



DISSERTATION

Geometry Education with Augmented Reality

ausgeführt zum Zwecke der Erlangung des akademischen Grades
eines Doktors der technischen Wissenschaften

unter Leitung von

Univ.-Prof. Dipl.Ing. Dr. Dieter Schmalstieg
am Institut für Softwaretechnik und Interaktive Systeme

eingereicht an der Technischen Universität Wien von

Mag. Hannes Kaufmann

Ospelgasse 1-9/1/39

1200 Wien

Matr.-Nr. 9225889

kaufmann@ims.tuwien.ac.at

Wien, im März 2004

Unter allen Mathematikern hat der Geometer den Vorteil
zu sehen,
was er studiert.

Felix Klein (1849-1925)

Doctoral Committee

Chair: Univ.-Prof. Dr. Dieter Schmalstieg
Vienna University of Technology

Reviewers: Univ.-Doz. Dr. Michael Wagner
Danube University Krems

Dr. Mark Billingham
HIT Lab New Zealand, University of Canterbury

Abstract

To fill the gap of next-generation user interfaces for mathematics and geometry education Construct3D is introduced a three-dimensional dynamic geometry construction tool that can be used in high school and university education. This system uses 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. By working directly in 3D space, complex spatial problems and spatial relationships may be comprehended better and faster than with traditional methods.

After a description of Construct3D's design various hardware setups are presented that are suitable for educational purposes. Immersive, semi-immersive, mobile and hybrid setups are studied for their applicability to geometry education. An immersive setup that uses head mounted displays is most favored by teachers and students. It allows users to actually "walk around" geometric objects which are fixed in space.

In order to adapt software and hardware to users' needs user interfaces were redesigned and in depth research was done on usability design. Development steps and results present how Construct3D's look and feel was considerably improved.

A wide range of selected geometric content from basic and advanced high school geometry to university education is presented. Diverse examples demonstrate the potential and robust constructive capabilities of the dynamic geometry application.

Finally results from two evaluations show that Construct3D is easy to use, requires little time to learn, encourages learners to explore geometry and can be used in a consistent way.

At the end an outlook is given on ongoing and future work and basic guidelines to developers of educational VR/AR applications are summarized.

Kurzfassung

Construct3D ist eine Geometrieapplikation zur Konstruktion dreidimensionaler dynamischer Geometrie für den schulischen und universitären Geometrieunterricht. Unter Verwendung von Augmented Reality (AR) – erweiterter Realität – ist die natürliche Zusammenarbeit zwischen Lehrenden und Lernenden möglich. Der Hauptvorteil bei der Benützung von AR liegt darin, dass Schüler dreidimensionale Objekte auch dreidimensional sehen und wahrnehmen können, die sie bisher nur berechnen oder traditionell in Normalrissen konstruieren konnten. Durch die direkte Arbeit mit nahezu greifbaren virtuellen Objekten im dreidimensionalen Raum können räumliche Probleme und Beziehungen der Objekte zueinander vermutlich schneller erfasst werden als es mit bisherigen Mitteln möglich war.

Nach einer Beschreibung der Funktionalität von Construct3D werden verschiedene Hardwarekonfigurationen vorgestellt, die sich zum Unterrichten mit Construct3D eignen. Immersive, halb immersive, mobile und gemischte Konfigurationen werden auf ihre Verwendbarkeit im Unterricht untersucht. Ein immersives Setup, das Datenbrillen verwendet, wird von Lehrern und Schülern bevorzugt. Es gibt Benutzern die Möglichkeit, direkt um das dreidimensional Objekt zu gehen, um es von verschiedenen Seiten zu betrachten.

Um Software und Hardware an die Bedürfnisse der Benutzer anzupassen wurde das Design von Benutzerschnittstellen überarbeitet und umfassende Überlegungen zur Verbesserung der Benutzerfreundlichkeit angestellt. Verschiedene Entwicklungsschritte und Ergebnisse beschreiben, wie die Arbeit und das Aussehen von Construct3D erheblich verbessert wurde.

Eine breite Palette geometrischer Beispiele aus verschiedensten Bereichen, sowohl für die schulische als auch die universitäre Geometrieausbildung, werden präsentiert. Die ausgewählten Beispiele demonstrieren das Potential und die konstruktiven Möglichkeiten, die das dynamische Geometrieprogramm bietet.

Die Resultate von zwei Evaluationen zeigen, dass Construct3D einfach zu bedienen ist, die Bedienung wenig Lernaufwand erfordert, Lernende ermutigt, Geometrie experimentell zu erfahren, und sich mit einheitlichen Prinzipien bedienen lässt.

Schlussendlich werden grundsätzliche Richtlinien für Entwickler von unterrichtsbezogenen VR/AR-Anwendungen zusammengefasst. Ein Ausblick auf aktuelle und zukünftige Forschungsarbeit wird gegeben.

Acknowledgements

I would like to thank all people who helped me in finishing this thesis.

First of all, I sincerely thank my parents Helga and Herbert Kaufmann for their love and for continuous support, be it financial or moral, during my whole studies up to the PhD. From the beginning they gave me the freedom and time to learn and to explore.

This work would not be here yet without the ongoing encouragement and support of Menega Sabidussi. She also contributed to the development of Construct3D with her excellent color design and her design of the wireless pen. I thank her very much.

I thank my advisor Dieter Schmalstieg for guiding me and for teaching me scientific work principles. In addition to Dieter I want to thank Christian Breiteneder for being a mentor and for giving me the freedom to pursue own research interests. With their support and their invaluable experience they taught me how to write project proposals and gave me the key to doing further research.

Throughout the years I shared my office with Gerhard Reitmayr who was main developer of Studierstube and OpenTracker for many years. I thank him for helping me with many programming related problems.

Many thanks go to all members of the VR group at the Vienna University of Technology, especially to Istvan Barakonyi for his ongoing help, to Tamer Fahmy for his work on Coin, to Thomas Psik for work on the wireless pen and the new widget implementation, to Florian Ledermann for his work on the Augmented Classroom and to Thomas Pintaric for his support at various events. I also thank Gerd Hesina for his help in my first year. Without their contributions to Studierstube this work would not have been possible.

Some features of Construct3D described in this dissertation are also the result of the work of undergraduate students. The ACIS integration, implementation of curves and surfaces, intersections, undo/redo and many other features have been implemented by Klaus Feiler. I thank him very much for his dedication and help. Reinhard Steiner implemented the speech interface for OpenTracker and Studierstube.

Throughout the years many colleagues came and went. For inspiring discussions I thank Anton Fuhrmann, Rainer Splechna, Klaus Dorfmueller and Erich Pohn. Complex multi-user setups need to be set up, configured and maintained. Therefore I thank our technicians Johannes Fromm and Albert Walzer for their help.

Cooperation with the Institute of Psychology at University Vienna and the resulting research project was only possible due to the trust of Judith Glück in our work and

an excellent cooperation with her. I thank her and Karin Steinbügl for their psychological work on Construct3D which is partly integrated into this thesis. Thanks to Andreas Dünser and Alexandra Totter for reading parts of the thesis.

The roots of this thesis are founded in my study of descriptive geometry at the Institute of Geometry at Vienna University of Technology. I thank Prof. Hellmuth Stachel and Prof. Helmut Pottmann for ongoing cooperation and Boris Odenahl for his help with content development.

Both teachers in our second evaluation, Werner Scharf but especially Andreas Asperl, contributed to the development of new content for Construct3D.

Since the beginning of my PhD studies I was member of the “ADI GZ/DG”, a group of 10 Austrian geometry teachers developing innovative geometric content. I thank all of them for letting me be part, for sharing their ideas, for inspiring discussions during our meetings and at the conferences in Strobl and for shaping geometry education in Austria. Without their encouraging and idealistic work and without the spirit of teachers gathering in Strobl, geometry education in Austria would be much different today. I especially thank my former teacher Georg Schilling for raising my interest in geometry.

Finally I thank my secondary advisor Michael Wagner for his support throughout the years and for sharing inspiring ideas.

Part of this research was funded by the Austrian Science Fund (FWF) contract no. P14470, P16803 and Y193, by the EU IST project Lab@Future (IST-2001-34204) and by Vienna University of Technology “Innovative Projekte 2002”.

Thank you all.

Table of Contents

1	INTRODUCTION	1
1.1	MOTIVATIONS AND PROBLEM STATEMENT	1
1.2	CONTRIBUTION.....	3
1.3	INDIVIDUAL PUBLICATIONS ABOUT THIS WORK	3
1.4	NEW AND ONGOING RESEARCH PROJECTS	4
2	RELATED WORK AND THEORETICAL FOUNDATIONS.....	5
2.1	AUGMENTED REALITY	5
	<i>Collaborative Augmented Reality</i>	6
	<i>Mobile Augmented Reality</i>	7
2.2	3D MODELING VR/AR APPLICATIONS	7
2.3	EDUCATIONAL VR/AR APPLICATIONS	9
	<i>Science Education</i>	9
	<i>Zoology</i>	10
	<i>Biology</i>	11
	<i>Mathematics</i>	11
2.4	GEOMETRY EDUCATION - EDUCATIONAL GEOMETRY SOFTWARE.....	12
	<i>Geometry Education in Austria</i>	13
	<i>Modern Ways of Teaching Geometry</i>	13
	<i>Dynamic 2D Geometry Software</i>	14
	<i>Dynamic 3D – Parametric Computer Aided Design</i>	16
2.5	PEDAGOGIC THEORY	17
2.6	PSYCHOLOGICAL ASPECTS: SPATIAL COGNITION	17
	<i>The Structure of Spatial Abilities</i>	18
	<i>Training Spatial Ability</i>	22
3	TECHNOLOGICAL FOUNDATIONS	27
3.1	STUDIERSTUBE	27
	<i>Open Inventor</i>	28
	<i>3D Event System</i>	29
	<i>Widget System</i>	29
	<i>Dynamic Application Loading</i>	30
	<i>Single Host - Multiple Users, Multiple Displays</i>	31
	<i>Distributed Inventor</i>	32
3.2	OPENTRACKER	33
3.3	GEOMETRIC MODELING KERNEL.....	33
	<i>Open CASCADE</i>	34
	<i>ACIS</i>	35

4	CONSTRUCT3D	37
4.1	SOFTWARE DESIGN	37
	<i>Overview.....</i>	37
	<i>Selection – Action.....</i>	39
	<i>Boolean Operations.....</i>	40
	<i>Intersection Curves.....</i>	41
	<i>Layers.....</i>	41
	<i>Top View, Front View, Side Views</i>	42
	<i>VRML Export and Import.....</i>	43
	<i>Undo and Redo in Multi-User Applications.....</i>	43
	<i>Collaboration Modes.....</i>	44
	<i>Distributed Construct3D</i>	45
4.2	IMPLEMENTATION	45
	<i>Class Hierarchy.....</i>	45
	<i>Object Creation.....</i>	47
	<i>Overall Scene Graph Structure</i>	47
	<i>Object Dependencies.....</i>	48
	<i>Object Deletion.....</i>	48
	<i>Serializing Intersection Operations.....</i>	49
	<i>Continuity for Dynamic 3D Geometry.....</i>	50
	<i>Implementation of Undo and Redo.....</i>	50
	<i>Object Manipulation.....</i>	51
	<i>Cross-Platform Development</i>	51
4.3	FILE FORMAT DESCRIPTION	52
5	EDUCATIONAL SCENARIOS.....	54
5.1	IMMERSIVE SETUP	54
	<i>Basic AR Lab Multi-User Setup</i>	54
	<i>Hybrid AR Classroom</i>	55
5.2	SEMI-IMMERSIVE SETUPS.....	56
	<i>Projection Screen Classroom</i>	56
	<i>Baron Table Setup.....</i>	57
5.3	HYBRID DESKTOP SETUPS	57
	<i>Distributed Hybrid Classroom</i>	57
	<i>Basic Desktop Interface with Speech Input</i>	60
	<i>Remote Collaboration</i>	61
5.4	MOBILE SETUPS	61
	<i>Mobile “YO!Einstein” Demonstration Setup.....</i>	62
	<i>Augmented Classroom - Mobile Collaboration.....</i>	64
5.5	LAB@FUTURE EVALUATION SETUP	66

6	USER INTERFACES AND USABILITY DESIGN.....	70
6.1	FROM MAGNETIC TO OPTICAL TRACKING.....	71
6.2	CYBERSICKNESS.....	73
	<i>Tracking Errors and Lag</i>	74
	<i>HMD Setup with Helmets</i>	75
	<i>Stereoscopic Viewing</i>	76
6.3	EXACT VERSUS DYNAMIC CONSTRUCTION.....	77
6.4	USER INTERFACE.....	79
	<i>The Personal Interaction Panel (PIP)</i>	79
	<i>Redesign of the Pen</i>	80
6.5	USABILITY DESIGN.....	82
	<i>Transparency</i>	82
	<i>Color Coding</i>	84
	<i>Improving User Interaction: Highlighting and Preview</i>	85
7	CONTENT DEVELOPMENT	89
7.1	EXAMPLES FOR DYNAMIC 3D GEOMETRY.....	89
	<i>Content for the Second Evaluation</i>	89
	<i>Tschupik-Cubes</i>	90
	<i>Surface of Revolution</i>	92
	<i>Intersection Curve of Two Cylinders</i>	93
	<i>Tangents in Points of Intersection Curves</i>	94
	<i>Centers of Gravity</i>	95
	<i>Content for Advanced Geometry Education</i>	96
	<i>Deflection Sheave</i>	96
	<i>Conic Sections – Proof of DANDELIN</i>	98
	<i>One-Sheeted Hyperboloid</i>	100
	<i>Villarceau’s Circles</i>	101
	<i>Two-Dimensional Geometry</i>	101
	<i>Spatial Interpretation of Planar Geometry</i>	102
7.2	TEACHING EXPERIENCES AND PEDAGOGIC CONSEQUENCES	106
7.3	LEARNING MODES.....	107
7.4	APRIL PRESENTATION AND INTRODUCTION	108
	<i>APRIL</i>	108
	<i>Construct3D Introduction</i>	108
	<i>Remarks</i>	110
8	EVALUATIONS.....	111
8.1	FIRST EVALUATION	111
	<i>First Version of Construct3D</i>	111
	<i>Subjects</i>	112
	<i>Methods</i>	112

	<i>Results</i>	113
	<i>Informal Evaluations and Trial Runs with High School Students</i>	115
8.2	PEDAGOGICAL BACKGROUND	116
8.3	SECOND EVALUATION	119
	<i>Study Design</i>	119
	<i>Results</i>	121
	<i>Discussion</i>	126
8.4	CONCLUSIONS FOR A PSYCHOLOGICAL EVALUATION OF CONSTRUCT3D	131
	<i>Conclusions and Research Questions for the Evaluation Study</i>	131
	<i>Conclusions for Training Design</i>	132
	<i>Planned Study Design</i>	132
9	CONCLUSION	134
9.1	FINDINGS	134
9.2	FUTURE WORK	136
	<i>Additional Features</i>	136
	<i>The Continuity Problem in 3D</i>	137
	<i>Desktop Interface</i>	137
	<i>Upcoming Evaluations</i>	138
	APPENDIX A: SECOND EVALUATION QUESTIONNAIRE	139
	BIBLIOGRAPHY	154

1 Introduction

1.1 Motivations and Problem Statement

During his studies of mathematics and geometry with the aim of becoming a teacher, the author gave countless private lessons to students of those subjects. In order to solve three dimensional mathematical but especially geometrical problems, spatial abilities are an important prerequisite [21, 39, 54]. Many students have difficulties solving tasks that require spatial visualization skills and spatial thinking. To get passing grades they use strategies such as learning construction steps by heart without fully understanding spatial problems and their solutions in 3D space.

The author's wish of pursuing a PhD was driven by the desire to help students to develop correct mental models of three dimensional problems and to improve their spatial thinking - to enable them to find solutions to geometric problems themselves. The hypothesis for our current and ongoing work is that if students see three-dimensional objects directly in 3D space and can *interactively* construct, touch and modify abstract geometric objects, they build mental models of complex geometric situations and of geometric properties more easily in real life. This supports spatial reasoning and helps to solve mathematic and geometric problems.

Based on psychological studies about mathematics and geometry education, national school authorities (e.g. Austria [2]) 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 subfactors, such as mental rotation, visualization, and environmental orientation. Many studies have shown that spatial abilities can be improved by well-designed training (e.g. [145]; see overview in section 2.6). Geometry education has proven to be a powerful means of improving these skills [45].

Recently with the emergence of Virtual Reality (VR) and related 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 Virtual Reality (VR) in training spatial ability [36, 129]. However, little to no work has been done towards systematic development of VR applications for practical education purposes in this field. This thesis aims to introduce Augmented Reality (AR), a technology closely related to VR, to mathematics and geometry education. The simultaneous sharing of real and virtual space in AR is an ideal match for computer-assisted collaborative education 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 (Figure 1). Augmented reality provides them with an almost tangible picture of complex three dimensional objects and scenes. It enhances, enriches and complements the mental pictures that students form in their minds when working with three dimensional objects.

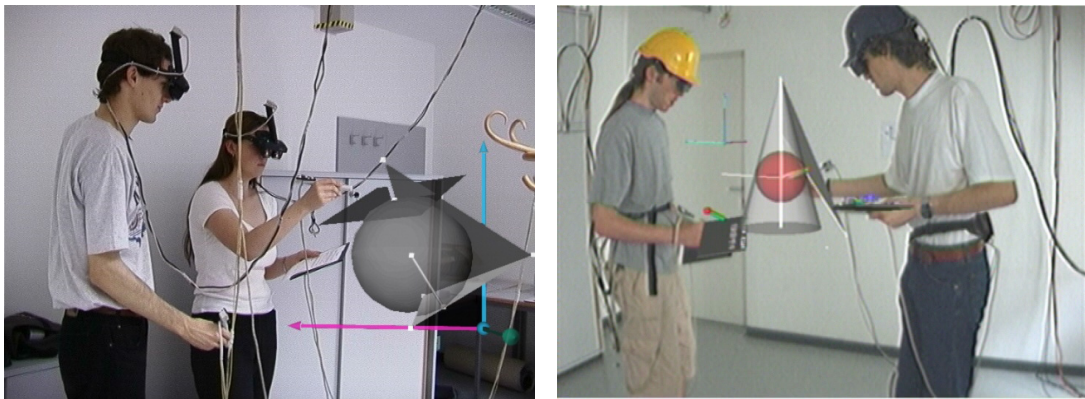


Figure 1: Students work with Construct3D.

However, there are a number of requirements for an augmented reality tool with the aim of effectively improving spatial skills. These have never been addressed by existing systems, nor studied in an educational context.

- No VR/AR application exists that offers the flexibility to construct and modify - to dynamically construct geometric content directly in 3D space (see section 2.4).
- No VR/AR application for actual use in high school or higher education has ever been developed with the main purpose of improving spatial skills.
- Hardly any evaluations can be found in literature which give hints to the actual learning transfer from a VR/AR learning environment to the real world.

1.2 Contribution

The work presented in this thesis summarizes our ongoing efforts towards filling these gaps.

Chapter 2 gives an overview of related work in all related research areas – collaborative Virtual and Augmented Reality systems, 3D modeling VR/AR applications and educational geometry applications, pedagogical theory and psychological theory.

In chapter 3 and 4 we outline the used software components and implementation of our system whereas in chapter 5 we specify various different hardware setups that were developed and tested for their applicability in classroom use.

General and specific guidelines about developing educational applications, about usability and user interface issues are summarized in chapter 6. This chapter is a recommendation for researchers who plan to do similar work on educational and/or 3D modeling applications.

The innovative content for teaching three dimensional dynamic geometry which we developed for actual use in geometry classes is outlined in chapter 7. Chapter 8 concludes with two informal evaluations that were conducted in an early and late phase of the development process.

For conclusions and planned future work please refer to chapter 9.

This thesis is not the end of our efforts to integrate AR technology into geometry education, it can rather be seen as the beginning. Recently a national research project got funded which allows us to conduct a comprehensive evaluation study in order to study the general and differential effects of training on several components of spatial ability. Mid- to long-term plans are to integrate Construct3D and the concept of teaching geometry with Augmented Reality in high school and higher education by collaborating with Austrian schools and external partners such as the Institute of Geometry at Vienna University of Technology.

1.3 Individual Publications about this Work

Results of this work have been published previously by the author. The following publications describe the preliminary outcome of the work:

H. Kaufmann, D. Schmalstieg, and M. Wagner, "Construct3D: a virtual reality application for mathematics and geometry education," *Education and Information Technologies*, vol. 5, pp. 263-276, 2000.

B. Greimel, A. Fuhrmann and H. Kaufmann, "Virtual Reality as an Innovative Setting for Simulations in Education". In *Proceedings of the World Association for Case Method Research (WACRA) conference 2001*, 2001.

H. Kaufmann, "Dynamische Geometrie in Virtual Reality," *Informationsblätter der Geometrie (IBDG)*, vol. 21, pp. 33-37, 2002.

D. Schmalstieg, H. Kaufmann, G. Reitmayr, A. Fuhrmann, R. Wegenkittl and R. Splechtna, „Spontaneous Collaboration in the Augmented Classroom”, *SIGGRAPH 2002 Emerging Technologies Proposal*, Technical Report, 2002.

H. Kaufmann and D. Schmalstieg, "Mathematics And Geometry Education With Collaborative Augmented Reality," *SIGGRAPH 2002 Educators Program. In SIGGRAPH 2002 Conference Abstracts and Applications*, pp. 37-41, 2002.

D. Schmalstieg, H. Kaufmann, G. Reitmayr, and F. Ledermann, "Geometry Education in the Augmented Classroom," Reviewed scientific demonstration at the IEEE and ACM International Symposium on Mixed and Augmented Reality 2002, TR-188-2-2002-14, 2002.

H. Kaufmann, "Construct3D: An Augmented Reality Application for Mathematics and Geometry Education," Video Description in Proceedings of ACM Multimedia Conference 2002, TR-188-2-2002-24, 2002.

H. Kaufmann, "Collaborative Augmented Reality in Education," Monte Carlo, Monaco, position paper for keynote speech at Imagina 2003 conference, Feb. 3rd, 2003.

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

1.4 New and Ongoing Research Projects

A number of research projects could be acquired with the help of Construct3D and parts of the work presented in this thesis have been published in proposals and publications of these projects.

Lab@Future, EU IST Project IST-2001-34204 coordinated by Davarakis C. (Systema Informatics, Greece), May 2002 – May 2005

Educating Spatial Intelligence with Augmented Reality, FWF Project P16803 coordinated by Breiteneder, C., Interactive Media Systems Group, TU Vienna in cooperation with the Institute of Psychology, University Vienna. Dec. 2003 – Dec. 2005.

Innovative Projekte 2002: “Collaborative Augmented Reality für den Einsatz in der Lehre”. A joint project between the Institute of Computer Graphics, the Interactive Media Systems Group and the Institute of Geometry of the TU Vienna. 2002-2004.

2 Related Work and Theoretical Foundations

The work presented in this thesis is based on a combination of four distinct areas of research (Figure 2).

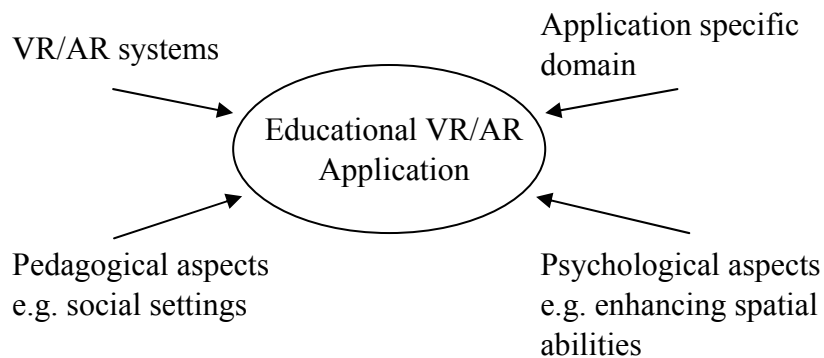


Figure 2: Combination of research areas in the case of Construct3D.

Our educational AR application is influenced by research work done on Augmented Reality systems, educational geometry software, geometry education and pedagogical aspects and last but not least by psychological aspects. In this chapter we will present selected related work from each of these areas.

In order to put Construct3D in its right context we chose related work of each area that was influential in the design of our educational augmented reality application and helps to grasp the complexity of interrelations between the mentioned areas. We also give an overview of work done on educational AR and VR applications since the beginnings in the early 90's until today.

2.1 Augmented Reality

A good definition of Augmented Reality (AR) is given in the survey by Azuma [5]. According to this definition, Augmented Reality is a variation of Virtual Reality

(VR). VR technology completely immerses a user inside a synthetic environment. While immersed, the user cannot see the surrounding real world. In contrast, AR allows the user to see the real world, with virtual objects superimposed upon or composited with the real world. Therefore, AR supplements reality, rather than completely replacing it. Ideally, it would appear to the user that the virtual and real objects coexist in the same space. AR can be thought of as the „middle“ ground between VR (completely synthetic) and telepresence (completely real) [107]. In terms of used technology, AR can be said to require the following three characteristics:

1. Combines real and virtual
2. Interactive in real time
3. Registered in 3D

This definition encompasses a variety of possible setups, including monitor-based interfaces, see-through head mounted displays (HMDs), projection displays, and various other combining technologies. As a user interface technique, AR provides the unique opportunity of adding computer generated information in-place rather than separated from a user's focus of interest in the real world. Therefore, AR systems have tremendous potential in areas such as live medical visualization overlaid onto real patients [6], construction and maintenance procedures (e. g., replacing printed manuals with superimposed information) [27], navigational and tourist assistance, robot and machinery operation, aircraft piloting (heads-up display), and so on.

Collaborative Augmented Reality

One of the most important purposes of an educational environment is to promote social interaction among users located in the same physical space [134]. This is supported by collaborative AR.

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 [13] or remote collaboration [24, 63].

Early examples of collaborative augmented reality were shown in the Shared Space project [10] which also demonstrated the use of AR for remote teleconferencing [11, 12] or support of same-space collaboration. Another work was EMMIE [24] that demonstrated a shared workspace enriched with virtual objects. It focused on managing the environment for different display modalities and users. The combination of mobile AR and remote collaboration between a mobile user and a stationary workspace was later investigated in the MARS project [63].

The Studierstube project [138] puts an emphasis on supporting collaboration in shared augmented reality workspaces [139]. Based on an underlying distribution

mechanism [57] 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 [137].

Mobile Augmented Reality

Augmented Reality and mobile computing are often mentioned together, as many mobile computing platforms rely on some kind of head-up or head-mounted display to provide continuous access to information, often coupled with hands-free operation. The ultimate goal is to make the mobile computer a part of the user's normal daily life [14]. Augmented Reality as a user interface for mobile computing is particularly powerful when the computer has access to information on location and situation, so that it can provide contextual information. A prominent example is Columbia's "Touring Machine", which is used as a mobile multimedia information system [38] and journalistic tool [62]. The follow up developments of the Mobile Augmented Reality System (MARS) [63] and situated documentaries [62] further explored the user interface aspects of such systems and potential applications to interactive presentations for tours through a university campus.

Applications of augmented reality to navigation were demonstrated by Thomas et al. [155]. The further development of the Tinmith system demonstrated the applicability to entertainment [154] and investigated the use of mobile AR for directly constructing models of real objects in place [115].

Recent developments focus on applying mobile AR interfaces to real applications to be deployed to end users. An interesting work is the Geist [80] project that aims to envelope users in an interactive story situated in real places throughout the historic center of Heidelberg.

Reitmayr et al. [124] built a series of mobile AR setups to investigate mobile collaborative applications [126] and implemented various applications to demonstrate the versatile usage of their system (also documented in [123]).

For a comprehensive review of literature in the area of AR is referred to [5, 123, 138].

2.2 3D Modeling VR/AR Applications

A large body of work has been done on 3D modeling in general. Although 3D input devices with six degrees of freedom (6DOF) have been used to enhance modelers, little modeling has been done in immersive virtual reality systems. In the following we limit the discussion to HMD based systems. Only minimally immersive VR systems (fishtank VR) or fully immersive VR systems, but no pure desktop systems are discussed here.

Some previous uses of HMD based systems have concentrated more on exploration of virtual worlds rather than creating or modifying them directly in virtual reality [20, 41]. A very good overview of 3D modeling systems with 6DOF input devices can be found in the work of Mine [109].

One of the earliest interactive design systems that used an immersive head-mounted display was the one built by Clark [30] in 1976. This pioneering system for use in the interactive design of free-form surfaces addressed many of the issues that developers of interactive design systems face today.

Liang and Green [95] developed JDCAD, an interactive 3D modeling system with a non-stereo head tracked display and a single 6 DOF input device. Their focus was an improvement in ergonomics and precision for engineering tasks. Development of JDCAD system was done at the University of Alberta and was continued on JDCAD+ with the addition of new animation editing and scene composition functions.

One of the few HMD based modeling systems called 3DM was presented by Butterworth et al. [23]. It includes several grid and snap functions, an extrusion tool for surface creation and some other interesting features. It lacks, however, many of the other aids and constraints that since have been found necessary for accomplishing precise work, as rated by Mine [109], who presented the Chapel Hill Immersive Modeling Program CHIMP. It is a test bed for various interaction techniques like its precursor ISAAC [109, 110].

These systems belong to a number of 3D modeling and design systems studying user interaction techniques. They are also used in university education with design or architecture students. DesignSpace by Chapin [28] is one of them. Another example is Bowman's Conceptual Design Space [16] – a real-time, interactive virtual-environment application which attempts to address the issue of 3D design in general and immersive design in particular. The Virtual Reality Aided Modeler (VRAM) [122] by the faculty of architecture at the Bauhaus-University Weimar, is an ongoing test bed for the application of 3D user interface techniques on a conceptual design tool for architects and industrial designers that runs with VRML97. SeamlessDesign by Kiyokawa [78] is another system with similar goals.

SmartSketches [136] is a modeler developed in a currently running EU Project (<http://sketch.inesc-id.pt/>). The goal is to develop innovative multimodal user interfaces combining sketches, gestures and speech among others, to accomplish design tasks in different contexts, ranging from small tablets to large-scale displays and immersive environments. User studies and task analysis will be conducted to identify critical bottlenecks in present-day conditions.

DIVEdit [148] is a prototype *collaborative* object modeler for virtual environments. It is implemented as an application in the DIVE system [42] for research on distributed virtual environments. DIVEdit combines collaborative solutions and immersive shaping, making an interactive, collaborative object

modeler. Preliminary user tests have been made in order to identify some questions and problems related to teamwork in virtual worlds.

A very comprehensive overview of nearly all existing 3D modeling systems and virtual reality supported conceptual design tools can be found on the web page of Ernst Kruijff [81]. For a comprehensive overview of interaction techniques in virtual environments we refer to [15] and [17].

2.3 Educational VR/AR Applications

Since the early 1990th researchers have been working on virtual reality applications for purely educational use ([8, 25, 97, 166] and many others). We want to point out some interesting educational applications in chronological order starting with the earliest ones.

Bricken and Byrne [20] worked with 60 students aged 10-15 years in a summer camp for 7 weeks and taught them to build their own 3D worlds with traditional 3D modelers. All students had regular access to the university VR lab to test their worlds. Video observations and informal surveys were conducted.

A very interesting project from the area of language learning is Zengo Sayu by Rose and Billingham [131]. By exploring a three dimensional world students learn Japanese. Building blocks can be moved in relation to other blocks. Japanese words for simple navigation such as “under, above, below, behind,...” can be studied that way. The system features speech recognition and recorded speech output.



Virtual Reality Roving Vehicles [130] is designed as an outreach program to bring VR to children in schools. Researchers put all VR hardware in a van and travel to schools. Students (age 10-18) get the chance to experience VR. A limited number of students learn how to create a virtual setting themselves. The goal is to teach children to build virtual worlds using constructivist learning theory, to help them understand and meet specific learning goals. The research mission is to test VR as a medium for making the teaching process transparent, so students can focus on content rather than falter with the mechanics of instruction.

Science Education

Water on Tap [25] 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 can also be selected. The next series of worlds was also done to explore the strengths and limits of virtual reality for science education. ScienceSpace [34] is a collection of immersive virtual worlds Newtonworld, MaxwellWorld and PaulingWorld. NewtonWorld provides an environment for investigating kinematics and dynamics of one-dimensional motion. MaxwellWorld (Figure 3 left) supports

the exploration of electrostatics, up to the concept of Gauss' Law, and PaulingWorld (Figure 3 right) 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.

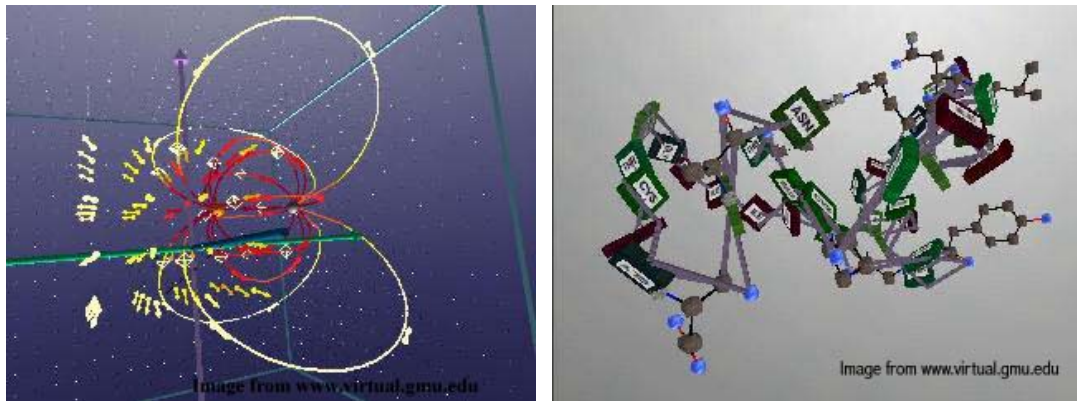


Figure 3: Left: MaxwellWorld – learning about electrostatic forces. Right: An amino acid in PaulingWorld – learning about molecular structures.

Zoology

In a totally different area of zoology the Virtual Gorilla Exhibit Project [4] was developed. This fascinating application teaches children about the interactions and habitat of gorillas at the Atlanta Zoo. A combination of desktop computer modeling and immersive VR is used. Therefore an exact model of the whole Gorilla habitat is used as you can see in the left picture of Figure 4. Students observe gorillas in the Zoo and model their behavior using a 3D software tool. Then they enter the immersive environment to explore their gorilla model. In Figure 4 (right) you can see a student using a head mounted display to explore the model. No formal assessment has been reported. Interviews with users elicited favorable responses in the sense of immersion, enjoyment, and successful communication of learning goals.



Figure 4: Left: 3D models of Gorillas and their habitat. Right: Students interact and view their own Gorilla models.

Biology

One of the most interesting projects in our opinion is NICE (Narrative-based, Immersive, Collaborative Environment) [133]. It is especially designed for children aged 6 to 10. Children can plant flowers, move clouds to water them or move the sun to make them grow faster (Figure 5). The main goal of the researchers was to explore virtual reality as a learning medium. Within this project a lot of interesting base work has been done such as good conceptual work and final evaluations. In section 8.2 we are using evaluation criteria as suggested by Roussos [134] which were developed partly within the NICE project.

52 second-grade children participated in the NICE evaluation. As usual when testing virtual environments some technical issues became apparent as well as usability issues. Shutter glasses for example that children were wearing in the CAVE environment, were too big for their heads. Most children had to hold the glasses all the time in front of their eyes until they got tired and just dropped them. On the scientific side the growth model of the plants was too simplified – the roots did not grow which made correct explanations difficult.



Figure 5: You can see Eddie interacting with the NICE garden in a CAVE environment.

Mathematics

In the area of mathematics education the most recent and most advanced project is CyberMath [153]. CyberMath is an avatar-based shared virtual environment aimed at improving mathematics education (Figure 6). It is suitable for exploring and teaching mathematics in situations where both the teacher and the students are co-present and physically separated. The first prototype was built on top of DIVE [42], a toolkit for building interactive shared distributed multi-user virtual environments. Due to problems with instability and usability of DIVE, the system was rewritten. In its current state, CyberMath is a desktop VR application with no support of immersive displays.

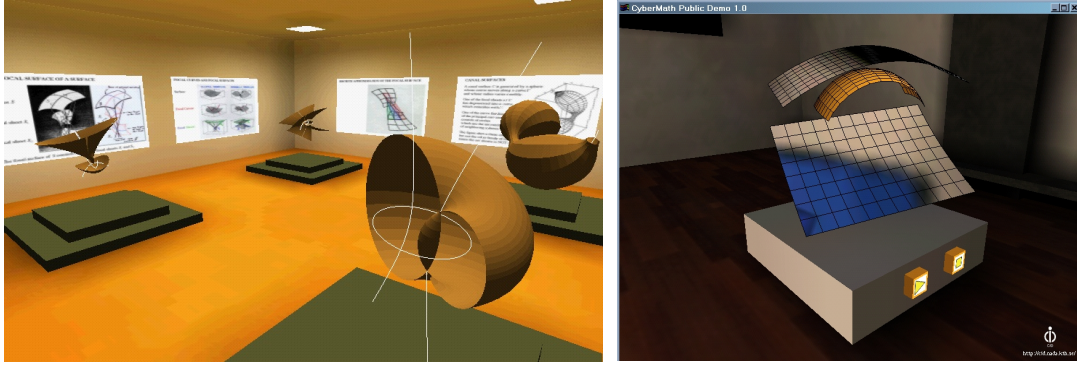


Figure 6: There are 4 different rooms on geometry and calculus with mathematical surfaces that can be explored. Some surfaces can also be interactively manipulated by changing properties or entering equations.

Many other related publications can be found on the web page of the Human Interface Technology Laboratory, University of Washington. Contributions have been made by the Support Initiative for Multimedia Applications (SIMA) project based at the Manchester Visualization Center. Various project reports and workshops such as “The Potential of VR for UK Higher Education” in 1995 are examples. A very good summary of educational applications is given by Mantovani [103] in the book section “VR Learning: Potential and Challenges for the Use of 3D Environments in Education and Training”.

In this context we can only present a small selection of work in the area of educational VR/AR applications but we chose projects which seemed most interesting to us and most advanced in their development process. It is interesting to note that nearly all of the educational projects we found during our literature research reached a certain point: Trial studies, evaluations in a smaller or bigger scale were conducted and then the projects ended. It is sad that no reports about continuous progress, no iterative development process and ongoing tests can be found in literature which go a step further. Therefore our work on Construct3D is already one of the longest developed educational applications so far. We are in the process of going one step further. It is already interesting to see how contents of the curriculum are adapted to the new learning environment (see chapter 7) and how ongoing technological improvements are integrated into our AR system (described in chapter 6). Since we will continue developing Construct3D the next years will show how Construct3D will change the learning and teaching process.

2.4 Geometry Education - Educational Geometry Software

In order to understand the author’s background and motivations and the environment in which the development of Construct3D started, we have to spend a few words on geometry education.

Geometry Education in Austria

Descriptive geometry courses are offered in many Austrian high schools for students from grade 7-12 and are not integrated into mathematics courses.

Geometry Education in Austria has a very long tradition. The Austrian tradition of a first-class geometry education has its roots in more than two centuries of world wide known geometers and top researchers in the field of descriptive geometry in middle Europe in general. Especially researchers from Germany, Hungary and Austria are driving forces in traditional descriptive geometry until today. Until the invention of computers the tradition and passion to descriptive geometry was tied to the fact that in order to do geometric drawings certain practical skills were required. Sometimes considered as art, wonderful constructions were drawn with pencil or black ink (tusche) on paper, carefully colored by using different techniques. Excellent examples of advanced and beautiful geometric constructions can be found in [18] or [167] and many other classical geometry books. Beautiful geometric constructions were not done to produce art, the meaning was rather to give the drawing a structure for better readability. Similar to source code which is hard to read without proper documentation, geometric constructions are hard to read if general stylistic conventions are not kept. Certain stylistic requirements were also passed from teachers to generations of high school students. Therefore some adults nowadays still connect descriptive geometry with troublesome hours of trying to beautify geometric drawings. In high school geometry education due to the large amount of time it required to do correct and good looking constructions, the actual geometric content often got obscured and played a minor role.

Since the mid 1980th and the emergence of computers in schools a silent revolution started and is still ongoing in geometry education in Austria. Most geometry teachers are teaching geometry in a very different way nowadays.

Modern Ways of Teaching Geometry

In Austrian schools the use of commercial 3D computer-aided design (CAD) software such as AutoCADTM, Autodesk Mechanical DesktopTM, Pro/EngineerTM, MicroStationTM 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 [146] or GAM [117] (developed by Austrian geometers specifically for students) which are frequently used.

It is important to note that while geometry education software shares many aspects with conventional 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 to 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 steep

learning curve. In contrast, geometry educators are interested in simple construction tools that expose the underlying process in a comprehensive way. In accordance to that our aim with Construct3D was not to create a professional 3D modeling package but a simple and intuitive to use 3D construction tool in an immersive virtual environment for educational purposes.

Stability and reliability are very important criteria for educational software in general. Correct results are absolutely important and if the software crashes students can easily lose their motivation. For learning, wrong results are worse than no results (many professional CAD packages still produce wrong results in specific cases). We consider robustness and reliability of educational software at least as important as for commercial software.

Geometric constructions with pencil on paper play a minor role in modern geometry courses and are mainly used while teaching basic theory to quickly visualize geometric principles. In some cases simple constructions on paper are just quicker to do than starting the CAD software of choice. The geometric content in the curriculum did not change much over the decades but there were big changes to the way it is taught. Because of that most geometry educators in Austria are currently in a phase of experimenting with new media and new tools – seeking for optimal pedagogical tools to teach various kinds of geometric content in an optimal way. Educators have to find for themselves which tools suit them, their students and their teaching methods best.

However a pedagogic tendency for systematic modern geometry education can already be seen. Studies show that students' spatial abilities benefit a lot if geometric sketching (hand drawings) are integrated in geometry education as it is the case in US American university courses [92, 93]. Computers are suited for solving more complex tasks that would take too much time to draw manually and offer new possibilities to gain insight into geometric problems (i.e. see dynamic geometry software below). A useful combination of geometric hand drawings (sketching) and educational CAD programs as well as dynamic geometry software is suggested as a foundation to build a future methodology of geometry education [66].

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

Dynamic 2D Geometry Software

Since a computer can record the way we construct geometric objects the software is able to quickly redo the construction after changing some parameters. This is the key concept of dynamic geometry: pick a point, move it and see immediately how the construction changes. This dragging capability is the fundamental improvement versus drawings on paper or static CAD models.

Comprehensive work on dynamic geometry was done by Kortenkamp in “Foundations of Dynamic Geometry” [79] who explains “Much more important (for educational purposes) is the fact that you can explore the dynamic behavior of a construction by moving it. You can see what parts of the construction change and which remain the same. You get by far more insight into this particular construction and geometry in general if you can experience what happens under movements. More sophisticated software will also give you another dimension of understanding by supporting loci, the traces of objects under movement of other objects, that are adjusted dynamically as well.”

The first software packages for dynamic geometry were Geometer’s Sketchpad [67], which appeared first in 1989, and Cabri Géomètre [83, 84], dating back to 1988. Since then a lot of work has been done to discuss aspects of using dynamic geometry software in education. Today, there are more than 40 packages for dynamic geometry. The most popular ones are Cinderella [127], Euklid [106], Geometer’s Sketchpad or Cabri Géomètre. All of them support two-dimensional geometry only.

The main problem of all of these implementations of dynamic geometry though are “ambiguities”. Kortenkamp [79] isolated the problem and solved it for the software Cinderella [127]. While a construction is done the user is responsible to resolve ambiguities that arise from an operation such as “intersection of a circle and line”. There are two possible solutions - a circle and a line have two intersection points if they intersect “properly”. Imagine you draw the tangent in one intersection point on a circle and continue to use this tangent for other constructions. Assume that by changing the radius of the original circle to make it smaller, the line does not intersect the smaller circle anymore. Therefore a tangent in a (non-existing) intersection point cannot be drawn. Now we increase the radius of the circle again and intersections occur again. Mathematically there are two possible intersection points. Which one is chosen by the software - which one will be used to draw the tangent? The intuitive answer is “the same as before” but the software must “remember” the solutions originally chosen to keep continuity. Even if the line is “turned around” by moving the end point to the starting point or vice versa the application must keep track. This and similar problems are resolved by Kortenkamp by using “complex tracing”. He describes “The main trick is that we carry out all calculations in complex space, and there we have all the powerful tools from complex analysis that we need to guarantee not only continuity, but even analytic behavior, as well as plenty of space to take detours around “bad” situations.” Cinderella is the only dynamic geometry software that handles ambiguities correctly.

