Part 1: State-of-the-Art in Virtual Clothing

Pascal Volino and Nadia Magnenat-Thalmann

MIRALab, University of Geneva - CH-1211 Geneva, Switzerland

Abstract

Virtual garment design and simulation involves a combination of a large range of techniques, involving mechanical simulation, collision detection, and user interface techniques for creating garments. Here, we perform an extensive review of the evolution of these techniques made in the last decade to bring virtual garments to the reach of computer applications not only aimed at graphics, but also at CAD techniques for the garment industry.

1. Introduction

The challenges of virtual garment simulation are numerous, and have attracted research efforts for more than a decade. First dedicated to the realistic simulation of the mechanical behavior of cloth, it soon evolved towards simulation of virtual garments on synthetic characters. While computer graphics gets the most obvious benefits from garment simulation on animated virtual characters, virtual prototyping of garment models is another major application field for the garment industry.

Virtual garment simulation is the result of a large combination of techniques that have also dramatically evolved during the last decade. Unlike the mechanical models used for existing mechanical engineering for simulating deformable structures, a lot of new challenges arise from the highly versatile nature of cloth. The central pillar of garment simulation obviously remains the development of efficient mechanical simulation models, which can accurately reproduce with the specific mechanical properties of cloth. However, cloth is by nature highly deformable and specific simulation problems arise from this fact. First, the mechanical representation should be accurate enough to deal with the nonlinearities and large deformations occurring at any place in the cloth, such as folds and wrinkles. Moreover, the garment cloth interacts strongly with the body that wears it, as well as with the other garments of the apparel. This requires advanced methods for efficiently detecting the geometrical contacts constraining the behavior of the cloth, and to integrate them in the mechanical model (collision detection and response). All these methods require advanced and complex computational methods where most important key issues remain computation speed and efficiency. For real-time applications however, only specific approximation and simplification methods allow the computation of garment animation, giving up some of the mechanical accuracy of the result in a result rather focused on visual realism.

Garment simulation, which started in the late eighties with very simple models such as Weil's approach [WEI 86], has taken much benefit from the increasing performance of computer hardware and tools as well as the development of specific simulation technologies which have nowadays lead to impressive applications not only in the field of simulation of virtual worlds, but also as design tools for the garment and fashion industry.

2. Initial Developments in Virtual Garment Simulation

In the field of computer graphics, the first applications for mechanical cloth simulation appeared in 1987 with the work of Terzopoulos et al [TER 87] [TER 88] in the form of a simulation system relying on the Lagrange equations of motion and elastic surface energy. Solutions were obtained through finite difference schemes on regular grids. This allowed simple scenes involving cloth to be simulated, such as the accurate simulation of a flag or the draping of a rectangular cloth. However, the first applications that really simulated garments started in 1990 (fig.1) with the considerations of many other technologies complementing cloth simulation [LAF 91] [CAR 92], such as body modelling and animation, and collision detection and response [YAN 93]. These applications innovated by providing the first virtual system allowing virtual garment patterns to be sewed together around a character.

Figure 1: *"FlashBack": Early virtual garments used context-dependent simulation of simplified cloth models.*

Since then, most developments were aimed at optimizing the accuracy and efficiency of the methods for simulating cloth accurately and efficiently, along the developments of actual applications and commercial products.

3. Mechanical Models

The accurate reproduction of the mechanical behaviour of cloth has always been a key issue for garment simulation. The mechanical behaviour of cloth is usually measured using standardized protocols, such as the Kawabata Evaluation System (KES), or the simpler FAST method, which are based on the experimental measurement of strain-stress curves for elongation, shearing and bending on normalized samples of fabric. Different representations of the cloth surface mechanics then allow the virtual reproduction of the behaviour of cloth.

Well known in mechanical engineering, the Finite Element method considers the cloth surface as being discretized in interpolation patches for a given order (bilinear, trilinear, quadrilinear), and an associated set of parameters (degrees of freedom) that give the actual shape to the interpolation surface over the element. From the mechanical properties of the material, the mechanical energy is computed from the deformation of the surface for given values of the interpolation parameters. An equation system based on the energy variation is then constructed with these degrees of freedom. Surface continuity between adjacent elements imposes additional constraint relationships. A large sparse linear system is built by assembling successively the contributions of all the elements of the surface, and then solved using optimized iterative techniques, such as the conjugate gradient method.

Finite elements have only had a marginal role in cloth simulation. The main attempts are described in [COL 91], [GAN 95], [EIS 96]. Most implementations focus on the accurate reproduction of mechanical properties of fabrics, but restrict the application field to the simulation of simple garment samples under elementary mechanical contexts, mostly because of the huge computational requirements of these models. Furthermore, accurate modelling of highly variable constraints (large nonlinear deformations, highly variable collisions) is difficult to integrate into the formalism of finite elements, and this sharply reduces the ability of the model to cope with the very complicated geometrical contexts which can arise in real-world garment simulation on virtual characters.

An easier and more pragmatic way to perform cloth simulation is the use of particle systems. Particle systems consider the cloth to be represented only by the set of vertices that constitute the polygonal mesh of the surface. These particles are moved through the action of forces that represent the mechanical behaviour of the cloth, which are computed from the geometric relationships between the particles that measure the deformation of the virtual cloth. Among the different variations of particle systems, the spring-mass scheme is the simplest and most widely used (fig.2). It considers the distance between neighbouring particle pairs as the only deformation measurement and interaction source representing the internal elasticity of the cloth.

Particle systems are among the simplest and most efficient ways to define rough models that compute highly deformable mechanical systems such as cloth with computation times small enough to integrate them into systems for simulating complete garments on virtual bodies. Among the main contributions on particle system models, early works considered simple viscoelastic models on regular grids with applications for draping problems with simple numerical integration schemes [SAK 91]. Accurate models started with Breen et al [BRE 94] on modelling the microstructure of cloth using parameters derived from KES behaviour curves and integration based on energy minimization. However, such accurate models required a lot of computation for solving problems that were restricted to draping. On the other hand, more recent models trade accuracy for speed, such as the grid model detailed by Provot et al [PRO 95] which additionally includes geometric constraints for limiting large deformation of cloth. Additional contributions from Eberhardt et al [EBE 96] with the simulation of KES parameters and comparison of the efficiency of several integration methods. Advanced surface representations were used in [DER 98], where the simulation model and collision detection takes advantage of the hierarchical structure of subdivision surfaces. Modelling animated garments on virtual characters is the specific aim of the work described by Volino et al [VOL 95] [VOL 97], which investigate improved spring-mass representations for better accuracy of surface elasticity modelling on irregular meshes.

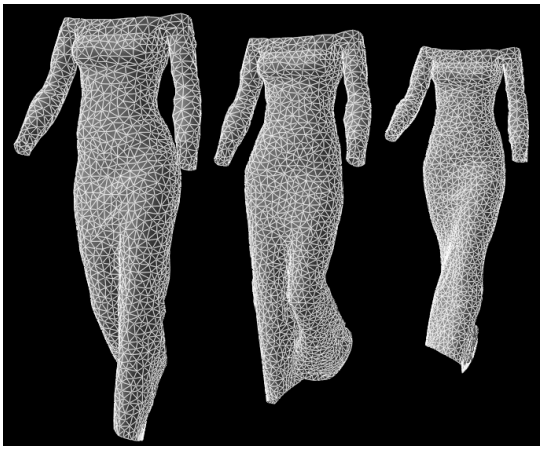


Figure 2: *Particle systems model cloth surfaces as point masses interacting with their neighbours using forces computed from their relative positions.*

While various models can be used to compute the force applied on each particle given their position and speed, these forces have then to be integrated along time to obtain the position and speed of the particle for the following time-steps using methods related to the integration of ordinary differential equation systems. Most recent however focus on improvements of the numerical integration methods in order to improve efficiency of the simulation.

Explicit integration methods are the simplest methods available for solving first-order ordinary differential systems. They consider the prediction of the future system state directly from the value of the derivatives. The best known techniques are the Runge-Kutta methods. Among them, the fast but unstable and inaccurate first-order Euler method, used in many early implementations, considers the future state as a direct extrapolation from the current state and the derivative. Higher order and more accurate methods also exist, such as the second-order Midpoint method, used for instance in early models by Volino et al [VOL 95], and the very accurate fourth-order Runge-Kutta method, used for instance by Eberhardt et al [EBE 96].

Beside considerations for accuracy, stability and robustness are other key factors to consider. For most situations encountered in cloth simulation, the numerical stiffness of the equations (stiff elastic forces, small surface elements) require the simulation time-steps to be small enough to ensure the stability of the system, and this limits the computation speed much more than accuracy

considerations. Adequate time-step control is therefore essential for an optimal simulation. A common solution is to use the fifth-order Runge-Kutta algorithm detailed in [PRE 92] which embeds integration error evaluation used for tuning the time-step adaptively [VOL 97].

In order to circumvent the problem of instability, implicit numerical methods are being used. For cloth simulation, this was first outlined by Baraff et al [BAR 98]. The most basic implementation of implicit method is the Euler step, which considers finding the future state for which "backward" Euler computation would return the initial state. It performs the computation not using the derivative at the current time-step, but using the predicted derivative for the next time-step. Besides the inverse Euler method, other, more accurate higher-order implicit methods exist, such as the inverse Midpoint method, which remains quite simple but exhibits some instability problems. A simple solution is to interpolate between the equations of the Euler and Midpoint methods, as proposed by Volino et al [VOL 00]. Higher-order methods, such as the Rosenbrook method, however do not exhibit convincing efficiencies in the field of cloth simulation. Multi-step methods, which perform a single-step iteration using a linear combination of several previous states, are other good candidates for a good accuracy-stability compromise. Among them, the second-order Backward Differential Formula (BDF-2) has shown some interesting performances, as used by Eberhardt, Hauth et al [EBE 00] [HAU 01] and Choi et al [CHO 02].

Whatever variation chosen, the major difficulty in using implicit integration methods is that they involve the resolution of a large and sparse linear equation system for each iteration, constructed from the Jacobian matrix of the particle forces against their position and speed. A commonly used simplification involves linearization of the mechanical model so as to obtain a linear approximation of the matrix that does not evolve along time, and on which initial construction and pre-processing allows efficient resolution method to be used, as for example like Kang et al [KAN 00], or even the matrix inverse to be pre-computed as done by Desbrun et al [DES 99]. A further simplification is to suppress completely the need of computing the matrix using an adapted approximation embedded directly in an explicit iteration. A big drawback of all these methods results from the approximation of the matrix that cannot take into account the nonlinearities of the model (mostly those resulting from the change of orientation of the surface elements during the simulation). While this is acceptable for draping applications, animations behave usually poorly because of excessive numerical damping, which also increases as the time-step becomes large.

The best numerical method for actually resolving the linear system seems to be the Conjugate Gradient method, as suggested by Baraff et al [BAR 98], with various variations and preconditioning schemes depending on how the mechanical model is formulated and geometrical constraints of the cloth integrated.

Most models using implicit integration schemes restrict themselves to using spring-mass systems, as their simple formulation eases the process of defining the linear system to be resolved. However, implicit integration methods can also be used for integrating accurate surface-based particle systems as the one described above, from derivation of the particle force expressions relatively to the particle positions and speeds. This in quite simply integrated into the implicit formulations described by Volino et al [VOL 00], and extended toward other advanced methods as by Hauth et al [HAU 01]. These formulations actually blur the line between particle systems and finite element methods, as the described particle system is indeed a first-order finite element method where the implicit resolution scheme corresponds to the energy minimization scheme of finite elements and the build of the linear system matrix to the assembly process of elements into the global system to be resolved. This is a key idea to design a new system which combines the accuracy of finite elements with the efficiency of the techniques used for particle system.

4. Real-Time Garment Animation

Real-time garment animation poses the challenging problem of how to perform very fast methods for the mechanical computation and collision detection. Accuracy has to be given up in favour of quicker methods that take advantage of geometrical approximations and contextual simplifications.

4.1. Physically-based Simulation

Implicit integration [BAR 98] [CHO 02] and speed-optimized derivatives [DES 99] [MEY 00] allow fast simulation of mechanical properties of cloth. However, the computation speed still remains slow for complex garments, and these methods are still limited by the maximum number of polygons they can animate in real-time.

In the specific of area optimizations for mechanical behaviour, James et al. [JAM 99] have worked on real time simulation. Their paper describes the boundary integral equation formulation of static linear elasticity as well as the related Boundary Element Method discretization technique. Their model is not dynamic, but rather a collection of static postures, limiting its potential applications. Debunne et al. [DEB 00] have also recently introduced a technique for animating soft bodies in real time. However, their method works on volumetric meshes and is therefore not applicable to thin objects such as cloth.

4.2. Collision detection

Collision detection is another bottleneck in the speed of cloth simulation. Besides the traditional methods, specific optimizations intend to address the problem of real-time simulation. For instance, Vassilev et al. [VAS 01] propose to use z-buffer for collision detection in order to generate depth and normal maps. The computation time of their collision detection does not depend on the complexity of the body. However, the maps need to be pre-computed before simulation, also restricting the real-time application.

4.3. Geometric approaches

Some other researchers have used geometrical approaches [WEI 86] [AGU 90] [HIN 90] [NGG 95]. Geometrical models do not consider the physical properties of the cloth, therefore providing techniques that produce fast results. However, these techniques are not able to reproduce the dynamics of clothes. Moreover, geometrical techniques require a considerable degree of user intervention. They can be regarded as a form of advanced drawing tools.

4.4. Hybrid Approaches

Hybrid approaches aim to combine nicely physically based deformation and geometric deformation. The idea of hybrid approach stems from the following observation: physically based simulations are slow to compute but produce realistic results while simulations based on geometric method are much faster but not really suitable to animate full clothes. By combining advantages of the both approaches, one can expect to have acceptable results within moderate

computation time. In most cases, physically based models are used to compute the global movements of garments and the details such as wrinkles are generated with geometric models.

Kang et al. [KAN 01] [KAN 02] improved the visual quality of the garments of small number of polygons by tessellating the triangles. With a cubic spline curve, their tessellation algorithm can simulate the wrinkles. Oshita et al. [OSH 01] use a similar approach to Kang et al. These both methods are mainly applicable to flat surfaces where physical simulation can be done with a relatively small number (<1,000) of polygons. However, highly curved surfaces, such as sleeves, need to be simulated with a higher number of polygons.

Recently, Hadap et al. [HAD 99] have proposed to simulate the cloth wrinkles with bump map. The global movements of the mesh are simulated with a particle system. The bump/displacement map is the product of a modulation map and a wrinkling pattern. The modulation map is generated by computing the local deformation of cloth triangles and the wrinkling pattern is designed by the CG artist.

4.5. Example-Based Methods

Example-based approaches, also known as data-driven deformations have become popular since several years. They have been successfully used for motion data and other graphical objects such as deformable human face and body models. The main idea of example-based is to construct a simulator that can learn deformations through a set of examples. This simulator can be considered as a black box computing for each input parameter (user interaction) the corresponding output (the shape of the deformable object). During the training phase, a set of meaningful input/output data is given to the simulator, which learns. During the runtime execution, the simulator upon receiving an input computes the corresponding output. If the given input data corresponds exactly to one of the samples, the simulator will return the output of the sample. Otherwise, the returned output will be interpolated using one of the interpolation techniques such as RBF, k-nearest neighbours, neural network, or linear interpolation, etc.

Grzeszczuk et al. [GRZ 98] have used a neural network to animate dynamic objects. They have shown that physically based models can be replaced by a large neural network that automatically learns to simulate similar motions by observing the models in action. The neural network is trained with a set of samples that have been computed with the physically based method. Not surprisingly, their simulator works in real-time. However, it has not been proven that the same method can be applied for complex simulation such as clothes.

Very recently, James et al. [JAM 03] have proven that the example-based approach can be successfully adapted to cloth simulation. The examples of cloth simulation they used had been calculated using commercial graphics package. Their approach makes use of Impulse Response Functions (IRF). Each IRF is indexed by the initial state x and two user-modeled parameter vectors, αI and αF, that describe the initial Impulse and persistent Forcing respectively.

5. Garment Design and Simulation

A Since the first developments to produce simulated garments on virtual characters [LAF 91] [CAR 92], cloth simulation and garment animation has made its way not only in computer research (fig.3) [VOL 97], but also into commercial products aimed both for 3D computer design and the garment industry.

Two kinds of products ace currently available: Those oriented for general cloth simulation and animation, and those specialized for draping and fitting garment models on virtual mannequins. The first category offer tools for simulating any kind of deformable surface mechanically. They usually offer a simple mechanical model containing only the basic mechanical parameters of cloth (stiffness, viscosity, bending, and gravity) modelled as a spring-mass particle system and simulated using state-of-the-art integration techniques. They allow the computation of realistic cloth animation, but do not provide any tool for designing garments. The also offer general collision detection schemes for interaction with any other objects. These tools are usually integrated as plug-ins into 3D design and animation frameworks. Among the main products, there is MayaCloth integrated into Maya [ALI 04], Reactor [DIS 04], Stitch [DIG 04], SimCloth [CHA 04] and ClothReyes [REY 04] for 3D Studio Max, Dynamics integrated into Cinema 4D [MAX 04].

Figure 3: *Virtual fashion: Real models and their virtual counterparts.*

The second category focuses on garment draping on virtual mannequins for visualization (virtual fashion, web applications) and prototyping purposes (garment design applications). The CAD applications specialize the simulation on pattern assembly and garment draping using accurate mechanical models of fabrics, while the visualization application take advantage of geometric techniques for generating quickly realistic dressed mannequins out of design choices. Both use pattern models imported from professional pattern design tools (Gerber, Lectra, Investronica). These tools also usually provide a standalone environment for setting up the simulation and visualizing the results. Among them, there is Toyobo

DressingSim [DRE 04], Browzwear V-Stitcher [BRO 04], or web applications such as FitMe.com [FIT 04] and MIRALab's Virtual Try-On [MIR 04].

Bibliography

[AGU 90] T. AGUI, Y. NAGAO, AND M. NAKAJMA, "An Expression Method of Cylindrical Cloth Objects-An Expression of Folds of a sleeve using Computer Graphics", Trans. of Soc. Of Electronics, Information and Communication, J73-D-II, 7, pp. 1095-1097, 1990.

[BAR 98] D. BARAFF, A. WITKIN, "Large Steps in Cloth Simulation", Computer Graphics (SIGGRAPH'98 proceedings), Addison-Wesley, 32, pp 106-117, 1998.

[BAR 03] D. BARAFF, A. WITKIN, M. KASS, "Untangling Cloth", Computer Graphics (SIGGRAPH'03 proceedings), Addison-Wesley, 2003 (to appear).

[BER 97] G. BERGEN, "Efficient Collision Detection of Complex Deformable Models Using AABB Trees", Journal of Graphics Tools, A K Peters Ltd, 2 (4), pp 1-14, 1997.

[BRE 94] D.E. BREEN, D.H. HOUSE, M.J. WOZNY, "Predicting the Drape of Woven Cloth Using Interacting Particles", Computer Graphics (SIGGRAPH'94 proceedings), Addison-Wesley, pp 365-372, July 1994.

[BRI 02] R. BRIDSON, R. FEDKIV, J. ANDERSON, "Robust Treatment of Collisions, Contact and Friction for Cloth Animation", Computer Graphics (SIGGRAPH'02 proceedings), Addison Wesley, 2002.

[CAR 92] M. CARIGNAN, Y. YANG, N. MAGNENAT-THALMANN, D. THALMANN, "Dressing Animated Synthetic Actors with Complex Deformable Clothes", Computer Graphics (SIGGRAPH'92 proceedings), Addison-Wesley, 26(2), pp 99-104, 1992.

[CUG 02] U. CUGINI, C. RIZZI, "3D Design and Simulation of Men Garments", WSCG Workshop Proceedings, 2002.

[CHO 02] K.J. CHOI, H.S. KO, "Stable but Responsive Cloth", Computer Graphics (SIGGRAPH'02 proceedings), Addison Wesley, 2002.

[COL 91] J.R. COLLIER, B.J. COLLIER, G. O'TOOLE, S.M. SARGAND, "Drape Prediction by means of Finite-Element Analysis", Journal of the Textile Institute, 82 (1), pp 96-107, 1991.

[COR 02] F. CORDIER, N. MAGNENAT-THALMANN, "Real-time Animation of Dressed Virtual Humans", Eurographics 2002 Proceedings, Blackwell Publishers, 2002.

[COR 03] F. CORDIER, H. SEO, N. MAGNENAT-THALMANN, "Made-to-Measure Technologies for Online Clothing Store", IEEE CG&A special issue on Web Graphics, IEEE Press, pp 38-48, 2003.

[DER 98] T. DEROSE, M. KASS, T. TRUONG, "Subdivision Surfaces in Character Animation", Computer Graphics (SIGGRAPH'98 proceedings), Addison-Wesley, 32, pp 148-157, 1998.

[DEB 00] G. DEBUNNE, M. DESBRUN, M.P. CANI, A. BARR, "Adaptive simulation of soft bodies in real-time", Computer Animation, Annual Conference Series, IEEE Press, 2000.

[DES 99] M. DESBRUN, P.SCHRÖDER, A. BARR, "Interactive Animation of Structured Deformable Objects", Proceedings of Graphics Interface, A K Peters, 1999.

[EBE 96] B. EBERHARDT, A. WEBER, W. STRASSER, "A Fast, Flexible, Particle-System Model for Cloth Draping", Computer Graphics in Textiles and Apparel (IEEE Computer Graphics and Applications), IEEE Press, pp 52-59, Sept. 1996.

[EBE 00] B. EBERHARDT, O. ETZMUSS, M. HAUTH, "Implicit-Explicit Schemes for Fast Animation with Particles Systems", Proceedings of the Eurographics workshop on Computer Animation and Simulation, Springer-Verlag, pp 137-151, 2000.

[EHM 01] S.A. EHMANN, M.C. LIN, "Accurate and Fast Proximity Queries Between Polyhedra using Convex Surface Decomposition", Computer Graphics Forum, Blackwell Ltd, 20, pp 500-510, 2001.

[EIS 96] J.W. EISCHEN, S. DENG, T.G. CLAPP, "Finite-Element Modeling and Control of Flexible Fabric Parts", Computer Graphics in Textiles and Apparel (IEEE Computer Graphics and Applications), IEEE Press, pp 71-80, Sept. 1996.

[FUH 03] A. Fuhrmann, C. Gross, V. Luckas, "Interactive Animation of Cloth Including Self-Collision Detection", Journal of WSCG, 11 (1) pp 141-148, 2003.

[FUI 03] A. FUHRMANN, C. GROSS, V. LUCKAS, A. WEBER, "Interaction-Free Dressing of Virtual Humans", Computer & Graphics, Pergamon Press, 27 (1) pp 71-82, 2003.

[HAU 01] M. HAUTH, O. ETZMUSS, "A High Performance Solver for the Animation of Deformable Objects using Advanced Numerical Metds", Eurographics 2001 proceedings, Blackwell, 2001.

[HIN 90] B.K. HIND, J. MCCARTNEY, "Interactive Garment Design", Visual Computer, Springler-Verlag, Vol. 6, pp. 53-61, 1990.

[HUB 96] P. HUBBARD, "Approximating Polyhedra with Spheres for Time-Critical Collision Detection", ACM Transactions on Graphics, 15 (3), pp. 179-210, 1996.

[GAN 95] L. GAN ET AL, "A Study of Fabric Deformation using Non-Linear Finite Elements", Textile Research Journal, 65(11), pp 660-668, 1995.

[GOT 96] S. GOTTSCHALK, M.C. LIN, D. MANOSHA, "OOBTree: A Hierarchical Structure for Rapid Interference Detection", SIGGRAPH 96 Conference Proceedings, pp. 171–180, Addison Wesley, 1996.

[GRZ 98] R. GRZESZCZUK, D. TERZOPOULOS, G. HINTON, "Neuroanimator: Fast neural network emulation and control of physics-based models", SIGGRAPH 98 Conference Proceedings, Annual Conference Series, pp. 9–20, Addison Wesley, 1998.

[HAD 99] S. HADAP, E. BANGARTER, P. VOLINO, N. MAGNENAT-THALMANN, "Animating Wrinkles on Clothes", IEEE Visualization '99. San Francisco, USA, published by IEEE Press, pp. 175-182, Oct. 1999.

[JAM 99] D. JAMES, D. PAI, "Accurate real-time deformable objects", SIGGRAPH 99 Conference Proceedings, Annual Conference Series, pp. 65–72, Addison Wesley, 1999.

[JAM 03] D. L. JAMES AND K. FATAHALIAN, "Precomputing Interactive Dynamic Deformable Scenes", ACM Transactions on Graphics (TOG), published by ACM SIGGRAPH Addison Wesley, Vol. 22(3), pp. 165-172, July 2003.

[KAN 00] Y.M. KANG, J.H. CHOI, H.G. CHO, D.H. LEE, C.J. PARK, "Real-Time Animation Technique for Flexible and Thin Objects", WSCG'2000 proceedings, pp 322-329, 2000.

[KAN 01] Y. M. KANG, J. H. CHOI, H. G. CHO, D. H. LEE, "An efficient animation of wrinkled cloth with approximate implicit integration", The Visual Computer Journal, Springer-Verlag, 2001.

[KAN 02] Y.M. KANG, H.G. CHO, "Bilayered Approximate Integration for Rapid and Plausible Animation of Virtual Cloth with Realistic Wrinkles", Computer Animation 2000 proceedings, IEEE Computer Society, pp 203-211, 2002.

