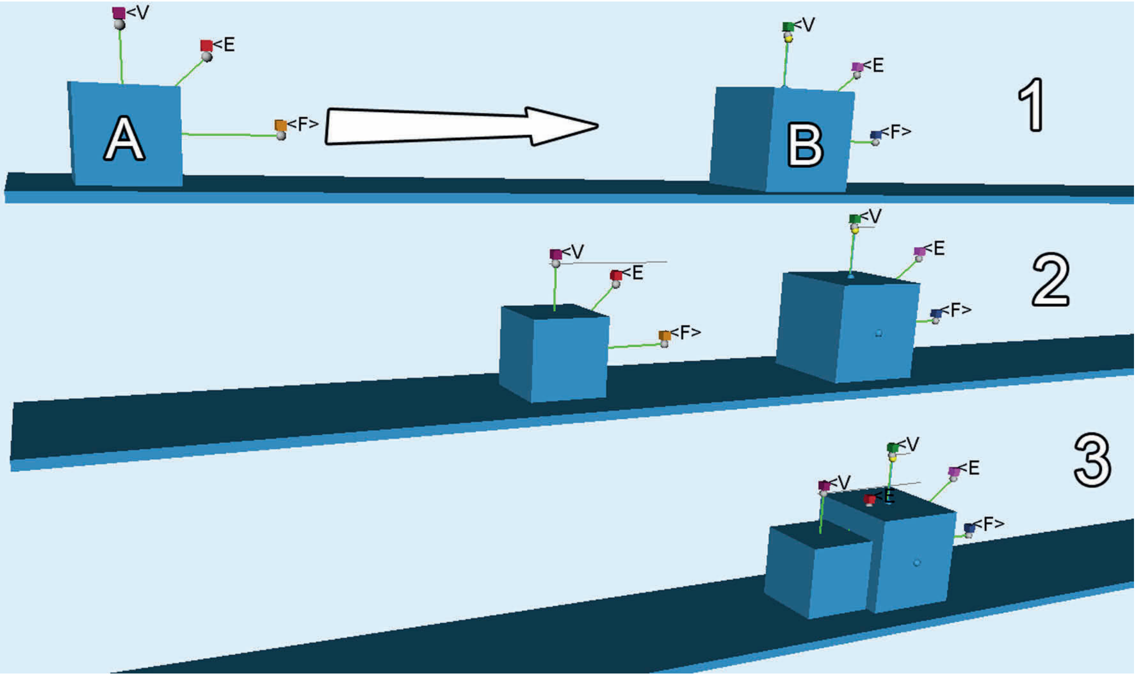


Figure 7: *Complete setup for a PhysicsPlayground based school exercise. Steps one to three visualize the simulation progress in PhysicsPlayground.*

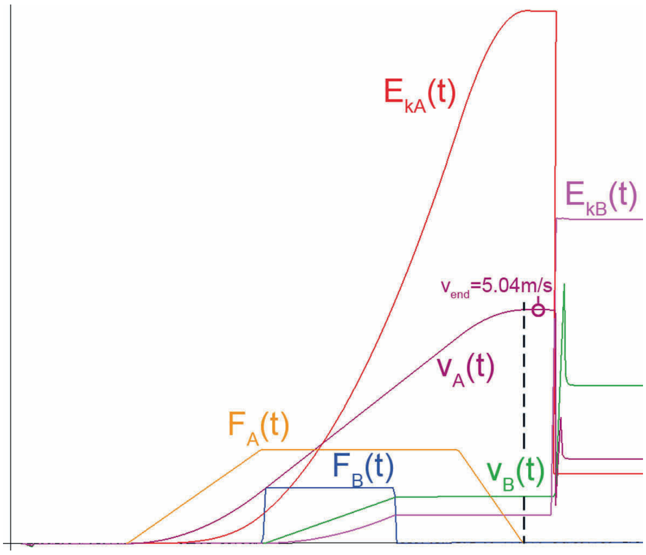


Figure 9: *Analyzer output for bodies A and B. The single curves were labeled afterwards.*

much the same as in Figure 8. This is exactly how it is supposed to be. The velocity graph $v_A(t)$ of the velocity adapter derives from $F_A(t)$. Final velocity is reached when the force F_A stops to affect body A. From this point on the black dotted line perpendicular to the abscissa (where $F_A(t)$ stops affecting A) intersects $v_A(t)$ at a point from which on v_{end} is reached. According to the analyzer $v_{end} = 5.04\ m/s$. Since the solution, namely v_{end} has been found using PhysicsPlayground students can try to calculate it next.

In the solution of Stark [2002] the area below the force function $F_a = \sum F * \Delta t$ is calculated first. To calculate v_{end} the area F_a is used as a parameter in a formula for constant accelerated movement $v_{end} = F_a/m$ whereby m is the mass of body A. After determining its mass within PhysicsPlayground we calculate the result v_{end} and get a final result of $5.00\ m/s$. This is very close to the result of the simulation.

5.2.1 Accuracy and Robustness

At this point we also want to discuss the matter of accuracy, precision and robustness of the simulation and the physics engine that is being used. In order to study physical experiments live in an interactive environment all simulations in PhysicsPlayground have to be performed in real time. Therefore a physics engine was chosen that is optimized for the gaming market and is able to perform all required calculations in real time. For optimization purposes this means that some physical properties cannot be calculated exactly but have to be approximated by the physics engine. In contrast to that results of experiments do not only have to look believeable and realistic but must also be correct in order to be usable in education. A tool that teaches wrong contents cannot be used in classrooms even if it only produces wrong results in a small percentage of cases. In addition educational tools must be very robust. If crashes or chaotic behavior of the simulation occur frequently students loose motivation quickly.

In [Seugling and Rölin 2006] multiple physics engines have been evaluated and the AGEIA/NVIDIA PhysX engine proved to be the most accurate, most precise and most robust of all 'real-time' engines targeted to the gaming market. Accuracy was compared using physical formulas to calculate properties such as friction, gyroscopic forces, stability of constrains and others. The theoretical values were then compared to the actual behavior of the physics engine. In this comparison AGEIA performed best but for further detail we refer to [Seugling and Rölin 2006]. In our example (in 5.2) the practical result of the experiment with $v_{end} = 5.04\ m/s$

is very close to the theoretical value of $5.00\ m/s$ and is considered sufficiently accurate for educational purposes. PhysicsPlayground has been designed with the assumption that experiments are explained and guided by a teacher in a classroom setting in any case. In such a setting teachers are supposed to discuss numerical errors of simulated experiments with students. Nowadays students are using a number of educational tools that produce numerical errors – such as numerical calculators, dynamic geometry or CAD packages – and therefore that topic is of general importance.

Apart from the above mentioned exercises and experiments PhysicsPlayground can be used to demonstrate a wide variety of mechanical devices as shown in Figure 1.

5.3 Motion Paths

Last but not least the application can also be used to visualize mathematical and geometrical contents such as kinematics (Figure 10). Through an appropriate linkage and rotational forces between three shapes A, B and C, we can cause shape B to rotate around the static shape A, and C rotate around B. When the trajectory is visualized through a path adapter on C, mathematical curves like epi- and hypocycloids are generated.

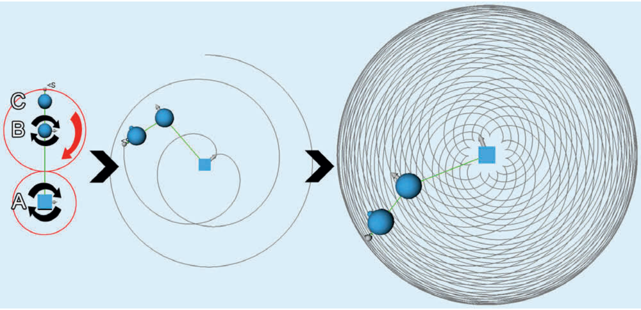


Figure 10: *An epicycloid in PhysicsPlayground.*

6 Conclusion

With PhysicsPlayground an AR application was developed that aids teaching and learning mechanics. The physics engine PhysX on top of the Studierstube framework provides the technical basis and manages all physical calculations. The accuracy of the simulations is good and considered sufficient for educational purposes compared to exact calculations which we used to verify our results.

The educational use cases demonstrate how versatile PhysicsPlayground can be integrated into physics lessons. The potential applications are manifold. Students can build virtual mock-up models of experiments to study physical properties, verify formulas, develop theories and actively participate in physics education in general. It fosters experimentation and understanding.

Up to now PhysicsPlayground was only evaluated by staff members of the Interactive Media Systems Group at Vienna University of Technology. No evaluation about its effectiveness for learning has been conducted yet. It would also be interesting to gather feedback on usability of the application as it uses its own GUI and interaction scheme. Therefore an expert-based usability evaluation including physics teachers and students from different grades would be a meaningful next step.

At trials with PhysicsPlayground we realized that there are two major strengths of the presented educational tool:

- Nearly haptic interaction when building and running physical experiments. Students are able to walk around objects and can view the experiments from different perspectives.
- The possibility to simulate experiments in real time enables quick variation of parameters and reconfiguration of an experiment. It encourages modifications.

In summary PhysicsPlayground is best suited for simulating and solving inherent three-dimensional physical problems that are hard to do in real life.

Acknowledgments

The authors would like to thank Mathis Csisinko for his support during the development of PhysicsPlayground.

References

ADAMO-VILLANI, N., AND WRIGHT, K. 2007. Smile: an immersive learning game for deaf and hearing children. In *ACM Proceedings of SIGGRAPH 2007 Educators Program*, ACM Publications, San Diego, USA.

ADAMO-VILLANI, N., CARPENTER, E., AND ARNS, L. 2006. An immersive virtual environment for learning sign language mathematics. In *ACM Proceedings of SIGGRAPH 2006 Educators Program*, ACM Publications, Boston, MA, USA.

AGEIA. 2008. *AGEIA PhysX Documentation*. NVIDIA, http://developer.nvidia.com/object/physx.htm.

BYRNE, C. 1996. Water on tap: The use of virtual reality as an educational tool. Tech. rep., University of Washington, College of Engineering, Washington.

DEDE, C., SALZMAN, M. C., AND LOFTIN, R. B. 1996. Sciencespace: Virtual realities for learning complex and abstract scientific concepts. *Proceedings of IEEE VRAIS '96*, 246–252.

GERTHSEN, C., AND VOGEL, H. 1993. *Physik - Ein Lehrbuch zum Gebrauch neben Vorlesungen, 17th edition*. Springer publishing, Berlin, Germany.

GOLDSTEIN, H., POOLE, C. P., AND SAFKO, J. L. 2001. *Classical Mechanics, 3rd Edition*. Addison Wesley, Reading, Massachusetts, USA.