Kortenkamp resolved the problem for two dimensional constructions only. For three-dimensional geometry he writes “In the three-dimensional setup we still can apply the theory of complex tracing, although it is a little bit more involved. The reason why we think that it is a challenging project that needs at least some years of

work does not lie in the area of continuity problems. [...] A much bigger problem is the exploding complexity of geometric operations in 3-space. If we just want to be able to intersect linear and quadratic objects, like we can do in 2D, we will have an enormous amount of special objects that occur as the intersection of quadrics. To create a versatile software for 3D will need much more manpower than the Cinderella project.”

In Construct3D we implemented dynamic geometry in a standard way as done by most other 2D dynamic geometry software authors. We cannot handle ambiguities correctly but internally use an “intelligent” way of remembering solutions to keep continuity to a certain extent. As mentioned above implementing complex tracing for three-dimensional geometry would require a few years of research for developing the mathematical foundations and finally implementation of all needed geometric algorithms. We would not be able to use a standard geometry kernel (as described in section 3.3). In Construct3D we keep continuity as long as all points stay in Euclidean space. As soon as one of multiple intersection points for instance becomes a point in infinity we loose track and loose continuity. Some of these problems can be handled using more sophisticated algorithms but in general this problem can only be solved by extending Kortenkamp’s mathematical theory to three-dimensional constructions. We consider this absolutely necessary in order to get correct dynamic 3D geometry applications. Maybe we can contribute to this very interesting area of research in the future.

Dynamic 3D – Parametric Computer Aided Design

There have been two revolutions in the history of CAD. The first revolution was a shift from paper-based drafting to computer-aided drafting. And the second revolution was a shift from computer-aided drafting to computer-aided design. The second revolution has just begun. The leading CAD technology is called parametric CAD.

In variational or parametric CAD we find a situation that is similar to what we have in dynamic geometry software. It should be possible to change parameters of a CAD construction in way that slight parameter changes do not cause big changes in the construction. This can be used for instance to have a single prototype construction which can be customized quickly or for data compression when we have to store a large number of similar objects. It is also used for easy and rapid construction of new models by starting with an approximate sketch that is made exact later. Not only the situation is similar to dynamic geometry the problems [60, 61] are similar too. Parts of these problems are discussed and have been solved in [79].

Many CAD applications already support parametric construction to a certain extent. Amongst those are 3DStudio Max™, Maya™, Autodesk Mechanical Desktop™, SolidWorks™, and many others. There are also research systems like the free parametric modeling tool VARKON (<http://www.tech.oru.se/cad/varkon/>).

However, no educational parametric (or dynamic) 3D tools which are especially tailored towards students in a similar way than the dynamic 2D programs mentioned above, are available yet. As far as we know none of the existing professional parametric CAD applications offer the dynamic flexibility that Construct3D offers. In addition their flexibility is limited by their desktop interfaces.

2.5 Pedagogic Theory

As Mantovani [103] 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. Constructivist theory provides a valid and reliable basis for a theory of learning in virtual environments [165]. As constructivism underlines, learning takes place when students can build conceptual models that are both consistent with what they already understand and with the new content. In order to ensure successful adaptation of old knowledge to new experience, flexible learning direction should be provided [103]. One possibility is to integrate known types of information and educational supports other than the 3D representation (such as audio and text annotations, images etc.). Another possibility is to carefully define specific tasks to the users/students through interaction with the teacher. We suggest the use of different learning modes in virtual environments from teacher-supported to autodidactic learning as described in section 7.3. Finally, VR environments can be tailored to individual learning and performance styles.

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 constructivist theory for the design of our second evaluation as described in section 8.2.

2.6 Psychological Aspects: Spatial Cognition

In this section we will give an introduction and literature review into the field of spatial cognition. In addition we speculate on the possible aspects of spatial abilities that might be affected by a training with Construct3D. To summarize the

findings presented here, section 8.4 concludes with a strategy how a psychological evaluation of Construct3D should be conducted.

Regarding spatial intelligence, a recent article by Durlach et al. [36] gives a very good overview of work that has already been done in the area of enhancing spatial skills within virtual environments but mainly identifies the indispensable need for comprehensive future research in this area. In a project proposal [19] for the Austrian science fund FWF we summarized literature on spatial cognition as described within this section.

There is a large psychological literature on spatial cognition, ranging from studies on the way people process simple spatial objects to studies about how people orient themselves in real environments. In the following, we will first discuss what subdimensions of spatial cognition can be distinguished, and which of them might be affected by a training using Construct3D. Then we will briefly discuss strategy issues in spatial cognition. Finally we will give an overview of previous training studies and discuss general principles of training spatial ability.

The Structure of Spatial Abilities

Several authors [26, 55, 56, 98, 99, 150] suggested how spatial ability can be structured into subdomains. Most of the proposed structures focused on relationships and similarities among spatial tests and were developed through analysis of test intercorrelations. Other aspects of spatial cognition, such as environmental orientation, have not been included in these analyses, but fit well into the structures proposed for spatial tests. In the following, we will briefly discuss approaches to structuring spatial ability.

Factorial structures of spatial ability. The most-cited structure was proposed by Lohman [98, 99]. He distinguished three factors. *Visualization* comprises all complex spatial tasks, e.g., surface development tasks, block design tasks and paper-folding tasks. *Spatial Relations* refers to tasks requiring speeded mental rotation of simple stimuli to determine whether they are identical or mirror images. *Spatial Orientation* comprises tasks that ask the participant what an object looks like from a different perspective. In addition, Lohman proposed two general dimensions along which spatial tasks can be classified: speediness vs. complexity and type of mental manipulation (mental movement vs. construction or synthesis).

Carroll [26] factor-analyzed a total of 94 data sets and distinguished five spatial factors. As in Lohman's model, he found a large factor *Visualization* that comprises all complex spatial tests that require a number of processing steps. The other four factors are more specific. *Spatial Relations* again refers to simple speeded mental rotation tasks. *Closure Speed* refers to the fast identification of incomplete figures. *Closure Flexibility* refers to the identification of given shapes embedded complex configurations. *Perceptual Speed* refers to the speed of comparing simple figures. The latter three factors refer to relatively specialized,

simple processes involved in, but not sufficient for, performance in complex spatial tasks.

Recently, Stumpf and Eliot [150] published a new analysis of the intercorrelations among 14 spatial tests. They based their selection of tasks on a content-based classification of spatial tests proposed by Eliot [37]. It is based on a conceptual, rather than empirical/correlational, analysis of the types of tasks used in the tests. Stumpf and Eliot applied facet theory and multidimensional similarity structure analysis as formerly used by Guttman et al. [55] (see also [56]); their findings replicated and extended those of other authors. Simple two-dimensional tests such as copying and maze tasks, visual memory tasks, and gestalt resolution tasks, formed single entities relatively distant from the other tasks. Complex three-dimensional tests that other authors would have grouped into the Visualization factor ended up quite close together in the center of the structure with two two-dimensional tasks located close to the center: figural rotation and paper formboard. Interestingly, there was a differentiation of three subsets of tasks within the center: First, there were tests of three-dimensional mental rotation – two types of block rotation tasks and the Perspectives test. Thus Lohman's Spatial Orientation factor was included in the rotation group here. Second, there was a sector containing Block and Intersection tasks. These are tasks requiring an analysis of internal features of spatial structures. The third sector comprised Paper Folding, Pattern Assembly, and Surface Development – tasks requiring construction through external manipulation of objects. In one of the two data sets that Stumpf and Eliot analyzed, speediness formed an additional dimension along which tests were ordered. Table 1 gives brief descriptions of all types of tasks that were in the center of the structural representation. These are also typical tasks of the Visualization factor as defined by other authors. Intercorrelations among these tests in Stumpf and Eliot's study ranged from 0.29 to 0.60.

<i>Task</i>	<i>Description</i>
Paper Formboard Tasks	Combine imaginatively the various parts of a figure to complete a whole figure.
Figural Rotation Tasks	Indicate which of several figures, when turned or rotated imaginatively, will be the same as a given figure.
Block Tasks	Estimate number of blocks, shape of blocks, intersection of block faces in a pile of blocks.
Intersection Tasks	Estimate intersections of two objects or one object with a plane
Block Rotation Tasks	Indicate which block, when turned or rotated imaginatively, is the same as a given block or object.

Paper Folding Tasks	Predict hole pattern in a piece of paper after being shown how the paper has been folded and where a hole has been pinched into it
Pattern Assembly	Reconstruct a given geometric figure using blocks with geometric patterns
Surface Development Tasks	Imagine how a three-dimensional object can be folded from a given two-dimensional figure.
Perspective Tasks	Predict what an object or configuration looks like from a perspective different from one's own

Table 1: Types of complex tasks grouped together in Stumpf and Eliot [150].

To conclude, the results of structural analyses of spatial ability are highly convergent. In addition to some simple, basic performance aspects, at least two factors are consistently reported: “Spatial Relations” (as labeled by Lohman [98]), which is speeded mental rotation, and “Visualization” which includes all complex, multi-step spatial tasks. Tasks involving three-dimensional mental rotation are somewhat intermediate and have been grouped into each of these two factors. Tasks requiring participants to imagine different perspectives either form a third factor or are grouped into Spatial Relations. There seems to be little differentiation within the Visualization factor. The most likely reason for this is that individual differences in strategy use may lead to larger interrelationships than specific task demands ([102, 145]; see next subsection). Only Stumpf and Eliot have found a substructure within Visualization, distinguishing three-dimensional mental rotation tasks, tasks requiring analysis of the internal structure of objects, and tasks requiring construction through external manipulation of objects.

Findings with spatial tests are highly dependent on which factor a test belongs to. For example, meta-analyses of gender differences in spatial ability tests [96, 160] have found relatively large gender differences in Spatial Relations, but none in Visualization.

Which factors might be affected by a training using Construct3D? Presumably training which helps participants to “handle” movements and transformations in three-dimensional space would largely affect performance in tasks using three-dimensional stimuli. It would require the imaginal manipulation and transformation of such stimuli. We do not expect such a training to improve simple, basic skills such as identifying incomplete figures. Rather the training should have effects on tasks comprised in the Visualization factor and on three-dimensional tasks in the Spatial Relations factor. In addition to effects on performance we believe that training with Construct3D will affect individuals’ strategies to solve tasks. In the following, we will very briefly review the literature on individual differences in spatial strategy use.

A strategy perspective. A different approach to thinking about spatial cognition is by analyzing the processes people actually use to solve spatial tasks. Basically, all spatial tasks can be solved in different ways, and the more complex a task is, the more different strategies can be used to solve it (overview in [100]). People differ in the strategies they use, and people shift their strategies within a task if necessary. A basic distinction is between holistic and analytic strategies [46, 47]. *Holistic strategies* involve representing and manipulating spatial information “in a spatial way”, that is, mentally using information about spatial relations between elements in the mental representation. A person using holistic strategies imagines, for example, how a stimulus figure is rotated or how a two-dimensional shape can be folded into a three-dimensional object. Holistic strategies often involve visualization but also include other ways of representing spatial relations, e.g., using one’s sense of direction while navigating. *Analytic strategies* reduce spatial information to an essentially non-spatial, serial format. For example, a route can be represented as a list of landmarks, and the spatial relations among the patterns on a cube can be represented as a list of relations among pairs of patterns [71]. Thus the complexity of spatial information is reduced and the focus is on parts of the object or the environment rather than on the object as a whole (which would include the spatial relations among the parts). Compared to holistic strategies, analytic strategies usually take more time, but less mental effort, as the information represented is less complex.

Note that analytic and holistic strategies should not be viewed as mutually exclusive categories. There are intermediate strategies such as mental rotation of parts of an object, and people often use more than one strategy to solve a task [71]. With increasing task difficulty both strategy variability and the frequency of analytic strategies increase. Most people can solve easy tasks (that is, tasks requiring simple spatial manipulations on simple stimulus objects) by holistic strategies, whereas with more complex spatial information or more complex manipulations, analytic strategies are required.

Gender differences in strategy use have frequently been found in studies of environmental orientation and navigation. Men more often acquire map-like knowledge about their environment, use Euclidean information and are better aware of relative directions than women. Women more often rely on landmarks and represent routes as lists of landmarks than men. These gender differences have been found in self-reported strategy use in new environments [88, 89], in use of landmarks and Euclidean information in learning routes from maps and giving directions [22, 33, 108], and in knowledge acquired from learning maps or from navigating through real or virtual environments [43, 44, 85, 90, 111, 141, 142]. With respect to spatial tests, gender differences in strategy use have not been studied much, but at least one study [46, 47] found that men more often used holistic strategies than women, and women more often used analytic and mixed strategies than men in two different spatial tests.

The strategy aspect is important here because it seems likely that a training with Construct3D improves visualizational skills. As Construct3D allows participants to “see” three-dimensional shapes and encourages to try out all kinds of manipulations on them, we expect that such a training leads participants to rely more on visualizational, that is, holistic strategies than they did before. Therefore in addition to test performance, we will assess the strategies individuals use before and after the training. For overviews of methods of strategy assessment and their respective pro’s and con’s, see [46, 47].

Training Spatial Ability

There has been a marked increase in training studies since 1996; however, these newer studies have largely been published in journals on education in geometry and not in psychological sources. The available literature clearly shows that spatial skills – as measured by standardized tests – can be substantially improved by means of training. Table 2 gives an overview of training studies published since 1995 and their results. Only studies that used pre-/posttest designs were included; thus excluding a few studies that only reported posttest differences. Thus, in all studies reported here, spatial ability tests were administered before and after the training. In most cases, the treatment was courses in graphics or geometry; in some cases, training programs specifically designed to foster spatial abilities were used. Evaluation of training effects was based either on comparisons among several experimental groups or on the comparison of an experimental training group to an untrained control group. As the table shows, most studies used one or more of the following spatial tests: the Mental Rotations Test, the Mental Cutting Test, the Differential Aptitude Test: Space Relations and the Purdue Spatial Visualization Test: Rotations. These tests will also be used in a future evaluation study (section 8.4).

Almost all studies listed in Table 2 found significant gains in spatial performance. Note however, that there may be a “file-drawer problem” [132] in the literature, as it is more difficult to publish studies that do not find significant effects. In control-group studies, the gains of the experimental groups were considerably higher than those of the control groups. Thus, increases in spatial performance are not due to practice effects. Comparisons of the effectiveness of different trainings (descriptive geometry, graphics, and special programs) indicate that especially trainings involving hand-drawing or sketching and physical modeling, that is, “hands-on practice”, appear to foster spatial performance.

MRT:	Mental Rotations Test (Vandenberg & Kuse, 1978 [159]). (redrawn version: Peters et al., 1995)	DG:	Descriptive Geometry
MCT:	Mental Cutting Test (CEEB, 1939) [1]	EG:	Experimental Group
DAT:SR:	Differential Aptitude Test: Space Relations (Bennett et al., 1973)	CG:	Control Group
PSVT:R:	Purdue Spatial Visualization Test: Rotations (Guay, 1977) [53]		

Authors, Year	Spatial Abilities Test(s)	Training, EG/CG	Main Results
Leopold et al., 1996 [94]	MRT	EG: DG course for freshman engineering students.	Significant gains in the EG, no sign. gains in the CG.
	PSVT:R	Duration: 1 or 2 semesters.	Gender differences in pre- and posttest favoring male students but larger gains for female students.
		CG: freshman mathematics students who had no DG.	
Gorska et al., 1998 [49]	MRT	EG: DG or Engineering Graphics courses for freshman engineering students at U.S., German, Japanese and	Significant gains at all universities.
	MCT	Polish universities. Duration: 10 – 14 weeks.	Significant gender differences favoring males on pre- and posttest, but higher gains for women.
	PSVT:R		
Saito et al., 1998 [135]	MCT	EG: Engineering graphics course for 1 st year mechanical engineering students:	Significant gains for the EG when compared to the gains in the CG after DG course as well as after mechanical drawing course.
		1 st semester: DG, 2 nd semester: mechanical drawing.	Higher gains for low-scoring subjects in the pretest.
		CG: no course.	Sign. correlations between pretest and semester-end test on DG.

Sorby & Baartmans, 1998 [143]	MRT (pre- and posttest)	EG: 3-D Spatial Visualization: A special pre-graphics course for freshman engineering students who failed the pre-PSVT:R ($\leq 60\%$). Duration: 3 month	EG: Significant gain scores on each test for each year (2 - 5 yrs.) Students in the EG outperformed those in the CG in their Subsequent graphics courses, had better retention rates and a lower average number of terms to graduation.
	MCT (pre- and posttest)		
	PSVT:R (pre- and posttest)		
	DAT:SR (pre- and posttest)		
Sorby & Gorska, 1998 [144]	MRT	5 different graphics courses for engineering students taught at Michigan Technological University. Duration: 3 month.	Large significant gains in courses where sketching or hand drawing was a primary focus of activity. Only marginally or not sign. gains in courses with larger computer components (like CAD).
	MCT		
	PSVT:R		
	DAT:SR		
Gittler & Glück, 1998 [45]	3DC (Three-Dimensional Cube Test (Gittler, 1990).	EG: CG course for pupils aged approx 16-18 years.	Significant gender differences favoring males at the pretest.
		Duration: 1 semester	Significant gains for the EG.
		CG: no course.	No gender differences at the posttest in the EG.
Field, 1998 [40]	MCT	EG: A 52-contact hour course in spatial visualization for undergraduate engineering students.	Significant gains in the EG compared to the CG.
		CG: no course.	
Sun & Suzuki, 1999 [151]	MCT	EG: Graphic science course for students with the Solid Simulator, a computer graphics software allowing generation, Boolean operations and dynamic projections of polyhedra as an additional instruction tool.	Significant gains in EG and CG. The gains in the EG were sign. higher compared to the CG.
		CG: Graphic science course without Solid Simulator.	

Larson et al., 1999 [86]	MRT, redrawn version	EG: A Virtual Reality Mental Rotation system.	Significant gender differences on the paper & pencil-pre-MRT favoring males.
	Virtual Reality MRT	Duration: approx. 1 hour.	No sign. gender differences on the paper&pencil-pre-MRT (but still men scored better).
		CG: no VR mental rotation training	VR-MRT: No gender differences (neither in solving duration nor in efficiency).
Leopold et al., 2001 [93]	MRT	EG: DG and/ or Engineering Graphics Courses for 1 st year engineering students at 3 Universities in the U.S.A., in Germany and Poland.	Significant correlations between pretests and final course examinations.
	MCT		EG: Sign. gain scores on each test at each university.
	DAT:SR	Duration 10 – 15 weeks.	Gains in the EG were higher (mostly sign.) compared to those in the CG.
		CG: Technical Drawing Course with very little DG or Computer Graphics only.	Beneficial course characteristics: smaller course section sizes, Hands-on sketching and drawing, DG contents.
Glück et al., 2002 [48]	IST: CCT (Intelligence Structure Test Battery: Cube Comparison Test (Amthauer, 1953)).	EG: Self training booklet “Rotating & Flipping”.	Significant gains in the EG.
		Duration: 3 – 6 days, 50 minutes each day.	
		CG: no training	

Table 2: Design and results of pre-/posttest spatial training studies published since 1996.

Moreover, in several studies gender differences at pretest were reduced or disappeared completely through the training. In all cases, women (who generally performed lower than men at pretest) had larger gain scores than men. Although this may sometimes be due to ceiling effects that prevent high-scoring participants from making large gains, the generality of the finding suggests that the trainings are especially conducive to the performance of persons scoring relatively low at pretest.

Some studies analyzed such aptitude-treatment interactions. These are individual differences in the degree to which participants profit from a particular training. Kirby and Boulter [77] found that participants low in pretest spatial performance profited most from traditional textbook instruction whereas participants high at pretest profited most from a holistic training. Thus baseline performance and the ability profile of participants may be an important predictor of training success. Kyllonen, Lohman, and Snow [82] tested the effectiveness of three different strategy trainings on performance in a paper-folding task. They found that holistic strategy training had best effects in participants with relatively low fluid intelligence. Analytic strategy training was most effective in participants with relatively low crystallized intelligence.

General principles of spatial training studies. Souvignier [145] gives a comprehensive review of training studies of spatial ability, based on a relatively broad definition of “training” that includes everyday experience with spatial activities and activities such as computer games. His general conclusion is that spatial ability can be improved through training. In addition Souvignier derives a number of principles for designing successful trainings of spatial ability from his review. First there exists a *power-generality tradeoff*. Trainings that have the strongest effects have these effects only on performance in one particular task being trained. Trainings that improve a broader range of spatial abilities have smaller effects. Souvignier argues that a successful spatial ability training should not be focused on one particular task, but rather on improving basic processes, which would allow for broad transfer to different, novel spatial tasks. Accordingly strategy flexibility, rather than one particular strategy, should be taught in order to train participants in selecting the best approach for each task. According to Souvignier training elements that should be conducive to learning transfer are (a) immediate feedback about the correctness of a solution, (b) opportunities for active, hands-on interaction with the stimulus material, and (c) explicit reflection on solution strategies.

With respect to hands-on experience as a predictor of training efficiency, we want to point again to the pedagogic theories of constructivism (section 2.5).

3 Technological Foundations

3.1 Studierstube

Studierstube [138] is a research software system for collaborative augmented reality (AR) applications, originally developed in an Austrian research project (FWF project P-11074-MAT). It provides the foundation and basic software design layers for our application Construct3D. Therefore it is necessary to describe some concepts and features of the system.

The term *Studierstube* denotes a place where insight can be gained. Researchers are often confronted with the investigation of highly complex and abstract theoretical issues, which need deep insight. *Studierstube* uses augmented reality to give the user an impression of such complex processes (Figure 7).

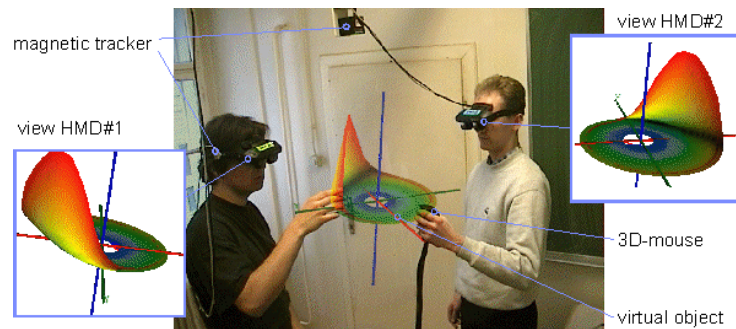


Figure 7: Two users are working with scientific visualization in an early *Studierstube* application.

The goal of the development of *Studierstube* is a software framework to support the technical requirements of augmented reality applications. It is a set of extension nodes to the Open Inventor [149] rendering library and an additional layer of objects that provide advanced runtime functions. It includes support for interaction based on 3D tracking events, rendering and output modes for all available virtual and augmented reality output devices, for developing distributed applications and user management functions to support multiple users in a single setup. The *Studierstube* platform has been tested in various fields, including flow

visualization, design, and collaborative games [138]. We summarize an excellent overview, given in [123].

Open Inventor

The Open Inventor (OIV) [149] rendering library is the basic software layer upon which *Studierstube* is built. It is a framework of C++ classes that implement a scene graph based rendering library using OpenGL. The principle architecture is that of an application framework with inversion of control that supports an event driven programming style whereby the application is typically composed as a set of callback functions that react to events issued by the framework.

The basic unit of OIV is a *node*. This is a C++ class type with additional functions to support runtime type system and serialization to and from an ASCII based text format. Nodes aggregate objects called *fields* that store a value of a certain type such as a string, a integer or floating point number, a 2D or 3D vector, or a rotation.

A dedicated node of type SoGroup can also associate a list of other nodes called *children* to form a hierarchical structure, the *scene graph*. Such a graph forms a directed acyclical graph and orders a set of nodes into a certain structure. The children of a group node are also ordered and can be accessed in a left-to-right fashion numbering the first child with index 0 up to the last child with index n.

The scene graph is traversed recursively by different mechanisms called *actions* to compute different data. The actions call different functions on the nodes to trigger certain behavior. For example, the GLRenderAction sends appropriate commands to the OpenGL library to draw the image represented by the scene graph. The SearchAction traverses a graph to find nodes of specified type or name. The WriteAction serializes a scene graph into the Open Inventor file format.

Each node type can define the behavior for each action separately by registering a function to be called by the action when it traverses a node of this type. Thus a double dispatch mechanism is created to provide a flexible implementation and extension mechanism. For a more detailed discussion of these concepts, see the Inventor Toolmaker [163].

In addition to the scene graph traversal another flow scheduling mechanism is provided by Open Inventor. Fields of nodes can be connected to receive updates from other fields forming a data flow network for small scale stream processing and event handling. A special class of objects called *engines* can be embedded in the data flow graph to process the change events and compute new updates to other fields. These objects are not part of the scene graph but only of the overlaid field network graph.

The Open Inventor API provides methods to construct and operate on the scene and field network graph. Additional sensors can observe changes to fields, nodes and the scene graph and report these to callback functions. Time based sensors can

trigger callback functions after a certain time span, in regular intervals or when the library has CPU time to spare.

Finally, the library provides a text based and a binary file format to serialize a scene graph and field network structure to persistent storage (see the Inventor Mentor [162]). Complex scene graphs can be constructed directly as text files and read in by a client application of the library. Both manual and automatic authoring of such structures becomes possible in an efficient and transparent way.

Studierstube uses all of the above concepts to extend the base library in several ways and provides AR-centered functionality in the form of new nodes, actions and engines.

3D Event System

Open Inventor supports only user interface events generated by a standard desktop interface consisting of mouse movements and key and mouse button presses. Such events are propagated into the scene graph via the `HandleEventAction` and are consumed by different types of nodes. A default set of manipulator nodes exist which implement a set of standard 3D interactions such as translating, scaling and rotating an object with the help of 3D widgets.

Studierstube extends the library to support more generic user input devices, generally 6DOF trackers. It implements a list of individual event channels that supply the scene graph with streams of 6DOF tracking events. These events consist of a channel id called station number, 3D position, a rotation, button states for up to 8 buttons, a time stamp and an event type discerning between movements or changes to the button state. They are propagated with a dedicated `Handle3DEventAction`.

Individual stations are associated with different input devices such as a user's head, a pen or a tablet that are manipulated by the user or with other tracked objects in the environment that are required for an application.

Widget System

A special group of nodes that interact with the 3D event system implement a set of standard widgets. Widgets are graphical objects that react on incoming 3D events and change their state based on a sequence of 3D events. Thus, they implement filters to compute a higher abstracted event from the stream of raw 3D events. Their state is represented by different graphical representations and changes to fields which are picked up by application in turn.

2D widgets represent a typical example of such objects. Studierstube implements a standard set consisting of toggle, push and radio buttons, lists, and linear sliders. They are represented by 3D geometry that can be manipulated by the user. For example, buttons have a box like default geometry that has different heights for the released and pressed state. Thus, it simulates the look and feel of a real button.

A traditional 2D graphical user interface is usually presented on a tracked tablet called the Personal Interaction Panel (PIP) [152] (Figure 8; also see section 6.4). The physical representation provides a natural way to interact with the virtual widgets and gives haptic feedback when an interaction device intersects the virtual widget and collides with the real tablet. The PIP supports to switch between different sets of widget groups similar to a tabbed dialog. Each pane can represent a user interface associated with a different application.

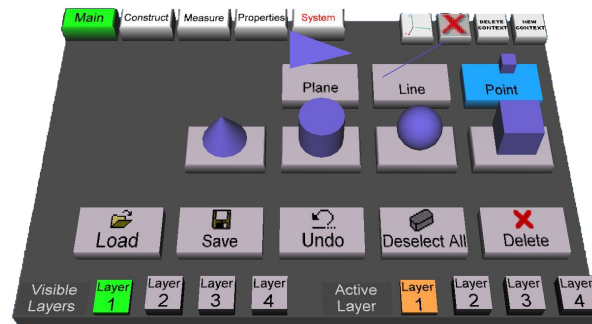


Figure 8: Construct3D menu system on the PIP.

3D widgets allow simpler but less restricted interaction. They encapsulate geometry that can be dragged and rotated by the user with direct manipulation. All objects in Construct3D are of that type in order to enable dragging and dynamic modification. The Raypicker widget implements a ray casting interaction that computes the intersection of a ray with given geometry. The ray is controlled by one or more 3D event channels. A more complex 3D widget is the WindowKit node which implements a box like container for geometry similar to a window in a 2D graphical user interface. The window's borders act as manipulators to move, rotate and resize it.

Dynamic Application Loading

Augmented reality applications are implemented as a sub scene graph in a Studierstube process. They are defined by implementing a new application node class that provides the application specific functionality. The application node can use any sub scene graph to create the required graphics, user interface elements and interaction methods. In addition applications can define their own specialized node types to store the required data structures reusing Open Inventor's rich set of field data types. At the same time, such a design enables the use of all Open Inventor operations on the application's data.

Because Open Inventor supports serialization of any scene graph to and from a file, applications can be loaded and saved at runtime. As an application will store all required data structures in fields and/or nodes of a sub scene graph, the application's scene graph already represents the application's state. Studierstube supports concurrent execution of several applications and provides an API to start, stop and save applications as well as a user interface for manual control of applications.

Single Host - Multiple Users, Multiple Displays

Collaborative work scenarios are supported by Studierstube through providing the necessary functionality to drive the hardware devices required for several simultaneous users and through an API to model resources for these users. Studierstube supports several independent output windows to drive multi-headed systems that offer a number of video outputs. Thus, several display devices can be connected simultaneously and provide personalized views to their users. Each output window can be configured independently in size, position and rendering method for stereo displays. The virtual cameras associated with each output window are controlled by independent input devices.

Moreover, the number of input devices is not limited allowing the use of as many devices and trackers as required to build a multi-user setup (Figure 9). A typical dual user setup for collaborative work will consist of two HMDs, two interaction devices and two PIPs, resulting in a total of 6 tracked devices and two output windows. A set of resources consisting of an output window and event channels for head-tracking and input devices are modeled as a user identified with a unique id within the software framework. Applications are notified on startup of the number of users and their configuration. 3D events are associated with a user, if they are representing one of the user's input devices. Therefore, applications can distinguish between users and react differently as required. The output display of each user can also be configured to use a private sub scene graph which is only rendered for this user e.g. a student sees his construction whereas the teacher can see the construction plus a possible solution to the problem. This mechanism enables personalized and private views.

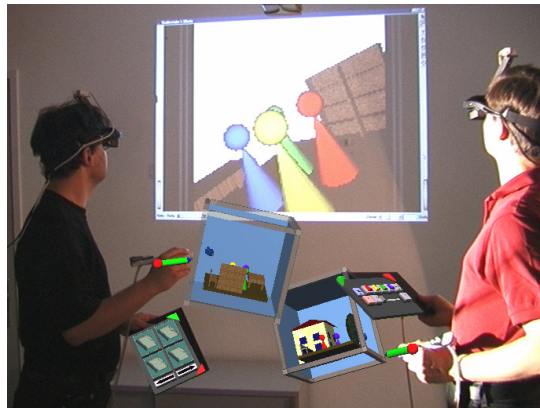


Figure 9: Multiple applications and multiple screens can be managed by the *Studierstube* system [137].

Multiple users can naturally collaborate in an augmented shared space. Instantaneous collaboration can be established as for instance users wearing the mobile system meet (our mobile system is described in section 5.1). Reitmayr and Schmalstieg [124] demonstrated collaboration between a mobile and a stationary user that is established as the mobile user walks into the Studierstube lab. The underlying distributed system transparently synchronizes graphical and application

data and even streams running 3D applications on the fly to newly joining members of a workgroup.

Distributed Inventor

Like *Studierstube*, most distributed virtual environment systems use a scene graph for representing the graphical objects in the application, but many systems separate application state from the graphical objects. To avoid this "dual database" problem [101], *Studierstube* introduces a distributed shared scene graph using the semantics of distributed shared memory. Distribution is performed implicitly through a mechanism that keeps multiple replicas of a scene graph synchronized without exposing this process to the application programmer or user. Our OIV extension Distributed Open Inventor (DIV) [57] uses OIV's notification mechanism to distribute changes.

The communication between different replicas uses a reliable multicast protocol. All hosts of replicas can send updates but the implementation only provides causally-ordered update semantics. Therefore updates sent by different hosts can be executed in different order on individual hosts.

A scene graph to be replicated is denoted by a special group node *SoDIVGroup*. It is configured with the necessary networking parameters and automatically establishes connection with other peers. Any changes to the sub scene graph of such a group node are communicated automatically. It also can be configured to retrieve a current copy of the sub scene graph from another peer upon joining a session. DIV supports a master/slave property for each peer. Only peers with the master property set actually generate updates. Peers with the master property set to slave only listen for updates and apply them to their local replica. To avoid inconsistencies in the updates of the replicas typically only one peer is a master within *Studierstube* and therefore controls the replicated scene graph alone.

As the scene graph can be distributed using DIV, so can be the applications embedded in it. For each application a dedicated *SoDIVGroup* is created as a parent to the application's scene graph. Then, a newly created application instance will be added to it and will therefore be distributed to all replicas of a scene graph. The programming model of making application instances nodes in the scene graph also implies that all application specific data are part of the scene graph, and thus are implicitly distributed by DIV.

At any point in time only one replica is a master and therefore controls the application. The other replicas only use the application's scene graph to render the personalized view for their outputs. By moving the master property between replicas simple forms of load distribution can be implemented.

We use the distribution of *Studierstube* in many of our scenarios (chapter 5).

3.2 OpenTracker

Tracking is an indispensable part of any VR and AR application. While the need for quality of tracking, in particular for high performance and fidelity, have led to a large body of past and current research, little attention is typically paid to software engineering aspects of tracking software. What is needed is a system that allows mixing and matching of different features, as well as simple creation and maintenance of possibly complex tracker configurations.

OpenTracker [125] is an open software architecture that provides a framework for the different tasks involved in tracking input devices and processing multi-modal input data in virtual environments and augmented reality applications. The OpenTracker framework eases the development and maintenance of hardware setups in a more flexible manner than what is typically offered by virtual reality 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. Transparent network access allows easy development of decoupled simulation models. Filters and transformations can be applied to tracking data. OpenTracker is available under an open source software license.

XML based configuration files are used to describe tracking configurations that usually consist of multiple input devices. Studierstube uses OpenTracker which enables us to build tracking configurations of high complexity such as the hybrid and mobile setups in sections 5.3 and 5.4.

3.3 Geometric Modeling Kernel

Early in the development of geometric modeling or CAD/CAM applications simple 3D object generation is implemented. At that point more advanced geometric algorithms are needed that operate on objects. Algorithms such as Boolean operations (union, difference, intersection) between geometric objects seem to be easy to implement at first sight. In reality it is extremely difficult to implement Boolean operations in a robust way. There are a large number of special cases, i.e. edges can lie on vertices or on faces of other objects, edges/vertices/faces of different objects can coincide, grazing cases might occur with models exhibiting inaccuracies and so on. Many open source and even professional geometric kernels have difficulties implementing Boolean operations in a reliable and stable way. One only has to look into release notes and bug lists of existing kernels. Boolean operations are an important basis and only mark a beginning when implementing advanced modeling algorithms.

In recent years the use of geometric modeling kernels became more wide spread and a good open source solution called Open Cascade [3] emerged. Nowadays developers - researchers as well as commercial entities - have the choice to either implement all necessary algorithms themselves to keep full control over data structures and algorithms, or to use an existing kernel. A big advantage of integrating existing kernels is that advanced modeling features can be offered in early product versions within reasonable development time. It saves a lot of time and development costs. A disadvantage is that commercial geometric modeling kernels such as ACIS[®] [32] by Spatial Technologies or Parasolid[®] [158] by Unigraphics Solutions (UGS) do not offer access to the source code, at least not for educational institutions. It is sometimes impossible to predict the exact outcome of various algorithms since the algorithms used in closed source kernels are unknown or insufficiently documented. In addition it is difficult to get bugs fixed by the developers of these kernels. To present-day ACIS[®] and Parasolid[®] are two of the best and most expensive commercial geometry kernels used widely in commercial CAD applications. Parasolid is the geometry engine inside SolidWorks, SolidEdge, DesignWave, Unigraphics and many other products. ACIS is being used in AutoCAD, Autodesk Inventor, Corel DESIGNER Professional SG, TurboCAD, MegaCAD, IronCAD, Cadkey, Universe Modeller and many more.

ACIS, Parasolid and OpenCascade use boundary representations of models. A review of some of the most frequently used kernels (CAD APIs) can be found online at http://www.pointwise.com/ggmns/review/review_table.html.

After initially developing basic intersection algorithms ourselves by using Open Inventor engines we chose to use an existing kernel for Construct3D in order to save development time. After initial reviews we decided to use OpenCascade.

Open CASCADE

Open CASCADE is an open source alternative to proprietary 3D modeling kernels. The Open CASCADE components are a set of C++ libraries that allow to develop custom technical and scientific applications. The majority of code is distributed in open source. Additional components have been developed for more specific needs. These are not in open source and can be purchased from the Open CASCADE company.

The decision in favor of this product was based on the fact that it was the only open source toolkit that offered sufficient functionality. In its current version 5.1 the 2D and 3D geometric modeling toolkit allows to

- create primitives such as prism, cylinder, cone and torus
- perform Boolean operations (addition, subtraction and intersection)
- calculate the intersection of two curves, surfaces, or a curve and a surface
- tweak constructions using fillets, chamfers and drafts

- model constructions using offsets, shelling, hollowing and sweeps
- compute properties such as surface, volume, center of gravity, curvature
- compute geometry using projection, interpolation, approximation

In addition there is a library providing visualization services (managing windows and view manipulation) and a data exchange module.

However in version 4.0 it was not as robust as we expected it to be. While integrating Boolean operations many problems were encountered. Open CASCADE algorithms caused unpredictable crashes and fixing the problems would have required to study and fix the toolkit's source code. Since working solutions were needed and Boolean Operations were just the beginning of more advanced modeling algorithms that we had in mind, we decided for another alternative. The university partner program of Spatial Technologies offers an ACIS license for educational purposes for free.

ACIS

The 3D ACIS[®] Modeler (ACIS) is Spatial's 3D modeling development technology used by developers worldwide, in industries such as CAD/CAM/CAE, AEC, animation, and shipbuilding. ACIS provides some of the world's most recognized software developers and manufacturers with the underlying 3D modeling functionality necessary for creating innovative, high-performance applications.

ACIS 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. The ACIS core functionality for 3D modeling provides features such as

- Intersect/subtract/unite any combination of curves, surfaces, and solids.
- Extrude/revolve/sweep sets of 2D curves into complex surfaces or solids.
- Interactively bend, twist, stretch, and warp combinations of curves, surfaces, and solids.
- Fillet and chamfer between faces and along edges in surface and solid models.
- Fit surfaces to a closed network of curves.
- Generate patterns of repetitive shapes.
- Hollow solids and thicken surfaces.
- Loft surfaces to fit a set of profile curves.
- Taper/offset/move surfaces in a model.

ACIS features 3D model management. It allows to track geometry and topology changes, calculate mass and volume, model sub-regions of a solid using cellular topology and unlimited undo/redo with independent history streams. Regarding model visualization algorithms are provided to tessellate surface geometry into polygonal mesh representation. Visualizations can be done using Tech Soft America's HOOPS application framework, available from Spatial and integrated with ACIS.

A separate component (PHL V5) is offered for hidden line removal to generate precise 2D projections. Components for surface healing and import and export of a wide range of 3D file formats are also available. Sophisticated memory management helps to find memory leaks and mechanisms are provided for tracking errors inside ACIS methods.

For educational purposes we only use a limited set of functionality provided by ACIS. The toolkit's API functions are very flexible but sometimes documentation is poor. As a consequence the exact results of functions are difficult to predict and can only be determined by lengthy trial and error tests. As a university partner institution we do not get access to source code and cannot fix arising problems ourselves. During extensive tests minor bugs were found but in general ACIS is robust and reliable.

4 Construct3D

Construct3D [72-75] is a three-dimensional dynamic geometry construction tool that can be used in high school and university education. It is based on the Studierstube AR system and uses augmented reality to provide a natural setting for face-to-face collaboration of teachers and students. The main advantage of using AR for constructing geometric objects is that students actually see three dimensional objects which they until now had to calculate and construct with traditional (mostly pen and paper) methods.

It features support for dynamic 3D geometry. A fundamental property of dynamic geometry software is that the dynamic behavior of a construction can be explored by moving it. It can be seen what parts of a construction change and which remain the same. By far more insight into a particular construction and geometry in general can be gained by experiencing what happens under movements.

In this chapter we begin with a description of basic concepts of Construct3D. All features are listed to give a comprehensive overview of the applications capabilities. Section 4.2 contains details of our rather traditional implementation of dynamic 3D geometry and the underlying data structures are described. Real applications of Construct3D's features are demonstrated in chapter 7: Content development.

4.1 Software Design

Overview

The current version of 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, 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.

At its start Construct3D initializes a 3D window which has maximum size to cover the “whole” virtual space. The user interface is initialized as well. The menu system is mapped to a hand-held tracked panel, the personal interaction panel (PIP) (section 6.4). The PIP allows the straightforward integration of conventional 2D interface elements like buttons, sliders, dials etc. as well as novel 3D interaction widgets. 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.

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. For selecting objects, point mode must be turned off (by clicking on a button on the PIP). If point mode is turned off the nearest objects in the scene is automatically highlighted. This indicates which object will be selected if the user presses the pen's button again. Various tests of this selection method showed that it is very convenient, quick and intuitive to use. Users constantly see which object will be select and can move their pen closer to the desired object if necessary.

A description of our user interface is given in 6.4. Details of object manipulation and how objects visually react to user input are given in 6.5.

We support generation of and operation on these basic object types: Points (either freely positioned in space or fixed on curves and surfaces), lines, planes, circles, ellipses, cuboids, spheres, cylinders, cones, B-Spline curves with an unlimited number of control points and variable degree, interpolated B-Spline curves, NURBS surfaces up to 8x8 control points and variable degree, interpolated NURBS surfaces and surfaces of revolution (rotational sweep surfaces).

The following geometric operations are implemented:

- Boolean operations (union, difference, intersection) on 3D objects (except NURBS)
- Intersections between all types of 2D and 3D objects resulting in intersection points and curves
- Planar slicing of objects
- Rotational sweep around an axis
- Surface normals in surface points
- Tangential planes in points of surfaces
- Tangents to all types of curves and in curve points

- Common tangent to two circles
- Plane normal to a line through any point; Line normal to a plane through any point
- Plane of symmetry;
- Angle bisector
- Mid point

All these operations consistently support dynamic modifications of their input elements and reevaluate the resulting elements accordingly.

In addition to these geometric features distances between two or more points can be measured. Necessary system operations such as selection and deselection of primitives, save, load, delete, undo and redo are mapped to a hand-held tracked panel, the personal interaction panel (PIP) (section 6.4). 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 allows the props to be used as multi-functional tools.

Selection – Action

“Selection-Action” is one of the most fundamental principles when working with Construct3D. In order to conduct any operation that requires input elements (e.g. generating a line out of two points), objects must be *selected* first. Once the input elements are selected, an action can be triggered by pressing the appropriate menu button on the PIP.

Table 3 summarizes for some selected operations which input elements are needed to generate new objects.