[KLO 97] J.T. KLOSOWSKI, M. HELD, J.S.B. MITCHELL, "Efficient Collision Detection Using Bounding Volume Hierarchies of k-dops", IEEE transactions on Visualization and Computer Graphics, IEEE Press, 4 (1), 1997.

[LAF 91] B. LAFLEUR, N. MAGNENAT-THALMANN, D. THALMANN, "Cloth Animation with Self-Collision Detection", IFIP conference on Modeling in Computer Graphics proceedings, Springer-Verlag, pp 179-197, 1991.

[LAR 01] T. LARSSON, T. AKININE-MÖLLER, "Collision Detection for Continuously Deformable Bodies", Proceedings of Eurographics, Short Presentations, Blackwell, pp 325-333, 2001.

[MET 03] J. METZGER, S. KIMMERLE, O. ETZMUSS, "Hierarchical Techniques in Collision Detection for Cloth Animation", Journal of WSCG, 11 (2) pp 322-329, 2003.

[MEY 00] M. MEYER, G. DEBUNNE, M. DESBRUN, A. H. BARR, "Interactive Animation of Cloth-like Objects in Virtual Reality", Journal of Visualization and Computer Animation, John Wiley & Sons, 2000.

[NGG 95] H. NG, R.L. GRIMSDALE, "GEOFF-A Geometrical Editor for Fold Formation", Lecture Notes in Computer Science Vol. 1024: Image Analysis Applications and Computer Graphic, R. Chin, et al., eds., Springer-Verlag, pp. 124-131, 1995.

[OSH 01] M. OSHITA, A. MAKINOUCHI, "Real-time Cloth Simulation with Sparse Particles and Curved Faces", Computer Animation, IEEE Press, 2001.

[PRE 92] W.H. PRESS, W.T. VETTERLING, S.A. TEUKOLSKY, B.P. FLANNERY, "Numerical Recipes in C", Second edition, Cambridge University Press, 1992.

[PRO 95] X. PROVOT, "Deformation Constraints in a Mass-Spring Model to Describe Rigide Cloth Behavior", Graphics Interface'95 proceedings, A K Peters Ltd, pp 147-154, 1995.

[SAK 91] Y. SAKAGUSHI, M. MINOH, K. IKEDA, "A Dynamically Deformable Model of Dress", Trans. Society of Electronics, Information and Communications, pp 25-32, 1991.

[TER 87] D. TERZOPOULOS, J.C. PLATT, H. BARR, "Elastically Deformable Models", Computer Graphics (SIGGRAPH'97 proceedings), Addison-Wesley, 21, pp 205-214, 1987.

[TER 88] D. TERZOPOULOS, K. FLEISCHER, "Modeling Inelastic Deformation: Viscoelasticity, Plasticity, Fracture", Computer Graphics (SIGGRAPH'88 proceedings), Addison-Wesley, 22, pp 269-278, 1988.

[VAS 01] T. Vassilev, B. Spanlang, "Fast Cloth Animation on Walking Avatars", Eurographics proceedings, Blackwell Publishers, 2001.

[VOL 94] P. VOLINO, N. MAGNENAT-THALMANN, "Efficient Self-Collision Detection on Smoothly Discretised Surface Animation Using Geometrical Shape Regularity", Computer Graphics Forum (Eurographics'94 proceedings), Blackwell Publishers, 13(3), pp 155-166, 1994.

[VOL 95] P. VOLINO, M. COURCHESNE, N. MAGNENAT-THALMANN, "Versatile and Efficient Techniques for Simulating Cloth and Other Deformable Objects", Computer Graphics (SIGGRAPH'95 proceedings), Addison-Wesley, pp 137-144, 1995.

[VOL 97] P. VOLINO, N. MAGNENAT-THALMANN, "Developing Simulation Techniques for an Interactive Clothing System", Virtual Systems and Multimedia (VSMM'97 proceedings), IEEE Press, Geneva, Switzerland, pp 109-118, 1997.

[VOL 99] P. VOLINO, N. MAGNENAT-THALMANN, "Fast Geometrical Wrinkles on Animated Surfaces", WSCG'99 Proceedings, IEEE Computer Society, 1999.

[VOL 00] P. VOLINO, N. MAGNENAT-THALMANN, "Implementing fast Cloth Simulation with Collision Response", Computer Graphics International Proceedings, IEEE Computer Society, pp 257-266, 2000.

[VOL 01] P. VOLINO, N. MAGNENAT-THALMANN, "Comparing Efficiency of Integration Methods for Cloth Simulation", Computer Graphics International Proceedings, 2001.

[WEI 86] J. WEIL, "The synthesis of Cloth Objects", Computer Graphics, SIGGRAPH 86 Conference Proceedings, Annual Conference Series, Vol. 20, pp. 49-54. Addison Wesley, 1986.

[YAN 93] Y. YANG, N.Magnenat-Thalmann, "An Improved Algorithm for Collision Detection in Cloth Animation with Human Body", Computer Graphics and Applications (Pacific Graphics'93 proceedings), World Scientific Publishing Co, 1, pp 237-251, 1993.

[ZAC 02] G. ZACHMANN, "Minimal Hierarchical Collision Detection", Proceedings of the ACM Symposium on Virtual Reality, pp 121-128, 2002.

[ALI 04] http://www.aliaswavefront.com/

[DIS 04] http://www.discreet.com/

[DIG 04] http://www.digimation.com/

[CHA 04] http://www.chaosgroup.com/

[REY 04] http://www.reyes-infografica.com/

[MAX 04] http://www.maxon.de/

[DRE 04] http://www.dressingsim.com/

[BRO 04] http://www.browzwear.com/

[FIT 04] http://www.fitme.com/

[MIR 04] http://www.miralab.ch/

Part 2: Techniques for Virtual Clothing

Pascal Volino and Nadia Magnenat-Thalmann

MIRALab, University of Geneva - CH-1211 Geneva, Switzerland

Abstract

Despite numerous methods available for cloth simulation, virtual garment prototyping has yet to find its way toward the garment industry, the main issues being simulation accuracy and the potentiality for reproducing the complex behavior of complex garment models. These goals can only be reached through an optimal combination of modeling techniques and numerical methods that combines high computation efficiency with the versatility required for simulating intricate garment designs. We here describe optimal choices illustrated by their integration in a design and simulation tool that allow interactive prototyping of garments along drape motion and comfortability tests on animated postures.

1. Introduction

Garment simulation for prototyping application still remains a challenge, because of the high complexity of the simulation context, which goes far beyond the current application fields of the available cloth simulation systems.

Facing the high requirements of garment designers toward not only realism but also quantitative mechanical accuracy on draping and motion, the complexity and diversity of real garment models makes it a real challenge to design a garment simulation system [12]. Among the main difficulties are:

* The size of the garments, which can be meter-long and yet need to be simulated at millimeter accuracy.

* The intricate and highly variable shape of the garments, which interacts through complex contact patterns with the body (which is itself a complex deformable entity), as well as other garments.

* The highly deformable nature of cloth, which translate very subtle mechanical variations into large draping and motion variations which modify completely the visual appearance of garment models.

* The highly intricate anisotropic and nonlinear mechanical behavior of garments, requiring accurate measurement, modeling, and complex numerical methods for their resolution.

Mechanical simulation of cloth is a topic that is currently well explored, and many systems are already able to capture and simulate the mechanical properties of cloth. Among the best known techniques, spring-mass-like particle systems [4] take a large share, because these methods are fast and versatile, but unfortunately quite inaccurate when it comes to modeling complex anisotropic and nonlinear mechanical behaviors. On the other side, accurate methods such as finite elements [6] need huge computational requirements, and are not suited for the simulation of complete garments.

Among existing tools aimed at garment prototyping, most of them are focused only on draping, and are not suited for delivering accurate simulations of garments on animated characters. Animation is however a key issue in garment prototyping, as the motion of garments accounts a lot for the final visual look-and-feel of a dressing style. This however requires the simulation to reach a higher level of accuracy, with simulation of viscous behavior of cloth materials along with numerical integration methods that preserve the actual motion of the cloth surfaces along time. While many cloth simulation applications are available as commercial products, some of them capable of dealing with complete garments, none offers the accuracy necessary for an actual garment prototyping application, and therefore none is currently used in the garment industry.

Figure 1: *Complex garments are still challenging to animate.*

We here intend to present how to fulfill the requirements of a garment designer, integrating state-of-the-art methods which combine both requirements of mechanical accuracy with the power and versatility required for simulating complete dressing styles on animated characters. Among the main features available for the garment designer are pattern design with interactive garment fitting evaluation,

high-quality animation previews on moving characters, along with the possibility of managing dressing styles composed by several complex multilayer garments with many different materials and seamings [16].

We are putting emphasis on the following aspects:

* In Section 2, we describe a general mechanical model for cloth, which combines the versatility of particle systems with the accuracy of surface-based models, able to simulate the complex anisotropic nonlinear viscoelastic behaviors required for the accurate reproduction of the behavior of cloth, not only for draping applications but also for dynamic simulation of garment motion on animated characters.

* In Section 3, we discuss the particular problem of numerical integration, putting particular emphasis on studying the suitability of the various techniques to dynamical simulation of moving cloth through evaluation of their numerical damping.

* Several additional techniques required for simulating complex garments are also discussed, such as collision detection and processing techniques in Section 4.

* Finally, we show in Section 5 the results and the potentialities of all these methods put together in a system which is actually used to design and simulate complex virtual garments models.

2. Simulating the Mechanics of Cloth

2.1. Mechanical Properties of Cloth

2.1.1. Description

The mechanical behavior of fabric is inherent in the nature and molecular structure of the fiber material constituting the cloth, and as well the way these fibers are arranged in the fabric structure. Fabric fibers can be organized in several ways in the cloth surface. The main structures are:

* **Woven Fabrics**: Threads are orthogonally aligned and interlaced alternately using different patterns (such as plain or twirl).

* **Knitted fabrics**: Threads are curled along a given pattern, and the curls are interlaced on successive rows.

* **Non-woven fabrics**: There are no threads, and the fibers are arranged in an unstructured way, such as paper fibers.

Woven fabrics are the most commonly used in garments. They are relatively stiff though thin, easy to produce, and may be used in a variety of ways in many kind of design. In contrast, knitted fabrics are loose and very elastic. They are usually employed in woolens or in underwear. This structure greatly influences the mechanical behavior of the fabric material, which is mainly determined by:

* **The nature of the fiber**: Wool, cotton, synthetic,...

* **The thread structure**: Diameter, internal fiber and yarn structure,...

* **The thread arrangement**: Woven or knitted, and particular pattern variation.

* **The pattern properties**: Tight or loose.

These properties are critical to the stiffness of the material, its ability to bend, and its visual appearance.

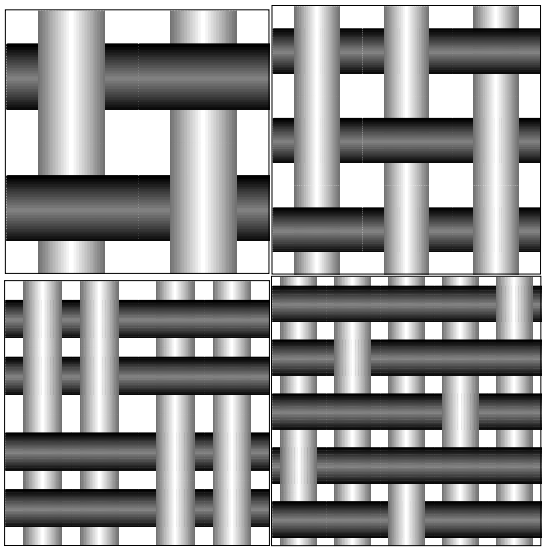


***Figure 2**: Different woven fabric patterns: Plain, Twirl, Basket, Satin.*

The mechanical properties of deformable surfaces can be grouped into four main families:

* **Elasticity**, which characterizes the internal forces resulting from a given geometrical deformation.

* **Viscosity**, which includes the internal forces resulting from a given deformation speed.

* **Plasticity**, which describes how the properties evolve according to the deformation history.

* **Resilience**, which defines the limits at which the structure will break.

Most important are the elastic properties that are the main contributor of mechanical effects in the usual contexts where cloth objects are used. Deformations are often small and slow enough to make the effect of viscosity, plasticity and resilience insignificant. One major hypothesis is that **quasistatic models** in the domain of elastic deformations will suffice for models intended to simulate the rest position of the garment on an immobile mannequin (draping). However, when a realistic animation is needed, the parameters relating energy dissipation through the evolution of the deformation are also needed, and complete **dynamic models** including viscosity and plasticity should be used.

Depending on the amplitude of the mechanical phenomena under study, the curves expressing mechanical properties exhibit shapes of varying complexity. If the amplitude is small enough, these shapes may be approximated by straight lines. This **linearity** hypothesis is a common way to simplify the characterization and modeling of mechanical phenomena.

It is common in elasticity theory to consider that the orientation of the material has no effect on its mechanical properties (**isotropy**). This however is inappropriate for cloth, as its properties depend considerably on their orientation relative to the fabric thread.

Elastic effects can be divided into several contributions:

* **Metric elasticity**, deformations along the surface plane.

* **Bending elasticity**, deformations orthogonally to the surface plane.

Metric elasticity is the most important and best studied aspect of fabric elasticity. It is usually described in terms of strain-stress relations. For linear elasticity, the main laws relating the strain e to the stress s involve three parameters, which are:

* **The Young modulus E**, summarizing the material's reaction along the deformation direction.

* **The Poisson coefficient ν**, characterizing the material's reaction orthogonal to the deformation direction.

* **The Rigidity modulus G**, pertaining to oblique reactions.

Along the two orthogonal directions **i** and **j**, these relations, named **Hook's Law**, **Poisson Law** and **Simple Shear Law** relating the stress **ε** to the strain **σ** are respectively expressed as follows:

$$\varepsilon_{ii} = \frac{1}{E_i}\sigma_{ii} \qquad \varepsilon_{jj} = \frac{\nu_{ij}}{E_i}\sigma_{ii} \qquad \varepsilon_{ij} = \frac{1}{G_{ij}}\sigma_{ij} \tag{1}$$

Cloth materials are two-dimensional surfaces for which two-dimensional variants of the elasticity laws are suitable. They are not isotropic, but the two orthogonal directions defined by the thread orientations can be considered as the main orientations for any deformation properties. In these **orthorombic** cloth surfaces, the two directions are called **weft** (**u**) and **warp** (**v**), and they have specific Young modulus and Poisson coefficients, $\mathbf{E_u}$, $\boldsymbol{\nu_u}$ and $\mathbf{E_v}$, $\boldsymbol{\nu_v}$ respectively. The elasticity law can be rewritten in terms of these directions as follows:

$$\begin{bmatrix}\sigma_{uu}\\ \sigma_{vv}\\ \sigma_{uv}\end{bmatrix} = \frac{1}{1-\nu_u\nu_v}\begin{bmatrix}E_u & \nu_v E_u & 0\\ \nu_u E_v & E_v & 0\\ 0 & 0 & G(1-\nu_u\nu_v)\end{bmatrix}\begin{bmatrix}\varepsilon_{uu}\\ \varepsilon_{vv}\\ \varepsilon_{uv}\end{bmatrix} \tag{2}$$

Energetic considerations imply the above matrix to be symmetric, and therefore the products $\mathbf{E_u}\,\boldsymbol{\nu_v}$ and $\mathbf{E_v}\,\boldsymbol{\nu_u}$ are equal. Considering isotropic materials, we also have the following relations:

$$E_u = E_v\nu_u = \nu_v G = \frac{E}{2(1-\nu)} \tag{3}$$

A similar formulation can be obtained for bending elasticity. However the equivalent of the Poisson coefficient for bending is usually taken as null. The relation between the curvature strain **τ** and stress **γ** is expressed using the flexion modulus **B** and the flexion rigidity **K** (often taken as null) as follows:

$$\begin{bmatrix}\tau_{uu}\\ \tau_{vv}\\ \tau_{uv}\end{bmatrix} = \begin{bmatrix}B_u & 0 & 0\\ 0 & B_v & 0\\ 0 & 0 & K\end{bmatrix}\begin{bmatrix}\gamma_{uu}\\ \gamma_{vv}\\ \gamma_{uv}\end{bmatrix} \tag{4}$$

While elasticity expresses the relation between the force and the deformation, viscosity expresses the relation between the force and the deformation speed in a very similar manner. To any of the elasticity parameters can be defined a corresponding viscosity parameter obtained by substitution of the stresses **ε** and **γ** by their derivatives along time **ε'** and **γ'**.

While the described linear laws are valid for small deformations of the cloth, large deformations usually enter the nonlinear behavior of cloth, where there is no more proportionality between strain and stress. This is practically observed by observing a "limit" in the cloth deformation as the forces increases, often preceding rupture (resilience), or remnant deformations observed as the constraints are released (plasticity). A common way to deal with such nonlinear models is to assume weft and warp deformation modes as still being independent, and replace each linear parameter $\mathbf{E_u}$, $\mathbf{E_v}$, **G**, $\mathbf{B_u}$, $\mathbf{B_v}$ by nonlinear strain-stress behavior curves.

2.1.2. Measuring the Mechanical Properties of Cloth

The garment industry needs the measurement of major fabric mechanical properties through normalized procedures that guarantee consistent information exchange between garment industry and cloth manufacturers. The **Kawabata Evaluation System for Fabric** (KES) is a reference methodology for the experimental observation of the elastic properties of the fabric material. Using five experiments, fifteen curves are obtained, which then allow the determination of twenty-one parameters for the fabric, among them all the linear elastic parameters described above, except for the Poisson coefficient.

Five standard tests are part of KES for determining the mechanical properties of cloth, using normalized measurement equipment. The **tensile test** measures the force/deformation curve of extension for a piece of fabric of normalized size along weft and warp directions and allows the measurement of $\mathbf{E_u}$ and $\mathbf{E_v}$ along with other parameters assessing nonlinearity and hysteresis. The **shearing test** is the same experiment using shear deformations, which allows the measurement of **G**. The **bending test** measures the curves for bending deformation in a similar way, and allows the measurement of $\mathbf{B_u}$ and $\mathbf{B_v}$. Finally, the **compression test** and the **friction test** allow the measurement of parameters related to the compressibility and the friction coefficients.

While the KES measurements allow determination of parameters assessing the nonlinearity of the behavior curves and some evaluation of the plasticity, other methodologies, such as the FAST method, use simpler procedures to determine the linear parameters only.

While the Kawabata measurements and similar systems summarize the basic mechanical behaviors of fabric material, the visual deformations of cloth, such as buckling and wrinkling, are a complex combination of these parameters with other subtle behaviors that cannot be characterized and measured directly.

In order to take these effects into account, other tests focus on more complex deformations. Among them, the draping test considers a cloth disk of given diameter draped onto a smaller horizontal disc surface. The edge of the cloth will fall around the support, and produce wrinkling. The wrinkle pattern can be measured (number and depth of the wrinkles) and used as a validation test for simulation models.

Tests have also been devised for measuring other complex deformation of fabric material, mostly related to bending, creasing and wrinkling.

2.2. Cloth Simulation Systems

Cloth being approximated as a thin surface, its mechanical behavior is decomposed in in-plane deformations (the 2D deformations along the cloth surface plane) and bending deformation (the 3D surface curvature).

The in-plane behavior of cloth is described by relationships relating, for any cloth element, the stress **σ** to the strain **ε** (for elasticity) and its speed **ε'** (for viscosity) according the laws of viscoelasticity. For cloth materials, strain and stress are described relatively to the weave directions weft and warp following three components: weft and warp elongation (**uu** and **vv**), and shear (**uv**). Thus, the general viscoelastic behavior of a cloth element is described by strain-stress relationships as follows:

$$\begin{aligned}
&\sigma_{uu}(\varepsilon_{uu},\varepsilon_{vv},\varepsilon_{uv},\varepsilon_{uu}',\varepsilon_{vv}',\varepsilon_{uv}')\\
&\sigma_{vv}(\varepsilon_{uu},\varepsilon_{vv},\varepsilon_{uv},\varepsilon_{uu}',\varepsilon_{vv}',\varepsilon_{uv}')\\
&\sigma_{uv}(\varepsilon_{uu},\varepsilon_{vv},\varepsilon_{uv},\varepsilon_{uu}',\varepsilon_{vv}',\varepsilon_{uv}')
\end{aligned} \qquad (5)$$

Assuming to deal with an orthotropic material (usually resulting from the symmetry of the cloth weave structure relatively to the weave directions), there is no dependency between the elongation components (**uu** and **vv**) and the shear component (**uv**). Assuming null Poisson coefficient as well (a rough approximation), all components are independent, and the fabric elasticity is simply described by three independent elastic strain-stress curves (weft, warp, shear), along with their possible viscosity counterparts.

In the same manner, viscoelastic strain-stress relationships relate the bending momentum to the surface curvature for weft, warp and shear. With the typical approximations used with cloth materials, the elastic laws are only two independent curves along weft and warp directions (shear is neglected), with their possible viscosity counterparts.

The issue is now to define a model for representing these mechanical properties on geometrical surfaces representing the cloth. These curved surfaces are typically represented by polygonal meshes, being either triangular or quadrangular, and regular or irregular.

Continuum mechanics are one of the schemes used for accurate representation of the cloth mechanics. Mechanical equations are expressed along the curved surface, and then discretized for their numerical resolution. Such accurate schemes are however slow and not sufficiently versatile for handling large deformations and complex geometrical constraints (collisions) properly. Finite Element methods express the mechanical equations according to the deformation state the surface within well-defined elements (usually triangular or quadrangular). Their resolution also involves large computational charges. Another option is to construct a model based on the interaction of neighboring discrete points of the surface. Such particle systems allow the implementation of simple and versatile models adapted for efficient computation of highly deformable objects such as cloth.

2.2.1. Spring-Mass Models

The simplest particle system one can think of is spring-mass systems. In this scheme, the only interactions are forces exerted between neighboring particle couples, similarly as if they were attached by springs (described by an force/elongation law along its direction, which is actually a rigidity coefficient and a rest length in the case of linear springs). Spring-mass schemes are very popular methods, as they allow simple implementation and fast simulation of cloth objects. There has also been recent interest in this method as it allows quite a simple computation of the Jacobian of the spring forces, which is needed for implementing semi-implicit integration methods (see Section 3).

The simplest approach is to construct the springs along the edges of a triangular mesh describing the surface. This however leads to a very inaccurate model that cannot model accurately the anisotropic strain-stress behavior of the cloth material, and also not the bending. More accurate models are constructed on regular square particle grids describing the surface. While elongation stiffness is modeled by springs along the edges of the grid, shear stiffness is modeled by diagonal springs and bending stiffness is modeled by leapfrog spring along the edges. This model is still fairly inaccurate because of the unavoidable cross-dependencies between the various deformation modes relatively to the corresponding springs. It is also inappropriate for nonlinear elastic models and large deformations. More accurate variations of the model consider angular springs rather than straight springs for representing shear and bending stiffness, but the simplicity of the original spring-mass scheme is then lost.

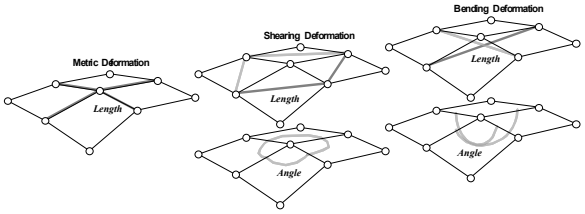


Figure 3: *Using length or angle springs for simulating cloth with a square particle system grid.*

2.2.2. An Accurate Particle System Model

Because of the real need of representing accurately the anisotropic nonlinear mechanical behavior of cloth in garment prototyping applications, spring-mass models are inadequate, and we need to find out a scheme that really simulates the viscoelastic behavior of actual surfaces. For this, we have defined a particle system model that relates this accurately over any arbitrary cloth triangle through simultaneous interaction between the three particles which are the triangle vertices. Such a model integrates directly and accurately the strain-stress model defined in Part 2.1 using polynomial spline approximations of the strain-stress curves, and remains accurate for large deformations.

In this model, a triangle element of cloth is described by 3 2D coordinates (**ua**, **va**), (**ub**, **vb**), (**uc**, **vc**) describing the location of its vertices **A**, **B**, **C** on the weft-warp coordinate system defined by the directions **U** and **V** with an arbitrary origin. They are orthonormal on the undeformed cloth (Fig.4). Out of them, a precomputation process evaluates the following values:

$$\begin{aligned}
Rua &= d^{-1}(vb-vc) & Rva &= -d^{-1}(ub-uc)\\
Rub &= -d^{-1}(va-vc) & Rvb &= d^{-1}(ua-uc)\\
Ruc &= d^{-1}(va-vb) & Rvc &= -d^{-1}(ua-ub)
\end{aligned}$$

$$\text{with}$$

$$d = ua(vb-vc)+ub(vc-va)+uc(va-vb) \qquad (6)$$

During the computation process, the current deformation state of the cloth triangle is evaluated using the current 3D direction and length of the deformed weft and warp direction vectors **U** and **V**. They are computed from the current positions **Pa**, **Pb**, **Pc** of its supporting vertices as follows:

$$U = Rua\ Pa + Rub\ Pb + Ruc\ Pc$$
$$V = Rva\ Pa + Rvb\ Pb + Rvc\ Pc \qquad (7)$$

The current in-plane strains **ε** of the cloth triangle is then computed with the following formula:

$$\varepsilon_{uu} = |U| - 1 \qquad \varepsilon_{vv} = |V| - 1$$
$$\varepsilon_{uv} = \frac{|U+V|}{\sqrt{2}} - \frac{|U-V|}{\sqrt{2}} \qquad (8)$$

We have chosen to replace the traditional shear deformation evaluation based on the angle measurement between the thread directions by an evaluation based on the length of the diagonal directions. The main advantage of this is a better accuracy for large deformations (the computation of the behavior of an isotropic material under large deformations remains more axis-independent).