HECKER, C. 1996. Physics, the next frontier. *Game Developer Magazine*, 12–20.

KAUFMANN, H., AND DÜNSER, A. 2007. Summary of usability evaluations of an educational augmented reality application. *R. Shumaker (Ed.): Virtual Reality, HCI International Conference (HCII 2007), LNCS 4563, Springer-Verlag Berlin Heidelberg*, 660669.

KAUFMANN, H., AND SCHMALSTIEG, D. 2003. Mathematics and geometry education with collaborative augmented reality. *Computers & Graphics 27*, 3 (Jun), 339–345.

KAUFMANN, H., AND SCHMALSTIEG, D. 2006. Designing immersive virtual reality for geometry education. In *Proceedings of IEEE Virtual Reality Conference 2006*, 51–58.

KÖRNER, W., HAUSMANN, E., KIELING, G., MENDE, D., AND SPRETKE, H. 1988. *Physik - Fundament der Technik, 10th edition*. VEB publishing company, Lipsk, Germany.

LIFSHITZ, E. M., AND LANDAU, L. D. 1982. *Course of Theoretical Physics : Mechanics, 3rd ed.* Butterworth-Heinemann.

MANTOVANI, F. 2003. Vr learning: potential and challenges for the use of 3d environments in education and training. In *Towards CyberPsychology: Mind, Cognitions and Society in the Internet Age*, G. Riva and C. Galamberti, Eds. IOS Press, Amsterdam, 207–225.

MEYER, B. 2007. *Physics Education in the Field of Mechanics with Virtual Reality*. Master's thesis, University of Applied Sciences Deggendorf and Vienna University of Technology.

PINTARIC, T., AND KAUFMANN, H. 2007. Affordable infrared-optical pose-tracking for virtual and augmented reality. *Proceedings of Trends and Issues in Tracking for Virtual Environments Workshop, IEEE VR 2007, Charlotte, NC, USA*, 44–51.

POHL, R. W. 1931. *Mechanik und Akustik - Einfhrung in die Physik, 2th edition*. Springer publishing company, Berlin, Germany.

SCHMALSTIEG, D., FUHRMANN, A., HESINA, G., SZALAVARI, Z. S., ENCARNACAO, L. M., GERVAUTZ, M., AND PURGATHOFER, W. 2002. The studierstube augmented reality project. *Presence - Teleoperators and Virtual Environments 11*, 1 (Feb), 33–54.

SEUGLING, A., AND RÖLIN, M. 2006. *Evaluation of Physics Engines and Implementation of a Physics Module in a 3d-Authoring Tool*. Master's thesis, Umea University, Department of Computing Science, Sweden, Umea.

STARK. 2002. *FOS, Abschluss-Prüfungsaufgaben mit Lösungen, Fachoberschule:*. Stark publishing company.

SZALAVARI, Z. S., AND GERVAUTZ, M. 1997. The personal interaction panel - a two-handed interface for augmented reality. *Computer Graphics Forum 16*, 3, 335–346.

TAXEN, G., AND NAEVE, A. 2001. Cybermath: Exploring open issues in vr-based learning. *SIGGRAPH 2001 Educators Program SIGGRAPH 2001 Conference Abstracts and Applications*, 49–51.

WINN, W. 1993. A conceptual basis for educational applications of virtual reality. *Technical Report TR 93-9*.

DESIGN OF A VIRTUAL REALITY SUPPORTED TEST FOR SPATIAL ABILITIES

Hannes KAUFMANN[1], Mathis CSISINKO[1], Irene STRASSER[2], Sabine STRAUSS[2], Ingrid KOLLER[2] and Judith GLÜCK[2]
[1]Vienna University of Technology, Austria
[2]Alps-Adriatic University of Klagenfurt, Austria

ABSTRACT: This paper focuses on the development of a new spatial ability test in virtual reality (VR). This test measures the ability to visualize and mentally manipulate three-dimensional objects directly in 3D space, and should thus have a higher ecological validity than previous spatial ability tests. Items are viewed through head mounted displays and manipulated by means of a wireless pen input device. As a dynamic tests consisting of a pretest, a training phase, and a posttest it does not only measure a person's current status but also his or her learning potential. Monitoring user interactions in a VR environment allows to measure test performance in ways not possible with traditional means. We describe design and development of the test and will present results of a pre-study with 240 participants conducted in early 2008.

Keywords: Spatial Abilities, Virtual Reality, Dynamic Testing.

……

1. INTRODUCTION

In a previous research project the main authors evaluated the effects of an augmented reality (AR) based geometry training on spatial abilities [5, 8, 15]. Two results were surprising and intriguing: (1) Classical paper-pencil spatial ability tests seemed to be not sensitive to some aspects of spatial performance, possibly due to their two-dimensional nature and limited difficulty range, and (2) in the control group (without any training), there were marked individual differences in performance increases between pre- and posttest. This suggests that individuals differ in their "learning potential" with respect to spatial abilities. These findings led us to the idea of developing a new spatial ability test that (a) measures spatial abilities in three-dimensional space, and (b) includes a training phase, so that learning potential as well as performance status can be measured.

The main goal of our current work whose initial findings are presented here is to develop a new means of measuring spatial abilities in an ecologically valid way. The most important innovation is that our measurement instrument is based on virtual reality, a technology that allows for the projection of virtual objects into real space. Wearing head mounted displays users can interact with three-dimensional objects in space. They can view them from different perspectives and construct or manipulate (e.g., intersect, transform or rotate) them. Thus, virtual reality offers possibilities far beyond those of classical spatial ability assessments. Spatial ability by definition mostly deals with objects and configurations in three-dimensional space. Previous spatial ability assessments, be they paper-pencil tests or computerized versions, are two- dimensional in nature and therefore require transformational processes that many real-life spatial tasks do not demand. The virtual three-dimensional stimulus material that we are developing is one important step closer to reality. Testees see the test items three-dimensionally and can view them from different perspectives (Figure 1).

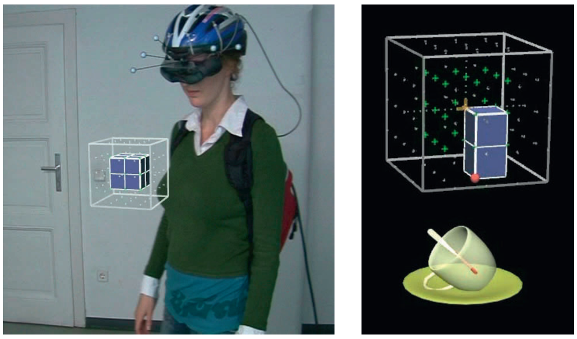


Figure 1: Left: Participant walks around a test item which appears fixed in space. Right: An animation shows a yellow cup rotating around a horizontal axis. It indicates that the test item must mentally be rotated the same way.

The construct validity of the test for measuring three-dimensional spatial ability is expected to be higher than that of existing tests. The new item material requires participants to visualize and reconstruct rotations of three-dimensional objects in space. Items can be designed to span a wide range of item difficulties (see 3.2).

2. RELATED WORK

2.1 Spatial Abilities and Spatial Ability Tests

Broadly spoken, spatial ability refers to the ability to mentally represent and manipulate visual-spatial information. Spatial ability is not a unitary concept but includes several heterogeneous "sub-abilities", each referring to different aspects. Various attempts have been undertaken to structure spatial ability into sub domains [2, 17, 27]. Most of these proposed structures focused on relationships and similarities among spatial tests and were based on the results of correlational and factor-analytical studies. The most-cited and widely accepted model was proposed by Lohman [17], who distinguished three sub factors of spatial ability: Spatial Relations (or Speeded Rotation), Visualization, and Spatial Orientation. Spatial Relations refers to highly speeded tasks requiring the mental rotation of simple two-dimensional [28, 17] or three-dimensional [25] objects. Visualization includes a broad spectrum of complex, multi-step spatial tasks that are administered under relatively unspeeded conditions. The majority of spatial ability tests (e.g. paper folding, form board, or surface development tests) are assigned to this sub dimension. Spatial Orientation refers to tasks in which a given object or an array of objects has to be imagined from another perspective [11]. This dimension is related to orientation and navigation in real or virtual environments ("large-scale spatial abilities"), whereas the former two refer to manipulating three-dimensional objects, such as constructing or visualizing mechanical objects ("small-scale spatial abilities"). An overview and classification of spatial ability tests is given by Eliot [6]. Interestingly, most stimulus types used in current spatial ability tests are still very close to Thurstone's original developments.

There are several shortcomings of traditional paper-pencil formats that are especially relevant when spatial ability is concerned, and some authors have argued that these may be one reason for the relatively low predictive power of spatial ability tests when it comes to real-life spatial tasks [10, 20, 13]. First of all, although most existing spatial ability tests – especially the more complex ones – aim at assessing three-dimensional spatial abilities, virtually all of them use two-dimensional presentations of the stimulus material. Thus, solving these test items requires participants to mentally transform a two-dimensional picture into a three-dimensional figure, to perform some mental manipulations on the figure, and to re-transform the result into a two-dimensional picture. One could argue that this 2D-3D transformation adds a difficulty facet that is not directly related to what is supposed to be measured.

A second shortcoming is that the large majority of existing tests require participants to select the correct solution to each item from a (small) number of response alternatives. Thus participants can solve tests by excluding the alternatives one by one, often by concentrating on single features, as opposed to mentally constructing the correct solution [7]. There are several possibilities for avoiding this problem. One is to have participants actively construct their solutions [18], which is facilitated by the use of computerized systems. Another is to present the task in a stepwise fashion so that participants need to keep track of a sequence of manipulations. In such cases, participants need to mentally manipulate at least parts of the stimulus in order to be able to identify the correct solution.

These two points of critique are true not only for paper-pencil, but also for the majority of computerized tests. Most computerized spatial ability tests that are currently available are simply computerized versions of existing paper-pencil tests [20] - see also overviews of available computerized tests e.g. [24].

New technologies such as virtual and augmented reality allow for the development of a new generation of spatial tasks that are three-dimensional in presentation and response format, and support active construction of solutions in three-dimensional space.

2.2 Applications of VR/AR for Testing Spatial Abilities

As early as the 1980s, some authors used computers to develop non-static spatial tasks involving moving stimuli [1, 13, 20]; an approach which seems to be currently rediscovered e.g. [3].

There are only few applications of virtual or augmented reality for testing spatial abilities so far. One is the Virtual Reality Spatial Rotation (VRSR) System [22]. Within this system a virtual version of the Mental Rotation Test (MRT) [29] was developed in which participants can physically rotate the stimulus material. Findings obtained with this test, which was intended for clinical use, show interesting differences to the standard MRT. The correlation between performance in the classical and the virtual MRT is only about 0.50. While gender differences favouring male participants are virtually always found with the standard version [30] no gender differences were found with the virtual version [19]. Recently a virtual version of the Mental Cutting Test was presented as well [12].