Object type + sub types	Required input elements
Line	2 points
Surface normal	1 point + 1 3D object (the point must lie on the object) – a normal in a point of a NURBS surface can be seen in the image below
Tangent	1 point + 1 circle (the point can lie on the circle or outside)
Angle bisector	3 points (the points determine the angle)
Plane	3 points 1 point + 1 line 2 lines that span a plane

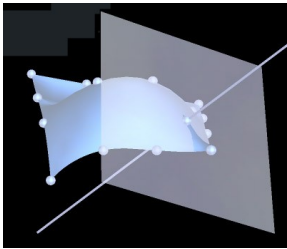
Plane normal to a line	1 point + 1 line	
Plane of symmetry	2 points	
Tangential plane	1 point + 1 3D object (the point must lie on the object) – see image of a NURBS surface.	
Circle	3 points (lying on the circle) 2 points (midpoint + point on the circle, both must lie in a plane) 2 points and a line (the line is normal to the plane that contains the circle)	
Ellipse	3 points (midpoint, end point of the main axis, point on the ellipse)	
B-Spline curve	arbitrary number of points through which the curve passes in the order the points have been selected	
Sphere	2 points (mid point + point on the sphere) 3 points (mid point + 2 points whose distance determines the radius)	
Cylinder	3 points (start and end point of the axis, point on the cylinder) 1 line + 1 point (axis + point on the cylinder) 2 lines (axis + tangent to the cylinder)	
Boolean Operation	2 “solid” objects (any 3D object but no NURBS surface and no sweep surface)	
Intersection	2 arbitrary objects (no points)	

Table 3: In order to create the objects on the left, elements on the right must be selected beforehand.

We omit the very detailed description of which input elements are needed for generating all objects in Construct3D. However, some features need a more detailed explanation.

Boolean Operations

All 3D objects except NURBS surfaces can either be interpreted as volumetric solids or as surface boundary representations. If we interpret them as solid objects we can do Boolean operations on them (Figure 10 right). As noted in Table 3

Boolean union, difference and intersection operations can be applied to any two solid objects except NURBS and sweep surfaces which internally exist as surfaces only.

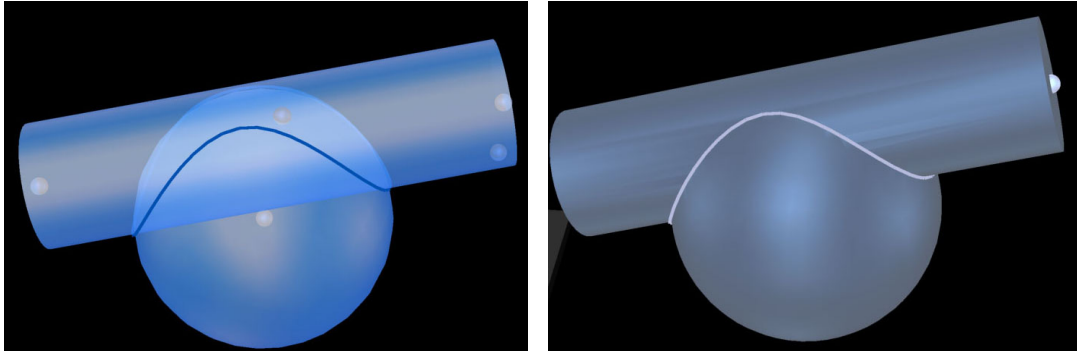


Figure 10: Left: Intersection curve of a cylinder with a sphere. Right: Boolean union of the same two objects. The resulting Boolean object is drawn opaque to give a solid impression. In addition the intersection curve is displayed.

Intersection Curves

For the construction of intersections, surfaces' boundary representations are used (Figure 10 left). Any two objects can be intersected in Construct3D, except points of course. Depending on the involved objects, results can be intersection points or intersection curves.

Layers

We are using a 3D-layer system very similar to the one used in current image editing programs. 3D-layers offer the possibility to arrange parts of a construction into overlapping sub-spaces that can be controlled independently.

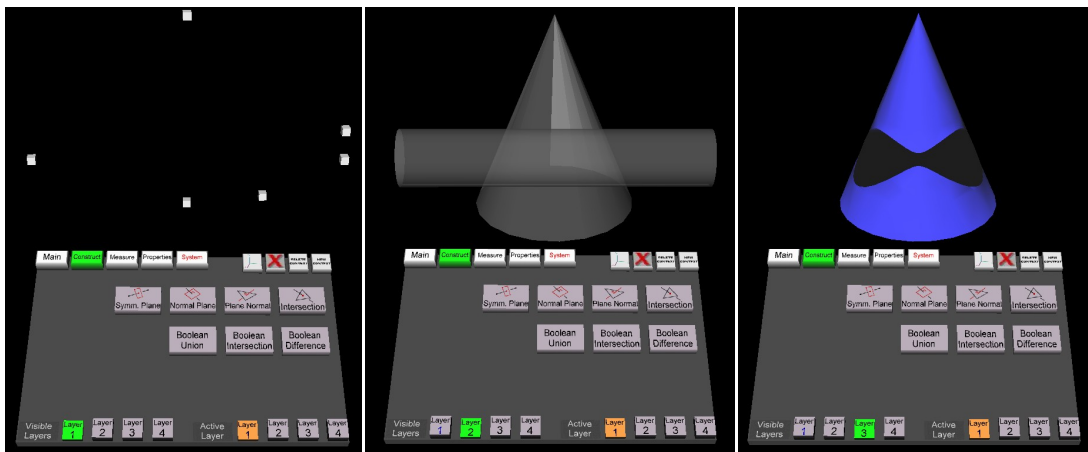


Figure 11: Left: Points used for the construction of cylinder and cone are in layer 1. Middle: Layer 2 contains the base objects for the Boolean difference operation. Right: We moved the resulting Boolean difference of cone minus cylinder into layer 3. In the lower left corner of each figure the green layer buttons on the PIP indicate currently visible layers.

Figure 11 demonstrates how different construction elements of a Boolean operation can be structured into various layers. Multiple layers can be visible at a time. In the given example (Figure 11) we can switch on layers 1 to 3 simultaneously but only one layer can be active at a time. An active layer is the one where new constructed elements are added. A flexible mechanism allows assigning existing objects to other layers than they are currently contained in. A selected object is moved to the appropriate layer using the PIP.

The layer 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. We mainly use layers for structuring constructions in order not to overload educational examples with too many geometric objects. The possibility of structuring constructions is fully supported by our color design of layers as described in section 6.5.

Top View, Front View, Side Views

In traditional geometry education (see 2.4) nearly all constructions are done in normal views such as top view, front view, left side view and right side view. In accordance with constructivist theory in order to ensure successful adaptation of old knowledge to new experience [103] it is of advantage to integrate known types of information. It was also suggested by teachers to integrate top view, front view and side views into Construct3D.

By using hardware accelerated rendering of the whole scene directly into texture memory we can provide all 4 views in real time. This feature can even be used as part of the construction as demonstrated in the examples about “spatial interpretation of planar constructions” at the end of this section. In Figure 12 we use top view, front view and right side view to show the special features of a curve called the VIVIANI window. In a front view the curve is part of a parabola, in side views (on the left side of the image) it is a circle.

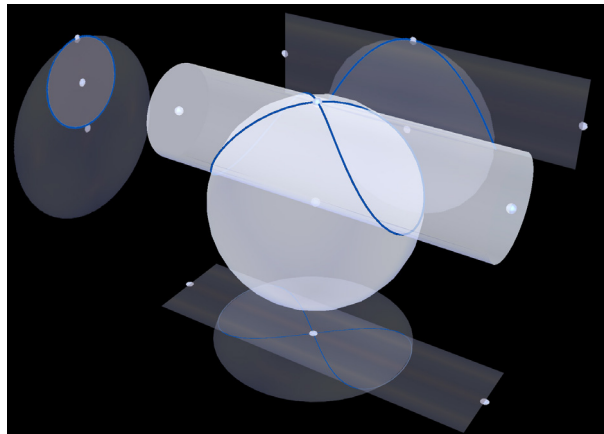


Figure 12: Normal views can be used to show special properties of curves such as the VIVIANI window in this case.

The VIVIANI window is a result of the intersection of a cylinder with a sphere. The cylinder has half the radius of the sphere and contains its center.

VRML Export and Import

The Open Inventor implementation Coin that we currently use supports VRML import and export. This enables us to export all geometric constructions from Construct3D as VRML files. This is very motivating for students since they can view their own constructions with a VRML web browser plugin at home or show them to friends on the web.

We utilize the VRML import feature as an interface to computational software such as Maple, Mathematica or Matlab. They all feature VRML file export of three dimensional mathematical plots such as surfaces, graphs or other output. We are now able to import these VRML files into our construction environment. In Figure 13 we demonstrate this feature. We computed the intersection curve of a sphere with a cylinder in Maple. The result is again a VIVIANI window (Figure 13 left). Whenever we load new VRML content it is put into a separate 3D window which can freely be placed in 3D space.

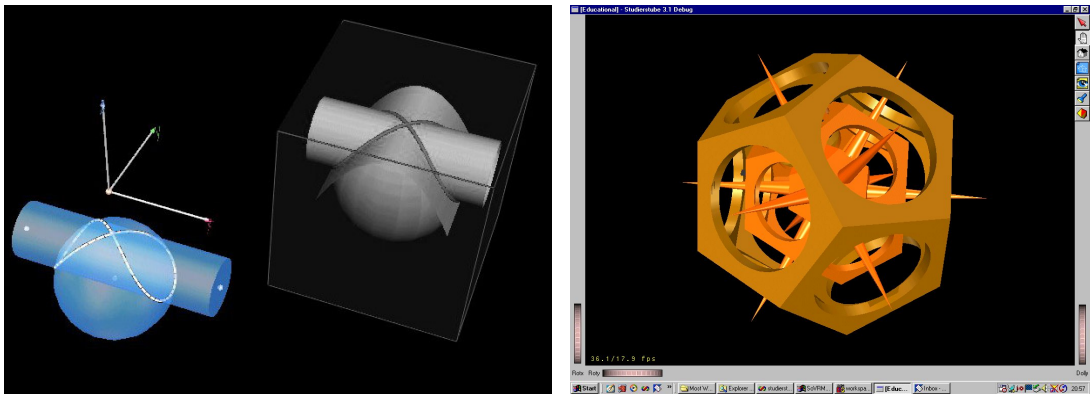


Figure 13: Left: VIVIANI's window is loaded as a VRML file into Construct3D (top right). The same intersection curve constructed with Construct3D (bottom left). Right: Any VRML content can be loaded into Construct3D in a separate 3D window.

Undo and Redo in Multi-User Applications

The ability to undo operations is a standard feature in most single-user interactive applications. However, many collaborative applications that allow several users to work simultaneously (i.e., on a shared document) lack undo capabilities. Those which provide undo generally provide only a global undo, in which the last change made by anyone to a document is undone, rather than allowing users to individually reverse their own changes. A very comprehensive work on the topic of undoing actions in collaborative systems was written by Prakash and Knister [118]. Work on undo in a collaborative graphics editing system was done by Chen and Sun [29]. They do not mention dependencies that occur between objects.

Construct3D features a global undo which means that the last change made by any user is undone. A local or selective undo – allowing users to individually reverse their own changes or reversing only specific changes is not trivial at all in a collaborative dynamic geometry application. In such an application it is very common that user B for instance continues working with objects generated by user A. If user A performs a local undo of his own last operation, he removes an object used by user B for the next construction steps. In such a case - if one base object is removed - the application has to remove all dependent objects of user B as well. This is not a desired application behavior for user B.

Such conflicts are only addressed in the work of Prakash [118] who says “In any undo scheme, it is important that undo behaves according to users’ expectations.” Since such conflicts cannot be automatically resolved by an application, we decided to use the rather simple global undo mechanism. In future versions of Construct3D it is still possible to implement local and selective undo. Assuming that there are no conflicts with constructions of other users we may allow users to undo their own last operation or to select an earlier operation for undo. Conflicts can be avoided by dependence checking to make sure that operations of users do not overlap through the use of locks. To which extent the use of locks is reasonable in an educational setting is a different question.

Collaboration Modes

Currently multiple users share a common construction space and can work collaboratively on constructions. We suggest to support different modes of collaboration in future versions to support more teaching scenarios. Based on our implementation of layers these basic modes would allow arbitrary selection of visibility per user and per layer:

- collaborative mode, i. e., everything is visible to everybody. This mode is currently implemented and used by default.
- independent mode, i. e., every student can only see the elements constructed by himself.
- 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. Students are enabled to work on the model themselves by request. The whole construction is visible to all users (collaborative mode). If later the teacher wants the students to practice on the same model, he switches to “independent” mode while the application is still running. Now each student can see the immutable specification and the elements that he constructed himself only 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 (teacher mode). 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.

Distributed Construct3D

For most of our scenarios (described in chapter 5) we need to distribute Construct3D to multiple clients. Therefore Construct3D is also running as a distributed application in a stable and robust way. Figure 15 shows a desktop setup where the construction of a NURBS surface - intersected by a B-Spline curve - is distributed to a client on the same machine (shown in the second window in the background).

4.2 Implementation

This section should not be taken as a final recipe to implement dynamic 3D geometry software but rather as a description of one possible approach. Construct3D grew over the years: Its software design changed dramatically especially after our first evaluation when we made the step from a static modeler to a dynamic geometry application as described in section 6.3. Functionality grew as well: new types of curves and surfaces and many other features were added and improved. Due to constant changes it may well be that a different software design would better fit the current tasks and functionality of Construct3D. We are mainly concerned with the constantly increasing complexity of our implementation. Most of the complexity arises because of the need to check for many special cases while users operate on geometric objects. This improves robustness and usability though. To quote a colleague and software design expert: “Complex behavior cannot always be implemented in a simple way”.

Class Hierarchy

We implemented the dynamic core of Construct3D in a rather traditional way similar to the straight forward implementation of most other dynamic geometry software (see [79] p.121ff or [106]). Usually a common root class specifying a general geometric object is subclassed into classes for different geometric objects, which are again subclassed in order to create the object definitions. A detailed overview of our class hierarchy is given in Figure 14.

We start our description on the left side of the diagram. Our main class C3D is the user interface class of Construct3D. Its superclass SoContextKit is a node kit class of the Studierstube framework (see the Open Inventor description in section 3.1). SoContextKit represents the actual application node containing the applications functionality. It acts as a base class to be used by the programmer to implement a custom application by deriving a new subclass from SoContextKit. Additionally the SoContextKit is supplied with window geometry associated with the application and PIP sheet geometry.

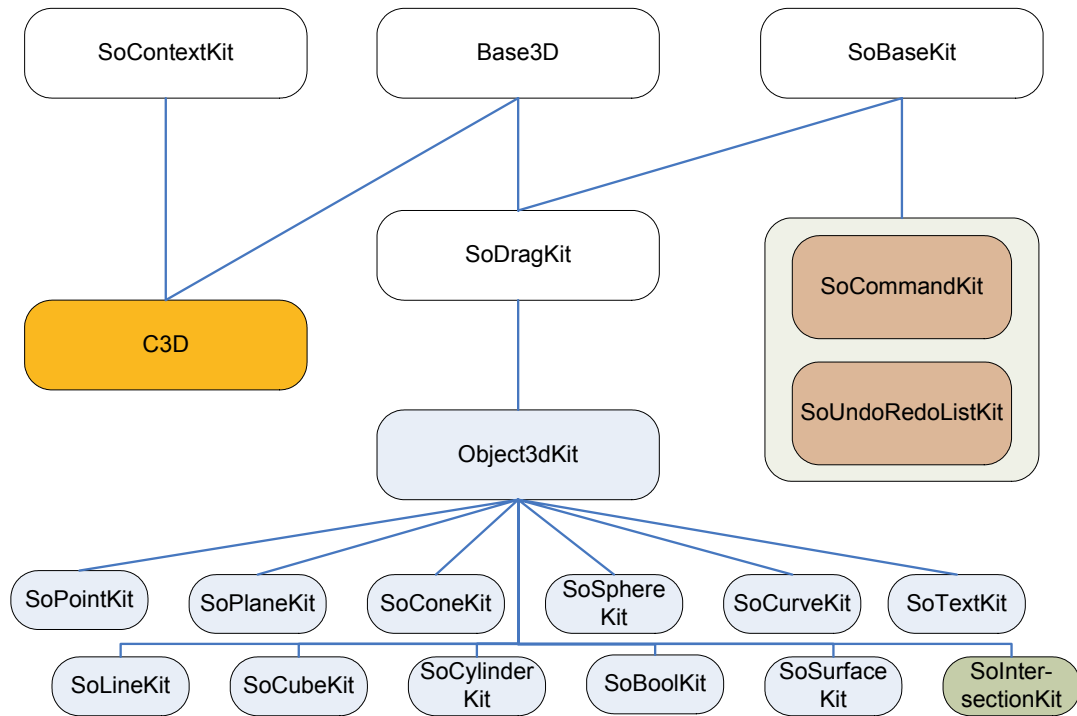


Figure 14: Class hierarchy tree of Construct3D. C3D as the user interface class manages the generation of geometric objects, derived from Object3dKit. Undo and redo functionality is provided by SoCommandKit and SoUndoRedoListKit.

C3D provides an interface to all methods operating on geometric objects. Therefore we connect widget functionality to methods for the *creation* of geometric objects in C3D. C3D must also handle events sufficiently that are propagated via Studierstube’s 3D event system. Not only events of pen button presses (to create new points for instance) are handled by Construct3D but also speech events for operating the menu system. For efficient event handling, C3D inherits methods from Studierstube’s Base3D node. In general the abstract Base3D class is used to add 3D event handling capabilities to scene graph nodes. These nodes are able to implement their own 3D event handling behavior.

C3D initializes geometric objects in methods such as “addPoint”, “addLine”,... which create Object3dKits. Object3dKit is a superclass for all geometric objects in Construct3D. Due to the dynamic functionality all objects can be dragged (moved and rotated), therefore Object3dKit is derived from an SoDragKit which is again a Studierstube node. SoDragKit allows to drag and drop objects in 3D. SoDragKit is derived from Base3D in order to grab and handle events for object dragging and from SoBaseKit which is the toplevel superclass for nodekits.

Object3dKit is the superclass for all geometric objects: SoPointKit, SoLineKit, SoPlaneKit, SoCubeKit, SoSphereKit, SoCylinderKit, SoConeKit, SoBoolKit, SoCurveKit, SoSurfaceKit, SoTextKit and SoIntersectionKit are derived from it.

Finally in the class `MaterialConstants` we define a number of methods for assigning correct materials to objects depending on the user's choice of color scheme. The details of our color scheme implementation are described in section 6.5. The file `C3DConstants` includes general constants for our construction environment.

All geometric objects are derived from `Object3dKit`. `Object3dKit` contains general functionality needed by all objects. It provides methods to set the selection and highlighting state of an object, to assign colors and materials according to the color scheme (described in section 6.5), to assign an object to a layer, to record the user identity of the user who created the geometry and to record all objects that are dependent on this object. There is a method for detecting if a user comes into the dragging area which returns if the object is interested in dragging. It is possible to generate "fixed" objects like an intersection point which cannot be dragged. The `Object3dKit` also registers if an object is deleted.

Object Creation

At creation of an object it stores an Open Inventor representation as well as its own ACIS presentation in an appropriate ACIS data type. If any property of the object changes because of an update of itself or of dependent objects, private methods of the object recalculate both representations and substitute old ones. Completely regenerating objects if properties change is a standard way of implementing dynamic changes in parametric CAD software to handle topological changes.

The Open Inventor representation is used for rendering the object, with the ACIS representation all internal calculations are done by using ACIS API functions. We utilize the ACIS geometry kernel (section 3.3) especially for calculating Boolean operations, all types of intersections, slicing, tangents and tangential planes, sweep surfaces as well as NURBS and B-Spline surfaces.

After creation the object is added to the scene graph.

Overall Scene Graph Structure

The root node of Construct3D (being an `SoContextKit` `Studierstube` application node) contains specific nodes that define the environment rendering style, lights, textures for top/front/side views (described in section 4.1) and the coordinate axes. The last child is a node called "selectionRoot" and it contains all geometric objects. All objects that the user adds are added *sequentially* as children of the selectionRoot. No matter which geometric operations are conducted there always remains a sequential list. We do not generate sub-structures below the selectionRoot. All objects are sequentially numbered. Each object's name encapsulates the object's number and position in the selectionRoot which makes it easy and efficient to find it.

We are mainly using the selectionRoot as a list to store geometric objects. It is also an Open Inventor `SoSelection` node that internally manages selection of nodes. We use this feature in Construct3D to manage selected objects.

One of the most important concepts of our implementation of dynamic geometry is handling object dependencies.

Object Dependencies

Dynamic modifications and dependencies between objects are handled by the objects themselves in a straight forward way. Assume a sphere is constructed out of a mid point and a point on its surface. We call the sphere a parent object and the points its children. At creation the sphere stores a list of its children. In addition sensors are registered with callback functions on its children. Whenever a child is updated, the sensor is triggered and the callback function recalculates the parent object. All of it is done in real time. Only a large queue of dependent objects that need to be updated or very complex calculations can cause the calculation to slow down. This is implemented consistently and works reliably.

In this context it is important to note that the list of children a parent object stores consists of *node names* of children and their sequential number in the scene graph. We cannot use pointers to child objects in this list because of our implementation of distributed Open Inventor. When distributing an application to other clients the scene graph is the distributed shared memory (as explained in detail in section 3.1). Therefore pointers to objects cannot be distributed since the memory structure on the client side is different and pointers would be incorrect. We refer to all objects and also search for them via their object names.

Object Deletion

Children also store a list of parent objects that depend on them. We use this list only in case of deletion of an object. In order to implement dynamic geometry in a consistent way all parent objects have to be deleted if one of their children is deleted. Therefore we go through the object's list of dependent objects and delete all parents. Otherwise for instance a sphere could exist without its mid-point or a point on its surface if they were deleted before. This would not be consistent. Currently our implementation gives the user a choice though. Assuming the user selects a point and chooses to delete it, the application automatically selects all dependent objects of this point. If the user is sure that he wants to delete the whole tree of the construction, deletion has to be confirmed again. Therefore we only delete an object if it has no parents or if all parents are deleted too.

In case of deletion objects are not removed from the scene graph but switched to invisible and ignored by the rendering traversal.

This has the advantage that we do not have to rearrange the list-like scene graph in case of deletion. We also do not have to recalculate dependencies or update dependency lists of objects because object names do not change. One argument that speaks for improved robustness is that in case of distributing the scene graph we cannot get into problems with consistency when deleting objects since we do not remove them from the shared scene graph. We only change field values. Nothing

serious happens in our case if one user deletes an object while a second user performs a valid operation on it concurrently – in case the second user gets master state before the system propagates the deletion to his client.

Another reason why we do not remove objects from the scene graph is that we want to keep the implementation of the undo and redo functionality as simple and robust as possible (as described later in our implementation of undo and redo).

Serializing Intersection Operations

Since the selectionRoot sub scene graph has a list-like structure because of arguments mentioned above, we must think about how to serialize intersection operations. Nested operations such as Boolean operations impose a tree-like structure. For CSG-trees an obvious choice is a structured scene graph that reflects the structure of Boolean operations. However, to keep the easy manageable list-like structure of objects, we link Boolean operation objects with their operand objects via name references.

The SoIntersectionKit stores a list of objects that have been used for the intersection operation and a reference to the resulting object. For example assume the user intersects a B-Spline curve with a NURBS surface (Figure 15). The SoIntersectionKit stores the names of the B-Spline curve and of the NURBS surface and registers sensors to both of them. It adds itself to the scene graph. In addition it calculates and generates all intersection points (which are of type SoPointKit) and adds them in sequential order to the scene graph too. It stores their names in a list. In case the B-Spline curve or NURBS surface are updated, the sensors of the SoIntersectionKit are triggered. Methods in the SoIntersectionKit reevaluate the intersection and update the position of the intersection points. If there are more solutions than before, additional points are added to the scene graph, if there are less, certain points are deleted (switched to invisible).

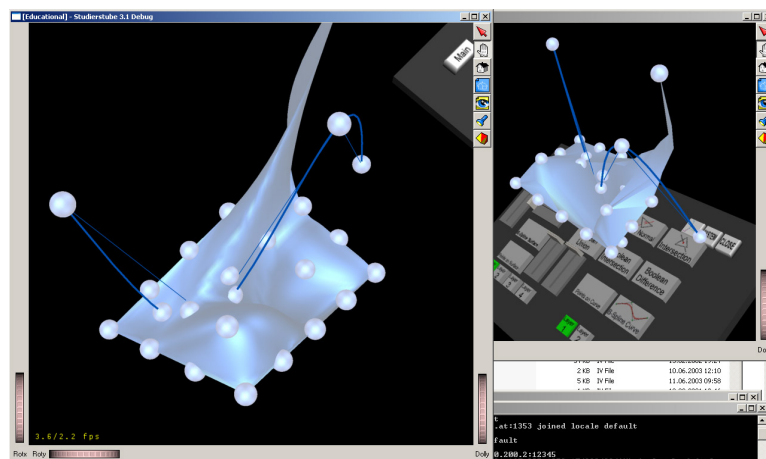


Figure 15: The left window shows a NURBS surface intersected by a B-Spline curve. The two intersection points can also be seen. This is distributed in real time to a second user (right window) who can see it from his own perspective.

An SoBoolKit contains reference to two objects that are combined. Updates are done exactly as in the case of an SoIntersectionKit by using sensors. SoBoolKit calculates the resulting Boolean object. It stores Open Inventor and ACIS representation and is treated as any other geometric object.

Continuity for Dynamic 3D Geometry

We want to add a few thoughts concerning the example in Figure 15 of an intersection between B-Spline curve and NURBS surface. From a dynamic geometric point of view deliberately adding and removing points if new solutions appear or disappear totally destroys continuity. It is vital to keep track where intersection points go after an update and which ones disappear and which new solutions appear. We implemented this in a very simple way, being fully aware that our implementation can only keep continuity in some cases but cannot fully resolve ambiguities (details on this problem in dynamic 2D geometry can be found in section 2.4 and in [79]). We “keep track” very simply by calculating the distance of an updated point to all previous points. We assume that an “old” point with the shortest distance to the “new” updated point corresponds to the previous location of that point.

In the example of an intersection between B-Spline curve and NURBS surface it is important to note that continuity is even theoretically very difficult to achieve. It is theoretically not trivial to determine the exact number of possible solutions for that intersection problem. Both elements are only piecewise rational. However, in order to keep track of the intersection points at least the maximum number of possible solutions must be known. New solutions can only be added to the scene if the maximum number of solutions is not surpassed already, otherwise they must correspond to existing solutions.

Implementation of Undo and Redo

In Construct3D we implemented a global undo which means that the last change made by any user is undone. This was rather straight forward to implement.

We use a classical *history list* which is documented [118] to be the most common way to record the sequence of operations. On startup of the application an SoUndoRedoListKit is initialized (right part of Figure 14). Whenever a user conducts an operation, it is added to the SoUndoRedoList. This list constructs a SoCommandKit which encapsulates all data of the current command. A command name, object name, object type, in case of a point its position in space, the user ID of the user who executes the command and a timestamp are saved in the list. Further details are given in section 4.3. At an undo the last operation n is reversed and an undo counter is set to $n-1$. A redo executes the command after the position of the undo counter again. Our undo behaves exactly like the undo in single user applications and was rated to be very effective during our evaluation (section 8.3).

Since children are not removed from the scene graph if deleted, our undo and redo implementation is very straight-forward and robust. All connections to children and parent objects are kept in case an object is deleted. It is only switched to an invisible state and is not updated if invisible. Undoing deletion is simple since we just switch to visible again.

Object Manipulation

We implemented a very intuitive way of object manipulation that was continuously refined from the first implementation until today. We use direct manipulation of objects in 3D space for all operations as described in detail in sections 6.4 and 6.5.

The implementation of highlighting is done in a straight-forward way. We constantly calculate all distances from the pen's position to all geometric objects in the scene. A private method of each object returns its shortest distance to the given pen position. At every event we get from the pen, we take its translation and reevaluate all distances. We determine the nearest object and highlight it.

We do highlighting by using a wireframe grid of the same model that we superimpose on it (see section 6.5, Figure 45 middle). In order to display the wireframe model we reference a second identical copy of the object's Open Inventor representation. If an object is highlighted it switches on the second model using a different rendering style (wireframe) and both Open Inventor models are rendered at the same time.

In order to implement the preview feature (explained in section 6.5, Figure 47) in an efficient way, we have the following approach. We generate a preview object exactly the same way as normal objects, we only change its rendering style to wireframe. If the pen is moved into a widget the corresponding Object3dKit checks if the input elements are valid and sufficient for its construction. If they are, we add the preview model to the scene graph and remove it again if the user's pen leaves the widget. Preview objects are obviously not added to the undo-redo history list.

We are aware that this simple implementation in combination with our flat sequential scene graph has consequences if users collaboratively work on constructions. Let us assume that user one produces a preview object. A problem occurs if user two decides to select and use the preview object of user one for further construction. As soon as user one removes the preview object (by moving his pen out of the widget) it is removed from the scene graph. This would result in an invalid operation that user two conducted in the meantime. It also introduces a gap into the sequential list of objects if one object in the middle is removed and would require reordering the list. In order to get hold of this problem we currently forbid selection and generation objects while a preview model is displayed.

Cross-Platform Development

Our implementation is done in C++ and is based on various frameworks. First of all we utilize our AR platform Studierstube (section 3.1) which is based on the Open

Inventor implementation Coin and uses OpenGL for rendering the scene graph. Qt from Trolltech is used as a multiplatform, C++ user interface framework. Qt supports the development of cross-platform GUI applications. OpenTracker (section 3.2) is used as our tracking framework. It uses ACE (ADAPTIVE Communication Environment) as a high-quality, high-performance network communication implementation. All network communication in OpenTracker and Studierstube is done via ACE. Xerces is used by OpenTracker for parsing XML configuration files.

Construct3D as a Studierstube application is dependent on all the libraries and toolkits mentioned above. In addition Construct3D depends on ACIS (currently in release 13) which is our geometry kernel.

Qt and ACIS are used with free educational licenses whereas all other software components are open source under different licenses.

Since all of our implementation is based on C++ and all frameworks are available for multiple platforms, Studierstube is running under Windows (from 95 to XP), Linux and MacOS X. Construct3D is running and has been tested under Windows and Linux too.

4.3 File Format Description

As mentioned in the previous section all construction steps are saved in an SoUndoRedoListKit. Since it is an Open Inventor node kit, we write out this list in the Open Inventor text file format. The UndoRedoList consists of SoCommandKits which encapsulate all data of a command in Construct3D. It is used for undoing and redoing operations but we also save it when saving a construction. It reflects in detail every single construction step of every user. When loading pre-saved constructions we only load the SoUndoRedoList into Construct3D. The whole construction history is loaded which allows users to undo (and afterwards redo) all steps of a loaded construction. Especially for educational purposes this is of advantage since students can “go back in time” to study how a construction was generated and can replay/redo it again.

In addition commands are saved in a convenient and readable format in the UndoRedoList. Since it is a text file it can be easily edited. Each entry – a CommandKit – consists of a command name, object name, object type, in case of a point its position in space, the user ID of the user who executes the command and a timestamp. Under various circumstances it is of good use to being able to change existing constructions or write new ones in this format directly. For example we had to create a regular tetrahedron for the 5th example of our evaluation (see section 7.1). Due to a lack of measuring and constrained modeling functionality in Construct3D we were not able to directly draw a regular tetrahedron. Therefore we calculated all coordinates of the vertices and entered them in a saved file. Below

we explain in brief how a simple “Hello World” example – constructing a sphere out of two points - looks like when saved with Construct3D:

```
#Inventor V2.1 ascii

#initializing the SoUndoRedoListKit
DEF undo_redo_List SoUndoRedoListKit {
  undo_redo_List [

# User 0 adds a point P_0 with coordinates (0.1, 0.5, 0.2)
    SoCommandKit {
      command "add"
      objectName "P_0"
# add this line to "lock" a point so that it cannot be moved
#      objectType "fixPoint"
      position 0.1 0.5 0.2
      userID 0
    },

# We select P_0
    SoCommandKit {
      command "select"
      objectName "P_0"
    },

# add a second point P_1 here, analog to P_0, and select it
    [...]

# add a sphere S_2 using the points P_0 and P_1
    SoCommandKit {
      command "add"
      objectName "S_2"
      objectType "Sphere"
      selectedObjectNames [ "P_0", "P_1" ]
      userID 0
    }
  ]
}
```

5 Educational Scenarios

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 describe the hardware setups that we built and summarize our experiences.

5.1 Immersive Setup

Basic AR Lab Multi-User Setup

In our standard Studierstube setup, we have 2 collaborating users wearing HMDs for a shared virtual space, and holding fully tracked interaction props in their hands (Figure 16). One dedicated host with 2 graphic ports renders stereoscopic views for both users. We are using this setup for demonstrations and also as part of other setups such as the Lab@Future evaluation setup (section 5.5). While it allows for first-class experiences on the students' side it significantly restricts the use in larger groups. We can add a second host to render views for a 3rd user and maybe 4th user. The number of users is restricted by the available space in the lab but mainly limited by the number of cameras of our optical tracking solution (see 6.1). More users in the environment cause more occlusions of devices and marker sets of other users in the camera images. This results in tracking "blackouts" and bad tracking quality. Our 4 cameras together with optimal marker body design (see section 6.1) are currently sufficient to track 2 users in very high quality. Tracking more than 3 users in the same environment would require considerably more, expensive cameras to be able to track the whole space even when occlusions occur for some cameras.

The unique feature and big advantage of this setup is that users can actually “walk around” geometric objects which are fixed in space. This is very much appreciated by students and teachers (also mentioned in the evaluations in chapter 8). It actively involves students. The geometric object is not abstract anymore but in spatial relation to the learner’s own body. We think these are key features when improving spatial abilities with Construct3D.

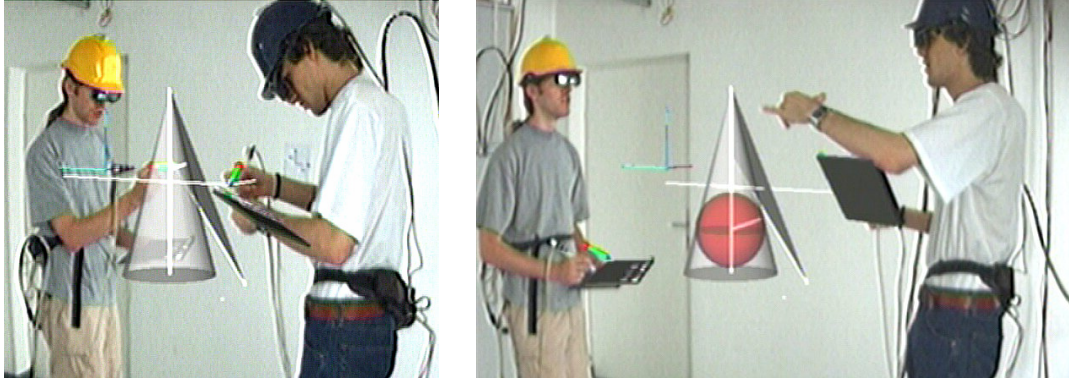


Figure 16: Construct3D (here shown in a tethered setup, recorded with a tracked DV-camera augmented in real-time) allows exploratory construction of geometric tasks. Seeing in 3D reduces the level of abstraction of geometric problems. Shown is a collaborative attempt to construct the proof of Dandelin (described in section 7.1).

Hybrid AR Classroom

We can imagine to integrate the basic AR lab setup in the following teaching scenario. In a school lab 2 students or a teacher and a student work with HMDs while the other students watch or do their own constructions on their desktop machines (for example by using setups as described in 5.3). 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 (Figure 17). Just like in conventional classrooms, students can take turns at using the HMD and work in front of the class.

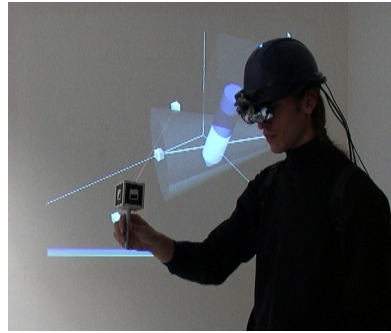


Figure 17: A teacher is working in Construct3D with the mobile AR setup while a live monoscopic video of his current construction is projected onto a projection screen behind him.

5.2 Semi-Immersive Setups

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 (Figure 18). 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 environments, and single-projector displays (without the capability to render stereoscopic images) are affordable for classroom use.

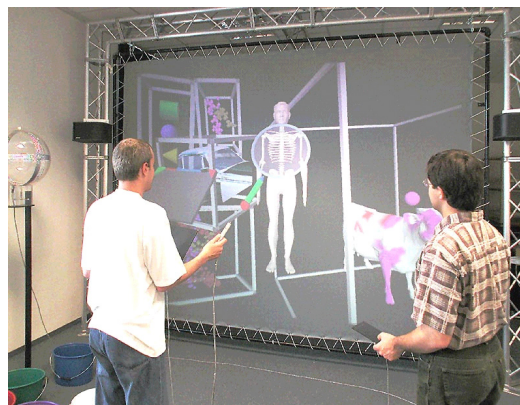


Figure 18: A back projection wall with polarized projection running the Studierstube platform. Users wear polarizing eyewear (not shown) to see and interact with stereoscopic images.

Baron Table Setup

In this semi-immersive VR setup one student who wears shutter glasses works in front of the BARCO Baron table (Figure 19) and other students standing next to him can watch the construction process. The stereoscopic images are rendered correctly for one “master” user only whose head and interaction devices (panel and pen) are fully tracked in 3D space. An advantage of this setup is that the immersive feeling is very good because of the huge screen size of the projection table. Usually the user stands very close to the table and his field of view is mainly covered by the projection screen. Objects seem to come out of the screen.



Figure 19: The BARCO Baron table whose projection screen has a size of 67” (1.7 meters diagonal). In both images a monoscopic video is displayed on the screen of the Baron table.

In an ongoing project “Collaborative Augmented Reality für den Einsatz in der Lehre” we are evaluating the educational applicability of this setup.

5.3 Hybrid Desktop Setups

We present different hybrid desktop setups that use low cost hardware only. These or similar configurations are affordable and can be set up in schools nowadays. Devices that we use include shutter glasses, an audio headset, an inexpensive tracked glove called P5 glove and webcams.

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. A number of scenarios are possible.

- The teacher’s construction may be distributed to all students and they watch it on their screens. They can view it from different angles, interact with it and continue working on it.

- Students do their own construction without seeing the teacher's work on their screens.

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 Figure 20). 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. A teacher can also choose a guided mode, e. g., by locking the students' views to the teacher's viewpoint.

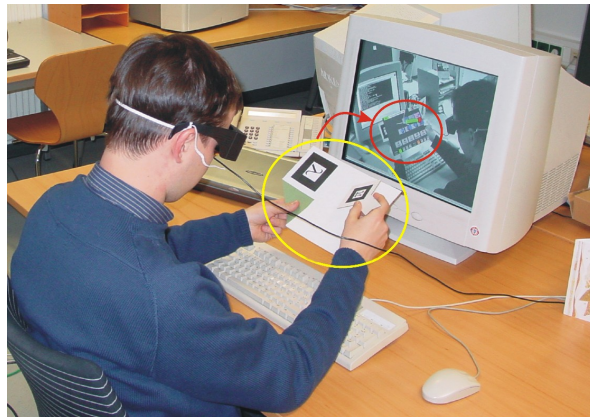


Figure 20: A user works with our desktop VR system.

In this setup (Figure 20) 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 in Figure 20) as an overlay over the video image.

An early version of this setup used hand held props made of paper as can be seen in Figure 20. However, in order to select menus and set points in space, a button is needed on a pen. In order to substitute the paper pen with an appropriate 3D input device we tried a position tracked glove by Essential Reality. The P5 glove was released at the end of 2002 at a very low price of 80 USD. It comes with a free SDK and drivers for Linux and Windows. For this price the quality is surprisingly good. It is optically tracked with low latency by using infrared LEDs which are mounted on the glove. It features bending sensors which are very accurate. They measure the angle when bending fingers. We implemented a clicking gesture which is activated by bending the index finger more than 50 degrees. Filters are applied to get smooth tracking data. Figure 21 shows the P5 glove in use.

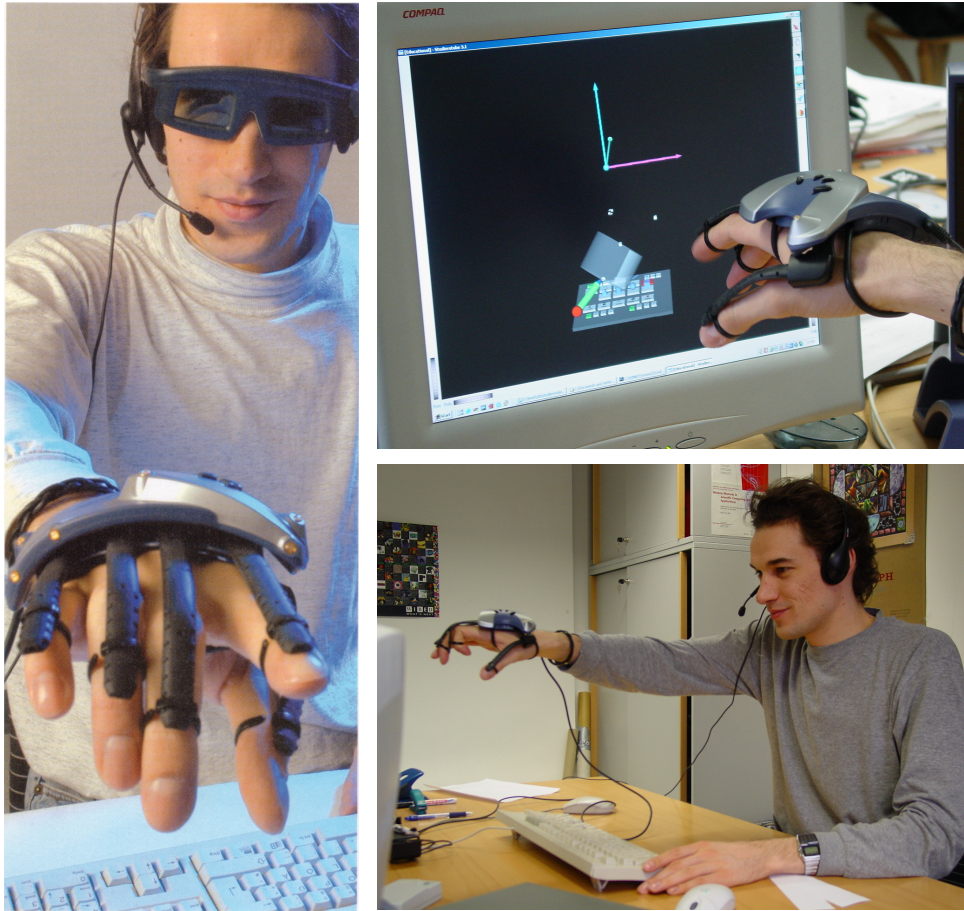


Figure 21: Left: The P5 glove used together with shutter glasses and a headset for our speech interface. By courtesy of Thomas Jantzen, published in Wiener Journal 03/2003. Top right: We control the pen in Construct3D with the P5. A desk mounted receptor, nearly as high as the monitor, for optical tracking of the glove is partly visible at the right border of the image. Bottom right: Bending the forefinger causes a click, by bending and holding points are dragged.

The P5 glove is a good quality, low cost and easy to use 3D input device. It is easy to use and there is no learning effort. The tracking volume is limited but sufficient to track the volume in front of the monitor. There is only one disadvantage to this type of input device, which is not a specific P5 glove problem. Geometric objects, especially points are small objects on the monitor and it is difficult to find and grasp them accurately in 3D space with monoscopic rendering only. Stereoscopic vision gives better depth perception. Measurements can be taken to improve exact construction in 3D space as described in section 6.3. However, after about 10 minutes of continuously using the glove (such as in the bottom right image of Figure 21) work starts to get strenuous. Without any support to rest the arm, elbow or hand on, the arm is getting tired by holding it out straight to select, set or drag objects. By resting the elbow on the desk, the working volume is very limited though. We do not know yet how to change our working style or how to adapt our

application to overcome this problem. For longer construction sessions work with the P5 is strenuous.

