Physical Cloth Simulation and Applications for the Visualization, Virtual Try-On, and Interactive Design of Garments

Dissertation
der Fakultät für Informations- und Kognitionswissenschaften
der Eberhard-Karls-Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von
Dipl.-Math. Michael Keckeisen
aus Ravensburg

Tübingen
2005

Mündliche Qualifikation: 02. 05. 2005
Dekan: Prof. Dr. M. Diehl
1. Berichterstatter: Prof. Dr.-Ing. Dr.-Ing. E.h. W. Straßer
2. Berichterstatter: Prof. Dr. R. Klein (Universität Bonn)

Erklärung

Hiermit erkläre ich, dass ich die Arbeit selbständig und nur mit den angegebenen Hilfsmitteln angefertigt habe und dass alle Stellen, die im Wortlaut oder dem Sinne nach anderen Werken entnommen sind, durch Angaben der Quellen als Entlehnung kenntlich gemacht worden sind.

Tübingen, Januar 2005 *Michael Keckeisen*

Zusammenfassung

Diese Dissertation befasst sich mit der physikalischen Simulation von Textilien und ihren Anwendungen in der Computergrafik für die Visualisierung, die virtuelle Anprobe und den interaktiven Entwurf von Bekleidung.

Zunächst wird die physikalische Modellierung von inneren und äußeren Kräften in der Simulation von Textilien behandelt. Zum einen wird ein Modell für innere Kräfte vorgestellt, das auf linearen Finiten Elementen für viskoelastische deformierbare Flächen basiert. Hierbei wird durch die Konstruktion eines lokalen Referenzrahmens für jedes Element die Anwendung Linearer Elastizitätstheorie ermöglicht. Zum anderen werden zwei Methoden für die Modellierung von Windeffekten in Textilsimulationen präsentiert. Im ersten Modell werden dynamische Strömungen durch die Navier-Stokes-Gleichungen beschrieben, während im zweiten Modell Windpartikel simuliert werden, die sich auf den Trajektorien eines globalen Windfeldes bewegen, um aerodynamische Effekte und Windschattenbereiche zu ermitteln.

Zur numerischen Approximation der Lösungen der entstehenden gewöhnlichen Differentialgleichungen wird im Anschluß eine Methode zur Parallelisierung der impliziten Zeitintegration vorgestellt, die speziell für Systeme mit verteiltem Speicher ausgelegt ist. Dabei wird eine statische Aufgabenzerlegung und -abbildung vorgenommen, und es wird gezeigt, dass eine substantielle Steigerung der Performanz durch Parallelisierung möglich ist.

Danach werden Anwendungen der vorgestellten Methoden beschrieben. Zum einen wird das entwickelte Kleidersimulationssystem *TüTex* vorgestellt, zum anderen wird auf die Anwendung des *TüTex*-Simulators im Maya-Plugin *tcCloth* und im Forschungsprojekt *Virtual Try-On*, das die virtuelle Anprobe von Bekleidung zum Ziel hatte, eingegangen.

Der letzte Teil dieser Arbeit beschäftigt sich mit der Simulation und dem Entwurf von Bekleidung in interaktiven virtuellen Umgebungen. Das im Rahmen dieser Arbeit entwickelte Virtual Reality System *Virtual Dressmaker* erlaubt die interaktive Konstruktion und Simulation von Kleidung unter

Verwendung von Eingabegeräten mit sechs Freiheitsgraden und einer Stereo-Visualisierung. Die vorgestellten Methoden werden auf der Basis von zwei Benutzerstudien mit herkömmlichen Interaktionstechniken verglichen. Schließlich werden virtuelle Schneiderwerkzeuge vorgestellt, die den interaktiven Entwurf von Textilien am dreidimensionalen, simulierten Modell mit vollautomatischer Modifikation der zugrunde liegenden planaren Schnittmuster erlauben.

Abstract

This thesis deals with the physical simulation of textiles and its applications in computer graphics for the visualization, virtual try-on, and interactive design of garments.

First, the physical modeling of internal and external forces in the simulation of textiles is treated. On the one hand, a model for internal forces is proposed that is based on linear finite elements for viscoelastic deformable surfaces. By constructing a local reference frame for each element the application of linear elasticity becomes possible. On the other hand, two methods for the modeling of wind effects in cloth simulations are presented. In the first model, the fluid flow is described by the Navier-Stokes equations, while in the second model wind particles are simulated that move on the trajectories of global wind fields in order to compute aerodynamic and lee effects.

Subsequently, in order to approximate numerically the solutions of the arising differential equations, a method to parallelize the implicit time integration is presented which is specifically designed for distributed memory systems. Here, static task decomposition and mapping is employed, and it can be shown that substantial performance improvements are possible due to parallelization.

After that, applications of the presented methods are described. First, the developed cloth simulator *TüTex* is presented. Then, the application of *TüTex* in the Maya plugin *tcCloth*, and in the research project *Virtual Try-On* are illustrated.

The last part of this work deals with the simulation and design of clothing in interactive virtual environments. The virtual reality system *Virtual Dressmaker* which has been developed within the scope of this work allows the interactive construction and simulation of garments using input devices with six degrees of freedom and stereo visualization. The employed methods are compared with conventional interaction techniques on the basis of two usability studies. Finally, virtual tailor tools are presented that allow

the interactive design of textiles by working with the three-dimensional simulated model while the modifications are transferred automatically to the underlying planar cloth patterns.

Acknowledgements

I thank Prof. Dr.-Ing. Dr.-Ing. E.h. Wolfgang Straßer for the opportunity to work on this dissertation at WSI/GRIS, and his constant support and advice.

Furthermore, I want to thank all my colleagues, without whom this work would not have been possible, especially Dr. Wolfgang Blochinger, Dr. Olaf Etzmuß, Dr. Michael Hauth, Stefan Kimmerle, Johannes Mezger, Dr. Stanislav L. Stoev, and Dr. Markus Wacker. Moreover, I thank the students Matthias Feurer, Martin Gruber, Christian Michel, Simon Pabst, and Bernhard Thomaszewski, who implemented parts of the described algorithms in the course of student and diploma theses.

Finally, I want to thank all the people at WSI/GRIS for the friendly working atmosphere and the good collaboration.

This work was partially supported by the bmb+f *Virtual Try-On* grant.

Contents

1 **Introduction** **1**
  1.1 Cloth Simulation in Computer Graphics 1
  1.2 Preliminaries and Notation 2
  1.3 Overview and Contributions 7

2 **Modeling Internal and External Forces of Textiles** **9**
  2.1 Introduction and Related Work 10
    2.1.1 Internal Forces 10
    2.1.2 External Forces 11
  2.2 Constructing Clothes from Planar Patterns 12
    2.2.1 Patterns and Seams 13
    2.2.2 The Sewing Algorithm 15
  2.3 A Linear Finite Element Model for Cloth Simulations 16
    2.3.1 Linear Elasticity 16
    2.3.2 Rotated Finite Elements 19
    2.3.3 Bend Forces 22
    2.3.4 Time Integration 23
    2.3.5 Material Parameters of Textiles 24
    2.3.6 Results 25
  2.4 Modeling Air Resistance and Wind 27
    2.4.1 Aerodynamics 28
    2.4.2 Wind Field Model I — The Navier-Stokes equations . . 29
      2.4.2.1 Approximating the Navier-Stokes equations . 30
      2.4.2.2 Wind effects on clothes 30
    2.4.3 Wind Field Model II — Particle Tracing 31
      2.4.3.1 Global Wind Fields 32
      2.4.3.2 Tracing Wind Particles 32
    2.4.4 Comparison 34
    2.4.5 Results 35
  2.5 Summary 36

3 Parallel Numerical Computation 39
3.1 Introduction and Related Work . . . 39
3.2 Parallel Time Integration on Distributed Memory Architectures 41
3.2.1 The Sequential Simulation Loop . . . 41
3.2.2 Design of a Parallel Cloth Simulator . . . 44
3.2.3 Task Decomposition and Task Mapping . . . 44
3.2.4 The Parallel Simulation Loop . . . 46
3.2.5 Performance Measurements . . . 48
3.2.5.1 Test Scenarios . . . 48
3.2.5.2 Results . . . 50
3.3 Summary . . . 52

4 TüTex and Virtual Try-On 53
4.1 Introduction and Related Work . . . 53
4.2 The TüTex Cloth Simulation Engine . . . 54
4.3 The tcCloth Maya Plugin . . . 55
4.4 Virtual Try-On . . . 57
4.5 Summary . . . 59

5 Interactive Cloth Assembly, Simulation and Design 61
5.1 Introduction and Related Work . . . 62
5.2 The Virtual Dressmaker . . . 65
5.2.1 Application Overview . . . 65
5.2.2 System Setup . . . 66
5.2.2.1 Virtual Environment . . . 66
5.2.2.2 Cloth Simulation Engine . . . 67
5.2.2.3 Application Architecture . . . 68
5.2.3 User Interface and Navigation in the Scene . . . 68
5.2.4 Interactive Cloth Assembly and Simulation . . . 71
5.3 Usability Evaluations . . . 72
5.3.1 Pilot Study . . . 72
5.3.1.1 Results of the Pilot Study . . . 73
5.3.1.2 Consequences of the Pilot Study . . . 74
5.3.2 Experiment Setup . . . 76
5.3.2.1 Test Scenarios and Questionnaire . . . 76
5.3.2.2 Participants . . . 79
5.3.2.3 Task description . . . 79
5.3.2.4 Hypotheses . . . 80
5.3.3 Results . . . 81
5.3.3.1 Measurements of Time and Precision . . . 81
5.3.3.2 Subjective Impressions . . . 81

5.3.4 Evaluation 83
5.3.5 Consequences of the User Studies 87
5.3.5.1 Ergonomics 87
5.3.5.2 Improved Navigation Methods 88
5.4 Interactive Garment Design 89
5.4.1 Virtual Pins 90
5.4.2 Virtual Scissors 90
5.4.3 Virtual Sewing Needle 94
5.4.4 Results 96
5.5 Summary 96

6 Outlook on Future Research **99**

Publications **101**

Bibliography **105**

List of Figures

1.1 A shirt discretized by a polygonal mesh 3

2.1 A man's shirt consisting of eight planar cloth patterns 13
2.2 Mapping of the undeformed to the deformed state 17
2.3 Polar decomposition of the deformation gradient 19
2.4 Material parameters 25
2.5 A shirt simulated with different material properties 26
2.6 A walking man wearing a shirt and trousers 27
2.7 The decomposition of wind forces 28
2.8 Flow primitives 32
2.9 A flag blowing in the wind 35
2.10 Two flags blowing in the wind (Navier-Stokes equations) . . . 35
2.11 Two flags blowing in the wind (particle tracing method) . . . 35
2.12 A man wearing a shirt and standing in a wind stream 37

3.1 Ghost nodes 47
3.2 First test scene: a square piece of cloth 49
3.3 Second test scene: a shirt 49
3.4 The piece of cloth partitioned for 4 and 12 processors 49
3.5 The shirt partitioned for 4 and 12 processors 49
3.6 Computation times of the cloth and shirt scenes 50
3.7 Speedup and efficiency for the cloth scene 51
3.8 Speedup and efficiency for the shirt scene 51

4.1 The TüTex cloth simulation engine 54
4.2 Interactive generation of patterns and seams in tcCloth 56
4.3 Simulation results computed within tcCloth 57
4.4 A woman wearing trousers of different sizes 58
4.5 Comparison of real and virtual clothes 59

5.1 Cloth assembly in the Virtual Dressmaker 63
5.2 The virtual reality hardware setup 67

5.3 The client-server configuration 68
5.4 The selection sheets 69
5.5 The navigation and simulation sheets 70
5.6 Interactively adjusting the cloth during the simulation 71
5.7 Pilot study: subjective impressions 74
5.8 Pilot study: results of the time measurements 75
5.9 User study: the test scenes 77
5.10 User study: differences between the test applications 77
5.11 User study: time measurements 82
5.12 User study: precision measurements 83
5.13 User study: results of the questionnaire 84
5.14 User study: evaluation of time and precision measurements . . 85
5.15 User study: Maya experts vs. Maya beginners 86
5.16 User study: results grouped by application order 87
5.17 Interactive garment design 89
5.18 Virtual pins 91
5.19 Virtual scissors: the cutting plane 92
5.20 Virtual scissors: cutting the patterns 93
5.21 A pullover is made shorter 94
5.22 A skirt is made tighter 95
5.23 The side and collar of a pullover are modified 97

Chapter 1

Introduction

1.1 Cloth Simulation in Computer Graphics

Cloth simulation has a long history in both computer graphics and textile engineering. Both areas aim at predicting the behavior of textiles on the basis of physical models and computer simulations. While textile engineers are interested in understanding the mechanical properties of textiles or even single threads, computer graphics has mostly focused on macroscopic visual effects for computer animations. Here, the main areas of research are the physically based modeling of deformable surfaces, the numerical solution of the arising differential equations, collision detection and response, and the photo-realistic rendering of textiles. Apart from applications in the film and computer games industry, such as the simulation of clothing for virtual actors, cooperations between computer graphics and the cloth manufacturing industry have recently become possible. A major topic of current research is the virtual fitting room scenario, where techniques for virtual selection, cloth assembly, try-on, and evaluation of individual clothing are developed. The goals in this area include internet-based clothes shops, mass production of made-to-measure-wear, and, taking this a step further, interactive garment design and tailoring. In 1996, David E. Breen proposed that "future apparel and CAD systems should allow fashion designers to experiment easily with a variety of fabrics and patterns on a 3D dynamic virtual mannequin before the actual garment is manufactured" [34]. The physical simulation of clothing and applications in this context are the subject of this work.

It is an appealing characteristic of cloth simulation research in computer graphics that it involves interdisciplinary topics from mathematics, physics, and computer science. This is reflected by the content of this work. The next chapter treats the physically based modeling of internal and external forces in cloth simulations, involving physical models of elasticity theory and fluid dynamics, as well as mathematical discretization techniques for the corresponding partial differential equations. Numerical algorithms and their parallelization for the solution of the ordinary differential equations arising from Newton's equation of motion are the topic of the third chapter. In the fourth chapter, the developed methods are integrated into a complete cloth simulation software package and applied for virtual try-on of made-to-measure garments. Finally, chapter five addresses human-computer interaction techniques and geometric algorithms for the interactive assembly, simulation, and design of clothing in three-dimensional immersive virtual environments.

1.2 Preliminaries and Notation

Geometrically, we model a piece of cloth as a polygonal mesh, as shown in figure 1.1. In particular, triangle meshes are suited very well to geometrically model arbitrary surfaces. This applies especially for garments that have to be assembled from several single cloth patterns, as described in section 2.2.

The task of a physical cloth simulation is to compute the movement of the vertices of the mesh, and thus the movement of the cloth, depending on internal and external forces that act on the textile. We denote the vector containing all 3D vertex positions with x, and the vector containing all 3D vertex velocities with v. Given the vector of forces f, Newton's equation of motion

$$f_i(x, v) = m_i \frac{d^2 x_i}{dt^2} \tag{1.1}$$

is used to compute the trajectory of each vertex x_i with mass m_i over time t. The internal forces in textiles depend on the deformation and thus on the current positions and velocities of the vertices. In general, the vector f contains both internal and external forces, i.e.

$$f = f_{\text{internal}} + f_{\text{external}}.$$

Internal forces comprise viscoelastic resistance due to tension, shearing, transversal contraction, and bending. External forces may include gravity, air resistance, wind effects, collision forces, and arbitrary other forces depending on the application.

In computer graphics, discrete systems like mass-spring and particle systems have been widely used to model internal forces in textiles. In these

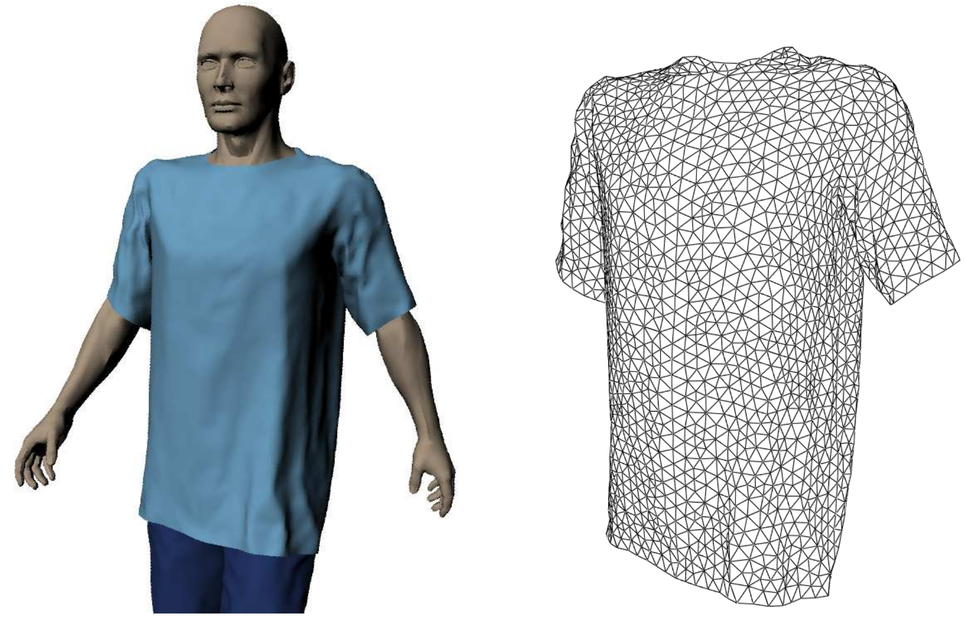


Figure 1.1: Textiles are discretized by polygonal meshes. The shirt on the left is represented by an unstructured triangle mesh, shown on the right.

models, the mesh topology defines, how the vertices interact and exert forces on each another. In the most simple case of a mass-spring system, the internal viscoelastic tension forces are defined by linear springs along the edges of the mesh. Thus, the tension force between two neighboring vertices x_i and x_j is given as

$$\begin{aligned} f_{ij}(x,v) = \quad & k_{ij}(\|x_i - x_j\| - l_{ij})\tfrac{x_i - x_j}{\|x_i - x_j\|} \\ & -d_{ij}\tfrac{\langle v_i - v_j, x_i - x_j\rangle}{\|x_i - x_j\|^2}(x_i - x_j), \end{aligned}$$

where the damping force has been projected to the spring's direction. Here, l_{ij} denotes the spring's rest length, and k_{ij} and d_{ij} are the spring's viscoelastic material constants, which are mostly chosen *suitable for nice visual effects*. Shear and bend forces can be handled similarly by additional (spring) forces, and the various mass-spring and particle systems for textiles which have been proposed in the past differ by the respective methods how to compute the forces based on neighboring vertices, e.g. [54, 101, 127, 26, 128, 39]. While these mechanical models are relatively simple and fit together well with efficient time integration methods, they share several well known disadvantages, most notably the resolution dependence and the difficulty to map real material parameters onto the textiles. With more and more demanding applications, however, like simulating the fit of clothing on real people or photo-realistic animations of virtual actors for entertainment purposes, the possibility to simulate different materials as realistically as possible becomes increasingly important. Continuum mechanics models are well known to solve these problems, but existing nonlinear finite element solutions are generally too slow to be applied in computer graphics applications [55, 77]. A particle system derived from continuum mechanics has been presented by Etzmuß [56, 57], but since it is based on finite differences, this approach works only for rectangle meshes. This problem will be addressed in section 2.3, where we propose an efficient finite element model based on linear elasticity which works with arbitrary unstructured triangle meshes. Similarly, the physical simulation of external forces is necessary to provide realistic virtual scenes. Aerodynamic effects like air resistance and wind can be integrated into cloth simulations as additional external forces in Newton's equation of motion. The main question in this context is how to model realistic wind flows without having too high computational costs, and how the additional external forces can be computed. This will be the topic in section 2.4.

Since Baraff and Witkin's work [26], implicit numerical time integration has established itself as a powerful numerical method to solve the stiff ordinary differential equations that arise in cloth simulations. Rewriting equation (1.1) as a system of first order ordinary differential equations

$$\dot{x}(t) = v(t) \tag{1.2}$$
$$\dot{v}(t) = M^{-1} f(x(t), v(t)), \tag{1.3}$$

and given the forces at a time t, we first have to compute $v(t)$ and then $x(t)$ by approximating these two equations.

The most simple explicit integration scheme, the forward Euler method, iteratively computes approximate solutions to equations 1.2 and 1.3 by

$$\begin{aligned} x(t+h) &= x(t) + hv(t) \\ v(t+h) &= v(t) + hM^{-1} f(x(t), v(t)), \end{aligned}$$

while the corresponding implicit integration scheme, the backward Euler method achieves this by

$$\begin{aligned} x(t+h) &= x(t) + hv(t+h) \\ v(t+h) &= v(t) + hM^{-1} f(x(t+h), v(t+h)), \end{aligned}$$

where h denotes the time step. The main difference between these two methods is the fact that the implicit method takes forces at time $t+h$ into account while the explicit integration scheme considers forces at time t. This yields more stable solutions at the cost of having to solve for $v(t+h)$ rather than just evaluating a formula as in the explicit Euler method. In general, the equations are linearized and solved by a conjugate gradient method [26, 110]. For a detailed comparison of explicit, implicit, and mixed implicit-explicit (IMEX) integration schemes we refer to the analysis given by Hauth [70, 68]. In this work, we will use implicit time integration methods. In particular, we will present a parallel implicit time integration method designed for distributed memory parallel architectures in chapter 3.

Current cloth simulation engines have become powerful enough to be used in applications relevant to textile industries. Major areas of research in this context are the virtual try-on of virtual garments on digital counterparts of real people on the one hand (cf. chapter 4), and the interactive design of clothing on the other hand (cf. chapter 5). In particular, interactive garment design was a research topic in computer graphics already in 1990, when geometric methods to model garment patterns as static three-dimensional surfaces were proposed by Hinds [73]. In the meantime, much progress has been made both in the research areas of cloth simulation in physically based modeling and of interaction techniques in virtual reality (VR). On the one hand, physical models and fast numerical algorithms now provide the simulation of the dynamic draping behavior of clothing at interactive frame rates, at least for not too detailed objects. On the other hand, virtual reality techniques and concepts, mostly offering input devices with six degrees of freedom (6DOF) and stereoscopic visualization, have proved to be a useful alternative to traditional desktop applications with two-dimensional input devices and monoscopic displays in many areas. For an introduction to interaction techniques in virtual environments, we refer to the text book presented by Vince [125], and Stoev's work [120].

General introductions to cloth simulation in computer graphics can be found in the books presented by House and Breen [75] and by Volino and Magnenat-Thalmann [128], Eberhardt's work [52], and the recent tutorial [10].

Table 1.1 summarizes some of the notation used in this work. Vectors are denoted with lower case, matrices with upper case letters. The components of a vector are denoted by subscripts, e.g. the position of the i-th vertex in the cloth mesh is written as x_i. The transpose of vector x is written as x^T, analogously, A^T denotes the transpose of matrix A. In the notation for continuum mechanics and finite elements, matrices relating to the rest positions are indicated by a bar. Matrices relating to the bend forces are indicated by tildes, and those relating to the viscous forces by a hat.

Notation

General notation

x	vector containing the vertex positions in the cloth mesh
v	vector containing the vertex velocities
n	vector containing the vertex normals
f	vector containing the forces that act on the vertices
m, M	vector, resp. diagonal matrix, containing the masses of all vertices
t	time
h	time step

Notation for continuum mechanics and finite elements

σ, σ^{ν}	elastic stress, resp. viscous stress
ϵ_G, ϵ_C	Green's strain, resp. Cauchy's strain
C, D	elasticity tensor, resp. viscosity tensor
J	deformation gradient
(e_1, e_2, e_3)	world coordinate system
(u_1, u_2)	coordinate system in the rest state
$\overline{x}$	vector containing the vertex positions in the rest state
d	vector containing the vertex displacements
T_m	m-th triangle in the mesh, with vertices a, b, c
Ω_{T_m}	area of triangle T_m
N_a^m	linear basis function for triangle T_m and vertex a, with $N_a^m = 1$ and $N_{b,c}^m = 0$
K_{ab}^m	local stiffness matrix for edge (a, b) in triangle T_m
R^m	rotation matrix for triangle T_m
A, B	global stiffness matrices for elastic, resp. viscous forces
Δ	Laplacian operator

Notation for air resistance and wind effects

f_L, f_D	lift forces, resp. drag forces that act on the cloth
C_L, C_D	air resistance coefficients for lift and drag
ρ	air density
u	velocity field of the wind
w	velocity field obtained from a superposition of local wind sources w_i
p	vector containing the positions of the wind particles

Table 1.1: Some of the notation used in this work.

1.3 Overview and Contributions

This work contributes to the research areas of physically based modeling of textiles in computer graphics, parallel numerical time integration, and human-computer interaction techniques for the interactive design of clothes.

Each of the next four chapters starts with an introductory section giving background information and an overview of related work. At the end of each chapter, we give a summary of the main results. In the following, we outline the content of the next chapters in more detail, and summarize the respective contributions.

In chapter 2, we treat the physical modeling of internal and external forces of textiles. First, we propose an algorithm that allows to construct three-dimensional virtual clothes, just as real ones, from two-dimensional planar cloth patterns (section 2.2). Thus, input data from textile CAD systems can be used for the physical cloth simulation which yields a natural description of the cloth's rest state. All the garments shown in this work are constructed in this way. Next, we present a new model for internal forces in textiles based on linear finite elements for viscoelastic deformable surfaces (section 2.3). By constructing a local reference frame for each element, linear elasticity can be applied in cloth simulations, what has not been possible before. This is achieved by removing the rotational part of the deformation gradient by a polar decomposition. Compared to previously used discrete models, this approach allows a straightforward integration of measured material parameters, and yields convincing visual results while the computation times remain in the same order of magnitude as in conventional mass-spring and particle systems.[1] After that, we show how to incorporate aerodynamic effects in cloth simulations (section 2.4). We present two new alternative wind field models, one based on the Navier-Stokes equations and one based on particle tracing in global wind fields. In the first model, we extend previous work on the fast approximation of complex fluid flows by computing the respective additional external forces which act on textiles within the flow. In the second model, we introduce the concept of wind particles which move on the trajectories of a global wind flow composed of individual local wind sources to compute aerodynamic effects on textiles. To the best of our knowledge, we are the first to model lee effects by detecting collisions between the wind particles and the deformable and rigid objects in the scene.

---

[1] In the meantime, our approach has been adopted by other computer animation research groups, cf. [98, 95, 76]. Moreover, the polar decomposition has been shown to be fast and stable compared with other methods to construct a local reference frame [71].

Following the physical modeling, we deal with the numerical solution of the resulting ordinary differential equations in chapter 3. We propose a new approach to parallelize implicit time integration schemes in cloth simulations, and present a parallel cloth simulator which is specifically designed for distributed memory parallel architectures (section 3.2). We apply static task decomposition and mapping for the simulation of unstructured triangle meshes, and show that substantial performance improvements for real-world sized problems are possible due to parallelization.

In chapter 4, we present applications of the previously described methods. First, we briefly outline the cloth simulation engine *TüTex* which has been developed within the computer graphics research group WSI/GRIS during the last years and into which the algorithms described in the previous chapters have been integrated (section 4.2). Then, we present the Maya plugin *tcCloth* that we developed on the basis of TüTex in order to provide an easy to use possibility to set up simulation scenes and a powerful rendering tool (section 4.3). Finally, we discuss the application of TüTex for the simulation of made-to-measure garments in an automatic clothing pipeline realized in the research project *Virtual Try-On* (section 4.4).

Interactive simulation and design of clothing in virtual environments is the topic of chapter 5. Human-computer interaction techniques for cloth assembly and simulation are realized in the *Virtual Dressmaker*, a virtual reality application that supports input devices with six degrees of freedom and real-time visualization on a large stereo display (section 5.2). In particular, the system allows the interactive manipulation of the clothes during the physical simulation. In order to evaluate the employed interaction and navigation concepts, we verify them against two traditional desktop modeling applications with standard input devices and monoscopic displays in two comparative user studies (section 5.3). The results suggest that the proposed virtual reality techniques significantly improve interactive cloth assembly with respect to both efficiency and precision. Finally, we develop virtual tailor tools which allow the interactive design and modification of clothing in 3D. We provide virtual scissors, a virtual sewing needle, and virtual pins, together with algorithms for cutting and sewing meshes in 3D (section 5.4). The mesh modifications are automatically transferred to the planar cloth patterns and immediately integrated into the physical simulation. Altogether, the Virtual Dressmaker is an entirely new and unique system enabling the construction of virtual prototypes of clothes while working directly with the three-dimensional deformable model in an immersive virtual environment.

An outlook on future research directions concludes this work in chapter 6.

Chapter 2

Modeling Internal and External Forces of Textiles

Different physical models for cloth simulations in computer graphics are mainly distinguished by their way of computing the forces that act within and on the textiles. The modeling of both internal and external forces is crucial for realistic simulation results, and we will address both topics in this chapter.

First, we describe how three-dimensional virtual clothes can be assembled from planar cloth patterns, and we present an algorithm for the automatic topological merging of the patterns along predefined seam lines (section 2.2). All garments shown in this and the following chapters are constructed in this way. Then, we present an efficient model based on finite elements for viscoelastic, highly flexible surfaces (section 2.3). It is particularly designed for numerically stiff materials such as textiles because it yields linear equations in each time step and allows fast time stepping in an implicit integration method. This is achieved by reducing the nonlinear elasticity problem to a planar, linear one in each step. The described approach allows to model different material properties and results in a physically accurate but also fast simulation. Finally, we show how to incorporate effects of wind fields in cloth simulations (section 2.4). We discuss two different approaches to model force fields describing air motion and show how these models can be extended to interact with deformable thin objects such as textiles. The first model is based on the Navier-Stokes equations, while the second method extends simple particle tracing methods by the effect of lee. In both cases, we present methods for simulating the interaction of cloth movements with the wind field, and we compare the models respective advantages and disadvantages.

2.1 Introduction and Related Work

Cloth simulations in computer graphics have to meet three requirements concurrently. First, they should be visually convincing and, depending on the application, reproduce real physical effects, for example the different draping behavior of textiles due to different material properties. Second, the computation should be fast. The speed requirements, again, depend on the application, ranging from several minutes or a few hours per frame, for instance for an off-line simulation to be applied in a cinema production, to interactive frame rates for applications in computer games or interactive garment design. Third, the simulation should be numerically stable with respect to large forces that occur during the simulation.

While it has been shown that numerical stability can be achieved with implicit integration methods, the development of algorithms that provide realistic results at relatively low computational costs is a major area of research for both internal and external forces in cloth simulations.

2.1.1 Internal Forces

Physical models for textiles can be separated into discrete models like mass-spring or particle systems and continuous models derived from elasticity theory. Many discrete systems have been designed specifically for rectangle meshes, such as the simple mass-spring system by Provot [101], the particle systems presented by Breen [35] and Eberhardt [54], and the recent contribution of Choi [39]. However, with rectangles it is difficult to model garments which have to be constructed from complex patterns. Models for triangular meshes have been presented e.g. by Baraff [26], Volino [127, 128], and Choi [40]. Particle systems on triangular meshes, however, are mostly resolution dependent and generally do not allow the modeling of specific material properties, for instance the weft and warp directions in textiles.

Models derived from continuum mechanics solve these problems, and nonlinear finite element based models [55, 77] deliver precise simulations, but they need hours or even days to find a solution even for static problems. Only for the simulation of volumetric objects, a fast nonlinear finite element solution has been proposed [48]. But it relies on an explicit time integration scheme that is not feasible in cloth animation. A particle system has been derived from continuum mechanics by Etzmuß [57]. But since this work employs finite differences to discretize the arising partial differential equations, it works for rectangle meshes only. Therefore, we propose an approach based on rotated linear finite elements.

Rotated or corotational finite element formulations have been employed in airspace engineering already in 1976 [47], but to use them in physical simulations in computer graphics is a relatively new idea. For the simulation of volumetric objects, a technique similar to the one we propose in section 2.3 has been exploited by Müller [97]. In order to remove the rotational dependence of Cauchy's strain, the authors construct a warp for each vertex in the mesh before applying the forces due to the linear strain formulation. While our approach is quite similar in spirit, we will derive the linear strain measure directly from the nonlinear Green's strain formulation. This will be done all within the framework of finite elements. We will see that for each element there is a natural rotation that can be constructed by means of a polar decomposition. In the meantime, our approach has been followed by Müller [98] and other authors [95, 76], and it could be shown that the polar decomposition is stable and fast compared to other methods that provide a local reference frame [71].