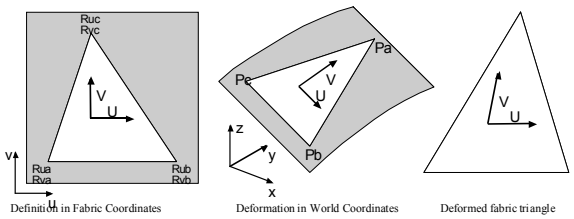


Figure 4: *A triangle of cloth element defined on the 2D cloth surface (left) is deformed in 3D space (center) and its deformation state is computed from the deformation of its weft-warp coordinate system (right).*

For applications that model internal in-plane viscosity of the material, the "evolution speeds" of the weave direction vectors are needed as well. They are computed from the current triangle vertex speeds **Pa'**, **Pb'**, **Pc'** as follows:

$$U' = Rua\ Pa' + Rub\ Pb' + Ruc\ Pc'$$
$$V' = Rva\ Pa' + Rvb\ Pb' + Rvc\ Pc' \qquad (9)$$

Then, the current in-plane strain speeds **ε'** of the triangle is computed:

$$\varepsilon_{uu}' = \frac{U \cdot U'}{|U|} \qquad \varepsilon_{vv}' = \frac{V \cdot V'}{|V|}$$
$$\varepsilon_{uv}' = \frac{(U+V)\cdot(U'+V')}{|U+V|\sqrt{2}} - \frac{(U-V)\cdot(U'-V')}{|U-V|\sqrt{2}} \qquad (10)$$

At this point, the in-plane mechanical behavior of the material can be expressed for computing the stresses **σ** out of the strains **ε** (elasticity) and the strain speeds **ε'** (viscosity) using the curves discussed in Part 2.2. Finally, the force contributions of the cloth triangle to its support vertices computed from the stresses **σ** as follows:

$$Fa = -\frac{d}{2}\left((Rua\ \sigma_{uu} + Rva\ \sigma_{uv})\frac{U}{|U|} + (Rua\ \sigma_{uv} + Rva\ \sigma_{vv})\frac{V}{|V|} \right) \qquad (11)$$
$$Fb = -\frac{d}{2}\left((Rub\ \sigma_{uu} + Rvb\ \sigma_{uv})\frac{U}{|U|} + (Rub\ \sigma_{uv} + Rvb\ \sigma_{vv})\frac{V}{|V|} \right)$$
$$Fc = -\frac{d}{2}\left((Ruc\ \sigma_{uu} + Rvc\ \sigma_{uv})\frac{U}{|U|} + (Ruc\ \sigma_{uv} + Rvc\ \sigma_{vv})\frac{V}{|V|} \right)$$

It is important to note that when using semi-implicit integration schemes (see Section 3), the contribution of these forces in the Jacobian **∂F/∂P** and **∂F/∂P'** can easily be computed out of the curve derivatives **∂σ/∂ε** and the orientation of the vectors **U** and **V**.

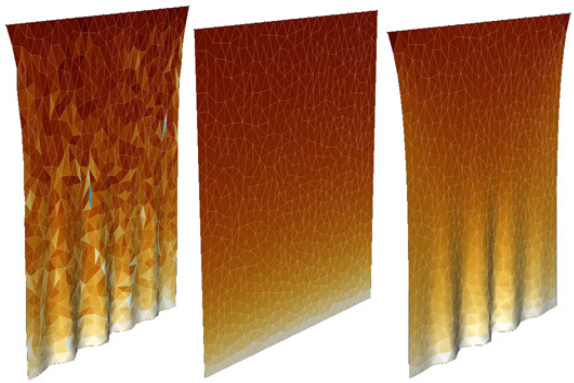


Figure 5: *Drape accuracy between a simple spring-mass system along the edges of the triangle mesh (left) and the proposed accurate particle system model (center). Color scale shows deformation. The spring-mass model exhibits inaccurate local deformations, along with an excessive "Poisson" behavior. This is not the case with the accurate model, which may still model the "Poisson" effect if needed (right, with a Poisson coefficient 0.5). The spring-mass model is also unable to simulate anisotropic or nonlinear models accurately.*

The model has been extended to handle anisotropic curvature stiffness through bending momentums applied along edges, for which the angle between the adjacent elements give an evaluation of the local surface curvature along the orthogonal direction. Additional mechanical features are integrated in the model as well for simulating the behavior of complex garment features (Fig.6).

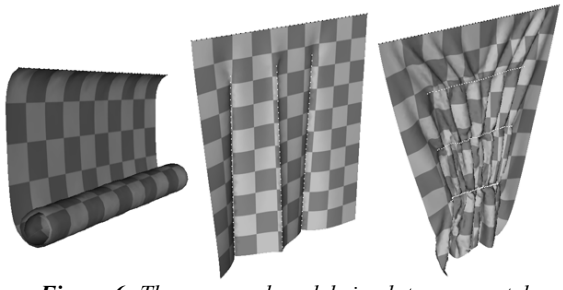


Figure 6: *The proposed model simulates accurately anisotropic bending stiffness, with possible rest curvature defined on the surface (left). Rest curvature may also be defined along precise lines (center) . Lines may also carry additional stiffness with their own custom rest length (right). All these features bring lot of potentialities for designing complex garment models.*

3. Numerical Integration

The equations resulting from the mechanical formulation of particle systems do usually express particle forces **F** depending on the state of the system (particle positions **P** and speeds **P'**). In turn, particle accelerations **P"** is related to particle forces **F** and masses **M** by Newton's 2nd law of dynamics. This leads to a second-order ordinary differential equation system, which is turned to first-order by concatenation of particle position **P** and speed **P'** into a state vector **Q**. A vast range of numerical methods has been studied for solving this kind of equations.

We have conducted extensive tests for benchmarking numerous integration methods, using performance, accuracy, stability and robustness as criteria. We have selected three candidates, each of which performs best in its own context:

* *1st-order semi-implicit Backward Euler*, which seems to be the best robust general-purpose method for any relaxation task (garment assembly and draping) [1] [14].
* *2nd-order semi-implicit Backward Differential Formula*, which offers increased dynamic accuracy along time (garment simulation on animated characters), at the expense of robustness (unsuited for draping during interactive design) [7].
* *5th-order explicit Runge-Kutta with timestep control*, which offers very high non-dissipative dynamic accuracy (accurate simulation of viscous and dissipative parameters in animated garments), at the expense of computation time (requires small time steps depending on the numerical stiffness, unsuited for stiff materials and refined discretizations) [4].

Our implementation integrates these three methods, and dynamically switches between them depending on the simulation context.

3.1. Discussing Integration Methods

3.1.1. Implicit Integration Methods

The most widely-used method for cloth simulation is currently the semi-implicit Backward Euler method, which was first used by Baraff et al [1] in the context of cloth simulation. As any implicit method, it alleviates the need of high accuracy for the simulation of stiff differential equations, offering convergence for large timesteps rather than numerical instability (a step of the semi-implicit Euler method with "infinite" timestep is actually equivalent to an iteration of the Newton resolution method) [15].

The formulation of a generalized implicit Euler integration is the following:

$$Q_{(t+dt)} - Q_{(t)} = Q'_{(t+\alpha\, dt)}\, dt \qquad (12)$$

The derivative value is not known at a moment after **t**, and is then extrapolated from the value at moment **t** using the Jacobian, leading to the semi-implicit expression which requires the resolution of a linear system:

$$Q_{(t+dt)} - Q_{(t)} = \left(I - \alpha \frac{\partial Q'}{\partial Q}_{(t)} dt \right)^{-1} Q'_{(t)}\, dt \qquad (13)$$

We have introduced the coefficient α so as to modulate the "implicitiсy" of the formula. Hence, $\alpha = 1$ is the regular implicit Backward Euler step (stable), whereas $\alpha = 0$ is the explicit Forward Euler step (unstable), and $\alpha = 1/2$ is the 2nd-order implicit Midpoint step (most accurate, at the threshold of stability).

The α parameter is a good handle for adjusting the compromise between stability and accuracy. While maximum robustness is obviously observed for large values, reducing its value increases accuracy (reduces numerical damping) at the expense of stability, and speeds up the computation as well (better conditioning of the linear system to be resolved).

Better accuracy can also be obtained through the use of the 2nd-order Backward Differential Formula (BDF-2), as described by Hauth et al [7]. This uses the previous state of the system for enhancing accuracy up to 2nd-order, with a minimal impact on the computation charge. Its generalized implicit expression is:

$$Q_{(t+dt)} - Q_{(t)} = \beta \left(Q_{(t)} - Q_{(t-dt)} \right) + Q'_{(t+\alpha\, dt)}\, \delta t$$
$$\text{with} \quad \beta = \frac{2\,\alpha - 1}{2\,\alpha + 1} \quad \text{and} \quad \delta t = \frac{2}{2\,\alpha + 1}\, dt \qquad (14)$$

And its semi-implicit expression is:

$$Q_{(t+dt)} - Q_{(t)} = \left(I - \alpha \frac{\partial Q'}{\partial Q}_{(t)} \delta t \right)^{-1} \left(\beta \left(Q_{(t)} - Q_{(t-dt)} \right) + Q'_{(t)}\, \delta t \right) \qquad (15)$$

While $\alpha = 1$ is the regular implicit BDF-2 step, $\alpha = 0$ is the explicit Leapfrog method, and $\alpha = 1/2$ is again the implicit Midpoint method. Best accuracy is offered for $\alpha = 1/\sqrt{3}$, where the method is 3rd-order (moderately stable).

Compared to Backward Euler, the main interest of the BDF-2 method is that it exhibits better accuracy for dynamic simulation over time (less numerical damping) for moderately stiff numerical contexts (at the expense of reduced robustness for nonlinear situations). For very stiff contexts however, this benefit disappears.

While it is possible to implement higher-order BDF methods, their interest is reduced by their lack of stability, and high accuracy could be more efficiently reached using high-order explicit methods. Stability of implicit methods is also affected by the nonlinearities of the mechanical model.

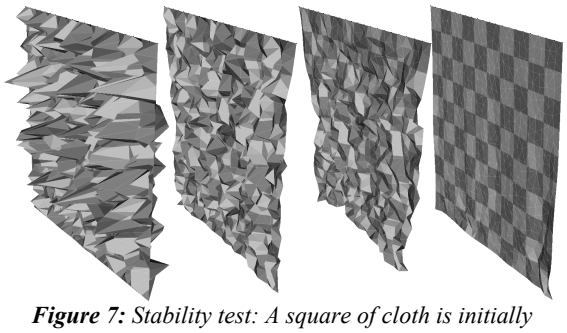


Figure 7: *Stability test: A square of cloth is initially deformed with large random perturbations, and then simulated using various timesteps.*

3.1.2. Explicit Integration Methods

Unlike implicit methods, explicit methods do not offer convergence to equilibrium if the timestep is too large compared to the numerical stiffness of the equations. On the other hand, they are very simple to implement, and much

compute much faster than their implicit counterpart for reaching a given accuracy. This is particularly true for high-order methods, which offer very high accuracy if the timestep is small enough, but diverge abruptly if it exceeds a threshold (related to the stiffness of the equations). This is why an efficient timestep control scheme is essential for the implementation of these methods.

While the explicit 1st-order Euler and 2nd-order Midpoint methods should be restricted to simple applications (beside their simplicity, they have no benefits compared to their implicit counterparts), a popular choice is the 5th-order Runge-Kutta scheme with embedded error evaluation [11]. It is a six-stage iteration process where the computed error magnitude can be used for controlling the adequate timestep very accurately, depending on accuracy and stability requirements. Unlike implicit methods, this method yields a very good guaranteed accuracy (resulting from the high-order, but which may require very small timesteps), which is particularly important for problems where energy conservation is a key issue (for example, evaluating the effect of viscous parameters in the motion of fabrics) (Fig.8). On the other hand, explicit methods are quite unsuited for the fast relaxation of the static cloth draping applications.

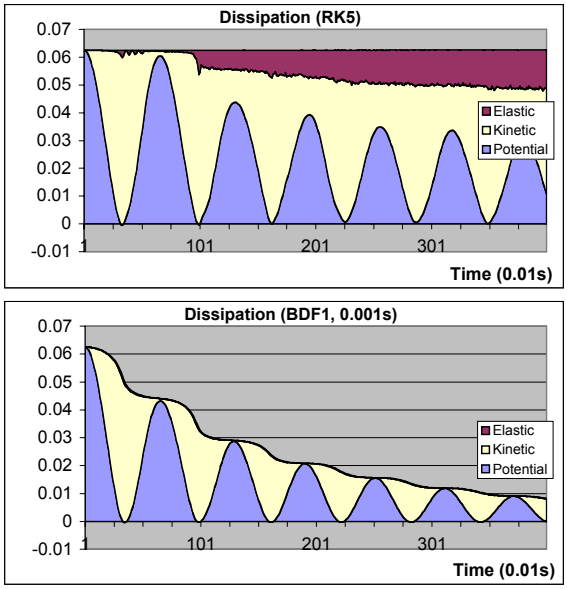


Figure 8: *Evaluation of numerical damping of various integration methods using energy dissipation plots along time(50cm x 50cm square cloth, initially horizontal, attached along one edge, linear isotropic 100N/m, 100g/m2, 2cm2 elements, no dissipative parameters). 5th-order Runge-Kutta (up) accurately preserves the total energy along time, a good amount of it being transferred to elastic energy through small-scale mesh jittering (timesteps between 0.0001s and 0.00001s). Implicit methods such as Inverse Euler (down) damp small-scale motion.*

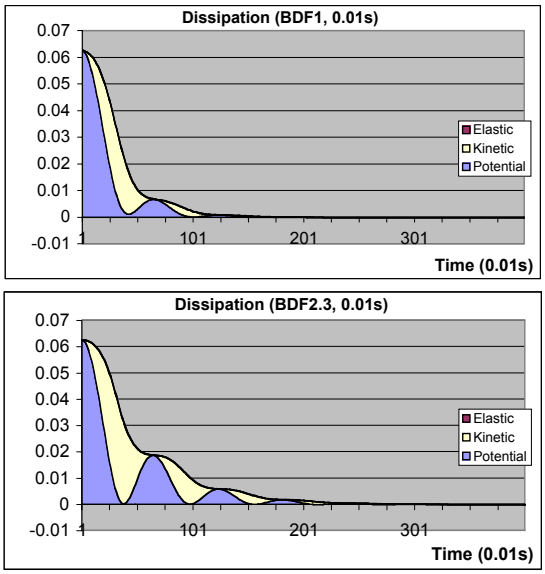


Figure 9: *The 3rd-order BDF2 variation (down) preserves energy significantly better than Inverse Euler (up).*

3.2. Implementation Issues

While there are no particular issues related to the implementation of explicit integration methods, semi-implicit methods require the resolution of large sparse linear equations systems, which are mainly constructed from the Jacobian of the mechanical law $\partial\mathbf{F}/\partial\mathbf{P}$ and $\partial\mathbf{F}/\partial\mathbf{P}'$ (their sparse structure relates the mechanical dependency between the particles). Among possible speed-up approximations, the Implicit-Explicit method described in [5] neglects the Jacobian terms generated by the non-stiff forces (which are then explicitly integrated).

A choice candidate for resolving this linear system is the Conjugate Gradient method, which is iterative and thus offers compromise between computation charge and symmetric accuracy, and which also allows efficient implementation for sparse systems.

Among possible optimizations are linearization schemes aimed at performing the computation using a constant approximation of the Jacobian, so as to implement preprocessing optimizations in the resolution. While giving reasonable benefits for draping applications, these approximations however generate large "numerical damping" that slow down convergence and alter highly the motion of the cloth along time [2] [3] [8] [10].

The only solution for simulating the accurate motion of cloth was indeed to use real value of the Jacobian corresponding to the current state of the system. We have taken advantage of the Conjugate Gradient method which only needs the Jacobian matrix products with given vectors to compute these products "on the fly" directly from the system state, skipping the sparse explicit storage of the matrix for each frame. Our system actually allows performing partial linearization of the Jacobian, so as to use the linearization ratio offering the best tradeoff between motion accuracy and stability, depending on the simulation context.

4. Collision Processing

Collision detection is indeed one of the most time-consuming tasks when it comes to simulate virtual characters wearing complete garments [9]. This task is performed through an adapted bounding-volume hierarchy algorithm, which uses a constant Discrete-Orientation-Polytope hierarchy constructed on the mesh, and optimization for self-collision detection using curvature evaluation on the surface hierarchy. This algorithm is fast enough for allowing full collision and self-collision detection between all objects of the scene with acceptable impact on the processing time (rarely exceeds 20% of the total time). Thus, body and cloth meshes are handled totally symetrically by the collision detection process, ensuring perfect versatility of the collision handling between the body and the several layers of garments [13].

Collision response is handled using a geometrical scheme based on correction of mesh position, speed and acceleration. This scheme ensures good accuracy and stability without the need of large nonlinear forces that alter the numerical resolution of the mechanical model. Our model simulates contact forces through a perfectly damped reaction model, associated to a Coulombian (solid) friction model.

The implemented collision model ensures full mesh-to-mesh collision response, which can deal with very complex multilayer collisions configurations involving several surfaces. The collision processing is therefore general enough for handling contacts between the several garments of a complex dress style, as well as the interactions between complex fold patterns when animating ample gestures. The model is also accurate enough for reproducing accurately friction behavior, allowing for example pants to hold to the waist with friction alone during character motion, without "cheating" using geometrical attachments. Good stability allows the simulation of complete multilayer garments with millimeter collision thickness despite large cloth speed and tension produced by complex character motion.

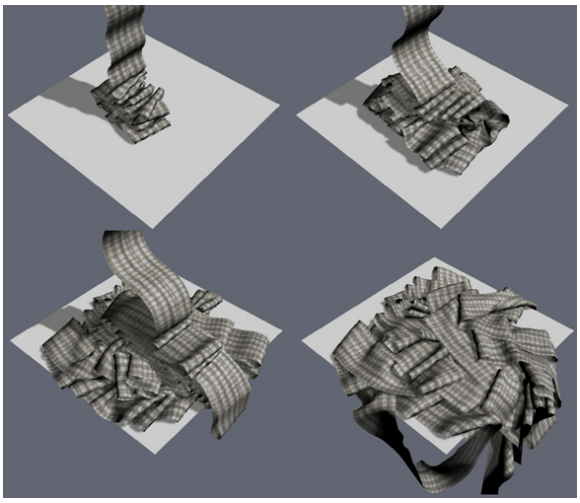


Figure 10: *Efficient and robust collision processing is important for simulating complex cloth.*

5. An Interactive System for Garment Prototyping

While essential, computational techniques alone are not sufficient for producing a powerful tool allowing accurate and convenient creation and prototyping of complex garments. We have integrated all these techniques into a garment design and simulation tool aimed at prototyping and virtual visualization, and allowing fashion designers to experiment virtually new collections with high-quality preview animations, as well as pattern makers to adjust precisely the shape and measurements of the patterns to fit the body optimally for best comfort.

The high level of interactivity required by these features necessitates simultaneous computation of the 3D garment updated immediately to each design modification done to the patterns. Our design and simulation tool provides a dual view of the garment, featuring both the 2D view of the pattern shapes cut on the fabric and the 3D view of the garment worn by a virtual character, with tight synchronization (Fig.11). Any editing task carried out in one view is directly displayed in the other view.

Figure 11: *Between real and virtual: Our system offers high-quality garment simulation, along with highly interactive pattern 2D-3D design and preview tools allowing complex garment models to be designed efficiently with many features such as seams, buttons, pockets, belts...*

5.1. Interactive Garment Editing Tools

The system features a fast Constrained Delaunay triangulation scheme that allows the discretization of complex patterns described as polygonal lines of control points (2D locations on the fabric). The system allows variable discretization densities over the mesh, as well as size anisotropy (elements elongated in a given direction), for representing adaptively complete garments from large surfaces to intricate details.

The interactivity of the system is based on two main features (Fig.12):

* *Mesh mapping update*: The 2D displacement of any control point of the pattern shape on the cloth surface immediately updates the mesh of that pattern on the cloth, while leaving the 3D drape position of the cloth constant. For obtaining this, each vertex of the mesh keeps track of a weighted sum of the pattern control points, which is computed during the triangulation process. This allows any measurement or shape editing to be directly taken into account by the mechanical simulation without any heavy recomputation, for immediate feedback of any pattern sizing adjustment.

* *Mesh topology reconstruction*: When the topology of the pattern mesh is changed (rediscretization, new features, etc), the 3D drape position of the new mesh is automatically recomputed from the drape position of the old one. During this process, advanced algorithms compute, for each mesh vertex of the new mesh, the location of the surface of the old mesh having identical 2D coordinates on the fabric. Extrapolation methods are used for computing the location of vertices which are located outside the old surface. This allows pattern design changes (new features, darts, seams...) to be added and modified without needing re-assembling and re-draping the garment on the virtual body.

5.2. Interactive Garment Prototyping

Put together, these techniques greatly enhance the workflow of garment prototyping. For instance, an initial garment could be quickly draped over a character using a rough mesh. Then, the designer could enhance the pattern shapes, while mesh mapping update automatically alter the mechanical state of the draped garment, changing the draping shape. Once the garment design is ready, a high-accuracy drape is automatically produced using topology reconstruction with a refined mesh.

Combined with the accuracy and speed of the proposed mechanical simulation engine, tasks such as comfortability evaluations are open to the garment designer, through the addition of several visualization tools, such as (Fig.13):

* Preview of fabric deformations and tensions along any weave orientation.

* Preview of pressure forces of the garment on the body skin.

* Immediate update of these evaluations according to pattern reshaping and sizing, fabric material change, and body measurements and posture change.

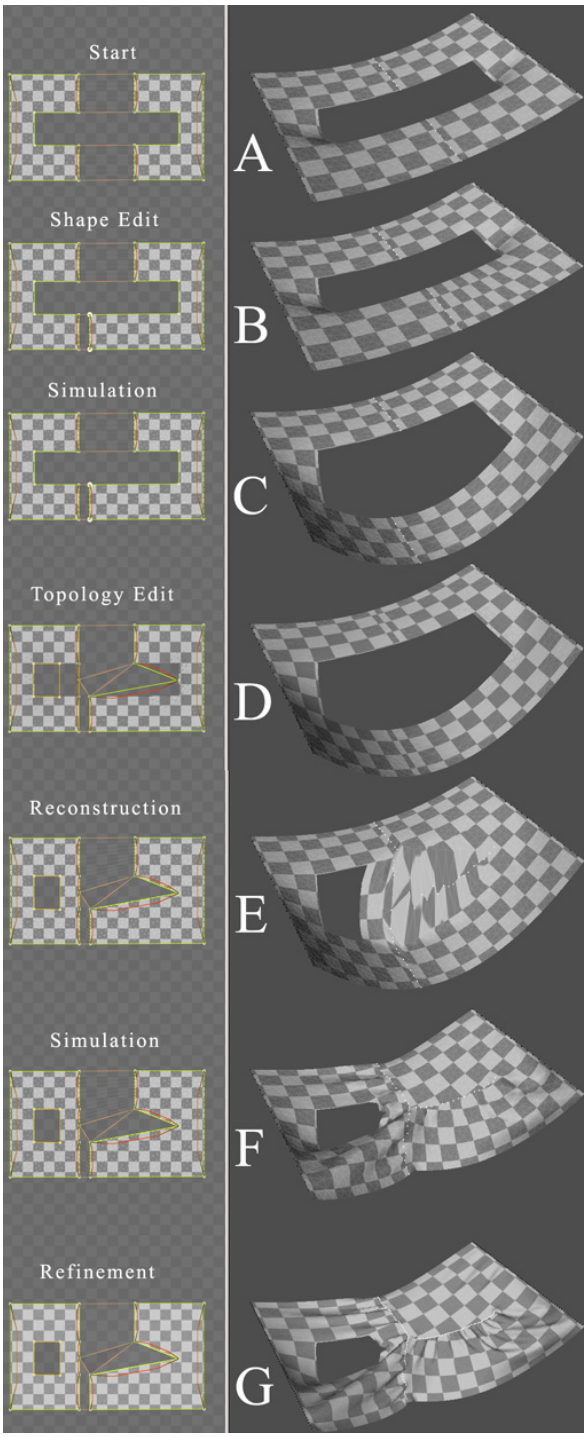


Figure 12: *Interactive garment design: From an initial draped pattern (A), editing the 2D contour of a pattern while keeping its 3D shape constant (B) allows mechanical simulation to smoothly drape the pattern to its new 3D shape (C). If pattern editing involves topology or seaming changes (D), the pattern surface is automatically reconstructed in the same position (E) using interpolation if necessary, and simulation performs the draping (F). The surface may be refined at any time (G) for more accuracy.*

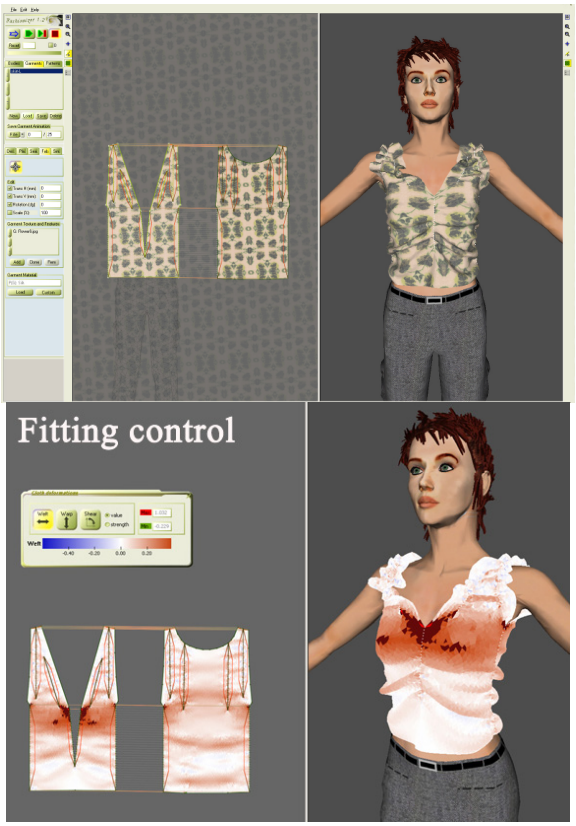


Figure 13: *Interactive garment prototyping and comfort evaluation: More than the garment shape and appearance design (left), the system also allows mechanical comfortability data to be evaluated directly with any change of the pattern design and sizing (right).*

Dynamic surface remeshing allows the best compromise between accuracy and computation speed to be selected adaptively according to the needs of the garment designer. For instance, while the garment assembly process can be carried out in a matter of seconds using an approximate mechanical model on a rough garment surface mesh, the garment designer may then switch to a more accurate model for tasks such as accurate draping and comfort evaluation. The model is still efficient enough to react interactively to design changes with garments made of ten thousands polygons, an accurate draping being obtained in a few minutes. Practical geometric accuracy is roughly limited by using 5 millimeters elements (Fig.14). Using time-accurate computation on animated characters, a high-quality catwalk is computed in a matter of a few hours.