Basic Desktop Interface with Speech Input

In addition to our integration of various 3D input devices we also tested speech input. We integrated a fully functional speech interface into Studierstube. All menu functions can be executed by using speech commands. Speech recognition is based on Microsoft's Speech API (SAPI). We implemented speaker independent command recognition. A set of commands is defined for each sub-menu (PIP sheet). The system does not have to be trained to a specific user. If a user says a command, it is recognized by using SAPI methods in OpenTracker (section 3.2). In OpenTracker we check if the given command lies within the command set of the current sub-menu. If it does we send it to the application where the appropriate function is triggered. By using command sets we get a better detection rate since only a limited number of commands must be recognized for each sub-menu. In addition we filter commands via command sets to avoid that menu items from different sub-menus are triggered.

We combined and tested speech input with all desktop setups presented in this section. It considerably increases working speed since we do not have to navigate the pen to the PIP anymore. On single user desktop workstations speech input is really very convenient to use.

Speech recognition may cause problems if there are many background noises or discussions going on in the surrounding environment. If multiple users work close to each other by using the speech interface the background noises might lead to undesired detection of speech commands and can execute unwanted functions of the application. For optimal speech recognition quality a good soundcard is also of advantage. Standard on-board soundcards provide very low quality.

Finally we present a basic desktop interface which we use for our own development work, for quick drafts and tests. The PIP is displayed in the lower right corner of the screen on top of a full screen window with 3D graphics. Rendering can be monoscopic or stereoscopic. We use the mouse to navigate a little yellow cone-shaped cursor on the PIP. The mouse cursor can be seen in the lower right corner of Figure 22 as a yellow dot. The whole menu system is operated with the mouse. In order to move the pen in 3D, to set and select points we still use keyboard input.

For developers who are used to keyboard input this is quick and convenient. Because of usability reasons keyboard input cannot be recommended for educational purposes though. For convenient use in schools we plan to work on an easy to use desktop interface for Construct3D as described in 9.2.

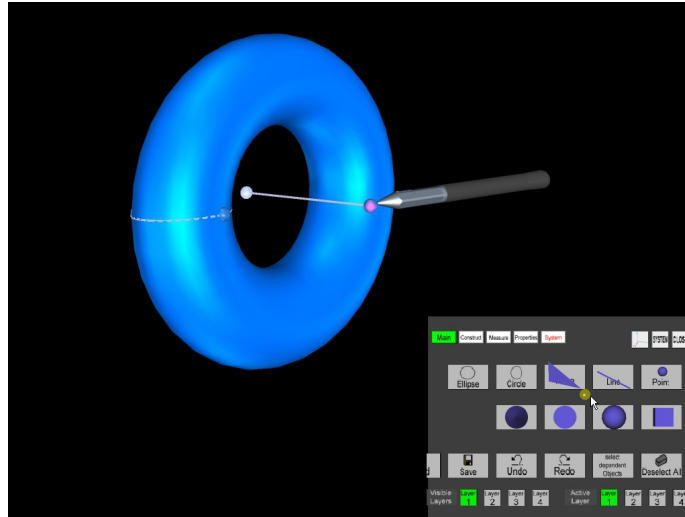


Figure 22: A torus drawn with the basic desktop interface. The PIP is displayed in a lower right window on top of the full screen window.

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. The Lab@Future evaluation setup 5.5 is also a remote collaboration setup.

5.4 Mobile Setups

In the national project “Mobile Collaborative Augmented Reality”, we explored the possibilities of a mobile 3D workspace for collaborative AR. Our system is characterized by the following properties: A mobile platform allows full stereoscopic real-time display. A freely configurable tracking system allows fusion of arbitrary sensors that complement each other to provide 6DOF manipulation of 3D objects in the near field, i.e., within arm’s reach. With our 3D user interface management system, users can arrange multiple arbitrary 3D applications around their body and carry them along.

Our first version (Figure 23 top) used a see-through HMD, tracking via a head mounted camera and an orientation tracker on the users head. The user interface consists of pen and tablet, both tracked by the camera. The tablet displays user interface components and provides tactile feedback to the user.

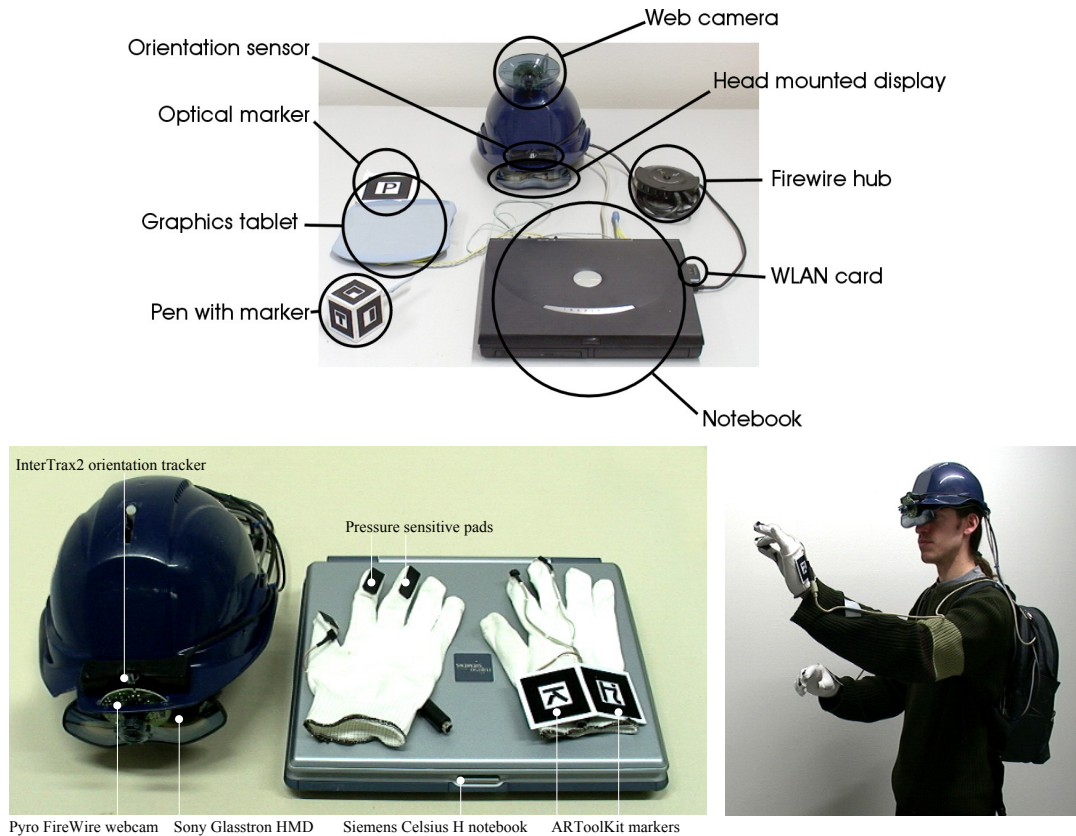


Figure 23: Top: The hardware configuration of our first mobile system. Bottom: The second mobile system. Left: Overview of the components. Right: A user wearing the system

The second version (Figure 23 bottom) substitutes tracked gloves for the tablet and pen interface. These allow two handed interaction and free the user's hands. A tablet like small interface be fixed to the user's wrist to provide feedback for 2D user interfaces presented there.

Mobile "YO!Einstein" Demonstration Setup

At a science fair called YO!Einstein at the Technical Museum Vienna we presented Construct3D on the second version of the mobile system in 2002 (Figure 24) and in 2003 (Figure 26).

In our presentation setup in 2002 a single cable was attached to the mobile user which connected the graphics port of the notebook with the video projector. This limited the mobility to the area around the projector. The projector displayed the video image as seen through the user's head-mounted camera plus the rendered virtual scene on a big projection screen. Visitors watched on the big screen the actions of the mobile user.



Figure 24: Upper left: A student wears the mobile backpack and works with Construct3D while others watch his construction of a projection screen. Upper right: Clicking is done by pressing a sensor on the thumb. Lower left: A student gets an introduction how to use the panel to select menu functions. Lower right: Excited students watch a construction.

In 2003 we demonstrated a similar setup but the major change was a wireless video transmission system. It allowed us to walk around the whole technical museum. Visitors of our stand still saw on the big screen what the mobile user saw and what he was doing (Figure 26 and Figure 26).



Figure 25: Left: Our setup in 2003 was fully mobile. The big projection screen is visible in the background. Right: Presenting our hardware. A tribute to colleagues for their help.



Figure 26: Interested crowds of students.

Augmented Classroom - Mobile Collaboration

The Augmented Classroom [140] is a visionary distributed scenario that blends collaborative augmented reality and tangible user interfaces. We demonstrated it at the ISMAR 2002 conference.

It allows two users equipped with wearable mobile AR systems (as in Figure 23) to interact in a shared workspace. They can help each other and point out interesting features in the models. 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. A tangible marker is placed in front of them on the table. The marker is identified by a camera above their heads and every construction they do is automatically attached to that marker by the system. To let a larger audience of students participate in the results of the ongoing construction of the mobile users, a projection screen presents the geometric model as well. Figure 27 gives a schematic overview of the collaboration scenario. On the right, two users wearing mobile AR kits interact with a 3D application on a table. On the left, a student interacts with another construction appearing on the overhead projection. Note how the cameras are arranged to observe the tangible objects used for interaction. All computers are interconnected with wireless LAN.

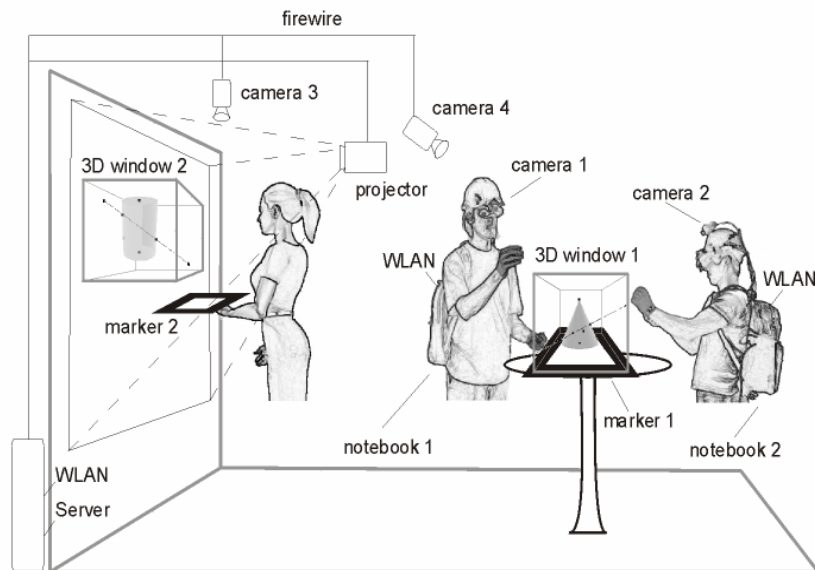


Figure 27: Schematic overview of the Augmented Classroom.

The work of the mobile users (i.e., a teacher and a student) is automatically attached to a tangible marker on the table. Whenever any other student feels like it, he can pick the marker on the table. By taking it in front of the projection wall users can inspect the construction that is tied to it (Figure 28). Any other marker that was registered with the system before can also be used. Students inspect the model from any side by moving and rotating the marker. The object displayed on the projection screen reacts to movements of the markers in an intuitive way.



Figure 28: Spontaneous collaboration at the projection wall. Users bring their work in form of markers. These are identified by a camera above the projection screen and the corresponding applications are loaded automatically.

Figure 29 shows that tangible markers can also be used in a setup where for instance the teacher wears a mobile AR kit only.

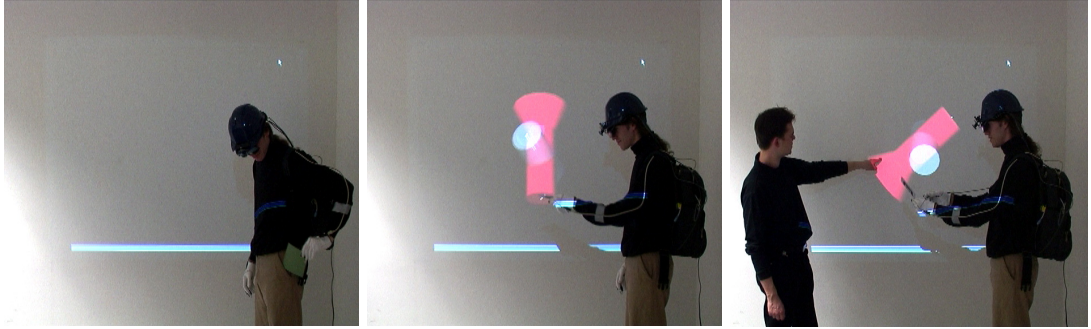


Figure 29: A hybrid configuration where a teacher wears a mobile augmented reality kit and sees overlaid graphics in his HMD. Simultaneously, the class sees what the teacher sees on the projection wall in the background. The teacher pulls an application out of the pocket (left, middle). A student approaches the projection to ask a question (right).

It is also possible that nobody wears any mobile kit and the teacher uses tangible markers only to present 3D models to the students. The projection could be stereoscopic as well on a polarized projection wall with students wearing polarized glasses. This would be an alternative to presenting 3D models in geometry classes. Nowadays teachers use wooden or plastic models of geometric objects or of proofs (such as a model of the proof of DANDELIN – see section 7.1). These are very expensive and difficult to get. A projection screen for presenting virtual models would be a very flexible alternative.

5.5 Lab@Future Evaluation Setup

Our work is also funded by an EU project called Lab@Future. The Lab@Future project is addressing the needs of European school education utilizing laboratory experiments at secondary school level. Lab@Future aims at developing new possibilities for technology-based laboratory education. The project goal is to provide groups of students with mobile and distant access to learning experiments. Construct3D is one of three experiments.

In our evaluation scenario there are six students working in two different remote labs. The first lab with two students features the immersive Construct3D setup (our “basic AR lab multi-user setup”), where the students wear HMDs, their head and their props (the Personal Interaction Panel and pen) are precisely tracked with a high-quality optical tracking system. In addition a teacher is with them to guide through the learning process. A tracked camera shows the teacher what the students are doing on a big BARCO projection table. He does not only see 3D graphics - the student’s construction – but also sees the students in relation to the geometric object, filmed by a camera shown as a video background (Figure 30). 3D graphics and the video stream are combined and displayed in real time



Figure 30: The teacher follows the construction process on the BARCO projection table. He cannot only see the video image of both students but can also see the virtual geometry they are working on, correctly registered with the position of the students.

The teacher can position his camera freely in 3D-space to get an optimal view of the students' work at all times. This enables him to follow the geometric construction and gives him a realistic impression of what is going on (Figure 31).



Figure 31: A photo of the teacher's view as seen on the BARCO table. Two students are working with Construct3D on a surface of revolution. The teacher sees their real and virtual pens and panels and also the 3D graphics.

The teacher's view is transmitted to a second BARCO table placed in a second lab where four students are participating in the experiment. For transmission of the video background we use the teleconferencing protocol H323. All four students sit in front of desktop PCs (one machine for each of them) with multimedia facilities and collaboration tools installed (text chat, audio conferencing and whiteboard). All 6 students are wearing headsets. The four observer students can follow ongoing constructions on their monitors with the help of distributed Studierstube and can communicate and collaborate via collaboration tools.

Meanwhile two students in the first lab are working on the actual construction using the immersive setup.

In the lab of the observer students the second BARCO projection table (Figure 30) also displays the teacher's view (Figure 31) of the current construction. Students can get up and walk to the BARCO table any time during the session and can watch and discuss the ongoing construction. They constantly communicate and collaborate with students in their room and with the teacher and students in the other room who wear HMDs.

Since this scenario takes place in two different physical places (remote labs) distance education is integrated in this experiment. The communication between the two groups is bi-directional since active students are able to send information to observer students and vice versa.

To integrate students at distant locations we automatically export the construction as a VRML model every 30 seconds. It is imported and refreshed at regular intervals in a multi-user world. This allows groups of students to watch the ongoing construction with a standard VRML web browser plugin.

Figure 32 shows students working in the AR lab. Figure 33 gives an overview of the activities in the lab of the observer students.



Figure 32: Snapshots from our internal tests – AR lab



Figure 33: Students in the 2nd lab communicate and collaborate with students in the AR lab. On the BARCO table (middle left), they see the teacher's view from the other lab.

6 User Interfaces and Usability Design

This chapter describes a practical approach, thoughts and methods used to improve the user interface and usability of Construct3D. It provides guidelines to usability design for educational AR applications in general but is especially tailored towards our application. All improvements described in this chapter were implemented with the intention of improving the educational processes of learning and teaching. As usability design must always be done 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.

As mentioned in chapter 2 many factors contribute to the development of an educational VR/AR application (see Figure 2). Therefore a range of attributes have to be optimized, including software- as well as hardware.

Our main target platform throughout the development process was our basic AR lab setup (described in section 5.1) with 2 users using HMDs. The reason why we optimize for this platform is that we expect superior results regarding improvement of spatial abilities (section 2.6) compared to the other setups described in chapter 5.

During the last few years most hardware parts in our lab were exchanged and replaced by newer technologies. Many software components of the underlying Studierstube system have been rewritten and the open source Open Inventor implementation from SGI was replaced by Coin – a more advanced open source implementation by the Norwegian company “Systems in Motion”. All these changes to the underlying software and hardware technology made adaptations necessary. Not all changes were automatically improvements at first sight but they offered chances to improve usability. We will start describing which hardware changes were conducted and how we adapted the user interfaces and application to the changed environment. Afterwards software improvements are discussed which are applicable to educational applications, 3D modeling applications and VR/AR applications in general.

6.1 From Magnetic to Optical Tracking

Until summer 2002 we were using a magnetic tracking system called “Flock of Birds” from Ascension Technology Corporation with an extended range transmitter. With an update rate of 100Hz we were able to track 6 different sensors which were used for head tracking and tracking of user interfaces (pen and pad) for 2 users. Figure 34 shows the magnetic tracker set up in our lab.



Figure 34: The magnetic tracking system in our lab. Left: The tracker server together with RS-232 interfaces was mounted in a cupboard. Right: The Extended Range Transmitter was positioned on a wooden construction near the ceiling of the room.

Magnetic tracking technology in general has its pros and cons. For educational purposes reliability is the biggest advantage of magnetic tracking. No matter what students do, no matter how they construct and if they occlude each others devices, tracking data is received at all times. Only strong electromagnetic fields for instance near monitors, distort the magnetic field of the tracker and cause distortions of tracking data. Our magnetic tracking system was very user friendly to operate. However, one major disadvantage were cables (as can be seen in Figure 34 left). A magnetic sensor plus cable was attached to every device – HMD, pen and panel. Figure 66 left from our first evaluation (section 8.1) shows the number of cables that were attached to the devices. We always had to take care that the cables of different users did not tangle.

Another disadvantage of magnetic tracking is “bad” tracking accuracy. In a volume of 3x3x3 meters we had accuracy deviations of 3 millimeters to 2 centimeters on average depending on the users distance to the extended range transmitter. Our first evaluation (section 8.1) revealed that the magnetic tracking system was not accurate enough to do exact 3D modeling with Construct3D without further adaptations of the application. As discussed in detail in section 6.3 this also influenced our decision to implement dynamic 3D geometry. Measurements how to cope with tracking inaccuracies in virtual 3D modeling environments are also briefly discussed there.

We were able to acquire an optical tracking system in summer 2002 from Advanced Realtime Tracking GmbH (ART). It is an infrared-based optical tracking system called ARTtrack1 with 4 cameras. Every camera (Figure 35 left) has a built in infrared spotlight and an on-board Linux server which performs basic image recognition to detect retro-reflective markers in the image. Retro-reflective markers reflect infrared light which is emitted by the camera's infrared spotlight. Every configuration of more than 3 markers - a so called "marker body" - is known to the system because of a one-time detection and calibration procedure. The distances between markers in each body must be unique for definite identification. A marker body must be visible on at least two camera images for the tracker server to calculate the marker body's position in 3D space.



Figure 35: Left: A camera of the ARTtrack1 optical tracking system. Right: All cameras are mounted to the ceiling in our lab (visible in the upper right corner of the image). They overview the lab and track retro-reflective marker bodies.

Optical tracking has big advantages compared to magnetic tracking. First of all optical tracking systems are known to provide very high tracking accuracy. Tracking data deviations in our setup are very small (maximally 1-3 millimeters), independent of the users position within the tracking volume. The tracking data update rate is 60Hz compared to 100Hz of the magnetic tracker but we are not limited in the number of tracked objects anymore. For the magnetic tracker we had 6 wired sensors. In case of the optical system the manufacturer's tracking software currently supports up to 20 concurrently tracked objects. This would theoretically enable us to track 6 users - all of them being head-tracked using tracked pen and PIP.

This theoretical limit is in contrast to a practical limit of 2-3 users within our tracking volume of 4x3x2 meters. More users in the environment cause more occlusions of devices and marker sets of other users in the camera images. A larger amount of marker sets on helmets can be tracked without difficulty since they are not occluded by other objects. Pen and PIP get easily occluded by user's bodies. Occlusions result in tracking "blackouts" and bad tracking quality. Our 4 cameras

together with optimal marker body design are currently sufficient to track 2 users in very high quality. Tracking more than 3 users in the same environment would require considerably more, expensive cameras to be able to “cover” the whole space even when occlusions occur for some cameras.

In practical use our optical tracking system is currently nearly as reliable as magnetic tracking which is very important for usability in general. We spent a lot of time investigating how to compensate for our initial tracking “blackouts”. Originally we used 4 markers on pen and PIP which were placed in a plane. This is suboptimal since it easily happens that all 4 points project to a line in one camera image and then two more camera images are needed to identify the spatial situation. We redesigned our marker bodies. For the pen we use 4 markers that form a tetrahedron. For the PIP we use 7 markers – 3 in the upper right corner of the panel, 4 in the upper left corner or vice versa. Some markers point “outwards” – from a user’s perspective to the left and right - so that if a user’s body occludes the PIP these markers are still visible by the camera behind the user.

Our design required a large amount of retro-reflective markers. Because of very high costs of these markers, we decided to build them ourselves. We researched retro-reflective materials and retro-reflective ink and got the best results with a retro-reflective foil from 3M. On an internal “marker making day” we built over 100 markers that we use internally for various research projects.

Finally we want to document an additional usability improvement that is related to tracker inaccuracies. Users noticed that while wearing an HMD the image slightly jittered. This is independent of the tracking technology and caused by a slight jitter in tracking data. We apply a filter only to HMD tracking data in order to compensate for that jitter and to deliver a stable image. This is also done to avoid symptoms such as cybersickness.

6.2 Cybersickness

Cybersickness is a term to describe a type of sickness experienced by users of head-steered VR/AR systems with symptoms that parallel symptoms of motion sickness [104]. Over 80% of individuals exposed to VR simulations of 20 minutes reported increases in sickness symptoms [31, 76, 164]. While anecdotal evidence for negative side effects is highest in immersive virtual environments (VEs), many other factors than simply having a visual scene in an HMD appear to contribute to the experience of negative side effects in VEs [121]. Not only are there numerous factors suspected of causing side effects in VEs, there are many symptoms that have been observed as well. Headaches, dizziness, vertigo, nausea, eyestrain, sweating, and in rare cases, vomiting can occur. It is the complex nature of both the causes and effects of motion sickness in VEs that creates problems for the

researcher attempting to study the issue. An excellent insight into this research area is given in [105].

We will give a brief overview of what we think is most relevant to our application and has potential to explain the negative side effects that were experienced by our users in both evaluations (sections 8.1 and 8.3). We will summarize possible usability improvements that have potential to reduce these effects. The most widely accepted theory as to the cause of cybersickness is the sensory conflict theory.

The sensory conflict theory (sensory rearrangement, sensory mismatch, perceptual conflict, cue conflict, or stimulus rearrangement) has developed as the main theory for motion sickness accepted by most scientists today [64, 65, 87, 105, 114, 121]. It is based on the premise that discrepancies between the senses which provide information about the body's orientation and motion cause a perceptual conflict which the body does not know how to handle.

We observed less severe and less frequent occurrences of cybersickness than reported in many other studies. In our application users have control over their movements and also have control over the movement of virtual objects. As Stanney and Kennedy [147] state “active motion is superior to passive motion in minimizing cybersickness” and “cybersickness may eventually be overcome under active control conditions, because users can predict and thus adapt to their movements in the VE”.

Therefore we believe that sensory conflict theory is of little relevance to the specific side effects that participants of our evaluation studies experienced. An aspect which is probably more relevant to our problem is low frame rate and lag.

Tracking Errors and Lag

Hettinger and Riccio [58] state that visually induced motion sickness can occur in two different situations. In the first, perceivable and excessive lags in the display's visual field from head motion in a HMD initiate visually induced motion sickness. They believe this problem will be eliminated as the speed of the VE controlling hardware and software improves. This is not such a problem in Virtual Reality but becomes especially obvious in Augmented Reality [5]. Since the user only sees virtual objects in VR applications, registration errors result in visual-kinesthetic and visual-proprioceptive conflicts (also reported in [114]). Because the kinesthetic and proprioceptive systems are much less sensitive than the visual system, visual-kinesthetic and visual-proprioceptive conflicts are less noticeable than visual-visual conflicts. For example, a user wearing a closed-view HMD might hold up his real hand and see a virtual hand. This virtual hand should be displayed exactly where he would see his real hand, if he were not wearing an HMD. But if the virtual hand is wrong by five millimeters, he may not detect that unless actively looking for such errors. The same error is much more obvious in a see-through HMD, where the conflict is visual-visual.

These errors are introduced not only by a lag of tracking data but also occur at slow rendering speed. As reported in our second evaluation (section 8.3) there were times during when frame rates dropped down to unacceptable levels causing jerking images. Of course this caused visual-visual errors. We are constantly working on optimizations to improve rendering speed in order to reduce these errors.

HMD Setup with Helmets

In our laboratory dual user HMD setup (section 5.1) we fixed each Sony Glasstron into a standard helmet of construction workers so that the glasses do not move freely on the head of the user but keep a rather fixed position (see Figure 36). The size of the circumference of a helmet can be adjusted to the size of the head. It is not possible though to adjust the helmet to the individual shape of the head. For head tracking retro-reflective markers are mounted on top of the helmet which theoretically give us a fixed distance between display and tracked reference point on the helmet (this distance is only theoretically fixed since the display of a Glasstron can be tilted which makes an exact registration of our interaction devices nearly impossible and recalibration would be necessary after each user moved the display to accommodate his eye position).



Figure 36: The Sony Glasstron is fixed inside the helmet to keep the distance to the set of retro-reflective markers (on top of the helmet) constant.

During evaluations and visits of teachers, students, researchers and other guests it turned out that the helmets do not fit all head sizes. They can not be adjusted flexibly enough. Especially women but also men sometimes comment that the helmet is heavy and causes great pressure on their head, in most cases on their forehead. After working with the HMDs for longer periods of time, these users usually have red pressure spots on their forehead. We do not underestimate the fact

that any kind of bad fitting head wear can also cause headache. This could be one of the most likely reasons why many of our users get headache.

Therefore we will substitute the construction workers helmets with light weight bicycle helmets. These are easy to adjust to the individual shape of the head and comfortable to wear.

Stereoscopic Viewing

In addition to the above mentioned theories which are also valid for monoscopic viewing, stereoscopic viewing introduces additional problems. A general problem with HMDs is that each person has a different intraocular distance (i.e., the distance between two pupils) meaning that stereo images have to be separated by that distance for correct binocular stereopsis. The eye distance can vary from 53mm to 73mm with an average of 63mm. Many HMDs do not provide a way to adjust for intraocular distance making stereo viewing problematic for some users. Additionally, because the images the user sees are always in focus and have the same focal depth, accommodation and vergence cue conflicts can occur when users look at objects with different virtual depths causing eye strain and discomfort [15]. This phenomenon also occurs with projection based displays and monitors but is more pronounced with HMDs since the screens are close to the user's eyes.

In our second evaluation there is a tendency noticeable between hours per week spent in front of computers and occurrence of negative side effects. The only negative side effects that occurred in both evaluations were headache, eye strain and one user with a migraine type of heavy after effects. In our evaluation data it is noticeable that many persons who work a lot with computers and also use it in class are less likely to get headache and eye strain. Due to the small sample of users we cannot make any justified statement but speculate that there actually is a correlation in our AR environment. Our hypothesis is that users whose eyes are used to computer screens accommodate more easily to the fixed focal depth of HMDs and are less likely to get accommodation conflicts. It is at least worth studying this aspect in detail in future evaluations to make a justified claim about this assumption.

We cannot do much to overcome accommodation and vergence cue conflicts since better and more flexible HMDs would be needed. Omura et al. [113] developed a system to alleviate these problems. They incorporated movable relay lenses into a HMD which are continuously adjusted using gaze direction. Another promising emerging technology are virtual retinal displays (VRD). The VRD, also called Light-Scanning Display, was invented at the Human Interface Technology Lab in 1991 [156]. It is based on the idea that images can be directly projected onto the retina. However, these types of display systems are still in the early stages of development.

Some studies [59, 147] suggest that users can adapt to the virtual environment after having used the system multiple times.

LaViola [87] summarizes “Cybersickness can present a significant problem for a number of individuals who use virtual environments both during and after the VE experience. Although the technological causes may pass with time, those causes based on individuality probably will not. Nevertheless, it is important to understand what the causes for cybersickness are so we can find way to reduce and possibly eliminate it.

Although the current cybersickness theories have flaws, they have been able to help determine the causes for cybersickness in some cases. They also have helped researchers develop some methods with which to reduce cybersickness and its associated symptoms. These cybersickness reduction methods have helped in some cases but not all of them. If a unified and complete theory which can determine the causes of cybersickness on an individual basis and provide the necessary predictive power is found, then perhaps cybersickness could be eliminated completely. Otherwise, just like motion sickness, cybersickness will be with us indefinitely.”

6.3 Exact versus Dynamic Construction

One of the fundamental changes in the development of Construct3D was based on the insight that exact construction by coordinates is difficult to accomplish directly in 3D space with 6 degrees of freedom because of various reasons. On the one hand our magnetic tracker caused inaccuracies from 3 millimeters to 2 centimeters in average in a working volume of 3x3x3 meters. On the other hand human hand-eye coordination is not sufficiently accurate. It is very difficult to spot a coordinate location exactly in 3D space (for example to set a point), to keep ones hand still and press a button. Most people’s hand trembles a bit when trying to find an accurate spot with a pen in 3D space. In addition users try to compensate tracker inaccuracies. As reported by our subjects in the first evaluation (section 8.1) this caused problems and inaccuracies of 5 mm to 1 cm in average.

For exact construction in virtual environments a number of user interface adaptations must be undertaken in order to support users when working directly in 3D space. We did not implement these but will present a few basic ideas. Many principles from traditional desktop CAD packages can be reused in 3D.

Bowman [17] and other studies suggest 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. As an example points can be set by utilizing normal projections such as top, front or side views. In a first step x and y coordinates could be chosen in a top view with the help of an adjustable grid. The missing z-coordinate could be added in a second step. In addition of adjustable grids, snapping functionality should be provided to snap points to objects. This is also useful for selection of specific surface features i.e., an excellent snapping functionality of Rhino3D[®] (www.rhino3d.com) provides

a wide range of snapping features. Not only vertices can be snapped but also mid and center points of lines and faces, intersection points, tangent points and lines, perpendicular objects, control points of spline curves and surfaces and many more.

In order to change the drawing plane and coordinate system CAD programs such as MicroStation (www.bentley.com) provide functions to set a user specific coordinate system (called UCS in AutoCAD® for instance). A very advanced user specific coordinate system functionally called AccuDraw® is implemented in MicroStation®. AccuDraw technology dramatically accelerates the design process by allowing to switch easily between linear and angular types of input. Drawing plane and coordinate system can dynamically be change. AccuSnap is a natural complement to AccuDraw. It streamlines the selection of geometric keypoints and design information.

A review of user interfaces in common CAD applications would be extremely interesting but is not of primary interest in our context. We refer to relevant papers as mentioned in section 2.2

We in contrast chose to take a different route than construction and modeling by coordinates in 3D. Based on experiences by teachers with educational dynamic geometry applications and parametric CAD we decided to take a route that was more beneficial to our educational intentions. 2D dynamic geometry applications (see a literature review of existing systems in section 2.4) are emerging in schools since the beginning of the 90's and provide an excellent way to explain geometric principles and ideas. No dynamic geometry application for educational purposes that supports three-dimensional construction existed at the time we finished our first evaluation. In our opinion Augmented Reality is a perfect interface environment for such an application and we decided to change Construct3D into a dynamic construction tool.

A fundamental property of dynamic geometry software is that you can explore *dynamic behavior* of a construction by moving it. You can see what parts of the construction change and which remain the same. You get by far more insight into this particular construction and geometry in general if you can experience what happens under movements. Since such a learning environment encourages experimentation it complies much better to all pedagogic theories (mentioned in section 2.5) than traditional 3D modelers. At the same time exact construction by coordinates loses its importance. It is still important to have powerful snapping functions to construct objects in correct relations to each other but coordinates as indications of position in space are of very low importance in a dynamic construction environment.

6.4 User Interface

The Personal Interaction Panel (PIP)

We chose to use the Personal Interaction Panel [152], a two-handed 3D interface composed of a position tracked pen and pad to control the application. It 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 allows the props to be used as multi-functional tools (Figure 37). Every application displays its own interface in the form of one or multiple PIP “sheets” which appear on the PIP. The pen and pad are our primary interaction devices.

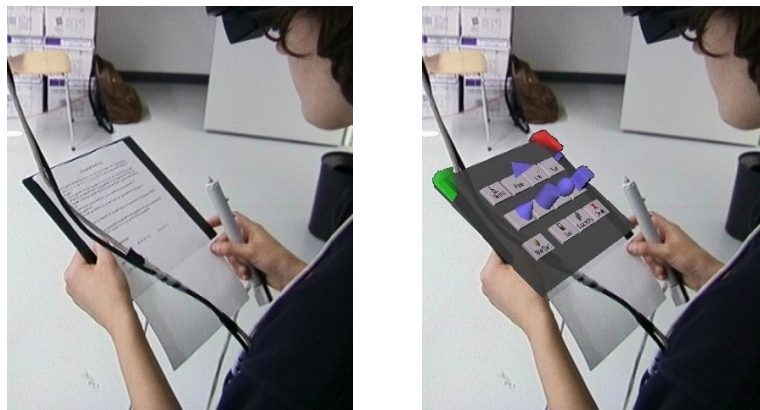


Figure 37: Working with the PIP. The pad is useful in multiple ways. By looking out from underneath the head mounted display at the menu panel, instructions on a sheet of paper can be read (left), by looking through the HMD the menu system with widgets can be seen (right).

Our source of inspiration for designing a user interface for this HMD-based application is based on various ideas, problems and suggestions from such diverse areas as user interfaces, user centered design, usability engineering, human computer interaction in general [7, 110, 116, 152] as well as current software used for 3D modeling.

From the beginning our intention was to keep the user interface very simple and intuitive to use. We started with a basic PIP sheet as can be seen in Figure 38. Large, textured 3D buttons are used with meaningful 3D icons floating above the buttons to allow easy and fast selection of menu items. Because of inaccuracies of the magnetic tracking system buttons had to be bigger at that time. For instance pressing buttons of a size of 1x1 cm is hard or impossible if the pen jitters because of inaccurate tracking data. With our optical tracking system and application of filters to compensate for jitter we solved this problem. However we are still using larger buttons which are quick to identify by the user and easy to select.

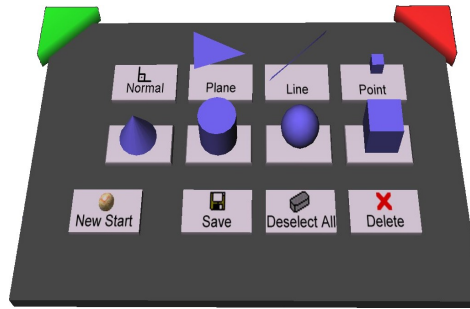


Figure 38: The original PIP sheet from our first evaluation in June 2002.

Menu selection is achieved by moving the pen into the appropriate widget until it is highlighted (it turns yellow) and by clicking on the button of the pen. The menu button turns red when clicked and moves down – a selection technique known to users from 2D window interfaces.

Later we adapted our menu system to reflect the enhancements of better widget support and a PIP sheet management on the Studierstube side. Multiple PIP sheets are supported which allow switching through multiple menus. In the upper left part of the PIP in Figure 39 five widgets for the selection of sub-menus can be seen. The user can switch through menus by clicking on the sub-menu buttons “Main”, “Construct”, “Measure”, “Properties” and “System”. On these different PIP sheets, widgets are placed for the execution of all features of Construct3D.

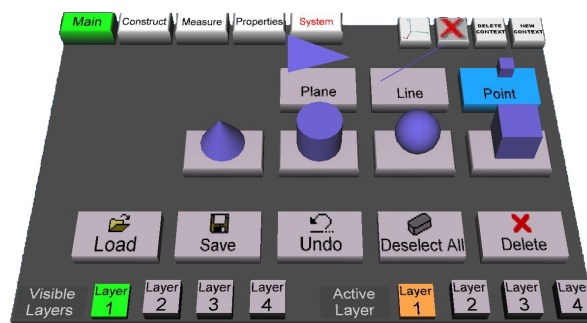


Figure 39: Multiple PIP sheets represent the menu system of Construct3D.

We are currently in a phase of redesigning the menu structure of Construct3D and its appearance. More flexible widget types are being implemented that support more flexible layout schemes. We will restructure widgets into more sub-menus to group features that operate on 2D or 3D objects. There will also be a group of important features available on all sheets at all times. In addition widgets will automatically be disabled if not applicable to current input elements. Teachers will also be able to disable features which are not suitable for specific learning tasks.

Redesign of the Pen

The second part of the PIP user interface, the tracked pen, had three buttons in its original design (upper left picture of Figure 40). A front – also referred as the primary – button, a secondary button a few centimeters behind the primary one and

a button at the tip. Our first evaluation (section 8.1) as well as various informal user tests revealed that users had difficulties in remembering and finding the correct button for specific tasks since they did not see the buttons very well through the HMD and could not feel them well enough. Since the 3-button solution caused problems and did not improve working speed we changed the design to a one-button pen.

Transmitting button clicks to the tracker server was another problem. While using wired magnetic sensors we had an additional wire attached to the pen to transmit the button signal. With the introduction of the optical tracking system we wanted to get rid of the additional cable and started work on a wireless pen. The final solution (bottom right image in Figure 40) of our wireless pen is professionally designed and manufactured. The wireless sender can send in three different frequencies which is the maximum numbers of pens we can use in one room. At its end markers are mounted for optical tracking. The end can be exchanged to mount markers for other tracking solutions i.e. ARToolkit markers. Its batteries can easily be exchanged and an LED at the bottom of the pen indicates button presses.

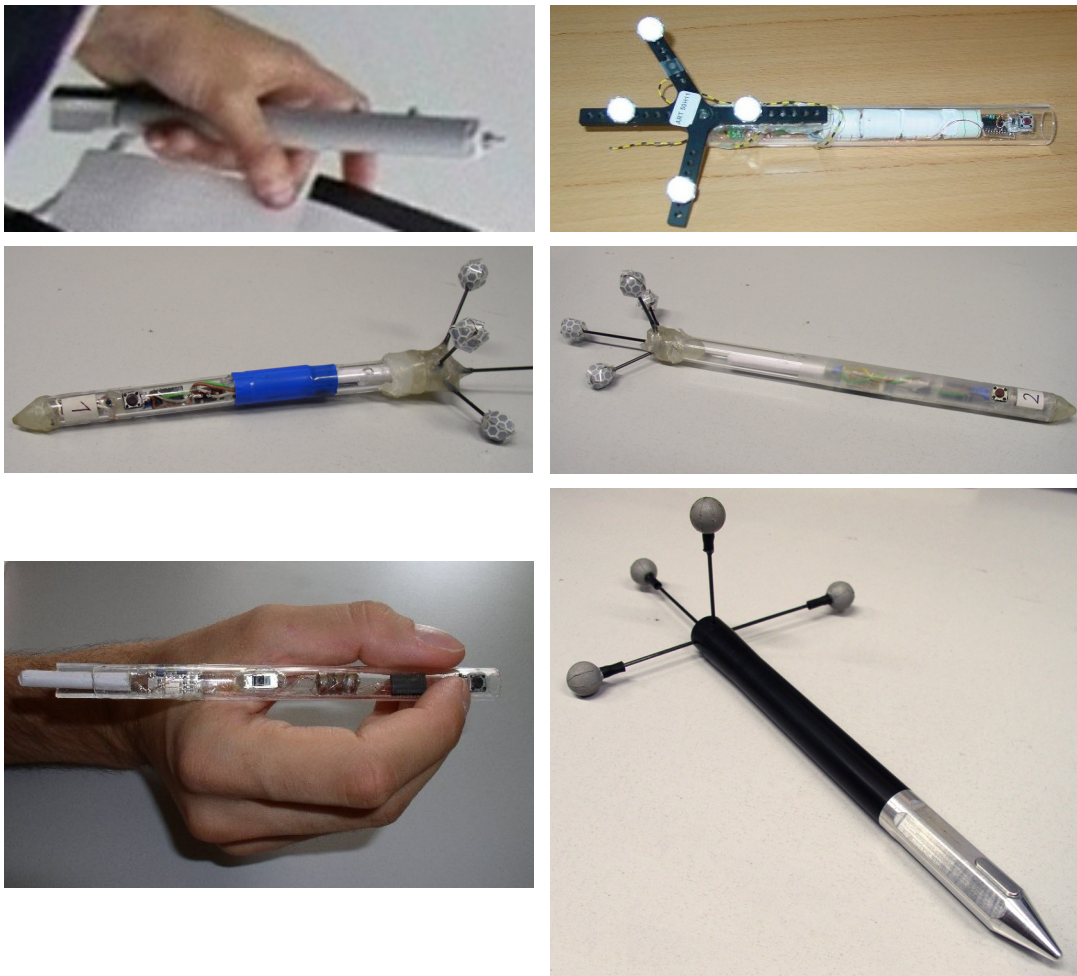


Figure 40: From the original (top left) to the final design (bottom right) of our wireless pen. Various intermediate steps show the development process.

The functionality of our pen is very similar to the 3-button pen called Cyberstilo [50] introduced by Graf et al. However, our pen with a length of 17cm and a diameter of 13mm is much smaller than the Cyberstilo and light weight. It is not rechargeable as the Cyberstilo but very well balanced in user's hands. In the second evaluation (section 8.3) it proved to be very user friendly and easy to handle by students.

The virtual model was also adapted to the changes. The old model of the virtual pen (unchanged since 1996) is shown in Figure 41 on the left whereas the new model of the pen is displayed on the right. The new model has a higher number of polygons than the old one but due to considerable improvements in rendering speed within the last years, this is acceptable given the visual quality improvement.

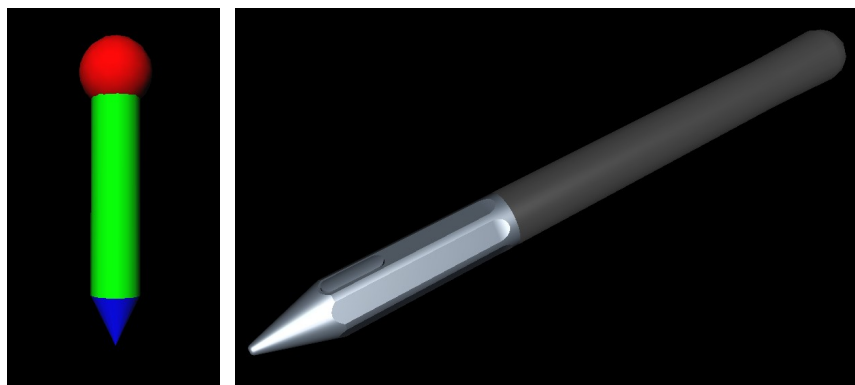


Figure 41: Left: Previous pen model. Right: The virtual model of the pen after its redesign.

6.5 Usability Design

In sections 6.1 and 6.4 we mainly describe hardware improvements that reflect on usability. In this section we summarize improvements of rendering quality and of quality in the visualization of geometric constructions. Visual design as we use it (described in “color coding”) encapsulates and conveys additional information to the user which would be difficult to present in textual form or via audio. We use this kind of information visualization to structure a construction, to explain construction steps that others have been doing and to improve the user's understanding of the construction. All usability improvements presented in this section are computationally expensive and cost more performance than most other algorithms in our application.