For an in-depth treatment of linear and nonlinear continuum mechanics together with associated finite element techniques we refer to the book presented by Bonet and Wood [32]. This work will also be our primary source for the basic equations of linear elasticity which we will apply in section 2.3.

2.1.2 External Forces

Aerodynamic effects allow to enhance the realism of an animated scene and thus are an important part of a cloth simulation system. For example, air resistance is a vital component which cannot be neglected if a realistic animation is desired [28]. Models for fluid dynamics can be essentially subdivided into two categories. Simple models which are commonly used in most computer graphics applications describe the wind flow by predefined flow functions. Here, global functions are defined to model the wind velocity. Either special flow primitives can be combined [133, 89] or visually pleasing functions introducing random turbulences [111, 118, 114] are taken into account to model even complex wind scenes. Many models use this method to move objects in the wind field through the scene, e.g. [102, 113, 133, 29].

However, fixed flow functions lack interaction with the user or objects in the scene. Hence, with increasing computational power, physically more accurate models can be taken into account. In many fields the Navier-Stokes equations are the standard mathematical formulation to model fluid dynamics. A vast literature exists on how to approximate these equations numerically. Unfortunately, it is quite difficult to apply most algorithms in computer graphics due to the enormous calculation times. Hence, faster fluid solvers were investigated for computer graphics applications. Kajiya [80], Yaeger [135], and Gamito [63] worked on fluid dynamics solvers in two dimensions, and many improvements and variants followed [38, 85]. Foster and Metaxas [59, 60], and Griebel [65] presented a solver for the fully three-dimensional Navier-Stokes equations. However, they employed explicit integration methods, and therefore had to use very small step sizes. To enable faster simulations, a solution with an unconditionally stable solver was introduced by Stam [115, 58, 116, 117], and the first of the two wind field field models we propose in this chapter is based on this work.

Modeling interaction of fluids with solid objects has been investigated by Takahashi [123] and Génevaux [64], while Wei [132] presented an approach to simulate lightweight objects like soap bubbles and feathers in a wind flow. For the interaction of highly deformable objects and especially cloth-like objects with wind fields only few models have been investigated. Simple models consist in the calculation of lift and drag forces from the surrounding velocity field [111, 101, 82, 81]. More complex interaction models calculate the wind force by a panel method [89] introducing local vortices.

In section 2.4, we show how the recent results in fluid dynamics for computer graphics presented by Stam can be exploited to simulate the interaction of wind flows with textiles. Moreover, we extend the more straightforward approach of global wind field functions by the effect of lee.

2.2 Constructing Clothes from Planar Patterns

In this section, we describe how three-dimensional virtual clothes, just as real clothes, can be constructed from planar patterns (figure 2.1). This results in a natural starting condition for cloth animations, and recent work on cloth simulations has focused on this approach [55, 128, 39]. The advantages of starting with planar cloth patterns are:

- Real clothes are also constructed from planar patterns. Thus, virtual garments can be designed like the real ones, e.g. by taking the patterns from a cloth manufacturer's CAD database.
- The planar patterns provide a natural parametrization for the rest state in the cloth simulation, independent of the chosen physical model. For

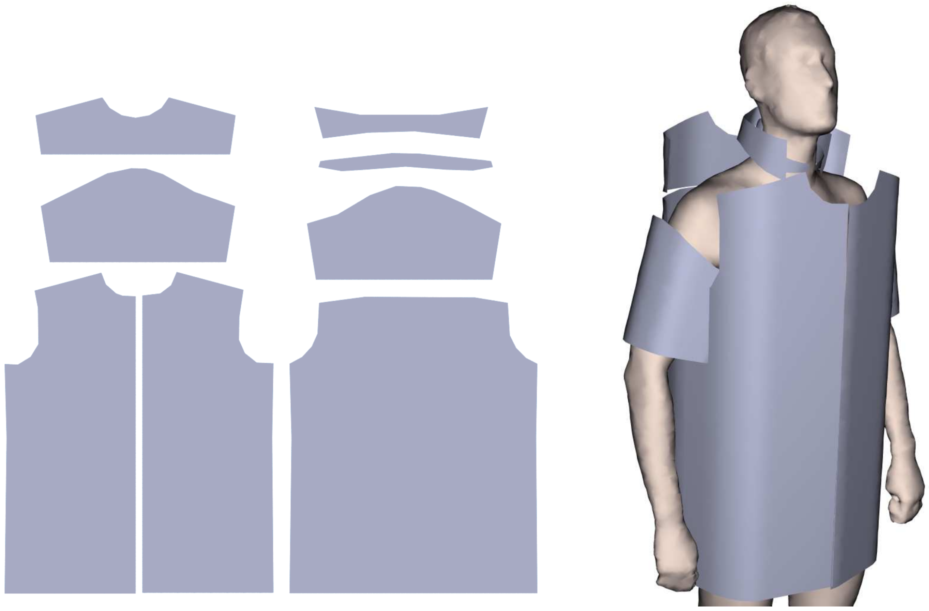


Figure 2.1: A man's shirt consisting of eight patterns. Before the sewing starts, the patterns are placed around the virtual character.

particle systems, the edges in the planar meshes can be taken as the spring's rest lengths. For finite element models, the discretized patterns allow the computation of the basis functions.

- Correct texture coordinates for rendering the simulated garments are obtained automatically from the planar rest state. They can be easily mapped to the three-dimensional garment during the sewing process, and provide a realistic visualization.

In the following, we first specify the input data for the sewing algorithm. Then we describe the sewing process, i.e. the topological merging of the patterns along seam lines, and the generation of a correct parameterization for simulating and rendering the virtual clothes.

In this section, we partially follow [5, 16].

2.2.1 Patterns and Seams

The input data for the sewing algorithm consists of planar patterns, seam information, and copies of the patterns positioned around the 3D figure. First, we need the two-dimensional cloth patterns from which the garment shall be constructed. The single patterns are assumed to be arbitrary triangulated polygons, for an example see the patterns of a man's shirt in figure 2.1, which are generated from real garment pattern shapes given by the boundary curves in a CAD system.

Second, the seam information has to be available. A seam belongs to exactly two (not necessarily distinct) patterns and consists of a starting point and an end point in the respective patterns. We assume that the patterns are triangulated in such a way that two corresponding seam lines have the same number of vertices. More precisely, each seam is given by a list of pairs of vertices (r_i, s_i) with r_i and s_i being the corresponding vertices on two not necessarily distinct planar patterns. Note that we exclude pathological cases like a vertex pair that belongs to a single triangle. Such a case can be handled by locally refining the given triangulation of the pattern. However, we have to account for the case that a vertex belongs to more than two seams, as this happens wherever more than two patterns are adjoined in a garment, e.g. when a sleeve is connected to the point where there is a seam between the front and the back part of a shirt. In our sewing algorithm, we will also handle these general situations.

Finally, we need copies of the planar patterns that have been positioned to the approximately correct places around the virtual character, as seen in figure 2.1. Here, the requirements, in order to provide a good initial position for the subsequent simulation, are:

- The patterns should not collide with the virtual character or with each other.
- The patterns should be as close to the virtual character as possible.
- Corresponding seams should be as close together as possible, and it has to be possible to connect them without penetrating the avatar.

This prepositioning can be done manually as described in [128], and in chapter 5, we will show that this task can be facilitated by virtual reality techniques. For the example given in figure 2.1, we used the prepositioning system described in [66, 62], which places the patterns automatically around the avatar based on feature points which have to be available both on the cloth and on the 3D character (cf. also section 4.4).

Given these prepositioned patterns, the garment can be connected along the corresponding seam lines.

2.2.2 The Sewing Algorithm

Sewing the prepositioned cloth patterns together can be achieved by topologically merging the patterns along the defined seam lines, as described in algorithm 1. For all pairs of vertices (r_i, s_i) in all seams, the corresponding triangles are connected, and the seam points are moved halfway between the two original points.

Algorithm 1 Sewing Clothes from Patterns

```
1: for all seams do
2:    for all vertex pairs $(r_i, s_i)$ in the seam do
3:       Move the coordinates of $r_i$ halfway in the direction of $s_i$.
4:       Replace $s_i$ by $r_i$ in all triangles of the prepositioned pattern that
         contain $s_i$.
5:       Store the local planar parametrization for the simulation engine (e.g.
         original lengths of edges connected to $r_i$ in the case of a mass-spring
         system).
6:       Store the planar coordinates of $r_i$ and $s_i$ as (multiple) texture coor-
         dinates for vertex $r_i$ in the 3D mesh.
7:       Remember that $s_i$ has become $r_i$ in case there are more seams con-
         taining $s_i$.
8:    end for
9: end for
```

Note that we assume a global numbering of the vertices r_i and s_i which is valid over all patterns. Step 7 in algorithm 1 is necessary, because several seams may contain the same vertex. Compared to the collection of the patterns, the sewing algorithm reduces the number of vertices in the mesh, since the vertices s_i are not needed anymore in the three-dimensional mesh that shall be simulated, and can be discarded. The number of edges may also be reduced, depending on the seams, while the number of triangles remains constant.

The planar cloth patterns provide a planar parametrization both for the physical cloth simulation and for a realistic visualization with textures. In the latter case, we provide each triangle in the sewn mesh with texture coordinates for each of its three vertices. This means that each vertex has separate texture coordinates for each triangle it belongs to. Thus, a realistic texturing along the seam lines is possible.

After the patterns have been topologically connected along the corresponding seams, the simulation computes the contraction of the resulting deformed triangles, while gravity is turned off. After the sewing simulation has reached a static solution, gravity is turned on and the cloth simulation, possibly with an animated virtual character, can be started.

2.3 A Linear Finite Element Model for Cloth Simulations

In the past, finite element solutions for cloth simulations have been used mainly in the textile research community rather than in computer graphics. The reason are the high computational costs for models of nonlinear continuum mechanics. Linear elasticity, however, employs a linear material law as well as a linear strain formulation. Therefore, it leads to a system of linear equations in each time step when applied in an implicit time integration scheme, rather than to nonlinear equations in the case of a nonlinear strain measure. Thus, it is much faster to compute. However, linear elasticity has not been applied in cloth simulations so far, because the linear strain formulation is only valid for small displacements and therefore not rotationally invariant. In this section, we address this problem by constructing a local reference frame for each triangle in the mesh which allows to remove the rotational part of the deformation and to apply linear elasticity for in-plane internal forces in textiles.

In parts, we follow Etzmuß, Keckeisen, and Straßer [2, 15].

2.3.1 Linear Elasticity

In continuum mechanics, the elastic internal forces of a deformable object are given by

$$f_{\text{internal}} = -\text{div}\sigma, \tag{2.1}$$

as described in [32, p.103]. Here, the continuous stress σ has to be computed from the continuous strain ϵ by some material law. If Hooke's linear material law is applied, stress is computed from strain by

$$\sigma = C\epsilon, \tag{2.2}$$

where C is the elasticity tensor which contains the material properties.

In order to measure strain in deformable surfaces, we consider a function s which maps the undeformed rest state of a deformable surface, locally

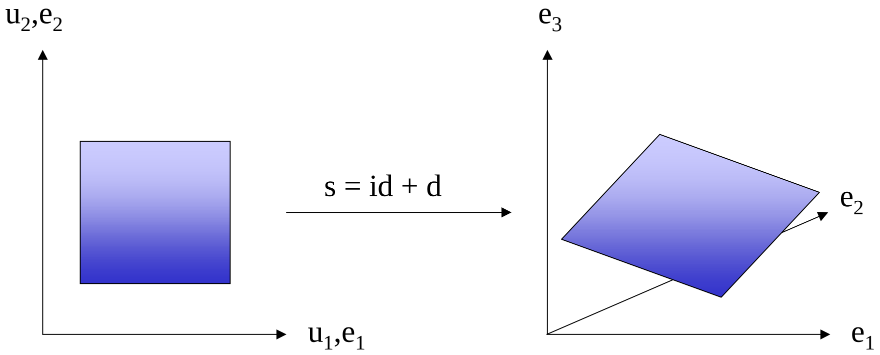


Figure 2.2: The undeformed rest state of an object (left) is mapped to the deformed state (right) by a function s. The displacement d is defined as $d = s - id$, where id is the identity mapping.

parameterized by (u_1, u_2), to the deformed state in world coordinates[1], as shown in figure 2.2. The displacement d is defined as $d = s - id$, where id is the identity mapping.

Then, as shown in [32, p.12f], a nonlinear strain measure is given by Green's symmetric strain tensor

$$\epsilon_G = \begin{pmatrix} \epsilon_{G_{1,1}} & \epsilon_{G_{1,2}} \\ \epsilon_{G_{2,1}} & \epsilon_{G_{2,2}} \end{pmatrix},$$

where in terms of the displacement d the entries are given by

$$\begin{aligned} \epsilon_{G_{1,1}} &= \frac{\partial d_1}{\partial u_1} + \frac{1}{2}\left[\left(\frac{\partial d_1}{\partial u_1}\right)^2 + \left(\frac{\partial d_2}{\partial u_1}\right)^2\right], \\ \epsilon_{G_{1,2}} = \epsilon_{G_{2,1}} &= \frac{1}{2}\left(\frac{\partial d_1}{\partial u_2} + \frac{\partial d_2}{\partial u_1}\right) + \frac{1}{2}\left(\frac{\partial d_1}{\partial u_1}\frac{\partial d_1}{\partial u_2} + \frac{\partial d_2}{\partial u_1}\frac{\partial d_2}{\partial u_2}\right), \\ \epsilon_{G_{2,2}} &= \frac{\partial d_2}{\partial u_2} + \frac{1}{2}\left[\left(\frac{\partial d_1}{\partial u_2}\right)^2 + \left(\frac{\partial d_2}{\partial u_2}\right)^2\right]. \end{aligned}$$

Here, the subscripts denote the vector components. As described in [32, p.64], this is equivalent to

$$\epsilon_G = \frac{1}{2}(J^T J - I)$$

---

[1] Since the clothes are given by planar patterns, we assume the rest state to be the identity mapping of the surface's parameterization. Moreover, we assume that the weft and warp directions in the clothes are aligned along the planar coordinate axes.

in terms of the deformation gradient $J = (\frac{\partial s}{\partial u_1}, \frac{\partial s}{\partial u_2})$.

This nonlinear strain measure, however, leads to high computational costs. Under the assumption that the displacement's variation is small, the nonlinear tensor can be linearized by dropping the quadratic terms, resulting in Cauchy's strain tensor

$$\epsilon_C = \begin{pmatrix} \frac{\partial d_1}{\partial u_1} & \frac{1}{2}(\frac{\partial d_1}{\partial u_2} + \frac{\partial d_2}{\partial u_1}) \\ \frac{1}{2}(\frac{\partial d_1}{\partial u_2} + \frac{\partial d_2}{\partial u_1}) & \frac{\partial d_2}{\partial u_2} \end{pmatrix}.$$

If we apply this linear strain measure to equation (2.1), together with a linear material law, we obtain a linear partial differential equation. Therefore, a finite element discretization leads to a system of linear equations and allows fast computations, if implicit integration for solving Newton's equation of motion is applied.

For the moment we assume that all forces and displacements are in the (e_1, e_2)-plane, and that the small displacement assumption holds. For this case, we state the finite element formulation for linear elasticity following [32, p.174]). We denote the linear basis functions of the linear finite element space by $N_a^m(u_1, u_2)$, $N_b^m(u_1, u_2)$, and $N_c^m(u_1, u_2)$ for each triangle T_m with vertices a, b, and c. We compute the form factors $\frac{\partial N_j^m}{\partial u_i}$ for $j = a, b, c$ and $i = 1, 2$ (these form factors replace the rest lengths of the springs in a mass-spring system), as well as the triangle areas Ω_{T_m} in the rest state. Then we construct the stiffness matrix as follows. For each triangle T_m and each vertex pair (a, b) of this triangle, we define a submatrix $K_{ab}^m \in \mathbb{R}^{2\times 2}$

$$[K_{ab}^m]_{ij} = \left(\sum_{k,l=1}^{2} \frac{\partial N_a^m}{\partial u_i} C_{ikjl} \frac{\partial N_b^m}{\partial u_j} \right) \Omega_{T_m}, \tag{2.3}$$

where C is the elasticity tensor.

The stiffness matrix $A \in \mathbb{R}^{3n\times 3n}$, where n is the number of vertices, is then assembled from all matrices K_{ab}^m for all triangles. It consists of $n \times n$ submatrices that are computed as follows:

$$[A]_{ab} = \begin{pmatrix} \begin{bmatrix} \sum\limits_{m|T_m\in\Gamma(a,b)} K_{ab}^m \end{bmatrix} & \begin{matrix} 0 \\ 0 \end{matrix} \\ 0 \quad 0 & 0 \end{pmatrix} \in \mathbb{R}^{3\times 3}$$

Here $\Gamma(a, b)$ is the set of all triangles containing vertices a, b. We denote the displacement, position, and velocity vectors that store the values of all n vertices by $d, x, v \in \mathbb{R}^{3n}$, respectively. The force vector is then computed by

$$f(x) = Ad = A(x - \bar{x}),$$

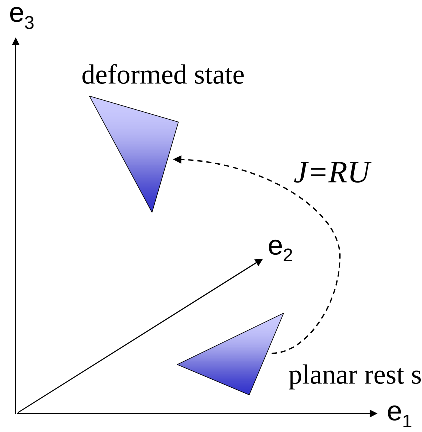


Figure 2.3: The deformation gradient J can be decomposed into a rotation R and a pure in-plane deformation U by a polar decomposition.

where $\bar{x}$ is the vector of positions in the rest state. This force formulation is valid as long as the surface remains in the (e_1, e_2)-plane.

2.3.2 Rotated Finite Elements

The problem of applying Cauchy's strain tensor for cloth simulations is that it is not invariant under rigid body rotations of the deformable object. This can be seen from the following example. Consider an object in the plane which is not deformed but simply rotated around the origin by some angle α, hence

$$s\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}.$$

Then, the displacement is given by

$$d\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{pmatrix} u_1\cos\alpha - u_2\sin\alpha - u_1 \\ u_2\sin\alpha + u_2\cos\alpha - u_2 \end{pmatrix},$$

and Cauchy's strain tensor yields

$$\epsilon_C = \begin{pmatrix} \cos\alpha - 1 & 0 \\ 0 & \cos\alpha - 1 \end{pmatrix},$$

which leads to incorrect non-zero forces for any angle $0 < \alpha < 2\pi$.

This rotational variance prevents the direct application of Cauchy's strain tensor for textiles, because cloth bends very easily, such that parts of the surface may undergo large relative rotations.

Hence, in each time step we are going to remove these rotations for each triangle separately before computing the deformation forces. This is equivalent to constructing a rotated rest state and linearising with respect to this rest state.

By a polar decomposition, the deformation gradient J can be decomposed into a rotation R and a pure deformation U (cf. figure 2.3)

$$J = R \cdot U.$$

The polar decomposition is carried out as described in algorithm 2 (cf. [32, p.68ff]).

Algorithm 2 Polar decomposition of the deformation gradient J

```
1: Compute $U^2 = J^T J$.
2: Compute the eigenvalues $\lambda_1, \lambda_2$ and eigenvectors $v_1, v_2$ of $U^2$.
3: Compute $U = \sqrt{\lambda_1} v_1 v_1^T + \sqrt{\lambda_2} v_2 v_2^T$.
4: Compute $R = JU^{-1}$.
```

In the previous example of a pure rotation, the deformation gradient J coincides with the rotation R, and we obtain $U = id$, and thus a displacement of zero, which leads to a correct force computation.

Since we use linear finite elements, the matrix J is constant on each triangle. Hence, the polar decomposition gives us a unique rotation R^m for each triangle T_m. This rotation is used to move the deformed triangle back to the (e_1, e_2)-plane, compute the elastic in-plane forces with linear elasticity there, and finally rotate the computed forces back to 3D space. Putting it together, one local stiffness matrix for edge (a,b) computes forces as follows:

$$f(x) = R^m K^m_{ab} (R^m)^T x - R^m K^m_{ab} \bar{x}.$$

Note that the planar rest state position $\bar{x}$ is already in 2D space and only the results of the multiplication need to be rotated to 3D space. Hence, we define the transformed local stiffness matrices as

$$\begin{aligned} Q^m_{ab} &= R^m \cdot K^m_{ab} \cdot R^T_m, \\ \bar{Q}^m_{ab} &= R^m \cdot K^m_{ab}, \end{aligned}$$

where $R^m \in \mathbb{R}^{3\times 2}, Q^m_{ab} \in \mathbb{R}^{3\times 3}, \bar{Q}^m_{ab} \in \mathbb{R}^{3\times 2}$ The global stiffness matrices are then assembled from the matrices Q^m_{ab} and $\bar{Q}^m_{ab}$:

$$[A]_{ab} = \sum_{m|T_m \in \Gamma(a,b)} Q^m_{ab}, \tag{2.4}$$

$$[\bar{A}]_{ab} = \sum_{m|T_m \in \Gamma(a,b)} \bar{Q}^m_{ab}.$$

The global force vector is computed by

$$f(x) = Ax - \bar{A}\bar{x}.$$

So far, we have considered elastic forces only. Therefore, we add viscous forces due to internal friction in the cloth by computing a viscous stress

$$\sigma^\nu = D\nu,$$

in analogy to the elastic stress in equation (2.2), where we replaced the elasticity tensor C by the viscosity tensor D, and the strain ϵ by the strain rate

$$\nu = \frac{d\epsilon}{dt}.$$

The viscous stress is added to the purely elastic stress. We model the damping forces by applying a viscosity tensor D which is constant and proportional to the elasticity tensor C. The finite element formulation is the same as in the elastic case in equation (2.3) except for the viscosity tensor that replaces the elastic tensor, i.e.

$$\left[\hat{K}^m_{ab}\right]_{ij} = \left(\sum_{k,l=1}^{2} \frac{\partial N^m_a}{\partial u_i} D_{ikjl} \frac{\partial N^m_b}{\partial u_j}\right) \Omega_{T_m}.$$

The transformed matrices $\hat{Q}$ are computed as in the elastic case and added to a global matrix B:

$$[B]_{ab} = \sum_{m|T_m \in \Gamma(a,b)} \hat{Q}^m_{ab}$$

These viscous forces are evaluated by

$$f(v) = Bv.$$

2.3.3 Bend Forces

In the last section, we have only treated elastic and viscous in-plane forces. It remains to add forces to model the bending resistance of textiles. Here, we use the Laplacian operator as bending measure and project it onto the surface normal direction to obtain a force which is orthogonal to the surface, given by

$$f(x) = P\Delta P x,$$

where $P = nn^T$ is the projection operator constructed from the surface normal n. The Laplacian operator is represented by its finite element approximation (cf. [33, p.54f]) by

$$\tilde{K}^m_{ab} = \begin{pmatrix} \frac{\partial N^m_a}{\partial u_1} B_1 \frac{\partial N^m_b}{\partial u_1} & 0 \\ 0 & \frac{\partial N^m_a}{\partial u_2} B_2 \frac{\partial N^m_b}{\partial u_2} \end{pmatrix} \Omega_{T_m},$$

where B_1 and B_2 are the bending moduli in the weft and warp directions, respectively.

These matrices are transformed by the rotations and a projection is carried out before and after the force computation:

$$\tilde{Q}^m_{ab} = P R^m \tilde{K}^m_{ab} (R^m)^T P.$$

It remains the question how to compute the surface normal. Since we cannot compute curvature on a flat triangle, n must not be the face normal. Hence we project the Laplacian onto all vertex normals of the triangle vertices by

$$\tilde{Q}^m_{ab} = \frac{1}{3} \sum_{i=a,b,c}^{3} P^m_i R^m \tilde{K}^m_{ab} (R^m)^T P^m_i,$$

where the projection operators $P^m_i = n^m_i (n^m_i)^T$ are composed of the three vertex normals n^m_a, n^m_b, n^m_c.

These local stiffness matrices are combined to a global matrix $\tilde{A}$ as in equation (2.4). Note that $\tilde{A}d = \tilde{A}x$ because the rest state is planar. Thus bend forces can be evaluated as

$$f(x) = \tilde{A}x.$$

The vertex normals depend on the position vector x as well as the rotation matrices R^m. However, these quantities are fixed during one time step such that the bend forces can be evaluated by the linear expression $\tilde{A}x$.

Analogously to viscous in-plane forces, viscous bend forces have to be integrated as well and added to the matrix B.

2.3.4 Time Integration

Applying the linear force computations for viscoelastic in-plane and bend forces presented in the last sections, Newton's equation of motion yields

$$M\frac{d^2x}{dt^2} = Ax - \bar{A}\bar{x} + \tilde{A}x + Bv + f_{\text{ext}},$$

where M is the mass matrix. As in most computer animation applications, we use a diagonal mass matrix, i.e. all masses are concentrated at the vertices. The mass of a vertex is computed as the Voronoi area [46, p.145ff] of that vertex multiplied by the mass density of the material. The forces f_{ext} consist of gravity and arbitrary other external forces, for instance due to aerodynamic effects as described in section 2.4.

In a preprocessing step, before the simulation loop starts, the form factors and areas of each triangle have to be computed. Then, in each time step of the simulation, we first have to compute the rotations R^m for each triangle T_m. After that, we construct the matrices $A, \tilde{A}, B$, and compute $\bar{A}\bar{x}$. Finally, we solve the system of equations which we obtain from an implicit time integration method.

Since the force computations are linear, the system of equations to be solved in each time step of an implicit integration scheme is linear. Therefore, it can be solved rapidly, for instance by a conjugate gradient method [26, 110]. For the implicit Euler method, for instance, we have to solve

$$\begin{aligned} v(t+h) &= v(t) + hM^{-1}((A+\tilde{A})x(t+h) + Bv(t+h) - \bar{A}\bar{x} + f_{\text{ext}}) \\ x(t+h) &= x(t) + hv(t+h) \end{aligned} \tag{2.5}$$

with time step h. From this, we obtain the linear system

$$\begin{aligned} &(M - h^2A - h^2\tilde{A} - hB)v(t+h) = \\ &Mv(t) + h(Ax(t) + \tilde{A}x(t) - \bar{A}\bar{x} + f_{\text{ext}}). \end{aligned} \tag{2.6}$$

The resulting linear finite element cloth simulator is outlined in algorithm 3. As described in section 2.2, we start the simulation with gravity turned off until the sewing process has finished. For collision detection, we use k-DOP hierarchies, and collision response is done with constraints [26]. As collision handling is not a topic of this work, we refer to the work presented by Mezger and Kimmerle [93, 9] for more details on this subject.

Algorithm 3 Cloth simulation with linear finite elements

```
Compute form factors and triangle areas in the planar state for each cloth pattern separately.
Place the cloth patterns around the avatar.
Merge patterns along the seam lines using algorithm 1.
loop
    Compute the rotations $R^m$ for each triangle $T_m$ using algorithm 2.
    Construct the matrices $A, \tilde{A}, B$ and compute $\bar{A}\bar{x}$.
    Compute the new velocities by solving equation (2.6) using the conjugate gradient method
    Compute the new positions by evaluating equation (2.5).
    Detect and correct collisions.
    if reached frame interval then
        Visualize the deformed mesh or write it to a file.
    end if
end loop
```

2.3.5 Material Parameters of Textiles

Woven textiles are orthotropic materials. This means, they possess two distinguished directions, namely the weft and warp directions of the threads in the cloth. Therefore, separate material parameters for these two directions have to be taken into account.

We obtained the elastic material parameters from Kawabata [86] measurements[2]. In particular, separate measurements for the two Young moduli in weft and warp direction C_{1111} and C_{2222}, the shear modulus $C_{1212} = C_{2121} = C_{2112} = C_{1221}$, and the bending moduli B_1 and B_2, which describe the curvature elasticity in weft and warp, have been carried out. We linearized the plots and thus estimated the linear elastic constants. The Poisson coefficients which control the parameter for transversal contraction C_{1122} cannot be estimated from the Kawabata experiments, and we chose a value of 0.2 in all experiments since this value is close to the numbers used in [78, 128]. The elastic material parameters for several materials are shown in figure 2.4. For each elastic parameter there is a corresponding viscous parameter that controls the respective damping. These viscous material parameters were chosen to be a constant fraction of the corresponding elastic parameter.

Note that these material properties are certainly only a coarse approxima-

---

[2] The measurements were carried out by the *Hohensteiner Institute* within the scope of the *Virtual Try-On* project described in chapter 4.

	Wool/ Viscose	Wool	Polyester/ Polyacrylics/ Acetate	Polyester	shear resistant material	bend resistant material
ρ $[kg/m^2]$	0.23	0.26	0.17	0.26	0.2	0.2
C_{1111} $[N/m]$	245	866	3057	2400	2000	2000
C_{2222} $[N/m]$	366	1391	1534	3600	2000	2000
C_{1212} $[N/m]$	0.38	0.51	1.22	5.23	1000	5.0
C_{1122} $[N/m]$	61.1	225.7	459.1	600	400	400
B_1 $[10^{-4}Nm]$	0.013	0.137	0.055	0.371	100	100
B_2 $[10^{-4}Nm]$	0.037	0.135	0.092	0.480	100	100

Figure 2.4: Material parameters: mass density (ρ), stretch moduli for weft (C_{1111}) and warp (C_{2222}), shear modulus (C_{1212}), transversal contraction (C_{1122}), and bend moduli for weft (B_1) and warp (B_2).

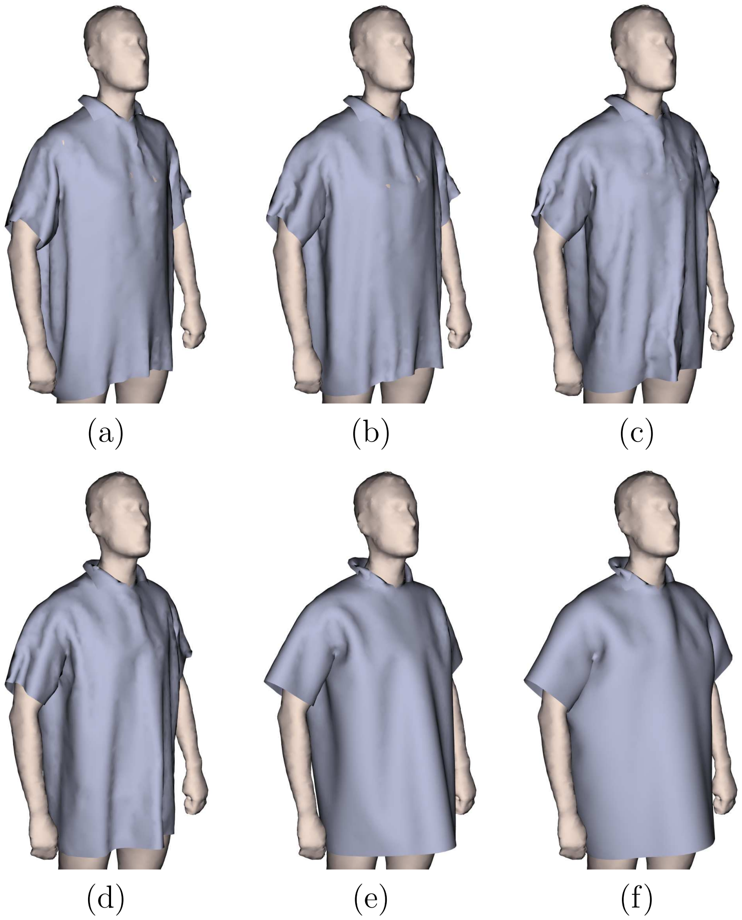
tion because only the elastic parameters have been measured and the force functions have been linearized. However, they are sufficiently accurate to show the different behavior of the materials, as shown in the next section.