5.3. Perspectives

Interactive design can already be used for creating fashion models that have enough realism for reproducing accurately the behavior of real garments. More than draping, our system is able to compute realistic animations thanks to good time-accurate numerical techniques applied to viscoelastic mechanical models of cloth.

The core technologies of this system are now being adapted to actual needs of the garment industry through collaborative projects, which deal on mechanical characterization of fabrics, virtual prototyping, manufacturing processes, e-commerce. Although some advances are still welcome in the area of efficiency and accuracy of mechanical simulation techniques, the challenge is now to create new tools that will ensure to the garment industry a smooth transition from tradition to novel possibilities offered by virtual simulation.

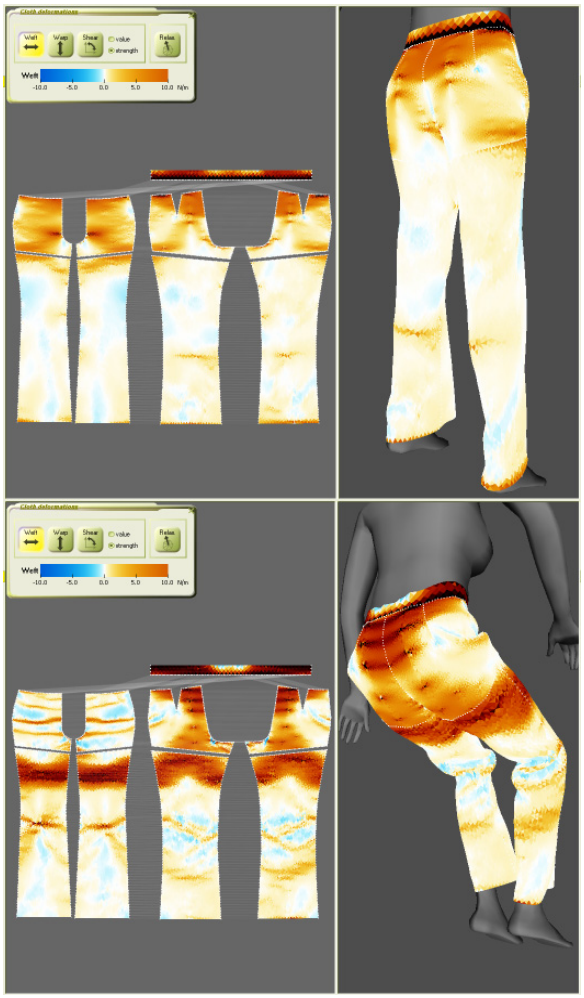


Figure 14: *Virtual prototyping: Displaying weft constraints on an animated body (from standing to sitting). Element size is 5mm.*

Bibliography

[1] D. BARAFF, A. WITKIN, Large Steps in Cloth Simulation, Computer Graphics (SIGGRAPH'98 proceedings), Addison-Wesley, 32, pp 106-117, 1998.

[2] K.J. CHOI, H.S. KO, Stable but Responsive Cloth, Computer Graphics (SIGGRAPH'02 proceedings), Addison Wesley, 2002.

[3] M. DESBRUN, P.SCHRÖDER, A. BARR, Interactive Animation of Structured Deformable Objects, Proceedings of Graphics Interface, A K Peters, 1999.

[4] B. EBERHARDT, A. WEBER, W. STRASSER, A Fast, Flexible, Particle-System Model for Cloth Draping, Computer Graphics in Textiles and Apparel (IEEE

Computer Graphics and Applications), IEEE Press, pp 52-59, Sept. 1996.

[5] B. EBERHARDT, O. ETZMUSS, M. HAUTH, Implicit-Explicit Schemes for Fast Animation with Particles Systems, Proceedings of the Eurographics workshop on Computer Animation and Simulation, Springer-Verlag, pp 137-151, 2000.

[6] J.W. EISCHEN, S. DENG, T.G. CLAPP, Finite-Element Modeling and Control of Flexible Fabric Parts, Computer Graphics in Textiles and Apparel (IEEE Computer Graphics and Applications), IEEE Press, pp 71-80, Sept. 1996.

[7] M. HAUTH, O. ETZMUSS, A High Performance Solver for the Animation of Deformable Objects using Advanced Numerical Methods, Eurographics 2001 proceedings, Blackwell, 2001.

[8] Y.M. KANG, H.G. CHO, Bilayered Approximate Integration for Rapid and Plausible Animation of Virtual Cloth with Realistic Wrinkles, Computer Animation 2000 proceedings, IEEE Computer Society, pp 203-211, 2002.

[9] J. METZGER, S. KIMMERLE, O. ETZMUSS, Hierarchical Techniques in Collision Detection for Cloth Animation, Journal of WSCG, 11 (2) pp 322-329, 2003.

[10] M. MEYER, G. DEBUNNE, M. DESBRUN, A. H. BARR, Interactive Animation of Cloth-like Objects in Virtual Reality, Journal of Visualization and Computer Animation, John Wiley & Sons, 2000.

[11] W.H. PRESS, W.T. VETTERLING, S.A. TEUKOLSKY, B.P. FLANNERY, Numerical Recipes in C, Second edition, Cambridge University Press, 1992.

[12] P. VOLINO, F,. CORDIER, N. MAGNENAT-THALMANN, From Early Virtual Garment Simulation to Interactive Fashion Design, Computer-Aided Design, 37, Elsevier, pp 793-608, 2005.

[13] P. VOLINO, N. MAGNENAT-THALMANN, Developing Simulation Techniques for an Interactive Clothing System, Virtual Systems and Multimedia (VSMM'97 proceedings), IEEE Press, Geneva, Switzerland, pp 109-118, 1997.

[14] P. VOLINO, N. MAGNENAT-THALMANN, Implementing fast Cloth Simulation with Collision Response, Computer Graphics International Proceedings, IEEE Computer Society, pp 257-266, 2000.

[15] P. VOLINO, N. MAGNENAT-THALMANN, Comparing Efficiency of Integration Methods for Cloth Simulation, Computer Graphics International Proceedings, IEEE Computer Society, 2001.

[16] P. VOLINO, N. MAGNENAT-THALMANN, Accurate Garment Prototyping and Simulation, Computer-Aided Design & Applications, 2(1-4), CAD Solutions, 2005.

Part3: Physical Models, Numerical Solvers for Cloth Animations and Virtual Cloth Design

Markus Wacker[1] Bernhard Thomaszewski[2] Michael Keckeisen[3]

[1]HTW Dresden, University of Applied Sciences Dresden, Germany

[2]WSI/GRIS, University of Tübingen, Germany

[3]TWT GmbH - Information & Engineering Technologies

Abstract

Physical models and numerical solvers are the main part of every physical cloth simulation. Application areas range from pure visual effects for film and entertainment industry to demanding virtual try-on scenarios, where in all cases a high degree of physical realism combined with fast computation times is needed. Therefore, research in cloth animation has focused on improving realism as well as computational speed, and significant advances have been made over the last years. In this part of the tutorial, we will discuss physical models for cloth ranging from simple mass-spring systems to continuum-based particle systems and fast finite element solutions. Then, we will describe several explicit and implicit time integration schemes and compare them with respect to stability and performance.

Finally, we will discuss interaction techniques to provide to the user a tailor and fitting room in order to manipulate garments in three-dimensional space.

1. Introduction and Overview

The area of physically based modelling is situated at the intersection of computer science, mathematics, physics, and material sciences. The animation of cloth is a particularly interesting application of physically based modelling, because it aims at fast animation solutions for rather difficult physical problems. Moreover, it addresses one of the major difficulties in creating realistic scenes with virtual actors.

The challenge of computer animation is to extract physical models for complex structures such as textiles, approximate them efficiently by mathematical algorithms, and run fast simulations with intelligent numerical methods. The range of methods proposed in literature is quite large. The techniques vary from simplified methods designed specifically for real-time applications to sophisticated methods that were designed to reproduce measured material properties. Due to improved algorithms and faster computers, it becomes possible more and more to use advanced physical models and still achieve fast animations.

In this part of the tutorial, we will first discuss several physical models that have been employed for cloth animation in the past, ranging from discrete mass-spring and particle systems to finite element solutions for continuous cloth models. Then, we will explain explicit and implicit numerical methods for the solution of the arising ordinary differential equations.

2. Physical Models

Models for the draping of cloth have been designed with different objectives. A common objective in computer graphics is to generate convincing and visually pleasing pictures and films. For that purpose, physics may be ignored or simplified to a certain extent. A different (engineering) objective is to preserve measured physical properties in order to map real materials onto a simulated cloth. This, for instance, is indispensable in commerce applications, in which a customer selects clothes based on a simulation and relies on the material properties for the fit. Additionally, in computer graphics, this also should lead to an animation that is fast and allows interaction with a complex scene.

$\mathbf{s}(u,v)$	deformed surface
$\mathbf{r}(u,v)$	(local, partial) rest state of surface
$\mathbf{d}(u,v)$	displacement
$\mathbf{x}_i$	particle positions
$\mathbf{v}_i$	particle velocity
ε	strain (tensor)
σ	stress (tensor)
C	elastic tensor
D	viscous tensor
$\langle \mathbf{a}, \mathbf{b} \rangle$	scalar product of vectors **a** and **b**
$\Delta \mathbf{s}$	Laplacian $\mathbf{s}_{xx} + \mathbf{s}_{yy} + \mathbf{s}_{zz}$
$\mathbf{s}_u$	partial derivative of **s** with respect to u

Table 1: *Notation in this section*

In the following, we will start with relatively simple mass-spring and particle systems. After that we will describe how discrete and continuous models aim at preserving real material properties.

2.1. Discrete Models

All models which we take into account have in common that they discretize the cloth by a polygonal mesh (figure 1). The vertices of this mesh are called particles or (mass) nodes. In discrete models, the mesh topology defines, how these particles interact and exert forces on one another.

Given the mesh describing the cloth, forces on each particle are computed depending on its own position and velocity, and the positions and velocities of a set of particles within its topological neighbourhood. When the force function F for each particle has been determined, Newton's equation of motion yields their respective movement. The trajectory of each particle with mass m_i at position $\mathbf{x}_i$ is computed by

$$F(x,v) = m_i \cdot \frac{d^2\mathbf{x}_i}{dt^2} \ . \tag{1}$$

Here x denotes the vector containing all particle positions and v the vector of all particle velocities. Note that since particle systems already represent a discretization in space, only a system of ordinary differential equations has to be solved. The systems presented in literature differ by their methods of computing the forces.

2.1.1. Mass-spring systems

In mass-spring systems, particle interaction is solely modelled by linear springs. Provot [43] proposes a mass-spring system for textiles and uses a rectangular mesh in which the particles are connected by structural springs to counteract tension, diagonal springs for shearing, and interleaving springs for bending as shown in figure 2. Forces by linear springs between two particles at $\mathbf{x}_i$ and $\mathbf{x}_j$ are given by

$$F^e_{ij}(x) = k_{ij}(\|\mathbf{x}_i - \mathbf{x}_j\| - l_{ij}) \frac{\mathbf{x}_i - \mathbf{x}_j}{\|\mathbf{x}_i - \mathbf{x}_j\|}, \tag{2}$$

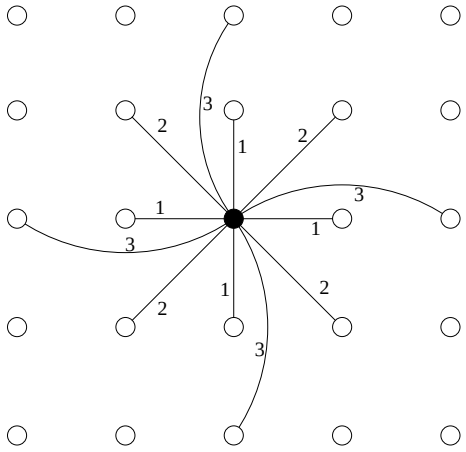


Figure 2: *Provot's mass-spring system with (1) structural springs, (2) shear springs, and (3) bending springs*

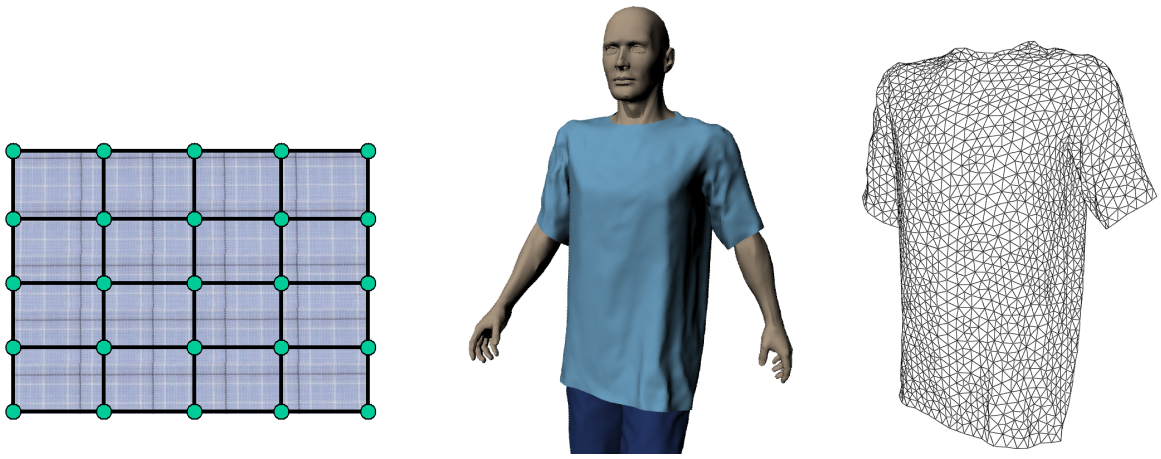
where k_{ij} is the elastic modulus of this spring and l_{ij} its rest length. The spring constant depends on the type of the spring. For the structural forces they are very large, whereas for the bend and shear forces the springs have small values. The different constants are related to the respective forces acting in real materials. Obviously, there is a strong interdependence between the different kinds of springs leading to nonlinear, uncontrolled effects. The diagonal shear springs, for instance, also lead to additional tension and transversal contraction.

Furthermore, in our model we need viscous forces to account for energy dissipation due to internal friction. These forces damp out kinetic energy and depend on the velocity of the object. It is a common technique to model these effects for each spring by

$$F^d_{ij}(x) = d_{ij}(\mathbf{v}_i - \mathbf{v}_j) \ . \tag{3}$$

Since these terms depend linearly on the involved velocities they are particularly well suited for the subsequent numerical integration. However, there are two major disadvantages of this simple formulation as it also penalises rigid body rotations of the respective particles. Moreover, high damping of a structural spring prevents the object from bending. Hence, this simplified damping makes the deformable object move rather stiffly. These effects are alleviated by modelling a stiff, damped spring accurately by

$$F^d_{ij} = d_{ij} \frac{\langle \mathbf{v}_i - \mathbf{v}_j, \mathbf{x}_i - \mathbf{x}_j \rangle}{\|\mathbf{x}_i - \mathbf{x}_j\|^2} (\mathbf{x}_i - \mathbf{x}_j) \ . \tag{4}$$



(a) (b) (c)

Figure 1: *Textiles are discretized by polygonal meshes. Image (a) shows a piece of cloth modelled as a quadrilateral mesh. The shirt in image (b) is represented by an unstructured triangle mesh (c).*

This is the same linear damping term (3) projected onto the direction of the spring. Unfortunately, in many cases this term complicates the implicit time integration (cf. section 3.1.3). Finally, in order to run the simulation, we only have to sum up all spring forces and plug them into equation (1).

In several approaches [43, 15, 34] for mass-spring systems another popular idea is exploited. It is motivated by a biphasic behaviour of textile materials as shown in figure 3 , i.e. initially the material yields to an exerted stress easily but appears to be extremely stiff in a second phase. This effect is imitated by rather small spring constants that model the first phase. In order to model the second, almost rigid phase, the system is post-processed after each time step if the springs are elongated too much. In this process, iteratively all particle positions are modified such that a certain maximum elongation is not exceeded. Such a post-processing is justified for simple mass-spring systems that do not model specific material properties anyway. Note that the result depends on the order in which the spring elongations are corrected.

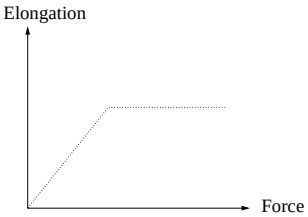


Figure 3: *Biphasic spring modelled by post-correction*

Recently, Choi and Ko [9, 8] have proposed another technique to model a biphasic behaviour of the structural springs. Only for elongation equation (2) is applied, for compression, the structural springs remain in rest length. To this end, the mesh particles have to be replaced out of the plane, resulting in a buckling behaviour of the mesh. This is done by a geometric condition which is integrated into the equation of motion. By this method wrinkles appear on the simulated piece of cloth when compressed like in real cloth.

Although simple mass-spring systems do not model any specific material and are not related to measured properties of real clothes, they are capable of producing pleasing animations that are sufficient in many computer graphics applications. Visually very convincing animations based on a particle system with a sophisticated approach to handle bending behavior of cloth were presented in [9].

2.1.2. Representations of Cloth as discrete Mechanism

In their book on cloth modelling [32], Donald House and David Breen state that *"Cloth is a mechanism, not a continuous material"*. Consequently, some attempts have been made to model clothes by the interaction of discrete threads that are interwoven in textiles.

Some discrete systems that have been developed in computer animation for the animation of clothes and other surfaces have the advantage that they allow fast simulations. In particular, particle systems have been successfully used for rapid animations. We can consider the quadrilateral mesh that is described by the mass nodes and structural springs in Provot's mass-spring system as a network of interwoven threads, in which one thread is given by a chain of structural springs. Different threads can interact at the mass points, where shear, bend, or other internal forces apply. In order to model the interaction of threads, more complex forces than

pure spring-forces are added to the system and yield a more general particle system.

Most particle systems use potential functions for tension, bend, and shear energy. These energies are chosen to correspond to standard experiments (Kawabata [35]) to measure textile properties. Hence, the measurements from one experiment are used to model one specific energy function. All energies are modelled on a rectangular grid, where each particle interacts with its four direct neighbours. The grid is aligned with two distinct directions that are apparent in textiles (in woven materials they are called weft and warp direction). The materials show different properties in these directions and each experiment has to be carried out for both directions.

The tension energy is evaluated for each particle and depends on the four neighbours of that particle in a rectangular mesh. The tension energy of a particle at position $\mathbf{x}_0$ is

$$E_t = \sum_{i=1}^{4} \begin{cases} \frac{1}{2} C_{t_1,i}(\|\mathbf{x}_0 - \mathbf{x}_i\| - l_i - h_{t_1,i})^3 & \text{if} \quad \|\mathbf{x}_0 - \mathbf{x}_i\| \geq l_i \,, \\ \frac{1}{2} C_{t_2,i}(\|\mathbf{x}_0 - \mathbf{x}_i\| - l_i - h_{t_2,i})^5 & \text{if} \quad \|\mathbf{x}_0 - \mathbf{x}_i\| \leq l_i \,, \end{cases} \tag{5}$$

where l_i are the rest lengths between particles and $C_{t,i}$ and $h_{t,i}$ are material parameters. They can be used to fit measured data by a piecewise linear curve. The energy is computed from a strain $(\|\mathbf{x}_0 - \mathbf{x}_i\| - l_i)$, and the strain-stress relation is modelled piecewise cubic or quintic. If we introduce a linear strain-stress relationship by replacing the exponents with 2 and set $h_{t,i} := 0$, we obtain linear spring energies related to the spring forces in equation (2).

The shear energy is modelled as

$$E_s = \sum_{i=1}^{4} \frac{1}{2} C_s(\phi_i - \frac{\pi}{2} - h_{s,i})^2 \tag{6}$$

and the bend energy as

$$E_b = \sum_{i=1}^{2} \frac{1}{2} C_b(\psi_i - \pi - h_{b,i})^2. \tag{7}$$

Here C_s, C_b and $h_{s,i}, h_{b,i}$ are the material constants. These energies implement hinges functioning like springs that linearly depend on the shear angle ϕ and the bend angle ψ, respectively. These are the angles formed by the incident edges as depicted in figure 4.

All derived energies are combined to compute the final forces to be plugged into equation (1):

$$F = -\text{grad}(E_t + E_s + E_b + E_{\text{external}}).$$

In this section, only elastic forces have been discussed. Viscous forces should be modelled in the fashion of section 2.2.7.

2.1.3. Triangular meshes

Until now, we only have considered particle systems based on rectangular meshes. Triangular mass-spring systems are

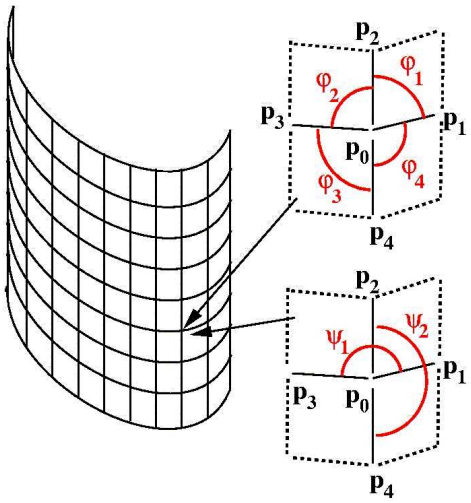


Figure 4: *Shear and bend energy in a particle system (image by Eberhardt [18])*

widely used as well and can be constructed with almost the same set of forces. However, their physical properties are hard to control and depend on the topology of the mesh. Furthermore, they usually show a very strong transversal contraction. This motivated Volino [49, 51] to extend the concept of triangular mass-spring systems. In a triangular mesh the deformation of each face is uniquely determined by the elongation of its edges. Forces acting on each of its particles can be formulated depending only on these (vectorial) elongations. This results in a particle system in which the forces on one particle do not only depend on adjacent edges but on the elongations of all edges of all faces incident to the considered particle. The coefficients of these dependencies are the material constants and allow a flexible modelling of the physical properties. This model was extended to modelling measured physical data in [53]. In a particle system model viscoelastic behaviour is relates to cloth triangles through simultaneous interaction between its three particles. The strain-stress relation is defined by polynomial spline approximations of the measured strain-stress curves.

The cloth simulation by Baraff and Witkin [3] also is based on a particle system for triangle meshes, however without the objective of fitting to real material data.

2.2. Continuous Models

Although clothes are not homogeneous, continuous objects, modelling them as discrete mechanism involves complications. As we cannot represent each single thread in a textile by an edge in the mesh, we have to choose a certain resolution of the object. If we want to be independent of this resolution, we need to represent a patch of textile as a continuous

material, which allows us to use low resolution models without loosing basic material properties.

From a continuum model a consistent discretization can be derived. Consistency here means that the computed solution of the discrete problem converges to the accurate solution for the continuum when the resolution is increased. That allows us to switch from one resolution to another without changing the properties of the cloth. Therefore we will describe the foundations of the continuous theory, and present a particle system that can approximate this theory. Moreover, we describe a linear finite element solution for cloth modeling which provides about the same performance as particle systems if implicit time integration methods (section 3.1.3) are employed.

2.2.1. Descriptions of Strain

Continuum mechanics is the standard theory to describe and model deformable objects, and the following elaborations are based on several text books [7, 10, 40, 46, 5].

The basic quantities of continuum mechanics are *strain*, which is a dimensionless deformation noted by ε, and *stress*, which is a force per length for surfaces or per area for volumes and is denoted by σ. In the case of a one-dimensional spring, these entities are scalars: the strain is its elongation per length, while the stress is the spring force. In the case of surfaces or volumes, these entities are tensors.