Transparency

Technical drawings, blue-prints and geometric constructions on paper in general all conform to certain stylistic requirements. Important aspects of constructions are visually enhanced, unimportant parts are de-emphasized but all parts of a building or of an engine are shown, even hidden parts. In modern technical drawings as well

as in computer generated images transparency is frequently used to show hidden parts such as parts inside an engine.

The importance of transparency in technical illustrations is documented by Diepstraten et al. [35]. They note “A major advantage of technical illustrations is that they provide a selective view on important details while extraneous details can be omitted. Technical illustrations are better suited to communicate the shape and structure of complex objects and they provide an improved feeling for depth, occlusion, and spatial relationships.” Further on they state that “It is quite remarkable that transparency is widely neglected in computer-based illustrations because books on traditional manual illustrations do provide effective techniques and rules for handling transparency in order to communicate the location of occluding and occluded objects.”

We are using transparency for geometric primitives since the beginning to enable users to see inside other objects. Direct manipulation of points inside other objects is only possible if these points can be seen.

In the first version of Construct3D we implemented a slider to give users the option to modify the transparency of objects themselves. This was not satisfactory since after a number of transparency changes many objects had different transparencies which caused confusion. It did not present a consistent look to learners. In order to get a consistent learning environment a professional graphics designer helped to design fixed transparency values for all objects and color schemes in general as described in the next section.

Correct transparent rendering is not trivial and computational expensive. Figure 42 (left) shows the best transparency mode of SGI Open Inventor that we used initially. Certain geometric primitives produced visually bad artifacts when rendered transparent. Figure 42 (right) demonstrates the best transparency mode of Coin combined with our color scheme and 6 lights in the scene that give visually nice spotlights. The midpoint of the sphere is clearly visible in both cases.

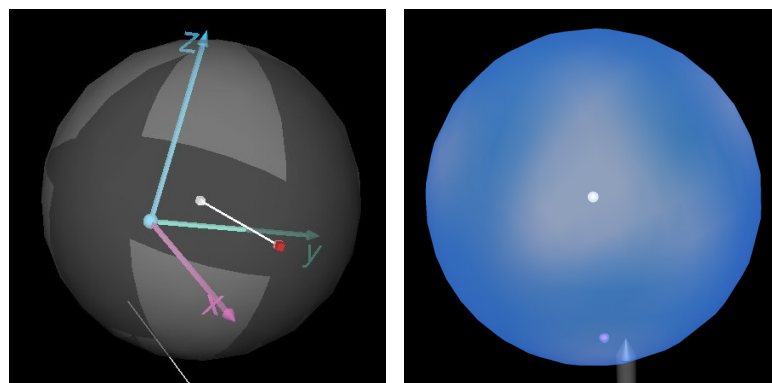


Figure 42: Left: A transparent sphere with rendering artefacts. Right: Current rendering of a transparent sphere in a fully correct computational expensive rendering mode.

One disadvantage of using transparencies is that shading or color differences are hard to see on objects. Models that are too transparent appear as blobs and in complex models it is difficult to see all edges. Therefore a useful transparent value must be found that allows seeing through multiple layers of nested objects and still enables the user to see the shape of the model.

Because of the above mentioned reasons we do not use transparency for complex objects such as Boolean objects, NURBS surfaces or sweep surfaces. In case of these objects which are drawn opaque users have the option to switch them individually to wireframe mode. It allows to see inside or behind these objects. Points for instance which are inside other objects must be accessible by the user at all times in order to be able to modify them. Figure 43 shows a rotational sweep surface in normal and wireframe mode.

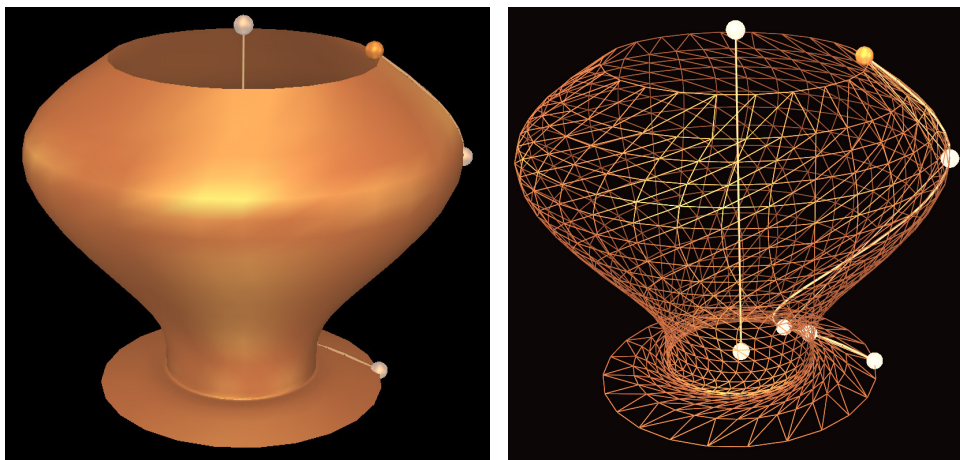


Figure 43: Left: The lower point of the axis and points on the B-Spline curve are hidden behind and inside the surface of revolution. Right: In wireframe mode all points are visible and easily accessible.

The Open Inventor implementation Coin supports in its latest version a fully hardware accelerated transparency mode based on OpenGL 1.5 extensions (fragment programs) by doing “depth peeling”. It provides correct rendering of transparencies.

Color Coding

A professional graphics designer developed a color scheme for Construct3D in order to structure geometric content. Two aspects have been considered as most important by us in an educational context.

For students, teachers and spectators it must be possible to distinguish between the work done by each single user. Therefore user information is encoded in colors. This means that each user is working within a separate color space. The current color scheme supports the choice of 4 different color spaces – a blue, orange, green and red one (see each line of Figure 44 for the different color spaces). In order to

give students more freedom of choice, each user is able to select the color scheme he prefers to work with.

In addition to encoding user information in the color scheme it was also considered important to have information about active and inactive layers visually present at all times. As mentioned in the description of our layer concept in section 4.1 only one layer can be active at a time but multiple layers can be visible. New objects are always drawn into the active layer. Inactive layers can show objects of previous stages of the construction. This structuring is important to give priorities to parts of a construction and to guide students through complicated steps. In traditional education teachers use colors and different drawing styles (e.g. dashed or dotted lines) to visually enhance complicated constructions and to structure construction steps.

In Figure 44 (next page) we show objects in active and inactive layers. In the lower left corner of each individual image inactive objects are displayed. They are “grayed out” and darker than “active colors”. In the upper right region active objects can be seen. All screenshots in the left column of Figure 44 show a comparison between *deselected* and inactive objects, the right column compares *selected* and inactive layers of the blue, orange, green and red color spaces. Active and inactive objects are clearly distinguishable as well as selected and deselected objects which was the main goal of this design.

Implementing this scheme proved to be troublesome. Since each geometric primitive is internally rendered in a slightly different way, we had to assign each primitive a different material so that they all look the same. In total we had to design more than 140 different materials for our objects in order to generate a unique look and feel. 6 lights were added to the scene to produce equal lightning conditions in the virtual environment independent of the user’s position. Texture-based lightmaps were applied to planes and cubes to make them appear like being lit. We designed these colors specifically for the virtual environment when viewing a scene with head mounted displays. They look different on monitors and appear less bright when viewed with see-through HMDs than on paper printouts.

Improving User Interaction: Highlighting and Preview

Highlighting is used as a method to indicate if a user’s pen is nearest to an object. If an object is highlighted the user knows that he can select it. With the aid of a professional graphics designer we performed extensive experiments on how to highlight objects and developed an efficient method. We tried using an additional color for highlighting but it proved to be not distinguishable from other colors in the same color scheme any more, no matter how we chose it. Especially when wearing see-through HMDs colors are not as bright as on monitors and are more difficult to distinguish.

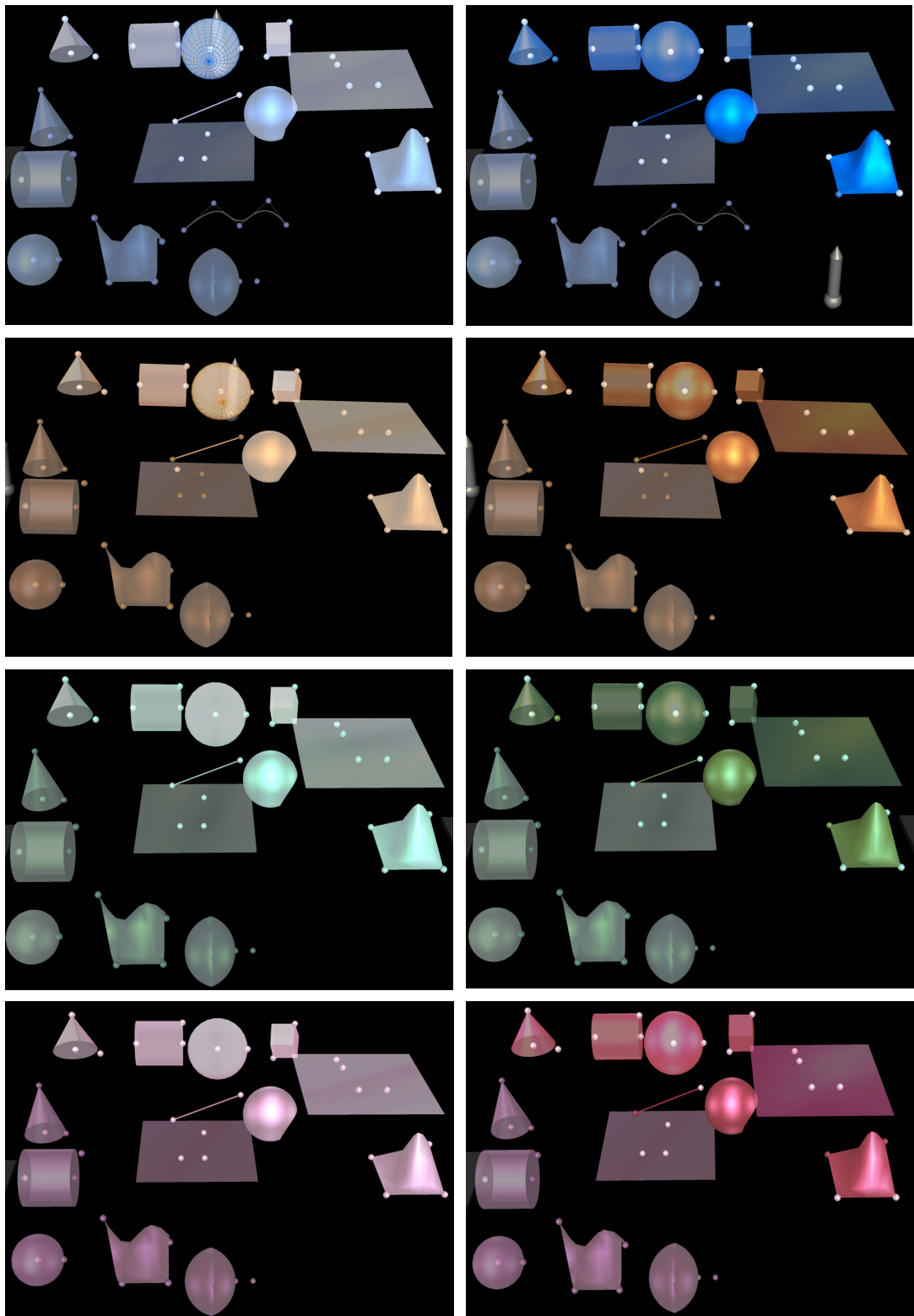


Figure 44: Construct3D color scheme.

Finally we chose a form of highlighting where we use a wireframe grid of the same model that we superimpose on it. This “highlighting grid” gives the impression of capturing and catching an object with a web which fits to the idea of selecting. Figure 45 (middle) shows how a highlighted object looks like.

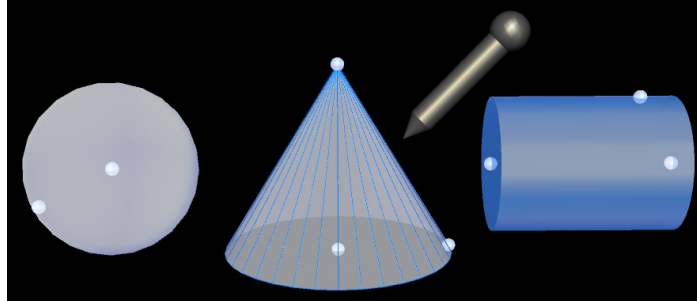


Figure 45: Highlighting the nearest object to the pen. From left to right: A deselected sphere, a highlighted cone and a selected cylinder.

Various internal tests of this selection method showed that it is very convenient and intuitive to use. During our whole second evaluation there were no difficulties with it. Users constantly see which objects are nearest - indicated by highlighting - and they can differ between selected and deselected objects.

Points which are very small objects are also highlighted by superimposing a wireframe grid on them. In addition points can be dragged. If a users gets very close to a point – within a “dragging area” of 5 centimeters in diameter – the point changes its color to a blazing blue, orange, red or green (depending on the color scheme used) indicating that it can be dragged. Figure 46 shows all possible states of points in all four color schemes. All these colors can easily be distinguished from short and large distances.

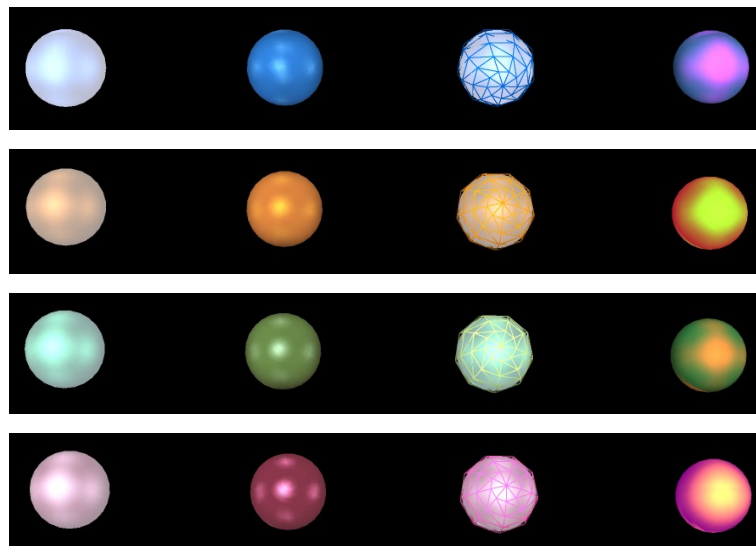


Figure 46: Points in all four color schemes – blue, orange, green and red. From left to right: Deselected point, selected point, highlighted point and drag-enabled point

The preview feature enables a user to see a preview of an object before he actually generates it. While the user moves his pen over a widget on the PIP a preview of the object is shown. Figure 47 demonstrates how this works. This gives visual feedback if an operation works with the given input elements and if it produces the desired result.

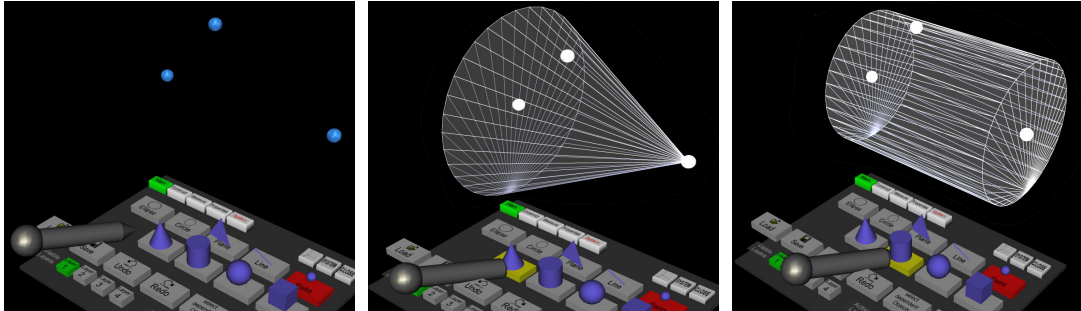


Figure 47: Preview feature. Left: 3 points are selected. Middle: Preview of a cone through the given 3 points. Right: Preview of a cylinder.

We show a preview for all possible operations, including intersections and Boolean operations, no matter how complex the resulting model is. The preview feature was not used by students in our second evaluation as we had hoped (see the discussion part of the second evaluation - section 8.3, page 127).

As a last method to aid object manipulation we implemented a gesture for deselecting all selected objects. During constructions it often happens that too many objects are selected or different objects need to be selected for the next operation. Instead of deselecting each object separately, we implemented a “deselect all” method. In addition to clicking a button on the menu this feature can be triggered by a gesture. Point mode must be turned off in order to use it. We tell users to activate this gesture by pointing the pen down and clicking the pen’s button.

Not a single one but the combination of all these usability improvements has big impact on the general look and feel when working with Construct3D.

7 Content Development

In order to demonstrate Construct3D's potential in dynamic 3D geometry, we constructed examples ranging in difficulty from basic high school to basic university geometry education. The first part of section 7.1 consists of five examples that were used in our second evaluation as content of teaching. In the second part additional examples are depicted that exploit dynamic 3D geometry and are hardly possible to teach similarly with existing CAD software.

Our experiences in teaching with Construct3D in six subsequent lessons during our evaluation and the consequences on content development are described in section 7.2.

To support different learning styles of students, we discuss various learning modes in section 7.3 ranging from autodidactic to teacher-guided learning. As a first implementation supporting some of these modes a Construct3D tutorial is described in section 7.4. It is based on the presentation authoring language APRIL.

7.1 Examples for Dynamic 3D Geometry

Content for the Second Evaluation

Five examples presented at the beginning of this section were generated for our second evaluation (section 8.3). The geometric content within these learning units is very diverse regarding its place in the curriculum. The primary goal for this diversity is to evaluate how different topics are taught by teachers in this new learning environment. We want to find out which content “fits best” to the environment and which pedagogic methods may be used. We aim to find guidelines for content or geometric principles which can best be taught by using Construct3D. It is equally interesting to investigate which content does not benefit much by this new medium and can be taught with traditional CAD programs and which content is even better taught by using paper and pencil sketches or drawings.

After presenting our examples we summarize our experiences regarding these aspects in section 7.2.

Note that since there is no specific topic from the geometry curriculum that is being taught throughout all 6 learning units, no significant learning progress on one specific topic can be expected. Our lessons address the following learning goals:

- Learning about Boolean operations (Tschupik-cubes)
- Learning about surfaces of revolution and their geometric properties
- Learning about intersection curves of surfaces of 2nd order (intersection curve of two cylinders). Learning methods how to construct tangents in points of intersection curves.
- Vector algebra: Is there more than one center of gravity (in a tetrahedron)?

Tschupik-cubes (taught within the first two units) are integrated into the geometry curriculum of grade 7 and 11 in Austria. Surfaces of revolution are taught in grade 12, intersection curves of cylinders mainly in grade 12 too. The last example with the tetrahedron fits into mathematics and geometry curriculum of grade 10 to 12.

Tschupik-Cubes

The Austrian professor Josef Tschupik taught geometry to generations of students at the University of Innsbruck. He invented so called Tschupik cubes (Figure 48) as a means to train spatial abilities. Today they are used worldwide to learn interpreting top, front and side views of objects, in spatial ability tests and other intelligence tests. Tschupik-cubes are “sculptures” which are cut out of a cube by using planar sections only. Traditionally they are used in two ways: Either top, front and side view of a Tschupid-cube are given and the student has to draw an axonometric view of the object or the axonometric view is given and top, front and side view must be drawn.

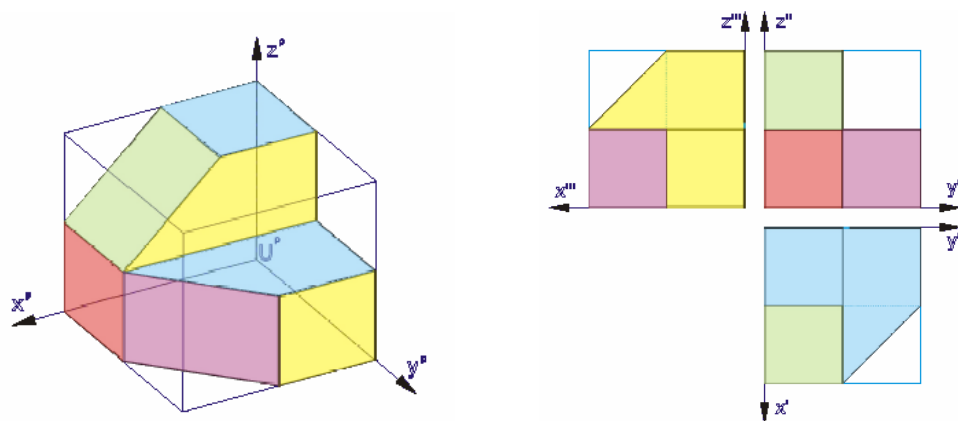


Figure 48: An example of a Tschupik-cube. Left: An axonometric view. Right: Top, front and right side view.

For our students the axonometric view of the Tschupik cube from Figure 48 was given. Their task was to model it with Construct3D in the virtual environment. A wireframe cube (as a reference frame) was also given as can be seen in Figure 48. They also had the choice to model one of the cubes given in Figure 49.

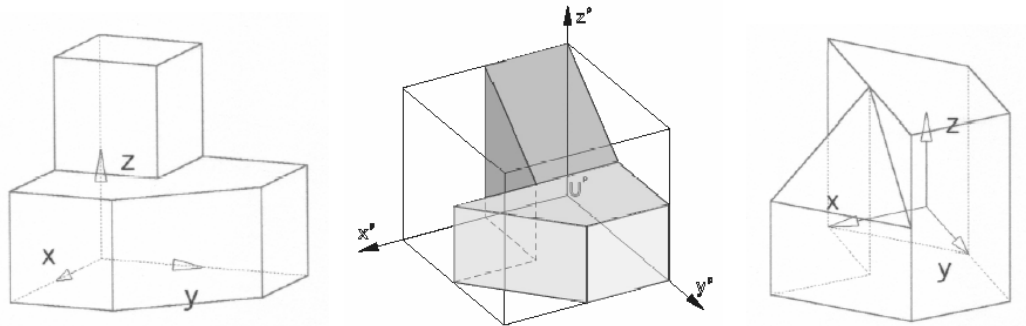


Figure 49: A choice of other Tschupik-cubes

We will briefly describe the activity during the first lesson to give insight how teaching and learning with Construct3D looks like. At the beginning of the lesson the teacher gives the assignment and assists students through their first steps with Construct3D. After a basic 5 minute introduction, two students start collaborating. They get 30 minutes to finish one Tschupik-cube but there is no requirement to finish the task (though nearly all students finished earlier). The teacher explains the interface and helps with geometric questions. The learners have to cut certain parts out of a cube to get the desired object. Only four functions are needed to complete the task. “Midpoint” to draw midpoints of edges of the cube, the “cube” function to draw smaller cuboids inside the given wireframe cube by using midpoints, “slice” to cut the cuboids and Boolean operations (union, difference and cut) to unite parts or cut cuboids.

Figure 50 shows the work of our students in their first lesson with Construct3D.

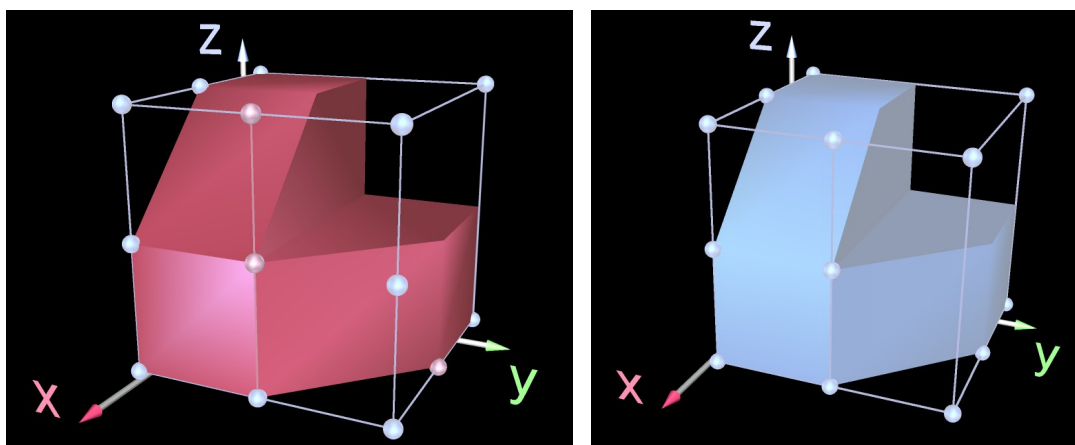


Figure 50: Tschupik cubes generated by students with Construct3D.

This double session introduced students to the menu system and basic functions of our application. It is the only example presented in this thesis that is not highly

dynamic. All constructions were purely static, no objects could be dynamically changed. We designed this example in cooperation with teachers with the purpose to give a simple introduction to the basic functionality and not to overload students by too many features at their first trial. However in future evaluations we plan to allow more freedom for experimentation from the beginning to better encourage creativity and learning by errors. It is important to understand from the beginning that Construct3D is not a static modeler and offers different functionality.

Surface of Revolution

In the second lesson we specifically use the special dynamic features of Construct3D. Regarding high-school geometry this is a “high-end” example that is not taught in traditional geometry education. This example can not be done with any other CAD program in a similar way.

Given is an axis. A surface of revolution must be constructed by rotating a B-Spline curve (cubic, 5-6 control points) around the axis. The control points can be dynamically modified at any time resulting in a change of the surface of revolution.

As a next task students have to construct the tangential plane in a point of the surface. Therefore they have to construct a meridian curve through the point which they get by intersecting the surface with a plane through the axis. They also have to rotate the point around the axis to get its “circle of latitude” on the surface of revolution. The tangential plane is defined by the two tangents to the circle of latitude and to the meridian curve.

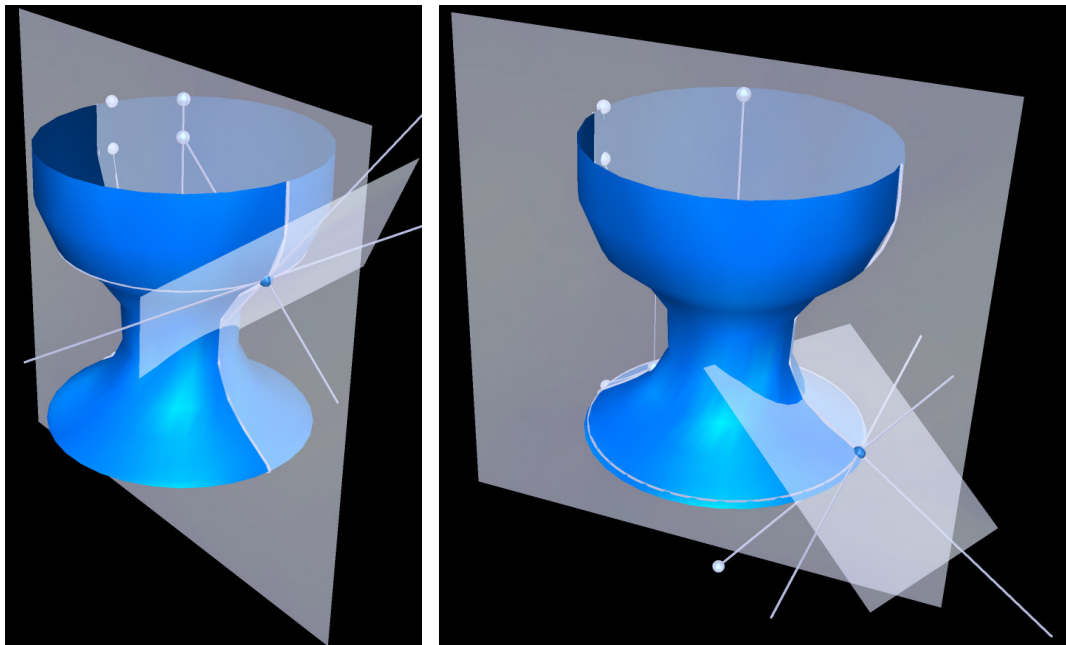


Figure 51: Left: Tangential plane in a point on a surface of revolution. The surface normal is intersected with the axis. Right: We are moving the point on the surface to see how the tangential plane changes.

As a last step students are asked to test if the surface normal (normal to the tangential plane) intersects the axis of rotation. This is true in all points of a surface of revolution. The result is shown in Figure 51.

This example is very dynamic by allowing the students to explore the surface in every aspect. They can change the surface at any point in time by changing points of the rotated B-Spline curve. They can move the point on the surface which results in a change of the meridian curve, the circle of latitude and the tangential plane in this point. In general the students can explore properties of the surface by themselves. In our second evaluation (section 8.3) half of the students did not learn about surfaces of revolution in geometry classes before. They were very impressed by this example. At the end some of them mentioned in the questionnaires that they actually learned new content in this lesson.

Intersection Curve of Two Cylinders

The third example reflects traditional geometry education. In order to construct an intersection curve of two surfaces we must find common points on both surfaces. Tangents in intersection points are usually constructed to approximate the curve in a better way and to learn about special points on the curve (e.g. inflection points). Though Construct3D has a function to draw intersection curves automatically, the goal of this session is to learn about a geometric *method* how to find common points of two cylinders. We will not use the built-in Construct3D functionality to calculate the intersection curve. Instead the “helper planes”-method is taught which uses planes to intersect both cylinders. If a plane is chosen parallel to the axes of the cylinders, the planar sections of each cylinder are generators. With traditional construction methods and also with CAD programs this is an optimal configuration and gives exact results. This method is not only applicable to the case of cylinder/cylinder intersections but can also be used in other cases of 2nd order surface intersections (e.g. cone/cylinder, cone/cone). Using planar sections is in general applicable when looking for common points between surfaces.

Two cylinders with axes are given. Students have to generate a plane which lies parallel to both axes of the cylinders. It must be intersected with both cylinders. Two generators on each cylinder are the resulting “intersection curves”. These must be intersected again to get points of the intersection curve (Figure 52). In general there are 4 such intersection points in a plane since both generators of one cylinder intersect both generators of the other cylinder. In the right image of Figure 52 all 4 intersection points can clearly be seen. By moving the plane up and down students will see that all points of the intersection curve can be constructed that way.

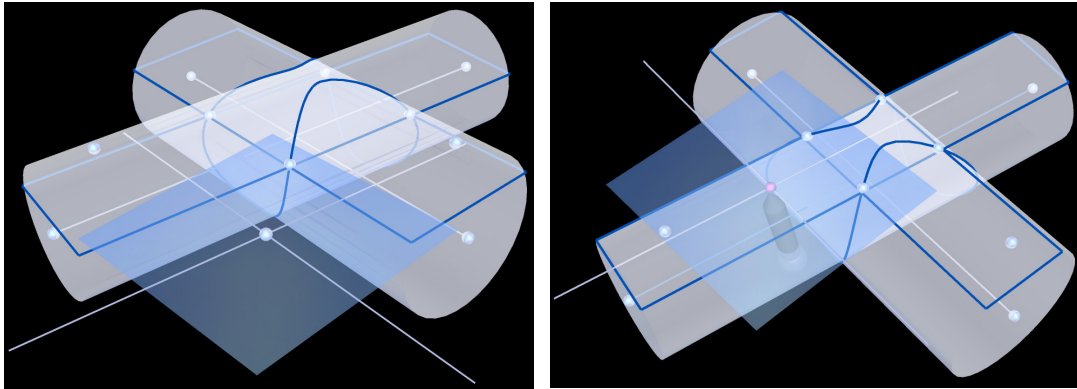


Figure 52: Left: A plane is intersected with both cylinders and the resulting generators are intersected again. Right: The plane is moved up and down. The four intersection points on the intersection curve that lie in the plane are clearly visible.

At this point students already know Construct3D and the interface. They need the teacher for guidance and helping them with geometric problems. It was interesting to see in our evaluation (section 8.3) that the students with traditional geometry education apparently knew about the “helper-planes”-method by heart, applied it and reached the result very soon. They were only a bit confused that the axes of the given cylinders were not parallel to the x/y-plane - such a special configuration is easier for constructions with traditional methods and they are used to that. Another group who mainly works with CAD programs used an experimental approach and chose a general plane. This also worked but with guidance by their teacher they finally also constructed the solution with a plane parallel to the axes. Both groups moved their plane up and down to see if all intersection points can be reached.

Tangents in Points of Intersection Curves

In this lesson students will practice the construction of a tangent in a point of the intersection curve. The goal is to learn that the tangent to an intersection curve of two surfaces is always the intersection of the tangential planes to each surface in a point of the curve.

At the beginning of the session we load the example from last lesson and continue working on it. We summarize what was learned in the last lesson. A tangent to the intersection curve in a specific point on the curve can be constructed by constructing a tangential plane to one cylinder and another tangential plane to the other cylinder in this point. Construct3D’s tangential plane function can be used. The intersection of these two tangential planes is the desired tangent (see Figure 53).

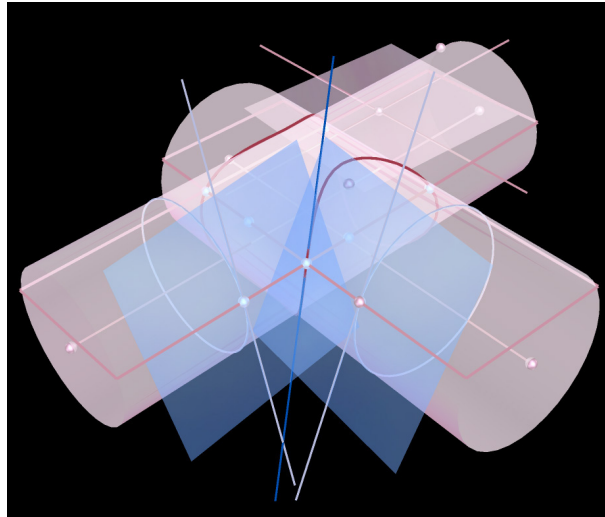


Figure 53: Tangent (dark blue line) in an intersection point as constructed by two students.

In addition we prepared a cone and a cylinder to train the learned methods again.

Centers of Gravity

In this lesson we focus on “the” center of gravity of 3D objects.

A general tetrahedron is given. Students have to construct its center of gravity. They probably start to construct the center of gravity of each triangle and connect those centers with opposite vertices. These lines intersect in one point. By dynamically changing the vertices of the tetrahedron it is easy to visually check that there is always an intersection point which can be called the center of gravity (Figure 54 left). As a next step the teacher suggests to construct midpoints of edges and connect midpoints of “opposite” edges. It turns out that these lines intersect in a point too (Figure 54 middle). Using dynamic geometry of Construct3D, students can change the tetrahedron to find out if and under which circumstances these lines intersect in one point. They always intersect.

A discussion starts and in the end the teacher explains that there is more than one center of gravity. There are 4 centers of gravity in 3D objects. If only the vertices of a tetrahedron have a weight and everything else is weightless, then the first intersection point that the students constructed is the center of gravity of the vertices. If we materialize the edges only the second point is the center of gravity of the edges. If only the faces of a tetrahedron have a weight, then we get a center of gravity of the faces (which is more difficult to construct) and if the whole tetrahedron is a solid object with weight we get the center of gravity of the whole volume (which is also more difficult to construct). In case of a general tetrahedron the center of gravity of vertices coincides with the center of gravity of edges (Figure 54 right).

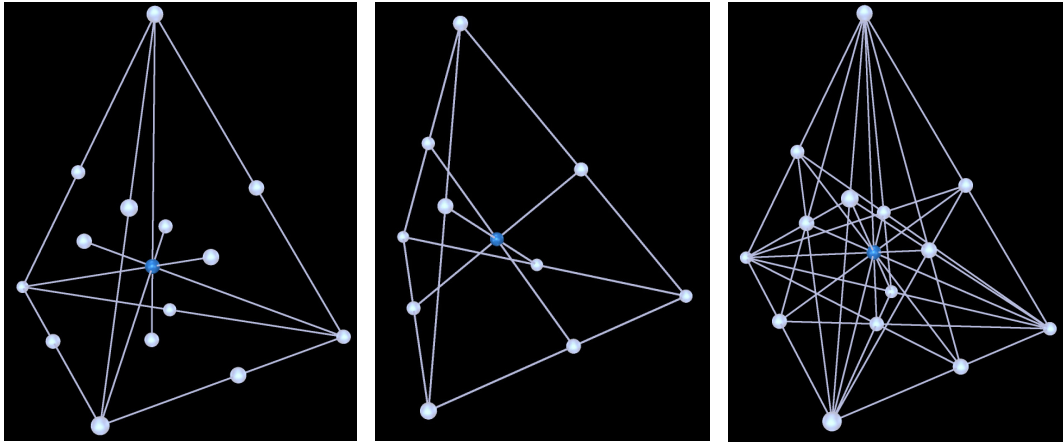


Figure 54: Left: Center of gravity if the vertices have a weight only (we omit all lines for the construction of centers of gravity for individual triangles). Middle: Center of gravity if edges are materialized. Right: All lines shown for the construction of both centers of gravity. They coincide in one point (colored blue).

Note that this last example used for the second evaluation uses 3-dimensional dynamic geometry and cannot be constructed with current CAD packages and dynamic 2D programs. We consider it a standard example for dynamic 3D geometry, like constructing the circumference of a triangle is a standard example for dynamic 2D geometry.

Content for Advanced Geometry Education

All coming examples are designed in a way to utilize dynamic geometry. Because of a lack of educational dynamic 3D geometry software with similar capabilities as Construct3D most of the examples cannot be constructed with other existing software so far.

We are aware that there is a very large number of interesting examples that can already be done with Construct3D. Our selection is purely random and just gives a glimpse of what is possible. Every educator reading this chapter can probably imagine many other examples that fit in here. We are right at the beginning of exploring the possibilities that dynamic 3D geometry and a tool like Construct3D offer to us. The examples in this section are not meant as a full coverage of the topic of dynamic 3D geometry but rather as a starting point for in-depth educational research.

Deflection Sheave

A deflection sheave or haul rope sheave (in German “Umlenkrolle”) is a terminal sheave that deflects a haul rope. It is used for example in ski tows or ski lifts.

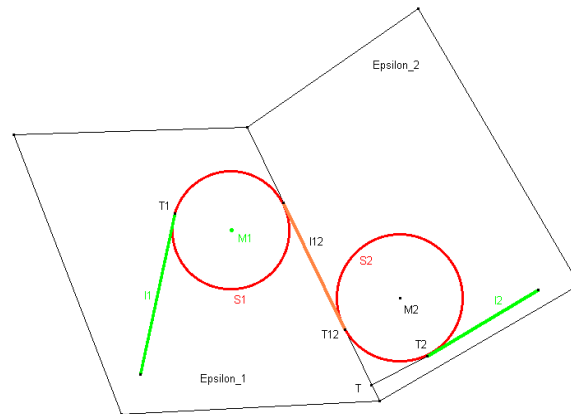


Figure 55: Explanatory draft of the problem

In this example a haul rope is being redirected from position l_1 by two deflection sheaves to position l_2 . The lines l_1 and l_2 and a point M_1 are given (drawn green in Figure 55). The two deflection sheaves are realised as circles. The midpoint of the first sheave is given as M_1 . The task is to construct both deflection sheaves and the touching points of the cables. The sheaves can have different radius. Our sketch (Figure 55) is similar to what teachers in traditional geometry education use to explain this example.

The benefit of doing this example in dynamic geometry is obvious. Learners can modify all given elements – start and end position of the rope (l_1 and l_2), the radius of both ropes can be varied and different configurations and special cases of this problem can be studied.

The solution to the problem is easy, once it is understood and a sketch is drawn. l_1 and M_1 define a plane. In order to find the circle - the first deflection sheave - with mid-point M_1 that touches l_1 we need a point on the circle. Therefore we construct the touching point T_1 of the circle with l_1 . By intersecting a normal plane through M_1 with l_1 we get T_1 . We draw the circle through T_1 . The full construction including all steps is displayed in Figure 56.

Next we need the plane that contains the second circle. We see in Figure 55 that in order to redirect the rope from one sheave to the next the rope touches both sheaves. We need a common tangent (draw in orange) to both sheaves. We intersect l_2 with the first plane to get T . From T we draw the tangent to the first sheave and get the orange line l_{12} . The second sheave touches both lines l_{12} and l_2 . Therefore its mid point M_2 has the same distance to l_{12} and l_2 . It lies on the angle bisector of l_{12} and l_2 . In order to draw the second circle we again need a point on it. We find the touching point T_{12} by constructing the normal plane to l_{12} through M_2 . Finally we get T_{12} and can draw the second sheave.

Since nearly all objects in this construction depend on each other, it is interesting to modify certain elements. A change of M_1 for instance results in an update of nearly the whole construction. Figure 56 shows one final solution.

Constructions in Construct3D are quite efficient regarding the number of construction steps needed to solve an example like this. As can easily be seen by comparing Figure 55 and Figure 56, the whole construction of the example only required us to draw 3 more elements (an angle bisector and 2 normal planes) than what we had to draw for the explanatory draft in Figure 55.

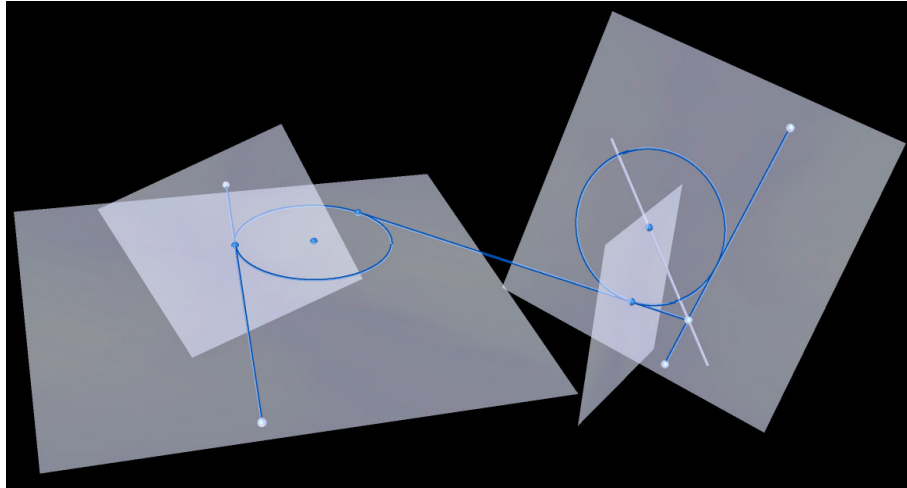


Figure 56: Full construction of the highly dynamic deflection sheave example.

Conic Sections – Proof of DANDELIN

Our first visualization of a geometric proof in dynamic 3D geometry is that of J. P. DANDELIN (1822). It shows that the intersection curve of a cone with a plane can only be a circle, ellipse, hyperbola or parabola. In a geometry course this interactive model can serve as the base for explaining the proof which is hard to understand without any model.

In our APRIL tutorial (section 7.4) we already introduce circle, ellipse, hyperbola and parabola as conic sections. Step by step a cone is intersected by a plane. The three possible cases of intersection are explained: We imagine a plane that is parallel to the intersecting plane through the apex of the cone. In case of a circular or elliptic intersection this parallel plane does not intersect the cone. In case of a parabola, the parallel plane touches the cone exactly in one generator. In case of a hyperbolic intersection the plane through the apex intersects the cone in two generators.

The construction of Dandelin visualizes these 3 types and is the basis for Dandelin's proof. A detailed description of the proof is out of the scope of this work. We refer the reader to [18, 167] or any other geometry book.

We constructed Dandelin's proof in Construct3D (Figure 57). Dynamic modifications of the cone and the intersecting plane allow visualization of all three cases of conic sections with the same virtual model. We only briefly explain the base elements in the proof that can be seen in all three images of Figure 57. The cone and its intersection plane are given. In case of an elliptic intersection two

spheres can be inscribed into the cone which touch the given plane. They are called Dandelin's spheres. We color them blue in Figure 57 and also draw their touching points with the given plane (also in blue). It turns out in the proof that these touching points are special points related to the intersection curve. For instance in the elliptic case (Figure 57 left) some properties are easy to verify dynamically: We take a point on the intersection curve and measure the sum of distances to both touching points. It is constant. We move the point on the intersection curve and the sum of distances stays constant. The fact that the sum of distances is constant from any point of the curve to the focal points is the definition of an ellipse. Therefore the touching points are the focal points and the intersection curve is an ellipse.