2.3.6 Results

First, we demonstrate the quality of our physical model by the example of a man's shirt, which is assembled from 8 patterns with 17 seams. The shirt consists of 8,300 triangles. The simulation is run with six different materials in a static scene. Four materials are created from measured data, while the remaining two are designed to show the effect of high shearing and bending resistance. These examples demonstrate the intuitive control obtained by valid material parameters. The material properties are listed in figure 2.4 and the corresponding simulation results are shown in figure 2.5. The collars in all images consist of the bending resistant material shown in figure 2.5(f).

Next, we show the application of our model to animated avatars in figure 2.6. A walking man wears a shirt consisting of 6,711 and trousers consisting of 6,360 triangles. Here, the shirt is made of the wool/viscose material and the trousers are made of polyester (see figure 2.4).

The computational complexity of this finite element simulation is similar to mass-spring systems. The main difference consists in the need to compute the polar decomposition of the deformation gradient for each triangle in each time step. When using a large error tolerance in the conjugate gradient method, this can take as much time as the solution of the linear system.



(a) (b) (c)

(d) (e) (f)

Figure 2.5: A shirt simulated with different material properties: (a) wool/viscose, (b) wool, (c) polyester/polyacrylics/acetate, (d) polyester, and fictitious clothes with high (e) shear and (f) bend resistance.

Since second-order methods like BDF(2) and implicit midpoint-rule have been successfully applied to particle system simulations [69, 68], we also evaluated these methods for this finite element model. However, we obtained the best results with the implicit Euler method, which has been used to generate all examples given in this section. As mentioned above, collision detection is done via a k-DOP hierarchy as described in [93, 9], and collision response is implemented by a constraint mechanism as in previous particle systems.

All results were computed as 25Hz animations on a Pentium 4 with 1.7 GHz. The shirts in figure 2.5 were computed with a time step of 0.02s and took an average of less than 3s per animation frame excluding the time

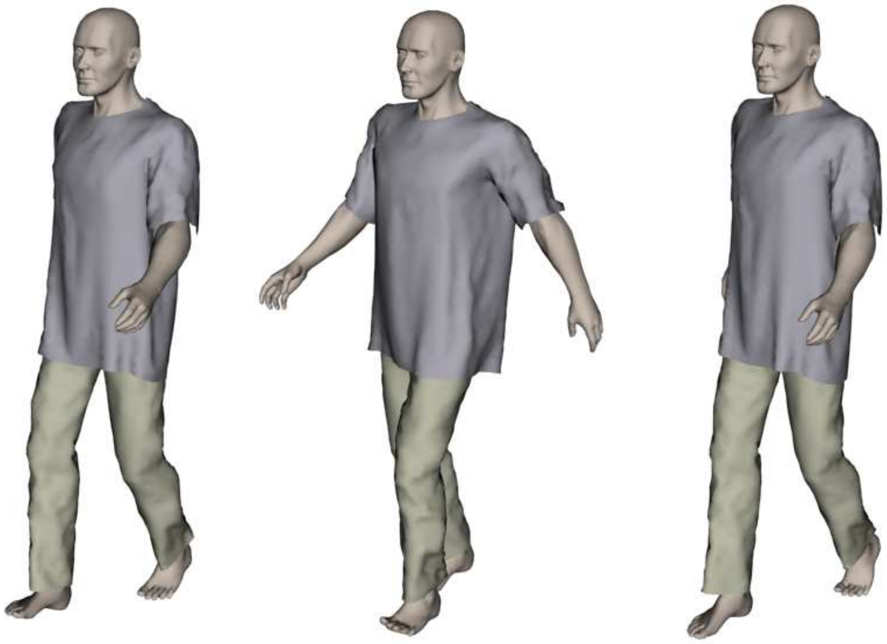


Figure 2.6: The images show frames of a cloth animation in a dynamic scene. A walking man is wearing a shirt made of wool/viscose and trousers made of polyester.

for collision handling. For the computation of the scene with a walking man wearing a shirt and trousers, we had to reduce the time step to 0.005s in order to guarantee a stable collision response. It took an average of 16s for the shirt and an average of 21s for the trousers per animation frame excluding the time for collision handling. The computational work varies according to the material stiffness, this explains the longer computation time for the stiffer trousers and is in accordance with the analysis of integration methods given in [129]. If measured material properties are not of interest, smaller values for C_{1111} and C_{2222} allow to reduce the computational work significantly.

2.4 Modeling Air Resistance and Wind

In this section we present two different wind field models and show how they can be used to model aerodynamic effects on textiles. The first model is based on Stam's work [115, 58, 116, 117] and calculates a numerical approximation of the Navier-Stokes equations. We extend the work presented by Stam by computing the interaction of the wind flow with textiles. The second model employs precomputed wind flows and particle tracing methods. We will show that even with this much simpler approach, realistic effects of wind fields on textiles including lee effects can be produced.

In this section, we follow parts of [6].

2.4.1 Aerodynamics

To incorporate wind effects in a physically based animation we have to apply additional external forces in the dynamical model of the deformable objects. Hence, given a wind flow represented by a velocity field in the scene we calculate the forces which are exerted on the simulated objects. In the following, we briefly describe the model we use to compute the effective aerodynamic forces such as wind force and air resistance, mainly following [111], before we propose two different approaches to model the actual wind flow in the next sections.

The wind force acting on objects in an air stream is decomposed into two components: the lift force F_L and the drag force F_D (see figure 2.7).

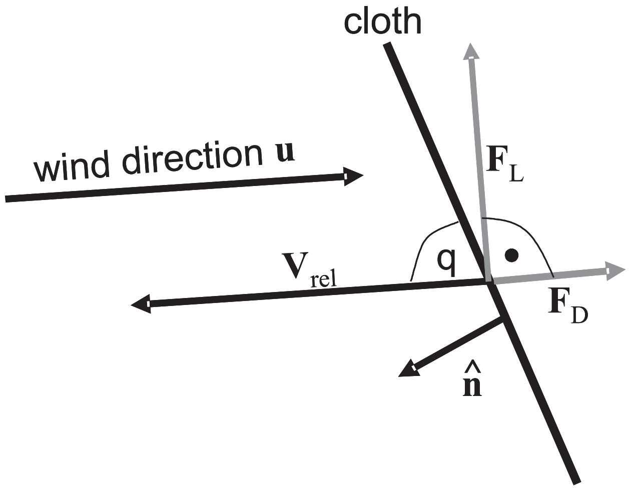


Figure 2.7: The decomposition of wind forces (side view).

The direction of the drag force F_D is diametral to the relative velocity $v_{rel} = v_{object} - u$, where v_{object} is the object's velocity and u is the wind velocity. Note that in the case of a windless situation, i.e. $u = 0$, we still have air resistance for moving objects. Since two-dimensional objects do not exhibit an inside and outside, the unit normal $\hat{n}_i$ of the i-th face of the object mesh (cf. figure 2.7) is replaced by

$$\tilde{n}_i = \begin{cases} \hat{n}_i & \text{if } n_i \cdot v_{i,rel} > 0 \ , \\ -\hat{n}_i & \text{else} \ . \end{cases}$$

The drag force per face is then given by

$$F_{i,D} = \frac{1}{2} C_D \rho |v_{i,rel}|^2 \Omega \cdot (\tilde{n}_i \cdot \hat{v}_{i,rel}) \cdot (-\hat{v}_{i,rel}) \ ,$$

where C_D is the specific air resistance coefficient, ρ the density of air, Ω the area of the corresponding face, and $\hat{v}_{i,rel}$ the unit relative velocity vector of the face.

The direction of the lift force, which is perpendicular to $v_{i,rel}$ and lies in the plane spanned by $v_{i,rel}$ and $\tilde{n}_i$, is given by

$$\hat{u}_i = (\tilde{n}_i \times \hat{v}_{i,rel}) \times \hat{v}_{i,rel} \ .$$

Then the lift force is calculated as

$$F_{i,L} = \frac{1}{2} C_L \rho |v_{i,rel}|^2 A \cos\theta \cdot \hat{u}_i \ ,$$

where C_L is the lift force coefficient, and θ is the angle between $v_{i,rel}$ and the respective face.

2.4.2 Wind Field Model I — The Navier-Stokes equations

The Navier-Stokes equations describe a precise mathematical model for fluid flows. The numerical algorithms used in computational fluid dynamics to solve these equations are designed for physical accuracy for engineering applications and are expensive in computation. But in our case, where this precision is not necessary, simplifications can be made which greatly reduce the computation costs. Thus, the wind is modeled as an incompressible fluid with constant density, described by the incompressible Navier-Stokes equations [41]:

$$\nabla \cdot u = 0, \tag{2.7}$$

$$\frac{\partial u}{\partial t} = -(u \cdot \nabla) u - \frac{1}{\rho} \nabla p + \nu \nabla^2 u + f_{\text{ext}}. \tag{2.8}$$

Here, u describes the (three-dimensional) velocity field, ν is the kinematic viscosity of the fluid, ρ its density, p the pressure in the wind field, and f_{ext}

accounts for external forces that act on the fluid. Equation (2.7) states that the velocity field should be incompressible while equation (2.8) describes the evolution of a velocity field over time. The first term on the right hand side reflects the change of velocity due to advection, while the second expression accounts for any external forces and acceleration caused by the local pressure gradient ∇p and by viscous drag depending on ν.

2.4.2.1 Approximating the Navier-Stokes equations

In order to numerically compute an approximation to the Navier-Stokes equations they first have to be discretized. For this, the computational domain is diced up into equally sized cubes forming a grid, and sample values of velocity and pressure are defined at the cell centers. Foster and Metaxas [59] use a finite difference approximation for the discretization of the operators in equation (2.8). Then they update the cell's velocities using an explicit integration scheme. Since time steps in explicit computations usually need to be very small, we follow the recent work presented by Stam [115, 58, 116, 117], who proposes an implicit integration scheme, which allows stable simulations with large time steps. While the linear terms in equation (2.8) are straightforward to solve implicitly, the term $-(u \cdot \nabla)u$ is nonlinear and deserves special attention. Here, an approach based on the method of characteristics is used to solve the advection equation [115]. Equation (2.8) does not provide a divergent-free velocity field. Therefore, using a Helmholtz-Hodge decomposition the divergence of each cell in the grid is projected to zero [117].

2.4.2.2 Wind effects on clothes

The major advantage of Navier-Stokes based approaches consists in the fact that the evolution of the wind flow over time is calculated. It enables us to model global effects like convection and diffusion on a physical basis. We present a model to exploit these wind models for calculating the interaction of deformable objects with the air flow by a boundary condition method. While rigid objects like walls will influence the fluid field but will not be affected by fluid forces themselves, deformable objects like cloth are supposed to both experience fluid forces and itself influence the fluid flow.

To prevent wind from flowing through objects, we require the Neumann boundary condition

$$\frac{\partial u}{\partial n} = 0$$

to be satisfied for the wind flow u at any boundary point of an object with normal n. This turns out to be a problem in the context of the aerodynamic model we want to apply. Consider a point on the boundary of a deformable object in the scene. On the one hand, we want the Neumann boundary condition to be satisfied. On the other hand, the wind velocity orthogonal to the object's surface is just what causes the aerodynamic forces, hence, simply setting the boundary according to the Neumann condition would mean that the fluid will not exert forces on the objects.

Therefore, we propose a method which meets both requirements as follows. For every deformable object the velocity value of the surrounding wind field for every vertex of the representing mesh is tracked. In the simulation loop, the boundary conditions are then set prior to any other operation: For every object in the scene, each triangle of its representing mesh is registered in the fluid grid, which means that the cell of the wind field occupied by the object is marked as occupied, and the wind velocity at the vertex positions of the object is recorded. Additionally, the normals of these vertices are stored. Then, the aerodynamic forces as described in section 2.4.1 are calculated. Finally, for every marked cell in the scene the previously stored normals are averaged in one space cell which are used to update the velocity at the cell to satisfy the Neumann boundary condition. Thus, the boundary conditions are met and yet aerodynamic forces are obtained from the flow field as described in sections 2.4.1.

In order not to have wind inside rigid objects, the path of the wind flow can be checked for object intersection. The collision detection of the cloth simulation system provides a simple method to deal with this issue. We simply have to detect which grid cells collide with the interior of rigid objects and set the respective velocity to zero.

2.4.3 Wind Field Model II — Particle Tracing

In this section, we combine the idea of creating global wind fields by predefined flow primitives with particle tracing in given flows. To define a wind scene we first build up the air flow by simple primitives such as parallel directed wind fields, vortices, etc. Then, we trace particles along the trajectories of each flow primitive and detect collisions between these particles and objects in the scene in order to detect windless areas and thus model the effect of lee. This method is very easy to implement and yields very plausible results.

2.4.3.1 Global Wind Fields

A simple approach to generate complex air flows is to define a wind field by mathematical functions which assign to each point in space a unique velocity value. As Wejchert [133] has shown, the combination of simple flow primitives already enables an animator to design complex wind fields. Some primitives common to fluid simulations are depicted in figure 2.8.

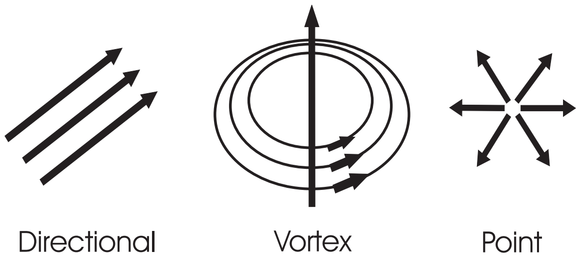


Figure 2.8: Flow primitives.

The approach to model solid objects in the scene taken by Wejchert consists in placing a wind source using a mirror principle in order to extinguish the air flow at the boundary of the object. While this works for simple objects, this model cannot handle objects exhibiting complex boundaries, and clearly this approach is not feasible with deformable objects like textiles. Another drawback of this model consists in the fact that the wind flow defined by the primitives will not react on objects in the scene. This means that tissues in the lee of other objects will be incorrectly affected by the wind flow just as objects that are fully exposed to it.

In order to solve the described problems, we propose a model which combines the simple global wind flow techniques with a particle tracing method. *Wind particles* are moved along the wind field to determine the effect of objects in the scene. This method can again be combined with the aerodynamic model described in section 2.4.1 in order to compute the external forces that act on the textiles.

2.4.3.2 Tracing Wind Particles

Similarly to the Navier-Stokes equations model, we first divide the scene into cube cells. There are two common approaches to discretising the continuous velocity field defined in space: one can either choose the midpoint of a cell

[115] or its six faces [59] to define the field. As usual, values between the defining points of the grid are interpolated using trilinear functions.

The basic idea of the particle tracing method is to trace wind particles through a field $w = \sum_i w_i$ defined by linear superposition of wind sources corresponding to flow primitives with respective velocity fields w_i. The field w does not account for lee effects caused by objects in the flow. Therefore we compute the effective wind field u containing these effects as follows. In our model, every wind source is also a particle source. These particles form an uncoupled particle system which can be considered as a wind gust. The wind particles are emitted into the velocity field w_i of the corresponding wind source which is defined on a grid. The specific emission intervals and amounts depend on the properties of the flow sources.

In every time step, each particle in a wind gust moves along its velocity field w_i defined by the corresponding wind source. Notice that the movement of the particles in a wind gust is only affected by the wind source they belong to. The other wind sources have no effect on these particles. To calculate the wind particles' positions we use the explicit Euler integration scheme. Note that we don't need to apply implicit method's here, because there occur no large forces, and the simulation of the wind particle movement stays stable also with large time steps in an explicit time integration method.

For a wind particle this results in a path from position $p(t)$ at time t to a new position $p(t+h)$ after a time step h, where $p(t+h)$ is computed by

$$p(t+h) = p(t) + w_i(p(t), t) \ .$$

As a particle moves along its path in space, all grid cells colliding with the path are updated with the velocity of the associated wind source with respect to the position of the particle. The particle might cross several grid cells on its way during a single time step. If this is the case, the path of the particle has to be subdivided into parts not exceeding the size of a grid cell. This path is then tested for collisions with the objects in the scene. The velocity field u is then computed as

$$u = \sum w_i$$

for each grid cell separately, where w_i are all those wind sources whose particles have reached the cell.

If a collision is detected at position p_{col}, the normal of the colliding object $n_{obj}(p_{col})$ is determined and the velocity of the particle is set to

$$w_i(p_{col}, t + \Delta t) = w_i(p_{col}, t) - (n_{obj} \cdot w_i(p_{col}, t)) \cdot n_{obj} \ .$$

This assures that the velocity component of the resulting field u which is orthogonal to the collision object's surface at p_{col} is zero, i.e.

$$u(p_{col}, t + \Delta t) \cdot n_{obj} = 0 \ ,$$

and thus no flow propagates through the object.

The wind force effective on objects in the scene is then computed from the velocity field u. Since u is determined using the wind particles, every point p that could not be reached by any wind particle will hold zero velocity even if w may hold a nonzero velocity. Thus, this method solves the problems described in section 2.4.3.1.

Note that the somewhat tempting simplification of tagging each cell to either have wind in it or not is not valid. Imagine the simple scene in which there are two directional wind sources with opposite wind directions. Let them further have equal velocity magnitude and no distance attenuation. If we now place a solid object in between these two sources an undesired effect would occur using this simplification: on both sides of the solid object all cells would be tagged as having wind. But evaluating the wind field at every cell we would obtain a zero velocity. This is due to the extinguishing effect of the superposition of the two wind sources. Therefore, it is crucial for the particles to have the associated velocity of their wind source and not the velocity resulting from the global superposition of all wind sources.

2.4.4 Comparison

Regarding physical accuracy, the Navier-Stokes model as described in section 2.4.2 is far superior to the global wind field model described in section 2.4.3. Swirls and vortices derived from dynamical characteristics of the fluid are simulated, which is not possible with predefined global flows. However, the fluid solver is quite complex and using a high grid resolution is computationally expensive. The global wind field model is much simpler and better suited for situations where a flexible and easy to implement tool is needed. Particle systems and collision handling algorithms are very common in (cloth) simulation engines and most functionality can be easily adapted for the integration of the proposed global wind field model. Even with this straightforward approach, realistic looking results can be achieved which is illustrated in the next section. Depending on the application, a major advantage of the second model is the possibility to combine simple flow primitives to a complex scene. This allows to control the simulation, which is important if a certain animation effect shall be achieved.

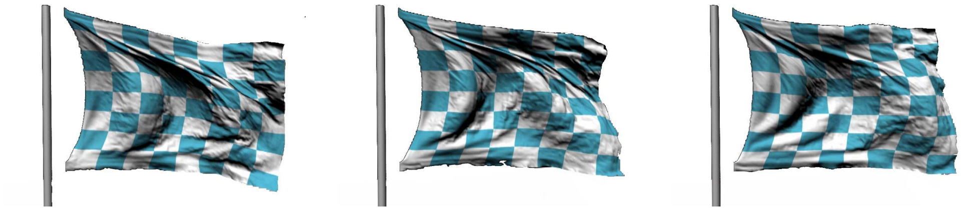


Figure 2.9: The images show frames of a flag blowing in the wind. The effects of the wind field on the cloth are computed with the particle tracing method.

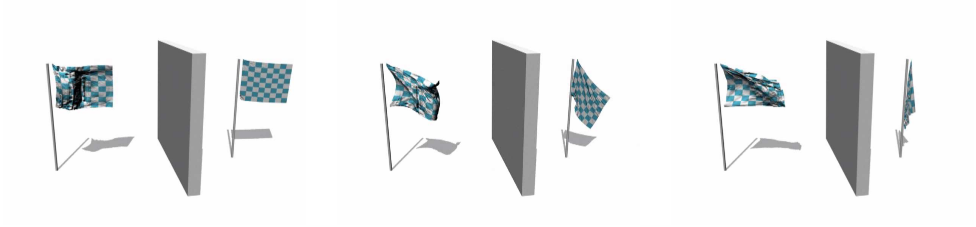


Figure 2.10: Two flags are simulated in a wind field coming from the left, computed by the Navier-Stokes equations method. The simulation starts with both flags unfolded. The wind is blocked by the wall, thus the right flag is in the lee.

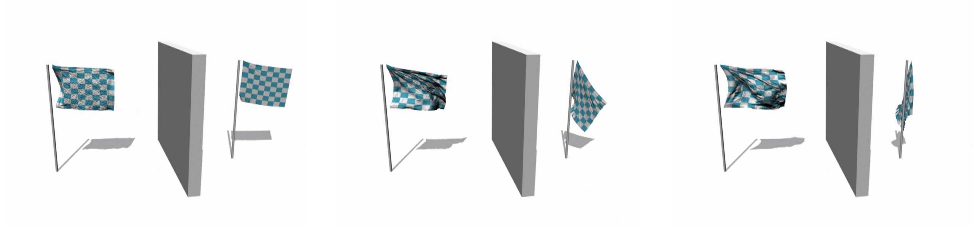


Figure 2.11: As in figure 2.10, two flags are blowing in the wind, here computed with the particle tracing method. Again, the wind is blocked by the wall, and the same lee effect as with the more complex Navier-Stokes equations method is obtained.

2.4.5 Results

In this section we present some practical results of the proposed methods. We computed the forces due to the wind models described in sections 2.4.2

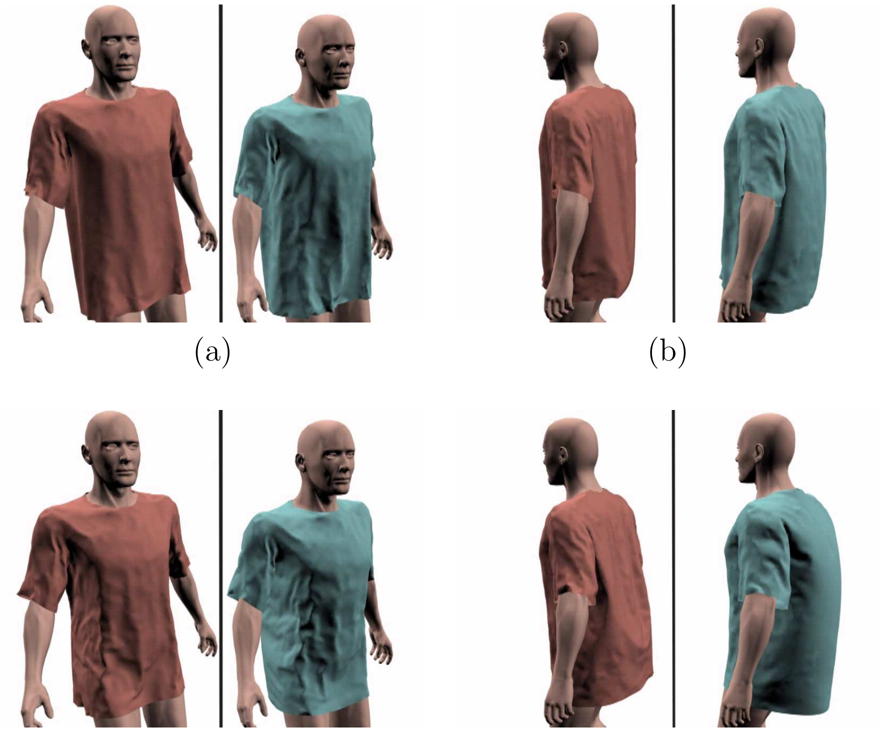
and 2.4.3, and integrated them as additional external forces into the finite element model described in section 2.3. Figure 2.9 shows a flag blowing in the wind. Here, the particle tracing method described in section 2.4.3 is used to model the effects of a directional wind field, coming from the left, on the flag. In figures 2.10 and 2.11 we show the ability of both the Navier-Stokes equations model (section 2.4.2) and the particle tracing method (section 2.4.3) to model the effect of lee. Two flags are exposed to a wind field, but the wind is blocked by a wall, so one of the flags is not affected by the wind. As can be seen, both models deal well with this situation and give similar results. This shows the particle tracing method's ability to simulate even complex wind flows. The images in figure 2.12 show a character wearing a shirt and standing in a wind stream coming from the front. Images (a) and (b) are snapshots from the beginning of the animation, images (c) and (d) show the result after the wind field has affected the clothes. To show the improved realism when simulating lee effects, we let the wind act on all polygons of the shirt on the right (no lee effect). For the shirt on the left we used the particle tracing method to simulate lee effects which, we think, considerably improves the realism of the simulation result.

2.5 Summary

In this chapter we proposed solutions for the modeling of internal and external forces in cloth simulations. We have shown that finite elements can be used efficiently for the simulation of clothes. Our formulation models material parameters more accurately than standard particle systems and independent of the mesh topology. Thus, it allows a flexible modeling with unstructured triangle meshes, and we are able to assemble arbitrarily shaped CAD patterns and simulate the clothing with realistic material properties.

Moreover, we presented two models for including wind effects into cloth simulation systems. The first concentrates on physically accurate computations using a numerical approximation of the Navier-Stokes equations, the second model incorporates a particle tracing method for global wind fields. As illustrated, both methods produce visually convincing results which are capable of enhancing the realism of computer animations. While the first model has a wider range of applications, the second one provides a simpler method which still delivers realistic effects, and gives an animator better control of the results.

While the computational costs of the described physical model are in the same order of magnitude as for mass-spring and particle systems, it is



(a) (b)

(c) (d)

Figure 2.12: A man wearing a shirt and standing in a wind stream coming from the front. Images (a) and (b) are snapshots from the beginning of the animation, images (c) and (d) show the result after the wind field has affected the clothes. For a comparison, the simulation of lee effects has been disabled for the shirt on the right. For the shirt on the left the particle tracing method has been used to simulate lee effects which gives more realistic results.

still necessary to reduce computational costs, especially for high resolution cloth models. A promising approach to increase computational performance is parallelization. This will be the topic of the next chapter.

Chapter 3

Parallel Numerical Computation

Computational performance and stability of numerical time integration in physical cloth simulations have been considerably improved by implicit integration schemes, compared to explicit methods. Nevertheless, the computational demands for physical cloth simulations are still very high. Especially if we are interested in high quality simulations for high resolution textiles and relatively long animation sequences, the desired computational performance is not yet reached. Therefore, we aim at improving computation speed by parallelization.

In the following, we present a parallel implicit time integration scheme designed for distributed memory parallel architectures, in particular clusters built of commodity components. We focus on efficient parallel processing of irregularly structured and real-world sized problems typically occurring in the simulation of garments. Our performance measurements show a high degree of parallel efficiency and scalability indicating the usefulness of our approach.

In this chapter, we partially follow [4].

3.1 Introduction and Related Work

Implicit time stepping, together with the conjugate gradient method to solve the systems of linear equations, has become the standard method to solve the ordinary differential equations that arise in cloth simulations, e.g. [26, 128, 129, 39, 40], and we also employ it in this work (cf. section 2.3.4). While many variants and mixed implicit-explicit methods have been proposed, e.g. [53, 69, 92, 81, 43, 36, 22, 70, 68], current algorithms are still far from computing high resolution textiles (for example a garment with about 18,000 vertices, as stated in [27]) in real time. Moreover, if we consider geometrically more complex garments, we might want to use even higher resolutions.

Parallel computing is able to significantly improve the performance of computationally intensive problems. The major challenges of applying parallel techniques to cloth simulation are the fine granularity, the relatively small problem sizes, and the irregularity of the problem structure. The fine granularity originates from the fact that each iterative time step in the simulation depends on the results of the previous step. This property makes an inter-step parallelization approach not feasible. Consequently, parallelization has to be carried out at the level of an individual simulation step. Typical problem sizes in cloth simulations, in terms of the number of unknowns to be computed within each step, are rather small compared to similar parallel simulation applications from other domains, e.g. mechanical simulations with up to 40 million degrees of freedom [20], which leads to a poor computation to communication ratio. Moreover, we are dealing with arbitrary triangle meshes and thus an unstructured problem. Therefore, naive approaches for task decomposition and mapping typically lead to unscalable parallel applications [87, p.420ff]. In this chapter, we address theses challenges, and present a parallel cloth simulator for unstructured triangle meshes that allows to achieve a high parallel efficiency for reasonably sized cloth simulation problems.

To our best knowledge only a few contributions to parallel computing for cloth simulations have been made in the past. Romero [103] presents a parallel cloth simulation method based on implicit integration designed for parallel non-uniform memory architectures. Lario [88] reports on a rapid parallelization approach of a multilevel cloth simulator using OpenMP. The most recent contribution in this field has been made by Zara [137, 138]. This work deals with parallel cloth simulation on PC clusters employing both explicit and implicit integration techniques. Zara's work is the most related to our own both in terms of the employed numerical algorithms and in terms of the target parallel architecture. The other two approaches are based on shared address space parallel computers which are certainly more easy to program but do not scale well and/or have a worse price/performance ratio compared to clusters built of commodity components. The main difference between our approach and the work of Zara is the way problem decomposition and task mapping is carried out. While we perform a completely static approach based on data partitioning, Zara carries out dynamic problem decomposition based on partitioning dynamically generated task dependency graphs. This fundamental difference is also reflected by the underlying parallel programming models. Zara employs the Athapascan task-parallel language while our work is based on the Portable Extensible Toolkit for Scientific Computation (PETSc) which provides an extensive set of data parallel primitives on top of the message passing programming model.

3.2 Parallel Time Integration on Distributed Memory Architectures

In the following, we outline the sequential simulation loop that occurs in an implicit time integration scheme for cloth simulations, independent of the chosen physical model. Then we describe our approach to task decomposition and task mapping, and present the simulation loop of our parallel cloth simulator. Finally, we verify the parallel implementation by carrying out performance measurements on a PC cluster.

3.2.1 The Sequential Simulation Loop

As introduced in section 1.2, implicit time integration with the backward Euler method means to iteratively solve the equations

$$\begin{aligned} x(t+h) &= x(t) + hv(t+h) && (3.1) \\ v(t+h) &= v(t) + hM^{-1}f(x(t+h), v(t+h)), && (3.2) \end{aligned}$$

in order to iteratively obtain an approximate solution of Newton's equation of motion. Mostly, equation (3.2) is reduced to a linear equation system

$$Av(t+h) = b.$$

This system of linear equations can be solved efficiently by a conjugate gradient method (as we did for the linear finite element model described in the last chapter, cf. section 2.3.4), which is an iterative method itself, as shown in algorithm 4.

Of course, the specific computation of A and b depends on the physical model that is used. In most cases, internal forces are modeled locally between vertices in the mesh that are connected by an edge. This means, the internal forces acting on a vertex depend only on the positions and velocities of its neighbors. Thus, the resulting matrix A is sparse and has non-zero entries

Algorithm 4 Modified Conjugate Gradient Method to solve $Av = b$

```
1: $i \leftarrow 0$
2: $r \leftarrow b - Av$
3: filter($r$) {enforce constraints}
4: $d \leftarrow P^{-1}r$
5: $\delta_{\text{new}} \leftarrow r^T d$
6: $\delta_0 \leftarrow \delta_{\text{new}}$
7: while $\delta_{\text{new}}/\delta_0 > \varepsilon_{\text{tol}}^2$ do {for given error tolerance $\varepsilon_{\text{tol}}$}
8:     $q \leftarrow Ad$
9:     $\alpha \leftarrow \delta_{\text{new}}/(d^T q)$
10:    $v \leftarrow v + \alpha d$
11:    if $i = 0$ then
12:        $r \leftarrow b - Av$
13:    else
14:        $r \leftarrow r - \alpha q$
15:    end if
16:    filter($r$) {enforce constraints}
17:    $s \leftarrow P^{-1}r$
18:    filter($s$) {enforce constraints}
19:    $\delta_{\text{old}} \leftarrow \delta_{\text{new}}$
20:    $\delta_{\text{new}} \leftarrow r^T s$
21:    $\beta \leftarrow \delta_{\text{new}}/\delta_{\text{old}}$
22:    $d \leftarrow s + \beta d$
23:    $i \leftarrow i + 1$
24: end while
```

only in the form of 3x3 blocks beginning at row $3i$ and column $3j$, if there is an edge between vertex i and vertex j. Moreover, A is symmetric. The sparsity of the occurring matrices can be exploited very well by applying specialized efficient numerical linear algebra data structures and low-level algorithms for the matrix-vector multiplications in the conjugate gradient method by avoiding unnecessary multiplications with zero [87, p.409ff]. Additionally, the convergence rate can generally be improved by preconditioning (steps 4 and 17 of algorithm 4), where a suitable preconditioning matrix P has to be chosen [110, p.47f]. If no preconditioning shall be used, P can be chosen to be the identity matrix. For an in-depth discussion of the conjugate gradient method, we refer to [110].