A number of studies have investigated orientation and navigation in virtual environments through which participants navigate using joysticks or other devices [9, 4, 31]. These applications have shown to be very fruitful for studying orientation processes; however they are still restricted to an essentially non-spatial format of presentation. Participants see the environment on a screen and many important cues that are automatically encoded during real-life locomotion in an environment are missing [26, 16]. The type of virtual-reality application that we are developing overcomes this problem. Participants can move around the objects they are working on and can virtually "touch" and manipulate them. Compared to the use of real three-dimensional objects, VR systems allow for the free creation of an unlimited number of objects and manipulations can be easily undone and redone.

3. DSTAR - SPATIAL ABILITY TEST

3.1 Technical Setup

The standard immersive setup used for our Dynamic Spatial Test in Augmented Reality (DSTAR) supports one user wearing a Sony Glastron stereoscopic see-through head mounted display (HMD) (Figure 2). It provides a stereoscopic view of the virtual environment. The user interacts with the system using a wireless pen. Position and orientation of head and pen are tracked using iotracker [21], a 4-camera infrared-optical tracking system which provides sub-millimetre accuracy. In this setup users can walk around test items to view them from different sides.

One dedicated quad-core PC is used as a tracking server which also renders the stereoscopic view for the user.

Figure 2: Outside view of a user constructing a solution. The user wears a head mounted display and uses a wireless pen as an input device.

It should be noted that we do not use the see-through feature of the HMD during test sessions; only the virtual environment can be seen. In pre-studies we observed that most inexperienced users prefer the non-see-through option. Not seeing the real world allows inexperienced users to focus and concentrate better on the task at hand. It immerses them further in the virtual environment and enhances the feeling of presence.

At the beginning of each test run the height of the test items can be adjusted to position them at a comfortable height for each person. Therefore a tracked prop is used that can be moved up and down a pole.

DSTAR is based on the Studierstube [23] AR/VR framework. It is a robust and widely used framework that provides an excellent base for developing AR/VR applications.

3.2 Test and Item Design

A new test session begins with the user watching a nine minutes instruction video that explains the DSTAR test environment. Then the actual testing in VR takes place which lasts approximately 30 minutes. When launching the test application a brief tutorial consisting of three test items starts. Tutorial items are very easy items helping the user to get familiar with the VR environment and interaction therein.

Test items in general consist of multiple simple and complex figures composed of single cubes. These figures are presented within a 4x4x4 transparent cubic grid. One such object is shown in Figure 3. The grid appears to be floating in mid space at a fixed location in the middle of the room (Figure 1 left). Items always contain several – typically two to four – steps. Each step consists of a virtual object positioned within the transparent grid plus a rotation of that object.

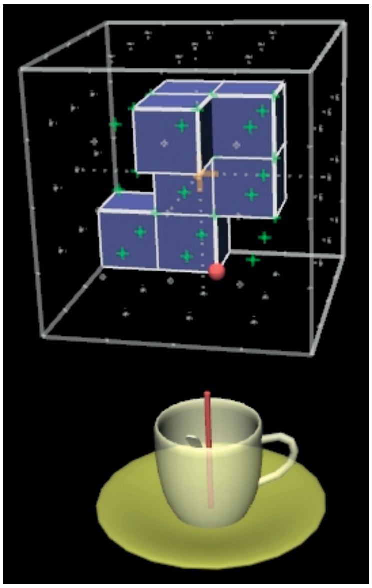


Figure 3: Complex figure as part of a test item. The red point indicates the center of rotation. The tea cup shows which rotation is performed on the given object. In this image the vertical axis is the axis of rotation.

A virtual tea cup (including handle and spoon) beneath the grid is used to demonstrate rotations. The tea cup is animated to visualize axis an angle of rotation (with the angle being 90 degrees or multiples). A non-symmetrical real life object was chosen to display rotations in an understandable way for the whole test population. Participants have to mentally rotate the figure around the given center of rotation (red dot in Figure 3) using the rotation displayed by the tea cup.

After memorizing the end position of the rotated object the user himself chooses when to advance to the next object in the sequence. There is no time limit. The subsequent object has again a (different) center of rotation with a tea cup below indicating which rotation to perform. There are usually two to four different objects/steps within an item. The positions of all mentally rotated objects have to be memorized and combined mentally. They result in a bigger connected object in the 4x4x4 grid.

In a final step this resulting object must be actively constructed by the testee inside a blank 4x4x4 grid (see Figure 4 right). This enforces mental construction of the whole solution. It avoids the strategy of excluding alternatives out of a pool of potential solutions based on specific features.

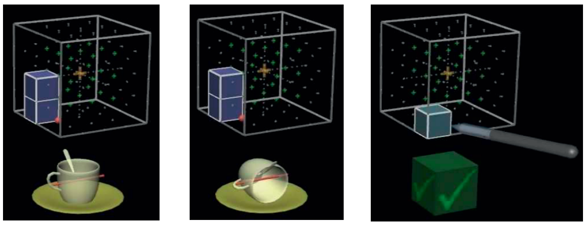


Figure 4: Left: Simple object which has to be rotated 90 degrees as indicated by the tea cup (middle). Right: Final construction of the solution in an empty grid.

Since most participants have never used a VR environment before, interaction has to be extremely intuitive. This is a major usability requirement of the test.

In order to keep the focus of the testee on the object in the center of the room, all interaction elements are arranged around that cube. All elements can be handled by using a single button on a wireless pen. When clicking the button while the pen is located at the tea cup, the animation showing the rotation is repeated. While viewing objects there are yellow arrows displayed at the left and right side of the cubic grid (Figure 5). They point to the left and right and indicate that the user can switch backwards and forwards to view previous and subsequent steps of the item. As long as the user does not choose to solve an item he can still study the whole sequence of objects (Figure 5). When entering solution mode the arrows disappear and a green checkbox appears below the cubic grid (Figure 4 right).

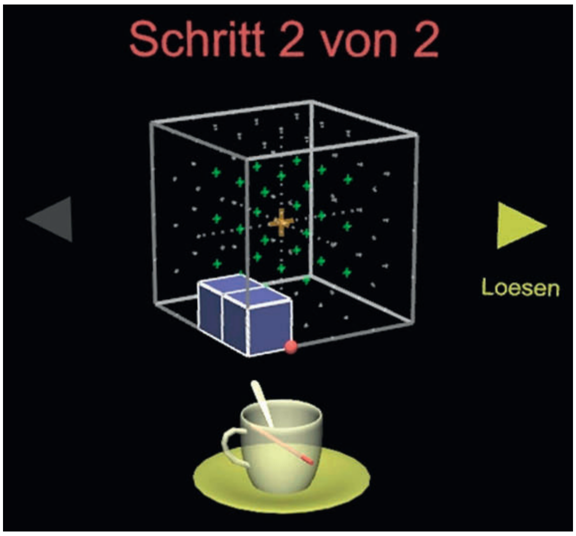


Figure 5: Entering solution mode by clicking the right arrow.

In solution mode the pen is used to draw a solution inside the empty 4x4x4 grid. Clicking the checkbox submits the constructed solution. The test item is concluded and the next item starts.

An item pool of 80 test items was generated following construction rules based on theoretical considerations. Parameters influencing the complexity of an item are "dimensionality" of the objects, rotation axes and their position, complexity of the single objects, their position within the grid, number of rotations per item and angle of rotation. Multiple rotations (two) per object are possible as well using multiple (two) tea cups below the cubic grid.

Given these parameters items are ranked in four categories: easy, medium, difficult, extreme. In chapter 4 we give details on the evaluation of items.

3.3 Advantages of using VR

One general benefit of computerized tests is that additional performance measures, e.g. response latencies and information on solution strategies, can be collected [9]. Self-report measures of strategy use can be problematic because they require a high amount of introspection [10, 8]. The non-reactive assessment of individual differences using computer logs offers a non-intrusive method of analyzing solution strategies; however, these possibilities have hardly been used yet.

Monitoring user interactions in a VR environment allows to measure test performance in ways not possible with traditional means. In addition to recording the time required to solve an item or certain steps, we monitor all user clicks when constructing a solution, (head) movement of the user around the item, number of forward/backward steps during the test and correctly solved items. All of these variables are surveyed to analyze if we can deduce strategies that the participant used to solve the tasks.

3.4 Dynamic Assessment of Spatial Abilities

Another important aspect is that we will develop a dynamic assessment of spatial ability. This is a test that measures both current performance level and potential for improvement. Dynamic tests consist of a performance (status) assessment, a training phase, and a second assessment of post-training performance. In this way, in addition to a single-time measurement of performance status, the degree to which an individual can profit from training is assessed. Previous research has shown that the potential to profit from training is important and valid additional information about a person because it levels out – at least partly – individual differences in relevant pre-assessment experience. For example women's performance deficit in spatial tasks compared to men can be linked to a lower degree of practice and may be leveled out, or at least reduced, by relatively brief trainings. In a previous project [5] we found marked increases in test performance from pretest to posttest even in the control groups, especially in participants with low pretest performance. From this perspective, assessing training profit in addition to pretest status increases the predictive power of a test.

4. EVALUATION AND RESULTS

In a pre-study 240 persons participated. Up to date we can present data of the first 152 participants (93 female, 59 male), aged 19-74 (mean 26), mainly students of cultural sciences (69.6%) and technical sciences (18.0%). The whole study is conducted in a dedicated room equipped with the specified hardware at the Alps-Adriatic University of Klagenfurt, Austria.

4.1 Pre-Study Evaluation Design

Testing included the DSTAR test consisting of six items with varying difficulty, tasks on spatial working memory and a paper-pencil standardized spatial test (3DW). Five parallel DSTAR test versions (each consisting of six items) were evaluated.

Using Item-Response-Models the validity of the theoretically based difficulty levels of the items was tested. Verbal reports concerning strategies were recorded and transcribed and after that categorized by independent coders; agreement of the category attribution between coders was high.

Since we were interested in inter-individual differences in applying strategies, participants were asked to describe the cognitive strategies they used to solve the spatial tasks in virtual reality. Our hypothesis was that individuals with high spatial abilities would use more differentiated strategies than less experienced persons.

4.2 Results

105 participants took part in a first pre-study where an initial set of 5 parallel tests with 6 items each was tested. The test duration varied between 24 (test 5) and 31 minutes (test 4) in average and also the difficulty of the parallel tests varied greatly even though they were designed to be equally challenging in theory.