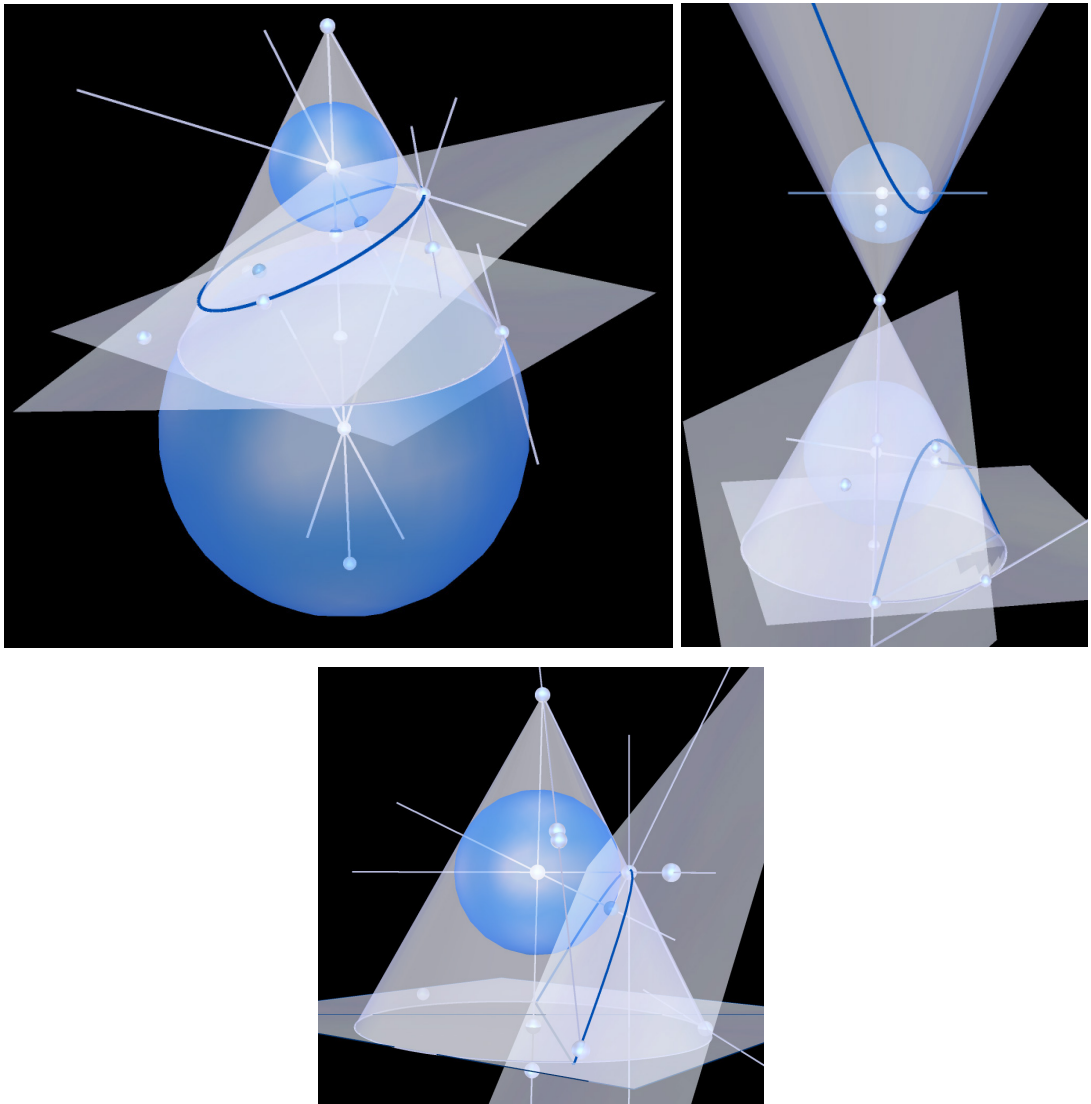


Figure 57: All three cases of elliptic (left), hyperbolic (right) and parabolic (middle) conic sections can be visualized with the same dynamic construction in Construct3D.

The visualization of this proof is a “live, interactive model” which the students can modify themselves. Students are able to interactively modify the cone and all relevant parts of the construction.

One-Sheeted Hyperboloid

Similar to the example of the surface of rotation during our evaluation this example explores the geometric properties of a one-sheeted hyperboloid. It serves as a model to explain all important geometric facts about this ring-type of quadric surface.

We start with the rotation (sweeping) of a line around another line (the axis of rotation). We assume that they are skew lines, in case of parallel or intersecting lines the results are cylinder and cone. When discussing the result of our sweeping operation most students already know the shape of the surface from atomic power plants. We explain that the hyperboloid can also be generated by rotating a hyperbola around an axis. In order to demonstrate that, a cross section of the hyperboloid is constructed with a plane through the axis of rotation (Figure 58). The cross section is a hyperbola. We rotate it again around the axis and get the identical hyperboloid again.

Next we demonstrate that for any surface of revolution a sphere with its center on the axis, touches the surface along a circle (Figure 58). In addition we can explain for instance that two classes of lines lie on a hyperboloid. If students already know about elliptic, parabolic or hyperbolic surface points, we can construct a tangential plane in a point of the hyperboloid. This example would be ideal to show that only hyperbolic surface points lie on this surface. A discussion about curvature would also be obvious in this context.

All parts of the model can be modified to demonstrate possible variants.

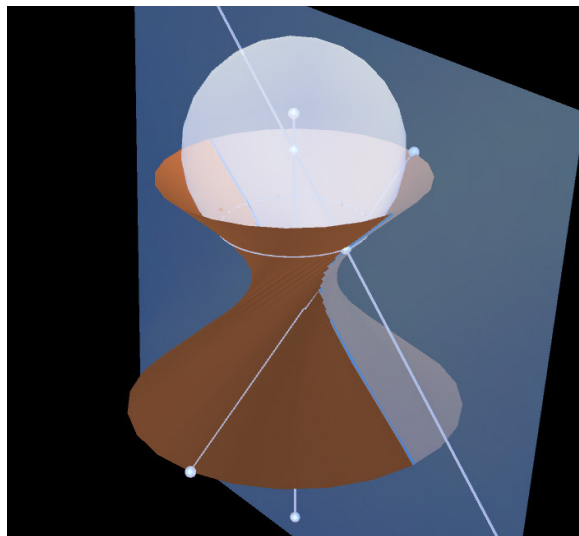


Figure 58: A one-sheeted hyperboloid with a cross section and a sphere touching along a circle.

Villarceau's Circles

In 1848 Y. VILLARCEAU (1813-1883) found a special case of a planar intersection of a torus. If a tangential plane that touches the torus in 2 different points, is intersected with the torus, the result are two congruent circles called "Villarceau's circles".

In order to encourage dynamic modifications, students should start with a general plane to experimentally find out if there are special intersections of a torus. Guided by a teacher they may find out about Villarceau's circles.

In order to do an exact construction later on we are looking for a tangential plane of a torus, that touches it in 2 different points. First we construct a cross section of the torus through the axis of rotation and get two circles as an intersection. A tangential plane to the torus, that also touches both circles must contain the common tangent to both circles. We draw the tangent and together with the normal to the plane containing both circles, we define the tangential plane. By using the slice operation, we intersect the torus and remove one half of it. Figure 59 shows the result.

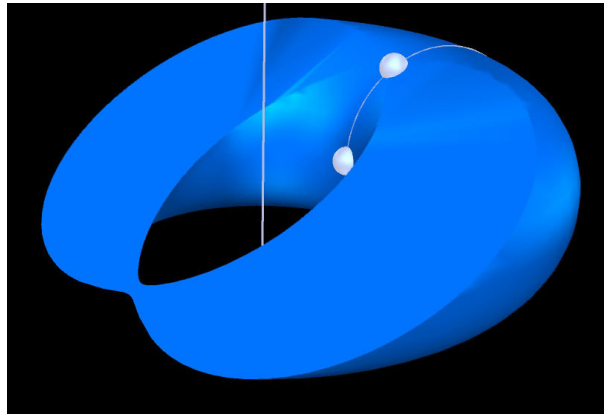


Figure 59: Villarceau's circles. Due to inaccuracies in the triangulation we only see an approximation of both circles – there should only be one singular common point at the bottom of the intersection.

Two-Dimensional Geometry

Of course geometry in a plane can also be done with Construct3D. Currently there is a limited set of functions for planar constructions (which we plan to extend) but basic constructions can already be done. In Figure 60 we constructed a dynamic version of Pappos' Theorem, one of the most fundamental theorems in projective geometry. If A , B , and C are three points on one line, D , E , and F are three points on another line, and AE meets BD at X , AF meets CD at Y , and BF meets CE at Z , then the three points X , Y , and Z are collinear. Pappos's theorem is self-dual. In Figure 60 points ABC as well as DEF can be moved on their lines. Start and end points of the lines define the plane and by moving them, the plane is updated as well.

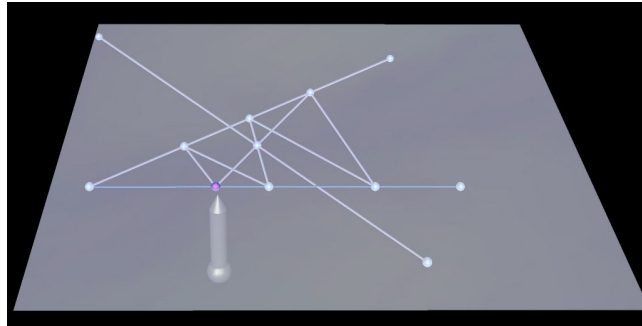


Figure 60: Pappos' Theorem.

Pascal's Theorem (Figure 61), discovered by B. Pascal in 1640 when he was just 16 years old is the dual of Brianchon's theorem. It states that, given a (not necessarily regular, or even convex) hexagon inscribed in a conic section, the three pairs of the continuations of opposite sides meet on a straight line, called the Pascal line (drawn in blue in Figure 61).

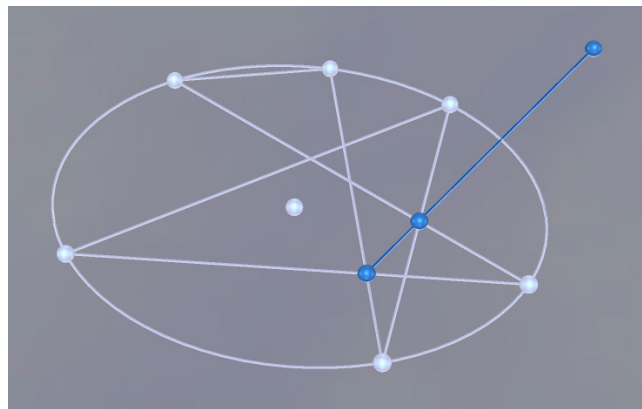


Figure 61: Pascal's Theorem.

However, for dynamic 2D constructions we recommend 2D applications such as described in section 2.4 which are specialised on these tasks. What has not been done yet is a combination of planar and spatial constructions as described below.

Spatial Interpretation of Planar Geometry

Interesting applications of Construct3D open up in the area of spatial interpretation of planar constructions. Since planar constructions can be embedded in 3D space, spatial interpretations of planar constructions are an obvious application area. In the Austrian geometry curriculum [2] there is also an option to teach "solving planar problems by spatial interpretation". In some cases problems with conic sections [161] in high school education can be solved very elegantly and smartly by using a spatial interpretation of the problem.

The basic principle of examples in this category is that a planar construction is interpreted as the normal projection or as the cross section of a spatial situation. The planar elements are then projected onto the spatial objects and the problem is solved in 3D which is much easier in these selected examples. The spatial solution is projected back into the plane to get the planar solution.

In our first example three points are given which lie in between two parallel lines. This planar problem requires finding an ellipse which touches both lines and goes through all 3 points.

This problem can be solved in 2D by methods of projective geometry, taught in university courses. With spatial interpretations this problem can be solved by high school students. The solution to this problem lies in the following spatial interpretation of the planar situation. We assume that both parallel lines are the contour of a cylinder. The axis of the cylinder must lie in the given plane and is the mid-line between both given lines. The radius is the distance from the mid-line to a given “contour” line. The 3 given points are projected onto this cylinder by a projection normal to the given plane. The whole construction can be seen in Figure 62. The normal line through each point intersects the cylinder in 2 different points.

After converting the planar problem into a spatial problem, the solution is easy. We need an ellipse (a planar intersection of the cylinder), that passes through the 3 given points. We only have to put a plane through the points on the cylinder. Since there are 2 solutions for each point, we get 8 possible planes. Therefore in general 8 possible solutions to this problem exist. Each plane through 3 points on the cylinder intersects the cylinder in an ellipse and the ellipse touches the cylinder in its contour. Projecting the ellipse back into the plane (via a normal projection) gives a solution.

In Construct3D we have the possibility to draw directly in 3D and we can follow the planar construction in parallel. At the beginning we use a plane which is parallel to the x/y-plane (drawn in blue in Figure 62). This is our reference plane where we construct the given elements. In order to follow spatial and planar construction in parallel and to see how a possible solution projects back onto the plane we switch on the *top view*. It displays our x/y-parallel plane undistorted and shows in real time how our spatial construction looks when projected back onto the plane. Figure 62 demonstrates this. At the bottom of the image the top view can be seen. You see the given left and right contours of the cylinder as well as 3 points in between these lines. It also shows one final solution to this planar problem – an ellipse passing through the given 3 points and touching both lines. The spatial construction in the upper part of the image shows the original plane (in blue), the 3 given points (in blue) which are projected onto the cylinder. It also shows the elliptic intersection of the cylinder through 3 points. To make it easier to “read” the construction we removed the plane which was intersected with the cylinder and resulted in the blue ellipse.

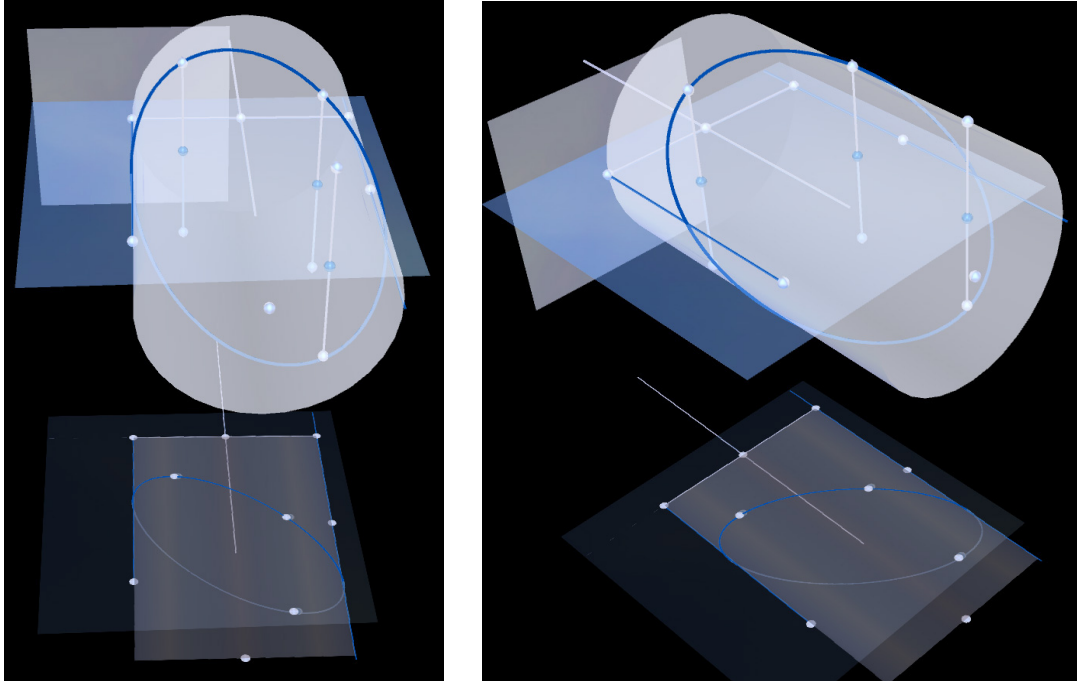


Figure 62: An ellipse through 3 points touching 2 parallel lines. Left: Another view of the spatial situation. We can dynamically change all given elements to study interesting configurations or special cases.

The same example with two intersecting lines (instead of parallel lines) can be constructed analogously by interpreting the given lines as contours of a cone.

The second example in this area that we would like to present is about the following geometric problem: A circle and two points within the circle are given. Find an ellipse, that hyperosculates the sphere and passes through both given points. Hyperosculation means that it has quadruple contact with the sphere (four common points). Figure 63 shows the problem and its solution.

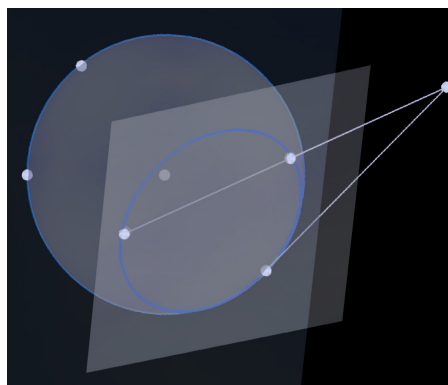


Figure 63: The ellipse hyperosculating the circle passes through two given points inside the circle.

In our spatial interpretation of this problem the circle corresponds to the contour of a sphere. We project both given points A , B onto the sphere via a normal projection. We get A' and B' . There are again 2 possible solutions for each point

on the sphere. The ellipse that we are looking for must touch the circle. Therefore we need to intersect the sphere with a plane that contains both points A' , B' and touches the circle. Such a plane must contain a tangent to the circle in order to touch it. We connect A' and B' and intersect their line with the given plane. We draw the tangent through the intersection point. Together with A' and B' this tangent spans the intersection plane. We only have to intersect it with the sphere and get the desired ellipse. Instead of projecting the ellipse back into the plane, we look at the top view where we see the planar situation (Figure 64).

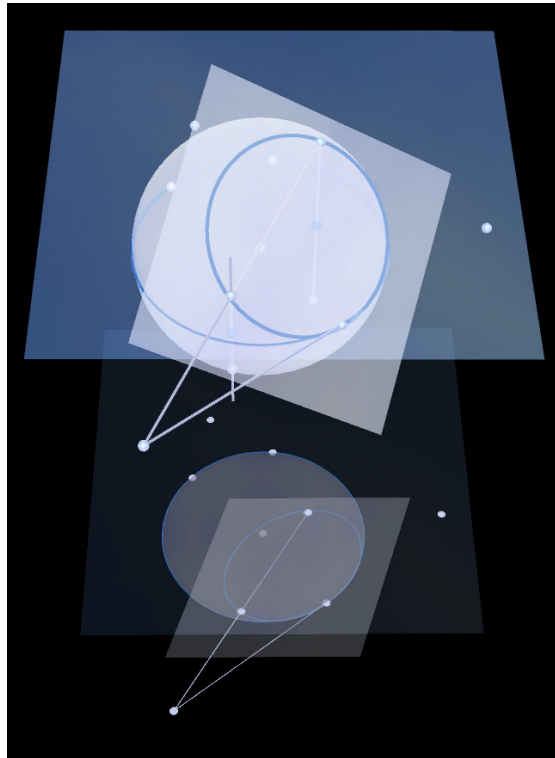


Figure 64: The blue plane contains the given elements, the top view below shows the planar solution.

If A and B lie outside the circle the example can be solved analogical. Instead of interpreting the circle as a sphere, it is interpreted as a one-sheeted hyperboloid. In that case A and B are projected onto the hyperboloid.

Many excellent examples for high school geometry and university education regarding spatial interpretations can be found in [68]. Outside high school geometry spatial interpretations of planar geometry are especially interesting in areas such as geometry of oriented circles [112] or kinematics [69]. Especially in education totally new examples and applications are possible when embedding and combining dynamic 2D geometry in dynamic 3D space. We think this is a very promising area for future work.

In our opinion spatial interpretations of planar constructions require a lot of spatial thinking and understanding of spatial problems. With the help of a tool like Construct3D spatial geometric ideas can be tested, developed and realized in a few

minutes. In these examples dynamic modifications are especially interesting in order to explore various configurations and special cases.

7.2 Teaching Experiences and Pedagogic Consequences

All e-learning content (except Tschupik-cubes which served as an introductory example) is highly dynamic and specifically targeted to the features of Construct3D. It encourages experimentation and is designed to fit constructivist theory (section 2.5).

An interesting aspect in our evaluation was to research which content fits best to our learning environment. We did this by asking experts (our teachers) about their opinion. It is obvious that the example of Tschupik-cubes can easily be taught by using hand drawings or simple educational geometry software. Utilizing Construct3D for such examples only, would be overkill. The strength of Construct3D lies in its dynamic functionality. Therefore examples that exploit the dynamic features and can maybe not be realized with any other software are most useful. During the evaluation it turned out that our example with the intersection curve between two cylinders was a great success. Students set a point on the curve and moved it. By guiding the pen with their hand along the curve they grasped the shape of the curve in 3D. It was obvious that students learned by doing, understood by moving objects in space. Students and their teachers were thrilled by moving tangential planes on a cylinder or on a surface of revolution. These are things which they have not seen or done before. Teachers commented that a great strength of Construct3D is its three-dimensional visualization power. Students see three-dimensional objects and constructions in 3D. It was exciting to experience when students had an ah-ha experience (“AHA Erlebnis”). In most cases this happened during or after dynamic modifications when they viewed the three dimensional object (maybe from a different angle) and suddenly understood what they did not understand before.

We noticed that it is important to encourage inexperienced users to modify objects, to move points. Otherwise the model stays static and is not much different to a CAD model. In the beginning many students did not do this by themselves. Similar to the development of examples for dynamic 2D geometry it will be a future challenge for educators to develop examples that utilize the dynamic features of dynamic 3D geometry and encourage students to use these features.

The pedagogical process for teachers is similar to what they are used to in modern high school geometry education but students as well as teachers face a totally new platform and construction environment that they have to get used to. After teaching five examples as explained in section 7.1 we asked a teacher about his impressions how *teaching* is different if students work collaboratively with Construct3D. He said that it is similar to his geometry classes where he mainly uses CAD programs. After an introductory phase students work on problems by themselves. When

teaching in our environment he sometimes felt dispensable since the learning process is explorative and students collaborate and work by themselves. We think that more teaching experience in this new environment is necessary before any statement about a possible methodology and teaching principles can be made.

We noticed that in some examples Construct3D provided more functionality than actually needed for that specific example. This partly confused students or provided very easy ways to solve geometric problems which was not helpful for the learning process. Therefore, as also possible in some dynamic 2D applications, we will implement functionality for the teacher to disable specific functions of Construct3D when needed.

We adjusted our examples to contents of the geometry curriculum. This was not very difficult to do since modern geometry teachers already interpret the curriculum in a way that supports teaching with new media. Geometry educators are working on new curricula to integrate for instance new curve and surface classes (i.e., B-Spline curves and surfaces) which are supported by CAD programs. These are not present in current curricula since they are hard or impossible to draw with traditional methods. Dynamic educational 2D applications (section 2.4) which are already integrated into modern geometry education are the precursors of dynamic 3D applications such as Construct3D.

We had the luck to work with very encouraged and motivated teachers who helped to design the five examples in section 7.1. Of course we think that in the long run the legal texts of curricula have to match modern geometry education and integrate new media as it is already done in practise by some motivated teachers. Dynamic geometry in 2D and especially in 3D offers new possibilities and allows to go new ways in education.

7.3 Learning Modes

Different learning situations and different learning styles require different presentation of content. In order to adapt our content to various learning styles and students' needs, we outline modes for teacher-supported and autodidactic learning. Some of these modes have already been implemented and tested in a tutorial of Construct3D as we describe in the next section.

1. *Teacher mode*: A teacher performs the whole construction and explains all steps. He has the possibility to use pre-constructed steps of the tutorial to switch back and forth in order to show various states of the construction. He teaches one or more students.
2. *Normal tutorial*: The whole construction or steps of it are "played" including explanations and after the whole construction or after predefined steps students have to repeat them. They are guided by a teacher.

3. *Auto-tutorial*: Students go through the tutorial themselves, listening to pre-recorded explanations of the steps. The instructions are given by recorded speech or a text-to-speech system. Learners have to understand the construction and should be encouraged to repeat it.
4. *Exam mode*: Students must do the whole construction by themselves. At the end there is a check button where the pre-recorded solution can be checked with the constructed solution.

In this context it is important to note that we believe that Construct3D can and will never substitute a teacher or classroom education as it is known today. It is designed to be a valuable addition, offering new chances and possibilities in mathematics and geometry education.

7.4 APRIL Presentation and Introduction

In order to get a standardized introduction into Construct3D for future evaluations we worked on a presentation based on APRIL (Augmented Presentation and Interaction Authoring Language) [91]. Our current tutorial supports “normal mode” and “auto-tutorial mode” only.

APRIL

APRIL (first mentioned in [91]) is a high-level descriptive language for authoring presentations in augmented reality (AR). AR presentations include, besides virtual objects and multi-media content, artefacts of the real world or the user’s real-world environment as important parts of the context and content of the presentation.

By using the existing APRIL-Studierstube binding we implemented an APRIL component as an interface to Construct3D. This component enables us to read in values of all widgets and to trigger them. Therefore we can actively trigger actions in Construct3D and can present a whole construction process. We implemented counters for each object type to read the number of objects of a specific type that the user already generated. Using a combination of widget control and object counting we know within the APRIL presentation what the user is doing. We can follow the construction and can react to his actions in order to keep the presentation in a controlled state.

Construct3D Introduction

The goal of the first presentation is to learn about the menu system of Construct3D and how to do simple constructions with the system. As an introductory example a conic section is constructed.

When the user enters the virtual environment which means that he puts on the HMD and takes pen and pad into his hands, the system welcomes him. All messages and instructions to the user are audio messages and are auto generated by a Text-to-Speech (TTS) system. For this purpose we use a TTS system called

Natural Voices from AT&T [9] which is still one of the best ones available regarding speech quality. We use German voices [70] since we don't want to introduce a language barrier when teaching high school students.

During a presentation there are three additional navigation buttons on the PIP to go forwards and backwards in a presentation or to repeat an explanation. In the beginning the menu system is explained and the user gets an introduction how to press buttons on the menu.

As a first task he has to set 3 points to create a cone. The user is encouraged to modify the cone by dragging the points to learn about the dynamic functionality. Audio commands constantly guide the user and the APRIL presentation interactively reacts to every user action. The next step is to set 3 points and to generate a plane. At every little step the system checks if a sufficient number of points is present and if the required objects (i.e., cone or plane) have been generated. If not, audio messages repeat the commands and explanations and inform the user about wrong actions. Next the user gets an explanation how to turn on and off point mode. He has to select cone and plane in order to intersect them. The female voice of the presenter explains how to switch to another menu to press the "Intersection" button. The intersection curve is generated. Finally the user is encouraged again to move points to see how the intersection curve changes in real time. The three different possibilities for conic sections, namely ellipse, hyperbola and parabola are explained and under which circumstances they occur.

A second example introduces Boolean operations. The user is guided to construct a cone and a sphere. After selecting both the system explains the functionality of the Boolean operations union, intersection and difference. By moving the pen over the menu buttons, the user sees the preview of these functions and is guided to choose a Boolean union operation. The user is congratulated for completing the tutorial. Figure 65 shows the results of both examples.



Figure 65: Left: A user finishes his first example of a conic section. Right: The second example explains the Boolean union of a cone and a sphere (greyscale image).

Remarks

After initial tests we noticed that for longer explanatory texts the German text-to-speech voice is not good enough. Listening to the monotonic pronunciation of the speaker is tiring. The lack of emphasis in the voice is not conducive to encourage learners. The quality of a TTS system very much depends on the language used. We think that it will take a few years until systems with German voices are mature enough to be used for educational purposes. In the same way as good educators use the right accentuation of relevant words to encourage learners, the speech system should be able to do that. Instead of a TTS system we will use human recorded speech in future tutorials.

At our initial tests with inexperienced learners we noticed once more that it is very important for an educational tutorial to check and react to all possible kinds of user input errors since learners often act in an unpredictable way.

8 Evaluations

In order to better understand the educational efficacy of VR/AR in learning, extensive evaluations are necessary. Two evaluations have been conducted with the goals to identify usability issues and to investigate Construct3D's strengths and weaknesses for learning and teaching geometry. The results of both evaluations are presented and discussed. An extensive psychological evaluation is currently in the planning phase and we present an outlook to its design.

8.1 First Evaluation

In June 2000 a first informal pilot study has been conducted to evaluate the efficiency of Construct3D and its value for mathematics and geometry education. We summarize the results as presented in [75].

First Version of Construct3D

In order to interpret the feedback of our participants correctly it is important to know with which system we were working at that time. Regarding hardware we were using a magnetic tracking system as described in section 6.1. Wired magnetic sensors were attached to all tracked devices, resulting in many cables to these sensors (see Figure 66 left). During work with the AR system care had to be taken not to tangle the cables.

We were using a pen (see Figure 66 right) with three buttons. Only the two buttons along the handle of the pen were used in Construct3D. The primary button (the one nearest to the tip) was used for menu selection and setting new points whereas object selection was done with the secondary button. This allowed construction of new points as well as selection at the same time without having to switch between modes.

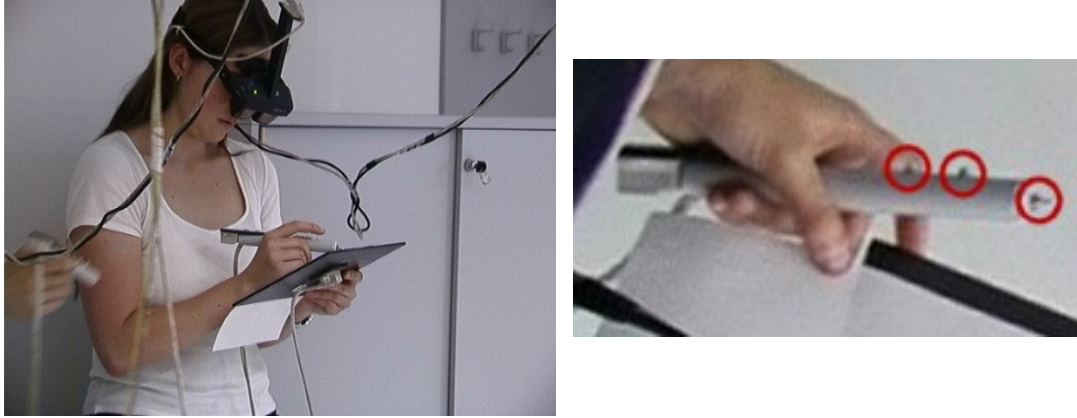


Figure 66: Left: Wired magnetic sensors are mounted to the back of the panel, to the end of the pen and to the head mounted display. Right: The original self-built pen had three buttons (see red circles) – one at the tip and two along its handle.

At that time Construct3D was not a dynamic but a static 3D modeling software. The idea was to construct points and objects by exact coordinates. For easier orientation in the three dimensional coordinate system we attached a simple virtual “grid” to the tip of the pen [7]. It consisted of three lines parallel to the coordinate axis. In addition, the x/y/z values in the given coordinate system were displayed left of the pen as a billboard (always facing the user) in centimeters with millimeter precision.

Of course the functionality of this early version was very limited and only basic objects, basic intersections and system functions were implemented.

These major differences are important to know as students refer to them in the upcoming summary of the results of our first evaluation. A more detailed description of the early system can be found in [75]. The results, written after the first evaluation in June 2000, are included in this work to better understand the development process, the insights we gained, the conclusions we drew. They influenced the development of our hardware and software systems from early 2000 to now.

Subjects

Our 14 participants (6 female, 8 male), aged 22-34, were students in Vienna; 13 of them had geometry education (descriptive geometry) in high school, 9 were students of mathematics and geometry with the aim of becoming high school teachers. On average, they had basic computer skills and good working knowledge of traditional CAD packages.

Methods

The test session consisted of two parts. The first part required each participant to solve a construction example from mathematics education with the help of a tutor in Construct3D. The example stems from vector analysis as taught in 10th grade in

Austria. For high school students, calculating the results is lengthy and rather complex.

In the second part all subjects completed a brief survey. The survey contained an informal section about VR in general and questions about Construct3D.

The following task was assigned to all participants at the beginning of the test: A sphere is given by its midpoint M and a tangent plane A, B, C . Construct the sphere. The line $[A, Q]$ intersects the given sphere. Draw the tangent plane to the sphere in the highest intersection point. The line and the backmost point of the plane span a new plane. Select this plane and interpret its intersection with the sphere. Save the file.

The coordinates of all points were given. Because of inaccuracies with measuring positions in the virtual world (caused by tracker hardware), a tolerance of 2 centimeters per coordinate was acceptable.

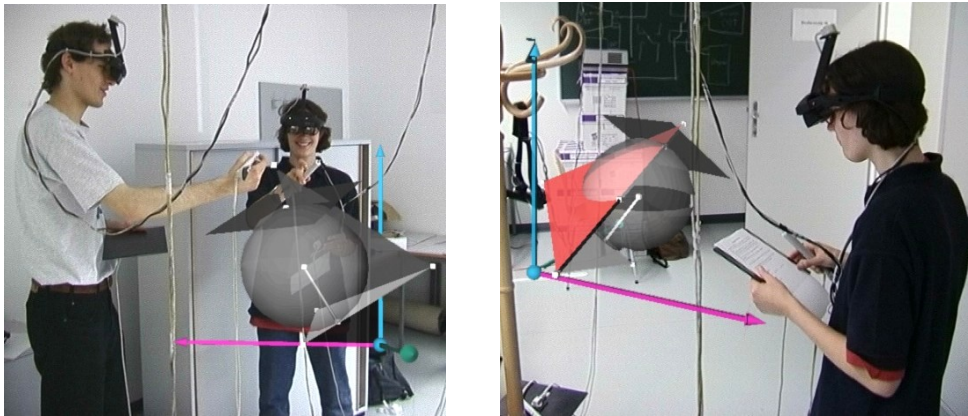


Figure 67: Working in Construct3D. The tutor assists the student while working on the model.

One of the reasons why we chose such a traditional example was to demonstrate how AR could be integrated into today's mathematics education without changes to the curriculum – though we believe that curricula will change once AR is integrated. We chose words such as “backmost” and “highest” to see if students had problems with spatial relationships in the virtual world.

Results

10 of our subjects experienced AR for the first time. All received a one minute introduction into the system consisting of an explanation how to put on the HMD, how to use the pen and the menu. In total, students took between 6 and 13 minutes to solve the task. This reflects the time they spent using the system. While working on their first construction, a tutor was inside the virtual world with them to help when problems occurred and to answer questions. He also explained the user interface. Problems that arose because of a lack of geometrical knowledge occurred only once. The goal of the assistant was to overcome a certain fear and disorientation when having first contact with the new medium.

It was very gratifying for us to see users work with Construct3D in such a 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 our 3D interface. After generating the first plane, a normal to the plane through the midpoint, the intersection point with the plane and the sphere, all students completed their work without further assistance.

The students' interactions with the system were interesting to watch. Using both buttons of the pen proved to be difficult due to ergonomic reasons (see section 6.4). After completing their task, some walked around the object, viewing it from different sides. It was clear that they were proud of what they "built". One student even got down on his knees and looked at the object from below.

The gestural action that we implemented for deselecting all objects proved to be easy and fast for all users. It was impressive to see how quickly students completed a task that would have taken them more than half an hour with pencil on paper.

The first part of the survey confirmed our observations. All students wanted to experience AR again and rated it as a very good playground for experiments. All thought that AR presents a rather good learning environment though questions arose of how to work with larger groups of students.

6 students felt a bit dizzy – some of them during, most of them after leaving the virtual world – probably a light form of cybersickness. We describe negative side effects and ways to reduce them in detail in section 6.2. 10 participants thought that it is easier to view a three dimensional world in AR than on a flat screen. Two of them found using a screen easier. These two had difficulties with the small field of view of the HMD, with orientation and spatial relationships in the virtual world.

The second part of the survey covered questions about Construct3D. All students, who had to do their construction using exact coordinates, reported problems with setting points accurately. Hand-eye coordination proved to be very difficult when spotting a point accurately in 3D space, freely without any haptic feedback, without constraints. As described in detail in section 6.3 this was one of the reasons why we finally implemented dynamic geometry.

A number of potential application areas were mentioned by students at that point: interactive conic sections, vector analysis, enhancing spatial abilities, intersection problems, experiencing space (for very young students) and building three dimensional worlds from two dimensional views.

Finally we asked all students what they liked best and least about Construct3D and what they would like to change. About constructing in AR, they liked walking around and inside objects, the "playful" way of constructing, that spatial relationships and complex three dimensional situations are directly visible. Regarding Construct3D the clearness of the menu system and the audio help system were mentioned positively.

Technical aspects caused problems that are reflected in participants' comments: "slow rendering speed", "bad calibration of the whole system which resulted in small difficulties clicking menu buttons", "inaccuracy of the pen due to position tracking inaccuracies" and "a small field of view of the HMDs". Concerning Construct3D's user interface, people criticize that they had difficulties with choosing the right buttons on the 3-button pen. Our solution to this problem is described in section 6.4. Further they suggested additional features which are already implemented today. One user did not like the transparency we used for solids. We later also resolved this issue by our usability design (section 6.5). As suggested by students we also implemented a snapping method that allows to snap points to other geometric objects.

To summarize the findings: The first evaluation resulted in very constructive feedback that helped to improve many aspects of Construct3D. Especially all aspects described in chapter 6 are in one way or another a result of the first evaluation.

Informal Evaluations and Trial Runs with High School Students

Between the first and second evaluation there were a lot of iterative design steps and trial runs. We were regularly visited by teachers, students, colleagues and friends who evaluated the system and gave feedback on its quality. This helped to constantly improve the application and adapt it to students' needs. Construct3D was not used by students on a regular basis in mathematics and geometry education.

A number of pictures demonstrate our activities during this time of development (Figure 68 and Figure 69).



Figure 68: The teacher explains construction steps to his students. They are watching and waiting for their turn.



Figure 69: Users are working concentrated on their constructions.

8.2 Pedagogical Background

For our second evaluation we studied practical approaches towards evaluation of virtual learning environments and reviewed literature on educational VR/AR applications (section 2.3).

Most evaluations so far concentrate on usability issues of the application rather than its efficacy for supporting learning. Roussos et al. [134] describe a general evaluation framework that encompasses, rather than restricts the multiple dimensions of the issues that need to be examined in virtual learning environments. Taking into account the multidimensionality of learning as well as virtual reality as a field, a number of technical, orientational, affective, cognitive, pedagogical and other aspects were included in the evaluation.

The *technical aspect* examines usability issues, regarding interface, physical problems, and system hardware and software. The *orientation aspect* focuses on the relationship of the user and the virtual environment; it includes navigation, spatial orientation, presence and immersion, and feedback issues. The *affective parameter* evaluates the user's engagement, likes and dislikes, and confidence in the virtual environment. The *cognitive aspect* identifies any improvement of the subject's internal concepts through this learning experience. Finally, the *pedagogical aspect* concerns the teaching approach: how to gain knowledge effectively about the environment and the concepts that are being taught.

As far as the methodological approach is concerned, integration of quantitative and qualitative methodologies seems the best way to face and to catch this complexity

[103]. Riva and Galimberti [128] presented a complex model of data analysis which supports the value of the mixed use of quantitative and qualitative tools.

Our evaluation strategy, similar to the one suggested by Roussos, tries to grasp the multiple dimensions involved in the learning process in a virtual environment. It does not focus on the learning process only but evaluates the system as a whole. The evaluation methods are based on the CIELT pyramid [157].

CIELT (Concept for Interdisciplinary Evaluation of Learning Tools) is a concept to support heterogeneous teams to define goals for the design and evaluation of e-learning systems by visualizing the connectivity of didactic, technical, pedagogical, psychological and evaluation aspects [52].

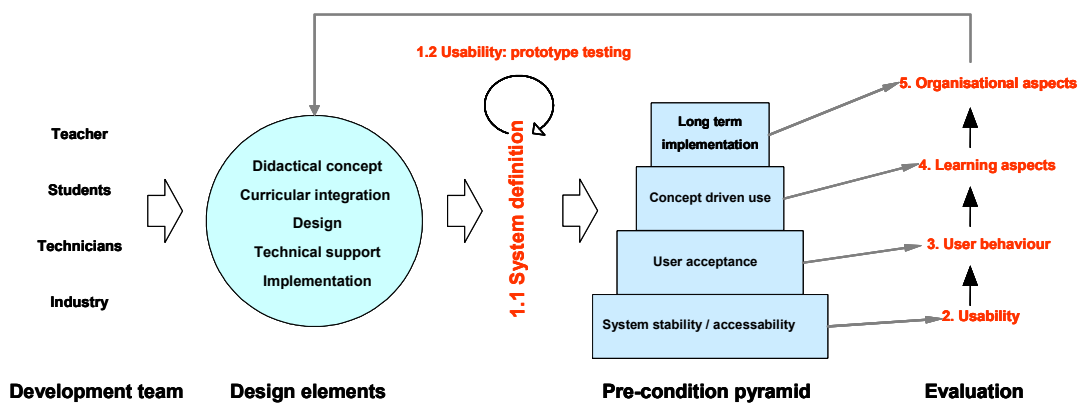


Figure 70: Overview of CIELT

On the left hand side of Figure 70 the different persons of the development team involved in a project are listed, displaying the heterogeneity of such teams. To the right, the design elements are considered, showing that different aspects such as didactical concepts, curricular integration, system design and so on must be integrated in order to define an e-learning system. Prototype testing focusing usability is considered to be the first logical evaluation step in the evaluation process. After conducting first usability tests the precondition pyramid needs to be considered. The pre-condition pyramid proposes different levels of requirements that need to be fulfilled for evaluating specific aspects of a e-learning system. In order to evaluate usability, point 2 (no longer examining a first prototype but a further developed system), the system must be accessible and stable. If this is not given, usability can not be examined properly. In order to examine user behavior (point 3) the system must be accepted by the users, concept driven use and long term implementation are requirements for evaluating learning aspects (point 4) and organizational aspects (point 5) respectively [51, 157].

For our second evaluation we decided to evaluate all aspects as suggested by Roussos. The CIELT concept encompasses all these aspects. Based on the CIELT pyramid (Figure 70) the following Table 4 shows the different levels of evaluation and the methods we used for the respective level. For each step in the CIELT

pyramid we added the names of aspects as suggested by Roussos. The more basic requirements for the e-learning evaluation are listed at the bottom of the table, then ascending to higher levels of evaluation.

<i>Levels focused by the CIELT evaluation concept</i>	<i>Framework by Roussos</i>	<i>Methods</i>
Organizational aspects		(Open) questions and discussions with teachers and students
Learning outcome	Pedagogical aspect, Cognitive aspect	
Learning process	Cognitive aspect	Observation, Expert ratings Informal questionnaire
User acceptance	Orientation aspect + Affective parameter	Questionnaire
Usability	Technical aspect	Questionnaire
Technical requirements	Technical aspect	Observation Questionnaire

Table 4: Overview of the different levels focused by the evaluation concept and the methods applied for the analysis of each level.

As a standardized usability questionnaire we used the ISONORM questionnaire by Prümper [120] and adapted it to our needs. The ISONORM questionnaire was derived from the software ergonomic standard DIN EN ISO 9241 Part 10 (German Industry Standard). This questionnaire is designed to test the usability quality of software following the ISO 9241 part 10 principles. It represents an operationalism of the seven dialog principles in the ISO-standard: suitability for the task, self-descriptiveness, controllability, conformity with user expectations, error tolerance, suitability for individualization as well as suitability for learning. A detailed explanation and description of these usability criteria can be found in [119]. Strong reliabilities are claimed for the sub-scales, although it appears there may be a strong inter-correlation between them as well. Downloads and an on-line version are available, as well as articles about it (all in German though).