As described in [26], the conjugate gradient method can be modified to realize constraints by using a filter function, e.g. for pinning vertices to specific points in the scene or for collision response. If a certain constraint

Algorithm 5 Sequential Simulation Loop

```
loop
    setup LES:
    Compute the matrix $A$ and the right hand side vector $b$.
    solve LES:
    Compute the new velocities by solving $Av(t+h) = b$ using the conjugate gradient method
    compute positions:
    Compute the new positions by evaluating $x(t+h) = x(t) + hv(t+h)$
    if reached frame interval then
        generate frame
    end if
end loop
```

shall be enforced on the movement of a vertex, the filter function simply has to modify the residuum of the respective vertex velocity accordingly (steps 3, 16 and 18 of algorithm 4). The filter function essentially projects the vertex velocity to the desired direction. For instance, if a vertex x_i shall not move in the vertical direction, we first set the vertical coordinate of v_i to zero before entering the conjugate gradient method. Then we do the same for the vertical coordinate of the residuum r_i in the filter function calls of each conjugate gradient iteration, such that the velocity in this direction remains zero.

Thus, the simulation loop in cloth simulations that employ an implicit Euler time integration scheme is structured as shown in algorithm 5.

First, the linear equation system (LES) has to be set up, i.e. the entries of the matrix A and the right-hand-side vector b have to be computed from the current vertex positions and velocities. Then, the new velocities are computed iteratively with the conjugate gradient method. Finally, the new positions are computed using equation (3.1). Mostly, there will be additional steps for collision detection and response (cf. algorithm 3 in section 2.3.4 of the last chapter).

In the following, we employ the physical model described in the last chapter to compute the linear equation system in each time step. However, all the presented methods apply also for arbitrary other models that compute internal forces based on neighboring vertices, such as conventional mass-spring and particle systems.

3.2.2 Design of a Parallel Cloth Simulator

We build our parallel cloth simulator on the basis of the physical model described in the last chapter and employ PETSc [25, 24] for enabling parallel execution. PETSc is a suite of parallel data structures and routines for the scalable parallel solution of scientific applications. It is based on the MPI (Message Passing Interface) standard [130] and supports an SPMD (Single Program Multiple Data) style of parallel programming which is located at a higher level of abstraction than the pure SPMD message passing programming model. PETSc has been used for parallelizing applications from a wide range of domains [23]. Besides ready-to-use standard components (e.g. parallel linear equation solvers), it also provides a rich set of lower-level primitives for dealing with advanced issues like the parallelization of problems on irregularly structured meshes.

The major design goals of our parallel cloth simulator are:

- Efficient parallel processing of irregularly structured problems typically resulting from the simulation of garments on unstructured triangle meshes.
- Achieving good parallel scalability on relevant problem sizes.
- Execution on cost-efficient distributed memory parallel architectures, in particular clusters built of commodity components.

We employ a data-parallel approach of parallel programming using static task decomposition and task mapping techniques to realize these goals.

3.2.3 Task Decomposition and Task Mapping

Both task decomposition and task mapping can be accomplished statically or dynamically. While dynamic approaches are more flexible and are also inescapable in some application domains, static schemes generally impose less parallel run-time overhead. Consequently, static approaches should be preferred whenever possible in order to achieve good scalability. This is especially important when dealing with relatively small problems which typically exhibit a poor computation to communication ratio.

We employ a static task decomposition and mapping scheme which is based on data decomposition. The goal of data decomposition is to partition the data structures on which computations are performed (which are essentially vectors and matrices in our case) in order to induce a decomposition into parallel tasks.

Generally, a partitioning method should optimize the following (commonly conflicting) objectives to attain high parallel efficiency: balancing of computational load, and minimizing communication overhead. Balancing computational load requires a task mapping that assigns to all processors the same amount of work. In our case in each step identical operations are carried out on individual data elements and all matrices are unstructured. Thus an even partitioning of the data structures, i.e. each compute node is responsible for the same amount of vertices, also leads to a balanced computational load. Moreover, this property makes it possible to limit the number of generated parts (and hence the number of induced parallel tasks) to the number of available processors. This results in a static one-to-one mapping between partitions, computation tasks, and processors, respectively. For matrices we choose a one-dimensional row-oriented parallel layout, vectors are aligned accordingly.

For minimizing inter-task communication it is not sufficient to generate any balanced partition, but we must also decide for each individual data element to which partition it should belong to according to data-dependencies occurring within its computation. Conceptually, this process delivers a new ordering of the data elements, called the parallel numbering. In the parallel numbering each processor owns a consecutive range of vertex numbers.

Below we identify all data dependencies within the simulation for a one-dimensional row-oriented parallel data layout:

- Setup LES
    - computation of the matrix and the right hand side requires communication of all non-local mesh neighbors.
- Solve LES (Conjugate Gradient Method)
    - dense vector inner product requires one all-reduce collective communication operation. Note that in general the communication overhead of parallel dense vector inner product does not depend on the ordering of the data elements [87, p.412f].
    - sparse matrix vector multiplication requires communication of non-local vector elements that correspond to non-zero entries of each matrix row [87, p.413ff].

Since the mesh structure is fixed and translates directly into the sparsity pattern of all involved matrices (non-zero entries indicate mesh neighbors), all task interaction patterns remain the same throughout the whole computation. Moreover, the Setup LES phase and the Solve LES phase can be optimized jointly.

In our case, we are dealing with unstructured sparse matrices, making specific partitioning schemes impossible [87, p.420ff]. Therefore, we employ generic graph partitioning techniques. The graph partitioning problem deals with determining a partitioning of a graph with equally sized parts and a minimum number of edge cuts for a given partitioning size. Since in our application edge cuts indicate inter-task communication, graph partitioning reduces the overall communication overhead and at the same time preserves a balanced workload.

3.2.4 The Parallel Simulation Loop

Algorithm 6 shows the pseudo code of our SPMD based parallel algorithm. Subsequently, we discuss the main steps of the algorithm and give some explanations how we have implemented the steps employing PETSc constructs.

Algorithm 6 SPMD based parallel algorithm

```
partition mesh
redistribute positions vector
loop
    update ghost values
    setup LES
    solve LES
    compute positions
    if reached frame interval then
        gather vector x
        if NODE-ID = 0 then
            generate frame
        end if
    end if
end loop
```

For mesh partitioning we use a parallel multilevel k-way graph partitioning method which is provided by the ParMetis [83, 84] graph partitioning library. PETSc features an interface for accessing ParMetis functionality in a straight forward manner. The result of this step is a so called *index set*

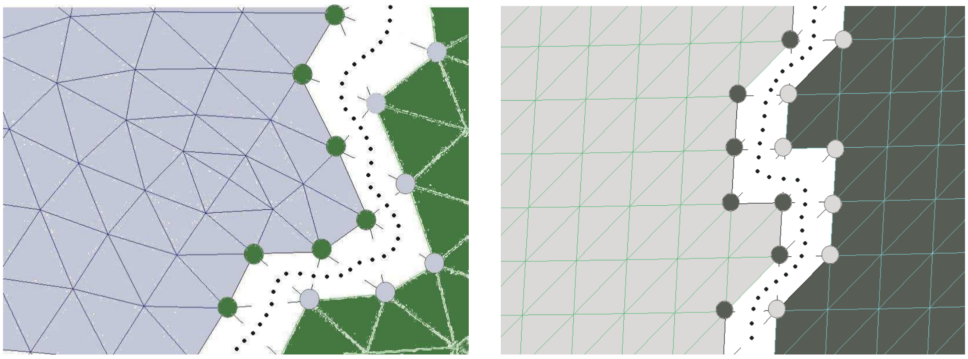


Figure 3.1: Examples of ghosted meshes. The left image shows a part of a shirt's triangulation, the right image is taken from a square piece of cloth (cf. section 3.2.5).

that represents a mapping between the application specific vertex numbering and the new parallel numbering induced by the partitioning process. Index sets are used for defining communication patterns of collective operations.

Before entering the parallel simulation loop, the initial positions vector has to be redistributed according to the determined parallel numbering. PETSc supports this process by means of its generic vector-scatter collective communication primitives.

Vertices that are adjacent of a cut edge (i.e. vertices located on the border of a partition) are accessed by both corresponding tasks during matrix setup and sparse matrix-vector multiplication steps. Such vertices are called ghost-points, because they physically belong to one, but logically belong to two processors. Figure 3.1 shows two examples of ghosted meshes. As PETSc is based on the message passing model, ghost-points have to be explicitly communicated by a collective communication operation at the beginning of each iteration of the simulation loop. Since this communication operation is highly performance critical, we use overlapping techniques. The part of the computation of the right hand side vector not depending on ghost vertices is carried out while the messages are in transit. In PETSc, overlapping of computation and communication is accomplished by placing code between calls of *VectorScatterBegin()* and *VectorScatterEnd()* primitives.

In order to realize a constraint enabled conjugate gradient method, we extended the parallel conjugate gradient component of PETSc by a filter-hook. This functionality allows us to register a custom hook function that is called within the conjugate gradient method at appropriate places providing access to internal variables which can be modified within the hook function applying a filter procedure.

For generating frames, the computed positions have to be gathered on one node and at the same time permuted to the original application specific numbering. This gathering and permutation process is accomplished by PETSc vector-scatter collective communication primitives.

3.2.5 Performance Measurements

In this section we first describe the test scenarios we used to evaluate our approach. Then we discuss the results of our measurements.

3.2.5.1 Test Scenarios

We verified our approach with two test scenes.

In the first test we simulated a quadratic piece of cloth under the influence of gravity (see figure 3.2). The cloth has a size of one square meter and consists of 22,500 vertices and 44,402 triangles. In the rest state, the vertices form a uniform 150x150 grid, where the vertical and horizontal edges have a length of $\frac{2}{3}$ cm each, while the diagonal edges have a length of about 1cm (see the right image in figure 3.1). We think this is an appropriate discretization if we want to model small folds and wrinkles.

The cloth is fixed at three points and slides onto the floor. To treat the collisions with the rigid floor, we implemented a very simple collision detection and response. At the end of each time step the vertical coordinate of each vertex is compared to the floor height. If a collision is detected we correct the vertex position and velocity. Obviously, this is straightforward to parallelize.

The second test scenario consists in a shirt which is fixed at two vertices (see figure 3.3). Here, the triangulation has no regular pattern, as can be seen in the left image in figure 3.1. The shirt consists of 31,046 vertices and 61,824 triangles[1].

Figures 3.4 and 3.5 show the cloth and the shirt meshes partitioned for 4 and 12 processors, respectively. The different partitions are indicated by randomly assigned colors.

---

[1] In [4], the numbers of vertices and triangles are given incorrectly as 35,024 and 69,648, respectively.

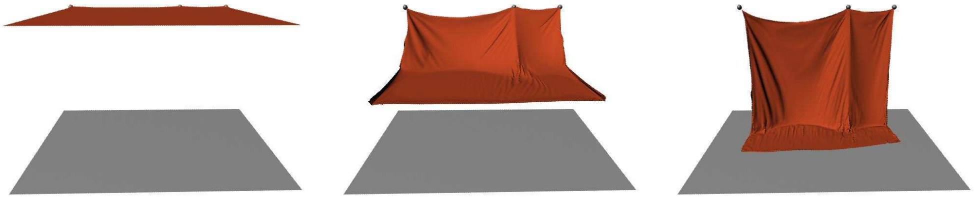


Figure 3.2: A piece of cloth consisting of 22,500 vertices is pinned at three points and dragged down by gravity until it slides onto the floor.

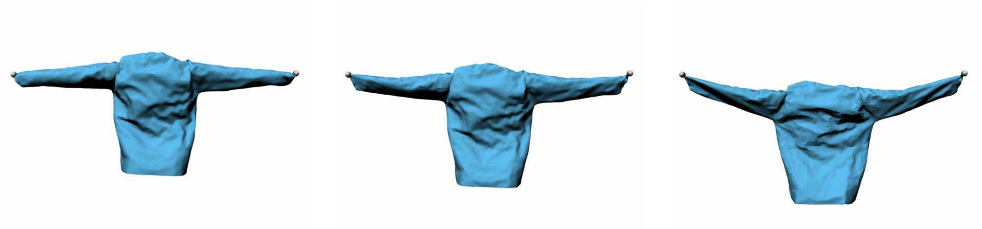


Figure 3.3: A shirt consisting of 31,046 vertices is pinned at two points and dragged down by gravity.

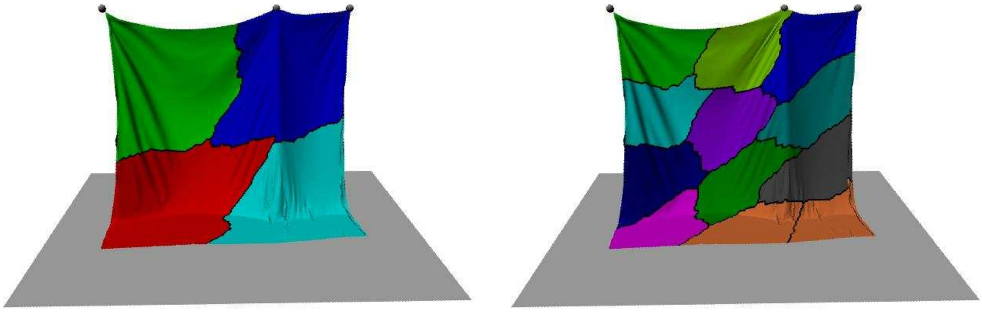


Figure 3.4: The piece of cloth partitioned for 4 and 12 processors.

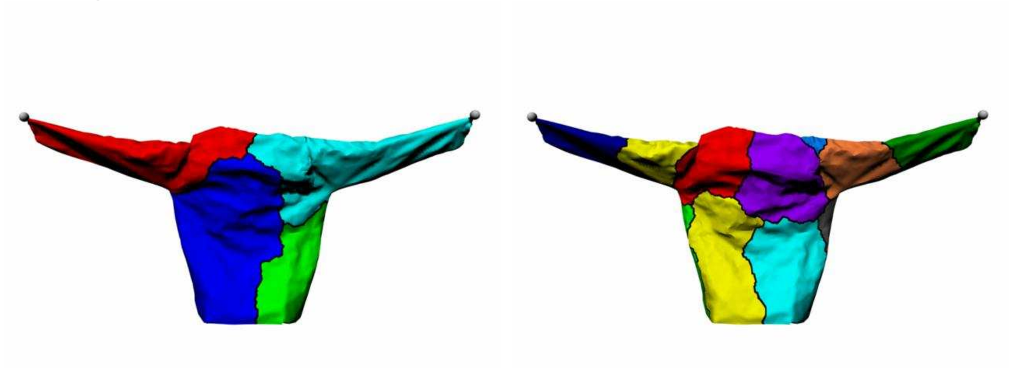


Figure 3.5: The shirt partitioned for 4 and 12 processors.

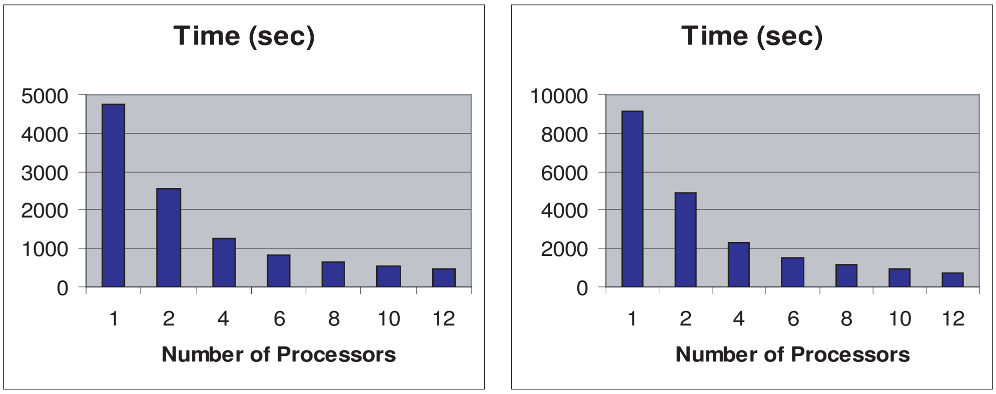


Figure 3.6: Computation times of the cloth and shirt scenes.

Both simulations were computed with time steps of $h = 0.001$s and with a relative error tolerance (relative decrease in the residual norm) of 0.001 in the conjugate gradient method, leading to an average of 647 iterations per time step for the square cloth and of 1,115 for the shirt. The first simulation ran for a simulation time of 0.48s and the second for 0.36s. Intermediate results were collected from all processor nodes and written to files at a rate of 25 frames per second.

These two test scenes are quite demanding for the numerical solver because in both examples there occur high internal forces in the textiles (note that there is no post-correction of the edge lengths undertaken as in [101]) and nearly all the vertices are subject to large relative movements for the whole simulation.

Of course, the overall computation time depends heavily on the specific physical configuration of the animated scene and on the numerical precision that shall be obtained. Changing the stiffness of the materials, setting up a scene where stronger or weaker forces in the cloth occur or changing the time step or the conjugate gradient error tolerance all influence the number of conjugate gradient iterations that have to be done, and possibly the visual results.

3.2.5.2 Results

For carrying out performance measurements we used a Linux based cluster. All compute nodes are equipped with Intel Xeon processors running at 2.667 GHz and with 2 GB of main memory. The nodes are connected by a Myrinet-2000 high-speed network.

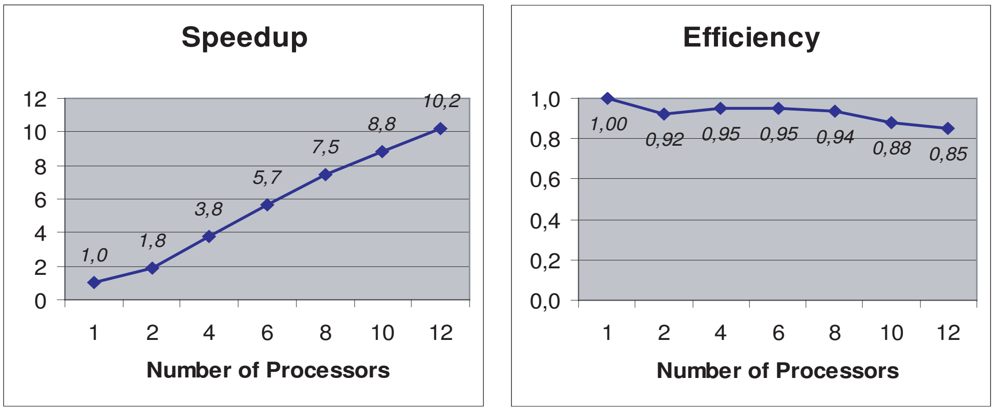


Figure 3.7: Speedup and efficiency for the cloth scene.

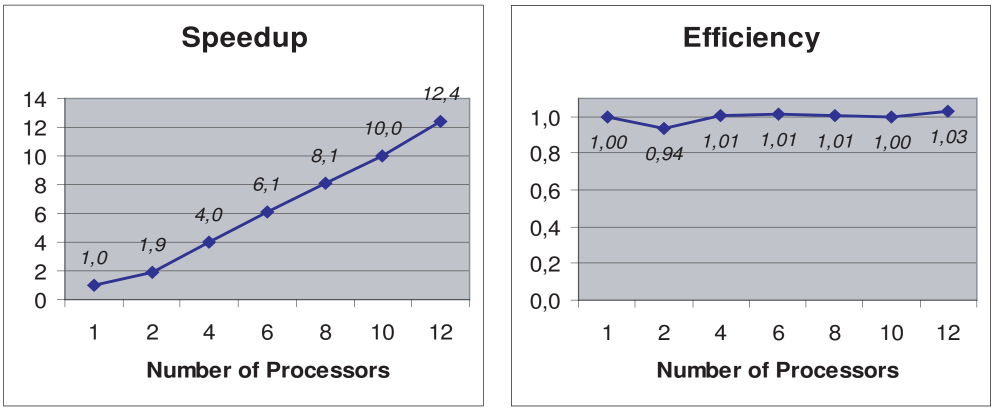


Figure 3.8: Speedup and efficiency for the shirt scene.

All presented performance results are based on the arithmetic mean of the wall-clock times of three individual program runs for each investigated setting. Figures 3.6, 3.7 and 3.8 show the results of the performance measurements for the cloth and the shirt scene. The time values given for one processor are based on a sequential version of our parallel simulator that employs sequential data structures and sequential arithmetic operations (PETSc is capable to automatically choose sequential primitives at run-time if only one processor is available). The results show that we achieve a high level of parallel efficiency preserving a high level of scalability in both test scenarios. For the shirt scene super-linear speedups could be observed. In data parallel applications the main source of super-linear speedups are memory cache effects. With an increasing number of processors the data working-set of each individual processor becomes smaller, often resulting in an increased cache hit rate.

3.3 Summary

We presented a parallel cloth simulation engine that is capable of substantially improving the computational performance of cloth simulations. In particular, we described a parallel realization of an implicit time integration scheme for cloth simulations on distributed memory architectures. We employed static task decomposition and mapping which is based on a partitioning of arbitrary unstructured triangle meshes. The performance measurements show that the employed methods scale well, indicating that parallel techniques are a promising approach to achieve high computational performance for high resolution cloth models.

Chapter 4

TüTex and Virtual Try-On

In this chapter we give an overview of the *TüTex* cloth simulation engine into which the methods described in the previous chapters have been integrated (section 4.2) and present two applications. First, the TüTex cloth simulator has been integrated into the standard modeling software Maya in form of the plugin *tcCloth* (section 4.3). Thus, it can be employed for a wide range of computer animation applications, while Maya provides modeling tools for the design of two-dimensional cloth patterns and a powerful graphical front-end for rendering the simulation results. Second, TüTex is the core part of an automatic virtual clothing pipeline which has been realized in the research project *Virtual Try-On* (section 4.4). Here, it is applied to calculate the drape and fit of individual made-to-measure garments on virtual human models.

4.1 Introduction and Related Work

Cloth simulation engines are part of many commercial modeling and animation software packages, e.g. Cinema4D (by Maxon), Poser (by Curious Labs), and Maya Unlimited (by Alias). These tools are mainly used to create computer animations for entertainment purposes. Moreover, cloth simulation techniques have become powerful enough to be used in applications relevant to textile industries, and virtual try-on of garments on avatars of real people is currently an area of active research [44, 12, 13, 10, 51, 126]. The idea is to support the customer's decision making by a realistic simulation and visualization of virtual garments, before the real garments are actually manufactured. Thus, reduced costs of stock keeping for both clothing manufacturers and retailers on the one hand, and the automatic manufacturing of made-to-measure garments based on the customer's personal preferences and needs on the other hand shall be achieved.

In the following, we first give an overview of the components of the TüTex cloth simulation engine [1, 9, 124], which is based on the algorithms described in the previous chapters. Next, we briefly outline the integration of TüTex into the commercial modeling software Maya [3]. Finally, we describe the application of TüTex in the Virtual Try-On project [11, 12, 13, 18, 19], and present some simulation results within this context.

4.2 The TüTex Cloth Simulation Engine

An overview of the TüTex cloth simulator is shown in figure 4.1. The main components consist in the underlying physical model, and algorithms for numerical time integration and collision detection and response. Additionally, interfaces for the input of avatars, cloth models, and material parameters are provided as well as for visualization and user interface components. All components of TüTex have been implemented in C++.

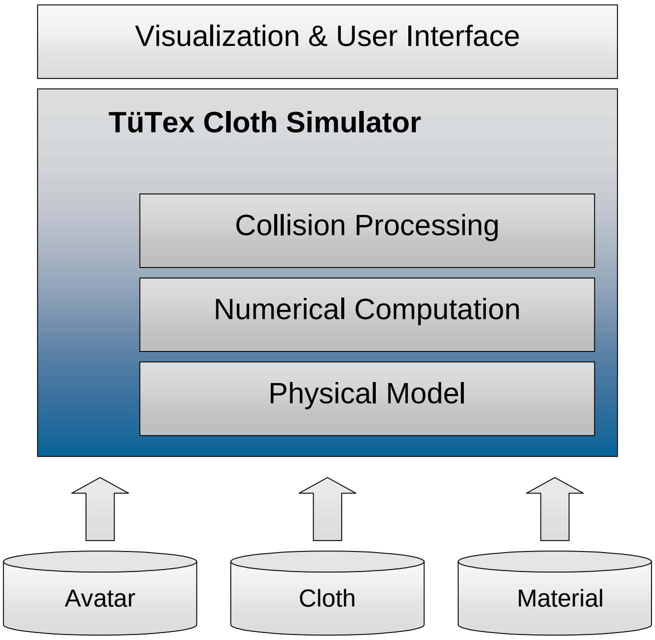


Figure 4.1: The components of the TüTex cloth simulation engine.

The **physical model** in TüTex consists of the linear finite element model described in section 2.3, together with the aerodynamic model presented in section 2.4.

Numerical computation is based on an implicit Euler time integration method (cf. section 2.3.4). In order to exploit the sparsity of the system matrices, the implementation of the ordinary differential equation solver, including the conjugate gradient method, is based on the Matrix Template Library (MTL) [112] which provides respective data structures and numerical linear algebra methods. The parallel implicit time integration scheme described in section 3.2 has been realized in *ParTüTex*, a parallel implementation of the TüTex cloth simulator. As described in the previous chapter, in this case MTL has been replaced by the Portable Extensible Toolkit for Scientific Computation (PETSc) [25, 24].

Collision detection is based on k-DOPs, and there are several **collision response** schemes available. As collision processing is not a topic of this work, we refer to [93, 9] for more details.

Avatars and other rigid collision objects can be imported into TüTex as keyframed mesh animation sequences. Linear or bicubic interpolation is used to generate frames depending on the cloth simulation's time step size.

Cloth models have to be provided as planar patterns, given as triangle meshes, together with copies of the patterns prepositioned around the avatar in 3D. The planar patterns provide the textile's rest state. Complex garments can be automatically assembled from the patterns as described in section 2.2.

Material parameters taken from Kawabata measurements are supported as described in section 2.3.5 and can be imported in form of XML files.

The output of the simulation engine consists in a sequence of triangulated meshes representing the animated textile. These meshes can be passed directly to a **visualization module** or saved to files and rendered off-line.

4.3 The tcCloth Maya Plugin

In order to provide a comfortable method to set up simulation scenes and to be able to directly visualize the cloth simulation results with a powerful rendering software, we integrated TüTex into Maya in form of the plugin *tcCloth* [3]. The plugin allows the user to interactively design planar cloth patterns together with seam lines, to position these patterns around a 3D figure using existing Maya tools, and to simulate the drape of the clothes on animated avatars.

The tcCloth plugin has been implemented partially in C++, and partially in form of Maya Embedded Language (MEL) scripts. Seams, patterns, and garments are realized as new nodes in Maya's scenegraph, the Maya Dependency Graph. Two-dimensional cloth patterns are defined by sketching the boundary, for instance with Maya NURBS curves. The resulting planar patterns are then triangulated, where the resolution can be adjusted as desired. Here, we use the Delaunay mesh generator which is freely available from [109]. Then, these patterns can be manually positioned around a 3D figure using standard Maya tools for moving and rotating objects. Additionally, Maya deformers can be applied, and the patterns can be scaled in order to model garments of different sizes. Finally, material parameters as well as simulation parameters like the time step size and the collision distance have to be set, and the cloth simulation can be started. The simulation results can be rendered directly in Maya. Because of the direct coupling with the simulation engine, it is also easily possible to apply several post-processing tools which are provided in Maya, like subdivision or mesh extrusion to improve the smoothness and the 3D impression of the clothes.

An example of the interactive generation of patterns and seams in tcCloth is shown in figure 4.2, TüTex cloth simulation results computed within the plugin are presented in figure 4.3.

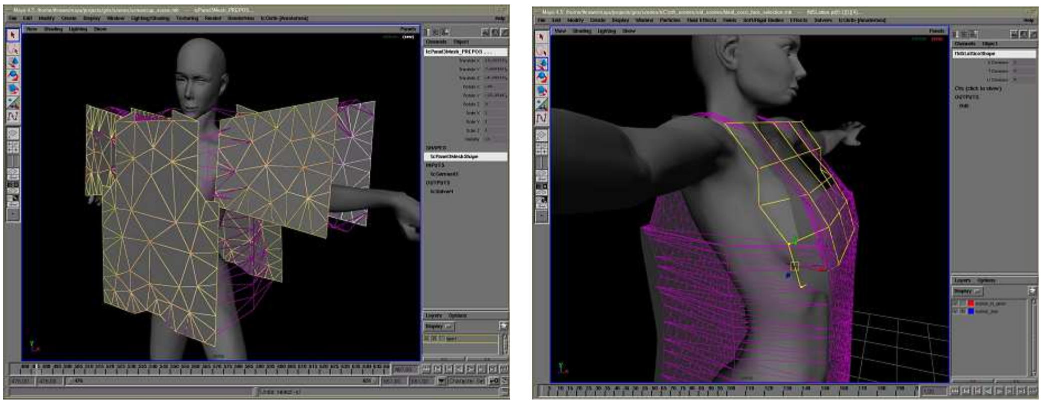


Figure 4.2: Interactive design and generation of patterns and seams in the Maya plugin tcCloth.

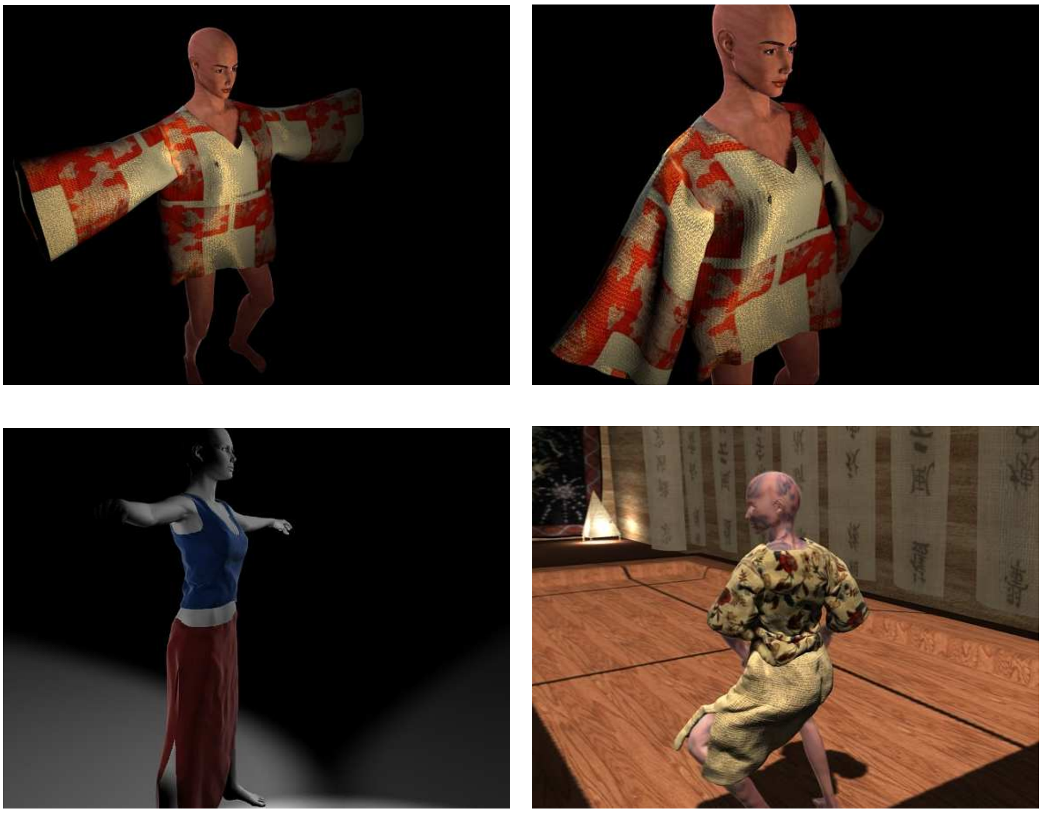


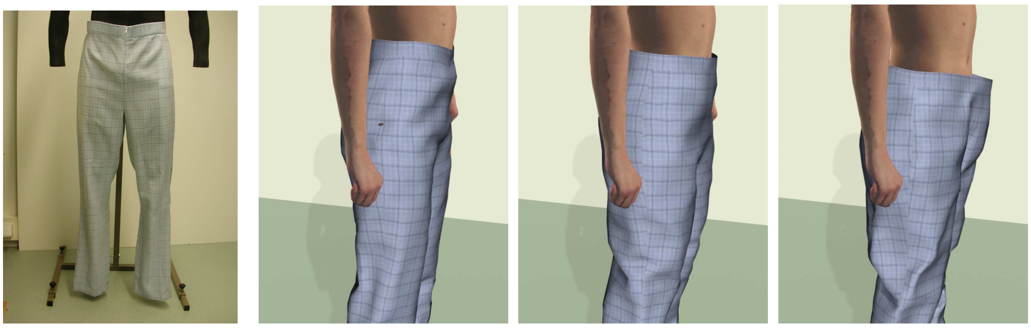
Figure 4.3: TüTex cloth simulation results computed within tcCloth.