Difficulty parameters of the items had an impact on the verbally reported solution strategies i.e. visualizing everyday life objects instead of complex figures, imagine a coordinate system to arrange the single cubes etc. Most persons do not explicitly begin to think consciously about solution strategies until a certain difficulty level is reached. As assumed we found varying differentiation of verbal strategy reports, depending on the number of correct solutions in the test, experience with spatial tasks, gender and educational background.

The results show expected gender differences in our test. Men solve 3,16 out of 6 items in average ($\sigma = 1{,}55$), women only 2,25 ($\sigma = 1{,}26$).

Based on the findings regarding difficulty of the parallel tests of the first study, all tests were redesigned and very difficult items were substituted by easier ones. In a second run the rest of the participants (135 persons) used the easier tests. We only have data of 38 participants of the 2nd study yet. They show that men solve 4,33 items and women 3,33 out of 6 (with similar standard deviation) in the new version.

We found evidence for strategy changes during participants' examination of the spatial tasks, but also adaptation on task difficulty by applying training experience through former test items. For example item number 4 was exactly the same in all parallel tests but the success rate of this specific item greatly varied throughout the test versions. This suggests that experience with former items influences performance.

All participants were asked to self assess their computer skills and computer usage per week. There are significant gender differences in both. This could mean that male participants have higher computer skills than female testees. In addition men use computers more hours per week than women amongst our participants.

It is interesting that self assessment regarding correctly solved items in the DSTAR test correlates high with the person's real result (0,756).

In order to improve usability and technical aspects of the DSTAR environment users were asked to rate the menu interface and usability in general (including informal comments). On a scale from 1 (min) to 5 (max) users rated the comprehensibility and usability of the menu interface with 4,65 ($\sigma = 0{,}72$) which is very high. There are no significant gender differences (male 4,53; female 4,72). This indicates that interaction design as described in 3.2 is functional.

Users reported other usability problems that can be summarized in two categories: wireless pen problems (mostly) and HMD problems. Since we limit test duration to a maximum of 30 minutes hardly any participants report side effects of using HMDs known as simulator- or cyber-sickness [14]. More often wireless communication of the pen interaction device failed and no button clicks were transmitted. This can be frustrating and interrupts concentration on the task. A solution is a new wireless pen that is currently being tested. For reliable communication a cabled version is also being considered.

5. CONCLUSION AND FUTURE WORK

Virtual reality applications promise an ecologically valid way to assess spatial abilities and offer - in addition to traditional tests - new possibilities to gather important data. The spatial test we are developing tries to make optimal use of these possibilities. We do not only measure how much the testee has already been trained but also how much he or she could profit from practice with spatial tasks. This may be of great importance for example in the selection of participants for technical schools or courses. Through its possibilities for hands-on interaction with the stimulus material, virtual reality offers particularly interesting possibilities for the training module of a dynamic test. Hands-on practice has been shown to be one of the most effective ways of improving spatial ability. Thus our technology has advantages for measuring and training spatial skills both compared to other computer technologies and compared to the manipulation of real objects, which has clear physical limitations (e.g., when it comes to intersecting or transforming objects).

We plan to extend DSTAR to a short-time dynamic test consisting of pretest, training and posttest taking place within one or two sessions. The pretest shall be also applicable as a stand-alone assessment tool.

ACKNOWLEDGMENTS

The authors thank Michael Mehling for his technical help and for item integration. Part of this research was funded by the Austrian Science Fund (FWF) contract numbers P16803-N12 and P19265.

REFERENCES

[1] M. Allahayar and E. Hunt, *The Assessment of Spatial Orientation using Virtual Reality Technique*, International Journal of Testing, 3 (2003), pp. 263-276.

[2] J. B. Carroll, *Human cognitive abilities. A survey of factor-analytic studies*, Cambridge University Press, Cambridge, UK, 1993.

[3] T. C. D'Oliveira, *Dynamic spatial ability: An exploratory analysis and a confirmatory study*, The International Journal of Aviation Psychology, 14 (2004), pp. 19-38.

[4] R. P. Darken, T. Allard and L. Achille, *Spatial Orientation and Wayfinding in Large-Scale Virtual Spaces: An Introduction*, Presence: Teleoperators & Virtual Environments, 7 (1998), pp. 101-107.

[5] A. Dünser, *Trainierbarkeit der Raumvorstellung mit Augmented Reality [Trainability of spatial ability with augmented reality] Fakulty of Psychology*, Vienna, University of Vienna. Doctoral thesis, 2005, pp. 242.

[6] J. Eliot, *The Classification of Spatial Tests*, in J. E. I. M. Smith, ed., *An International Directory of Spatial Tests*, NFER-Nelson, Windsor, 1983, pp. 11-15.

[7] G. Gittler, *Dreidimensionaler Würfeltest. Ein raschskalierter Test zur Messung des räumlichen Vorstellungsvermögens*, Beltz, Weinheim, 1990.

[8] J. Glück, A. Dünser, K. Steinbügl and H. Kaufmann, *Warning: Subtle aspects of strategy assessment may affect correlations among spatial tests*, Perceptual and Motor Skills (2007), pp. 123-140.

[9] J. Glück and S. Fitting, *Spatial strategy selection: Interesting incremental information*, International Journal of Testing, 3 (2003), pp. 293-308.

[10] J. Glück, H. Kaufmann, A. Dünser and K. Steinbügl, *Geometrie und Raumvorstellung – Psychologische Perspektiven*, Informationsblätter der Geometrie (IBDG), 24 (2005).

[11] J. P. Guilford and W. S. Zimmerman, *Guilford-Zimmerman Aptitude Survey B.*, Sheridan Psychological Services, Orange, CA, 1947.

[12] N. Hartman, P. Connolly, J. Gilger and G. Bertoline, *Developing a Virtual Reality-Based Spatial Visualization Assessment Instrument*, *2006 ASEE Annual Conference & Exposition: Excellence in Education*, Chicago, IL; USA, 2006.

[13] E. Hunt, J. W. Pellegrino, R. W. Frick, S. A. Farr and D. Alderton, *The ability to reason about movement in the visual field*, Intelligence, 12 (1988), pp. 77-100.

[14] H. Kaufmann and A. Dünser, *Summary of Usability Evaluations of an Educational Augmented Reality Application*, in R. Shumaker, ed., *HCI International Conference (HCII 2007)*, Springer-Verlag Berlin Heidelberg, Beijing, China, 2007, pp. 660-669.

[15] H. Kaufmann, K. Steinbügl, A. Dünser and J. Glück, *General Training of Spatial Abilities by Geometry Education in Augmented Reality*, Annual Review of CyberTherapy and Telemedicine: A Decade of VR, 3 (2005), pp. 65-76.

[16] R. L. Klatzky, J. M. Loomis, A. C. Beall, S. S. Chance and R. G. Golledge, *Spatial updating of self-position and orientation during ral, imagined , and virtual locomotion*, Psychological science, 9 (1998), pp. 1998.

[17] D. F. Lohman, *Spatial ability: A review and reanalysis of the correlatonal literature*, Stanford University School of Education, Aptitude Research Project, Stanford, CA, 1979.

[18] D. F. Lohman and P. C. Kyllonen, *Individual differences in solution strategy on spatial tasks*, in D. F. Dillon and R. R. Schmeck, eds., *Individual differences in cognition*, Academic Press, New York, 1983, pp. 105-135.

[19] T. D. Parsons, P. Larson, K. Kratz, M. Thiebaux, B. Bluestein, J. G. Buckwalter and A. A. Rizzo, *Sex differences in mental rotation and spatial rotation in a virtual environment*, Neuropsychologica, 42 (2004), pp. 555-562.

[20] J. W. Pellegrino and E. B. Hunt, *Computer-controlled assessment of static and dynamic spatial reasoning*, in R. F. Dillon and J. W. Pellegrino, eds., *Testing: Theoretical and Applied Perspectives*, Praeger, New York, 1989, pp. 174-198.

[21] T. Pintaric and H. Kaufmann, *Affordable Infrared-Optical Pose-Tracking for Virtual and Augmented Reality*, *Proceedings of "Trends and Issues in Tracking for Virtual Environments" Workshop, IEEE VR 2007*, Charlotte, NC, USA, 2007, pp. 44-51.

[22] A. A. Rizzo, J. G. Buckwalter, U. Neumann, C. Kesselman, M. Thiebaux, P. Larson and A. Van Rooyen, *The Virtual Reality Mental Rotation Spatial Skills Project*, CyberPsychology and Behavior, 1 (1998), pp. 113-120.

[23] D. Schmalstieg, A. Fuhrmann, G. Hesina, Z. S. Szalavári, L. M. Encarnacao, M. Gervautz and W. Purgathofer, *The Studierstube augmented reality project*, Presence - Teleoperators and Virtual Environments, 11 (2002), pp. 33-54.

[24] Schuhfried, *http://www.schuhfried.at/en/home.html*, last accessed: 22.4.2008.

[25] R. N. Shepard and J. Metzler, *Mental rotation of three-dimensional objects*, Science, 171 (1971), pp. 701-3.

[26] D. H. Shin, D. P. S. and X. Wang, *View Changes in Augmented Reality Computer-Aided-Drawing*, ACM Transactions on Applied Perceptions, 2 (2005), pp. 1-14.

[27] H. Stumpf and J. Eliot, *A structural analysis of visual spatial ability in academically talented students*, Learning and Individual Differences, 11 (1999), pp. 137-151.

[28] L. L. Thurstone, *Primary mental abilities*, University of Chicago Press, Chicago, 1938.

[29] S. G. Vandenberg and A. R. Kuse, *Mental Rotations: a group test of three-dimensional spatial visualization*, Perceptual and Motor Skills, 47 (1978), pp. 599-604.

[30] D. Voyer, S. Voyer and M. P. Bryden, *Magnitude of sex differences in spatial abilities: A meta-analysis and consideration of critical variables*, Psychological Bulletin, 117 (1995), pp. 250-270.

[31] D. Waller, E. Hunt and D. Knapp, *The transfer of spatial knowledge in virtual environment training*, Presence - Teleoperators and Virtual Environments, 7 (1998), pp. 129-143.

ABOUT THE FIRST AUTHOR

1. Dr. Hannes Kaufmann is assistant professor at the Institute of Software Technology and Interactive Systems, Vienna University of Technology and head of the VR group since 2005. After completing his PhD thesis on “Geometry Education with Augmented Reality” he did post-doc research in the EU-IST R&D project Lab@Future. He managed and participated in research projects in the fields of virtual and augmented reality, spatial abilities, computational geometry and educational geometry software. Since 1999 he develops Construct3D, a virtual reality application for mathematics and geometry education. He initiated and led the development of the optical tracking system iotracker. The main author can be reached at kaufmann@ims.tuwien.ac.at or through postal address: Institute of SW-Technology and Interactive Systems, TU Vienna, 1040 Vienna, Austria.