Surfaces are more complicated than a one-dimensional spring, and the description of strain is more involved. Textiles can be described as regular surfaces (in the sense of differential geometry [16]). The deformation of a regular surface embedded in $\mathbf{R}^3$ is described by a strain tensor with respect to a certain undeformed reference state. In this equilibrium state, denoted by $\mathbf{r}$, the object is not deformed, and the elastic energy is zero. Let $\mathbf{r}$ be parametrised over a domain $U \times V$. Under forces the rest state deforms to a state $\mathbf{s}(u,v)$. Its displacement is a mapping $\mathbf{d}$ defined by $\mathbf{d}(u,v) = \mathbf{s}(u,v) - \mathbf{r}(u,v)$ as depicted in figure 5.

The difference of the first fundamental forms I_s and I_r of the current state and the equilibrium state of the object describes the in-plane strain and defines a nonlinear strain tensor [39]

$$\tilde{G} = \tfrac{1}{2}(I_s - I_r) = \tag{8}$$
$$\tfrac{1}{2}\begin{pmatrix} \langle \mathbf{s}_u, \mathbf{s}_u\rangle & \langle \mathbf{s}_u, \mathbf{s}_v\rangle \\ \langle \mathbf{s}_u, \mathbf{s}_v\rangle & \langle \mathbf{s}_v, \mathbf{s}_v\rangle \end{pmatrix} - \tfrac{1}{2}\begin{pmatrix} \langle \mathbf{r}_u, \mathbf{r}_u\rangle & \langle \mathbf{r}_u, \mathbf{r}_v\rangle \\ \langle \mathbf{r}_u, \mathbf{r}_v\rangle & \langle \mathbf{r}_v, \mathbf{r}_v\rangle \end{pmatrix}.$$

For planar surfaces, the deformation is defined uniquely by the difference of the metrics of these states. As a piece of cloth is a surface embedded in three-dimensional space, the curvature tensors (second fundamental forms) have to be taken into account as well. Terzopoulos and Fleischer [48] developed a model for animated surfaces based on the energy due to these tensors.

Commonly, the rest state $\mathbf{r}$ is assumed to be the identity mapping. Then $\tilde{G}$ coincides with Green's strain tensor

$$G = \frac{1}{2}\begin{pmatrix} \langle \mathbf{s}_u, \mathbf{s}_u\rangle - 1 & \langle \mathbf{s}_u, \mathbf{s}_v\rangle \\ \langle \mathbf{s}_u, \mathbf{s}_v\rangle & \langle \mathbf{s}_v, \mathbf{s}_v\rangle - 1 \end{pmatrix}.$$

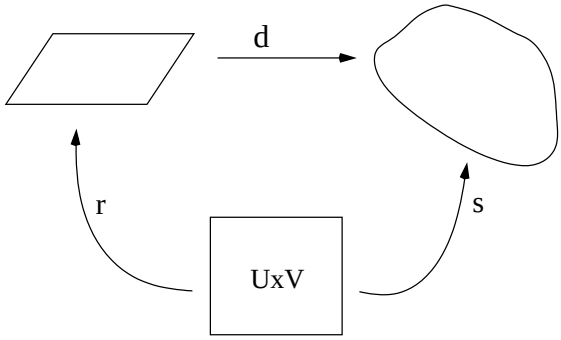


Figure 5: *The reference configuration: the rest state is parametrised by a mapping* $\mathbf{r}$ *on a space* $U \times V$*. By deformation* $\mathbf{d}$ *it transforms into the deformed (strained) configuration, which is parametrised by the mapping* $\mathbf{s}$*.*

Green's tensor, unfortunately, is nonlinear and yields fourth order terms in the energy formulation, which are computationally very costly and lead to various numerical problems. Linear elasticity theory would allow much faster animations. It makes use of the linear approximation of Green's tensor, called Cauchy's strain tensor. It is obtained by neglecting terms of order higher than one in the displacement $\mathbf{d}$ in the components of Green's tensor, here for two dimensions:

$$\begin{aligned} \langle \mathbf{s}_u, \mathbf{s}_u\rangle - 1 &= \langle \mathbf{e}_1 + \mathbf{d}_u, \mathbf{e}_1 + \mathbf{d}_u\rangle - 1 \\ &= 2\langle \mathbf{d}_u, \mathbf{e}_1\rangle + O(\mathbf{d}^2) \\ \langle \mathbf{s}_v, \mathbf{s}_v\rangle - 1 &= \langle \mathbf{e}_2 + \mathbf{d}_v, \mathbf{e}_2 + \mathbf{d}_v\rangle - 1 \\ &= 2\langle \mathbf{d}_v, \mathbf{e}_2\rangle + O(\mathbf{d}^2) \\ \langle \mathbf{s}_u, \mathbf{s}_v\rangle &= \langle \mathbf{e}_1 + \mathbf{d}_u, \mathbf{e}_2 + \mathbf{d}_v\rangle \\ &= \langle \mathbf{d}_u, \mathbf{e}_2\rangle + \langle \mathbf{d}_v, \mathbf{e}_1\rangle + O(\mathbf{d}^2)\,, \end{aligned} \tag{9}$$

where $(\mathbf{e}_1, \mathbf{e}_2)$ is the Cartesian basis of $\mathbf{R}^2$. Thus, Cauchy's tensor can be written as

$$\varepsilon = \begin{pmatrix} \mathbf{d}_u^1 & \frac{1}{2}(\mathbf{d}_v^1 + \mathbf{d}_u^2) \\ \frac{1}{2}(\mathbf{d}_v^1 + \mathbf{d}_u^2) & \mathbf{d}_v^2 \end{pmatrix},$$

where the superscripts denote the vector components. Linear elasticity is based on this linearised strain tensor and yields much simpler formulations. In particular, it results in linear partial differential equations. These linear equations are widely used in engineering and lend themselves to finite element formulations very easily.

The strain tensor of linear elasticity, as we have seen, is derived from Green's strain tensor by linearisations in the deformations, i.e. the displacement $\mathbf{d}$ (figure 5) is assumed

to be small, and all terms of order two or higher in **d** are neglected in the strain tensor. For this reason the linear theory is not appropriate for highly flexible objects like clothes, it only applies to small displacements. It is therefore not invariant under rigid body rotations and leads to unphysical behaviour if the object or a part of it is rotated. As animated surfaces can bend strongly, the displacements become very large, although the respective deformations remain small. Thus, a linearisation with respect to a global reference frame is not applicable to cloth animation. However, we will see later (in sections 2.2.5 and 2.2.6) that a linearisation with respect to a local reference frame is feasible.

2.2.2. The equation of motion

So far, we only have dealt with the description of strain. In linear elasticity, also the relation between the stress tensor σ and strain ε is assumed to be linear, and the dependence is given by the elastic tensor C. This is formulated by Hooke's law:

$$\sigma_{ij} = C_{ijkl}\,\varepsilon_{kl}\ . \tag{10}$$

C is a symmetric rank-4 tensor containing the material properties. Here, symmetry means $C_{ijkl} = C_{klij}$ as well as $C_{ijkl} = C_{jikl}$. Hooke's law is used to compute the stress tensor in the equation of motion of a continuous elastic material:

$$\rho\frac{\partial^2 s}{\partial t^2} - \operatorname{div}\sigma = f, \tag{11}$$

where ρ is the mass density and the divergence of the stress σ yields the force density due to the interior energy of the elastic object. f denotes an external force density (e.g. the gravity force density $\rho \cdot g$). Equation (11) is a partial differential equation (PDE) that has to be solved over time and the parameter domain of the surface. A standard procedure is to semidiscretize the system in space with finite differences or finite elements. This eliminates all spatial derivatives in the equation and reduces the PDE to an ordinary differential equation (ODE) in time t that can be solved by a suitable integration method. This way, we reduce the PDE (11) to the ODE of the form (1).

2.2.3. Bend forces

Bend forces cannot be derived from the standard strain tensor because they are out-of-plane forces and arise only due to a volumetric property of the (thin) object (an ideal surface does not exhibit bending forces). Hence, there are basically two possibilities of deriving bending forces. First, we can simply add some bending forces to the in-plane forces, derived for instance from a Laplacian formulation or from a thin plate energy. Second, the cloth can be modelled as a thin volumetric object. In this case, shell theory provides the appropriate mechanism. Shells are thin objects which span over surfaces. The shell is parameterised by a mid-surface **s**:

$$\mathbf{x}(u,v,w) = \mathbf{s}(u,v) + w\mathbf{d}(u,v),$$

i.e. the object position is described by the mid-surface parameters u,v and the height over the mid-surface w in the direction of a director **d** that has unit length.
From this description a three-dimensional strain and stress tensor is derived. Shells comprise (in-plane) membrane forces as well as bending forces. Some authors [19, 33, 23] model textiles and other thin deformable objects accurately by a nonlinear shell theory. In order to solve the resulting nonlinear PDE's, the system is discretized by finite elements, and Newton's method is used to compute an equilibrium solution.

2.2.4. Description of Clothes in a Linear Theory

A characteristic property of textiles is orthotropy. Linear, orthotropic materials possess two orthogonal symmetry axes in continuum mechanics. This reduces the number of free elastic material constants (entries in the elastic tensor) to four related to in-plane deformations. The symmetry axes or directions in a woven material are the weft and warp directions. The textiles show very different physical behaviour in these directions, and this characterises the materials. Therefore, material measurements are carried out for the two directions independently [35]. The two Young moduli $k_1 := C_{1111}$ and $k_2 := C_{2222}$, the shear modulus $\mu := C_{1212} = C_{2121} = C_{2112} = C_{1221}$, and the parameter $C_{1122} = C_{2211}$, which controls the transverse contraction, have to be taken into account as elastic constants to model the in-plane stress. Additionally, the bending moduli B_1 and B_2 describe the curvature elasticity in the weft and warp directions.

Unfortunately, the strain/stress relation in textile materials is highly nonlinear, and approximations to these nonlinear properties have been an issue in computer animation. Generally, the nonlinear stress-strain relation is modelled by a piecewise linear function. This can be achieved by updating the the elastic tensor according to the current slope of the strain/stress curve [18]. The updating is carried out before each time step. Another approach [53] consists in the approximation of the measured stress/strain by a polynomial spline already mentioned in section 2.1.3.

2.2.5. Continuous Theory and Particle Systems

As we have seen, Cauchy's stress tensor is not valid for large displacements. Green's strain tensor, however, impedes a rapid numerical solution and makes implicit time integration very difficult (see chapter 3).

In this section, we will show how the diagonal components of the strain tensor can be linearised locally to alleviate the solution of the ODE. From this model, we can derive a particle system that models continuous objects by a finite difference discretisation. This elastic model will be summarised here briefly. For details we refer to the full article [20].

Given the deformed surface $\mathbf{s}(u,v)$, we approximate the

stress tensor by

$$\sigma = \begin{pmatrix} k_1(\|\mathbf{s}_u\| - 1) & \frac{1}{2}\mu < \mathbf{s}_u, \mathbf{s}_v > \\ \frac{1}{2}\mu < \mathbf{s}_u, \mathbf{s}_v > & k_2(\|\mathbf{s}_v\| - 1) \end{pmatrix}, \qquad (12)$$

where k_1, k_2, and μ are the elastic material constants. This tensor is plugged into the equation of motion for continuous objects (11). In particular, in the strain tensor the dominating diagonal components have been linearised with respect to a local rest frame. Thus, the numerical solution is alleviated. From the diagonal components of the stress tensor we derive the following expression for the stress by finite difference discretization:

$$\sigma_{ii} = \frac{k_{1,2}}{l^2}(\|\mathbf{x}_i - \mathbf{x}_j\| - l),$$

where l is the rest distance between the particles at $\mathbf{x}_i$ and $\mathbf{x}_j$. Linear spring forces result from approximating the divergence of the diagonal entries in the stress tensor

$$f_{i,j} = \frac{k_{1,2}}{l^2}\left[(\mathbf{x}_i - \mathbf{x}_j) - l\frac{\mathbf{x}_i - \mathbf{x}_j}{\|\mathbf{x}_i - \mathbf{x}_j\|}\right], \qquad (13)$$

Note that the $f_{i,j}$ are actually force densities as we have continuous objects.

Additionally to the stretch forces, shear forces can be approximated similarly, and bend forces can be derived from a thin plate energy (see [20]).

For rectangular meshes it can be verified that this particle system approximates the accurate continuum model with linear precision, i.e. we have convergence to the accurate solution when the resolution is increased. For general quadrilateral meshes, the solution still converges, but does not possess the approximation property.

2.2.6. A Linearised Finite Element Model

The approach described in the last section works only for regular quadrilateral meshes. Therefore, a linearised finite element model for cloth which works for arbitrary unstructured triangle meshes has been developed [21]. 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 (cf. section 3.1.3).

For the moment we assume that all forces and displacements are in the (x,y)-plane, and that the small displacement assumption holds. For this case, we state the finite element formulation for linear elasticity as presented in several text books (e.g. [5]). Given the surface $\mathbf{s}(u_1, u_2)$, we denote the linear form functions of the linear finite element space by $N_j(u_1, u_2)$. For each triangle $T_m(a,b,c)$ we compute the form factors $\frac{\partial N_j}{\partial u_i}, j = a,b,c, i = 1,2$ in the rest state (these form factors replace the rest lengths of the springs in a mass-spring system). Also we compute 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 $K^m_{ab} \in \mathbb{R}^{2\times 2}$

$$\left[K^m_{ab}\right]_{ij} = \left(\sum_{k,l=1}^{2} \frac{\partial N_a}{\partial u_i} C_{ikjl} \frac{\partial N_b}{\partial u_j}\right)\Omega_{T_m}, \qquad (14)$$

where C is the elastic tensor that contains all elastic material properties.

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

$$[A]_{ab} = \begin{pmatrix} \left[\sum_{m|T_m\in\Gamma(a,b)} K^m_{ab}\right] & 0 \\ 0 \quad 0 & 0 \end{pmatrix} \in \mathbb{R}^{3\times 3}$$

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

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

where x is the vector of the vertex positions in the deformed state and $\bar{x}$ the vector of positions in the rest state. This force formulation is valid as long as the surfaces remain in the (x,y)-plane.

As mentioned previously, large rotations prevent the usage of the linear strain formulation. Hence, in each time step we are going to remove these rotations before computing the deformation forces (similar ideas have been used for three-dimensional deformable objects in [42] and [30]). This is equivalent to the construction of a rotated rest state and the linearisation with respect to this rest state. The key here is the polar decomposition of the deformation gradient. The deformation gradient J can be decomposed into a rotation R and a pure deformation U with

$$J = R \cdot U.$$

In the two-dimensional case the polar decompositioncan be calculated directly via an eigenanalysis with respect to J and is carried out as follows:

1. Compute $U^2 = J^t J$
2. Compute the eigenvalues λ_1, λ_2 and eigenvectors $\mathbf{v}_1, \mathbf{v}_2$ of U^2
3. Compute $U = \sqrt{\lambda_1}\mathbf{v}_1\mathbf{v}_1^t + \sqrt{\lambda_2}\mathbf{v}_2\mathbf{v}_2^t$
4. Compute $R = JU^{-1}$

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 a planar state in the (x,y)-plane, compute the elastic forces with linear elasticity there, and finally rotate the computed forces back in 3D space. For that purpose, we have to transform the displacement vectors before the multiplication with the local

stiffness matrix. The force vector that is the result of the multiplication has to undergo the inverse transformation. One local stiffness matrix for edge (a,b) computes forces as follows:

$$F(x) \;=\; R^m K^m_{ab} (R^m)^t \mathbf{x}_a - R^m K^m_{ab} \bar{\mathbf{x}}_a$$

Note that the planar rest state position $\bar{\mathbf{x}}_a$ 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

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

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

$$\begin{aligned} [A]_{ab} &= \sum_{m|T_m \in \Gamma(a,b)} Q^m_{ab} \,, \\ \left[\bar{A}\right]_{ab} &= \sum_{m|T_m \in \Gamma(a,b)} \bar{Q}^m_{ab} \,. \end{aligned} \tag{15}$$

The global force vector is computed by

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

Similarly, bend forces can be derived from a finite element formulation of the Laplacian operator projected onto the surface normals. Then, they can be evaluated as

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

Moreover, linear viscous forces can be derived by replacing the elastic tensor C by the viscosity tensor D (cf. equation (17)). These viscous forces result in a similar matrix formulation and are evaluated by

$$F(v) = Bv.$$

For a detailed discussion of bend forces and viscosity we refer to [21]. Summarizing all forces, Newton's equation of motion becomes

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

where M is the mass matrix, and f_{ext} are external forces. One time step in the animation is then computed as follows:

1. Compute the rotations R^m for each triangle T_m
2. Construct the matrices $A, \tilde{A}, B$ and compute $\bar{A}\bar{x}$
3. Carry out a time step.

The achievement of this procedure is that the forces are linear, such that a system of linear equations has to be solved in each time step, if an implicit time integration method is used. For the implicit Euler method (see section 3.1.3), for instance, we have to solve

$$\begin{aligned} v_1 &= v_0 + hM^{-1}(Ax_1 + \tilde{A}x_1 + Bv_1 - \bar{A}\bar{x} + f_{\text{ext}}) \\ x_1 &= x_0 + hv_1 \end{aligned}$$

with time step h. Note that the subscripts denote the time indices here. From this, we obtain the linear system

$$\begin{aligned} (M - h^2A - h^2\tilde{A} - hB)v_1 \;&= \\ Mv_0 \;+\; h(Ax_0 + \tilde{A}x_0 - \bar{A}\bar{x} + f_{\text{ext}}&). \end{aligned}$$

2.2.7. Viscosity and Hysteresis in Textiles

Regarding the nonlinear behaviour of cloth figure 6 shows a typical strain/stress curve that can be measured when a textile patch is stretched and relaxed. Hysteresis effects appearing as in this figure are due to energy dissipation. When the material deforms under external forces, the strain/stress relation is described by the upper branch of the hysteresis. When it is released and contracts again, this relationship is described by the lower branch and the area between both branches is proportional to the dissipated energy.

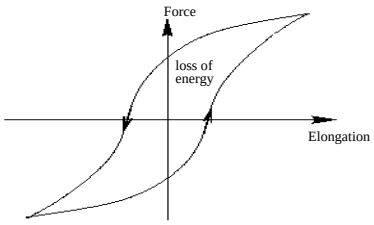


Figure 6: *Hysteresis*

There are several causes for energy dissipation when modelling deformable objects. Here, we focus exclusively on intrinsic effects, i.e. those that do not depend directly on external forces like friction due to fluid flow (wind). Internal friction depends on the relative motion of parts of the deformable object, that is, on the strain rate. The strain rate tensor is defined as

$$\nu_{ij} = \frac{d\varepsilon_{ij}}{dt} \,. \tag{16}$$

The forces generated by internal friction are often called viscous in analogy to Newtonian fluids. A viscous stress can be computed analogously to equation (10):

$$\sigma^v_{ij} = D_{ijkl}\, \nu_{kl}, \tag{17}$$

where the "viscosity tensor" D has the same structure as the elastic tensor C. The viscous stress is added to the purely elastic stress. In most implementations the viscosity tensor D is chosen constant and proportional to the elastic tensor C. This produces a material called a Kelvin-Voigt solid, the simplest viscoelastic solid. Note that the spring damping forces as stated in equation (4) can be derived from this model. In order to fit the typical behavior of cloth, a more

complex and visco-elastic model has to be chosen. For instance a constant Q model allows to reproduce a measured hysteresis curve [25].

3. Numerical Simulation

In this section, we will describe several explicit and implicit time integration methods, in parts following Hauth [29]. A further in-depth comparison of integration methods for cloth simulations has been presented by Volino and Magnenat-Thalmann [52, 53].

3.1. Methods for Numerical Integration

As we have seen in the previous section mechanical systems are often given as a second order ordinary differential equation accompanied by initial values

$$x''(t) = f_v\big(t, x(t), x'(t)\big)\ ,$$
$$x(t_0) = x_0, x'(t_0) = v_0\ . \tag{18}$$

The differential equation can be transformed into a first order system by introducing velocities as a separate variable:

$$\begin{bmatrix} x(t) \\ v(t) \end{bmatrix}' = \begin{bmatrix} v(t) \\ f_v\big(t, x(t), v(t)\big) \end{bmatrix}, \quad \begin{bmatrix} x(t_0) \\ v(t_0) \end{bmatrix} = \begin{bmatrix} x_0 \\ v_0 \end{bmatrix}. \tag{19}$$

For the next few sections it will be convenient to write this ODE in the more abstract form

$$y'(t) = f\big(t, y(t)\big), \quad y(t_0) = y_0, \tag{20}$$

before we will come back to the special setting (19) for eventually gaining computational advantages.

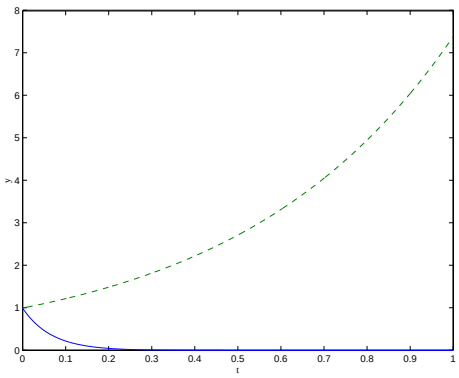
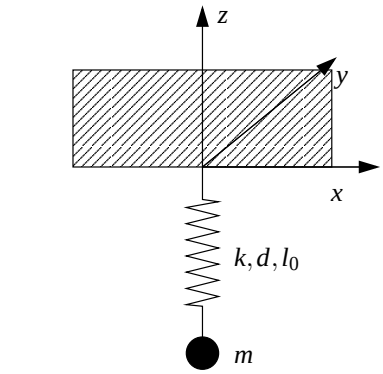


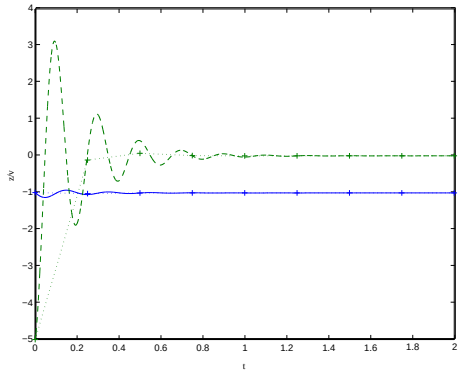
Figure 7: *Solutions of example 1 for $\lambda = 2$ (dashed) and $\lambda = -15$ (solid)*

Throughout the following discussion we will use the following examples:

1. $y' = \lambda y, y(0) = 1$ with $\lambda = 2, -15$ for $t \in [0,1]$ (figure 7).



(a) The mechanical system of example 3.



(b) Exact solution of example 3: z (solid), z' (dashed), and an implicit Euler solution (see below) for large timesteps.

Figure 8: *Example 3.*

2. The overdamped wave equation $y'' = \lambda/2y + \lambda y'$ with $\lambda = 5$ for $t \in [0,10]$ and starting values $y(0) = 0, y'(0) = 1$. It has the analytical solution $y(t) = 1/15\sqrt{15}e^{1/2\left(-5+\sqrt{15}\right)t} - 1/15\sqrt{15}e^{-1/2\left(5+\sqrt{15}\right)t}$
3. This example is based on a simple mechanical system (figure 8(a)): A particle p with mass m connected to the origin using a spring with stiffness k, damping d and rest length l_0, is pulled down by gravity. This setting is described by the ODE

$$\frac{d^2z}{dt^2} = \frac{k(l_0 - z)}{m} - \frac{d}{m}\frac{dz}{dt} + g_z\ . \tag{21}$$

We set the parameters $m = 0.1$, $k = 100$, $l_0 = -1$, $d = 1$, $g_z = -10$, $z_0 = -1$, $v_{0,z} = -5$ and simulate the interval $t \in [0,2]$ (figure 8(b)).

3.1.1. Explicit methods

The oldest and most simple method of integration is the so called forward or explicit Euler method. Time is discretised into steps of length h. To get a formula for advancing a timestep, h the differential quotient on the left hand side of (20) is replaced by the forward difference quotient

$$\frac{y(t+h)-y(t)}{h} \approx y'(t) = f(t,y(t)) \; . \tag{22}$$

Thus we get the integration formula for advancing a single timestep

$$y(t+h) = y(t) + hf(t,y(t)) \; . \tag{23}$$

Iterating this method gives a sequence of numerical approximations Y_n. Geometrically this method can be interpreted as straightly following the tangent of the solution at the actual state and then recalculating the slope for the next step.

There are several criteria for evaluating an integration method:

- convergence
- accuracy
- stability
- efficiency

Convergence means that for $h \to 0$ the numerical solutions Y_n meet the analytical. All useful methods must be convergent, so we won't discuss non-convergent methods or criteria for convergence. More interesting is the accuracy or order of a method. By this we mean how fast a method converges for $h \to 0$, or with other words how accurate the solution is for a given h. By using a taylor expansion for the exact solution after a single timestep

$$y(t+h) = y(t) + hy'(t) + h^2/2y''(t) + O(h^3) \tag{24}$$

we find that for the numerical approximation Y_1 produced by an explicit Euler step

$$y(t_1) - Y_1 = O(h^2) \; . \tag{25}$$

If we continue the method using the *numerical solution* Y_1 as a starting value for the next timestep we lose [26] a power of h for the global error

$$y(t_n) - Y_n = O(h) \; . \tag{26}$$

This means that the explicit Euler method converges linearly or has *order* 1. We will analyse stability and efficiency of the method later.