4.4 Virtual Try-On

The goal of the research project Virtual Try-On[1] was the realization of a complete clothing pipeline for individual made-to-measure garments. In the Virtual Try-On scenario, for instance in an internet clothing store or a real shop, a customer shall be enabled to chose some garments according to his size and personal preferences, virtually try them on, and order them after he has been convinced of the fit and the look of the clothes. Only then, the garments are actually manufactured, thus reducing costs for manufacturers as well as retailers, and accounting for individual wishes of the customer.

---

[1] The Virtual Try-On project was funded by the Bundesministerium für Bildung und Forschung (bmb+f), cf. [126].



(size 36) (size 40) (size 46)

Figure 4.4: A woman who has size 36 is wearing different trousers. On the left, a photo of the real trousers is shown, the next three images show simulated trousers of size 36, 40 and 46, respectively.

In this context, the Virtual Try-On project aimed at

- using real world input data for virtual human models and garments,
- achieving a realistic simulation and visualization of the drape and fit of virtual clothes,
- providing an automatic clothing pipeline from the construction over the simulation to the visualization of virtual clothes on virtual human models.

Based on an automatic prepositioning of cloth patterns around the 3D figure[2] [66, 62], the TüTex cloth simulator, and a rendering engine[3] [106, 91, 96], a prototype of an automatic pipeline for virtual clothes could be realized, as detailed in [12, 13]. In this pipeline, the real world input data comprises three-dimensional avatars of the person that shall be clothed in form of textured body scans[4], garments which are constructed from cloth patterns taken from textile manufacturers' CAD databases[5], and measured material properties. The latter consist of material parameters taken from Kawabata mea-

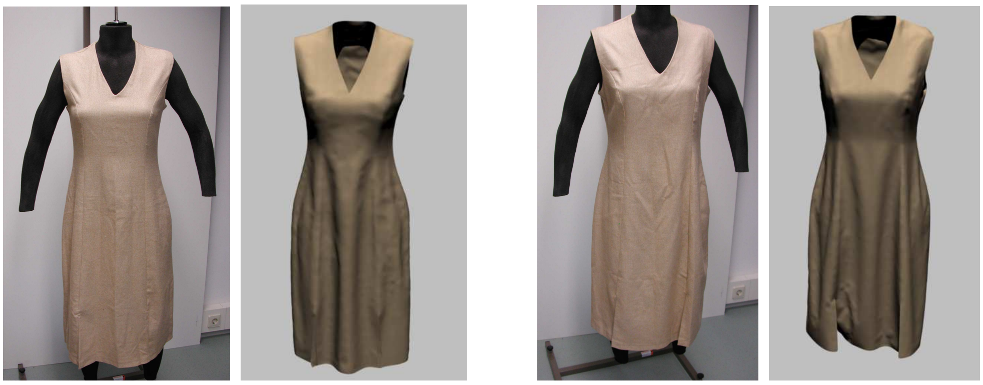
---

[2] The prepositioning algorithms have been developed by the *Fraunhofer IGD Darmstadt*.

[3] The rendering engine has been developed by the *University of Bonn*.

[4] The body scans were provided by the *tecmath AG*.

[5] The garment patterns were provided by the *Hohensteiner Institute*.



(size 38) (size 42)

Figure 4.5: Comparison of real clothes and simulation results: On the left, a real and a virtual dress of size 38 are shown. The right images show a real and simulated dress of size 42.

surements[6] for the physical cloth simulation, as well as reflection properties of textiles for varying viewing angles and lighting conditions[7] [96].

The simulation of the drape and fit of virtual clothes lies at the core of the Virtual Try-On scenario. Here, the TüTex cloth simulation engine has been employed not only to compute the sewing and the draping behavior of the textiles, but also to obtain a meaningful prediction of the fit of the garments on the respective avatar, as shown in the examples in this section (note that all images in these examples have been rendered in Maya). The first example shows a woman who has (european) size 36 wearing trousers of different sizes. As shown in figure 4.4, the simulation is able to predict the fit of the trousers on the woman's body. Figure 4.5 shows a comparison of real and virtual dresses. Again, the TüTex cloth simulation engine computes the different drape of the garments, and the different sizes can be easily distinguished

4.5 Summary

In this chapter, we gave an overview of the TüTex cloth simulation engine and discussed its application in the Maya plugin tcCloth, and in the national research project Virtual Try-On. Currently, the Maya plugin tcCloth is prepared to be made freely available for the use in research and education. The

---

[6] The Kawabata measurements were carried out by the *Hohensteiner Institute.*

[7] The reflection measurements were carried out by the *University of Bonn.*

results of the Virtual Try-On project were recently made known to a wide public at the CeBIT 2004 [19].

The described Virtual Try-On scenario does not yet allow real-time user interaction with the simulated clothes. In the next chapter, we will extend the so far described methods and develop interaction concepts for the assembly, simulation and design of clothing in immersive 3D virtual environments.

Chapter 5

Interactive Cloth Assembly, Simulation and Design in Virtual Environments

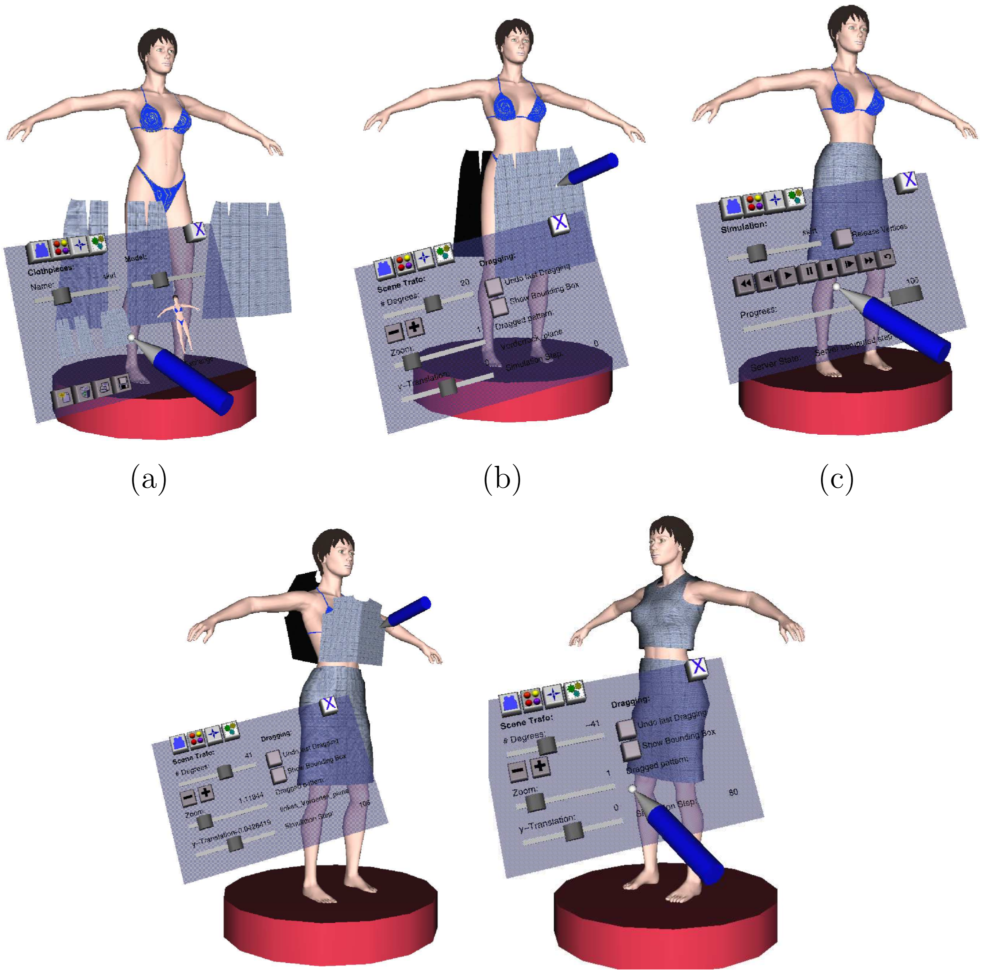
Virtual environments have become very powerful in the last years such that real world problems can be tackled and solved with these systems. Virtual prototyping is a major area of research, and the integration of physical simulations into virtual reality systems has emerged as a challenging problem. Especially in the case of deformable objects, there are high demands not only to the realism and speed of the physical simulation, but also to intuitive human-computer interaction techniques that allow a direct manipulation of the simulation results in an immersive virtual environment. Taking the virtual try-on scenario described in the last chapter a step further, the design of clothing in 3D including an automatic generation and modification of the corresponding planar cloth patterns is a major goal in this area.

In this chapter, we propose techniques for the interactive assembly, simulation, and design of clothing in virtual environments. The three main contributions of this chapter are the following: First, we develop interaction and navigation methods specifically designed for assembling virtual garments from planar patterns, and for grabbing and moving parts of the virtual clothes during the physical simulation. These techniques are realized in the virtual reality application *Virtual Dressmaker*, which supports input devices with six degrees of freedom and stereo visualization (section 5.2). Next, we present the results of two usability studies that were carried out in order to compare the developed techniques with traditional interaction in standard desktop modeling software (section 5.3). Finally, we propose virtual tailor tools like virtual scissors, a virtual sewing needle, and virtual pins for the interactive modification and design of virtual clothes in 3D. The modifications are automatically transferred to the planar patterns, directly integrated into the physical simulation, and immediately visualized in the Virtual Dressmaker (section 5.4).

5.1 Introduction and Related Work

Typically, current desktop 3D modeling software has the disadvantage of restricting the user to 2D input devices and, mostly, monoscopic displays. While the creation of 2D cloth patterns can be done comfortably with standard 2D graphical editors and 2D input devices like a mouse or a 2D pen, interaction in a three-dimensional virtual scene like the pre-simulation positioning of the patterns as shown in figure 5.1, pulling the cloth into shape during the simulation, or modifying the design with virtual tailor tools can be substantially improved by interaction devices supporting six degrees of freedom. In this context the users' acceptance to adopt to such virtual environments has to be carefully investigated. The study of human performance in virtual environments is important in order to evaluate the efficiency of the users' interaction, to evaluate their needs, and to enable them to naturally interact with the virtual world.

In the past, research in computer aided garment design has focused on 2D desktop applications. In [73], geometric methods to model static 3D garments based on planar patterns are proposed. A 2D application for the design, prepositioning, and sewing of cloth patterns, combined with a physically based cloth simulation, is presented in [128], and another 2D application for the design of garments in 3D, making use of the modeling tool Maya, is presented in [45]. Moreover, as mentioned above, there are several commercial 3D modeling tools that include cloth simulation modules such as Maya, Cinema4D or Poser. None of these, however, allows the use of 6DOF interaction devices. In [128, p.248f], the authors state that they experimented with 6DOF tracking devices and the use of stereo displays. They describe problems with inaccurate 3D perception due to the lack of head tracking and the inability to precisely position cloth patterns due to the inaccuracy of the spatial tracking technology. They summarize that in the context of a "fully interactive 3D simulation system, current visualization and interaction tools remain unappealing to people who, despite living in a 3D world, are still used to simple 2D representations of the traditional graphic schemes and design tools". In the following, we try to show that it is possible to overcome these



(a) (b) (c)

(d) (e)

Figure 5.1: The Virtual Dressmaker: Image (a) shows the parts of a skirt, inserted into the scene. In the next step, the patterns have to be placed around the body (b). Picture (c) shows the simulated skirt. In (d) and (e) these steps are repeated for a top.

problems by the use of specialized interaction techniques and virtual reality technology.

It has been shown in various contexts that virtual reality applications are meanwhile well beyond the experimental stage, and virtual prototyping using 6DOF input devices and stereoscopic visualization has shown to be an important alternative to 2D desktop solutions, e.g. in virtual assembly simulation [136], in various medical applications [104, 74], tele-robotics [72], and scientific simulations and visualization [79, 121]. This progress is mainly based on two development directions, namely the research on rendering techniques in classical computer graphics and on user interfaces and interaction techniques in virtual environments. Important contributions in the development of interaction techniques were made by Buxton and Myers [37] and Guiard [67]. Buxton and Myers reported on significant performance increase when bimanual navigation/selection is applied compared to accomplishing the task unimanually. Guiard's work showed how a tool in the non-dominant hand is used to define a (coarse oriented) coordinate system, a kind of reference frame, while the dominant hand is used for fine positioning relative to that coordinate system. Exploiting the above observations on the asymmetric use and coordination of human hands, several research groups developed two-handed interaction techniques for different virtual environments. The resulting tools are known under various names like: *pen and palette* [105], *pen and tablet* [21], *physical clipboard* [119], *3D-Palette* [31], the *personal interaction panel (pip)* [122], and *virtual palette and remote control panel* [42]. With their simplicity and intuitive utilization, these concepts provide an excellent basis for the development of virtual reality applications.

As described in section 2.2, the cloth assembly process can be automated to a certain level (cf. [66, 62]). However, these methods require predefined labels on the patterns indicating their relative positions to the body, an information that is not always available. Furthermore, in the garment design process, the tailor himself might want to experiment with various relative positions of the patterns, making automatic placement impossible at that stage. Thus, interactive manual cloth assembly is an important tool for virtual try-on and design of clothes.

The cutting of 3D deformable models is an important topic in the research area of virtual surgery [61, 49, 90, 30]. Commercially available surgical simulators like KISMET, LapSim, xitact, SimSurgery, Reachin, and Simbionix provide tools for cutting and sewing of human tissue. However, in this field of application, the approaches concentrate on simple algorithms like canceling the underlying mesh elements, in general tetrahedra, or stitching parts of a tissue together by a filament. Hence there is no volume conservation during cutting, and sewing is done by a separate simulation of the threads, e.g. [100]. Moreover the three-dimensional case usually deals with only one mesh without seams, contrary to the situation with garments. Examples of cutting and sewing pieces of cloth in a physical simulation are given in [127], however without details about the algorithms and interaction techniques involved. In [92] an algorithm for animating "cloth-like"objects in real-time is presented, which is based on a simplified implicit integration scheme. The approach is demonstrated through examples in a virtual reality environment similar to our setup. In this work, however, the virtual environment is used mainly as a presentation medium and no 3D/6DOF interaction tools are described.

5.2 The Virtual Dressmaker

The Virtual Dressmaker is an application for the interactive assembly and physically based interactive simulation of cloth. In particular, the system allows to interactively grab, move, cut, and sew parts of the clothes during the simulation. The proposed application consists of a VR-interface client that provides a large stereo projection display combined with 6DOF interaction and a server for physically based cloth simulation.

In this section, we follow parts of [7, 17].

5.2.1 Application Overview

The proposed application allows the user to choose some garment and precisely position the patterns around the figure using a set of tools based on 6DOF grabbing and manipulation. After choosing a particular material, the patterns can be sewed together, and the integrated cloth simulation engine computes the movement and drape of the clothes. The intermediate steps of the simulation are displayed during the computation, while the user can still navigate in the scene and select and drag parts of the clothes. At any time the user can jump back to the pre-simulation state, make corrections or choose a new or additional garment, set another garment material etc. This scenario allows the user to try-on clothes with different sizes and find the appropriate ones. Furthermore, different behavior of the garments can be examined, depending on the material properties of the simulated cloth patterns.

As detailed in section 5.4, the Virtual Dressmaker includes interactive tailor tools like virtual scissors and a virtual sewing needle to allow the interactive design of garments directly in 3D. The design tools are integrated into the physical simulation and the user gets an immediate visual feedback of the design modifications.

The input of our application consists of a 3D character model and garment patterns with seam information. The 3D character model can be an artificial figure designed with a 3D modeling tool like Curious Labs' Poser or a 3D scan of a real person. The clothes have to be provided as 2D patterns together with seam information, as described in section 2.2.

5.2.2 System Setup

Before discussing the interaction concepts, we first give an overview of the main components of our application. These components can be divided into three main parts: the virtual reality setup, the cloth simulation engine, and the proposed application itself.

5.2.2.1 Virtual Environment

The hardware on which the proposed system is based, consists of an electromagnetic 6DOF tracker (*Ascension, Flock of Birds*) and a stereo table top display (*Barco, Baron*) called the *Virtual Table*. The Virtual Table is a large screen tilted about its horizontal axis, on which the images are back-projected in stereo (alternating projection of pictures generated for the left and right eye combined with an LCD shutter).

The tracker is used for determining the position and the orientation of three receivers. The first is used to track the position and orientation of the viewer's head (see figure 5.2). The virtual camera is attached to this receiver. The second is attached to a physical pen with two buttons, which the user holds in the dominant hand. The virtual counterpart of the pen is used to manipulate the 3D virtual buttons, sliders, and other interaction elements projected on the personal interaction panel (pip), as well as the objects in the scene. The third receiver is attached to the panel which is realized as a transparent palette and on which the interaction elements are back-projected [107]. Due to the transparency, the user can manipulate the tools on the pip as if they were on its surface. This enables the application of two-handed interaction according to the *pen and palette* paradigm as described in [122]. In this way, intuitive and flexible interaction is realized that shares some features with common desktop interaction techniques.

The basic interaction elements we work with are part of the Studierstube framework [108]. This is an object-oriented library extending the standard Open Inventor [99, 134] functionality, allowing for transparent processing of tracker events delivered by the tracker system. These events are propagated through the Open Inventor scenegraph and can be used to define virtual 3D

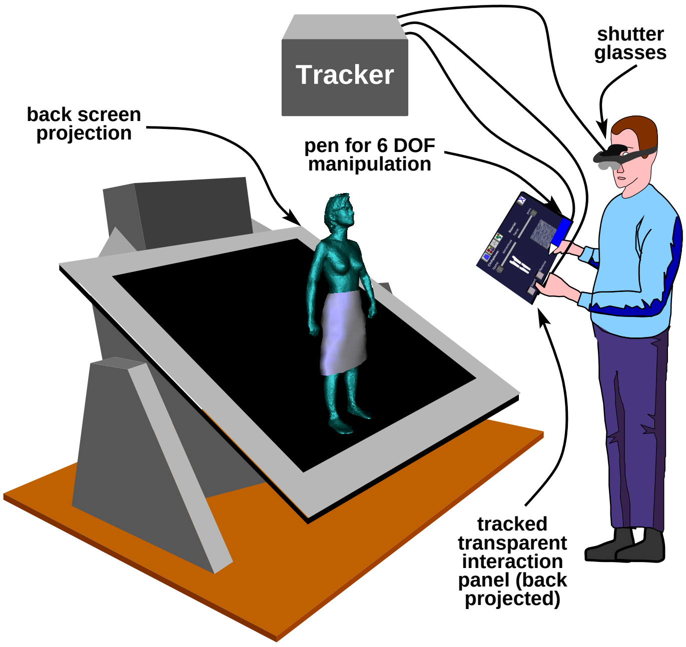

Tracker
shutter glasses
back screen projection
pen for 6 DOF manipulation
tracked transparent interaction panel (back projected)

Figure 5.2: Hardware setup of the virtual environment. The tracker determines the position and orientation of three sensors, attached to the pen, the palette, and the stereo glasses, respectively. The images on the palette are back-projected via the table display.

interface elements like buttons, sliders etc. Thus, 3D user interfaces can be build that are akin to common 2D desktop interaction.

5.2.2.2 Cloth Simulation Engine

The interaction techniques within the Virtual Dressmaker application are independent of the particular choice of cloth simulation engine, and both the finite element method described in section 2.3 and a particle system based on the work presented by Etzmuß [56] have been integrated into the application. Both models allow to achieve interactive frame rates depending on the geometric complexity of the clothes. Due to the modular client-server application architecture described in the next section, a flexible interface to integrate any other cloth simulator that allows fast and stable simulations based on two-dimensional cloth patterns is provided.

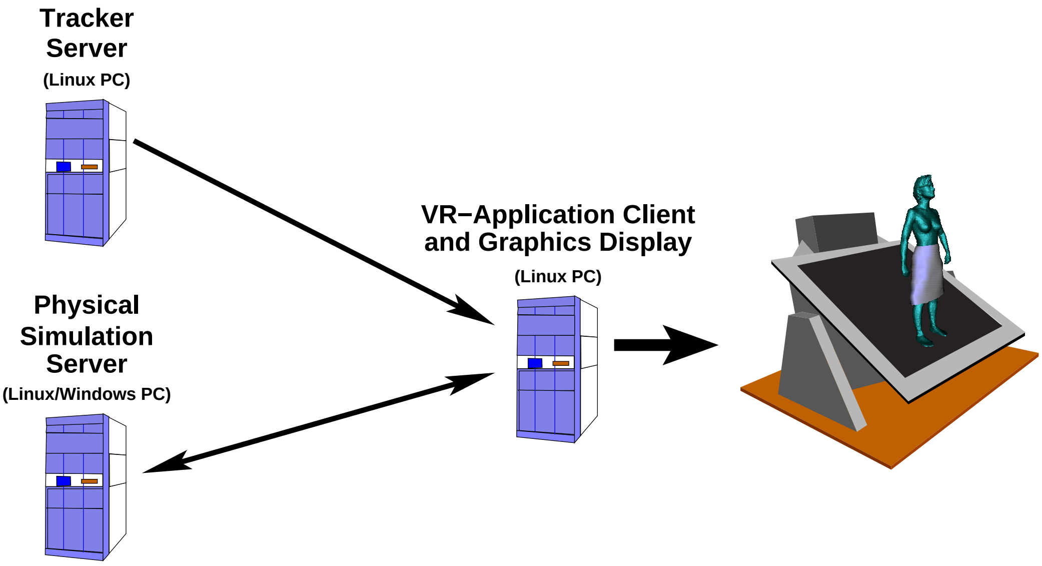

Tracker Server
(Linux PC)
Physical Simulation Server
(Linux/Windows PC)
VR−Application Client and Graphics Display
(Linux PC)

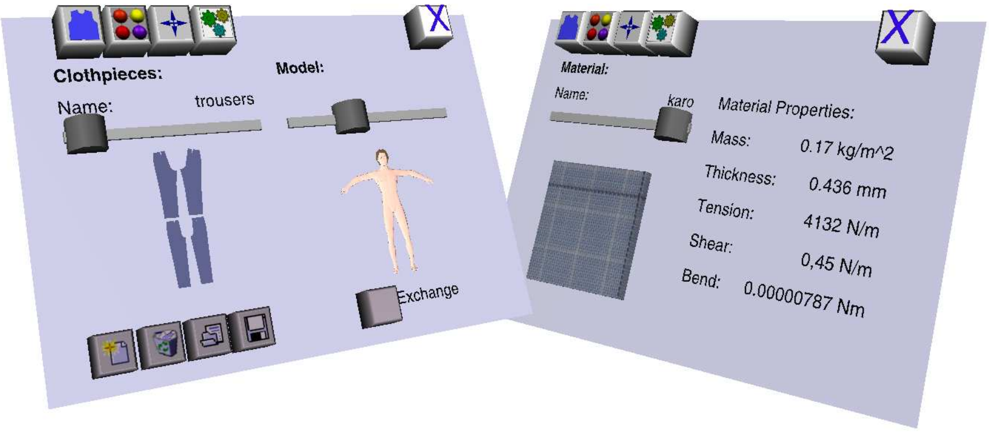
Figure 5.3: The Virtual Dressmaker's client-server configuration. The physical simulation is coupled with the VR application by a socket interface.

5.2.2.3 Application Architecture

Based on the hardware components, the Studierstube framework, and the TüTex simulator, we have built a socket-based heterogeneous client-server system that consists of two servers and a main client application. The first dedicated server is responsible for the 6DOF tracking and sends the events generated by the tracker system to the main application (see figure 5.3). The second server is used for computing the physically based simulation. Each time a simulation is started, the server sends the computed intermediate results to the application, such that a smooth relaxation of the simulated cloth can be observed on the screen.

5.2.3 User Interface and Navigation in the Scene

According to the *pen and palette* paradigm [122], we display various virtual tools on the surface of the palette. Depending on the functions of the interface elements, they are grouped into categories. Each of these categories defines a *sheet*. In other words, a sheet is a set of interface elements with similar logical functionality.



Clothpieces:

Name: trousers

Model:

Material:

Name: karo

Material Properties:

Mass: 0.17 kg/m^2

Thickness: 0.436 mm

Tension: 4132 N/m

Shear: 0,45 N/m

Bend: 0.00000787 Nm

Figure 5.4: The sheets for selecting garments, figure, and material properties.

First, there are sheets for selecting garments, body and cloth material properties (figure 5.4). Once the desired clothes are selected, the patterns may be inserted into the scene. Additionally, a set of material properties can be assigned to the currently selected cloth. The interactively positioned patterns can be saved and loaded allowing to extend the design process over several interactive sessions with the application.

Once the cloth pieces are inserted into the scene, they may be freely manipulated applying the interaction pen for grabbing and moving them. However, it is often impossible to correctly position a given cloth piece without navigating in the scene. Since the user cannot move freely around the virtual 3D-body, we had to implement a strategy for navigation in the scene space. In a first step, we realized this with a set of simple sliders, which most of the users know from traditional desktop applications as depicted in figure 5.5. Initially, the 3D figure is placed on a turntable that defines the axis of rotation of the objects in the scene. The turntable also indicates the coordinate system for the subsequent physical simulation.

We provided one slider for each of the scene transformations: rotation about the vertical axis, scale of the scene, and translation along the vertical axis. With these interaction elements the user is able to manipulate the entire scene, while the possible movements are constrained to simulate the

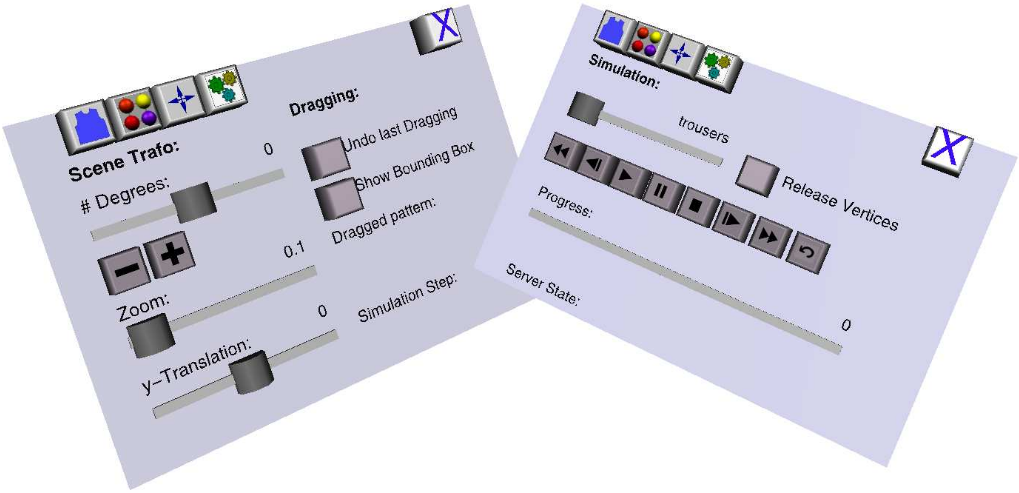


Figure 5.5: The (left) navigation and (right) simulation sheets.

motion of a real tailor standing in front of a real human.

As a result of the usability studies described in section 5.3, we examined alternative navigation concepts. They will be discussed in detail in section 5.3.5.2.

Similar to traditional desktop applications, the navigation sheet contains an “undo” button, as well as fields for displaying the current progress of the simulation running on the simulation server. In addition, the name of the currently picked cloth piece is displayed on this sheet as a text string in order to help the user to quickly identify the picked pattern and place it correctly.

The simulation sheet, as seen in figure 5.5 represents the communication interface between the VR application and the computation server. We use simple VCR-like buttons for displaying Play, Stop, Rewind etc. functionality to control the simulation. The Play button, for example, starts the simulation, and if the result of the simulation is not as expected, the user may interrupt the simulation with the Stop button, undo the simulation with the Reset button and readjust the cloth patterns. Beside the interaction elements, this sheet includes several status fields similar to the navigation sheet. These fields display the current state of the simulation server and inform the user about the simulation progress.

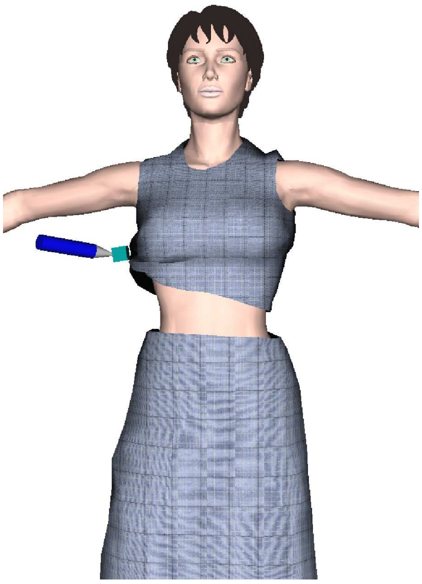


Figure 5.6: Selecting and dragging parts of the clothes during the simulation.

5.2.4 Interactive Cloth Assembly and Simulation

A typical working session within the Virtual Dressmaker starts by choosing a human model, which can be a body scan or some modeled 3D figure, in the corresponding menu on the palette. Then, a set of garment patterns can be chosen and positioned interactively around the avatar (see figure 5.1). Both the interaction with the menus on the palette and the manipulation of the patterns in the scene is done with the pen. Hence, in the terminology of [94], *local selection* is used throughout the application. After positioning the patterns, the sewing and draping process can be started.

During the physical cloth simulation, the user can still navigate freely in the scene and even select and drag parts of the simulated cloth. In our system, this can be accomplished by utilizing the pen to select parts of the clothes, or more precisely, vertices of the underlying mesh. The selected points are visualized by small cubes, which can be moved to the desired destination. The transformations of the selected vertices are sent to the server and integrated into the simulation as constraints (figure 5.6). With the "Release vertices"button on the simulation sheet (figure 5.5), the constraints are released, and the cloth relaxes due to internal forces and gravity. This technique allows pulling the simulated garments into shape, just like a real person does after putting on real clothes, and it is a basic tool for virtual garment design enabling the tailor to experiment with different cloth drapes.

5.3 Usability Evaluations

In order to quantitatively verify the proposed interaction and navigation methods, we carried out two usability studies, investigating the efficiency of user interaction in the Virtual Dressmaker application compared to standard desktop modeling software. In section 5.3.1 we report on a pilot study that was run to establish a proof of concept for our application. In the rest of this section 5.3, we will describe the results of a subsequent extended usability evaluation. In particular, we investigate the interaction tools for positioning garment patterns and compare the results to interaction with common desktop applications. We measure completion time and precision of the positioned patterns and evaluate the subjective impressions of the users.

The results show that the presented interaction tools provide a valuable method for cloth assembly and design. Moreover, certain features of current 3D desktop design tools can be exploited for further improvements of VR interaction.

In this section, we follow parts of [8] and [14] (cf. also [7, 17]).

5.3.1 Pilot Study

In a pilot study we investigated the Virtual Dressmaker application in comparison to the 3D modeling application CosmoWorlds (by SGI). The goal of this study was to establish a proof of concept of our prototype for cloth design, to evaluate the navigation tools, and to learn more about the user's interaction needs in virtual environments. The goal in the tasks was to position garment patterns into transparent bounding boxes. Only the completion time was measured, and precision was not evaluated explicitly. After completing the tasks, the participants rated their subjective impressions in both applications.

The experiment consisted of two dress assembly tasks that had to be accomplished first using the Virtual Dressmaker application, and second with the help of the desktop modeling tool CosmoWorlds, which supports standard Inventor-based interaction in a 3D scene (cf. section 5.3.2.1). Another reason for choosing CosmoWorlds was the relatively easy to use and intuitive interaction concept. In addition, with CosmoWorlds, we could guarantee that most of the users had no experience with both the Virtual Dressmaker and CosmoWorlds, which allowed for a fair comparison of the interaction concepts. In both applications, we compared the users' performance in terms of time needed to complete a given task and protocoled and evaluated their subjective impressions.