We adapted the questionnaire and removed one of seven aspects called suitability for individualization since this kind of individualization (of the menu system and screen configuration) as mentioned in the questionnaire can be found in some professional desktop applications but is not relevant to our AR application. We also removed two questions which were not relevant to AR applications in general.

The ISONORM questionnaire covers usability aspects and partly the orientation aspect in our evaluation. To learn about users' likes and dislikes and their learning experiences - to cover the affective, cognitive and pedagogical aspects - we added an informal questionnaire that allowed a lot of open answers. The informal part can only give a descriptive impression of the participants opinions and cannot substitute a full evaluation of these aspects as it will be conducted later this year within the Lab@Future project and also next year within our national project.

Our full questionnaire as given to the students is attached in Appendix A (in German only).

8.3 Second Evaluation

Three and a half years after the first pilot study we conducted another informal evaluation. However, these two studies are not comparable. The software fundamentally changed in these three years, the evaluation setup was different and also the participants were high school students whereas in the first evaluation the majority of participants were students who studied geometry at university with the aim of becoming teachers. Therefore they had a special interest and knowledge in geometry education. Instead of teaching students with the setup one time for 20 minutes as in the first study, we had 5 training sessions lasting 6 hours in total for this second evaluation.

Study Design

14 students from two high schools in Vienna participated in our evaluation. All are 12th grade students and will graduate in a few months. All attend to geometry classes (descriptive geometry) since the beginning of grade 11. Half of the students are taught geometry using traditional paper and pencil construction methods. From now on we will refer to them as the first group in this document. The other half ("second group") uses CAD programs - especially MicroStation - regularly in classes for doing most of the constructions. The first group of students hardly uses any computers in school, some of them use computers at home whereas students from the second group are very experienced computer users.

Since our evaluation study coincided with the Lab@Future EU project evaluation, we had a hardware and software setup exactly as described in section 5.5. Figure 32 and Figure 33 on page 68 show our participants during the evaluation. During the lessons two students worked in pairs with head mounted displays and were taught by a teacher who watched their constructions while the other four students were sitting in a remote lab. Please refer to details in section 5.5.

6 hours of training with Construct3D were parted into 5 training sessions. Table 5 shows the content taught in each lesson in respect to the pedagogical goals

regarding the geometry curriculum. The content is described in detail in section 7.1.

Lesson No.	Content	Pedagogical goal
1+2	“Tschupik”-Cubes	Introduction into Construct3D; Boolean Operations (beginning of 11 th grade)
3	Surfaces of Revolution	Learning geometric properties of surfaces of revolution; Modeling a surface of revolution (end of 12 th grade)
4	Intersection of two cylinders – Part 1: Point-wise construction of the intersection curve	Learning methods to construct intersection curves. Using method of cutting planes (“Schichtenebenen”) (12 th grade)
5	Intersection of two cylinders – Part 2: Tangents of the intersection curve	Learning how to construct tangents to intersection curves. (12 th grade)
6	Tetrahedron – center of gravity	Vector algebra: Learning about “the” center of gravity in 3D objects (11 th /12 th grade)

Table 5: Content and pedagogical goals of the 5 training sessions.

Both groups of students (7 students each) visited us 5 times for 6 hours in total. We combined the first two lessons to give each pair of students an introduction into Construct3D and let them work with the system for 30 minutes. Four students were introduced into the system within one hour.

It is important to note that not all 7 students of a group worked with the system for all 6 hours. A pair of students usually worked with the system for a whole lesson and we did not exchange pairs during lessons. After the first two lessons the students decided amongst themselves with the teacher who would be working with the system in the next lessons. We tried to find a balance so that the students equally got the chance to construct in augmented reality. Because of organisational issues (such as illness, an exam or obligatory registration at the Austrian army) three students could not participate at one of the six lessons. 2 students used Construct3D only for 30 minutes during the introduction. They both got headache as will be mentioned later on in the results and rather wanted to continue the

evaluation as observers in the remote lab. We had to find a balance so that all those who wanted to work with Construct3D got the chance to do so. Most of the students used Construct3D for 2 hours, some longer (4 of them for 3 hours). What is most important for our evaluation is the fact that all of them used the system for at least 30 minutes or more. Therefore they can judge about its usability (evaluated with the ISONORM questionnaire) and can answer our general questions regarding user acceptance, user behaviour, technical requirements and organisational aspects.

Results

First of all we will summarize the results of our ISONORM usability questionnaire. All questions in the standardized ISONORM questionnaire (Appendix A) must be answered on a scale from “---“ to “+++”. For evaluation purposes we converted this to a numerical scale from -3 to +3 with “---“ as -3, “-/+” as 0 and “+++” as 3. As a total mean value from 14 participants of our study for all 6 categories we got 1,44 on a scale from -3 (worst) to +3 (best). The detailed results for each category are visualized in Table 6.

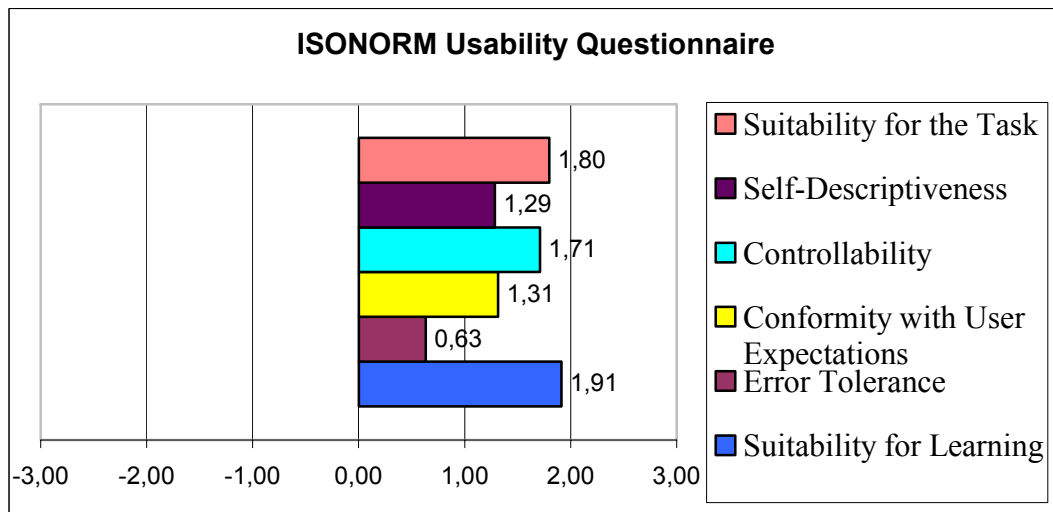


Table 6: Results of the ISONORM usability questionnaire in 6 categories.

We will go into detail on each of the six categories. Within the category “suitability for the task” there are 5 questions plus a question that we added ourselves. We ask if the software performs slow or fast. This question is not included in the standard questionnaire but seems important to us. The mean value is 0,64 (on a scale from -3 to +3 again). We will discuss reasons later on why certain values are higher and others lower. In all upcoming questions high positive values reflect a positive response (+3 is the best), negative values are a negative response (-3 is the worst). All negative values are colored in red, all very positive values ($\geq 2,00$) are bold green. In the right column values for standard deviations are listed. They give an impression how users’ answers varied.

The standardized 5 questions in the first category are if the software

Questions in the category: Suitability to task	Mean value	Standard deviation
is complex/easy to use	2,36	0,50
offers no/all functions to solve tasks efficiently	2,14	1,10
offers no/sufficient functionality to repeat recurring tasks automatically	0,92	1,89
needs (no) superfluous input	1,64	1,50
suits the tasks good/bad	1,93	0,92
Questions in the category: Self-Descriptiveness	Mean value	Standard deviation
provides a bad/good overview over its functionality	1,29	1,59
uses incomprehensible/comprehensible terms in menus	2,29	0,47
informs sufficiently/does not inform about valid and invalid input	0,29	1,64
Questions in the category: Controllability	Mean value	Standard deviation
offers no/an opportunity to stop the task and continue at the same point later on	2,29	1,07
forces to follow an/no unnecessary strict sequence of steps	1,29	1,44
allows no/an easy change between menus	2,50	0,52
ss designed in a way so that the user cannot/can influence how and which information are displayed	1,69	1,49
forces unnecessary/no unnecessary breaks during work	0,79	1,48
Questions in the category: Conformity with user expectations	Mean value	Standard deviation
complicates/eases orientation because of inconsistent / consistent design	1,86	0,53
does not inform/informs if input was successful or not	1,00	1,30
informs insufficiently/sufficiently what it currently does	0,36	1,82

reacts with difficult/easy to predict processing time	0,79	1,76
Cannot/can be used by consistent schemes	2,57	0,65

Questions in the category: Error Tolerance	Mean value	Standard deviation
is designed in a way that small errors can/cannot have big consequences	1,36	1,65
informs too late/immediately about wrong inputs	0,00	2,12
outputs difficult/easy to understand error messages	0,31	1,75
requires in case of errors a generally high/low effort to correct them	2,07	1,21
does not/does give concrete tips for error correction	-0,57	1,83

Questions in the category: Suitability for Learning	Mean value	Standard deviation
requires a lot/very little time to learn	2,29	1,07
does not/does encourage to try new functions	2,43	0,76
requires/does not require that the user remembers many details	2,00	0,78
is designed in a way that things you learned once are memorized badly/well	2,29	1,14
is difficult/easy to learn without somebody's help or a manual	0,57	1,34

The final part of the ISONORM questionnaire are general questions about users' experience with computers. 9 of the students from age 17-19 (mean value 17,67) are male, 5 are female. They work in average 6,45 years with computers and spend 20,77 hours per week in front of computers. The time per week varies a lot from 1 hour for a participant to 50 hours per week for another. They judge their own knowledge and skills in working with Construct3D as 2,08 (again on a scale from -3 to +3). The first group of students works with CAD3D only in geometry classes, the second group uses a wide variety of software privately and in school, and works with CAD3D and MicroStation in geometry classes.

In addition to the ISONORM questionnaire we ask general questions about Construct3D. These are structured into user acceptance and motivational aspects,

learning in Augmented Reality, unpleasant side effects, organisational and practical issues and open comments.

First we asked about user acceptance and motivational aspects. The “fun factor” – if students had fun working with Construct3D – was very high with 1,21 on a scale from (1 = best to 5 = worst). On the same scale the question if they want to work with the application again was rated with 1,79. Only one student gave a rating of 4, mentioning that he got headache. Back on the scale from -3 to +3 we asked the computer experienced 2nd group of students about their motivation *before* the evaluation to participate. With a mean value of 2,00 they were highly motivated. Though all of them would participate again, we could only raise the high motivation of half of them. The motivation of the other half of students did not raise during the evaluation (probably based on some technical problems that we will elaborate later).

The next set of questions concerned learning aspects in Augmented Reality. From now on only the scale from -3 to +3 is used. The general question if users think that learning geometry in Augmented Reality is possible is rated with 1.31. Based on the fact that all geometric content (except surfaces of revolution for one group) was already taught in their geometry classes, the question if they learned something new gets -0,38. Students from the 1st group comment that surfaces of revolution were new and 2 users comment that due to better viewing possibilities offered by Construct3D (compared to traditional methods and CAD programs) they get a better imagination of 3D objects. Asked if they understood geometric content better the result is 0,23 in average (the 1st “computer inexperienced” group says 1,42, in contrast the mean value of the 2nd group is -1,16). As a reason why they understood things better learners say “because I can walk around the object”, “it supports my imagination”, “better viewing capabilities therefore better imagination”, “it was very descriptive, can better imagine it because it’s 3D”, “I see everything in 3D, can even go *around* the object thereby changing my view”.

10 participants say they comprehend the three dimensional objects and the geometry better by working with a head-mounted display whereas 3 say by viewing it on the monitor. When asked for their “first time” experience with a construction tool, 9 think their first construction with Construct3D was easier to do whereas 5 say that their first work with a desktop based CAD tool was easier. Asked if they can imagine to work with Construct3D without ever having worked with a desktop based CAD application before, 13 say yes and only one says no. The next question was targeted towards the traditional approach of teaching geometry which is based on teaching how to construct in normal projections (top/front/side views). We ask if students can imagine working with Construct3D without ever having learned to construct in normal views (top view, front view). 8 say yes and see no problems in constructing directly in 3D but 6 say no and all of them comment that “there would be a lack of basic knowledge”. It is interesting to note though that both our teachers do not agree on the latter. They think there are

no problems working with Construct3D without prior knowledge of constructions in normal views.

We asked learners about their opinion of prerequisites that are needed before working with Construct3D. Opinions vary: One thinks that basic spatial abilities are required, two think no prior knowledge is necessary, two others think that basic knowledge about CAD software is of advantage and half of the subjects think basic geometric knowledge is required.

We tried to find a balance regarding the difficulty of the content (see a description of the five examples in section 7.1). We were not sure if it was too difficult for the students because they also had to concentrate on how to use a totally new software and were immersed in a new learning environment. On average (-3 = very difficult, +3 = very easy) the content seemed rather easy (mean value 1,31) to them. We were also surprised by the answers to the question if students would rather want to work alone with Construct3D (none), with a colleague (1 student), only with a teacher (1 student) or like in our evaluation with a colleague and a teacher (9 students). Interestingly 2 students added a fifth option – they would rather want to work in small groups.

We were interested if participants can imagine the use of Construct3D in its current state in high school or university geometry education if the costs of the hardware were low. 10 say yes and suggest the following application areas “geometry, mathematics, industrial arts, university study of architecture, design and drafting”. 4 say no because “the current version crashes too often; it requires too much previous knowledge and is difficult to control for the teacher; working on the PC with CAD software is easier”. Except for costs, can students imagine other reasons that could avoid the usage of Construct3D in classes? Some say no, others see reasons that could hinder application of this technology in schools such as “headache is unpleasant; for collaborative work discipline is needed; the will of teachers to use new technologies; a lot of free space is needed for constructing; negative side effects; availability for 2 students only; quality of software (e.g. no crashes); difficult to control for the teacher; technology is in an early stage; hardware requirements; rendering speed”.

As mentioned several times before, a topic that came up were negative side effects when working with HMDs. 8 students did not feel any negative side effects, 6 students did. They reported “headache after working for one hour”, “headache after approx. 20 minutes or fatigue because of exertion”, “headache after working for 30 minutes”, “feeling tired after working for one hour”, “working for 45 minutes, felt tired and slightly dizzy” and one even reported that after working for one hour he had headache for 2 days. We will discuss these issues later and go into detail in section 6.2. When asked if they felt disoriented in the virtual world 13 said no and only one agreed.

Last but not least there is an organisational question and some questions open for comments. Regarding the organisation of the evaluation we only asked the 2nd group, and students' feedback was very positive. They would all participate again, found the financial compensation of €70 for 5 training sessions sufficient, said that it was no big additional effort for them to visit us 5 times and only one said that his time schedule was a bit tight during that week. They visited us in their leisure time in the afternoon.

Finally three questions are left, namely what students liked best, least and what should be improved. These are open questions and we report the students' free answers. Our participants liked best "the work with Construct3D", "3D vision", "spatial image and the dynamic change even if you only move a point", "collaboration, work with HMD, pen and audio headsets", "the possibility to change everything afterwards", "3D work, headsets", "work with the HMD", "that it was a new experience", "*cool* to test a new technology that will be common in a few years", "working in the VR lab", "new technology, 3D glasses, 3D vision and perception" and "constructing with the system".

What they like least was "headache afterwards", "crashes of the software", "headache afterwards", "the waiting if something was broken or crashed and we couldn't do anything" and "that the program is not technically mature yet".

Suggestions for improvements are "headache should not occur", "the HMDs should be improved – some see double images, I get headache. Computers crashed", "helmet and battery should get light weight", "the software", "rendering speed should get better", "stability and framerate should improve", "HMDs and helmets should be adjusted to persons; reduce dizziness and fatigue", "speed of the program".

Finally, open comments by the students were mostly positive, sometimes even enthusiastic.

Discussion

In general we think the usability results of the ISONORM questionnaire are very good. All mean values of the 6 main categories are positive and some are very high. However we want to go into detail to try to explain why some aspects got better and others not such good rating. Regarding the main categories it is clearly visible in Table 6 that "error tolerance" got the lowest rating.

A detailed look at the questions in this category reveals that the worst grade (of the whole questionnaire) was given to "the software does not give concrete tips for error correction". We think that there is a good reason why Construct3D does not give detailed tips for error correction. On the one hand it is technically difficult to detect the exact source of the error and react to it accordingly by maybe analyzing the situation and the reasons why the error occurred. On the other hand – and that is valid for the whole category of "error tolerance" – the pedagogic theory of

constructivism on which Construct3D is based suggests explorative learning. Construct3D encourages experimentation and encourages learning by trial and error. Therefore checking for errors before they actually happen is not in our interest and is not the task of Construct3D. Therefore also the bad rating of “software informs too late about wrong inputs” is understandable for us. The reason for errors can be manifold and for correction of geometric errors students should think critically and can ask their teacher for advice. However it is true that Construct3D could output more error messages (for example via generated voice) if a geometric operation fails because of errors of the geometry kernel or wrong input elements. If multiple objects are selected we could also apply a filter to choose the correct ones for the specific operation. We are glad to see that users appreciate the fact that “in case of errors the software generally requires low effort to correct them” which got a rating of 2,07. This confirms that the undo and redo functionality (as described in section 4.2) works in practice.

The categories “self-descriptiveness” and “conformity with user expectations” also got lower grades than the rest. Regarding self-descriptiveness of Construct3D we already have plans to improve it. We will add better labeling and tool tips to the widgets on our PIP in order to explain all menu items in a better way. We are also restructuring the menu system to give a better overview of the functionality.

In order to inform the user about valid and invalid input we already provide the preview function. We noticed during the evaluation that the preview function was not utilized by the students in a way we designed it to be. If a preview wireframe model could not be generated by the application and did not appear, students did not interpret that as a clue for invalid input elements. Sometimes they did not check if there is a preview model and in most cases they clicked on the button to generate the final model disregarding the fact that the function could not be executed. One problem could be that users have to look away from their menu on the virtual model to see if the preview is successful. This constant looking and checking gets tedious after some time of work. Experienced users do it automatically but for inexperienced it may be tiring. We suggest and plan to add audio messages in the following way. On every preview trial a voice message or unobtrusive sound reports either successful model generation, invalid input elements or an internal algorithmic problem if this is the reason for not generating the preview. This must be integrated in a way not to overload the user with constant audio messages.

We assume that this will also solve some problems mentioned in the category “conformity with user expectations”. We expect that especially the low rating of “informs insufficiently what it currently does” can be improved with better self-descriptiveness and an improved preview function as mentioned above.

An obvious problem during the evaluation was that the software’s response time is unpredictable (rating: 0,79). Our question if the software performs fast or slow was rated 0,64. The opinions are unanimous on that. The reasons for slow rendering and unpredictable processing time are manifold. They are mainly founded in our

complex evaluation setup. During our pre-tests of the setup with 2 or 3 people we did not notice reduced performance. The problems got obvious when all 6 students plus their teacher used Construct3D in our distributed Studierstube setup. Such a large scale test of our distributed system [57] was not done before. It revealed interesting issues. Our implementation of communication between host and clients is not as scalable as needed for such a large user base. As explained in more detail in section 3.1 one master sends updates of the scene graph to all slaves via a reliable multicast protocol. If packets get lost or clients do not receive all updates within a certain period of time due to network congestion, they ask the host to send updates again. Since we also had communication software such as audio chat, video chat and a whiteboard running simultaneously at our evaluation which consumed network bandwidth as well, we assume that a lot of updates of the distributed system got lost. We speculate the master got continuous requests to send again, driving the master's performance down to an unacceptable level. These performance breakdowns were unpredictable and low frame rate caused jerking images. They did not occur in all our examples and only happened from time to time. In rare cases we had to stop or restart distribution of the scene graph in order to allow both users with HMDs to continue working on the construction with usual frame rates between 15-30 fps. Due to students' high motivation they tolerated these problems more than we assumed they would. We plan to investigate the problematic issues and want to improve the scalability of our distributed system.

Apart from these problems most of the other ratings are very high and very promising. Especially the categories "suitability for learning" and "suitability for task" receive the highest grading which is very important for us. To summarize all marks above 2,00 (which means it got the highest grading '+++'), the majority of students think: Construct3D is "easy to use", "offers all functions to solve tasks efficiently", "uses comprehensible terms in menus", "offers good opportunities to stop the task and continue at the same point later on", "allows to change easily between menus", "can be used by consistent schemes" (highest overall grade!), "requires in case of errors low effort to correct them", "requires little time to learn", "encourages to try new functions", "does not require the user to remember many details" and "is designed in a way that things you learned once are memorized well". We are very happy that this confirms and summarizes our main priorities for the development of Construct3D. In our opinion the highest priorities for an educational application that complies with concepts of constructivist theory are that it

- is easy to use and requires little time to learn,
- encourages learners to try new functions,
- can be used consistently and is designed in a way that things you learned once are memorized well.

We think these priorities cannot be described in a strictly technical way by specifying certain functional requirements for an application. Of course there are useful guidelines that should be followed but in the end good usability of an application is a sum of many small aspects. We worked in many different areas (as described in chapter 6) to convey the right “feeling” to users. We hope that our contribution in this area helps developers of other educational virtual environments.

Apart from the ISONORM questionnaire, our general questions about Construct3D also revealed interesting and some unexpected results.

User acceptance was generally very high as well as the “fun-factor”. The high initial motivation of our participants was even increased for half of them during the evaluation, the other students’ motivation decreased probably because of slow rendering performance, negative side effects or higher expectations.

The next aspect are effects of training on learning. All students basically agree that Augmented Reality is suited very well for learning geometry. The first group learned new content - surfaces of revolution. Their rating on the question if they learned new content was 0,83 whereas the second group said -1,5. We had hoped to see higher results on their subjective rating regarding if they understood geometric content better (mean 0,23). However, if we have a look at the results per group the 1st group with traditional geometry education gives a high grading (1,42) whereas the mean value of the 2nd group is negative with -1,16. Students from the first group reason that they understood better because they could walk around objects, saw everything in 3D and could imagine objects better. We did not get answers on this question from students of the 2nd group. We speculate that due to their constant work with CAD programs in geometry courses they already have a better imagination of geometric objects and geometric relations in 3D than those who only draw with pencil on paper. There are many possible interpretations and without having more data they are only a matter of speculation.

The majority of learners (10 versus 3) understand constructions better when viewing them stereoscopically through the HMD compared to a monitor. All except one can imagine working with Construct3D without ever having worked with another CAD application before. As prerequisites for the work with Construct3D they mainly identify basic geometric knowledge. Half of the students throughout both groups think that it is necessary to know how to construct in normal views (top/front views) before working with Construct3D. Most of them comment that otherwise basic geometric knowledge is probably missing. It is interesting that both teachers do not share this view. They see no problems in working with the application without knowing about constructions in normal views. The students’ answers are also consistent with their opinion about prerequisites. All those who think that basic geometric knowledge is a prerequisite before working with Construct3D think that knowledge of construction in normal views is important too. They apparently identify basic geometric knowledge with the knowledge to construct in normal views. This points to a misconception since

knowledge about geometry, geometric relations, about geometric properties of curves and surfaces is independent of representation – be it in a normal views on paper or in 3D in Augmented Reality.

It was interesting for us to read that 9 students prefer to work with a teacher and a second student since we thought students would prefer to work alone or with a colleague but without their teacher. We were wrong. The content did not seem to be too difficult (mean value 0,31).

We are a bit concerned about the fact that nearly half of the students felt negative side effects since we constantly asked them during the evaluation whether they feel dizzy or get headache. They always confirmed that they are ok. In the anonymous evaluation questionnaires it turned out that some of them actually felt negative side effects. One female student reported headache and eye strain after 20 minutes of work in the virtual environment but she did not stop working and wanted to use Construct3D again (in total she worked for 3 hours with the system). In retrospect we know that our one hour lessons were simply too long for continuous work with an HMD. Since negative side effects are a general 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 HMDs. We summarized possible reasons of cybersickness that may be relevant to our virtual environment in section 6.2. There we identified some problematic issues such as accommodation problems, low frame rate, lag or bad fitting helmets. We will work with top priority on reducing negative side effects.

Nearly all students can imagine using the current version of Construct3D in high school or university education. They gave very useful comments and suggestions for future improvements though. Our future plans to improve software and hardware and our ideas for applicability in present-day schools are summarized in section 9.2. We did not consider some practical issues in detail before this second evaluation. For a setup similar to our lab setup working space of at least 3x3 meters is needed. Let us assume a school really wants to build up a similar setup. Even if there is money to buy the hardware there is often a lack of space in schools. A teacher also criticized the fact that only two students can be taught at a time in the immersive setup. A combination with a desktop setup would be required that integrates and allows all other students to actively participate in the construction. For real applicability of dynamic 3D geometry in today's schools we plan to work on a very simple desktop interface for Construct3D which is easy to learn and uses a mouse only.

The students' open comments in the final section of our questionnaire speak for themselves and fully reflect the positive results as mentioned above as well as the problems that occurred. It was very encouraging that one teacher said he would prefer to use Construct3D in the next semester in his courses already instead of using other CAD programs. The main advantages for him are stereoscopic viewing with the HMD which greatly enhances students' imagination in his opinion, the

possibility to walk around objects and the ease of learning how to work with Construct3D.

8.4 Conclusions for a Psychological Evaluation of Construct3D

As mentioned before, an extensive psychological evaluation of Construct3D is planned. We intend to address the following research questions in our study: effects of training on performance in tasks similar to the training; transfer of the training effect to more distant spatial tasks; effects of training on strategy use; dependency of individual training effects on pretest spatial ability, verbal ability, and reasoning ability and gender differences. We give an outlook on the study design and the instruments used.

Conclusions and Research Questions for the Evaluation Study

The first conclusion that follows from the multifaceted structure of spatial ability (section 2.6) is that a good training study should use *more than one spatial test*, in order to cover several aspects of spatial ability. Which factors might be affected by a training using Construct3D? It seems likely that training which helps participants to “handle” movements and transformations in three-dimensional space largely affects performance in tasks using three-dimensional stimuli that require the imaginal manipulation and transformation of such stimuli. We do not expect the training to have effects on basic skills such as identifying incomplete figures. Thus, the training should mostly foster performance in tasks from the visualization factor, and possibly the spatial relations factor as far as three-dimensional stimuli are concerned. For this reason, we intend to use several visualization tests, including perspective-change tasks (which covers Lohman’s spatial orientation factor), and one task from the spatial relations factor that uses three-dimensional stimuli. Within the visualization factor, tests differ in closeness to what is being trained with Construct3D. Thus, we can assess the bandwidth of transfer to other tasks by grouping the tests according to their closeness to the content of the tutorial. We will use the following spatial ability tests: Mental Cutting Test (MCT), Object Perspective Test (OPT), Mental Rotations Test (MRT), Purdue Spatial Visualizations Test: Rotations (PSVT:R) and Differential Aptitude Tests: Spatial Relations (DAT:SR).

Especially if we find broad transfer of effects with Construct3D, it seems likely that the underlying factor that the training affects is *strategy use*. The training might generally lead participants to use holistic/visualizational strategies more often in all kinds of spatial tasks. To test this hypothesis, we will include strategy assessments for all tests in pre- and posttest.

An important aspect is the question of *aptitude-treatment interactions*. Does the training have particularly strong effects on individuals with certain patterns of verbal, reasoning, and spatial abilities at pretest? Based on Souvignier’s review,

one might expect to find particularly strong effects in participants with relatively high baseline spatial ability. However, since training allows participants to practice visualization by simply “drawing objects into space”, it might also be especially effective for participants with relatively low pretest spatial skills. We intend to measure reasoning and verbal ability at pretest and to conduct an exploratory analysis of aptitude-treatment interactions.

Finally, any evaluation study in the spatial domain should assess effects of the training on *gender differences* in performance. As women more often than men use analytic strategies on spatial tests, training with Construct3D which focuses on visualization might have different effects on female than on male participants. As with aptitude-treatment interactions, we do not have specific predictions here. However, given the current publicity of issues of cognitive gender differences and their origin, it is certainly interesting to see how different types of training affect such differences.

Conclusions for Training Design

Souvignier’s [145] review allows for some conclusions for the design of the training. First, the training should not focus on only one specific skill. As the experience that Construct3D offers – the possibility to freely construct and change objects in three-dimensional space – is per se a new way of interacting with spatial information, we expect it to have rather general effects on students’ ability to imagine spatial information and transformations. Souvignier’s suggests to train strategy flexibility, rather than use of one particular strategy. This may be more difficult to meet because Construct3D clearly offers a visualizational/holistic approach to dealing with spatial information. We will investigate how effective this type of instruction is for participants with different aptitude profiles. Concerning the training elements that Souvignier views important, two are clearly present in Construct3D: Immediate feedback and opportunities for hands-on interaction.

With respect to the design of content, we will draw upon constructivist theories of learning. We intend to integrate familiar types of information in the 3D environment and to use several learning modes in the training, from teacher-supported to autodidactic learning (see section 7.3).

Planned Study Design

We will conduct a pre-/posttest experiment with four different training groups and one untrained control group. Participants will be 11th and 12th grade students, attending secondary schools in Vienna. Each group will consist of 50 students. At pretest, all participants will be presented with a battery of spatial tests (including strategy assessments) and tests of verbal and reasoning ability. Then, each participant will be randomly assigned to one of five groups:

- The “Construct3D group” is being trained by using Construct3D.

- The “standard instruction/tutoring group” works with a tutor using standard teaching methods without Construct3D or any other computer support.
- The “standard instruction/school group” participates in pre- and posttest, but is taught the subject matter of the tutorial in their normal school classes between pre- and posttest.
- The “untrained control group” participates in the pre- and posttest but does not receive any training.

We think about adding a 5th group working with a basic desktop version of Construct3D (see section 9.2).

The reason we use an individual tutoring group is because this condition is probably “harder to beat” than standard class-wise descriptive geometry (DG) instruction. The condition is parallel to teaching with Construct3D in that it involves individual sessions. As good tutors will adapt to the needs of each participant and will use well-practiced methods to improve their understanding, it will be hard for Construct3D to lead to better performance gains than tutoring. Tutors are likely to use hand-drawing and sketching as teaching methods, thus, they will provide a “hands-on” experience that has proven to be particularly effective in other training studies. However, we expect that training with Construct3D leads to broader transfer of training gains than tutoring.

9 Conclusion

In this chapter we recapitulate our findings and summarize guidelines for developers of educational VR/AR applications. We give an outlook on future work that we plan to conduct and want to encourage other researchers to look into various research areas that are worthwhile investigating.

9.1 Findings

Our literature review revealed that nearly all previous educational VR/AR projects ended after a period of trial studies and first evaluations. No reports about continuous progress or ongoing development can be found in literature. Our work on Construct3D goes one step further. In this thesis we describe an ongoing development process that started four years ago and will continue at least for the next two years. We report about two evaluations but cannot estimate yet how Construct3D will change the learning and teaching process in the long term.

In an early phase of our work we realized that certain functionality must be provided by our application to be useful for teaching geometry. In order to implement sufficient functionality (summarized in chapter 4) in a reasonable time we decided to use a commercially available geometry kernel. We do not regret this though we are aware of dependencies between our application and external toolkits.

During the development of Construct3D specific user needs became obvious and new features had to be implemented. It also improved robustness of Studierstube, distributed Inventor and Coin. Application development was driving technological development and vice versa. Regarding hardware setups (chapter 5) the immersive setup with HMDs (section 5.1) was identified as most promising because of its advantages compared to other setups. It confirmed that applications must fit to the technology used and vice versa.

Usability design (chapter 6) is an absolute necessity to adapt existing hardware and software to users' needs. Good usability is a sum of small iterative improvements

which serve the application's purpose. Their combination improves usability by a far greater extent than each component alone. In order to quantify improvements regular evaluations are of importance. Developers of AR/VR applications usually give the advice to start evaluations at a very early stage to learn about potential problems and to iteratively improve an application.

In chapter 8 we gathered evidence from both evaluations that actually seeing things in 3D and interacting with them can enhance students' understanding of three-dimensional geometry. In our opinion the highest priorities for an educational application complying with concepts of constructivist theory are that it

- is easy to use and requires little time to learn
- encourages learners to try new functions
- can be used by consistent schemes and is designed in a way that things you learned once are memorized well.

Prerequisites are that all necessary educational tasks can be carried out with the application. Users rated that Construct3D does all of that in an excellent way.

At the evaluations some users reported negative side effects (section 6.2) when working with Construct3D for a longer time. Therefore future work periods will be limited to 20-30 minutes. We think that the problem of cybersickness must be faced instead of avoided by using other display technologies. Ways how to reduce side effects in our setup are described. However, a very low percentage of users might feel effects that cannot be compensated with technical adaptations only (we had one user at the 2nd evaluation). For those, learning with Construct3D in an immersive setup is not convenient. It is rather reasonable to let them work with desktop based versions instead.

The advantages of working in an immersive AR setup are manifold. Users see their own body and hand as well as the effects of their actions while working. The construction process physically involves students and resembles handcraft more than traditional computer operation. A spatial relationship between the user's body and the geometric object in 3D space is established. We believe these are key factors in the potential success of using AR for teaching geometry.

We developed Construct3D to a point where it can be productively used for educational purposes. Due to a wide range of features useful content (presented in chapter 7) which utilizes dynamic functionality is easy to produce for teachers and students in a very short time. Our tool allows novel ways of teaching. More and more teachers are expressing their interest in Construct3D and some are visiting us regularly with their students.

In addition to its applicability in geometry education, Construct3D provides a platform to do evaluations of very complex matters such as training spatial

abilities. Previous VR/AR applications did not reach the state of providing a robust framework for extensive evaluations on these matters.

We present a system for conducting research on learning with new media. Since development of our application is ongoing we will see how teaching with this medium will change. Time will show how teachers will adapt content to the advanced features provided by our tool. Dynamic 3D construction and visualization capabilities are its fundamental strengths. Construct3D enables teaching of new dynamic geometric content that is too difficult or impossible to draw with pencil on paper or with existing CAD programs.

We hope that our contribution helps educators, students as well as developers of other educational virtual environments.

9.2 Future Work

The next paragraphs list ideas of possible additional features. Depending on Construct3D's future practical applications some features might be of higher importance than others. In general development of educational geometry or CAD software is a never ending process.

Additional Features

Current dynamic 2D programs offer the possibility to save reoccurring construction steps in macros. A macro is a part of a construction that can be reused for other input elements, comparable to a subroutine in a computer program. We could implement a macro by storing a specific part of our UndoRedoList which encapsulates the construction steps that the user defines as a macro. Later it can be applied to other input elements. Therefore we would basically have to exchange all references to the old input elements with names of new input elements in the SoCommandKits contained in the macro. Finally the adapted macro-part can be added to the UndoRedoList.

Some dynamic 2D applications also provide functionality to set dependencies for independent objects at a later state. They enable the user for instance to fix an existing “free” point to a curve. This is not possible in Construct3D yet but could be implemented if the need arises.

Currently many curves and surface classes are not supported by Construct3D. Driven by the demand we might implement curves and surfaces such as hyperbola, parabola, polyhedra, quadrics, general sweep surfaces, canal surfaces, helical surfaces, spiral surfaces and many more.

There is no way of doing constrained modeling in Construct3D yet. It is not possible to assign a fixed length to a line or a fixed radius to a sphere for instance. As a consequence we were not able to construct a regular tetrahedron when

developing our content in section 7.1. Basic features of constrained modeling would be useful for various purposes. In case we want to use Construct3D for spatial kinematics, constrained modeling is an absolute must.

Another possible application area would be differential geometry. Dynamic visualization of difficult concepts or proofs is a strength of Construct3D. In order to do differential geometry additional features would be useful such as drawing Gaussian curvature and principal curvatures in a surface point, curvatures in points on curves, drawing indicatrices and many more. We can imagine a wide range of very useful, interesting and visually appealing geometric properties that could be visualized. Since ACIS returns “arbitrary” derivatives in points of curves and surfaces, implementation would be relatively easy and is only limited by development time.

Another limit might be accuracy and general problems of calculations with boundary representations. We noticed that in some cases internal representation in ACIS (especially of sweep surfaces) is not accurate enough to do exact geometric calculations. For instance the intersection of a tangential plane in a point of a sweep surface with the surface results in curves that cannot be geometrically correct. We have to investigate that further. Additional research into problems with boundary representations might be necessary. This also leads us to the problem of continuity (described in section 2.4).

The Continuity Problem in 3D

Kortenkamp [79] solved the continuity problem by developing the theory of complex tracing. It can also be applied to 3D geometry. All geometric algorithms that operate in complex space would have to be implemented from scratch though. As he notes himself the complexity of geometric operations in 3D explodes. “If we just want to be able to intersect linear and quadratic objects, we will have an enormous amount of special objects that occur as the intersection of quadrics.” This is just for linear intersections with quadrics. If we think of intersecting a B-Spline curve with a NURBS surface as described in section 4.2 things are getting much more complex. We agree with Kortenkamp that development of a 3D software that handles ambiguities correct will take many years and a lot of manpower. However we think that in order to model projective space correctly this is absolutely necessary. Such a goal is worthwhile to pursue not only for educational purposes but for many more application areas (parametric CAD, spatial kinematics,...). Maybe we can contribute to it in future research projects.

Desktop Interface

Various discussions with teachers at our second evaluation (section 8.3) revealed that there is a general wish for a basic desktop interface of Construct3D that can be directly used in schools without additional hardware requirements. We also noticed that students can collaborate more efficiently with colleagues wearing HMDs if they can contribute to the construction by working on their desktop machines with

a basic interface. We plan to provide students and teachers with such a basic desktop interface. Many advantages get lost when reducing Construct3D to a desktop application though it seems to be the most realistic way of incorporating dynamic 3D geometry into schools today. Currently we are in the planning phase of the desktop interface and think about ways how to do 3D input most efficiently and intuitively without the requirement of additional expensive hardware.

Upcoming Evaluations

A final evaluation will be conducted in the Lab@Future EU project within this year. It will serve us as an additional usability study since we will test the following improvements:

As noted in section 6.2 the helmets used in the current AR lab setup are too heavy and not flexible enough. We will try to reduce negative side effects by replacing current helmets with light weight biker helmets which are easy to adjust to various head sizes.

We will restructure the menu system (section 6.4) of Construct3D to provide a better overview of the system's features.

As discussed in section 8.3 scalability of our distributed system has to be improved to be able to distribute Construct3D to a larger group of PCs. Therefore we will work on scalability and will improve robustness and reliability of distributed Construct3D.

The final Lab@Future evaluation will also serve as a preparatory study for an extensive psychological evaluation of Construct3D. As outlined in section 8.4 we will conduct a psychological evaluation investigating the effects of training with Construct3D on learners' spatial abilities. The study will involve 250 students in 5 different training groups and will be conducted within the scope of a 2-year national research project in cooperation with psychologists from the Institute of Psychology at University Vienna.

Construct3D is in a robust and mature state. In this thesis we demonstrate its applicability to a wide range of application areas and show content which can be developed with a user-friendly interface very quickly. We notice a growing interest of students and teachers in our application and see first signs of an integration of Construct3D into regular geometry courses. We are excited about the promising outlook and hope to be able to contribute to the way three-dimensional geometry is taught to future generations of learners.

Appendix A: Second Evaluation Questionnaire

We used the ISONORM questionnaire by Prümper [120] as a usability questionnaire and adapted it to our needs as described in section 8.2. We added questions to evaluate users' likes and dislikes (affective parameters), the orientation aspect as specified by Roussos [134] and pedagogical aspects.

All participants in our second evaluation were Austrian high school students therefore our original questionnaire is presented here in German.

Anweisung

Im Folgenden geht es um die Beurteilung von Softwaresystemen auf Grundlage der Internationalen Norm ISO 9241/10.

Das Ziel dieser Beurteilung ist es, Schwachstellen bei Softwaresystemen aufzudecken und konkrete Verbesserungsvorschläge zu entwickeln.

Um dies zu bewerkstelligen, ist Ihr Urteil als Kenner des Softwaresystems von entscheidender Bedeutung! Grundlage Ihrer Bewertung sind Ihre individuellen Erfahrungen mit dem Software-Programm, das Sie beurteilen möchten.

Dabei geht es nicht um eine Beurteilung Ihrer Person, sondern um Ihre persönliche Bewertung der Software mit der Sie arbeiten.

Falls Sie über manche Eigenschaften der Software nicht Bescheid wissen und daher Fragen nicht mit Sicherheit beantworten können, fragen Sie bitte noch nach oder lassen Sie das Feld leer.

Die Beurteilung bezieht sich nur auf die Arbeit im Labor (4. Stock) mit

**Construct3D - Unterrichtssoftware zur Konstruktion
dreidimensionaler dynamischer Geometrie in Augmented Reality**

Noch ein Hinweis zur Beantwortung des Beurteilungsbogens:

Beispiel Nr.1:

Die Software ...	---	--	-	-/+	+	++	+++	Die Software ...
ist schlecht.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	ist gut.

Im ersten Beispiel wird danach gefragt, wie gut, bzw. wie schlecht die Software ist.

Der Benutzer beurteilt in diesem Fall die Software zwar als gut, sieht jedoch noch Verbesserungsmöglichkeiten.

Beispiel Nr.2:

Die Software ...	---	--	-	-/+	+	++	+++	Die Software ...
ist langsam.	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	ist schnell.

Im zweiten Beispiel beurteilt der Benutzer die Software als ziemlich langsam.

Füllen Sie bitte den Beurteilungsbogen äußerst sorgfältig aus und lassen Sie keine der Fragen aus!

Die Auswertung der Daten erfolgt anonym.

Aufgabenangemessenheit

Unterstützt die Software die Erledigung Ihrer Lernaufgabe, ohne Sie als Benutzer unnötig zu belasten?

<i>Die Software ...</i>	---	--	-	-/+	+	++	+++	<i>Die Software ...</i>
ist kompliziert zu bedienen.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	ist unkompliziert zu bedienen.
arbeitet langsam.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	arbeitet schnell.
bietet nicht alle Funktionen, um die anfallenden Aufgaben effizient zu bewältigen.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	bietet alle Funktionen, um die anfallenden Aufgaben effizient zu bewältigen.
bietet schlechte Möglichkeiten, sich häufig wiederholende Bearbeitungsvorgänge zu automatisieren.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	bietet gute Möglichkeiten, sich häufig wiederholende Bearbeitungsvorgänge zu automatisieren.
erfordert überflüssige Eingaben.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	erfordert keine überflüssigen Eingaben.
ist schlecht auf die Anforderungen der Lernaufgabe zugeschnitten.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	ist gut auf die Anforderungen der Lernaufgabe zugeschnitten.

Selbstbeschreibungsfähigkeit

Gibt Ihnen die Software genügend Erläuterungen und ist sie in ausreichendem Maße verständlich?

<i>Die Software ...</i>	---	--	-	-/+	+	++	+++	<i>Die Software ...</i>
bietet einen schlechten Überblick über ihr Funktionsangebot.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	bietet einen guten Überblick über ihr Funktionsangebot.
verwendet schlecht verständliche Begriffe, Bezeichnungen, Abkürzungen oder Symbole in Menüs.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	verwendet gut verständliche Begriffe, Bezeichnungen, Abkürzungen oder Symbole in Menüs.
liefert in unzureichendem Maße Informationen darüber, welche Eingaben zulässig oder nötig sind.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	liefert in zureichendem Maße Informationen darüber, welche Eingaben zulässig oder nötig sind.

Steuerbarkeit

Können Sie als Benutzer die Art und Weise, wie Sie mit der Software arbeiten, beeinflussen?

<i>Die Software ...</i>	---	--	-	-/+	+	++	+++	<i>Die Software ...</i>
bietet keine Möglichkeit, die Lernaufgabe an jedem Punkt zu unterbrechen und dort später ohne Verluste wieder weiterzumachen.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	bietet die Möglichkeit, die Lernaufgabe an jedem Punkt zu unterbrechen und dort später ohne Verluste wieder weiterzumachen.
erzwingt eine unnötig starre Einhaltung von Bearbeitungsschritten.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	erzwingt keine unnötig starre Einhaltung von Bearbeitungsschritten.
ermöglicht keinen leichten Wechsel zwischen einzelnen Menüs.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	ermöglicht einen leichten Wechsel zwischen einzelnen Menüs.
ist so gestaltet, daß der Benutzer nicht beeinflussen kann, wie und welche Informationen am Bildschirm dargeboten werden.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	ist so gestaltet, daß der Benutzer beeinflussen kann, wie und welche Informationen am Bildschirm dargeboten werden.
erzwingt unnötige Unterbrechungen der Arbeit.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	erzwingt keine unnötigen Unterbrechungen der Arbeit.