As a next step we will introduce methods of higher order. For this a centered difference estimation for $y'(t+h/2)$ in the differecial equation (20) is used leading to

$$\frac{y(t+h)-y(t)}{h} \approx y'(t+h/2) = f(t,y(t+h/2)) \tag{27}$$

and thus as an iteration scheme

$$Y_{n+1} = Y_n + f(t,y(t_n+h/2)) \; . \tag{28}$$

But how do we find $y(t_n+h/2)$? For an estimation we use an explicit Euler step to get

$$k_1 = \; Y_n + \tfrac{h}{2} f(t,Y_n) \; , \tag{29}$$

$$Y_{n+1} = \quad Y_n + hk_1 \; , \tag{30}$$

the so called *explicit midpoint* rule. The estimation by forward Euler, although not very accurate is good enough, as the function evaluation is multiplied by the timestep to advance to the next approximation. So by a taylor expansion we find a local error of $O(h^3)$ leading to a global error of

$$Y_n - y(t_n) = O(h^2) \tag{31}$$

for the explicit midpoint rule.

Generalizing the idea of using function evaluations at s intermediate points $t+c_jh$ leads to the Runge-Kutta methods defined by a Runge-Kutta matrix (a_{ij}), weights b_i, abscissae c_j and the equations

$$\begin{aligned} k_i &= Y_n + h\sum_{j=1}^{s} a_{ij}k'_j \\ &\text{with } k'_i = f(t_n + c_ih, k_i) \text{ for } \quad i = 1,\ldots,s \\ Y_{n+1} &= Y_n + h\sum_{i=1}^{s} b_ik'_i \; . \end{aligned} \tag{32}$$

The coefficient set can comfortably be specified like in table 2. If the matrix (a_{ij}) is strictly upper, all inner stages k_i only

c_1	a_{11}	a_{12}	$\cdots$	a_{1s}
c_2	a_{12}	a_{22}	$\cdots$	a_{2s}
$\vdots$	$\vdots$		$\ddots$	$\vdots$
c_s	a_{s1}	a_{s2}	$\cdots$	a_{ss}
	b_1	b_2	$\cdots$	b_s

Table 2: *General Runge–Kutta method*

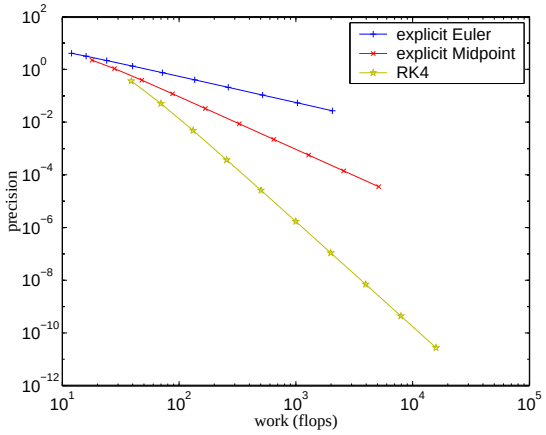
depend on k_j with $j < i$ and thus can be computed one after the other.

The most famous one is given in table (3) together with the explicit midpoint rule interpreted as a Runge-Kutta method. This special method by Runge and Kutta possesses order 4.

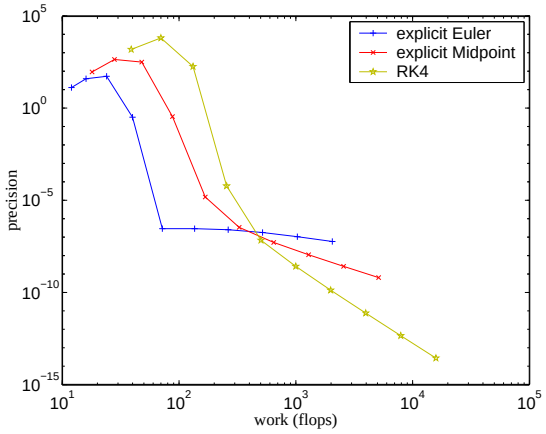
0	0	0
$\frac{1}{2}$	$\frac{1}{2}$	0
	0	1

0				
1/2	1/2			
1/2	0	1/2		
1	0	0	1	
	1/6	2/6	2/6	1/6

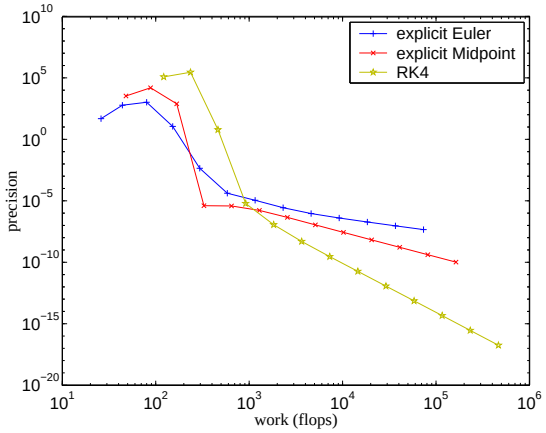
Table 3: *Explicit midpoint and "the" Runge-Kutta method*



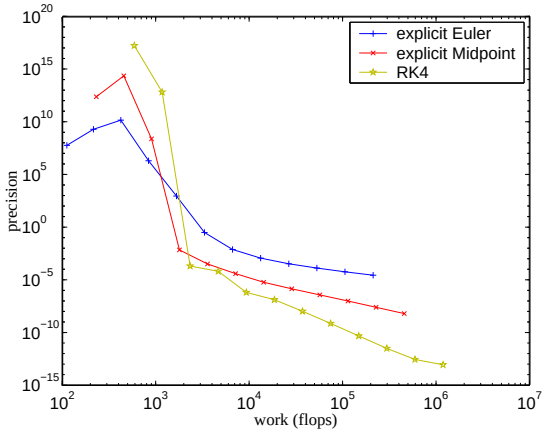
(a) Example 1a: $y' = 2y$



(b) Example 1b: $y' = -15y$



(c) Example 2: $y'' = -5y' - 5y$



(d) Example 3

Figure 9: *Work precision diagrams for the explicit Euler, explicit midpoint and RK4 methods.*

By using algebraic relations for the coefficients, it is possible to construct explicit Runge-Kutta methods of arbitrary high order resulting in many inner stages with many function evaluations. But for most practical applications order 4 is sufficient.

Having constructed all these methods, we now will try them on our examples. The diagrams in figure 9 were produced by solving the examples using different timesteps and measuring the number of floating point operations needed for achieving the specified accuracy when compared with the (analytical) reference solution. In the work-precision diagram the y-axis shows the error $\|Y_{t_{end}} - y(t_e nd)\|$ as a function of the required number of floating point operations. The first example with $\lambda = 2$ (figure 9(a)) shows exactly the expected behaviour: when reducing the time step and thus investing more work, the numerical solutions converge against the reference solution. Moreover the slope of the curves in the double logarithmic plot exactly match the order of the method. But in all the other examples (figure 9(b)-9(d)) this behaviour only shows up after an initial phase, where the solver produces completely wrong results. This is the point where stability comes in. We will now analyse this by using the simplest example where it occurs, i.e. example 1 with $\lambda < 0$.

3.1.2. Stability

The equation for example 1 is called *Dahlquist's test equation*

$$y' = \lambda y, \qquad \lambda \in \mathbb{C} \; . \tag{33}$$

Its exact solution to an initial value $y(0) = y_0$ is given by

$$y(t) = e^{\lambda t} y_0 \; . \tag{34}$$

This equation serves a tool for understanding and evaluating the stability of integration methods. We have seen, that in the damped case characterised by $\mathrm{Re}\,\lambda < 0$ convergence is only achieved by very small timesteps. In this case, since the exponent is negative, the analytical solution is bounded for $t \to \infty$. Therefore a meaningful numerical method is required to deliver a bounded solution. An integration scheme that yields a bounded solution is called *stable*.

If we apply the explicit Euler's method with a fixed step size h to (33) the n-th point of each solution is given by:

$$Y_n = (1 + h\lambda)^n y_0 \tag{35}$$

For the explicit Euler scheme the numerical solution given by (35) is bounded if and only if $|1 + h\lambda| < 1$, i.e. for $h\lambda$ in the unit ball around -1. A similar analysis can be carried out for the other methods resulting in respective restrictions of the admissible step size.

This analysis explains the sharp bend in figures 9(b)-9(d). Only when the step size drops below a certain limit dictated by λ ($h < \lambda^{-1}$ in case of the forward Euler method) the numerical solutions can converge. If the damping is increased, i.e. $\mathrm{Re}\,\lambda \to -\infty$, then for the explicit Euler necessarily $h \to 0$ for the solution to be stable. This means the step size is artificially limited and it cannot be increased beyond the stability limit in order to save work sacrificing some – but not all – accuracy.

3.1.3. The implicit Euler method

To construct a method that better suits our needs we go back to (20) and substitute the differential quotient by a *backward* difference quotient for $y(t+h)$

$$\frac{y(t+h) - y(t)}{h} \approx y'(t+h) = f(t+h, y(t+h)) \; . \tag{36}$$

This time this results in the integration formula

$$Y_{n+1} = Y_n + h f(t+h, Y_{n+1}) \; , \tag{37}$$

the so called backward or implicit Euler method. As its explicit variant this method can be shown to have order 1. Now the numerical solution only is given implicitly as the solution of the possibly nonlinear equation

$$Y_{n+1} - h f(t+h, Y_{n+1}) - Y_n = 0 \; . \tag{38}$$

If we apply this method to the Dahlquist equation we get the recurrence formula

$$Y_n = (1 - h\lambda)^{-n} y_0 \; . \tag{39}$$

Thus, the numerical solution Y_n of equation (33) remains bounded for $|(1 - h\lambda)^{-1}| < 1$. If we assume $\lambda < 0$, this holds for arbitrary $h > 0$. Thus there is no restriction on stepsize, the method is *unconditionally stable*. Figure 11 shows the work-precision diagrams for the implicit Euler method and our examples. We observe that we never loose stability and we especially do not miss the solution by several orders of magnitude compared to the explicit methods, although we loose some accuracy when the timesteps become large.

As a practical tool for graphically visualizing the stability properties of a method we define the *stability region* $\mathcal{S}$ to be the set of parameters, for which the integration method yields a bounded solution:

$$\begin{aligned} \mathcal{S} := \{ z := h\lambda \in \mathbb{C} : \; & \text{the numerical integration} \\ & \text{of equation (33) with step size } h \text{ and} \\ & \text{parameter } \lambda \text{ is stable} \}. \end{aligned} \tag{40}$$

Methods that contain the complete left half-plane in $\mathcal{S}$ are called *A-stable* or *unconditionally stable*. They are well suited for the stable integration of stiff equations. Obviously, the implicit Euler scheme is A–stable, whereas its explicit counterpart is not. The stability regions of all presented methods are shown in figure 10.

After reviewing the process that led us to the definition of the stability region, we can outline a more general idea that will allow us to determine easily the stability of more complex methods. The idea for analysing both Euler methods applied to (33) was to find a closed expression describing the *stability function* $\mathcal{R}$. This function maps the initial value y_0 to the value Y_1, performing a single step of the method

$$\mathcal{R} : y_0 \mapsto Y_1 \; . \tag{41}$$

Thus $Y_n = R(h\lambda)^n y_0$. For the explicit Euler method we found in (35)

$$\mathcal{R}(z) = 1 + z \; , \tag{42}$$

for the implicit version in (39)

$$\mathcal{R}(z) = \frac{1}{1 - z} \; . \tag{43}$$

The definition for the stability region now reads

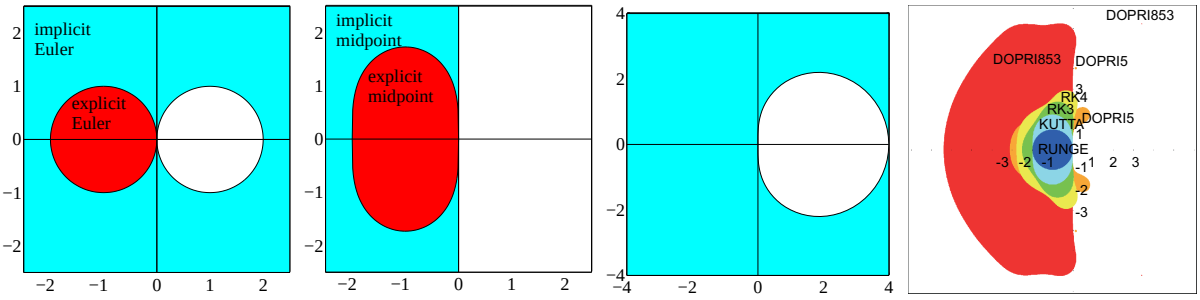
$$\mathcal{S} = \{ z \in \mathbb{C} : |\mathcal{R}(z)| < 1 \} \; . \tag{44}$$

3.1.4. Methods of higher order

To find a higher order method, we go back to equation (28) and do not replace the $y(t + h/2)$ but take the formula as an implicit definition of $y(t+h)$. We get the *implicit midpoint rule*

$$Y_1 = Y_0 + h f\Big(\frac{Y_1 + Y_0}{2} \Big). \tag{45}$$

Further on we use a simplified notation for advancing one step, writing Y_0 and Y_1 instead of Y_n and Y_{n+1}.



(a) Euler methods (b) midpoint rules (c) BDF(2) (d) RK methods

Figure 10: *Stability regions (shaded) of the methods.*

Alternatively the midpoint rule can be derived as a *collocation method* with $s = 1$ internal nodes, i.e. by constructing a polynomial interpolating the particle trajectories at a given, fixed set of s nodes [27]. This idea allows for the construction of implicit Runge-Kutta methods with arbitrary order. In contrast to explicit methods the matrix (a_{ij}) ceases to be strictly lower triangular. These methods are computationally more expensive, so we just stick to the midpoint rule. Its stability function is given by

$$\mathcal{R} = \frac{1+z/2}{1-z/2}. \tag{46}$$

As $\mathcal{R} \leq 1$ for any $\mathrm{Re}\, z < 0$ the implicit midpoint rule is A-stable.

As another possible choice we now introduce *multistep methods*. They are computationally inexpensive because they have no inner stages and some of them are A-stable. A multistep method with k steps is of the general form

$$\sum_{j=0}^{k} \alpha_j Y_{k-j+1} = h \sum_{j=0}^{k} \beta_j f_{k-j+1}, \tag{47}$$

with $f_{n+j} := f(t_{n+j}, Y_{n+j})$. Here we also have 'history points' with negative indices. The coefficient α_k is required to be nonzero; for variable time step sizes the coefficients depend on the last stepsizes, which we have omitted here for the ease of demonstration. Important special cases are the class of *Adams methods* where $\alpha_0 = \cdots = \alpha_{k-2} = 0$:

$$Y_1 = Y_0 + h \sum_{j=0}^{k} \beta_j f_{k-j+1} \tag{48}$$

and the class of *BDF–methods* (**b**ackward **d**ifferentiation **f**ormulas) with $\beta_0 = \cdots = \beta_{k-1} = 0$:

$$\sum_{j=0}^{k} \alpha_j Y_{k-j+1} = h\beta_k f_1. \tag{49}$$

If the formula involves the right-hand side f_1 at the new approximation point Y_1 the method is said to be implicit. BDF-methods are always implicit. The coefficients can again be constructed by a collocation approach. The stability regions of implicit and explicit Adams methods are bounded and located around the origin, thus they are not interesting for large time steps.
BDF-methods were the first to be developed to deal with stiff equations and possess an unbounded stability region covering a sector within the negative complex half-plane. Therefore they are widely used today. For $k+1$ points, these methods possess order $k+1$ and only one nonlinear system has to be solved, whereas s coupled systems have to be solved for an s-stage implicit Runge-Kutta method.

The BDF-method for $k = 1$ is just the implicit Euler method, for k=2 the method is given as

$$Y_1 = \frac{4}{3}Y_0 - \frac{1}{3}Y_{-1} + \frac{2}{3}hf(t+h, Y_1) \tag{50}$$

The stability region of BDF(2) and the other implicit methods are shown in figure 10.

3.1.5. The Verlet method

As a last method we will discuss a scheme commonly referred to as leapfrog or Stoermer-Verlet method. It is especially efficient if (18) is given as the second order system

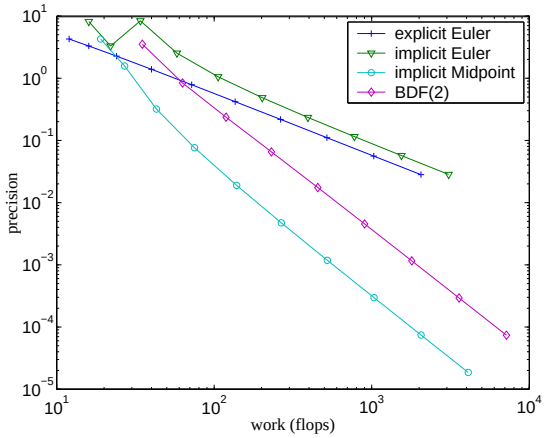
$$x''(t) = f_v\big(t, x(t)\big), \tag{51}$$

i.e. $f_v\big(t, x(t), x'(t)\big) = f_v\big(t, x(t)\big)$. It is not applicable to general first order systems of the form (20).

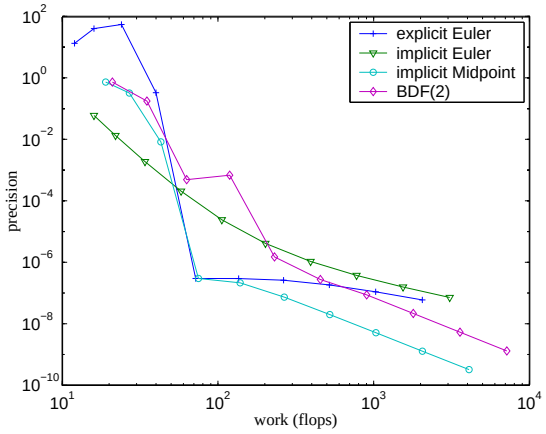
To derive it, we use centered differences at a staggered grid (figure 12) i.e. we now approximate v at $t + (2i+1)h/2$ and x at $t + ih$ by centered differences

$$\frac{v_{n+1/2} - v_{n-1/2}}{h} = f(x_n) \tag{52}$$

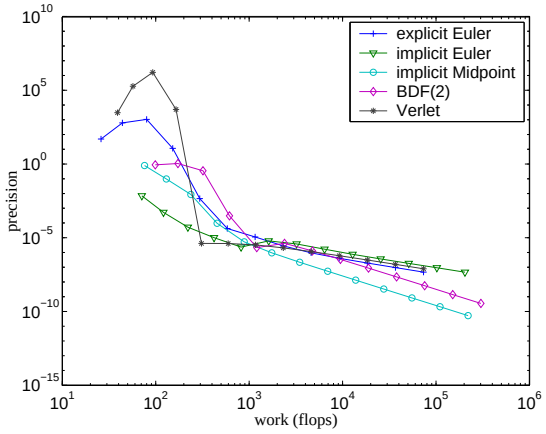
$$\frac{x_{n+1} - x_n}{h} = v_{n+1/2} \tag{53}$$



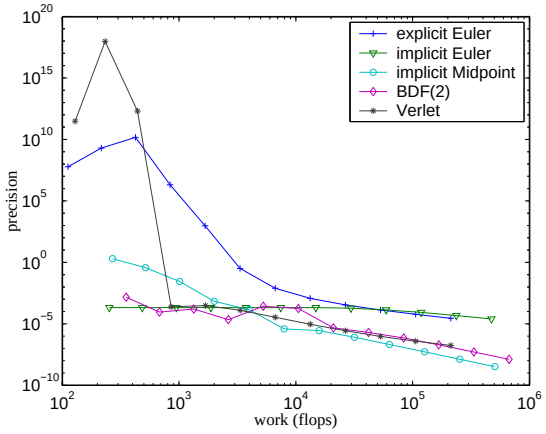
(a) Example 1a: $y' = 2y$



(b) Example 1b: $y' = -15y$



(c) Example 2: $y'' = -5y' - 5y$



(d) Example 3

Figure 11: *Work precision diagrams for the implicit Euler, implicit midpoint, BDF(2) and Verlet (whenever possible) methods. The results for Euler are for comparison and the same as in figure 9.*

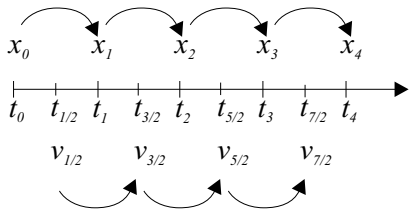


Figure 12: *Staggered grids for the Verlet method.*

thus

$$v_{n+1/2} = v_{n-1/2} + hf(x_n) \tag{54}$$

$$x_{n+1} = x_n + hv_{n+1/2}. \tag{55}$$

The method possesses order 2 as one can see by substituting (54) into (55) resulting in the second order centered difference

$$\frac{x_{n+1} - 2x_n + x_{n-1}}{h} = f(x_n). \tag{56}$$

From this equation an alternative formulation of the Verlet scheme as a multistep method can be derived

$$v_n - v_{n-1} = hf(x_n) \tag{57}$$

$$x_{n+1} - x_n = hv_n, \tag{58}$$

which omits the half steps and staggered grids from above. Now for second order equations which do not possess the

form of (51) one may replace $f(x_n)$ by $f(x_n, v_{n-1})$ at the expense of some stability. Correctly the replacement had to be with $f(x_n, v_n)$ but this would result in a implicit method. Now we can apply the method to examples 2 and 3.

Figure 11 shows the work precision diagrams for the implicit methods. Even for large time steps these methods give approximations to the exact solution. After a somewhat uneven convergence phase all the explicit methods converge smoothly for decreasing time step to the exact solution with a slope given by their order.

Up to now we have omitted a discussion of the stability properties of the leapfrog method. From the examples it can be observed that the method cannot be unconditionally stable. Indeed some more delicate computations show that the method only delivers a bounded solution for arbitrary $h > 0$ for purely oscillatoric equations, i.e. for second order ordinary differential equations of the form (51) (with no damping term) and a contractive right hand side. Nevertheless the method remains well behaved in the presence of low damping. In figure 13 we applied the Verlet method to the undamped wave equation $y'' = 5y$ and it is clearly one of the best choices over a wide range of accuracy requirements.

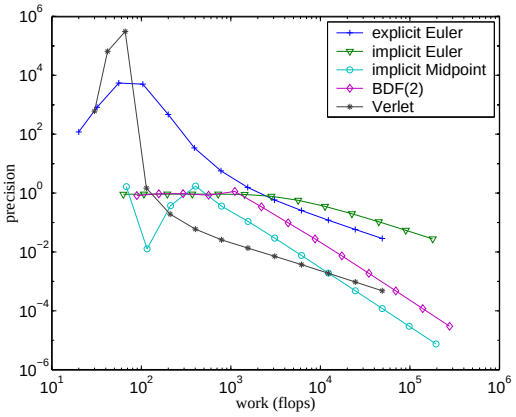


Figure 13: *Work precision diagram for the explicit/implicit Euler, implicit midpoint, BDF(2) and Verlet methods for the wave equation $y'' = -5y$ without damping.*

3.1.6. Selecting an efficient method

Which method is best for a certain application? This question is nearly impossible to answer a priori. The only choice is to try a set of methods and to evaluate which one performs best. Choosing the methods to try, though can be done based on theoretical considerations and observations of the problem at hand. A possible strategy is shown in figure 14.

The same statement holds for predicting the efficiency of a method. Generally, implicit methods require more work per step. On the other hand one may be able to use time steps that are several magnitudes larger than the ones explicit methods would allow. Although accuracy will suffer, the integration won't be unstable. If evaluations of the right hand side function are cheap, a step with RK(4) is faster than an implicit step with BDF(4). On the other hand if it is cheap to compute a good sparse approximation to the jacobian, it may be more efficient to solve the linear system with a few cg iterations than to perform 4 full function evaluations.

3.2. Solving nonlinear systems

All implicit methods, unless they use a linear force formulation as the one described in section 2.2.6, require the solution of a nonlinear system. The implicit Euler method for example reduces our integration problem to the solution of the nonlinear system

$$Y_1 - hf(Y_1) - Y_0 = 0. \tag{59}$$

The other methods yield a system of similar form, namely

$$Y_1 - hf(\tfrac{1}{2}(Y_1 + Y_0)) - Y_0 = 0 \tag{60}$$

$$Y_1 - \tfrac{2}{3}f(Y_1) + \left(-\tfrac{4}{3}Y_0 + \tfrac{1}{3}Y_{-1}\right) = 0 \tag{61}$$

for the midpoint and BDF(2) rule, respectively. This is a nonlinear system of dimension $6N$. This system must be solved with Newton's method to allow for arbitrary step sizes independent of λ.

3.2.1. Newton's method

For the nonlinear system $G(Y) = 0$ we compute a numerical solution by the following algorithm:

Algorithm 1: Newton's Method

(1) **for** $k = 1, 2, \ldots$ **until** convergence **do**
(2) Compute $G(Y^{(k)})$.
(3) Compute $J^{(k)} = \frac{\partial}{\partial Y} G(Y^{(k)})$.
(4) Solve $J^{(k)} s^{(k)} = -G(Y^{(k)})$.
(5) $Y^{(k+1)} := Y^{(k)} + s^{(k)}$
end

Applying Newton's method reduces the problem to the successive solution of linear systems. In a classical Newton method this is achieved by Gaussian elimination. This introduces a lot of non-zero elements into the factors. Although reordering techniques alleviate the effects, this approach is currently too expensive for fast cloth animations.
A lot of authors [3, 50] use iterative methods to solve the linear system. We will also use the conjugate gradient (cg) method here to solve the linear systems in each Newton step. However, this changes the convergence behaviour of the outer Newton method, which is referred to as an inexact Newton method[44], given by algorithm 2.