The users' task was to go through the complete process of garment assembly twice, once for a skirt and once for a short dress. We used simple models in order to keep the trial time reasonable. The skirt consisted of three, the dress of four cloth patterns. In the initial situation the patterns were placed in front of the 3D body. To create equal test conditions for users with and without knowledge of garment creation and design, the goal positions for the cloth pieces were defined by colored transparent bounding boxes around the 3D figure.

The participants were asked to place the patterns into the corresponding bounding boxes within the Virtual Dressmaker application using the 6DOF interaction devices pip and pen, as well as stereo glasses with head tracking, as described in section 5.2.2. Then, we asked the participants to do the same within the desktop application CosmoWorlds, using a 2D mouse and a monitor display, starting with exactly the same initial conditions. In order to compare the users' performance, the time used for the four tasks was recorded. Afterwards, the participants had to fill out a short questionnaire about their prior experience with 2D or 6DOF interaction tools and their subjective impressions of the experiment. In particular, they were asked to rate isolated interaction stages such as picking, positioning, scene navigation, and overall interaction. In this way, we aimed to separate the application features for which 6DOF interaction is suitable from those which require further improvement.

5.3.1.1 Results of the Pilot Study

We recruited 19 test users aged between 25 and 30 years (14 male and 5 female) to perform the experiment. All of them were computer science PhD students, graduate students, or students from other departments of our university. Some of them had experiences with 2D interaction modeling software, only a few with 6DOF interaction.

In order not to measure the learning effect of the participants between the first and the second trial, we first demonstrated both applications. Afterwards, we let the participants try the interaction tools on their own without recording the interaction time. After these two steps, we started to measure the time spent on the interactive positioning with both applications.

The results of the time measurements are shown in figure 5.8. Each group of two neighboring columns represents the time spent by one test

6DOF compared to 2D	better	equal	worse
Positioning	95%	5%	0%
Picking	42%	17%	42%
Navigation in the Scene	48%	26%	26%
Overall Interaction	95%	5%	0%

Figure 5.7: Summary of the subjective impressions of the test users in the pilot study.

user. In each diagram, the left column displays the time spent in the Virtual Dressmaker, the right column displays the time spent in CosmoWorlds for completing the given task. The last two columns on the right show the mean values. The users are grouped according to their prior experiences with 3D software: the people in group A had worked with 6DOF interaction tools before, users in group B only with 3D desktop interaction tools, and those in group C had no experience with 3D software at all.

As shown, all users accomplished the tasks faster in the Virtual Dressmaker than in CosmoWorlds. This observation holds for all participants in all groups, independent of their 3D/6DOF interaction experience. The overall mean time used in the Virtual Dressmaker was 92 seconds, compared to 245 seconds in CosmoWorlds. Note that even the users with only desktop 3D interaction experiences were faster with our application compared to performing the same task in CosmoWorlds.

Evaluating the questionnaire, we ascertained that 95% of the users rated translating and rotating objects in the scene as "better"in our application than in CosmoWorlds, while the results with respect to picking varied more: 42% rated picking as "better", 42% as "worse"and 17% couldn't decide. Similarly, scene navigation was rated "better"by 48%, "worse"by 26% and 26% were uncertain. However, overall interaction was rated "better"by 95% of the users and 79% could imagine working regularly with 6DOF devices instead of common 3D desktop tools in this context. The answers to the questionnaire are summarized in figure 5.7.

5.3.1.2 Consequences of the Pilot Study

As a consequence of the reported subjective impressions of the test users, we improved the tools for picking objects and scene navigation. First, we added visual feedback when the pen penetrates the picking range of an object by highlighting the respective object in a different color. Moreover we designed

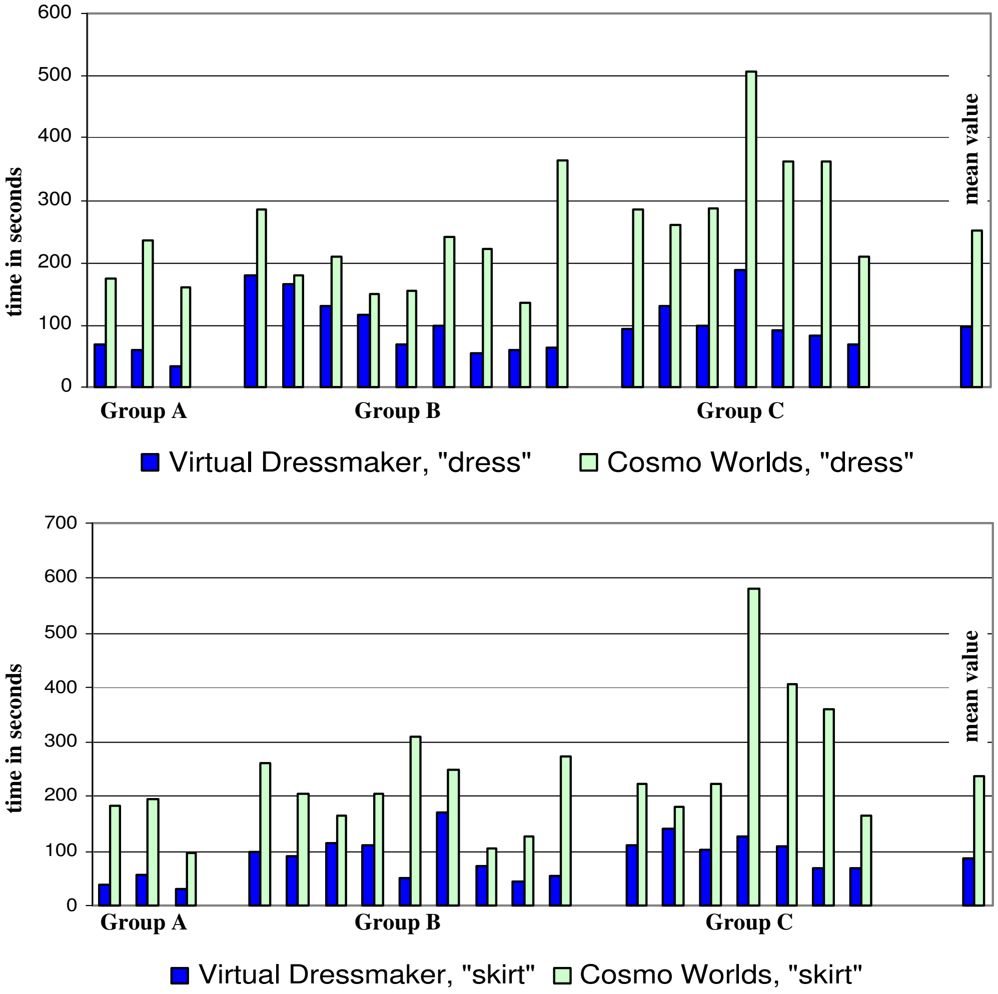


Figure 5.8: Results of the pilot study in the two test scenarios. The performances were grouped into three categories: experts in VR software (group A), experts in 3D desktop tools (group B), and beginners (group C).

the *Navigation Sphere*, a new navigation concept which can be used as an alternative to the previously described navigation sheet. The Navigation Sphere provides exactly the same degrees of freedom as the three sliders on the navigation sheet and is always shown in the lower right corner of the virtual scene (see figure 5.9 (c)). Its position is fixed in space with respect to the user. The user has to pick the sphere with the second button on the pen and can then navigate in the scene. Left-right movement with the pen is related to rotation around the vertical axis, up-down movement to vertical translation, and in-out movement to zooming in and out of the scene. Rotations of the pen are not taken into account. With this tool the user does not have to switch to the palette while working with the pen in the scene.

We verified these improvements in a subsequent usability study which we will describe in the remainder of this section 5.3. In order to obtain meaningful results, we extended the number of participants, added the widely used standard modeling software Maya (by Alias) to our test applications, and measured both time and precision in the given experiments.

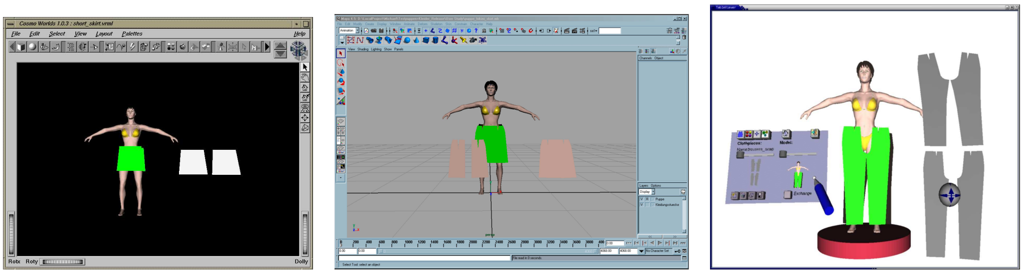
5.3.2 Experiment Setup

In order to evaluate the interaction within our virtual reality application and compare it with standard techniques on desktop computers, we customized three test scenarios with three different hardware and software configurations. The usability evaluation consisted in practical tasks of positioning garment patterns around an avatar in the Virtual Dressmaker and in the desktop applications CosmoWorlds and Maya, accompanied by a questionnaire.

5.3.2.1 Test Scenarios and Questionnaire

In the following, we briefly describe the employed desktop modeling applications. A summary of the differences between all three applications compared in the user study is displayed in figure 5.10.

The Maya Desktop Application. The first application is the commercial modeling tool Maya by Alias with 2D input devices (mouse and keyboard) and 2D output (21” monitor). At the beginning, the Maya scene is displayed in a perspective view (figure 5.9(b)). The user can switch (pressing the space key) to a split or single mode, where the front, top, and side views besides the perspective view can be selected. To navigate in the scene, Maya provides useful shortcuts: Pressing down the “Alt”-key the user can rotate the scene with the mouse holding the left mouse button. Holding the middle mouse button is related to translation, and holding the right mouse button, the scene can be scaled. For manipulating the garment patterns, the participants were allowed either to use the navigation buttons in the toolbar or the keyboard shortcuts (q: selecting, w: translation, e: rotation) in combination with the mouse. When objects are selected, they are highlighted and displayed as wire frames. If the user chooses a manipulation tool, a local coordinate system is shown for the object. By holding the left mouse button,



(a) (b) (c)

Figure 5.9: The three test scenes in the user study. Image (a) shows the simple skirt in CosmoWorlds, image (b) a skirt consisting of three patterns in Maya. In (c), four patterns of a pair of trousers are displayed in the Virtual Dressmaker application. Between the lower patterns of the pair of trousers the Navigation Sphere is displayed.

the user then can manipulate the pattern. In addition, by clicking on one coordinate axis (translation), or one coordinate circle (rotation), the other degrees of freedom are fixed which allows very precise positioning.

The CosmoWorlds Desktop Application. The desktop modeling tool CosmoWorlds by SGI (figure 5.9 (a)) provides standard Inventor-based interaction in 3D scenes. The use of the mouse to navigate in the scene and to manipulate the patterns is similar as in Maya. However, in CosmoWorlds we did not allow the use of keyboard shortcuts. Hence, the participants had to switch between scene and pattern navigation by using the buttons provided in the toolbar. In CosmoWorlds, the degrees of freedom for the manipulation of objects are indicated by a bounding box. By clicking on one side of the bounding box the movement of the objects is constrained to this

	interaction hardware	display	predefined views	two-handed interaction	keyboard shortcuts	selection of multiple patterns
Virtual Dressmaker	6DOF pen	stereo	yes	yes	no	no
Maya	2D mouse	mono	yes	yes	yes	yes
CosmoWorlds	2D mouse	mono	no	no	no	no

Figure 5.10: Summary of the differences between the test applications.

plane. By clicking on a handle in the middle of each side of the bounding box, the rotation is constraint to one axis.

For all three applications we modeled the same test scenes consisting of three user tasks. The first showed a woman in the middle of the scenery, where two white garment patterns (black on the back side) of a very simple skirt were placed in front of the avatar. Moreover, there were two transparent green patterns positioned around the waist of the woman indicating the respective goal positions (figure 5.9 (a)). This scenario was used to familiarize the participants with the application and the handling of the respective system. The test users were allowed to work with this example for a while, and no time or precision measurements were taken. The second test scenario (figure 5.9 (b)) consisted of the same woman but with a more complicated skirt of three patterns. Again, the white patterns were placed in front of the woman whereas their green counterparts around the waist indicated the goal position. The last task (figure 5.9 (c)) was to place the four patterns of a pair of trousers around the female avatar.

The questionnaire complemented the practical part of the user study and was divided into three parts. For the answers of each question we used a balanced rating. The first part investigated the participants' prior experience with computers, virtual environments, and the software used in the study. On a scale from +3 (very good) to -3 (no experience) the test users had to assess their experience with computers, computer games, VR systems, Maya, CosmoWorlds, the Virtual Dressmaker application, and 3D software in general.

The second part of the questionnaire was directed at the participants' subjective impressions and feelings during the tests. For each application, again on a scale from +3 to -3, the test users had to rate the navigation and orientation in the scene, the grabbing and navigation of garment patterns, and the 3D impression.

Finally, the questionnaire's last part treated questions concerning the navigation elements and the subjective impressions in the Virtual Dressmaker application. The users had to compare the quality (from -3 to +3) of the sliders with the Navigation Sphere and rate the 6DOF navigation in the scene. The last two questions intended to determine the users impressions of the efficiency of their work in the virtual environment, asking for the personal impression of time spent during the interaction and the precision solving the given tasks. At the end the users had the possibility to write down problems, suggestions, and ideas for improvements.

5.3.2.2 Participants

We recruited 29 volunteers (26 male, 3 female) aged between 23 and 35 (average 26) among the university's students of computer graphics. 27 of the participants finished all tasks. We supposed that they were all familiar with standard input devices (keyboard, mouse). This was confirmed by the participants own assessment (average 2.4). All users had normal vision or were corrected to normal vision. For all systems there were no problems using them wearing glasses or contact lenses.
We carefully differentiated in the questionnaire between different levels of interaction and VR experience. Thus, most participants had significantly good experience in computer games (av. 1.4) and 3D software (av. 0.6), but the practice in VR systems (av. -0.1) and especially the Virtual Table (av. -1.7) was low, as expected. Also the experience with CosmoWorlds was extremely low (av. -1.9) whereas the group was split concerning the experience with Maya since some students were recruited from a lecture using Maya for the practical exercises.

5.3.2.3 Task description

We installed the test environments in a dedicated room of our department. It consisted of two standard PC workstations with keyboard, mouse, and standard monitor, on which Maya and CosmoWorlds were running. The virtual environment described above for the Virtual Dressmaker application completed the installation for the user study. For each system, there was one personal instructor present to explain the system and to help with arising problems and questions. The order of applications was chosen randomly for each participant, such that learning and fatigue effects for the last applications could be neglected. Not exceeding three volunteers at a time (one for each application), the test users first had to complete the preliminary part of the questionnaire concerning their previous knowledge and experience. Then, the respective application and its handling was explained to them by the instructor. In the Virtual Dressmaker application the user was questioned about the stereo impression of the displayed scene. Only when the 3D impression was correct we proceeded with the instruction.

All the tasks consisted in placing garment patterns to the respective goal positions, which were provided in front of the avatar at the beginning. Of course, the avatar and the goal positions could neither be selected nor moved.

First, we presented the test scenario with two garment patterns of a simple skirt as described in section 5.3.2.1 in order to allow the participant to experience the application. Being familiar with this task the user was shown the second scenario of a tripartite skirt. He was asked to match the garment patterns as precisely and quickly as possible to the respective green equivalents around a female avatar. For this task we recorded the time for completing the task and stored the final position of the patterns for the subsequent precision analysis. The final task for each application consisted in positioning four garment patterns of a pair of trousers around the avatar. Again completion time and final position of the garment were taken for evaluation. After completing the tasks in each application the user was asked to fill in the questionnaire and rate his personal impressions, concerning on the one hand the navigation and orientation in the scene, on the other hand the navigation, grabbing, and positioning of the patterns, and the 3D impression. Then he or she continued with the next application.

At the end of the tour, each user completed the questionnaire comparing the navigation tools of the VR environment and judging his personal impression of efficiency placing the patterns with respect to time and precision.

5.3.2.4 Hypotheses

For the evaluation of our user study, we developed two hypotheses:

(H1) The 6DOF techniques in the Virtual Dressmaker application allow faster and more precise interaction compared to common 2D interfaces.

We expect significantly shorter completion times and lower error rates in positioning the garment patterns. This is expected to be the case even for participants who have prior experience with standard 2D input devices only.

(H2) User interface shortcuts and tools reducing the complexity of the switching between different navigation tools and/or views enormously effect the quality of the results in the given tasks.

We expect that the results with Maya will be superior to those with CosmoWorlds, given that the participants are allowed to use keyboard shortcuts only in Maya. Here, we want to learn more about the users' preferences and habits for navigation in 3D scenes and thus to be able to improve the usability of navigation tools in virtual environments.

5.3.3 Results

The measured data for the participants' performance consisted in the time for completing the positioning tasks and the distance between the interactively positioned patterns and the target patterns to evaluate precision. Each pattern was saved and evaluated separately. Moreover, we asked for the test users' subjective impressions concerning the respective application after each task.

5.3.3.1 Measurements of Time and Precision

The time for completing each task was measured for each participant in each scenario. Both tasks of positioning the skirt and the pair of trousers (figure 5.11) prove that the users accomplished the given tasks in about half the time in the Virtual Dressmaker compared to Maya, and in about a third the time compared to CosmoWorlds.

Simultaneously to the time measurements, we saved the positioned garment patterns of each task of each user for subsequent analysis of the positioning precision with respect to the indicated goal positions. From the final positions we calculated two precision measures. The first was the average distance for each cloth taking the average of all Euclidean distances between points of the positioned mesh and the respective points on the goal mesh. Moreover, we investigated the maximal distance of the cloth calculating the maximal Euclidean distance of corresponding points on the positioned mesh and the goal mesh. Figure 5.12 shows both precision measures, the average and the maximal distance, where we can observe that the users achieved better results in Maya and the Virtual Dressmaker than in CosmoWorlds. For the pair of trousers, the precision in the Virtual Dressmaker application is better than in Maya whereas for the skirt the precision in Maya is best.

5.3.3.2 Subjective Impressions

The evaluation of the questionnaire concerning the handling of the respective application gives the following result (figure 5.13): The simplicity and the precision of grabbing and the overall navigation is voted best in Maya and slightly worse in the Virtual Dressmaker application, while in CosmoWorlds it is voted significantly worse. Best votes are given for the Virtual Dressmaker application for positioning and rotating chosen objects.

At the end, the volunteers were asked to judge the interaction tools in

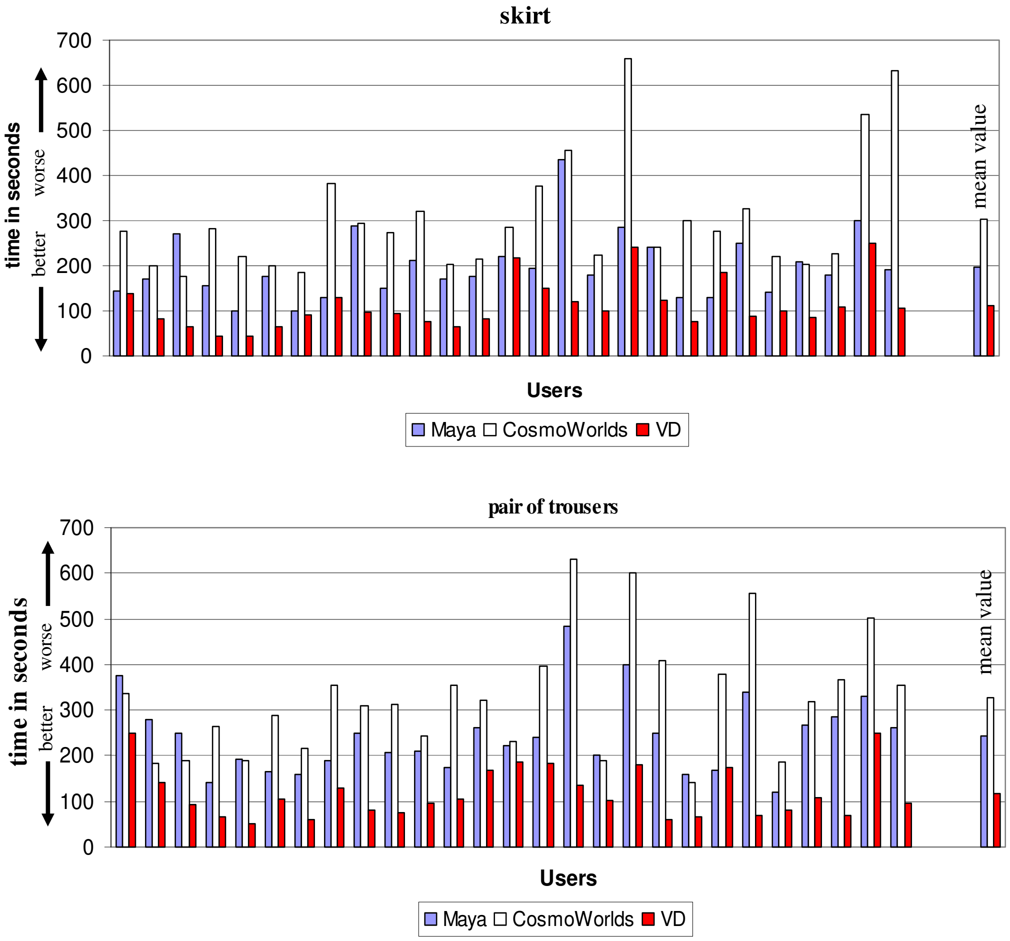


Figure 5.11: Results of the time measurements for positioning the skirt and the pair of trousers. Each group of three columns represents the recorded times for one user in the three test applications.

our VR application. Here, 60% of the users preferred the Navigation Sphere (cf. section 5.3.1.2), the remaining 40% rated the sliders on the navigation sheet (cf. section 5.2.3) better. All of the users appreciated 6DOF as input device for positioning (av. 2.1, var. 0.7).

The subjective impression of the users' efficiency resulted in the following: Both time and precision were rated to be better for the VR application compared to the desktop applications (time, av. 1.7, var. 1.4; precision av. 1.47, var. 1.15) and were in correspondence with the results of the measurements.

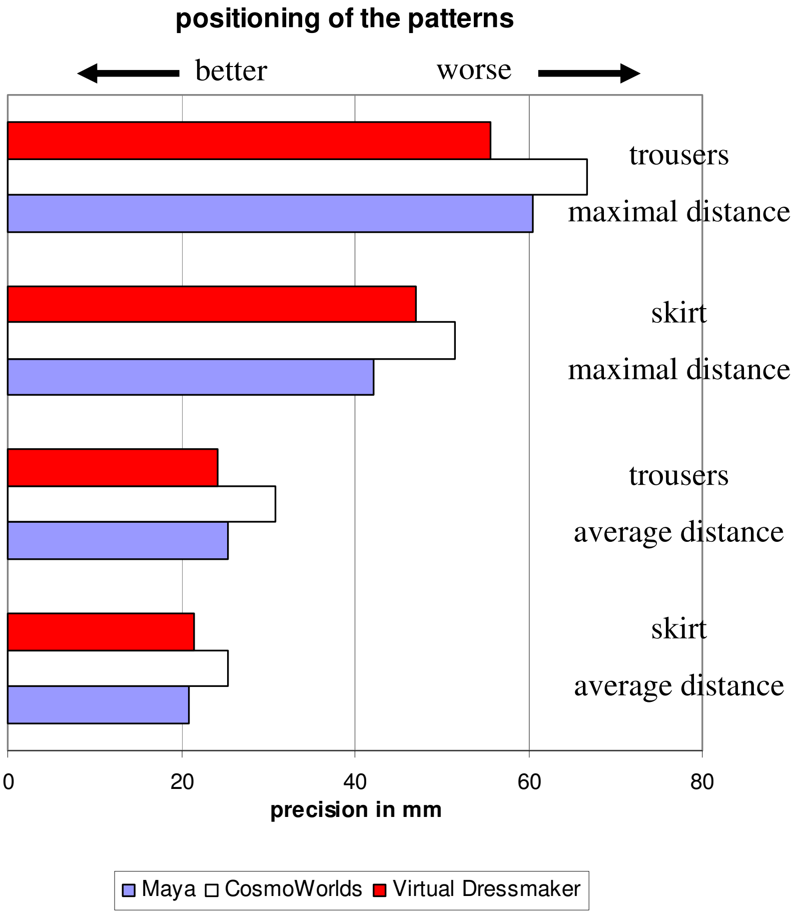


Figure 5.12: Results of the precision analysis of the positioned patterns.

5.3.4 Evaluation

Regarding our first hypothesis (H1) we can confirm the raise of efficiency by shorter positioning times. The precision did not vary as much as the computation time. Hence, it seems that the users tried to achieve the same precision in all tasks, and the completion times represent a good measure of the efficiency.

If we look at time and precision simultaneously for each user (figure 5.14) we can see that the values for the Virtual Dressmaker application are situated closer to the origin than for Maya and CosmoWorlds, which means that the interaction with this application is more efficient with respect concurrently to time and precision.

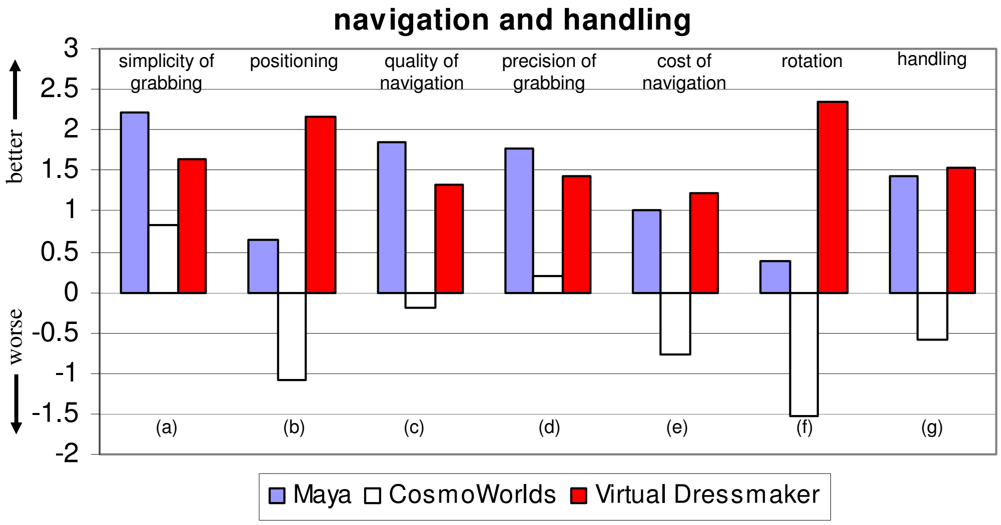
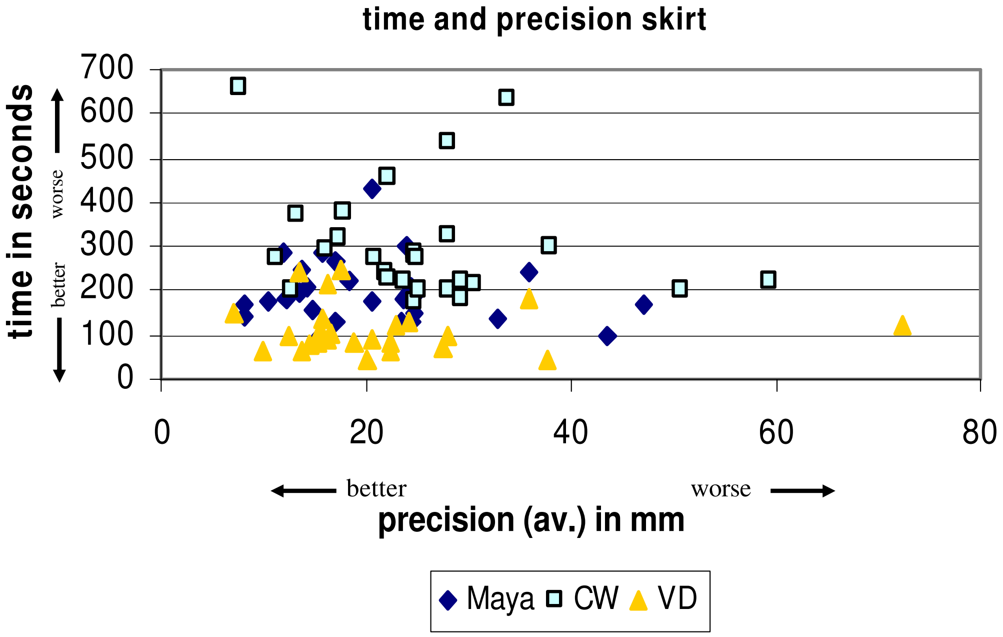


Figure 5.13: Results of the questionnaire concerning the user's subjective impressions of navigation and interaction in each application.

Regarding the order of the applications and the prior experience of the users in Maya we receive the following results. In figure 5.15, we plotted the average precision and completion times of the Maya specialists and the ones not having much knowledge in Maya (indicated by (M. beg). Only minor differences can be noted between these groups while for the skirt task the non-experts performed even slightly better than the experts. We obtain the same impression by figure 5.16, where the completion times and precision of the patterns are grouped for the participants who began with Maya (Maya 1.), Virtual Dressmaker (VD 1.), and CosmoWorlds (CW 1.). The measures are given for each application and are indicated by - Maya, - VD, - CW. We do not note any significant dependencies also in this case either. Hence, our hypothesis (H1) is approved. The user's personal impressions affirm our hypothesis, too (see section 5.3.3.2). However, the user study shows that with respect to precision, 2D desktop applications are preferred by the users. We noted that constrained movements (translation and rotation of the patterns) help enormously to increase the precision of the positioning.

To evaluate the second hypothesis, we first compare the results of our pilot study and the new ones with respect to the new interaction tools such as the Navigation Sphere and the highlighting for picking objects. From the subjective impressions (figure 5.13), we note that the participants prefer navigation tools which can be easily switched to without paying much at-



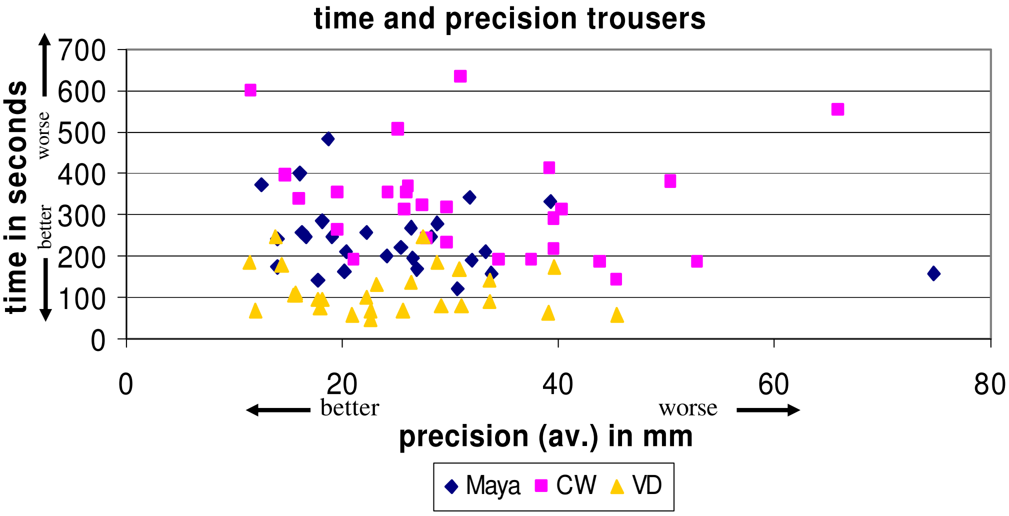


Figure 5.14: Combined evaluation of the time and precision measurements. Altogether, the Virtual Dressmaker achieved the best results, followed by Maya and CosmoWorlds.

tention to the switching itself. Furthermore, better feedback techniques like the highlighting ease the completion of the tasks. Second, we compare the performance in Maya and CosmoWorlds, and we can state that the reduction of switching between tools and viewpoints dramatically increases efficiency.