Towards a Universal Implementation of 3D User Interaction Techniques

Mathis Csisinko* Hannes Kaufmann†

Institute of Software Technology and Interactive Systems
Vienna University of Technology

ABSTRACT

This paper presents a versatile - write once, use everywhere - approach of standardizing the development of three-dimensional user interaction techniques. In order to achieve a platform and application independent implementation of 3D interaction techniques (ITs), we propose to implement the related techniques directly in the tracking middleware. Therefore a widely used tracking framework was extended by a Python binding to allow straight forward scripting of ITs. We cluster existing 3D ITs, into those which can be fully, partly or not implemented in the tracking middleware. A number of examples demonstrate how various interaction techniques can quickly and efficiently be implemented in the middleware and are therefore fully independent of the underlying application. We hint at how this approach can be used to decouple menu system control from the application with the final goal to help establishing standards for 3D interaction.

Keywords: 3D interaction techniques, 3D user interfaces, tracking middleware, OpenTracker Python binding.

Index Terms: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, augmented, and virtual realities; H.5.2 [Information Interfaces and Presentation]: User interfaces—Interaction styles; I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction techniques

1 INTRODUCTION AND MOTIVATION

Providing taxonomies and classifications of user interaction techniques (ITs) [3, 5] is important to establish a set of well known, common ITs within the research community. Whereas a number of standard ITs can be considered common knowledge by now, platform independent or even application independent ”standard” implementations of these do not exist. Usually application developers hand-code ITs for their applications themselves. It therefore causes additional programming effort for the developer to experiment with different ITs, which could enhance quality and helps to improve usability of an application though. Decoupling ITs from the application, handling them on a different level can lead to establish application independent, reusable, standard implementations of ITs.

In virtual environments tracking is an indispensable part to acquire data of input devices. Over the past few years, several attempts were made to create a new generation of tracking frameworks, paying more attention to flexibility, modularity and other software engineering aspects. On one hand the aim of these toolkits is to support a high number of various tracking devices. On the other hand they allow extensibility, customization and provide (not so common) features like data filtering and sensor fusion. Combining both aspects in a framework in a highly configurable manner leads to very flexible tracking middleware.

The main requirements for tracking middleware are that complex tracker configurations should be supported by mixing provided features. In addition, it should still be possible to create and maintain such tracker configurations in a simple and convenient way. Tracking data preprocessing in terms of filtering and merging is ideally supported by the framework to create rather complex tracking behaviour. Easy authoring of the tracking configuration or even dynamic reconfigurability for mobile or ubiquitous computing applications is also desirable.

Currently user interaction (UI) techniques in virtual environments are usually implemented on the application level. Given the possibility to create complex tracking behaviour directly on the tracking middle(ware) level, the idea to implement 3D user interface techniques on that higher level becomes quite obvious. Since most application developers are using tracking middleware in order to get hardware support for various interaction devices (without having to reinvent the wheel), this middleware might serve as a common ground for future UI standards. The advantages of such an approach are versatile:

- 3D UI techniques can be specified in a reusable way and independently of the application. By using a scripting language for authoring ITs, rapid prototyping is supported as well. This proves to be very powerful since reconfiguration, adaptation and exchange of related 3D UI techniques becomes very easy.
- ITs do not have to reside in a proprietary way on the application level and do not have to be reimplemented for each application any more. Even better, a repository containing a collection of ITs can be established. It is to be contacted at runtime in order to download, set up and execute script code without program termination of the tracking system.
- Furthermore, dynamic IT adaptation depending on hardware setup and runtime reconfiguration becomes achievable, since all relevant data is accessible from within the middleware. For example, if a user changes his location in a system with heterogenous tracking setups, ITs can be dynamically adapted according to available hardware at a new location.
- The work load of the application process can be effectively decreased if it does not have to process and interpret huge amounts of raw tracking data.
- Data handling can be more efficient, when tracking middleware is able to set up the data according to the application’s needs. Considering distributed MR systems specifically, network traffic can be reduced if there is no need to send large amounts of raw tracking data over the network.
- 3D UI techniques can be handled directly on the tracking server. In distributed environments interpretation of the data is done once for all clients, avoiding inconsistencies in case a client does not receive all raw data and therefore interprets the IT differently (e.g. gesture interpretation).
- Standard or reference implementations of known ITs could be shared by the research community.

---

[*] e-mail:csisinko@ims.tuwien.ac.at

[†] e-mail:kaufmann@ims.tuwien.ac.at

After summarizing related work in the area of existing tracking frameworks (section 2), which our work is based on, we describe the design and implementation (in sections 3 and 4) of our approach. In section 5 results are presented, giving a number of examples and sample implementations. To generalize our approach we cluster existing 3D ITs (based on Bowman's taxonomy [5]) into three types of middleware support in section 6. This section goes into detail on ITs that can be fully realized in tracking middleware, ITs where specific helper functions can be implemented in the middleware to support the application, and finally ITs which are extremely application dependent and can only be realized on the application level. In the remaining part (sections 7 and 8) we summarize our work and describe future work, for example, on how tracking middleware can be exploited to standardize menu system interaction and widget control in virtual environments.

2 RELATED WORK

A number of 3D interaction techniques have been developed by various researchers during the past decade. A comprehensive and recent overview is given by Bowman in [5]. We based our clustering of techniques (section 6) on his taxonomy. Tasks are categorized into *selection*, *manipulation*, *navigation*, *system control*, *symbolic input*, and into more complex *tasks* such as modeling.

Various attempts to standardize 3D user interaction can be found in literature from previous years.

InTML [8] is an interesting approach to describe interaction techniques in a standardized way. In its aim to be high-level, reusable and extensible, it is similar to our work. InTML is not supposed to be a standalone toolkit. Instead, it relies on other VR toolkit implementations. In InTML 3D UI techniques are described in an XML based markup language. In order to get executable code, this description has to be translated, interpreted or compiled. Currently, work is spent on an InTML implementation on top of Java3D and VRJuggler.

A uniform approach for specifying mixed reality interfaces, including 3D UI techniques was recently published by Figueroa et al. [7]. In this ambitious attempt to describe components of 3D user interfaces, a formal specification model and a corresponding XML based description language including pseudo code were introduced. This Interface Component Description Language (ICDL) seems not to be suitable for direct implementation. The current focus of development on ICDL appears rather to be on theoretical specification than on practical implementation.

As described in the previous section, tracking middleware toolkits are specialized in handling tracking data of various devices on an abstract level. Combining input of several devices, performing preprocessing and filtering are interesting features in these frameworks, which shall be referred to in the following.

One of the early software toolkits dedicated to developing interactive and immersive graphics applications is MR Toolkit [19]. It provides device abstraction and network transparency for tracking devices. Applications are decoupled from the actual tracking devices and programmers can substitute real devices with virtual ones for debugging and testing purposes. Active development is discontinued but it showed the way forward in this field of research.

VRPN (Virtual-Reality Private Network) [22] is one of the most well known and popular device-independent and network-transparent frameworks for peripheral devices used in MR systems. It supports a wide variety of input devices and different types of data such as 6DOF pose data, button states, analogue values, incremental rotations and more. A device can offer interfaces for several types, and devices can be layered by connecting device outputs to inputs of other devices. VRPN runs on all common platforms and is used in research and industrial projects.

OpenTracker [17] is an open software architecture that provides another framework for the different tasks involving tracking input devices in MR applications. The OpenTracker framework eases the development and maintenance of hardware setups in a more flexible manner than what is typically offered by VR development packages. This goal is achieved by using an object-oriented design based on XML, taking full advantage of this technology by allowing to use standard XML tools for development, configuration and documentation. A multi-threaded execution model takes care of tunable performance. Filters and transformations can be applied to tracking data. XML based configuration files are used to describe tracking configurations that usually consist of multiple input devices. Transparent network access allows easy development of decoupled simulation models.

We use OpenTracker and extended it by a Python binding to allow scripting as it will be elaborated in detail in sections 3 and 4. In the following we compare our approach with existing work.

The previously mentioned dependence of InTML [8] on other toolkits is one of the main differences to our approach. The same deficiency seems to apply to the ICDL [7] language: no direct implementation appears to exist without the necessity to translate the specification. OpenTracker is already widely used and not depending on other third-party toolkits.

Another interesting feature not found in InTML is the possibility to embed interpretable program code in terms of an easy-to-learn scripting language. As a direct effect of the expressiveness of programming languages, this enhancement also results in a massive increase of expressiveness in describing 3D UI techniques.

Our approach is not limited to OpenTracker only since interfacing with VRPN [22] is possible and easy: OpenTracker contains a built-in module to obtain VRPN data from network and is also capable of directly transmitting tracking data in native VRPN format. With the lack of tracking data preprocessing features in VRPN, but with its support for a rich set of devices, combining OpenTracker with VRPN proves to be very powerful. The example in section 5.2 demonstrates the use of VRPN input.

In contrast to related work the contribution of our paper is manifold. We introduce scripting of interaction techniques in tracking middleware. This tight integration of user interaction techniques and tracking provides increased flexibility as described in section 1 and offers new possibilities (see summary in sections 7 and 8). Decoupling ITs from the application is a methodical general approach which helps to reduce application interface code, enhances performance and eases (distributed) MR system development. Scripting ITs once on the middleware allows platform independent usage on multiple systems.

3 DESIGN

Considering the features, OpenTracker is an appropriate framework to integrate 3D UI techniques directly in middleware. We give a brief overview of relevant functionality.

3.1 Data Flow Concept

A main concept of OpenTracker is to build up a data flow network in a modular way, which consists of several steps of data acquisition and manipulation. Breaking up complex behaviour in a number of basic operations results in a data flow graph. Nodes in this directional acyclic graph represent the entities, where tracking data processing occurs, while tracking data is communicated unidirectionally over the interconnecting edges between the nodes. Data is inserted into this graph by special source nodes, processed by intermediate nodes and forwarded to external outputs by sink nodes.

Figures 2, 4 and 6 are example illustrations of data flow graphs: In these representations, data is propagated unidirectionally from top (sources) to bottom (sinks).

From the perspective of a node, data is exchanged by ports, namely by input and output ports. To allow nonlinear graphs, each node can have multiple input ports and the data generated at its single output port can act as input for several other nodes. This multiple input property is desirable in order to perform more complex computations, when data from different origins is involved. On the other hand, supplying multiple nodes with the same output data of a single node offers a transparent mechanism of data reusability.

There exist three different edge types: The most commonly used is the event edge type (implementing the event generator interface), where data is pushed from the origin successively through the graph. In contrast to this push-driven mechanism, event queue and time dependent edge types are pull-driven: A history of previous events is kept and a polling mechanism has to be established. This is usually more suitable for performing processing on data at different instants. With event queueing, events ordered by time can be retrieved by index, while the time dependent interface allows to directly query by instant.