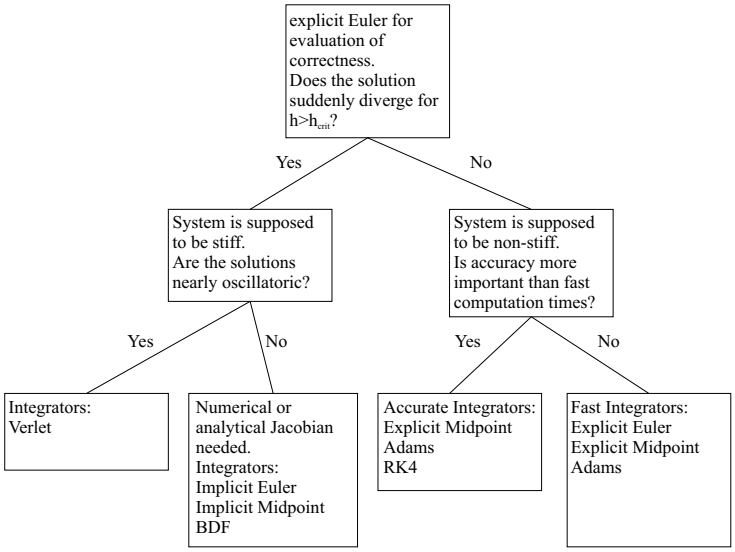


Figure 14: *Selecting a method.*

Algorithm 2: Inexact Newton Method

(1) **for** $k = 1,2,\ldots$ **until** convergence **do**
(2) Compute $G(Y^{(k)})$.
(3) Compute $J^{(k)} = \frac{\partial}{\partial y}(Y^{(k)})$.
(4) Find $s^{(k)}$ with $J^{(k)}s^{(k)} = -G(Y^{(k)})) + r^{(k)}$, such that $\|r^{(k)}\| \leq \eta_k \|G(Y^{(k)})\|$.
(5) $Y^{(k+1)} := Y^{(k)} + s^{(k)}$
end

3.2.2. Residual control

The error of the iterative solution of the linear system is formulated in terms of the residual, which is easily computationally accessible, whereas the actual error cannot be computed. The tolerance of the linear iteration is decreased proportionally to the monotonically decreasing residual of the nonlinear iteration.
An analysis of this method[44] shows that it converges under rather weak additional assumptions. If the classical Newton method converges and the scalar tolerances $\eta^{(k)}$ are uniformly bounded by an $\eta < 1$, the inexact method converges. In literature the $\eta^{(k)}$ are referred to as *forcing terms*. Note that this additional assumption is also necessary: For $\eta = 1$, $s^{(k)} = 0$ would be admissible and the iteration would stagnate.
The inexact method then at least converges linearly, whereas Newton converges superlinearly. By choosing the η_k to converge to zero sufficiently fast[44], the convergence of the inexact Newton method can be forced to have an order > 1. In a neighbourhood of the solution the convergence usually speeds up. By extrapolating the solution of the previous time step we obtain a good initial value for the new solution and the method converges quickly using the constant bound $\eta = 0.02$ without imposing a too strict tolerance on the linear solver.

3.2.3. Inexact simplified Newton methods

The efficiency of the Newton method can be further improved by another approximation. In the simplified version of Newton's method the Jacobian $J^{(k)}$ is approximated by $J^{(0)}$. Such a scheme can be rewritten in the form of an inexact Newton method, if the linear system is written as follows and J is chosen as approximation to $J^{(k)}$

$$\begin{aligned} Js^{(k)} &= -G(Y^{(k)}) + (J - J^{(k)})s^{(k)} + r^{(k)} \\ &=: -G(Y^{(k)}) + \tilde{r}^{(k)} \end{aligned} \tag{62}$$

The residual $r^{(k)}$ is replaced by the larger $\tilde{r}^{(k)}$, which can be bounded if $J \approx J^{(k)}$. By choosing $\tilde{\eta}^{(k)}$ appropriately, the method still converges. In fact, we trade some accuracy approximating $J^{(k)}$ against accuracy in solving the linear system and up to a certain limit the method still behaves as before.

This degree of freedom can be further exploited by even not computing $J^{(0)}$ but a sparser approximation of it. In [28], cloth is modelled using the continuity based approach from section 2.2.5, and the choice of the approximated Jacobian is motivated by observing that the dominating stiffness is induced into the system by linear spring forces from (13). These are split into a linear and a nonlinear part.
Then, the right-hand side of the system is approximated by the linear expression

$$F_{\text{lin}}(x,v) = \sum_{j|(i,j)\in E} \left[\frac{k_{ij}}{l_{ij}^2}(x_i - x_j) + \frac{d_{ij}}{l_{ij}^2}(v_i - v_j) \right] \tag{63}$$

Writing this equation in matrix notation

$$F_{\text{lin}}(x,v) = Kx + Dv, \tag{64}$$

the full Jacobian of the system is approximated by the Jacobian of $F_{\text{lin}}(x,v)$. Then, (62) is applied with

$$J = I - h\gamma \begin{pmatrix} 0 & 0 \\ K & D \end{pmatrix}, \tag{65}$$

where h is the time step and γ depends on the integration method used. This system of dimension $6N$ can be reduced to a system of dimension $3N$ by exploiting the linear relation between position and velocity [29].
This choice of the Jacobian has two major advantages over the full Jacobian. First, J is inexpensive to compute and only changes when either the material constants or the step size changes. Second, we reduce the entries in the Jacobian to approximately a third of the entries in the sparsity pattern of the full Jacobian. Hence an iteration of the linear solver only requires a third of the original time. Obviously this is a major speed-up for the solver. The resulting algorithm is surprisingly simple.

Algorithm 3: Inexact Simplified Newton's Method

(1) Compute $J \approx \frac{\partial}{\partial Y} G(Y^{(0)})$.
(2) **for** $k = 1, 2, \ldots$ **until** convergence **do**
(3) Compute $G(Y^{(k)})$.
(4) Find $s^{(k)}$ with $Js^{(k)} = -G(Y^{(k)})) + r^{(k)}$,
such that $\|r^{(k)}\| \leq \tilde{\eta}_k \|G(Y^{(k)})\|$.
(5) Update $Y^{(k+1)} := Y^{(k)} + s^{(k)}$
end

3.2.4. Adaptive time stepping

Newton's method can also be used to control the step size of the ODE solver. If the convergence of Newton's method is poor, the time step h is reduced such that the solution of the previous time step is a better start value for the current time step and achieves a faster convergence.

3.3. Comparison of methods

We are now able to describe many of the current approaches for cloth simulation in a unifying way.

Explicit integration methods, mostly together with a geometric post-correction step as described in section 2.1.1, have been used for instance by Provot [43], Volino [54], and Fuhrmann [22].

Baraff and Witkin [3] formulate nonlinear constraints and use their linear approximation for the construction of an implicit solution method commonly referred to as linear implicit Euler. This way the system to be solved also becomes linear and can be solved efficiently by a conjugate gradient method (see also [2]). This method corresponds to the solution of a nonlinear system with only one Newton iteration. Because the nonlinear part is not integrated, with high stiffness one may encounter problems which were addressed in [17, 28].

Following [3], implicit integration has been employed with varying physical models for instance in [50, 9, 8, 21]. Desbrun et al. [15, 41] also use a linear implicit method combined with Provots model [43] (section 2.1.1). But instead of linearizing the whole system, they split it in a linear and nonlinear part and use a precomputed inverse of A for solving the linear part of the equations. They don't aim at solving the equation completely, as they don't integrate the nonlinear term explicitly. Instead the angular momentum is corrected to account for the nonlinear part. With this algorithm one can neither change the stepsize h nor deal with an Jacobian depending on t. Debunne et al. [14] use a simplified version of a linear implicit Euler method. To solve the differential equation arising from their geometric nonlinear Green model, they apply an implicit Euler method using a single Newton iteration. The Jacobian is approximated by the 3x3 block-diagonal of the linear Cauchy tensor, allowing a very fast inversion and application.

4. Conclusions for Physical Models and Numerical Solvers

We described physical models for cloth ranging from mass-spring and particle systems to finite element models. Moreover, we discussed and compared several explicit and implicit time integration methods. All these techniques form the basis of current physically based cloth simulation engines, and are suitable to be combined with hybrid approaches for real-time animation of clothing such as [12, 13].

Now we want to proceed to suitable interaction techniques for cloth in order to provide manipulation tools for fitting garments and tailoring personalised clothes. We present an overview of a real time application for cloth simulation in virtual environments followed by special virtual tailor tools.

5. The Virtual Dressmaker

In this section, we present a virtual tailor room which allows the interactive design and modification of clothing in a 3D Virtual Environment. In particular, we propose algorithms and interaction techniques for sewing and cutting garments during a physical cloth simulation, including the automatic modification of the underlying planar cloth patterns. The usability of the described methods is shown in a Virtual Reality application for interactive garment design that provides nearly live-size stereo projection combined with two-handed 6DOF interaction.

5.1. Introduction

Interactive garment design was a research topic in computer graphics already fourteen years ago, when geometric methods to model garment patterns as static three-dimensional surfaces on a workstation were proposed [31]. In the meantime, physical models and fast numerical algorithms now provide the simulation of the dynamic draping behaviour of clothing at interactive frame rates. Furthermore, Virtual Reality techniques and concepts, mostly offering input devices with six degrees of freedom and stereoscopic visualisation, have proved to be a useful alternative to traditional desktop applications with two-dimensional input devices and monoscopic displays in many areas.
While virtual prototyping in general has become a major area of research, the interactive design and simulation of clothing in Virtual Environments is still a relatively new topic. In [37, 38, 55, 36] an approach to interactive cloth assembly and simulation in Virtual Reality is proposed. This application, called the *Virtual Dressmaker*, allows to construct and manipulate clothes coupled with physically based cloth simulation. Usability evaluations have shown that Virtual Environments can considerably improve the user performance in the assembly tasks compared to traditional desktop applications.

5.2. The Virtual Environment

Our Virtual Reality hardware setup consists of a desktop PC with a stereo table top display, a Virtual Table by Barco, with a visible screen of 140x105 cm^2 combined with LCD stereo shutter glasses for alternating projection (60 Hz). As input device we use an electromagnetic 6DOF tracker (Ascension, Flock of Birds) with three receivers. One receiver is attached to the shutter glasses and used to track the user's head position and to determine his gaze direction. The second receiver is attached to a transparent palette on which the interaction elements grouped in different sheets are back projected from the Virtual Table. The third receiver is attached to a pen with two buttons and is held in the dominant hand of the user. The virtual counterpart of the pen in the virtual scene is used to select, drag, and drop garment patterns, to operate the buttons and sliders on the palette, and to navigate in the scene. Hence, the application provides two handed interaction while allowing the user to exploit well-known concepts of a desktop application like menus and sliders on the palette. Similar techniques for two handed interaction in VR have been explored and used extensively, e.g. in [1, 4, 47, 11, 6].

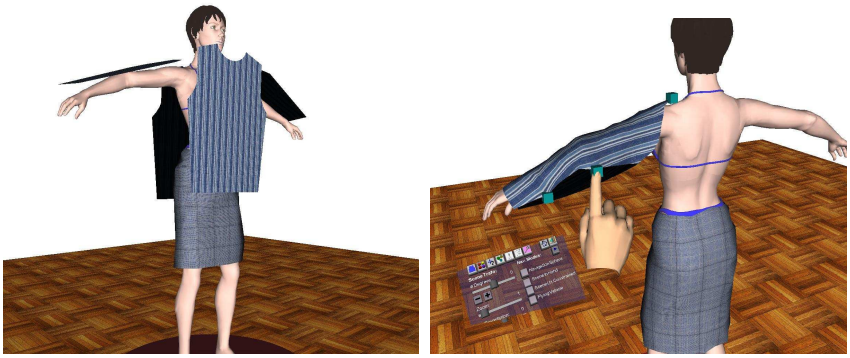
Our application is based on the software framework Studierstube API [45] developed at the University of Vienna. It provides a library extending the standard Open Inventor functionality to handle tracker events in order to embed desktop interaction elements within a VR framework. The application is coupled with a cloth simulation module to sew the prepositioned patterns together and drape the cloth over an avatar.

5.3. Interaction Elements

The Pen For assembling garments in the Virtual Dressmaker, the pen is used to freely manipulate the position of the garment patterns. By choosing a garment pattern, i.e. by pressing the front button on the pen, the object is selected, highlighted in orange, and is attached to the pen such that it can be moved in the space. To navigate the avatar, it is placed on a turntable which can be rotated around and translated along the vertical axis. Moreover, there is a possibility to zoom in and out of the scene and to move the camera position (viewing position). We therefore integrated the well known navigation methods *Scene in Hand* and *Flying Vehicle* [56] into the application which can be applied by pressing the pen's second button. Moreover, we improved the Scene in Hand tool by constraining the navigation possibilities to the degrees of freedom of the turntable. 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 because the user is not prescinded from the working area and remains immersed in the scene.

Advanced Interactive Prepositioning In order to preposition cloth patterns that have to be bent during the sewing process, such as the sleeves of the pullover, we use the cloth simulation engine in combination with the input devices. To this end, Virtual Pins are integrated into the physical cloth simulation as constraints. Thus, we start the simulation for the sleeve only with the seams not yet connected. For this process we provide a menu on the palette to simulate the last touched pattern. Hence the user chooses the relevant pattern and starts the single pattern simulation. Then, we let the user pin parts of the cloth to arbitrary positions, while for the rest of the sleeve the according drape is simulated (Figure 15). When a satisfactory position is reached, the regular sewing and simulation process can be started.

The Palette Sheets The palette provides 3D widgets for buttons and sliders, resembling a 2D menu within the 3D Virtual Environment. Thus, we are able to display various



(a) (b)

Figure 15: *The sleeves of the pullover are prepositioned interactively. Image (a) shows the flat patterns around the 3D figure. In (b) the user wraps a sleeve around an arm using Virtual Pins.*

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. In order to group the interface elements in meaningful sets, we have implemented different sheets. There is a sheet to choose the avatar and the garment patterns from the database. Moreover, a navigation sheet is provided for precise positioning of the camera. For the simulation, a material sheet and a simulation sheet is implemented. Finally, for the tailoring process, the user can choose between a cutting and a sewing sheet.

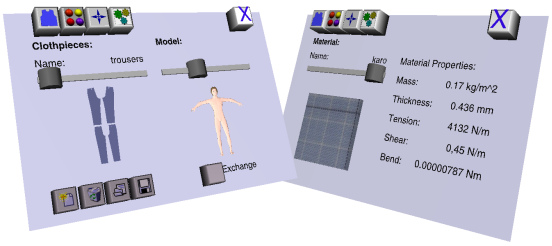


Figure 16: *The navigation and material sheets.*

6. 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. This results in a natural starting condition for cloth animations, and recent work on cloth animation has focused on this approach [18, 19, 51, 9, 24, 21]. 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 triangulation of the planar patterns provide a natural parametrisation for the cloth simulation. For particle systems, the edges in the planar meshes can be taken as the spring's rest lengths [18, 51, 9]. For finite element models, the discretised patterns allow the computation of the basis functions [19, 21].
- Correct texture coordinates for rendering the simulated garments are obtained automatically by the planar rest state. They can be easily mapped to the three dimensional garment during the sewing process and provide a realistic visualisation.

A typical work flow within the Virtual Dressmaker starts by choosing a human model in the corresponding menu on the palette. This can be a body scan or some modelled 3D figure. Then, a set of garment patterns can be chosen and positioned interactively around the avatar (see Figure 18). Both the interaction with the menus on the palette and the manipulation of the patterns in the scene is done with the pen. 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 pick and move parts of the garments. The design and development of virtual tools for the Virtual Dressmaker is independent of the cloth simulation engine which is decoupled from the Virtual Reality application via a client-server interface.

In the following, we first specify the necessary input data for our tailoring algorithms. Then, we describe the sewing process (i.e. the topological mergence of the patterns along seam lines) and the generation of a correct rest state for the physical cloth simulation.

6.1. 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 woman's skirt in Figures 17, which are generated from real garment pattern shapes given by the boundary curve in a CAD system.

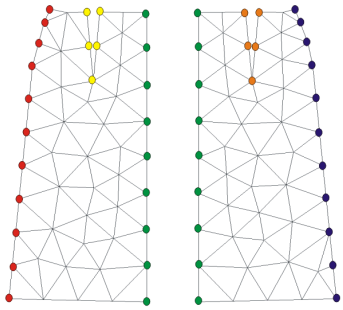


Figure 17: *The seam points of two triangulated planar patterns. The yellow and orange seams belong to the same pattern, whereas the green ones belong to two different patterns. Note that the red and blue seams on the very left respectively right hand side belong to borders on a third pattern of the skirt.*

Second, the seam information has to be available. A seam belongs to exactly two (not necessarily distinct) patterns and consists of a starting 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 (u_i, v_i) with u_i and v_i being the corresponding vertices on two not necessarily distinct plane patterns (see Figure 17). Note that we exclude pathological cases like a vertex pair that belongs to a single triangle. Such a case can be handled by refining the given triangulation of the pattern locally. A new seam can be created by selecting them in pairs with the pen resulting in 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 a button on the corresponding menu.

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 18. In order to provide a good initial position for the subsequent simulation, the requirements 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.
- It should be possible to connect the corresponding seams without penetrating the avatar.

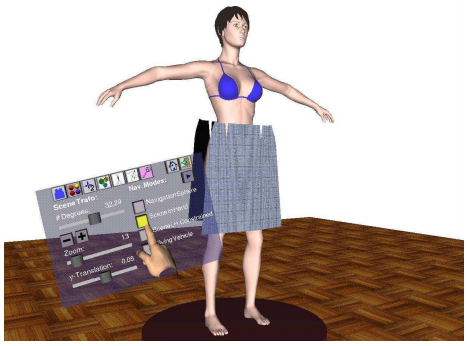


Figure 18: *Cloth patterns prepositioned around a 3D figure.*

Given these prepositioned patterns, the garment can be connected along the corresponding seam lines. Sewing the prepositioned cloth patterns together can be achieved by topologically merging the patterns along the defined seam lines. For all pairs of vertices (u_i, v_i) in all seams, the corresponding triangles are connected, and the seam points are moved halfway between the two original points.

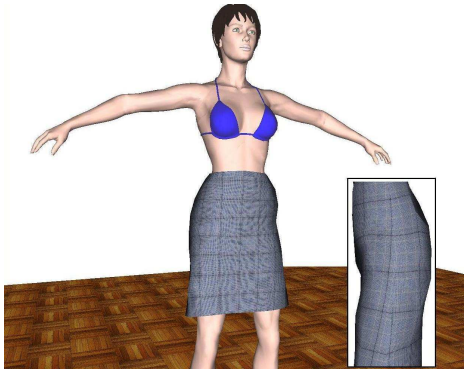


Figure 19: *The result of the simulation with correct texture coordinates along the seam line, where two patterns are connected.*

7. Interactive Cutting

In this section, we present methods for the interactive cutting of clothes in 3D, including automatic modification of the planar patterns and the seam information. First, we describe how cuts can be defined on the garments, then we explain how they are realised on the 3D mesh and how they are transferred to the planar cloth patterns.

In order to define a cut on the simulated 3D mesh, the user

has to mark points on the garment with the pen. Between these selected vertices, the mesh will be cut in straight lines by the cutting algorithm described in the next section.

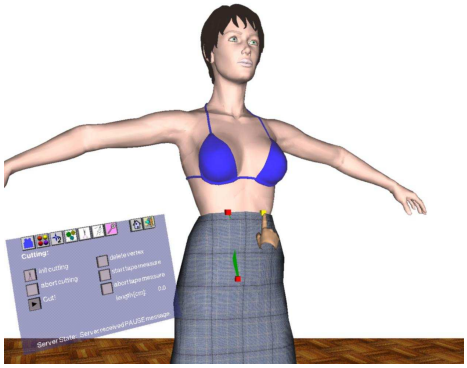


Figure 20: *A cut in a skirt is defined by the user.*

Then, the corresponding patterns and seams are modified accordingly, and the physical simulation continues computing the drape and movement of the garment. An example is given in Figure 20.

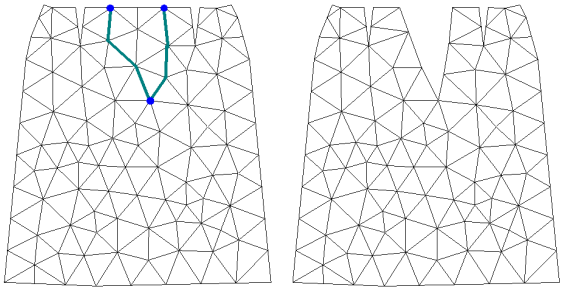


Figure 21: *The front pattern of the skirt before (left) and after (right) the cutting (cf. Figure 20). In the left pattern, the vertices selected by the user, and the edges that have to be cut are highlighted. Note that the cutting line in the patterns is not necessarily straight, as shown in the right image, due to the deformed state of the 3D mesh during the cutting.*

For the underlying algorithm, 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 modelled along the edges of the mesh. Since the second approach in general does not result in straight cuts, the vertices of the cut edges should 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 6. Thus, the special case of cutting along a seam line can be handled simply by removing the corresponding seams.

As mentioned above, in order to obtain a straight cutting line, we then 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 24). 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.

7.1. The Cutting Disk

The Cutting Disk provides an interactive cutting tool for executing plain cuts in the three-dimensional setting. It is most valuable for shortening garments like pairs of trousers or a shirt. Additionally it can be used to transform a dress into a skirt and a top as shown in Figure 23.

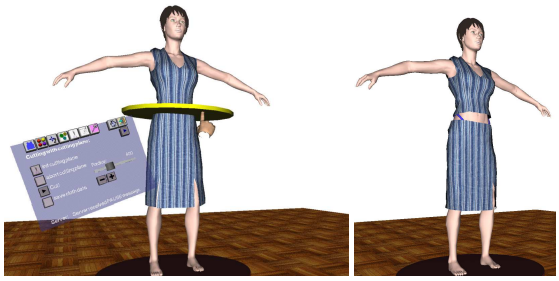


Figure 23: *The use of the Cutting Disk: A cutting disk is placed at the waist of an avatar to cut a dress. A smooth cut is calculated by our algorithm and the dress is split into a top and a skirt which becomes apparent during the course of the simulation.*

To define a cut the user can interactively position a thin cylinder, where the radius is adjustable. By the final cutting position, the user determines the plane where the garment mesh has to be cut. To determine the cutting line, we have to find a path on the edges of the mesh, where the corresponding triangles are cut by the cylinder. Therefore,we store all triangles affected by the Cutting Disk. The algorithm starts with the closest mesh vertex of the concerned triangles to the cutting plane. To find the cutting path, we choose the neighbouring vertex of the actual end of the cutting line which belongs to a cut triangle and which is the closest to the cutting plane. To avoid undesired closed loops (e.g. the vertices of one triangle), we keep track of the chosen vertices which can never be visited again except the start vertex. This vertex can only be reached when at most half of the affected triangles in the cutting list have been visited. The found vertices

(a) (b) (c)

Figure 22: *Image (a) shows the simulated pullover, together with the skirt. In (b) the user has shortened the pullover by cutting off a broad band on the lower part. Note that the cutting lines in this example cross two seams. In image (c) the design of the collar has been modified interactively.*

of the path are then projected onto the Cutting Disk. The smoothing of the cut and the modification of the underlying cloth patterns is done as described in section 7.

The examples given in Figures 20 and 22 show some results of the proposed algorithms and interaction techniques for garment design. In Figure 22 a pullover is shortened, and the design of the collar is changed. All simulation images are directly captured from our Virtual Dressmaker application while accomplishing the given task.

Figure 24: *A pullover is shortened by cutting a piece off. The first image shows the cut without aligning the vertices along the cut edges, in the second the vertex positions have been corrected and a straight cut is obtained. All the modifications are transferred to the planar patterns.*

References

[1] I. Angus and H. Sowizral. Embedding the 2D interaction metaphor in a real 3D virtual environment. *Stereoscopic Displays and Virtual Reality Systems. Proceedings SPIE*, 2409:282–293, 1995.

[2] U. Ascher and E. Boxerman. On the modified conjugate gradient method in cloth simulation. *The Visual Computer*, 2003.

[3] David Baraff and Andrew Witkin. Large Steps in Cloth Simulation. In *Computer Graphics (Proc. SIGGRAPH)*, pages 43–54, 1998.

[4] M. Billinghurst, S. Baldis, L. Matheson, and M. Philips. 3D palette: A virtual reality content creation tool. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology (VRST-97)*, pages 155–156, New York, September 15–17 1997. ACM Press.

[5] Javier Bonet and Richard D. Wood. *Nonlinear continuum mechanics for finite element analysis.* Cambridge University Press, 2000.

[6] D. A. Bowman, M. Setareh, M. S. Pinho, N. Ali, A. Kalita, Y. Lee, J. Lucas, M. Gracey, M. Kothapalli, Q. Zhu, A. Datey, and P. Tumati. Virtual-SAP: An Immersive Tool for Visualizing the Response of Building Structures to Environment Conditions. In *Proceedings of IEEE Virtual Reality*, 2003.