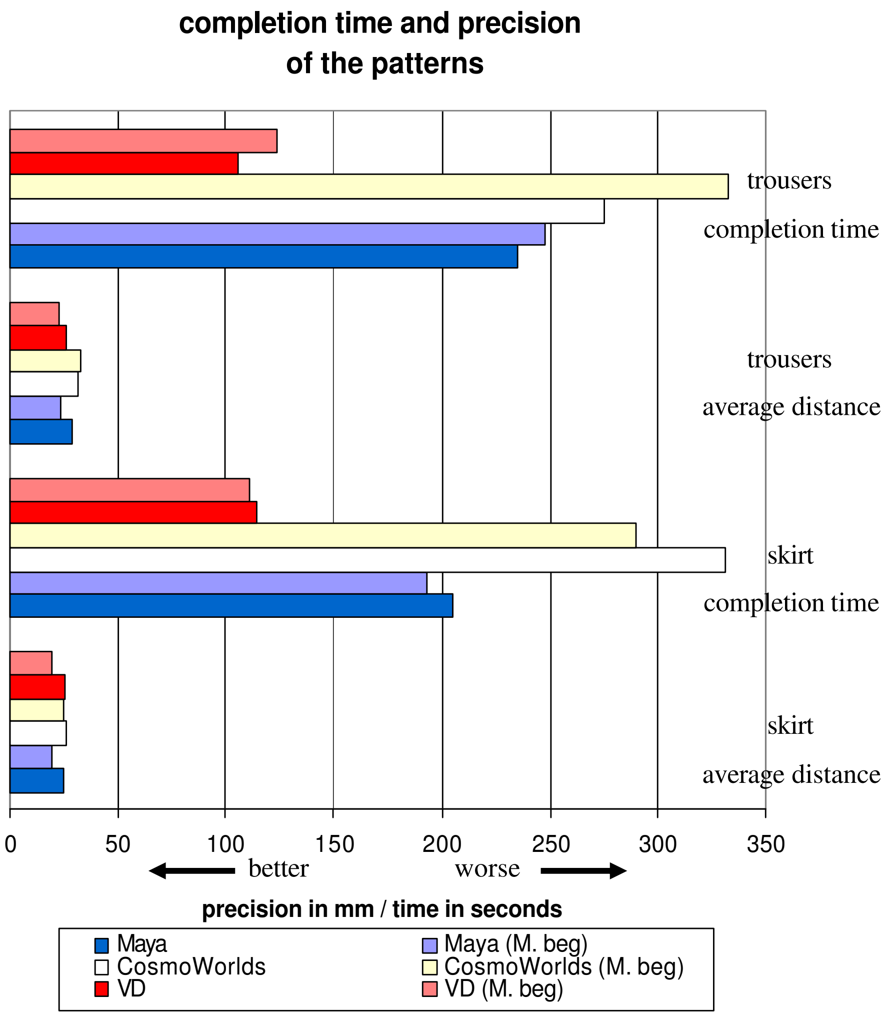


Figure 5.15: Results of the measurements grouped in Maya experts versus Maya beginners (M. beg).

As approved from the results shown in figures 5.11, 5.12, and 5.14, we observe that shortcuts ease the application and should be the subject of further investigations to improve the handling of VR applications. Finally, the view dependent rendering of our scene with the 3D stereo glasses also reduces the navigation in the scene since the user can change his view by simply looking from another angle. Therefore, shortcuts and predefined views allow to enhance the efficiency of the navigation considerably, and hypothesis (H2) is approved, too.

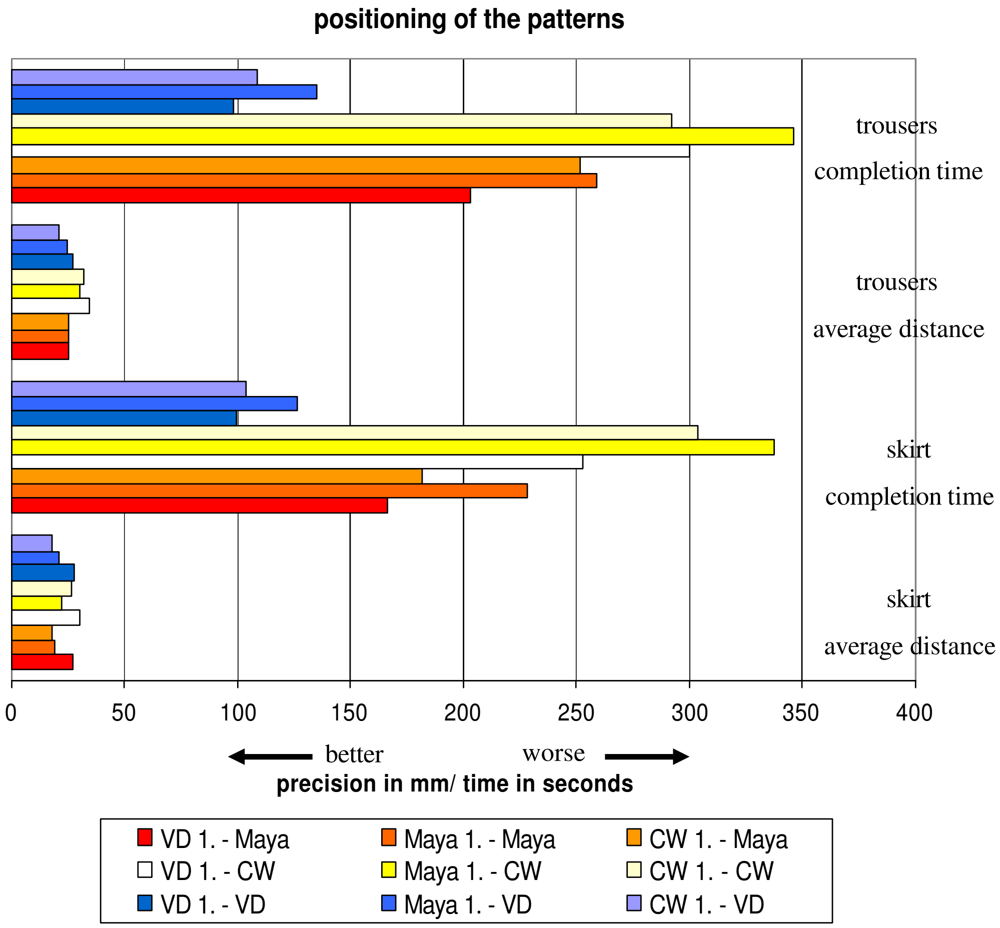


Figure 5.16: Results of the measurements grouped according to the order in which the users tested the applications.

5.3.5 Consequences of the User Studies

Critical observations during the user study from both the instructors and the users provided valuable suggestions for improvements of our virtual reality application.

5.3.5.1 Ergonomics

The ergonomics of the pen (e.g. sharp edges) and the shutter glasses (not fitting) should be improved in order to minimize the distraction of the users, and to enable them to concentrate on the given task. The same holds for the disturbing cables of the electromagnetic tracking device. Moreover, problems with the calibration of real and virtual pen and palette became evident and turned out to distract the users.

5.3.5.2 Improved Navigation Methods

The results of the usability evaluations indicate that the cloth assembly task is considerably improved by virtual reality input devices and stereo displays compared to common 2D desktop applications with respect to both efficiency and precision. However, the evaluations also show that the test users were not completely satisfied with the provided navigation tools. At the time of the user studies, two different navigation methods were available. First, it was possible to use sliders on the palette for rotation around the figure, translation in the vertical direction and zooming. Second, scene navigation was possible via a Navigation Sphere located in the lower right corner of the scene, which allows the same degrees of freedom as the sliders. In both cases, we constrained the navigation possibilities in order to model natural movement that would be possible in a comparable real scene. The disadvantage of both tools is that they require the user to trigger a navigation tool with the pen while working with the garments in the 3D scene.

To enhance the ease of navigation and as a result of our usability studies, we integrated the well known navigation methods *Scene in Hand* and *Flying Vehicle* [131] into the application which both can be used without the need to trigger any tools but simply by using the pen's second button. Moreover, we improved the Scene in Hand tool by constraining the navigation possibilities in the same way as for the described sliders and the Navigation Sphere. This results in a *Constrained Scene in Hand* navigation tool, which we think is the optimal choice among the given methods in the context of cloth assembly and design. It provides the same navigation constraints as the Navigation Sphere but the user is less distracted from the working area and remains immersed in the scene. Currently, the choice of navigation tool is up to the user and visualized in the application by the respective virtual counterpart of the pen. Thus, if the Flying Vehicle is chosen, the virtual pen is replaced by an arrow, whereas we use the 3D model of a hand for the Scene in Hand and the Constrained Scene in Hand tools.

As can be seen from the comparison of Maya and CosmoWorlds, navigation shortcuts could possibly improve scene navigation in virtual reality applications. The users found it very distracting and complicated to switch to the panel to change the scene view, and we noticed that the limiting factor of all applications was the permanent switching between different tools and views. Especially for virtual environments, this point seems to be capable of essentially improving acceptance and usability.

5.4 Interactive Garment Design

In this section, we extend the previously described interaction techniques by adding virtual tailor tools for interactive garment design during the physical simulation (figure 5.17). We provide virtual pins, virtual scissors, and a virtual sewing needle which allow the user to cut into the clothes and to add new seams in 3D as desired. The manipulations are immediately integrated into the simulation giving the user the possibility to modify the shape and appearance of garments in an efficient way directly on a dressed avatar. The modifications in 3D are automatically transferred to the underlying planar patterns. With this approach, it is possible to construct virtual prototypes of cloth patterns, and thus of clothes, by directly working on the complete three-dimensional and dynamically simulated model.

We follow parts of [5, 16].

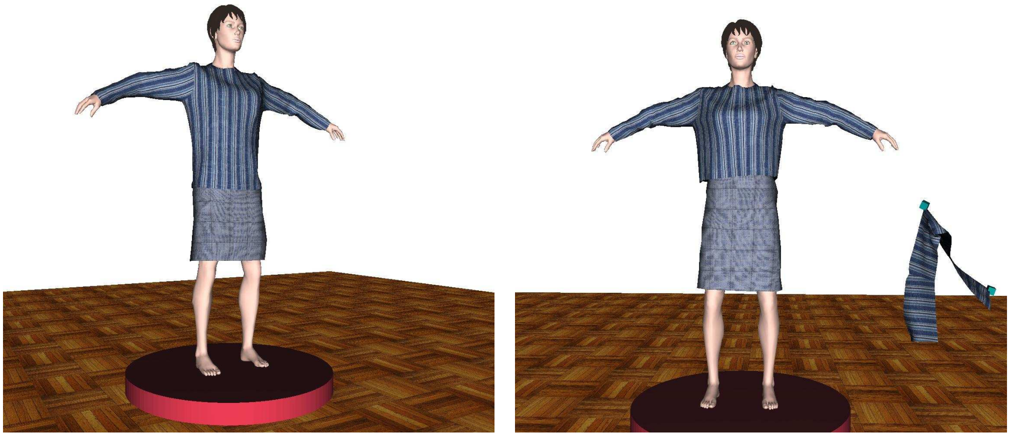


Figure 5.17: On the left, a woman wearing a pullover and a skirt is shown. On the right, the user has interactively shortened the pullover by cutting off a broad band on the lower part of the pullover using virtual scissors. The modified planar patterns are generated automatically.

5.4.1 Virtual Pins

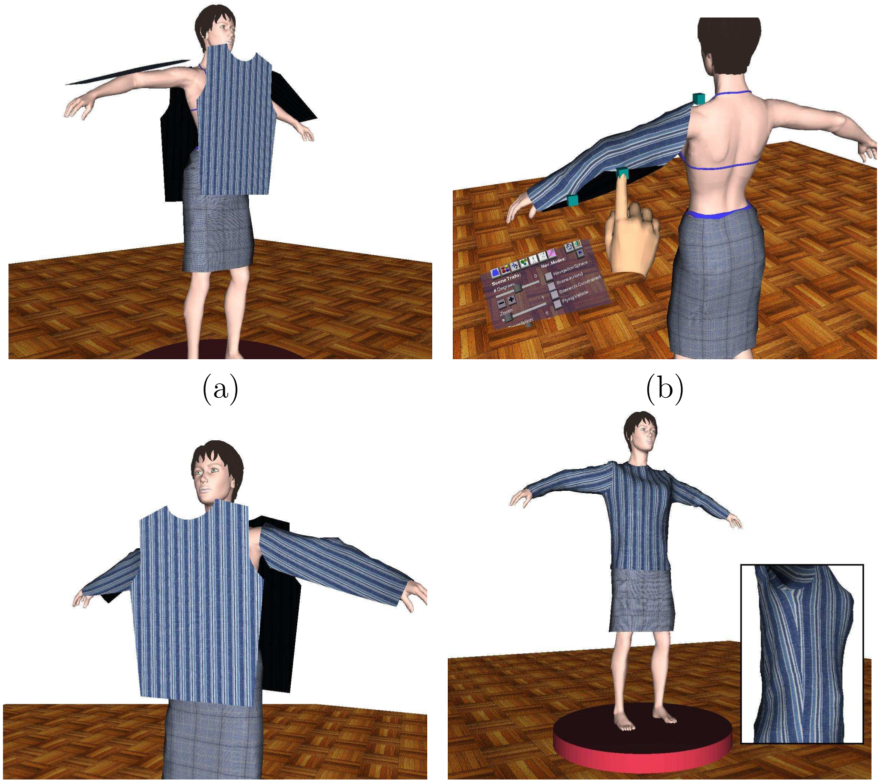
With the previously described interaction methods for cloth assembly, it is not possible to use cloth patterns that have to be bent in order to enable the sewing process, such as the sleeves of a pullover. In such cases, there are seam lines on opposite borders of the pattern that have to be connected with each other, which is not possible if the pattern remains flat. Hence, we use the cloth simulation engine in combination with the input devices in order to model the effect of wrapping the pattern around the arm with virtual pins. These virtual pins are integrated into the physical cloth simulation as constraints (cf. section 5.2.4).

To this end, we provide the possibility to simulate a single pattern with the seams not yet connected. To wrap the sleeve around the arm, the user selects the relevant pattern and starts the single pattern simulation. Then, the user is enabled to pin parts of the cloth to arbitrary positions, while for the rest of the sleeve the drape is simulated (figure 5.18). When a satisfactory position is reached, the regular sewing and simulation process can be started.

5.4.2 Virtual Scissors

In this section, we want to provide the user with virtual scissors that allow to cut the dressed garment. In order to define a cut on the simulated 3D mesh, the user has to mark points on the garment, i.e. vertices of the underlying mesh, with the pen. Between these selected vertices, the mesh will be cut in straight lines, and the corresponding patterns and seams are modified accordingly. Then, the physical simulation continues computing the drape and movement of the garment.

There are two essentially different ways of modifying a mesh to model the effects of cutting. On the one hand the affected triangles in the mesh can be split, on the other hand the cuts can be modeled along the edges of the mesh. Since the second approach in general does not result in straight cuts, the vertices of the cut edges subsequently have to be repositioned without affecting the underlying simulation. In both cases the number of vertices in the mesh is increased by a cut, while the number of triangles remains the same only in the latter case. We choose to cut the meshes along the edges and to recalculate the position of the vertices, because this approach fits well together with our sewing algorithm described in section 2.2. Thus, the special case of cutting along a seam line can be handled simply by removing the corresponding seams (see the left image in figure 5.23).



(a) (b)

(c) (d)

Figure 5.18: The sleeves of the pullover are prepositioned interactively using virtual pins. Image (a) shows the flat patterns positioned around the 3D figure. In image (b), the user wraps a sleeve around an arm. Image (c) shows the pullover in the final prepositioned state ready for sewing. The simulation result is shown in image (d).

The method for cutting clothes which we propose is summarized in algorithm 7 and will be described in the following.

Given two vertices v_1 and v_2 which define a cut, we first check if both of them are on the same seam. In this case, we assume that the user wants to unsew the chosen seam. Then, we only have to remove the vertex pairs in the corresponding seam list between v_1 and v_2, and no further modifications on the mesh (or on the patterns) are necessary.

If v_1 and v_2 are not seam points of the same seam, we have to find the optimal sequence of edges for a straight cut between v_1 and v_2. Therefore, we determine a *cutting plane*, which contains the vertices v_1 and v_2, and which

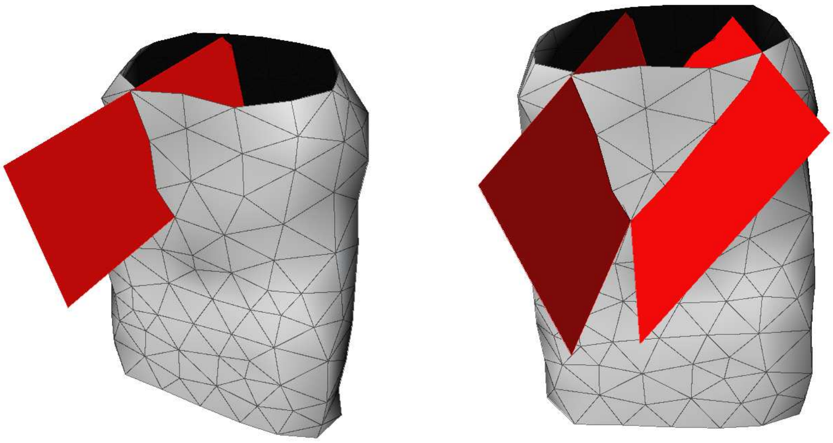


Figure 5.19: A woman's skirt is cut with the virtual scissors. In the left image, the cutting plane has been computed, and the vertices along the cut edges have been straightened. For the second cutting plane shown on the right, the vertices have not been moved yet.

is approximately orthogonal to the mesh surface. The cut is then realized as the intersection between the cutting plane and the mesh (see figure 5.19).

Algorithm 7 Cutting Clothes in 3D

```
Require: start vertex $v_1$, end vertex $v_2$
if $v_1$ and $v_2$ are seam points of the same seam then
    Remove the part of the seam between $v_1$ and $v_2$.
else
    Determine a path along the edges of the garment mesh between $v_1$ and $v_2$ using Dijkstra's algorithm.
    Average the vertex normals along these edges, and compute the cutting plane.
    Compute a new sequence of edges between $v_1$ and $v_2$, such that the edges are as close as possible to the cutting plane.
    Split the 3D mesh and the respective planar patterns along these edges, and straighten the cutting line in the 3D mesh by moving the corresponding vertices in the direction of the cutting plane.
    Transfer the position changes of the vertices in the 3D mesh to the planar patterns using barycentric coordinates.
end if
Continue the physical simulation.
```

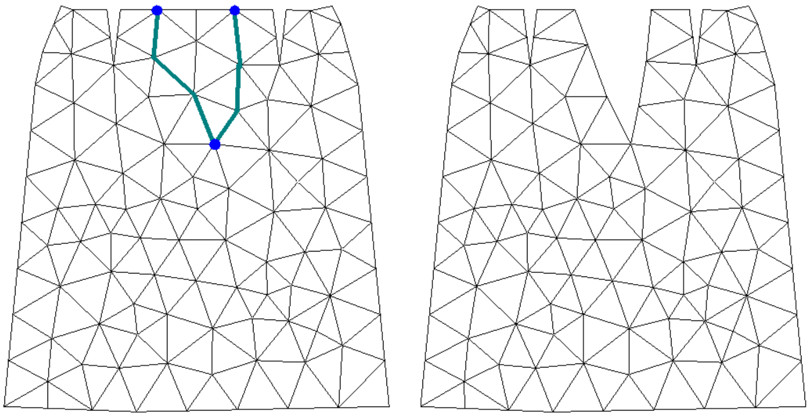


Figure 5.20: The front pattern of the skirt before (left) and after (right) the cutting (cf. figures 5.19 and 5.22). In the left pattern, the vertices selected by the user, and the edges that have to be cut are highlighted. Note that the final cutting lines in the planar patterns are not necessarily straight, as shown in the right image, due to the deformed state of the 3D mesh during the cutting.

To achieve this, we split the mesh along a sequence of edges close to the cutting plane and align the vertices on both sides of the plane to ensure a straight cutting line. Finally we update the underlying cloth patterns and seam information by the new mesh information to obtain a consistent representation of the meshes.

To determine the cutting plane, we first use Dijkstra's algorithm [50] to compute an initial path between v_1 and v_2 along the edges of the mesh. Then, we average the vertex normals of the concerned edges in order to estimate the average surface normal along the cutting line on the mesh. The cutting plane is then given by the span of this normal and the vector between v_1 and v_2. This provides a plane which is preferably orthogonal to the draped cloth and works well if we assume the concerned part of the garment to be locally smooth, i.e. with not too much curvature.

Now, we compute a new sequence of edges between v_1 and v_2 using a modified Dijkstra's algorithm, requiring in addition that the edges are as close to the cutting plane as possible. Then we split the 3D mesh and the respective patterns along these edges. If no seam points are involved, the mesh can be split by doubling vertices and edges between v_1 and v_2 and thus generating new borders of the mesh. Cutting over seam points requires the modification of the underlying seam information, and involves several

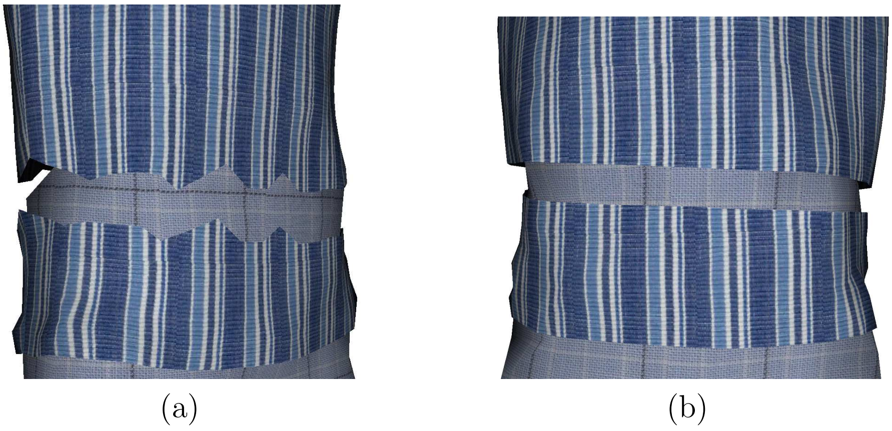

(a) (b)

Figure 5.21: A pullover is shortened by cutting a piece off. Image (a) shows the cut without aligning the vertices along the cut edges. In (b) the vertex positions have been corrected and a straight cut is obtained. All the modifications are automatically transferred to the planar patterns.

special cases that have to be handled separately, depending for example on how much of the seam is involved in the cut. An example of a cut over seams can be seen in figures 5.17 and 5.21.

In order to obtain a straight cutting line, we move the vertices of the concerned edges in the 3D mesh towards the direction of the cutting plane, remaining in the plane spanned by the respective triangle (figure 5.21). These changes of vertex positions in the 3D mesh are then transferred to the planar cloth patterns using barycentric coordinates. With this method, the relative displacements between the deformed state and the planar patterns are conserved, and a correct rest state for the continuation of the physical simulation is provided.

Note that we do not cut along the initial sequence of edges given by Dijkstra's algorithm, because moving the vertices onto a straight cutting line could be hindered by vertices which lie between the edges and the optimal cutting line and which are not chosen by the first run of the algorithm.

5.4.3 Virtual Sewing Needle

Similar to defining a cut on the 3D garment, a new seam consisting of several vertices can be created by selecting them in pairs with the pen resulting in

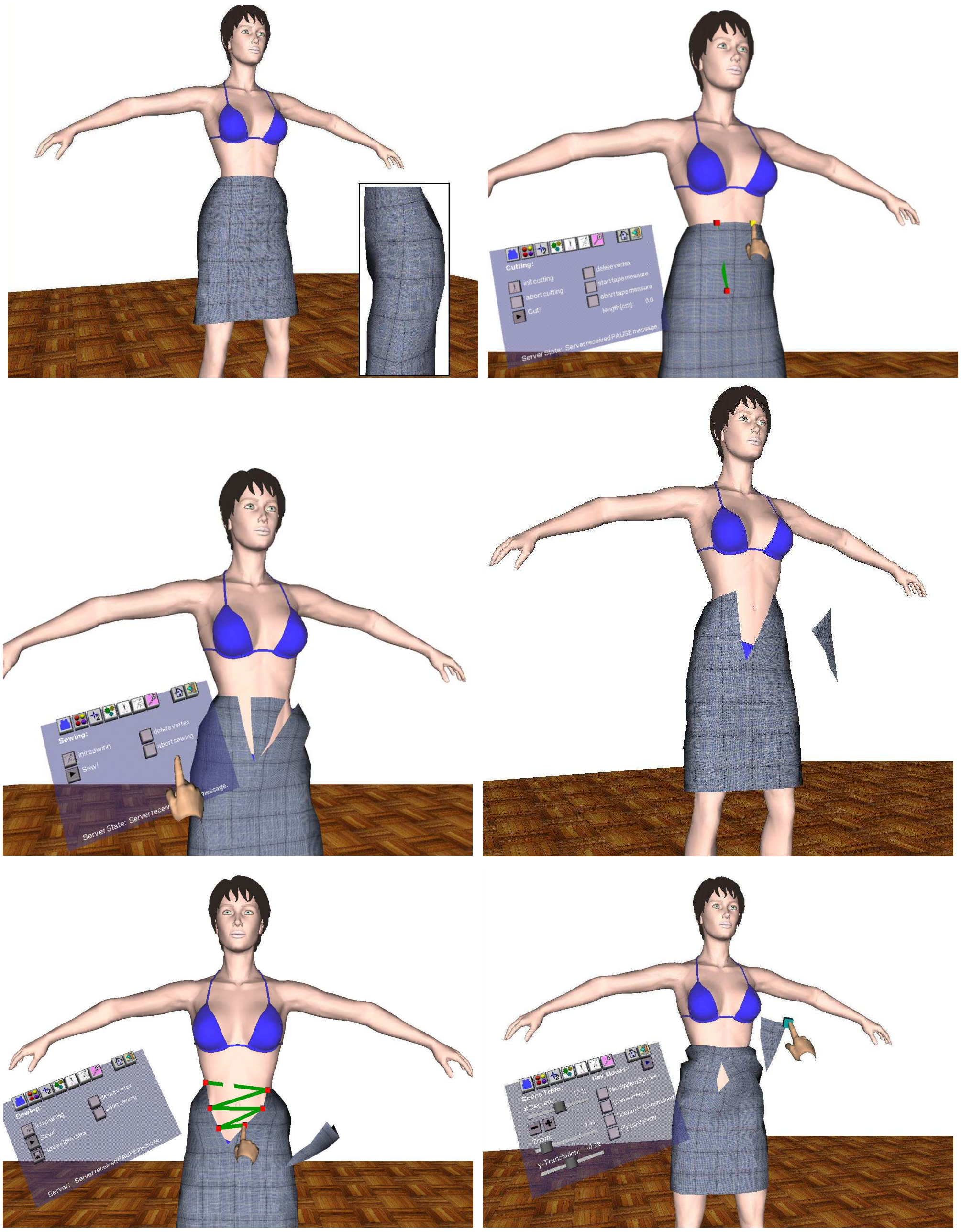


Figure 5.22: A piece of cloth is cut off a skirt in order to make it tighter. First, the cut is defined (top row), then the virtual scissors separate a piece of cloth (middle row). The automatically modified planar pattern is shown in figure 5.20. Finally, the now tighter skirt is sewn again (bottom row).

a seam list of vertex pairs. The seams are shown to the user by seam lines and for wrongly placed seams, the user can cancel the last defined seam by pressing a button on the corresponding menu. Then, each vertex pair is sewed and the complete cloth mesh is created as described in algorithm 1 in section 2.2. For an example, see figure 5.22.

5.4.4 Results

The examples given in figures 5.17, 5.22, and 5.23 show some results of the proposed algorithms and interaction techniques for the design of garments. In figure 5.17, a pullover is shortened by cutting a broad band off from its lower border. A skirt is tightened in figure 5.22 by first removing a piece of fabric and then sewing the skirt again at the cut borders (a so called *dart*). In figure 5.23, a seam of the pullover is opened to change it's look, and the design of the collar is changed. All simulation images are directly captured from our Virtual Dressmaker application while accomplishing the given task.

5.5 Summary

The system described in this work is intended to be a part of a complete virtual tailor room. In this scenario, the whole process from the design over the tailoring to the try-on and customization of virtual garments shall be supported by virtual reality tools and 3D interaction techniques. To reach this goal, high requirements for all aspects of the application including the human-computer interaction techniques, the mesh modification algorithms, and the cloth simulation engine have to be met.

We have presented the Virtual Dressmaker, an application that allows to select, assemble, on-the-fly interact, and try-on various garments on 3D characters within an immersive virtual environment. The Virtual Dressmaker is a self-contained application including an integrated cloth simulation engine, realized through a flexible client-server system architecture. The proposed application provides two handed interaction combined with a head-tracked stereoscopic view and immerses the user into an almost live-size working environment. It has been verified by two usability evaluations, that this working environment provides a considerable speed-up of the garment assembly process compared to standard 2D interface tools. We have experimented with several methods for scene navigation in this context, and proposed new navigation concepts which constrain the user's movement in a way that resembles

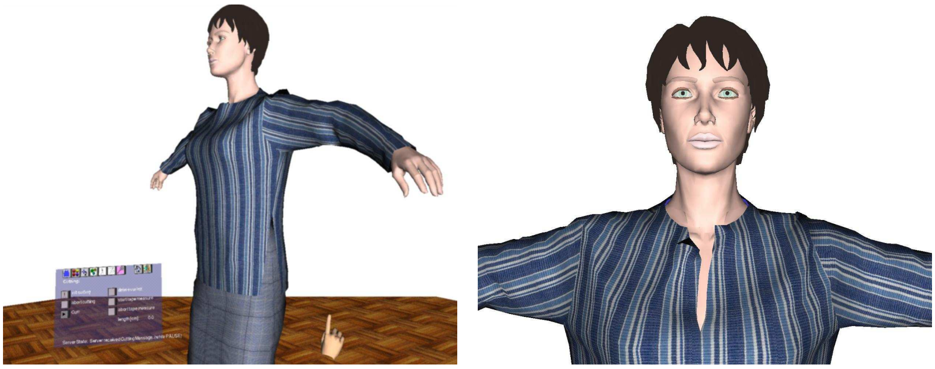


Figure 5.23: In the left image, a seam has been partially opened to change the design of the pullover's side. On the right, the collar has been modified using the proposed virtual tailor tools.

a real tailor's movement around a real mannequin. Furthermore, it is possible to select and drag parts of the clothes during the simulation, a tool that is difficult to use with 2D interaction.

Moreover, we developed virtual tailor tools for the interactive design of garments. In particular, virtual pins, a virtual pair of scissors, and a virtual sewing needle have been presented, which allow to cut, sew, and design garments in 3D. The modifications are automatically transferred to the concerned planar patterns, and the seam information is updated accordingly. Thus, virtual prototypes of clothes can be constructed in the Virtual Dressmaker application by working directly on the 3D models.

Chapter 6

Outlook on Future Research

In their book "Cloth Modeling and Animation", Donald H. House and David E. Breen state that "the Holy Grail for cloth modeling and animation would be the development of technologies that allow the average person, as well as the expert, to design real clothing for real people", and that "such a system would have wide applications across the clothing industry, from fashion design to direct consumer use." [75, p.332]. In this work, we presented contributions to several aspects of the research in this context, including simulation quality, computation speed, and applications for virtual try-on and virtual tailoring. In order to realize a complete virtual tailor room, there are several challenges for future research in all of these areas.

Simulation Quality. Regarding internal forces in textiles, it should be investigated if alternative models for bend forces, for instance derived from thin-plate energies or shell theory models could be integrated into the presented framework for rotated linear finite elements without too much loss of performance. This is particularly important, as bending has a relatively strong influence on the draping behavior of textiles. Next, it will be of importance to model details like seams and predefined folds in garments, as they also occur frequently in real garments, for instance in the collar of a jacket. Here, a combination with rigid body simulation techniques appears to be interesting. This also would allow to integrate accessories like buttons, zippers etc. into the physical simulation.