3.2 Multi-modality of Events

Within OpenTracker, the contents of events are not restricted to predefined data records. Instead, each event can consist of multiple, dynamically created attributes of certain data types.

Apart from predefined generic data types, custom type registration is also supported. Exploiting this possibility allows to pass further configuration data (not necessarily tracking data) between nodes, which can be very helpful in the attempt to implement more complex behaviour directly in tracking middleware. Intercommunication between several nodes in order to pass configuration data becomes feasible.

4 IMPLEMENTATION

OpenTracker is an open source, extensible tracking framework that enables developers to add support for new input devices; providing additional methods for filtering or preprocessing by implementing new nodes is possible. Consequently, the same applies when trying to integrate a 3D UI technique directly in OpenTracker.

4.1 Monolithic Approach

At first, one might be tempted to implement a single simple node, which meets the requirements of a particular IT and fits the user's needs. Although this might be suitable for some purposes, the method has its deficiencies since it is rather proprietary. The functionality of the interaction technique is completely hidden in code and not disclosed in the tracking configuration. With a set of special nodes, each of them tailored to implement a particular 3D UI technique, one can not take advantage of similarities of related techniques. Even if reusing some common code parts by inheritance is possible to some extent, this is not reflected in the configuration file.

In addition, this is not very suitable for rapid prototyping: Each time a new 3D UI technique is to be implemented, it is mandatory to rebuild the whole framework. A more universal solution is desired.

4.2 Modular Approach

In a second attempt, some basic and multi-purpose operations have to be identified. In order to implement a particular 3D UI technique, an appropriately chosen selection from this set of basic operations has to be combined properly to achieve a more complex and meaningful behaviour. This building block style construction process allows synthesizing new 3D UI techniques without code altering and makes functionality more obvious, as it is not completely hidden in code. Such a modular attempt can also be seen as a discrete way, opposed to the integrated way described before.

It is positive that quite a lot of these general multi-purpose operation nodes are already pre-existing in OpenTracker and are reusable in this new context. For example, coordinate transformation nodes in many different flavours were implemented in the past and can be reused to perform calculations on point and orientation data. Merging multiple inputs in several ways as well as various gate nodes were used for other purposes before.

As it turns out, one disadvantage of this modular attempt can be seen in the high complexity of the resulting tracker configuration and - compared to programming languages - the reduced expressiveness of its elements. Figure 2, as referred to in section 5 illustrates this complexity issue for a rather simple task.

4.3 Scripted Approach

This leads us to a third attempt, which tries to overcome the deficiencies of both mentioned methods in 4.1 and 4.2. Embedding a scripting language radically increases flexibility, ensures high expressiveness and offers the possibility for rapid prototyping. No framework rebuilding is required and the 3D UI technique program code is closely coupled to its tracking configuration.

We selected Python [18] as a quite expressive scripting language. It supports object-oriented programming and is easy to learn without much effort. Even though it is an interpreted language, code execution is speeded up by using a byte code representation of script code. Utilizing a powerful wrapping tool, namely SWIG (Simplified Wrapper and Interface Generator) [1, 6], the creation of some kind of "glue code" to access OpenTracker objects from within the domain of Python was straightforward. With this automatically generated wrapper code, Python code can deal with OpenTracker objects in a native and similar way as in C++. Even parameter passing and sharing objects at the interface between Python and OpenTracker is possible to achieve strong data coupling between both languages.

Python script code is called in the event processing mechanism of OpenTracker. For each script node, execution is carried out in methods of a Python class, whose name can be freely chosen. Due to the fact that to the script programmer the same set of interfaces as in C++ is offered, it nicely integrates into the concept of OpenTracker. Also, the Python class exactly resembles C++ class implementations and argument passing is done in the same natural way as in C++, but without the necessity to recompile the whole framework. Apart from rapid prototyping (testing the effects of new feature implementation candidates before actually extending the code base), this also offers the possibility to get familiar with OpenTracker in an easy way.

Going more into detail concerning the data processing mechanism in event generation, events are usually inserted into the data flow graph actively by sources, represented by child nodes of the tracking tree. Subsequently, each child node pushes events further down the tracking tree by notifying its parents. Events do not necessarily have to be processed in this strict manner of being inserted by sources and being removed by sinks though. Event generation and filtering in intermediate nodes is possible as well. In order to accomplish that, the event generator interface in the node class offers a method to notify and update all immediate parent nodes of an event. They get notified whether the event was newly generated, processed or manipulated by any child of this particular node. As a consequence of this notification mechanism, a certain method is called, whenever an event is received from any child. In that call the actual event and information about its origin are passed as arguments to the method.

In tracking trees of depth greater than 2 this mentioned notification method is usually called recursively. Assuming that at least one script node is present somewhere in the tracking tree, we illustrate the integrated concept: The internal C++ part of the script node is notified about an event and passes it along with other parameters to the specified Python method. This Python method itself is free to perform any operation on this event (can even ignore the event), but usually pushes it in a modified way further down the tracking tree. This is achieved by calling the same notification mechanism method as in C++ from within the domain of Python. Due to the wrapping code, the corresponding C++ notification method is executed. In respect to other tracking nodes, there is absolutely no difference, whether an event is created in native C++ code or Python was involved in any stage of its generation.

Even passing events from one Python script node to another, whether directly or indirectly by having intermediate C++ nodes in between, causes no problems: There is only one single Python interpreter present, which operates on the pregenerated byte code of the specified Python code module, which contains class definitions for each Python node. However, it should be noted that the call stack of the notification mechanism in event generation can contain several calls to Python in between of native C++ calls.

5 EXAMPLES AND RESULTS

In order to demonstrate some characteristics of the implementation techniques (as briefly discussed in section 4) two examples are elaborately presented in the following.

In a first simple example we will compare the modular and the scripted approach by using a mouse wheel to trigger a button event. In a second example the Go-Go interaction technique [16] is fully implemented involving scripted and modular features. In that example input tracking data received from a device via VRPN is used.

5.1 Mouse Wheel to Button Conversion

Figure 1 shows a tracked input device, a wireless pen, that we use in our multi-user optical tracking setup. Optical tracking makes use of the retroreflective marker body attached to the end of the wand. In terms of operating system device type classification it belongs to the group of mouse devices, but some of its mouse features are not directly usable in a 3D environment without conversion. The device appears to the tracking system as a passive prop, except for digital button information, which is transceived wirelessly via Bluetooth. In order to rejoin data coming from different tracking subsystems, native button input of this device is merged with position and orientation data supplied by the optical tracking system.

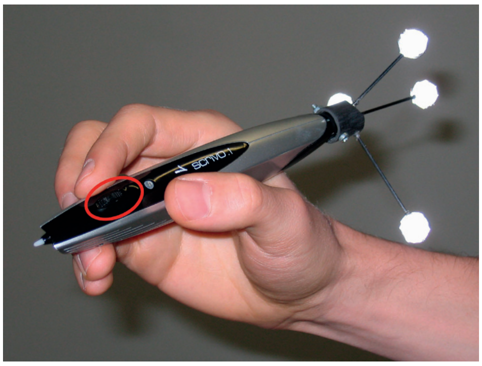


Figure 1: An optically tracked wand/pen. Within the red circle a scroll wheel is visible.

In some applications we use the mouse wheel as a button (by pressing down the wheel). Experience showed though, that some users do not press the wheel but rather tend to scroll the wheel, expecting the application to react to it as if it were a button press. Therefore it may be desirable for several applications to assign button functionality to the mouse wheel. Such a mapping proved to be flexible and very convenient for most users.

At first glance, one might expect that the task of converting data coming from the mouse wheel into button state information is rather simple. The mouse wheel is typically used for scrolling though. Rolling the wheel therefore results in incremental value changes over time which is different to digital button information. The information conversion has to be of differential nature: If the wheel is rolled over a specific time period, a pressed button state should be generated. Whenever the mouse wheel is standing still, a released button state is to be detected. Consequently, the crucial task is to identify the presence or absence of mouse wheel movement. Although this can be done in a monolithic way by creating a special node (as described in section 4.1), a modular implementation (as outlined in section 4.2) shall be depicted in the following.

5.1.1 Modular Approach

We briefly describe the ”building blocks” of the data flow graph. Following the right path in figure 2, data input coming from the mouse input device is scaled by $(0,0,1)$ to eliminate all 2D mouse position data in the x and y component. The remaining z component contains mouse wheel information, which is passed to a timestamp generator node. This is due to the fact that the following node depends on getting events on a regular basis. The TimestampGenerator stores the last received event and resubmits it unmodified each time a given timeout is elapsed.

The next node in the chain computes the difference between attributes of two events separated by a given timeframe. The minuend in this operation is the actual received event, the subtrahend a previously recorded past event as described before. The output of this operation is feed to two range filter nodes. These nodes inspect the position attribute and perform event passing or blocking depending on conditions inspecting the length of the position vector. While the RangeFilter on the left blocks all events when a movement is detected, the behaviour of the other is strictly complementary.

The following simple state machine consists of just two nodes resembling each of the complementary button states (pressed and released) and is only present for convenience: An interface method or helper node (ActiveGate) can be used to retrieve the name or index of the currently active state. In this configuration events are passing the state machine at any instant without modification.

The EventUtility nodes in the next step synthesize button data events. As attribute creation is only happening if an event is pushed through one of these nodes, button state generation is according to the actual state.

The parent Merge operation inspects the events of its child nodes and generates its own events with only the button attribute set. With this semantics it is possible to remerge the data flow paths again: No data combination of child node events is needed and the originating paths of these child node events are not relevant any more. In contrast to this implicit path joining mechanism, the final button information is computed arithmetically in the last step. Plain button information coming from the mouse input device is combined with the result of this conversion in a bitwise OR operation.

5.1.2 Scripted Approach

Corresponding to what we described in section 4.3, the same functionality can also be expressed in a few lines of Python code (see figure 3). This specific Python class which is derived from a base class encapsulates everything needed in an object-oriented way. The instantiation operation method is similar to constructors in C++. It is called on construction of the Python node to allow initialization of any class instance data attributes. Although the attributes are comparable to member variables in C++, they are not declared and can be introduced later too. It can be done in C++ style though to conform to C++ programming patterns. In the case of performing custom initialization, it is essential to call the proper instantiation method of the base class. Otherwise, important initialization tasks for linking this Python class instance to its C++ counterpart will remain uncompleted.