Erwartungskonformität

Kommt die Software durch eine einheitliche und verständliche Gestaltung Ihren Erwartungen und Gewohnheiten entgegen?

<i>Die Software ...</i>	---	--	-	-/+	+	++	+++	<i>Die Software ...</i>
erschwert die Orientierung, durch eine uneinheitliche Gestaltung.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	erleichtert die Orientierung, durch eine einheitliche Gestaltung.
lässt einen im Unklaren darüber, ob eine Eingabe erfolgreich war oder nicht.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	lässt einen nicht im Unklaren darüber, ob eine Eingabe erfolgreich war oder nicht.
informiert in unzureichendem Maße über das, was sie gerade macht.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	informiert in ausreichendem Maße über das, was sie gerade macht.
reagiert mit schwer vorhersehbaren Bearbeitungszeiten.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	reagiert mit gut vorhersehbaren Bearbeitungszeiten.
lässt sich nicht durchgehend nach einem einheitlichen Prinzip bedienen.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	lässt sich durchgehend nach einem einheitlichen Prinzip bedienen.

Fehlertoleranz

Bietet Ihnen die Software die Möglichkeit, trotz fehlerhafter Eingaben das beabsichtigte Arbeitsergebnis ohne oder mit geringem Korrekturaufwand zu erreichen?

<i>Die Software ...</i>	---	--	-	-/+	+	++	+++	<i>Die Software ...</i>
ist so gestaltet, dass kleine Fehler schwerwiegende Folgen haben können.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	ist so gestaltet, dass kleine Fehler keine schwerwiegenden Folgen haben können.
informiert zu spät über fehlerhafte Eingaben.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	informiert sofort über fehlerhafte Eingaben.
liefert schlecht verständliche Fehlermeldungen.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	liefert gut verständliche Fehlermeldungen.
erfordert bei Fehlern im Großen und Ganzen einen hohen Korrekturaufwand.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	erfordert bei Fehlern im Großen und Ganzen einen geringen Korrekturaufwand.
gibt keine konkreten Hinweise zur Fehlerbehebung.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	gibt konkrete Hinweise zur Fehlerbehebung.

Lernförderlichkeit

Ist die Software so gestaltet, dass Sie sich ohne großen Aufwand in sie einarbeiten konnten und bietet sie auch dann Unterstützung, wenn Sie neue Funktionen lernen möchten?

<i>Die Software ...</i>	---	--	-	-/+	+	++	+++	<i>Die Software ...</i>
erfordert viel Zeit zum Erlernen.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	erfordert wenig Zeit zum Erlernen.
ermutigt nicht dazu, auch neue Funktionen auszuprobieren.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	ermutigt dazu, auch neue Funktionen auszuprobieren.
erfordert, dass man sich viele Details merken muss.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	erfordert nicht, dass man sich viele Details merken muss.
ist so gestaltet, dass sich einmal Gelerntes schlecht einprägt.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	ist so gestaltet, dass sich einmal Gelerntes gut einprägt.
ist schlecht ohne fremde Hilfe oder Handbuch erlernbar.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	ist gut ohne fremde Hilfe oder Handbuch erlernbar.

Allgemeine Fragen zu Construct3D

Wie sehr hat Ihnen die Arbeit mit Construct3D Spaß gemacht?



5



4



3



2



1

Überhaupt
nicht

Nicht sehr

Naja

Ja

Ja, sehr

Wie gerne würden Sie wieder mit Construct3D arbeiten?



5



4



3



2



1

Überhaupt
nicht

Nicht sehr

Weiß nicht

Ja

Ja, sehr gerne

Wenn „Nicht sehr“ oder „Überhaupt nicht“ – Warum nicht?

**Haben Sie bei der Arbeit mit Construct3D neue geometrische Inhalte gelernt?
(Antwort bitte einringeln)**

überhaupt **sehr viel**
nichts

---	--	-	-/+	+	++	+++
-----	----	---	-----	---	----	-----

Wenn Ja (+, ++, +++), welche?

Haben Sie geometrische Zusammenhänge oder geometrischen Lehrstoff, den Sie bereits im Unterricht gelernt haben besser verstanden? (Antwort bitte einringeln)

überhaupt **sehr viel**
nichts

---	--	-	-/+	+	++	+++
-----	----	---	-----	---	----	-----

Wenn Ja(+, ++, +++), was denken Sie ist der Grund dafür?

Inwiefern denken Sie, dass man grundsätzlich in Augmented Reality Geometrie lernen könnte?

gar nicht **sehr gut**

---	--	-	-/+	+	++	+++
-----	----	---	-----	---	----	-----

Haben Sie bei oder nach der längeren Arbeit im Labor mit Construct3D ein Schwindelgefühl oder Kopfweh oder andere unangenehme Dinge gespürt?

☐ Ja

☐ Nein

Wenn Ja, wie lange haben Sie ungefähr damit gearbeitet und welche „Nebenwirkungen“ haben Sie gespürt?

Haben Sie sich in der virtuellen Welt orientierungslos gefühlt?

- ☐ Ja
- ☐ Nein

Wobei haben Sie das dreidimensionale Modell / die Geometrie besser verstanden und besser begreifen können?

- | | |
|-----------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------|
| <input type="radio"/> beim Betrachten der verschiedene Ansichten der 3-dimensionalen Welt am Bildschirm (im 5. Stock) | <input type="radio"/> beim Betrachten des Modells im Labor mit der Datenbrille |
|-----------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------|

Was erschien Ihnen beim ersten Versuch einfacher:

- ☐ Konstruieren am Bildschirm mit einem CAD Paket?
Wenn ja, mit welchem?
- ☐ Konstruieren im Labor mit Construct3D?

Könnten Sie sich vorstellen mit Construct3D zu konstruieren ohne jemals zuvor mit einem ‘herkömmlichen’ CAD Paket (CAD3D, Microstation, ...) am Bildschirm konstruiert zu haben?

- ☐ Ja
- ☐ Nein

Könnten Sie sich vorstellen mit Construct3D zu konstruieren ohne jemals das Konstruieren in Parallelrissen (Grundriss/Aufriss) erlernt zu haben?

- ☐ Ja
- ☐ Nein

Sehen Sie dabei Probleme?

Könnten Sie sich einen Einsatz der aktuellen Version von Construct3D in der Ausbildung (universitärer bzw. schulischer Einsatz) im Geometrie- oder Mathematikunterricht vorstellen wenn die Kosten für die Ausrüstung gering wären?

☐ Ja ☐ Nein

Wenn Ja welche Anwendungsgebiete fallen Ihnen ein?

Wenn Nein warum nicht? Verbesserungsvorschläge?

Sehen Sie außer den Kosten andere Schwierigkeiten, die einen Einsatz von Construct3D im Unterricht verhindern?

Würden Sie lieber alleine, mit einem Kollegen oder nur mit dem Lehrer arbeiten oder wie gehabt mit einem Schulkollegen + Lehrer. (bitte Zutreffendes unterstreichen)

Waren die Aufgaben, die gelöst werden mussten

Zu schwer Zu leicht

---	--	-	-/+	+	++	+++
-----	----	---	-----	---	----	-----

Wie sehr waren Sie vor den Tests motiviert mitzumachen?

Gar nicht sehr

---	--	-	-/+	+	++	+++
-----	----	---	-----	---	----	-----

Hat sich die Motivation/Begeisterung im Laufe der Tests gesteigert oder gesenkt? (Zutreffendes bitte unterstreichen)

Wie war für sie das Drumherum – Organisation und zeitliche Einteilung in der Schule - war der zeitliche Aufwand zum normalen Schulbesuch recht groß? Würden Sie wieder mitmachen? Wie oft oder in welchem Abstand könnten Sie sich vorstellen wieder zu Übungseinheiten zu kommen? War die Entschädigung angemessen? (Bitte freie Antworten was auch immer Sie dazu sagen möchten)

Was hat Ihnen am besten gefallen?

Was hat Ihnen am wenigsten gefallen?

Was muss man Ihrer Meinung nach als Vorwissen haben – was sollte man „können“ bevor man mit dem Programm gut arbeiten kann:

Falls es bisher im Fragebogen noch nicht genannt wurde, was sollte Ihrer Meinung nach verbessert werden?

Freie Kommentare:

Zum Schluss

Zum Schluss bitten wir Sie, noch folgende Fragen zu beantworten:

Seit wie vielen Jahren arbeiten Sie überhaupt schon mit Computern?	Jahre
Wie viele Stunden arbeiten Sie pro Woche durchschnittlich mit Computern?	Stunden
Wie viele Stunden haben Sie insgesamt mit Construct3D im Labor gearbeitet?	Stunden
Wie gut beherrschen Sie die beurteilte Software?	sehr <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> sehr schlecht gut	
Wie gut schätzen Sie Ihre Kenntnisse in DG / ACG ein?	sehr <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> <input type="radio"/> sehr schlecht gut	
Mit welchen PC-Programmen haben Sie schon gearbeitet? (bitte die Wichtigsten/Häufigsten namentlich auflisten)		
Davon:	Geometrie- Unterrichts- Programme (z.B. CAD3D) CAD Programme z.B. Microstation, 3DStudio, Maya

Wie alt sind Sie?	Jahre
Ihr Geschlecht?	m/w

VIELEN DANK FÜR DIE MITHILFE!!!

Bibliography

- [1] "CEEB Special Aptitude Test in Spatial Relations (MCT)," College Entrance Examination Board, USA 1939.
- [2] "Lehrplan der AHS-Oberstufe," in *Österreichisches Bundesministerium für Bildung, Wissenschaft und Kultur*: <http://www.bmbwk.gv.at/>, 1989.
- [3] "OpenCascade 4.0. Open-Source Toolkit for 3D modelling," <http://www.opencascade.com>, 2001.
- [4] D. Allison, B. Wills, D. Bowman, J. Wineman, and L. F. Hodges, "The Virtual Reality Gorilla Exhibit," *IEEE Computer Graphics and Applications*, vol. 17, pp. 30-38, 1997.
- [5] R. Azuma, "A Survey of Augmented Reality," *PRESENCE: Teleoperators and Virtual Environments*, vol. 6, pp. 355-385, 1997.
- [6] M. Bajura, H. Fuchs, and R. Ohbuchi, "Merging Virtual Objects with the Real World: Seeing Ultrasound Imagery Within the Patient," in *Proceedings of SIGGRAPH '92*. New York: ACM Press, 1992, pp. 203-210.
- [7] P. Baudisch, "The Cage: Efficient Construction in 3D using a Cubic Adaptive Grid," in *Proceedings of the 9th ACM Symposium on User Interface Software and Technology (UIST '96)*, 1996, pp. 171-172.
- [8] J. T. Bell and H. S. Fogler, "The Investigation and Application of Virtual Reality as an Educational Tool," *American Society for Engineering Education 1995 Annual Conference*, 1995.
- [9] M. Beutnagel, A. Conkie, J. Schroeter, Y. Stylianou, and A. K. Syrdal, "The AT&T Next-Gen TTS system," Berlin, Germany, Paper SASCA_4 1999.
- [10] M. Billinghurst, S. Baldis, E. Miller, and S. Weghorst, "Shared Space: Collaborative Information Spaces," in *Proceedings of the 7th International Conference on Human-Computer Interaction (HCI '97)*. San Francisco, USA, 1997.
- [11] M. Billinghurst, J. Bowskill, J. Morphett, and M. Jessop, "A wearable spatial conferencing space," in *Proceedings of ISWC '98*. Pittsburgh, Penn., USA, 1998, pp. 19-20.
- [12] M. Billinghurst and H. Kato, "Real world teleconferencing," in *Proceedings of CHI '99*. Pittsburgh, PA, USA, 1999.

- [13] 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, 2001.
- [14] M. Billinghurst, S. Weghorst, and T. Furness III, "Wearable Computers for Three Dimensional CSCW," Human Interface Technology Laboratory, University of Washington R-97-10, 1997.
- [15] D. Bowman, E. Kruijff, J. J. LaViola Jr., and I. Poupyrev, *3D User Interfaces: Theory and Practice*: Addison/Wesley, To be published 2004.
- [16] D. A. Bowman, "Conceptual Design Space - Beyond Walk-through to Immersive Design," in *Designing Digital Space*, D. Bertol, Ed. New York: John Wiley & Sons, 1996.
- [17] D. A. Bowman, "Interaction Techniques for Common Tasks in Immersive Virtual Environments: Design, Evaluation, and Application," in *Ph.D. thesis, Virginia Polytechnic & State University*, 1999.
- [18] H. Brauner, *Lehrbuch der konstruktiven Geometrie*. Wien: Springer, 1986.
- [19] C. Breiteneder, H. Kaufmann, D. Schmalstieg, and J. Glück, "Educating Spatial Intelligence with Augmented Reality," Vienna University of Technology & Vienna University, Vienna, Project Application to the Austrian Science Fund (FWF) 2003.
- [20] M. Bricken and C. Byrne, "Summer Students in Virtual Reality: A Pilot Study on Educational Applications of VR Technology," in *Virtual Reality, Applications and Explorations*, A. Wexelblat, Ed. Cambridge, MA: Academic Press Professional, 1993.
- [21] 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.
- [22] L. N. Brown, C. J. Lahar, and J. L. Mosley, "Age and gender-related differences in strategy use for route information: A "map-present" direction-giving paradigm," *Environment and Behavior*, vol. 30, pp. 123-143, 1998.
- [23] J. Butterworth, A. Davidson, S. Hench, and T. M. Olano, "3DM: A Three Dimensional Modeler Using a Head-Mounted Display," *Computer Graphics. Proceedings 1992 Symposium on Interactive 3D Graphics*, vol. 25, pp. 135-138, 1992.
- [24] A. Butz, T. Höllerer, S. Feiner, B. MacIntyre, and C. Beshers, "Enveloping Users and Computers in a Collaborative 3D Augmented Reality," in *Proceedings of the 2nd IEEE and ACM International Workshop on Augmented Reality '99 (IWAR '99)*. San Francisco, CA, 1999, pp. 35-44.
- [25] C. Byrne, "Water on Tap: The Use of Virtual Reality as an Educational Tool," University of Washington, College of Engineering, Washington 1996.

- [26] J. B. Carroll, *Human cognitive abilities. A survey of factor-analytic studies*. Cambridge, UK: Cambridge University Press, 1993.
- [27] T. P. Caudell and D. W. Mizell, "Augmented Reality: an application of heads-up display technology to manual manufacturing processes," in *Proceedings of the Twenty-Fifth Hawaii International Conference on Systems Science*. Kauai, Hawaii, 1992, pp. 659-669.
- [28] W. L. Chapin, T. A. Lacey, and L. Leifer, "DesignSpace: A Manual Interaction Environment of Computer Aided Design," *Proceedings of the ACM's SIGCHI 1994 Conference: CHI'94 Human Factors In Computing Systems*, 1994.
- [29] D. Chen and C. Sun, "Undoing Any Operation in Collaborative Graphics Editing Systems," in *Proceedings of the 2001 International ACM SIGGROUP Conference on Supporting Group Work*. Boulder, Colorado, USA, 2001, pp. 197-206.
- [30] J. H. Clark, "Designing Surfaces in 3-D," *Communications of the ACM*, vol. 19, pp. 454-460, 1976.
- [31] S. V. G. Cobb, J. R. Wilson, S. Nichols, and A. Ramsey, "Virtual reality-induced symptoms and effects(VRISE)," *PRESENCE - Teleoperators and Virtual Environments*, vol. 8, pp. 169-186, 1999.
- [32] J. R. Corney and T. Lim, *3D Modeling with ACIS*: Saxe-Coburg Publications, 2002.
- [33] J. M. J. Dabbs, E. L. Chang, R. A. Strong, and R. Milun, "Spatial ability, navigation strategy, and geographic knowledge among men and women," *Evolution and Human Behavior*, vol. 19, pp. 89-98, 1998.
- [34] 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.
- [35] J. Diepstraten, D. Weiskopf, and T. Ertl, "Transparency in Interactive Technical Illustrations," *Computer Graphics Forum*, vol. 21, pp. 317-326, 2002.
- [36] 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, 2000.
- [37] J. Eliot, "The Classification of Spatial Tests," in *An International Directory of Spatial Tests*, J. E. I. M. Smith, Ed. Windsor: NFER-Nelson, 1983, pp. 11-15.
- [38] S. Feiner, B. MacIntyre, T. Höllerer, and A. Webster, "A Touring Machine: Prototyping 3D Mobile Augmented Reality Systems for Exploring the Urban Environment," presented at Proceedings of International Symposium on Wearable Computers, 1997.

- [39] 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.
- [40] B. W. Field, "A course in spatial visualisation," in *Proceedings of the 8th International Conference on Engineering Computer Graphics and Descriptive Geometry (ISGG)*. Austin, Texas, USA, 1998.
- [41] S. S. Fisher, M. McGreevy, J. Humphries, and W. Robinett, "Virtual Environment Display Systems," *Proceedings of the Workshop on Interactive 3D Graphics*, 1986.
- [42] E. Frécon and M. Stenius, "DIVE: A scaleable network architecture for distributed virtual environments," *Distributed Systems Engineering Journal (DSEJ)*, vol. 5, pp. 91-100, 1998.
- [43] L. A. Galea and D. Kimura, "Sex differences in route-learning," *Personality and Individual Differences*, vol. 14, pp. 53-65, 1993.
- [44] A. C. Gibbs and J. F. Wilson, "Sex differences in route learning by children," *Perceptual and Motor Skills*, vol. 88, pp. 590-594, 1999.
- [45] 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.
- [46] J. Glück, "Spatial strategies: Kognitive Strategien bei Raumvorstellungsleistungen [Spatial strategies -Strategy use in spatial cognition]," University of Vienna, Ph.D. Thesis 1999.
- [47] J. Glück and S. Fitting, "Spatial strategy selection: Interesting incremental information," *International Journal of Testing*, vol. in press, 2003.
- [48] J. Glück, R. Machat, M. Jirasko, and B. Rollett, "Training-related changes in solution strategy in a spatial test: An application of item response models," *Learning and Individual Differences*, vol. 13, 2002.
- [49] R. Gorska, C. Leopold, S. Sorby, and K. Shiina, "International comparisons of gender differences in spatial visualization and the effect of graphics instruction on the development of these skills," in *Proceedings of the 8th International Conference on Engineering Computer Graphics and Descriptive Geometry (ISGG)*. Austin, Texas, USA, 1998.
- [50] H. Graf, M. Koch, and A. Stork, "Cyberstilo, Towards an Ergonomic and Aesthetic Wireless 3D-Pen," in *IEEE VR 2004 Proceedings; Workshop "Beyond Wand and Glove Based Interaction"*. Chicago, IL, 2004.
- [51] S. Grund, G. Grote, and L. Windlinger, "CIELT: Concept and Instruments for Evaluation of Learning Tools," Institut für Arbeitspsychologie ETH, Zürich, Report 2003.
- [52] S. Grund, L. Windlinger, and G. Grote, "Concept for Interdisciplinary Evaluation of Learning Tools (CIELT)," in *4th International Conference on New Educational Environments*, F. F., C. Lutz, P. Schulz, and L. Cantoni, Eds. Lugano: Manno, 2002, pp. 2.4 11-12.14 14.

- [53] R. B. Guay, "Purdue Spatial Visualization Test: Rotations," Purdue Research Foundation, West Lafayette, IN 1977.
- [54] 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.
- [55] R. E. Guttman, E. Epstein, M. Amir, and L. Guttman, "A structural theory of spatial abilities," *Applied Psychological Measurement*, vol. 14, pp. 217-236, 1990.
- [56] R. E. Guttman and I. Shoham, "The structure of spatial ability items: A faceted analysis," *Perceptual and Motor Skills*, vol. 54, pp. 487-493, 1982.
- [57] G. Hesina, "Distributed collaborative augmented reality." Wien, 2001, pp. 102 Bl.
- [58] L. J. Hettinger and G. E. Riccio, "Visually induced motion sickness in virtual environments," *PRESENCE - Teleoperators and Virtual Environments*, vol. 1, pp. 319-328, 1992.
- [59] K. J. Hill and P. A. Howarth, "Habituation to the Side Effects of Immersion in a Virtual Environment," *Displays*, vol. 21, pp. 25-30, 2000.
- [60] C. M. Hoffmann, "How solid is solid modeling?," in *Applied Computational Geometry*, vol. LNCS State-of-the-Art-Survey, M. Lin and D. Manocha, Eds.: Springer Verlag, 1996, pp. 1-8.
- [61] C. M. Hoffmann, "Solid Modeling," in *CRC Handbook on Discrete and Computational Geometry*, J. E. Goodman and J. O'Rourke, Eds. Boca Raton, FL: CRC Press, 1997, pp. 863-880.
- [62] T. Höllerer, S. Feiner, and J. Pavlik, "Situated Documentaries: Embedding Multimedia Presentations in the Real World," in *Proc. ISWC '99 (Third Int. Symp. on Wearable Computers)*. San Francisco, CA, 1999, pp. 79-86.
- [63] 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, pp. 779-785, 1999.
- [64] P. A. Howarth and P. J. Costello, "Studies into the Visual Effects of Immersion in Virtual Environments," VISERG Report 9603 1996.
- [65] P. A. Howarth and P. J. Costello, "The Occurrence of Virtual Simulation Sickness Symptoms when an HMD was used as a Personal Viewing System," *Displays*, vol. 18, pp. 107-116, 1997.
- [66] M. Husty, "Editorial," *IBDG - Informationsblätter der Geometrie*, vol. 22, pp. 1, 2003.
- [67] N. Jackiw, "The Geometer's Sketchpad Version 3." Berkeley: Key Curriculum Press, 1995.

- [68] W. Jank, "Räumliche Deutungen," Institute of Geometry, Wien, Manuscript.
- [69] W. Jank, "Räumliche Deutung von Bahnen ebener Bewegungen," *Journal of Geometry*, vol. 50, pp. 95-99, 1994.
- [70] M. Jilka and A. K. Syrdal, "The AT&T German text-to-speech system: realistic linguistic description," in *Proceedings of ICSLP 2002*. Denver, Colorado, 2002.
- [71] M. A. Just and P. A. Carpenter, "Cognitive coordinate systems: Accounts of mental rotation and individual differences in spatial ability," *Psychological Review*, vol. 92, pp. 137-172, 1985.
- [72] H. Kaufmann, "Dynamische Geometrie in Virtual Reality," *Informationsblätter der Geometrie (IBDG)*, vol. 21, pp. 33-37, 2002.
- [73] H. Kaufmann, "Collaborative Augmented Reality in Education," Monte Carlo, Monaco, position paper for keynote speech at Imagina 2003 conference TR-188-2-2003-01, Feb. 3rd, 2003.
- [74] H. Kaufmann and D. Schmalstieg, "Mathematics and geometry education with collaborative augmented reality," *Computers & Graphics*, vol. 27, pp. 339-345, 2003.
- [75] H. Kaufmann, D. Schmalstieg, and M. Wagner, "Construct3D: a virtual reality application for mathematics and geometry education," *Education and Information Technologies*, vol. 5, pp. 263-276, 2000.
- [76] R. S. Kennedy and K. M. Stanney, "Aftereffects from virtual environment exposure: How long do they last?," in *Proceeding of the 42nd annual meeting of the Human Factors And Ergonomics society*, 1998, pp. 1476-1480.
- [77] J. R. Kirby and D. R. Boulter, "Training räumlicher Fähigkeiten durch abbildende Geometrie (Training of spatial abilities through transformational geometry)," *Zeitschrift für Pädagogische Psychologie*, vol. 2, pp. 146-155, 1998.
- [78] K. Kiyokawa, H. Takemura, and N. Yokoya, "SeamlessDesign for 3D object creation," *IEEE Multimedia*, vol. 7, pp. 22-33, 2000.
- [79] U. H. Kortenkamp, "Foundations of Dynamic Geometry," Swiss Federal Institute of Technology, Zürich, Switzerland, PhD Dissertation 1999.
- [80] U. Kretschmer, V. Coors, U. Spierling, D. Grasbon, K. Schneider, I. Rojas, and R. Malaka, "Meeting the spririt of history," in *Proceedings of VAST 2001*. Glyfada, Athens, Greece, 2001.
- [81] E. Kruijff, "An overview of virtual reality supported conceptual design tools," <http://viswiz.imk.fraunhofer.de/~kruijff/cvde.html>, 2003.
- [82] P. C. Kyllonen, D. F. Lohman, and R. E. Snow, "Effects of aptitude, strategy training, and task facets on spatial test performance," *Journal of Educational Psychology*, vol. 76, pp. 130-145, 1984.

- [83] 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*, vol. 121, C. Keitel and K. Ruthven, Eds. Berlin: Springer, 1993, pp. 48-67.
- [84] C. Laborde, "Designing tasks for Learning Geometry in a computer based environment," in *Technology in Mathematics Teaching - a bridge between teaching and learning*, L. Burton and B. Jaworski, Eds. London: Chartwell-Bratt, 1995, pp. 35-68.
- [85] M. Lanca, "Three-dimensional representations of contour maps," *Contemporary Educational Psychology*, vol. 23, pp. 22-41, 1998.
- [86] P. Larson, A. A. Rizzo, J. G. Buckwalter, A. Van Rooyen, K. Kratz, U. Neumann, C. Kesselman, M. Thiebaut, and C. Van Der Zaag, "Gender issues in the use of virtual environments," *Cyberpsychology and Behavior*, vol. 2, pp. 113-123, 1999.
- [87] J. J. LaViola Jr., "A discussion of cybersickness in virtual environments," *ACM SIGCHI Bulletin*, vol. 32, pp. 47-56, 2000.
- [88] C. A. Lawton, "Gender differences in way-finding strategies: Relationship to spatial ability and spatial strategies," *Sex Roles*, vol. 30, pp. 765-779, 1994.
- [89] C. A. Lawton, "Strategies for indoor wayfinding: The role of orientation," *Journal of Environmental Psychology*, vol. 16, pp. 137-145, 1996.
- [90] C. A. Lawton and K. A. Morrin, "Gender differences in pointing accuracy in computer-simulated 3D mazes," *Sex Roles*, vol. 40, pp. 73-92, 1999.
- [91] F. Ledermann and D. Schmalstieg, "Presenting an Archaeological Site in the Virtual Showcase," in *Proceedings of the 4th International Symposium on Virtual Reality, Archeology, and Intelligent Cultural Heritage (VAST 2003)*. Brighton, UK, 2003, pp. 119-126.
- [92] C. Leopold, R. A. Gorska, and S. A. Sorby, "International Experiences in Developing the Spatial Visualization Abilities of Engineering Students," in *Proceedings of the 9th International Conference on Geometry and Graphics*. Johannesburg, South Africa, 2000, pp. 81-91.
- [93] C. Leopold, R. A. Gorska, and S. A. Sorby, "International Experiences in Developing the Spatial Visualization Abilities of Engineering Students," *Journal for Geometry and Graphics*, vol. 5, pp. 81-91, 2001.
- [94] C. Leopold, S. A. Sorby, and R. Gorska, "Gender differences in 3-D visualization skills of engineering students," in *Proceedings of the 7th International Conference on Engineering Computer Graphics and Descriptive Geometry*. Cracow, Poland, 1996, pp. 560-564.
- [95] J. Liang and M. Green, "JDCAD: a highly interactive 3D modeling system," *Computers & Graphics*, vol. 18, pp. 499-506, 1994.

- [96] M. C. Linn and A. C. Petersen, "Emergence and characterization of sex differences in spatial ability: A meta-analysis," *Child Development*, vol. 56, pp. 1479-1498, 1985.
- [97] B. Loftin, M. Engelberg, and R. Benedetti, "Applying virtual reality in education: A prototypical virtual physics laboratory," in *Proceedings of the IEEE 1993 Symposium on Research Frontiers in Virtual Reality*. Los Alamitos, CA: IEEE Computer Society Press, 1993, pp. 67-74.
- [98] D. F. Lohman, "Spatial ability: A review and reanalysis of the correlational literature," Stanford University School of Education, Aptitude Research Project, Stanford, CA Tech. Rep. No. 8, 1979.
- [99] 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.
- [100] D. F. Lohman and P. C. Kyllonen, "Individual differences in solution strategy on spatial tasks," in *Individual differences in cognition*, D. F. Dillon and R. R. Schmeck, Eds. New York: Academic Press, 1983, pp. 105-135.
- [101] B. MacIntyre and S. Feiner, "A distributed 3D graphics library," in *Proceedings of ACM SIGGRAPH '98*. Orlando, Florida, USA, 1998, pp. 361-370.
- [102] N. J. Mackintosh, *IQ and human intelligence*. Oxford, UK: Oxford University Press, 1998.
- [103] 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.
- [104] M. E. McCauley and T. J. Sharkey, "Cybersickness: perception of self-motion in virtual environments," *Presence: Teleoperators and Virtual Environments*, vol. 1, pp. 311-318, 1992.
- [105] M. K. McGee, "Assessing Negative Side Effects in Virtual Environments," Virginia Polytechnic Institute and State University, Blacksburg, Virginia, Master Thesis 1998.
- [106] R. Mechling, "EUKLID DynaGeo von innen - ein Blick hinter die Kulissen," <http://www.dynageo.de>, 2000.
- [107] 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.
- [108] L. K. Miller and V. Santoni, "Sex differences in spatial abilities: Strategic and experiential correlates," *Acta Psychologica*, vol. 62, pp. 225-235, 1986.
- [109] M. Mine, "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.

- [110] M. R. Mine, "ISAAC: a meta-CAD system for virtual environments," *Computer Aided Design*, vol. 29, pp. 547-553, 1997.
- [111] S. D. Moffat, E. Hampson, and M. Hatzipantelis, "Navigation in a "virtual" maze: Sex differences and correlation with psychometric measures of spatial ability in humans," *Evolution and Human Behavior*, vol. 19, pp. 73-87, 1998.
- [112] E. Müller and J. L. Krames, *Vorlesungen über Darstellende Geometrie, II. Band: Die Zyklographie*. Wien: Franz Deuticke, 1929.
- [113] K. Omura, S. Shiwa, and F. Kishino, "3D Display with Accomodative Compensation (3DDAC) Employing Real-Time Gaze Detection," *Society for Information Display Digest*, pp. 889-892, 1996.
- [114] R. Pausch, T. Crea, and M. J. Conway, "A Literature Survey for Virtual Environments: Military Flight Simulators Visual Systems and Simulator Sickness," *PRESENCE - Teleoperators and Virtual Environments*, vol. 1, pp. 344-363, 1992.
- [115] W. Piekarski and B. H. Thomas, "The tinmith system - demonstrating new techniques for mobile augmented reality modelling," in *Proceedings of AUIC2002*. Melbourne, Vic, Australia, 2002.
- [116] J. S. Pierce, A. Forsberg, M. J. Conway, S. Hong, R. Zeleznik, and M. Mine, "Image Plane Interaction Techniques in 3D Immersive Environments," 1997.
- [117] E. Podenstorfer, "GAM," <http://www.gam3d.at/>, 2003.
- [118] A. Prakash and M. J. Knister, "A framework for undoing actions in collaborative systems," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 1, pp. 295-330, 1994.
- [119] J. Prümper, "Der Benutzungsfragebogen ISONORM 9241/10: Ergebnisse Zur Reliabilität und Validität," in *Software-Ergonomie '97*, R. Liskowsky, Ed. Stuttgart, 1997.
- [120] J. Prümper and M. Anft, "Die Evaluation von Software auf Grundlage des Entwurfs zur internationalen Ergonomie-Norm ISO 9241 Teil 10 als Beitrag zur partizipativen Systemgestaltung - ein Fallbeispiel," in *Software-Ergonomie '93: Von der Benutzungsoberfläche zur Arbeitsgestaltung*, K. H. Rödiger, Ed. Stuttgart: Teubner, 1993, pp. 145-156.
- [121] E. C. Regan, "An investigation into nausea and other side-effects of head-coupled immersive virtual reality," *Virtual Reality*, vol. 1, pp. 17-32, 1995.
- [122] H. Regenbrecht, E. Kruijff, D. Donath, H. Seichter, and J. Beetz, "VRAM - a Virtual Reality Aided Modeler," in *Proceedings of eCAADe2000*. Weimar/Germany, 2000.
- [123] G. Reitmayr, "On Software Design for Augmented Reality," Vienna University of Technology, Vienna, Austria, PhD Dissertation 2004.

- [124] G. Reitmayr and D. Schmalstieg, "Mobile collaborative augmented reality," in *IEEE and Acm International Symposium on Augmented Reality, Proceedings*. Los Alamitos: IEEE COMPUTER SOC, 2001, pp. 114-123.
- [125] 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.
- [126] G. Reitmayr and D. Schmalstieg, "A Wearable 3D Augmented Reality Workspace," in *Proceedings of ISCW 2001*. Zurich, Switzerland, 2001.
- [127] J. Richter-Gebert and U. H. Kortenkamp, "The Interactive Geometry Software Cinderella: Version 1.2 (Interactive Geometry on Computers)," 1999.
- [128] G. Riva and C. Galimberti, "Complementary Explorative Multilevel Data Analysis," in *Towards Cyberpsychology*, G. Riva and C. Galimberti, Eds. Amsterdam: IOS Press, 2001.
- [129] 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.
- [130] H. Rose, "Assessing Learning in VR: Towards Developing a Paradigm Virtual Reality in Roving Vehicles," University of Washington, Human Interface Technology Laboratory, HITL Report No. R-95-1 1995.
- [131] H. Rose and M. Billinghamurst, "Zengo Sayu: An Immersive Educational Environment for Learning Japanese," University of Washington, Human Interface Technology Laboratory, Report No. r-95-4 1995.
- [132] R. Rosenthal, "The 'File Drawer Problem' and Tolerance for Null Results," *Psychological Bulletin*, vol. 86, pp. 638-641, 1979.
- [133] M. Roussos, A. Johnson, J. Leigh, C. Vasilakis, C. Barnes, and T. Moher, "NICE: Combining Constructionism, Narrative, and Collaboration in a Virtual Learning Environment," *Computer Graphics*, vol. 31, pp. 62-63, 1997.
- [134] M. Roussos, A. Johnson, T. Moher, J. Leigh, C. Vasilakis, and C. Barnes, "Learning and Building Together in an Immersive Virtual World," *PRESENCE - Teleoperators and Virtual Environments*, vol. 8, pp. 247-263, 1999.
- [135] T. Saito, K. Suzuki, and T. Jingu, "Relations between spatial ability evaluated by a mental cutting test and engineering graphics education," in *Proceedings of the 8th International Conference on Engineering Computer Graphics and Descriptive Geometry (ISGG)*. Austin, Texas, USA, 1998.
- [136] P. Santos, H. Graf, T. Fleisch, and A. Stork, "3D Interactive Augmented Reality in Early Stages of Product Design," *HCI International 2003, 10th Conference on Human - Computer Interaction*, pp. 1203-1207, 2003.

- [137] D. Schmalstieg, A. Fuhrmann, and H. G., "Bridging Multiple User Interface Dimensions with Augmented Reality," *Proceedings of ISAR 2000*, pp. 159-164, 2000.
- [138] 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, 2002.
- [139] D. Schmalstieg, A. Fuhrmann, Z. Szalavári, and M. Gervautz, ""Studierstube" - An Environment for Collaboration in Augmented Reality," in *Proceedings of Collaborative Virtual Environments '96*. Nottingham, UK, 1996.
- [140] D. Schmalstieg, H. Kaufmann, G. Reitmayr, and F. Ledermann, "Geometry Education in the Augmented Classroom," Reviewed scientific demonstration at the IEEE and ACM International Symposium on Mixed and Augmented Reality 2002 TR-188-2-2002-14, 2002.
- [141] S. Schmitz, "Gender-related strategies in environmental development: Effects of anxiety on wayfinding in and representation of a three-dimensional maze," *Journal of Environmental Psychology*, vol. 17, pp. 215-228, 1997.
- [142] S. Schmitz, "Gender differences in aquisition of environmental knowledge related to wayfinding behavior, spatial anxiety and self-estimated environmental competencies," *Sex Roles*, vol. 41, pp. 71-93, 1999.
- [143] S. A. Sorby and B. J. Baartmans, "A longitudinal study of a pre-graphics course designed to improve the 3-D spatial skills of low visualizers," in *Proceedings of the 8th International Conference on Engineering Computer Graphics and Descriptive Geometry (ISGG)*. Austin, Texas, USA, 1998.
- [144] S. A. Sorby and R. A. Gorska, "The effect of various courses and teaching methods on the improvement of spatial ability," in *Proceedings of the 8th International Conference on Engineering Computer Graphics and Descriptive Geometry (ISGG)*. Austin, Texas, USA, 1998.
- [145] E. Souvignier, "Training räumlicher Fähigkeiten. [Training spatial abilities.]," in *Handbuch Kognitives Training*, K. J. Klauer, Ed. Göttingen: Hogrefe, 2001, pp. 293-319.
- [146] H. Stachel, J. Wallner, and M. Pfeifer, "CAD-3D für Windows," <http://www.geometrie.tuwien.ac.at/software/cad3d/>, 2003.
- [147] K. M. Stanney and R. S. Kennedy, "The psychometrics of cybersickness," *Communications of the ACM*, vol. 40, pp. 66-68, 1997.
- [148] M. Stenius, "Collaborative object modelling in virtual environments," KTH, Stockholm, Sweden, Master Thesis 1996.
- [149] P. Strauss and R. Carey, "An object oriented 3D graphics toolkit," *Proceedings of ACM SIGGRAPH '92*, 1992.

- [150] 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.
- [151] X. Sun and K. Suzuki, "Evaluation of Educational Effects of the Solid Simulator," *Journal for Geometry and Graphics*, vol. 3, pp. 219-226, 1999.
- [152] Z. S. Szalavári and M. Gervautz, "The Personal Interaction Panel - A Two-Handed Interface for Augmented Reality," *Computer Graphics Forum*, vol. 16, pp. 335-346, 1997.
- [153] 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.
- [154] B. Thomas, B. Close, J. Donoghue, J. Squires, P. D. Bondi, M. Morris, and W. Piekarski, "Arquake: An outdoor/indoor augmented reality first person application," in *Proceedings of ISWC2000*. Atlanta, Georgia, USA, 2000, pp. 139-146.
- [155] B. H. Thomas, V. Demczuk, W. Piekarski, D. H. Epworth, and B. Gunther, "A wearable computer system with augmented reality to support terrestrial navigation," in *Proceedings of ISWC'98*. Pittsburgh, USA, 1998, pp. 168-171.
- [156] M. Tidwell, R. S. Johnston, M. D., and T. A. Furness, "The Virtual Retinal Display - A Retinal Scanning Imaging System," *Proceedings of Virtual Reality World '95*, pp. 325-333, 1995.
- [157] A. Totter, L. Scott, G. Sven, and G. Gudela, "Evaluation Methodology," in *Deliverable D7.1, Lab@Future EU Project*, 2003.
- [158] UnigraphicsSolutions, "Parasolid V15.1," <http://www.eds.com/products/plm/parasolid/>, 2003.
- [159] S. G. Vandenberg and A. R. Kuse, "Mental Rotations: a group test of three-dimensional spatial visualization," *Perceptual and Motor Skills*, vol. 47, pp. 599-604, 1978.
- [160] 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*, vol. 117, pp. 250-270, 1995.
- [161] G. Weiss, "Räumliche Deutung von Ellipsenkonstruktionen.," *IBDG - Informationsblätter der Geometrie*, vol. 1998, pp. 5-12, 1998.
- [162] J. Wernecke, *The Inventor Mentor: Programming Object-Oriented 3D Graphics with Open Inventor*, 2nd ed: Addison-Wesley, 1993.
- [163] J. Wernecke, *The Inventor Toolmaker: Extending Open Inventor*, 2nd ed: Addison-Wesley, 1994.
- [164] J. R. Wilson, S. Nichols, and C. Haldane, "Measurement of presence and its consequences in virtual environments," *International Journal of Human-Computer Studies*, vol. 52, pp. 471-491, 2000.

- [165] W. Winn, "A Conceptual Basis for Educational Applications of Virtual Reality," Technical Report TR 93-9, 1993.
- [166] W. Winn, "The Impact of Three-Dimensional Immersive Virtual Environments on Modern Pedagogy," Human Interface Technology Laboratory, University of Washington, Discussion paper for NSF Workshop. HITL Technical Report R-97-15, 1997.
- [167] W. Wunderlich, *Darstellende Geometrie, Band 2*. Mannheim; Wien: Bibliogr. Inst. (BI-Hochschultaschenbücher; 133), 1967.

Curriculum Vitae

Hannes Christian KAUFMANN

Birth date and place: April 9th 1974, 11:06am CET in Amstetten, Austria

Marital status: Single

Parents: Herbert Kaufmann, high school teacher
Helga Kaufmann, kindergarten director

School education

1980 - 1984	Primary School (Volksschule) in Ybbs
1984 - 1992	Junior and High School (Realgymnasium) in Wieselburg (focus on mathematics and sciences)
June 16 th 1992	Graduation (Matura) from BRG Wieselburg with honors
Oct 1992 - Jun 1999	Studies in mathematics and descriptive geometry at the Vienna University of Technology
July 1 st 1999	Graduation to M.S. „Magister der Naturwissenschaften“ from the Vienna University of Technology with honors. Thesis: „Geometry of developable surfaces with computational methods“ („Rechnergestützte Geometrie der Torsen“), advisor: Helmut Pottmann
Oct 1999 – Mar 2004	Doctoral program in computer science at the Institute of Software Technology and Interactive Systems at Vienna University of Technology
March 2004	Dissertation “Geometry Education with Augmented Reality”, advisor: Dieter Schmalstieg

Work experience

1994	Software development on CAD software (CAD-DG) for the Institute of Geometry under direction of Univ.-Prof. H. Stachel.
1995 - 1996	Student assistant at the Institute of Geometry
1995	Programming control software to analyse soil samples for their

	hydrological conductivity in the laboratory (Federal Institution of soil water research management).
1995 - 2002	Tutor at the computer center at University of Vienna
1996 - 1998	Webmaster of the Institute of Geometry. Generation of the institute's web pages.
1997	Content development (11 th and 12 th grade) for the mathematics CD-ROM „Mathe Trainer for Austrian high schools“, authorized by the Austrian ministry of education for use as an educational material in high schools.
1997	Co-founding of the company Geometrek (www.geometrek.com) - 3D web solutions.
1997-1998	Software development for the educational project „Fächerübergreifende Projekte im Geometrieunterricht“ of the Institute of Geometry.
1998	Co-founding of the company Rename.net (www.rename.net). Internet homepage redirection service.
1998 - 2000	Part time employee at UMA information technology AG (http://www.uma.at/).
1997 - 1999	Scientific work on a project about „Geometric design of developable surfaces“ (P12252) funded by the Austrian Research Fund (FWF) under direction of Univ.-Prof. Dr. H. Pottmann.
1999 – present	Collaboration with a group of Austrian geometry teachers (ADI-GZ/DG) with the aim of creating didactic innovative content for geometry education.
2000	Development of an ADI CD-ROM with interactive content. The Austrian ministry of education buys a general license for usage of our CD-ROM in all higher Austrian schools.
2000 – 2002	Scientific work on an FWF funded project “Mobile Collaborative Augmented Reality” (P14470-INF) at the Institute of Software Technology and Interactive Systems under direction of Ao.Univ.-Prof. Dr. Dieter Schmalstieg.
2002 – 2003	Civil service at the Austrian children's organisation “Kinderfreunde” in Vienna, 2 nd district.
Feb. 2003 - present	Scientific work on the EU IST Project Lab@Future (IST-2001-34204) coordinated by Davarakis C. (Systema Informatics, Greece) at the Institute of Software Technology and Interactive Systems under direction of Univ.-Prof. Dr. Christian Breiteneder.