Computation Speed. In this work we showed that substantial performance speedups for cloth simulations can be achieved by parallel computing. In order to realize a full parallel cloth simulator it will be necessary to add parallel algorithms for collision detection and response for arbitrary collision objects. While it is possible to handle collisions between the textiles and rigid bodies by simply reproducing all rigid collision objects on all computation nodes, the efficient parallel processing of cloth self-collisions seems to be far from trivial. In principal, every part of a garment might collide with every other part, depending for instance on the movement of the character, thus it seems to be an interesting question how to optimize the mesh partitioning in this case. Another challenge will be to obtain interactive frame rates in parallel simulations with large numbers of triangles. To achieve this, all necessary data to produce a frame, i.e. the vertex positions, have to be collected on one display node from all computing nodes. For a real-time simulation this should be possible at least 25 times a second, which results in high requirements for the network communication. An interesting alternative to parallelization on PC clusters is the use of specialized hardware like (clusters of) GPUs or FPGAs. The latter possibility is currently investigated at WSI/GRIS in order to accelerate explicit and implicit time integration methods in physical simulations in general.

From an algorithmic point of view, the resolution independence of the presented finite element model allows the consistent use of multi-resolution schemes, therefore numerical multigrid methods and cascadic conjugate gradient methods should be considered as alternatives to the standard conjugate gradient method for solving the linear equation systems in implicit time integration schemes.

Virtual Try-On & Virtual Tailoring. In general, current textile CAD databases do not contain all the information that is necessary to enable a fully automatic three-dimensional visualization and simulation. Mostly, the boundaries of the garment patterns are available only with seam allowances, and the seam line information is missing. While this is not an original computer graphics problem, it is nevertheless a very important topic if virtual try-on shall be applied automatically for large fashion lines. Regarding the proposed system for virtual tailoring and garment design, improved interaction tools, for instance for the interactive definition, visualization, and modification of seams, material parameters, and textures have to be added. In addition, alternative virtual reality interaction and display devices and a possibility for multi-user interaction are desirable to make the interaction more efficient. Finally, in order to not only model the look but also the feel of cloth, the integration of tactile feedback devices could significantly enhance the perception of virtual textiles.

Publications

Articles in Conference Proceedings and Journals

[1] O. Etzmuß, M. Keckeisen, S. Kimmerle, J. Mezger, M. Hauth, and M. Wacker. A Cloth Modelling System for Animated Characters. In *Proc. Graphiktag*, 2001. (Cited on page 54.)

[2] O. Etzmuß, M. Keckeisen, and W. Straßer. A Fast Finite Element Solution for Cloth Modelling. In *Proc. Pacific Graphics*, 2003. (Cited on page 16.)

[3] M. Gruber, C. Michel, S. Pabst, M. Wacker, M. Keckeisen, and S. Kimmerle. tcCloth - An Interactive Cloth Modeling and Animation System. In *Proc. Graphiktag*, 2004. (Cited on pages 54 and 55.)

[4] M. Keckeisen and W. Blochinger. Parallel Implicit Integration for Cloth Animations on Distributed Memory Architectures. In *Proc. Eurographics Symposium on Parallel Graphics and Visualization*, 2004. (Cited on pages 39 and 48.)

[5] M. Keckeisen, M. Feurer, and M. Wacker. Tailor Tools for Interactive Design of Clothing in Virtual Environments. In *Proc. ACM Symposium on Virtual Reality Software and Technology*, 2004. (Cited on pages 13 and 89.)

[6] M. Keckeisen, S. Kimmerle, B. Thomaszewski, and M. Wacker. Modelling Effects of Wind Fields in Cloth Animations. In *Proc. WSCG*, 2004. (Cited on page 28.)

[7] M. Keckeisen, S. L. Stoev, M. Feurer, and W. Straßer. Interactive Cloth Simulation in Virtual Environments. In *Proc. IEEE Virtual Reality*, 2003. (Cited on pages 65 and 72.)

[8] M. Keckeisen, S. L. Stoev, M. Wacker, M. Feurer, and W. Straßer. A User Study On Advanced Interaction Techniques in the Virtual

Dressmaker Application. In *Proc. International Conference on Human-Computer Interaction*, 2003. (Cited on page 72.)

[9] S. Kimmerle, M. Keckeisen, J. Mezger, and M. Wacker. TüTex: A Cloth Modelling System for Virtual Humans. In *Proc. 3D Modelling*, 2003. (Cited on pages 23, 26, 54 and 55.)

[10] N. Magnenat-Thalmann, F. Cordier, M. Keckeisen, S. Kimmerle, R. Klein, and J. Meseth. Simulation of Clothes for Real-time Applications. In *Proc. Eurographics 2004, Tutorial 1*, 2004. (Cited on pages 5 and 53.)

[11] M. Wacker, M. Keckeisen, S. Kimmerle, and J. Mezger. Fortschritte in der Textilsimulation für Fashion on Demand. In *Tagungsband 1. Paderborner Workshop Augmented Reality Virtual Reality in der Produktentstehung*, 2002. (Cited on page 54.)

[12] M. Wacker, M. Keckeisen, S. Kimmerle, W. Straßer, V. Luckas, C. Groß, A. Fuhrmann, R. Sarlette, M. Sattler, and R. Klein. Virtual Try-On: Virtuelle Textilien in der Graphischen Datenverarbeitung. *Informatik Spektrum*, 27(6):504–511, 2004. (Cited on pages 53, 54 and 58.)

[13] M. Wacker, M. Keckeisen, S. Kimmerle, W. Straßer, V. Luckas, C. Groß, A. Fuhrmann, M. Sattler, R. Sarlette, and R. Klein. Simulation and Visualisation of Virtual Textiles for Virtual Try-On. *Research Journal of Textile and Apparel*, 2005. Accepted for publication. (Cited on pages 53, 54 and 58.)

[14] M. Wacker, M. Keckeisen, S. L. Stoev, and W. Straßer. A Comparative Study On User Performance in the Virtual Dressmaker Application. In *Proc. ACM Symposium on Virtual Reality Software and Technology*, 2003. (Cited on page 72.)

Technical Reports

[15] O. Etzmuß and M. Keckeisen. A Linearised Finite Element Model for Cloth Animation. Technical Report WSI-2003-2, Universität Tübingen, 2003. (Cited on page 16.)

[16] M. Keckeisen, M. Feurer, and M. Wacker. Interactive Cutting and Sewing of Virtual Clothes. Technical Report WSI-2004-5, Universität Tübingen, 2004. (Cited on pages 13 and 89.)

[17] M. Keckeisen, S. L. Stoev, and M. Feurer. The Virtual Dressmaker: Interactive Cloth Assembly and Try-On in Virtual Environments. Technical Report WSI-2002-6, Universität Tübingen, 2002. (Cited on pages 65 and 72.)

Exhibitions

[18] *Virtual Try-On*. Internationale BMBF-Statustagung *Virtuelle und Erweiterte Realität*, 2004, Leipzig, Germany, February 19-20. (Cited on page 54.)

[19] *Virtual Try-On*. CeBIT 2004, Hannover, Germany, March 18-24, Hall 11, Booth D32. (Cited on pages 54 and 60.)

Bibliography

[20] M. F. Adams and J. Demmel. Parallel Multigrid Solver for 3D Unstructured Finite Element Problems. In *Proc. ACM/IEEE Conference on Supercomputing*, 1999. (Cited on page 40.)

[21] I. Angus and H. Sowizral. Embedding the 2D Interaction Metaphor in a Real 3D Virtual Environment. In *Proc. SPIE*, 1995. (Cited on page 64.)

[22] U. Ascher and E. Boxerman. On the modified conjugate gradient method in cloth simulation. *The Visual Computer*, 19(7–8):526–531, 2003. (Cited on page 40.)

[23] S. Balay, K. Buschelman, W. D. Gropp, D. Kaushik, M. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang. PETSc home page. http://www.mcs.anl.gov/petsc, 2001. (Cited on page 44.)

[24] S. Balay, K. Buschelman, W. D. Gropp, D. Kaushik, M. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang. PETSc users manual. Technical Report ANL-95/11 - Revision 2.1.5, Argonne National Laboratory, 2002. (Cited on pages 44 and 55.)

[25] S. Balay, W. D. Gropp, L. C. McInnes, and B. F. Smith. Efficient management of parallelism in object oriented numerical software libraries. In E. Arge, A. M. Bruaset, and H. P. Langtangen, editors, *Modern Software Tools in Scientific Computing*, pages 163–202. Birkhauser Press, 1997. (Cited on pages 44 and 55.)

[26] D. Baraff and A. Witkin. Large Steps in Cloth Simulation. In *Proc. SIGGRAPH*, 1998. (Cited on pages 3, 4, 5, 10, 23, 39 and 42.)

[27] D. Baraff, A. Witkin, and M. Kass. Untangling Cloth. In *Proc. SIGGRAPH*, 2003. (Cited on page 40.)

[28] K. S. Bath, C. D. Twigg, J. K. Hodgins, P. K. Khosla, Z. Popovic, and S. M. Seitz. Estimating Cloth Simulation Paramters from Video. In *Proc. SIGGRAPH Symposium on Computer Animation*, 2003. (Cited on page 11.)

[29] B. Becker, D. Lane, and N. Max. Unsteady Flow Volumes. In *Proc. Visualization*, 1995. (Cited on page 11.)

[30] D. Bielser, P.Glardon, M. Teschner, and M. Gross. A State Machine for Real-Time Cutting of Tetrahedral Meshes. In *Proc. Pacific Graphics*, 2003. (Cited on page 64.)

[31] M. Billinghurst, S. Baldis, L. Matheson, and M. Philips. 3D Palette: A Virtual Reality Content Creation Tool. In *Proc. ACM Symposium on Virtual Reality Software and Technology*, 1997. (Cited on page 64.)

[32] J. Bonet and R. D. Wood. *Nonlinear continuum mechanics for finite element analysis*. Cambridge University Press, 2000. (Cited on pages 11, 16, 17, 18 and 20.)

[33] D. Braess. *Finite Elemente*. Springer, 1997. (Cited on page 22.)

[34] D. E. Breen. Computer Graphics in Textiles and Apparel Modeling (Guest Editor's Introduction). *IEEE Computer Graphics and Applications*, 16(5):26–27, 1996. (Cited on page 1.)

[35] D. E. Breen, D. H. House, and M. J. Wozny. Predicting the Drape of Woven Cloth Using Interacting Particles. In *Proc. SIGGRAPH*, 1994. (Cited on page 10.)

[36] R. Bridson, S. Marino, and R. Fedkiw. Simulation of Clothing with Folds and Wrinkles. In *Proc. SIGGRAPH Symposium on Computer Animation*, 2003. (Cited on page 40.)

[37] W. Buxton and B. A. Myers. A Study in Two-Handed Input. In *Proc. ACM CHI 86 Conference on Human Factors in Computing Systems*, 1986. (Cited on page 64.)

[38] J. Chen, N. da Vittorio Lobo, C. Hugues, and J. Moshell. Real-Time Fluid Simulation in a Dynamic Virtual Environment. *IEEE Computer Graphics and Applications*, 17(3):52–61, 1997. (Cited on page 12.)

[39] K.-J. Choi and H.-S. Ko. Stable but Responsive Cloth. In *Proc. SIGGRAPH*, 2002. (Cited on pages 3, 10, 12 and 39.)

[40] K.-J. Choi and H.-S. Ko. Extending the Immediate Buckling Model to Triangular Meshes for Simulating Complex Clothes. In *Eurographics Short Presentations*, 2003. (Cited on pages 10 and 39.)

[41] A. Chorin and J. Marsden. *A Mathematical Introduction to Fluid Dynamics*. Springer, 1990. (Cited on page 29.)

[42] S. Coquillart and G. Wesche. The Virtual Palette and the Virtual Remote Control Panel: A Device and an Interaction Paradigm for the Responsive Workbench. In *Proc. IEEE Virtual Reality*, 1999. (Cited on page 64.)

[43] F. Cordier and N. Magnenat-Thalmann. Real-time Animation of Dressed Virtual Humans. *Proc. Eurographics*, 2002. (Cited on page 40.)

[44] F. Cordier, H. Seo, and N. Magnenat-Thalmann. Made-to-Measure Technologies for an Online Clothing Store. *IEEE Computer Graphics and Applications*, 23(1):38–48, 2003. (Cited on page 53.)

[45] U. Cugini and C. Rizzi. 3D Design and Simulation of Men Garments. In *Proc. WSCG*, 2002. (Cited on page 62.)

[46] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry — Algorithms and Applications*. Springer, 1997. (Cited on page 23.)

[47] B. de Veubeke. The dynamics of flexible bodies. *Int. J. Engrg. Sci.*, pages 895–913, 1976. (Cited on page 11.)

[48] G. Debunne, M. Desbrun, M.-P. Cani, and A. H. Barr. Dynamic Real-Time Deformations using Space and Time Adaptive Sampling. In *Proc. SIGGRAPH*, 2001. (Cited on page 10.)

[49] H. Delingette, S. Cotin, and N. Ayache. A Hybrid Elastic Model allowing Real-Time Cutting, Deformations and Force-Feedback for Surgery Training and Simulation. In *Proc. Computer Animation Conference*, 2000. (Cited on page 64.)

[50] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959. (Cited on page 93.)

[51] E-Tailor project web page. http://www.atc.gr/e-tailor. (Cited on page 53.)

[52] B. Eberhardt. The Emperors New Cloth or How to calculate virtual textiles. Habilitation, Universität Tübingen, 2002. (Cited on page 5.)

[53] B. Eberhardt, O. Etzmuß, and M. Hauth. Implicit-Explicit Schemes for Fast Animation with Particle Systems. In *Eurographics Workshop on Computer Animation and Simulation*, 2000. (Cited on page 40.)

[54] B. Eberhardt, A. Weber, and W. Straßer. A Fast, Flexible Particle-System Model for Cloth Draping. *IEEE Computer Graphics and Applications*, 16(5):52–59, 1996. (Cited on pages 3 and 10.)

[55] J. W. Eischen, S. Deng, and T. G. Clapp. Finite-Element Modeling and Control of Flexible Fabric Parts. *IEEE Computer Graphics and Applications*, 16(5):71–80, 1996. (Cited on pages 4, 10 and 12.)

[56] O. Etzmuß. *Animation of Surfaces with Applications to Cloth Modelling*. PhD thesis, Universität Tübingen, 2002. (Cited on pages 4 and 67.)

[57] O. Etzmuß, J. Groß, and W. Straßer. Deriving a Particle System from Continuum Mechanics for the Animation of Deformable Objects. *IEEE Transactions on Visualization and Computer Graphics*, 9(4):538–550, 2003. (Cited on pages 4 and 10.)

[58] R. Fedkiw, J. Stam, and H. W. Jensen. Visual simulation of smoke. In *Proc. SIGGRAPH*, 2001. (Cited on pages 12, 27 and 30.)

[59] N. Foster and D. Metaxas. Realistic animation of liquids. *Graphical models and image processing: GMIP*, 58(5):471–483, 1996. (Cited on pages 12, 30 and 33.)

[60] N. Foster and D. Metaxas. Modeling the Motion of a Hot, Turbulent Gas. In *Proc. SIGGRAPH Symposium on Computer Animation*, 1997. (Cited on page 12.)

[61] S. F. Frisken-Gibson. Using Linked Volumes to Model Object Collisions, Deformation, Cutting, Carving, and Joining. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):333–348, 1999. (Cited on page 64.)

[62] A. Fuhrmann, C. Groß, V. Luckas, and A. Weber. Interaction-Free dressing of Virtual Humans. *Computers and Graphics*, Vol. 27(1):71–82, 2003. (Cited on pages 14, 58 and 64.)

[63] M. Gamito, P. Lopez, and M. Gomes. Two-dimensional Simulation of Gaseous Phenomena Using Vortex Particles. In *Proc. of the 6th Eurographics Workshop on Computer Animation and Simulation*, pages 3–15, 1995. (Cited on page 12.)

[64] O. Génevaux, A. Habibi, and J.-M. Dischler. Simulating Fluid-Solid Interaction. In *Proc. Graphics Interface*, 2003. (Cited on page 12.)

[65] M. Griebel, T. Dornseifer, and T. Neunhoeffer. *Numerical Simulation in Fluid Dynamics: A Practical Introduction.* SIAM, Philadelphia, 1998. (Cited on page 12.)

[66] C. Groß, A. Fuhrmann, and V. Luckas. Automatic pre-positioning of virtual clothing. In *Proc. Spring Conference on Computer Graphics*, 2003. (Cited on pages 14, 58 and 64.)

[67] Y. Guiard. The kinematic chain as a model for human asymmetrical bimanual cooperation. In A. Colley and J. Beech, editors, *Cognition and action in skilled behavior*, pages 205–228. Amsterdam: North-Holland, 1988. (Cited on page 64.)

[68] M. Hauth. *Visual Simulation of Deformable Models.* PhD thesis, Universität Tübingen, 2004. (Cited on pages 5, 26 and 40.)

[69] M. Hauth and O. Etzmuß. A High Performance Solver for the Animation of Deformable Objects using Advanced Numerical Methods. In *Proc. Eurographics*, 2001. (Cited on pages 26 and 40.)

[70] M. Hauth, O. Etzmuß, and W. Straßer. Analysis of Numerical Methods for the Simulation of Deformable Models. *The Visual Computer*, 19(7–8):581–600, 2003. (Cited on pages 5 and 40.)

[71] M. Hauth and W. Straßer. Corotational simulation of deformable solids. In *Proc. WSCG*, 2004. (Cited on pages 7 and 11.)

[72] O. Heguy, N. Rodriguez, and H. Luga. Virtual Environment for Cooperative Assistance in Teleoperation. In *Proc. WSCG*, 2001. (Cited on page 64.)

[73] B. K. Hinds and M. J. Interactive Garment Design. *The Visual Computer*, 6(2), 1990. (Cited on pages 5 and 62.)

[74] L. F. Hodges, B. O. Rothbaum, R. Alarcon, D. Ready, F. Shahar, K. Graap, J. Pair, P. Hebert, B. Wills, and D. Baltzell. Virtual

vietnam: A virtual environment for the treatment of chronic post-traumatic stress disorder. *CyberPsychology & Behavior*, 2(1):1–9, 1999. (Cited on page 64.)

[75] D. H. House and D. E. Breen, editors. *Cloth Modeling and Animation.* AK Peters, 2000. (Cited on pages 5 and 99.)

[76] G. Irving, J. Teran, and R. Fedkiw. Invertible Finite Elements for Robust Simulation of Large Deformation. In *Proc. SIGGRAPH Symposium on Computer Animation*, 2004. (Cited on pages 7 and 11.)

[77] L. Janski and V. Ulbricht. Numerical Simulation of Mechanical Behaviour of Textile Surfaces. *Zeitschrift für Angewandte Mathematik und Mechanik*, 80(S2):S525–S526, 2000. (Cited on pages 4 and 10.)

[78] L. Janski and V. Ulbricht. FE-Modellierung des freien Faltens von Textilien. In *Beiträge zur Modellierung und Identifikation, Berichte des Instituts für Mechanik, Universität Kassel*, 2001. (Cited on page 24.)

[79] V. Jaswa. CAVEvis: distributed real-time visualization of time-varying scalar and vector fields using the CAVE virtual reality theater. In *Proc. IEEE Visualization*, 1997. (Cited on page 64.)

[80] J. Kajiya and B. von Herzen. Ray Tracing Volume Densities. In *Proc. SIGGRAPH Symposium on Computer Animation*, 1984. (Cited on page 12.)

[81] Y.-M. Kang, J.-H. Choi, H.-G. Cho, and D.-H. Lee. An efficient animation of wrinkled cloth with approximate implicit integration. *The Visual Computer*, 17(3), 2001. (Cited on pages 12 and 40.)

[82] Y.-M. Kang, J.-H. Choi, H.-G. Cho, D.-H. Lee, and C.-J. Park. Real-time Animation Technique for Flexible and Thin Objects. In *Proc. WSCG*, 2000. (Cited on page 12.)

[83] G. Karypis and V. Kumar. Parallel Multilevel k-way Partitioning Scheme for Irregular Graphs. In *Proceedings of the 1996 ACM/IEEE Conference on Supercomputing*, 1996. (Cited on page 46.)

[84] G. Karypis and V. Kumar. Multilevel k-way Partitioning Scheme for Irregular Graphs. *Journal of Parallel and Distributed Computing*, 48(1):96–129, 1998. (Cited on page 46.)

[85] M. Kass and G. Miller. Rapid, Stable Fluid dynamics for Computer Graphics. In *Proc. SIGGRAPH Symposium on Computer Animation*, 1990. (Cited on page 12.)

[86] S. Kawabata. *The Standardization and Analysis of Hand Evaluation.* The Textile Machinery Society of Japan, Osaka, 1980. (Cited on page 24.)

[87] V. Kumar, A. Grama, A. Gupta, and G. Karypis. *Introduction to Parallel Computing — Design and Analysis of Algorithms.* Benjamin/Cummings, 1994. (Cited on pages 40, 42, 45 and 46.)

[88] R. Lario, C. Garcia, M. Prieto, and F. Tirado. Rapid Parallelization of a Multilevel Cloth Simulator Using OpenMP. In *Third European Workshop on OpenMP*, 2001. (Cited on page 40.)

[89] L. Li, M. Damoran, and R. K. Gay. Aerodynamic force models for animating cloth motion in air flow. *The Visual Computer*, 12(2):84–104, 1996. (Cited on pages 11 and 12.)

[90] C. Mendoza and C. Laugier. Simulating Cutting in Surgery Applications using Haptics and Finite Element Models. In *Proc. IEEE Virtual Reality*, 2003. Poster paper. (Cited on page 64.)

[91] J. Meseth, G. Müller, and R. Klein. Reflectance Field based real-time, high-quality Rendering of Bidirectional Texture Functions. *Computers and Graphics*, 28(1):103–112, 2004. (Cited on page 58.)

[92] M. Meyer, G. Debunne, M. Desbrun, and A. Barr. Interactive Animation of Cloth-like Objects in Virtual Reality. *The Journal of Visualization and Computer Animation*, 12(1):1–12, 2001. (Cited on pages 40 and 64.)

[93] J. Mezger, S. Kimmerle, and O. Etzmuß. Hierarchical Techniques in Collision Detection for Cloth Animation. *Proc. WSCG*, 2003. (Cited on pages 23, 26 and 55.)

[94] M. R. Mine. Virtual Environment Interaction Techniques. Technical Report TR95-018, 1995. citeseer.ist.psu.edu/mine95virtual.html. (Cited on page 71.)

[95] N. Molino, Z. Bao, and R. Fedkiw. A Virtual Node Algorithm for Changing Mesh Topology During Simulation. In *Proc. SIGGRAPH*, 2004. (Cited on pages 7 and 11.)

[96] G. Müller, J. Meseth, M. Sattler, R. Sarlette, and R. Klein. Acquisition, Synthesis and Rendering of Bidirectional Texture Functions. In *Eurographics 2004, State of the Art Reports*, pages 69–94, 2004. (Cited on pages 58 and 59.)

[97] M. Müller, J. Dorsey, L. McMillan, R. Jagnow, and B. Cutler. Stable Real-Time Deformations. In *Proc. SIGGRAPH Symposium on Computer Animation*, 2002. (Cited on page 11.)

[98] M. Müller and M. Gross. Interactive Virtual Materials. In *Proc. Graphics Interface*, 2004. (Cited on pages 7 and 11.)

[99] Open Inventor Architecture Group. *Open Inventor C++ Reference Manual: The Official Reference Document for Open Systems.* Addison-Wesley, 1994. (Cited on page 66.)

[100] D. Pai. STRANDS: Interactive Simulaton of Thin Solids using Cosserat Models. *Proc. Eurograpics*, 2002. (Cited on page 64.)

[101] X. Provot. Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behavior. In *Proc. Graphics Interface*, 1995. (Cited on pages 3, 10, 12 and 50.)

[102] W. T. Reeves. Particle Systems — A Technique for Modeling a Class of Fuzzy Objects. In *Proc. SIGGRAPH*, 1983. (Cited on page 11.)

[103] S. Romero, L. F. Romero, and E. L. Zapata. Fast Cloth Simulation with Parallel Computers. In *Proc. Euro-Par*, 2000. (Cited on page 40.)

[104] B. O. Rothbaum, L. F. Hodges, and S. Smith. Virtual reality exposure therapy abbreviated treatment manual: Fear of flying application. *Cognitive and Behavioral Practice*, 6:234–244, 2000. (Cited on page 64.)

[105] E. Sachs, A. Roberts, and D. Stoops. 3-draw: A tool for designing 3D shapes. *IEEE Computer Graphics and Applications*, 11(6):18–26, Nov. 1991. (Cited on page 64.)

[106] M. Sattler, R. Sarlette, and R. Klein. Efficient and Realistic Visualization of Cloth. In *Proc. Eurographics Symposium on Rendering*, 2003. (Cited on page 58.)

[107] D. Schmalstieg, L. M. Encarnação, and Z. Szalavári. Using Transparent Props For Interaction With The Virtual Table. In *Proc. Symposium on Interactive 3D Graphics*, 1999. (Cited on page 66.)

[108] D. Schmalstieg, A. L. Fuhrmann, M. Gervautz, and Z. Szalavári. 'Studierstube' - An Environment for Collaboration in Augmented Reality'. In *Proc. Collaborative Virtual Environments*, 1996. (Cited on page 66.)

[109] J. R. Shewchuck. Triangle – A Two-Dimensional Quality Mesh Generator and Delaunay Triangulator. http://www-2.cs.cmu.edu/~quake/triangle.html. (Cited on page 56.)

[110] J. R. Shewchuck. An Introduction to the Conjugate Gradient Method Without the Agonizing Pain. Technical Report CMU-CS-TR-94-125, Carnegie Mellon University, 1994. http://www.cs.cmu.edu/ quake-papers/painless-conjugate-gradient.ps. (Cited on pages 5, 23 and 42.)

[111] M. Shinya and A. Fournier. Stochastic Motion: Motion under the Influence of Wind. *Proc. Eurographics*, 1992. (Cited on pages 11, 12 and 28.)

[112] J. G. Siek and A. Lumsdaine. The Matrix Template Library: A Unifying Framework for Numerical Linear Algebra. In *ECOOP Workshops*, pages 466–467, 1998. (Cited on page 55.)

[113] K. Sims. Particle animation and rendering using data parallel computation. In *Proc. SIGGRAPH*, 1990. (Cited on page 11.)

[114] J. Stam. A General Animation Framework for Gaseous Phenomena. ERCIM Research Report R047, 1997. (Cited on page 11.)

[115] J. Stam. Stable fluids. *Proc. SIGGRAPH*, 1999. (Cited on pages 12, 27, 30 and 33.)

[116] J. Stam. A simple fluid solver based on the FFT. *Journal of Graphics Tools: JGT*, 6(2):43–52, 2001. (Cited on pages 12, 27 and 30.)

[117] J. Stam. Real-time fluid dynamics for games. *Proc. Game Developer Conference*, 2003. (Cited on pages 12, 27 and 30.)

[118] J. Stam and E. Fiume. Turbulent Wind Fields for Gaseous Phenomena. *Proc. SIGGRAPH*, 1993. (Cited on page 11.)

[119] R. Stoakley, M. J. Conway, and R. Pausch. Virtual reality on a WIM: Interactive worlds in miniature. In *Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems*, pages 265–272, 1995. (Cited on page 64.)

[120] S. L. Stoev. *A Toolset for Visualization, Interaction, and Rendering in Virtual Environments.* PhD thesis, Universität Tübingen, 2001. (Cited on page 5.)

[121] S. L. Stoev, M. Feurer, M. Ruckaberle, and W. Straßer. Exploring the Past: a Toolset for Visualization of Historical Events in Virtual Environments. In *Proc. ACM Symposium on Virtual Reality Software and Technology*, 2001. (Cited on page 64.)

[122] Z. Szalavári and M. Gervautz. The personal interaction panel - a two-handed interface for augmented reality. *Proc. Eurographics*, 1997. (Cited on pages 64, 66 and 68.)

[123] T. Takahashi, H. Fujii, A. Kunimatsu, K. Hiwada, T. Saito, K. Tanaka, and H. Ueki. Realistic Animation of Fluid with Splash and Foam. *Proc. Eurographics*, 2003. (Cited on page 12.)

[124] TüTex web page. http://www.tuetex.de. (Cited on page 54.)

[125] J. Vince. *Virtual Reality Systems.* Addison-Wesley, 1995. (Cited on page 5.)

[126] Virtual Try-On project web page. http://www.virtualtryon.de. (Cited on pages 53 and 57.)

[127] P. Volino and N. Magnenat-Thalmann. Developing simulation techniques for an interactive clothing system. In *International Conference on Virtual Systems and Multimedia*, 1997. (Cited on pages 3, 10 and 64.)

[128] P. Volino and N. Magnenat-Thalmann. *Virtual Clothing.* Springer, 2000. (Cited on pages 3, 5, 10, 12, 14, 24, 39 and 62.)

[129] P. Volino and N. Magnenat-Thalmann. Comparing Efficiency of Integration Methods for Cloth Animation. In *Computer Graphics International Proceedings*, 2001. (Cited on pages 27 and 39.)

[130] D. W. Walker and J. J. Dongarra. MPI: a standard Message Passing Interface. *Supercomputer*, 12(1):56–68, 1996. (Cited on page 44.)

[131] C. Ware and S. Osborne. Exploration and Virtual Camera Control in Virtual Three Dimensional Environments. In *Proc. Interactive 3D Graphics*, 1990. (Cited on page 88.)

[132] X. Wei, Y. Zhao, Z. Fan, W. Li, S. Yoakum-Stover, and A. Kaufman. Blowing in the wind. In *Proc. SIGGRAPH Symposium on Computer Animation*, 2003. (Cited on page 12.)

[133] J. Wejchert and D. Haumann. Animation aerodynamics. *Computer Graphics (Proc. SIGGRAPH)*, 25(2):19–22, July 1991. (Cited on pages 11 and 32.)

[134] J. Wernecke. *The Inventor Mentor*. Addison-Wesley, 1999. (Cited on page 66.)

[135] L. Yaeger and C. Upson. Combining Physical and Visual Simulation. Creation of the Planet Jupiter for the Film 2010. In *Proc. SIGGRAPH Symposium on Computer Animation*, 1986. (Cited on page 12.)

[136] G. Zachmann and A. Rettig. Natural and robust interaction in virtual assembly simulation. In *8th ISPE International Conference on Concurrent Engineering: Research and Applications ISPE/CE2001*, 2001. (Cited on page 64.)

[137] F. Zara, F. Faure, and J.-M. Vincent. Physical Cloth Animation on a PC Cluster. In *Fourth Eurographics Workshop on Parallel Graphics and Visualisation*, 2002. (Cited on page 40.)

[138] F. Zara, F. Faure, and J.-M. Vincent. Parallel Simulation of Large Dynamic Systems on a PC Cluster: Application to Cloth Simulation. *International Journal of Computers and Applications*, 2004. (Cited on page 40.)

Lebens- und Bildungsgang

Name:	Michael Keckeisen
Familienstand:	verheiratet (seit 19. 01. 2001)

04. 05. 1973	geboren in Ravensburg
1980 - 1984	Grundschule Ravensburg-Oberzell
1984 - 1993	Gymnasium St. Konrad, Ravensburg, mit Abschluss Abitur
1993 - 1994	Zivildienst beim Rettungsdienst des Deutschen Roten Kreuz, Ravensburg
1994 - 2000	Studium der Mathematik mit Nebenfach Informatik an der Eberhard-Karls-Universität Tübingen
10/1998 - 04/1999	Auslandsstudium an der Sussex University, Brighton, Großbritannien, im Rahmen des Internationalen Studiengangs Mathematik (ISM)
1999	Studienarbeit (bei Prof. Dr. W. Küchlin) im Fach Informatik *”Solving Cyclotomic Polynomials by Radical Expressions”*
2000	Diplomarbeit (bei Prof. Dr. P. Schmid) im Fach Mathematik *”p-adische L-Reihen und Klassenzahlformeln”*
12/2000	Diplom in Mathematik, Nebenfach Informatik
seit 02/2001	Wissenschaftlicher Mitarbeiter an der Fakultät für Informatik, Universität Tübingen, Lehrstuhl für Graphisch-Interaktive Systeme WSI/GRIS (Prof. Dr.-Ing. Dr.-Ing. E.h. Wolfgang Straßer)