Just like in C++, the method onEventGenerated is called, whenever an event is received from any child. To briefly explain that

```
class ConvertZToButtonNode(PythonNode):
  def __init__(self,pythonNode):
    PythonNode.__init__(self,pythonNode);
    self.eventQueue = [];
  def onEventGenerated(self,event,generator):
    self.eventQueue = filter(lambda prevEvent: prevEvent.time + 250 >= event.time,self.eventQueue) + [Event(event)];
    localEvent = Event();
    if event.getPosition()[2] - self.eventQueue[0].getPosition()[2] == 0:
      localEvent.setButton(event.getButton());
    else:
      localEvent.setButton(event.getButton() | 1);
    self.updateObservers(localEvent);
```

Figure 3: Python code for mouse wheel to button conversion

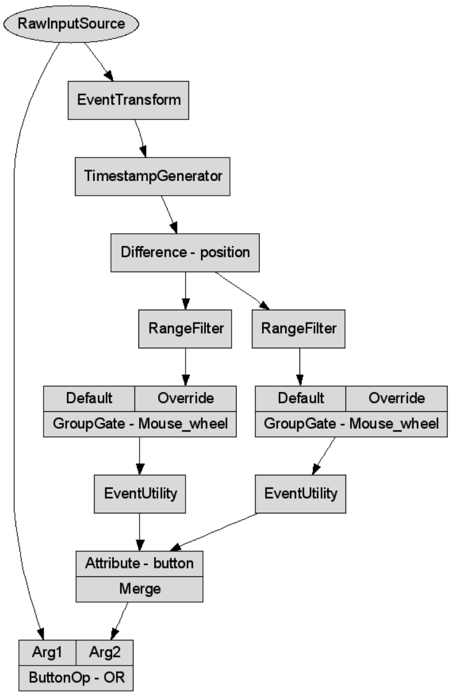


Figure 2: Data flow graph of the modular approach: Mouse wheel to button conversion

method: Data attribute eventQueue contains an ordered list of the last received events. The first line of code in this method deals with keeping the list up to date: The event received at this instant is appended at the end of the list and all past events not used any more in the front are eliminated. This single line statement demonstrates a powerful feature of the Python language, expressing rather complex operations in an elegant and brief way. Afterwards, the local event is generated. Depending on the detection of change in the z component (containing the mouse wheel movement) button state information is determined involving physical button input. Finally, the local event is propagated by calling updateObservers in the same manner as in C++.

It is notable that all operations performed on events involve calling wrapped C++ code. So, throughout the whole method, interfacing with C++ is used extensively.

The tracking configuration as illustrated in figure 4 is rather short this time: Again the TimestampGenerator node has to be present to allow button state fallback, when no movement is detected and device events might not occur until another action is performed with this device. But everything else was substituted by a single Python script node. Even the final bitwise button merging operation was taken care of in the Python node, which is configured by identifying the class to use with its name.

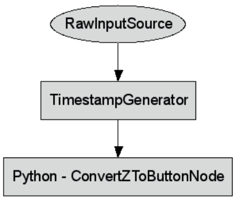


Figure 4: Data flow graph of the scripted approach: Mouse wheel to button conversion

5.2 Go-Go Interaction Technique

The Go-Go interaction technique [16] is an example of how to overcome the limited arm length of a user, who is immersed in virtual reality. The limitation of the user's limbs is directly related to the limited reach experienced by the user, when he is interacting in space but not moving (direct 3D interaction). An attempt to increase the interaction space is to scale the coordinate system depending on the distance between the user's head or torso and the hand or tracked interaction device. While retaining full tracking precision in the immediate surrounding space by leaving the coordinate system unchanged, points farther away are virtually pulled towards the user by scaling down the coordinate system. Alternatively, this transformation can also be seen as virtually extending the user's arm.

The conversion between the real and transformed coordinate system is defined by the following equation expressing the relation between the real (R_r) and virtually transformed (R_v) vector lengths of the user-device distance:

$$R_v = \begin{cases} R_r & R_r < D \\ R_r + k \cdot (R_r - D)^2 & R_r \geq D \end{cases}$$

D determines the radius of the sphere around the user separating near-by objects and those located too far away to be in the user's reach. k is a coefficient in the range $0 < k < 1$ specifying the scaling factor for the non-linear coordinate transformation part.

Supposing, that the origin of the coordinate system is already relative to the user's position, which can be achieved by other standard transformation nodes, the few lines of Python code in figure 5 perform the crucial part of the remaining non-linear transformation characterizing the Go-Go interaction technique.

This time the Python code operates just on the current event without the need for storing a history of previously handled events. Parameters d and k can be manipulated at runtime.

```
class GoGoInteractionTechniqueNode(PythonNode):
  def __init__(self,pythonNode):
    PythonNode.__init__(self,pythonNode);
    self.d = 0.5;
    self.k = 0.75;
  def onEventGenerated(self,event,generator):
    localEvent = Event(event);
    position = event.getPosition();
    distance = (position[0] ** 2 + position[1] ** 2 + position[2] ** 2) ** 0.5;
    if distance > self.d:
      factor = 1 + self.k / distance * (distance - self.d) ** 2;
    else:
      factor = 1;
    localEvent.setPosition((factor * position[0],factor * position[1],factor * position[2]));
    self.updateObservers(localEvent);
```

Figure 5: Python code for the Go-Go interaction technique

In this particular 3D UI technique, one must take care of the coordinate system origin (head or shoulder of the user) as mentioned before. This can easily be done by using standard OpenTracker nodes, as illustrated in figure 6. Therefore this example demonstrates a combination of existing modular ”building blocks” and a simple Python script.

Two VRPNSource nodes represent tracking information directly coming from VRPN [22]. The top node supplies tracking data for the dynamically changing coordinate system on which the remaining VRPNSource node is depending. Therefore the positional information is extracted by means of the attribute functionality in the Merge node first.

In a next step, EventInvertTransform inverts positional data, which is needed to define the proper coordinate system for the remaining VRPN tracking input connected to the EventDynamicTransform node. Its coordinate system shall be relative to the tracking position of the top VRPNSource node. The output is now enhanced by Go-Go interaction technique features as depicted before.

Afterwards, the initial coordinate system transformation is reverted, as the EventDynamicTransform node is configured complementarily to the former one. The result of this transformation is again tracking data which is expressed in an absolute world coordinate system.

Finally, modified position information is again remerged with all other remaining attributes of the tracking device.

If desired, it is of course possible to move Python code back into the OpenTracker C++ code base. This time such a modified implementation seems to comply more with the modular approach (described in section 4.2), as the original implementation was already mentioned as being a mixture of the modular and scripted approach. A C++ tracking node with Go-Go interaction technique features is similar in its class definition and calling mechanism to the original Python node. So, it is as simple as that to move on from a rapid prototype to a permanent implementation in OpenTracker code.

However, the advantage of keeping scripted implementation is to maintain additional flexibility. Only with this approach a repository of 3D UI techniques can be established in order to truly distribute program code at runtime without a priori knowledge of particular techniques at compile time. It is even possible that no information about a particular technique exists at runtime until the script code is obtained from the repository.

6 CLASSIFICATION

3D user interaction techniques have been classified into sets of categories by various researchers. We have chosen a similar classification system like Bowman et al. [5]. In addition to this taxonomy, we introduce an orthogonal property to describe the level of support.

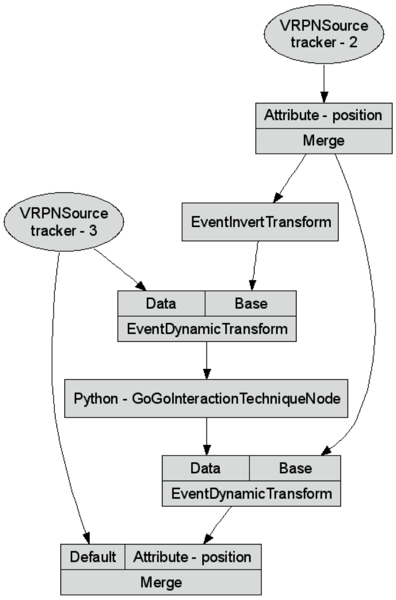


Figure 6: Data flow graph of the Go-Go interaction technique

In terms of supporting 3D interaction techniques in tracking middleware, we observed three different types:

- *Full support*: The 3D UI technique can be directly and fully implemented in tracking middleware.
- *Helper function*: Important tasks of the 3D UI technique have to be carried out on the application level, but some data setup and preprocessing tasks can be accomplished in middleware.
- *Lack of support*: Currently, the 3D UI technique has to be part of the application level. Tracking middleware can not assist in performing the 3D UI technique in a useful way.

In the following, 3D UI techniques (classified in a similar, but simplified manner to Bowman et al. [5]) can be assigned to these three types of support. However, it should be noted that the three levels of support and the implementation techniques as described in section 4 are independent of each other and should not be confused.

6.1 Full support

Most *navigation* and travel techniques can be fully supported in tracking middleware.

Generally speaking, physical locomotion and steering techniques (including semiautomated steering) require transformation or generation of a (relative) coordinate system: Depending on the particular technique, position and orientation data can be determined by simply inspecting input data of travelling-related tracking devices. For example, the tracking middleware can compute a user's position by using data from a steering wheel, combining it with data coming from an acceleration and brake pedal. Of course, this is also possible for any other tracked device used for steering (e.g. gaze-directed steering or steering by pointing).

In addition, some route-planning techniques are well supported: These two-step techniques require a state machine and usually involve input from another digital device (e.g. a button). Considering route-planning as marking points along a travelling path, the implementation is also easily possible. The state machine basically reacts to various events in order to switch between path-planning and actually carrying out the plan.

A practical example would be full support of the ChairIO interface by Beckhaus et al. [2]. All *navigation* interaction techniques as described in the paper (including initialisation) can be fully implemented in tracking middleware. Decoupling these techniques from the application allows flexible use of this innovative interface in various MR applications.

Selection and *manipulation* techniques are not that easy to implement, but core features of the Go-Go [16] and the World-in-Miniature technique [20] are fully supported. Both techniques rely on coordinate space transformation and can be done in tracking middleware. The Go-Go technique is characterized by its non-linear coordinate space transform: At a certain configurable minimum distance around the user, the coordinate system is scaled as described in section 5.2. Scaling is also the key for the World-in-Miniature (WIM) technique, where the user faces an additional down-scaled representation of the virtual environment. By performing these scaling operations directly in middleware the IT is completely transparent to the application and is replaceable.

Various *selection* techniques (e.g. lasso) make use of gestural interaction. Interpretation of gestures such as deletion, selection and other forms of manipulation can be handled directly by maintaining a history of previously recorded events as in section 5.1. Gestural commands can also be used for tasks in *system control*. In addition, *system control* and *symbolic input* techniques like voice input can be processed perfectly in middleware. The application is only informed of the recognized command to react properly.

6.2 Helper Function

The main cause for not achieving full support for a certain 3D UI technique is the object based nature of the particular technique. Usually, almost all *selection* techniques are object based by definition. For such an object based technique, data about all selectable objects would have to be accessible from within tracking middleware. So, a mapping between these objects and corresponding representations in middleware could be set up. In practical applications with a huge number of selectable objects this might not be suitable, since there is a lot of communication effort to keep the object mappings up to date. However, helper support can be achieved by preprocessing input data in middleware, before actually performing object selection in the application.

For example, a two-handed pointing technique can be handled by the application just like a simple ray-casting technique [4], when the middleware performs corresponding data setup. The tracking framework carries out all data conversion making the IT transparent to the application, the application implements only simple ray-casting. This data conversion and transparency attempt is one possibility to perform helper functions in middleware.

Also, precalculation can be done in the Flashlight and Aperture [9] techniques: Even if the selection itself can not be performed in middleware, the selection cone dimensions can be generated by the OpenTracker framework and expressed by custom attributes. So, additional data generation for further use in the application is another option for this helper function category.

A combined IT like HOMER [4] can be done partly in tracking middleware. Coordinate transformation in the manipulation mode can take place in the tracking framework, even if switching between selection and manipulation mode is controlled by application. This requires bidirectional tracking data transfer: Custom events identifying mode changes can be communicated back from the application to the tracking toolkit to indicate the current mode. The scaled-world grab technique [12] is similar in its approach by applying scaling in the manipulation step. These combined techniques demonstrate the possibility for bidirectional communication between tracking middleware and application.

This two-way communication can also be used in *system control* techniques. Meta information about graphical menus can be sent from the application to tracking middleware. Graphical representation of visual feedback has to reside on the application level, but command selection takes place in middleware. In return, the application is informed of user interface manipulation results, executes commands and provides feedback. This allows exchangeable and extensible *system control* techniques. Handling the menu interface this way, speech input can easily be managed too.

Another practical use case of bidirectionality is the definition of constraints within the application. In *modeling tasks*, constraining 3D (6DOF) input is important for precise input. Mine [11] and other authors (e.g. [3]) suggest in various studies that for direct input in 3D space six degrees of freedom are not expedient most of the time. Therefore it is very reasonable to restrict the user's input to two dimensions, for instance by using supporting planes. Positional and orientation data can be restricted independently. The defining parameters of a plane can be transferred to the tracking framework to establish 3D input coordinate restriction (e.g. snapping to that plane).

6.3 Lack of Support

Unfortunately, some 3D UI techniques have to remain being completely implemented on the application level and tracking middleware can not help with certain subtasks. ITs exhibiting strong coupling with objects or their virtual representation are more likely to fall in this category.

One example of this would be: The Voodoo dolls technique [15] requires access to objects, which are defined in the application. Therefore any processing of these objects can only be handled on the application level.

Image plane ITs [14] might be too difficult to implement in middleware as well because of the same reasons. An application-only implementation seems to be more practical, as even gesture processing IT subtasks are strongly related to virtual objects.

The same applies to other techniques like the Magic Mirror [10] or the Through The Lens technique [21]: Viewport rendering for graphics mapped onto interaction devices has to reside on the application level. Coordinate space transformation calculations are rather complex and not practical to implement in middleware.

7 CONCLUSION

We demonstrate three approaches and implementation techniques to extend the functionality of tracking middleware. By extending OpenTracker to allow Python scripting, a wide range of possible applications emerge: We are currently establishing a central repository of common ITs. With such a repository dynamic IT adaptation can be easily achieved by utilizing network communication.

Rapid prototyping of 3D interaction techniques is important for testing, studying and evaluating user performance and usability of MR applications. Our scripting approach enables testing of new experimental techniques without changing software components of the application or the VR framework.

Strong interoperability between OpenTracker and VRPN takes advantages of both tracking frameworks: Numerous tracking devices are supported by both and ready for use. Many different processing features such as data filtering and merging are provided. Transformations and more complex tasks (e.g. some ITs) can be applied to input data (as described in section 5).

ITs written once can be reused by others in their own applications as long as the application itself or the related MR framework uses or supports VRPN or OpenTracker as tracking middleware. In this context the idea of an IT repository appears to be very interesting again in order to build up a library of various 3D UI techniques.

8 FUTURE WORK

A survey in 1992 [13] showed that about 50% of application development code (and time) was used for the applications' user interfaces. From our experience with developing complex MR environments, we believe that this may hold true for MR applications as well. Decoupling user interface code from the application could be a first step towards standardizing building blocks (widgets) which should be independent of the application or the higher level VR framework. Since most application developers are using tracking middleware in order to get hardware support for interaction devices (without having to implement it themselves), tracking middleware might serve as a common ground for future user interaction standards.

We think that by using an approach as described in this paper, most code regarding menu system interaction can be decoupled from the application and handled by the tracking middleware. A standardized XML file, specifying the menu layout and widgets could be passed by the application onto the tracking middleware which then sends higher level commands back to the application in case of user interaction. Widget behaviour itself could all be handled by the tracking framework if desired. Another advantage of this approach is obviously consistent application behaviour in distributed MR scenarios with minimal network load. We plan to continue our work in this area.

Furthermore, we believe that building IT repositories is a good way to move on towards the goal of a flexible and adaptive tracking system. In a pervasive MR scenario ITs must be accommodated to local hardware setups. The scripted approach is most appropriate for choosing a suitable IT for each setup and location in a dynamic adaptation process.

ACKNOWLEDGEMENTS

We thank all developers of OpenTracker, especially Gerhard Reitmayr, for their excellent work throughout the years. This work was funded in part by the Austrian Science Fund FWF project P19265.

REFERENCES

[1] D. M. Beazley. SWIG: an easy to use tool for integrating scripting languages with C and C++. In *4th Annual Tcl/Tk Workshop '96, July 10–13, 1996. Monterey, CA*, pages 129–139, Berkeley, CA, USA, 1996. USENIX.

[2] S. Beckhaus, K. J. Blom, and M. Haringer. Intuitive, hands-free travel interfaces for virtual environments. In *VR2005: Proceedings of the Workshop "New directions in 3D User Interfaces"*, pages 57–60. Shaker Verlag, 2005.

[3] D. A. Bowman. *Interaction techniques for common tasks in immersive virtual environments: design, evaluation, and application*. PhD thesis, 1999. Adviser-Larry F. Hodges.

[4] D. A. Bowman and L. F. Hodges. An evaluation of techniques for grabbing and manipulating remote objects in immersive virtual environments. In *SI3D '97: Proceedings of the 1997 symposium on Interactive 3D graphics*, pages 35–ff., New York, NY, USA, 1997. ACM Press.

[5] D. A. Bowman, E. Kruijff, J. J. LaViola, and I. Poupyrev. *3D User Interfaces: Theory and Practice*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 2004.

[6] T. L. Cottom. Using SWIG to bind C++ to Python. *Computing in Science and Engineering*, 05(2):88–96, c3, 2003.

[7] P. Figueroa, R. Dachselt, and I. Lindt. A conceptual model and specification language for mixed reality interface components. In *VR '06: Proceedings of the Workshop "Specification of Mixed Reality User Interfaces: Approaches, Languages, Standardization"*, pages 4–11, Washington, DC, USA, 2006. IEEE Computer Society.

[8] P. Figueroa, M. Green, and H. J. Hoover. InTml: a description language for VR applications. In *Web3D '02: Proceeding of the seventh international conference on 3D Web technology*, pages 53–58, New York, NY, USA, 2002. ACM Press.

[9] A. Forsberg, K. Herndon, and R. Zeleznik. Aperture based selection for immersive virtual environments. In *UIST '96: Proceedings of the 9th annual ACM symposium on User interface software and technology*, pages 95–96, New York, NY, USA, 1996. ACM Press.

[10] J. Grosjean and S. Coquillart. The magic mirror: A metaphor for assisting the exploration of virtual worlds. In *Proceedings of 15th Spring Conference on Computer Graphics*, pages 125–129, 1999.

[11] M. R. Mine. Working in a virtual world: Interaction techniques used in the chapel hill immersive modeling program. Technical Report TR96-029, UNC Chapel Hill, Chapel Hill, NC, USA, 1996.

[12] M. R. Mine, J. Frederick P. Brooks, and C. H. Sequin. Moving objects in space: exploiting proprioception in virtual-environment interaction. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 19–26, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[13] B. A. Myers and M. B. Rosson. Survey on user interface programming. In *CHI '92: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 195–202, New York, NY, USA, 1992. ACM Press.

[14] J. S. Pierce, A. S. Forsberg, M. J. Conway, S. Hong, R. C. Zeleznik, and M. R. Mine. Image plane interaction techniques in 3D immersive environments. In *SI3D '97: Proceedings of the 1997 symposium on Interactive 3D graphics*, pages 39–ff., New York, NY, USA, 1997. ACM Press.

[15] J. S. Pierce, B. C. Stearns, and R. Pausch. Voodoo dolls: seamless interaction at multiple scales in virtual environments. In *SI3D '99: Proceedings of the 1999 symposium on Interactive 3D graphics*, pages 141–145, New York, NY, USA, 1999. ACM Press.

[16] I. Poupyrev, M. Billinghurst, S. Weghorst, and T. Ichikawa. The go-go interaction technique: non-linear mapping for direct manipulation in VR. In *UIST '96: Proceedings of the 9th annual ACM symposium on User interface software and technology*, pages 79–80, New York, NY, USA, 1996. ACM Press.

[17] G. Reitmayr and D. Schmalstieg. An open software architecture for virtual reality interaction. In *VRST '01: Proceedings of the ACM symposium on Virtual reality software and technology*, pages 47–54, New York, NY, USA, 2001. ACM Press.

[18] G. V. Rossum. *Python reference manual*. Stichting Mathematisch Centrum, Amsterdam, The Netherlands, 1995.

[19] C. Shaw, M. Green, J. Liang, and Y. Sun. Decoupled simulation in virtual reality with the MR toolkit. *ACM Trans. Inf. Syst.*, 11(3):287–317, 1993.

[20] R. Stoakley, M. J. Conway, and R. Pausch. Virtual reality on a WIM: interactive worlds in miniature. In *CHI '95: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 265–272, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co.

[21] S. L. Stoev, D. Schmalstieg, and W. Straßer. Two-handed through-the-lens-techniques for navigation in virtual environments. In *Proceedings of the Eurographics Workshop on Virtual Environments*, pages 51–60, 16-18 May 2001.

[22] R. M. Taylor, T. C. Hudson, A. Seeger, H. Weber, J. Juliano, and A. T. Helser. VRPN: a device-independent, network-transparent VR peripheral system. In *VRST '01: Proceedings of the ACM symposium on Virtual reality software and technology*, pages 55–61, New York, NY, USA, 2001. ACM Press.