[7] Dietrich Braess. *Finite Elemente*. Springer, 1997.

[8] K.-J. Choi and H.-S. Ko. Extending the Immediate Buckling Model to Triangular Meshes for Simulating Complex Clothes. In *Eurographics Short Presentations*, pages 187–192, 2003.

[9] Kwang-Jin Choi and Hyeong-Seok Ko. Stable but Responsive Cloth. In *Computer Graphics (Proc. SIGGRAPH)*, pages 604–611, 2002.

[10] Philippe G. Ciarlet. *Mathematical elasticity. Vol. I.* North-Holland Publishing Co., Amsterdam, 1992. Three-Dimensional Elasticity.

[11] Sabine Coquillart and Gerold Wesche. The virtual palette and the virtual remote control panel: A device and an interaction paradigm for the responsive workbench. In *Proceedings of the IEEE Virtual Reality*, pages 213–216, 13-17 March 1999.

[12] F. Cordier and N. Magnenat-Thalmann. Real-time Animation of Dressed Virtual Humans. *Computer Graphics Forum*, 2002.

[13] 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.

[14] Gilles Debunne, Mathieu Desbrun, Marie-Paule Cani, and Alan H. Barr. Dynamic Real-Time Deformations using Space and Time Adaptive Sampling. In *Computer Graphics (Proc. SIGGRAPH)*, 2001.

[15] Mathieu Desbrun, Peter Schröder, and Alan Barr. Interactive Animation of Structured Deformable Objects. In *Graphics Interface*, pages 1–8, 1999.

[16] Manfredo P. DoCarmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1976.

[17] Bernhard Eberhardt, Olaf Etzmuß, and Michael Hauth. Implicit-Explicit Schemes for Fast Animation with Particle Systems. In *Eurographics Computer Animation and Simulation Workshop*, 2000.

[18] Bernhard Eberhardt, Andreas Weber, and Wolfgang Straßer. A Fast, Flexible Particle-System Model for Cloth Draping. *IEEE Computer Graphics and Applications*, 16(5):52–59, 1996.

[19] Jeffrey W. Eischen, Shigan Deng, and Timothy G. Clapp. Finite-element modeling and control of flexible fabric parts. *IEEE Computer Graphics and Applications*, 16(5):71–80, September 1996 1996. ISSN 0272-1716.

[20] Olaf Etzmuß, Joachim Groß, and Wolfgang Straßer. Deriving a Particle System from Continuum Mechanics for the Animation of Deformable Objects. *IEEE Transactions on Visualization and Computer Graphics*, 2003.

[21] Olaf Etzmuß, Michael Keckeisen, and Wolfgang Straßer. A Fast Finite Element Solution for Cloth Modelling. *Proc. Pacific Graphics*, 2003.

[22] A. Fuhrmann, C. Groß, and V. Luckas. Interactive Animation of Cloth including Self Collision Detection. In *Journal of WSCG*, 2003.

[23] Eitan Grinspun, Petr Krysl, and Peter Schröder. CHARMS: a simple framework for adaptive simulation. In *Computer Graphics (Proc. SIGGRAPH)*, 2002.

[24] Clemens Groß, Arnulph Fuhrmann, and Volker Luckas. Automatic pre-positioning of virtual clothing. In *Proc. of the Spring Conference on Computer Graphics*, pages 113–122, 2003.

[25] Joachim Groß, Olaf Etzmuß, Michael Hauth, and Gerhard Bueß. Modelling viscoelasticity in soft tissues. In *Int. Workshop on Deformable Modeling and Soft Tissue Simulation*, 2001.

[26] E. Hairer and G. Wanner. Solving Ordinary Differential Equations I & II. Springer-Verlag, Berlin, 1996.

[27] E. Hairer and G. Wanner. Solving Ordinary Differential Equations II. Springer-Verlag, Berlin, 1996.

[28] M. Hauth and O. Etzmuß. A High Performance Solver for the Animation of Deformable Objects using Advanced Numerical Methods. In *Computer Graphics Forum*, pages 319–328, 2001.

[29] M. Hauth, O. Etzmuß, and W. Straßer. Analysis of Numerical Methods for the Simulation of Deformable Models. *The Visual Computer*, 2003.

[30] Michael Hauth and Wolfgang Strasser. Corotational simulation of deformable solids. In *Proc. WSCG 2004*, pages 137–145, 2004.

[31] B. K. Hinds and J. McCartney. Interactive Garment Design. *The Visual Computer*, 6(2), 1990.

[32] Donald H. House and David E. Breen, editors. *Cloth Modeling and Animation*. A K Peters, 2000.

[33] Lukas Janski and Volker Ulbricht. Numerical simulation of mechanical behaviour of textile surfaces. *Zeitschrift für Angewandte Mathematik und Mechanik*, 80(S2):525–526, 2000.

[34] Young-Min Kang, Jjeong-Hyeon Choi, Hwan-Gue Cho, Do-Hoon Lee, and Chan-Jong Park. Real-time animation technique for flexible and thin objects. In *WSCG*, pages 322–329, February 2000.

[35] S. Kawabata. *The Standardization and Analysis of Hand Evaluation*. The Textile Machinery Society of Japan, Osaka, 1980.

[36] Michael Keckeisen, Matthias Feurer, Markus Wacker, and Wolfgang Straßer. Tailor Tools for Interactive Design of Clothing in Virtual Environments. In *Proc. ACM VRST*, 2004.

[37] Michael Keckeisen, Stanislav L. Stoev, Matthias Feurer, and Wolfgang Straßer. Interactive Cloth Simulation in Virtual Environments. In *Proc. IEEE Virtual Reality*, 2003.

[38] Michael Keckeisen, Stanislav L. Stoev, Markus Wacker, Matthias Feurer, and Wolfgang Straßer. A User Study On Advanced Interaction Techniques in the Virtual Dressmaker Application. In *Proc. HCI International*, 2003.

[39] E. Klingbeil. *Tensorrechnung für Ingenieure*. BI Wissenschaftsverlag, 1989.

[40] L. D. Landau and E. M. Lifschitz. *Elastizitätstheorie*. Akademischer Verlag, 1989.

[41] 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.

[42] M. Müller, J. Dorsey, L. McMillan, R. Jagnow, and B. Cutler. Stable Real-Time Deformations. In *Proc. of SIGGRAPH Symposium on Computer Animation 2002*, 2002.

[43] Xavier Provot. Deformation constraints in a mass-spring model to describe rigid cloth behavior. In Wayne A. Davis and Przemyslaw Prusinkiewicz, editors, *Graphics Interface '95*, pages 147–154, May 1995. ISBN 0-9695338-4-5.

[44] W. C. Rheinboldt. *Methods for Solving Systems of Nonlinear Equations*, volume 70 of *CBMS-NSF regional conference series in applied mathematics*. SIAM, second edition, 1998.

[45] Dieter Schmalstieg, Anton L. Fuhrmann, Michael Gervautz, and Zsolt Szalavári. 'Studierstube' - An Environment for Collaboration in Augmented Reality. In *Proceedings of Collaborative Virtual Environments '96, Nottingham, UK, Sep. 19-20*, 1996.

[46] Hans Stephani and Gerhard Kluge. *Theoretische Mechanik*. Spektrum Akademischer Verlag, 1995.

[47] Zs. Szalavári and M. Gervautz. The personal interaction panel - A two handed interface for augmented reality. *Computer Graphics Forum (Proceedings of EUROGRAPHICS'97)*, 16(3):335–346, 1997.

[48] D. Terzopoulos and K. Fleischer. Deformable models. *The Visual Computer*, 4:306–331, 1988.

[49] P. Volino and N. Magnenat-Thalmann. Developing simulation techniques for an interactive clothing system. In *International Conference on Virtual Systems and Multimedia '97*, pages 109–118, 1997.

[50] P. Volino and N. Magnenat-Thalmann. Implementing fast cloth simulation with collision response. In *Computer Graphics International Proceedings*, 2000.

[51] P. Volino and N. Magnenat-Thalmann. *Virtual Clothing*. Springer, 2000.

[52] P. Volino and N. Magnenat-Thalmann. Comparing efficiency of integration methods for cloth animation. In *Computer Graphics International Proceedings*, 2001.

[53] P. Volino and N. Magnenat-Thalmann. Accurate garment prototyping and simulation. In *Computer-Aided Design & Applications,*, volume 2, 2005.

[54] Pascal Volino, Martin Courshesnes, and Nadia Magnenat-Thalmann. Versatile and Efficient Techniques for Simulating Cloth and Other Deformable Objects. In *Computer Graphics (Proc. SIGGRAPH)*, pages 137–144, 1995.

[55] Markus Wacker, Michael Keckeisen, Stanislav L. Stoev, and Wolfgang Straßer. A Comparative Study On User Performance in the Virtual Dressmaker Application. In *Proc. ACM VRST*, 2003.

[56] Colin Ware and Steven Osborne. Exploration and virtual camera control in virtual three dimensional environments. In *Proceedings of the 1990 Symposium on Interactive 3D Graphics*, pages 175–183. ACM Press, 1990.

Part 4: Real-Time Garment Animation

Frédéric Cordier and Nadia Magnenat-Thalmann

MIRALab, University of Geneva - CH-1211 Geneva, Switzerland

Abstract

In this chapter, we describe two methods for cloth animation in real-time. These two algorithms work in a hybrid manner exploiting the merits of both the physical-based and geometric deformations. They make use of predetermined conditions between the cloth and the body model, avoiding complex collision detection and physical deformations wherever possible. Garments are segmented into pieces that are simulated by various algorithms, depending on how they are laid on the body surface and whether they stick or flow on it. Tests show that these methods are well suited to fully-dressed virtual human models, achieving real-time performance compared to ordinary cloth-simulations. An application in mixed and augmented reality is demonstrated.

1. Introduction

One of the most challenging areas in research is in the development of a robust methodology for simulating clothes in real-time. In order to define a cloth simulation system that is able to simulate complex garments realistically, whilst maintaining a reasonable computation time, a deeper study of the cloth model and the identification of its behavior at different levels are necessary.

This study is not intended to integrate yet another more precise physical model of garment behavior, but rather focus on the real-time constraints for the simulation and the visual cloth motion features to which an observer is sensitive. Most of the existing approaches use a general-purpose simulation method using collision detection and physical simulation for the whole garment. Unfortunately, simulations that simply calculate all potentially colliding vertices may generate a highly realistic movement, but do not provide a guaranteed frame time. A new simulation model should be implemented that avoids heavy calculation of the collision detection and particle system wherever possible.

Our assumption is that the whole cloth does not need to be simulated with a general-purpose simulation method; instead many optimizations can be made. For example, the trouser will never collide with the arms. Collision detection may be simplified by restricting the collision detection to only potentially colliding surfaces. Also, stretched garments do not need to be simulated with a complex physical method. It can be simply simulated by keeping an offset between the garment and the underlying skin surface. In this chapter, we will demonstrate how the computation cost can be greatly reduced by making use of predetermined conditions between the cloth and the body model, avoiding complex collision detection and physical deformations wherever possible. We will present two approaches.

The first approach works in a hybrid manner exploiting the merits of both the physically based and geometric deformations. Garments are segmented into pieces that are simulated by various algorithms, depending on how they are laid on the body surface and whether they stick or flow on it. Tests show that the method is well suited to fully-dressed virtual human models, achieving real-time performance compared to ordinary cloth-simulations.

The main idea of the second approach consists of developing a cloth simulator that can learn the cloth behavior through a sequence of pre-computed cloth simulation. This approach enables us to simulate the garment features such as wrinkles, gathers… in real-time. We setup the problem as simulating the garments in two phases. The first phase, a rough mesh reproduces the dynamic behavior of the garments. Its physical properties are defined by the pre-calculated sequence. In the second phase, the fine mesh simulates the details of the garments.

2. Previous Work

The history of research on real-time cloth is relatively recent. Researchers have concentrated mainly on two aspects of real-time cloth animation: simulating the physical properties of garments and collision handling.

2.1. Numerical Solvers

Probably the most common technique for simulating the physical properties of clothes is the particle system. Simulation process is broken down into calculating the internal forces and solving the system of partial derivative equations (PDE). The latter point has attracted much interest in the field of real-time applications, since it requires high computation power.

The explicit Euler method [BAR 98] has been one of the first numerical solvers. Unfortunately, this method is notorious for its instability when using large time steps and stiff equations. Several improvements have been proposed to reduce instability, such as the Verlet integration [KAC 03] and the explicit Euler combined with inverse dynamics [PRO 95] [VAS 01]. Unfortunately, the simulation quality is sacrificed in favor of computation speed, due to the approximations employed in these models.

The implicit Euler method presented by Baraff et al. [BAR 98] performs the computation not by using the derivative at the current time, but the predicted derivative at the next time step. Unlike explicit Euler integration, the implicit Euler method offers higher stability while using large time-steps and clothes with stiff mechanical properties. A major drawback of this numerical solver, however, is the computation of a large linear system,

More recently, researchers worked on saving the computation time of the linear system solver. Desbrun et al [DES 99] proposed solving the linear system with a pre-computed inverse matrix. Kang et al. [KAN 02] proposed further optimization with a direct update formula for the positions and velocities of the cloth vertices. As indicated by the authors, these methods are not intended to provide a physically-correct cloth animation. Our approach to that problem is a data-driven mass-spring system: the simulation is corrected with a set of functions built from the pre-simulated animation. By doing so, we bring the deformation of the mass-spring system closer to the original cloth behaviour.

Another approach to fast garment deformations is hybrid approaches. They aim for a neat combination of physically based deformation and geometric deformation. Cordier et al. [COR 02] proposed to segment the cloth into pieces and simulate these by different algorithms, depending on how they lie on the body surface and whether they adhere to it or flow over it. Others have noted that wrinkle deformation is geometric in nature and therefore can be computed with a geometric method. Wrinkles can be generated either by tessellating the cloth mesh [KAN 02] or rendering details on texture using bump mapping [HAD 99]. The main difficulty is defining a fold function that can simulate all kinds of wrinkle patterns. Moreover, determining the location and shape of wrinkles is left to CG artists. One of our contributions is a geometric wrinkling method that is "trained" by using a pre-simulated cloth sequence, rather than relying on users.

2.2. Collision Handling

Collision detection is usually one of the bottlenecks in real-time animation. The problem is particularly acute in the case of clothes because these objects are highly deformable. Some methods exploit graphics hardware to compute collisions on bump maps [VAS 01]; others use implicit surfaces to check collisions on the body [RUD 00], or voxel trees, which partition the space hierarchically [MEY 00]. Using frame coherency to reduce computation cost has been explored by Zhang et al [ZHA 02]. In this work, we propose a data-driven collision detection method; we use the pre-simulated sequence to localize the collision checks to neighbouring cloth regions that have high probability to collide.

2.3. Data-driven Approaches

The idea of building an interpolator from examples or pre-simulated data has proven to be a valuable tool in a variety of areas of CG, e.g. for modelling a variety of human body shapes and for motion synthesis. The basic idea is to build an interpolation space filled with a set of pairs of input parameters and the targeted graphical objects. During the run-time simulation, given an input, the interpolator will produce the corresponding output by interpolating the examples available in the interpolation space. Cloth animation depends on a high number of parameters and therefore a data-driven approach is difficult to adapt. Very recently, James et al. [JAM 03] resented such an approach, where physics-based deformation and collision detection are both handled in a unified framework. By blending of pre-computed orbits rather than using a mass-spring system, previous unseen results could be achieved, such as garments with stiff mechanical properties in real-time. However, they show little DoF to the clothes under simulation; Instead of resorting to a data-driven approach for the entire simulation, we seek a neat combination of a data-driven approach with the mass-spring system. Unlike previous works, our simulator allows a much higher degree of interaction, as it is often needed in animating clothes on moving characters.

3. Hybrid Geometric and Physical deformations

When observing a garment worn on a moving character, we notice that the movement of the garment can be classified into several categories depending on how the garment is laid on and whether it sticks to, or flows on, the body surface. For instance, a tight pair of trousers will mainly follow the movement of the legs, whilst a skirt will float around the legs. The first part of the study is to identify all the possible categories:

* Garment regions that stick to the body with a constant offset. In this case, the cloth follows exactly the movement of the underlying skin surface.

* Garment regions that flow around the body. The movement of the cloth does not follow exactly the movement of the body. In case of a long skirt, the left side of the skirt can collide with the right legs.

* Garment regions that move within a certain distance to the body surface are placed in another category. The best examples are shirtsleeves. The assumption in this case is that the cloth surface always collides with the same skin surface and its movement is mainly perpendicular to the body surface.

These three categories are animated with three different cloth layers (Figure 1). The idea behind the proposed method is to avoid the heavy calculation of physical deformation and of collision detection wherever possible, i.e. where collision detection is not necessary. The main interest of our approach is to pre-process the target cloth and body model so that they are efficiently computable during runtime. The skin and the garment are divided into a set of segments and the associated simulation method is defined for each. For each layer, we propose solutions and explain why they have been chosen.

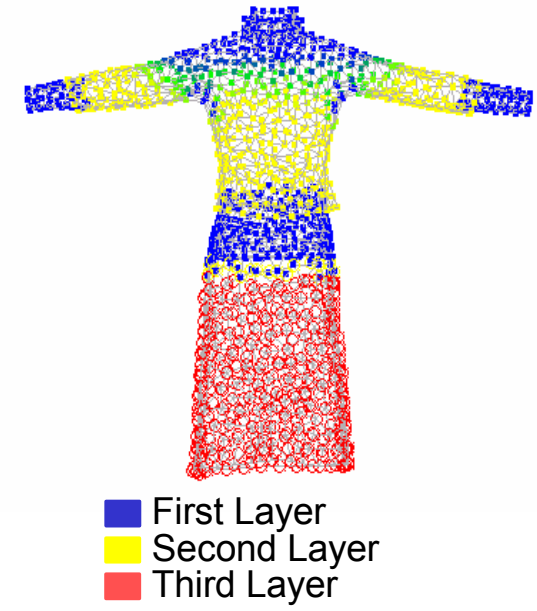


Figure 1: *Garment segmentation.*

3.1. Layer 1: "Stretch clothes"

Tight clothes keep constant distances with the underlying skin surface. The deformation of this layer follows the deformation of the underlying skin. We choose to use the skin deformation method to animate this layer. This method does not involve any collision detection or physical deformation. It has no impact on the calculation time. Therefore, it is necessary to construct the skeletal information of this cloth layer. This information is defined by mapping the attachment information of the underlying skin to these cloth vertices. Each vertex of the garment mesh is associated to the closest triangle, edge or vertex of the skin mesh.

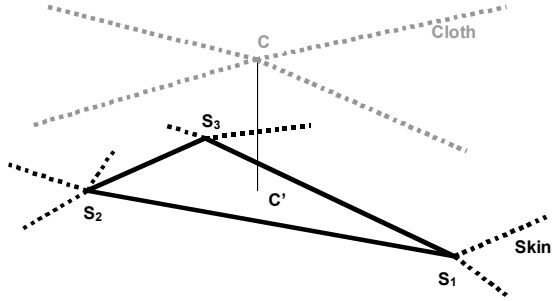


Figure 2: *Mapping the attachment information of the mesh.*

In the Figure 2, the garment vertex C is in collision with the skin triangle S1S2S3. We define C' as the closest vertex to C located on the triangle S1S2S3. We then define the barycentric coordinates of C' with S1, S2 and S3. The attachment information of C is calculated by combining the attachment information of S1, S2 and S3 weighted with the barycentric coordinates.

3.2. Layer 2 "Loose cloth"

For loose clothes, the relative movements of clothes to the skin remain relatively small, keeping a certain distance from the skin surface. To get an intuitive understanding of such cases, consider the movement of sleeve in relation with the arm: for a certain region of the garment, the collision area falls within a fixed region of the skin surface during simulation. With this in mind, the scope of the collision detection can be severely limited. A basic assumption made is that the movement of the garment largely depends on that of the underlying skin and yet it should not follow the skin surface rigidly. It is necessary to simulate the local displacement of the garment from the skin surface.

Two different methods have been developed, one for cloth deformation on the limbs (trousers and sleeves), the other one for the deformation of cloth on the trunk.

3.2.1. Sleeves and trousers

In this approach, we use the assumption that cylinders can approximate the limbs. Each vertex is kept on a disc that is attached to the skin.

Initially, the position of the disc centres are calculated using the attachment information defined in the pre-processing stage. They are obtained by mapping the attachment information of the underlying skin to the cloth vertices (see Section 5). The axe of the discs is parallel to the limb joint.

During simulation, the movement of the vertices on their disc follows the equation of the rigid body motion. Vertices move independently to each other. Using the 2nd Newton's law, velocity and position are easily calculated with gravity force. In case a vertex leaves its disc, a kinematic correction [WIT 01] is applied to the velocity and the position to put the vertex back on the disc surface. The Figure 3 shows a cross-section of a limb with a garment.

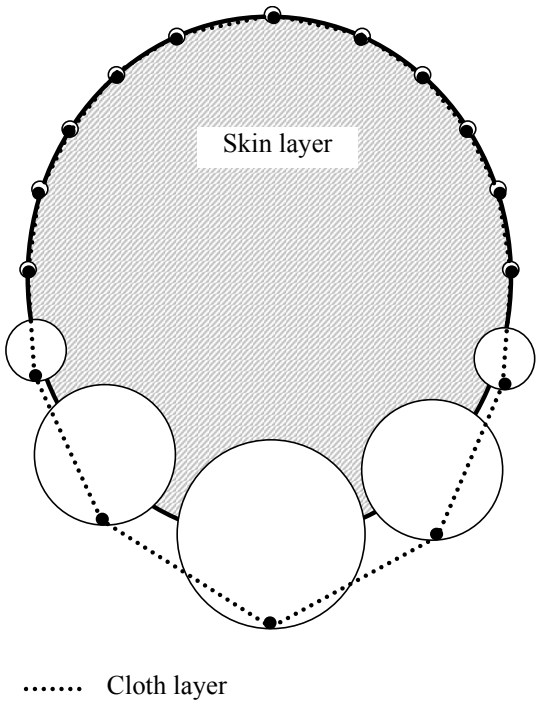


Figure 3: *Cross section of a limb with a garment.*

The size of the disc is a function of the angle between the cloth acceleration vector and the normal to the skin surface. The cloth acceleration vector is the subtraction of the gravity to the acceleration of the associated skin surface. This function is defined in a way that the size of the disc follows roughly the catenary shape, i.e. the shape of a hanging wire. It can be proven that if a heavy flexible cable is suspended between two points, then it takes the shape of a curve with the equation:

$$y = c + a\cosh\left(\frac{x}{a}\right) \qquad (1)$$

Figure 3 gives an example of the variation of the size of the disc along the body surface.

We approximate the limbs (arms, legs…) to cylinders and then in order to determine the size of the discs for the cloth deformation we consider the equation of the circle (the cylinder that approximates the limb) and the equation of the catenary (the garment hold by the limb) as shown on the graph in Figure 4.

The equation of the half disc of diameter **2r** in Cartesian coordinate system (**x**, **y**) is:

$$y = \pm\sqrt{r^2 - x^2} \qquad (2)$$

The equation of the catenary in the same coordinate system (Figure 4):

$$y = a\left(\cosh\frac{x}{a} - \cosh\frac{r}{a}\right) \qquad (3)$$

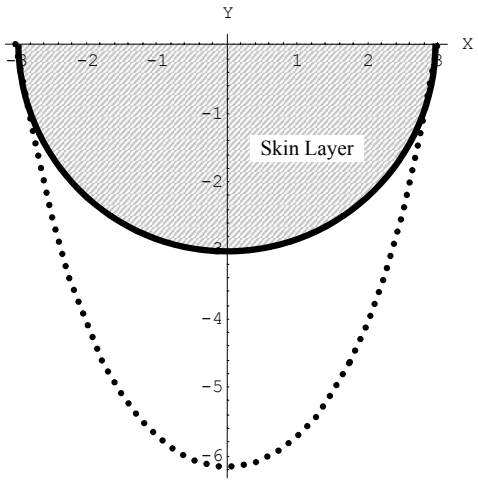


Figure 4: *Equation of the catenary and the disc.*

Where **2r** is the diameter of the limb and a is the scaling factor of the catenary curve. It is used to control the size (largeness) of the garment. We define the ideal size of the disc as the difference of the two equations above:

$$a\left(\cosh\frac{x}{a} - \cosh\frac{r}{a}\right) - \sqrt{r^2 - x^2} \qquad (4)$$

We define a quadratic equation to approximate the size of the disc. This equation should be fast to compute, as it will be used intensively to calculate the movement of every vertex at every frame. This approximation is done by fitting the polynomial function P(x) to the three points P0(x=-r), P1(x=0) and P2(x=r).

$$P(x) = C\left(\left(\frac{x}{r}\right)^2 - 1\right) \quad \textit{where} \quad C = a\left(1 - \cosh\frac{r}{a}\right) + r \qquad (5)$$

In this equation, *C* is a constant that is related to the size (largeness) of the cloth. This parameter is defined by the user at the pre-processing stage.

sin(θ) = x/r is evaluated by projecting the acceleration component **OA** to the normal **ON** of the skin surface as shown in Figure 5. **OA** is the acceleration applied to the cloth. It is the subtraction of the acceleration of the skin vertex **O** to the gravity component.

This method ensures that the rest shape of the garment takes the shape of real garment hanged on a limb. All the discs containing the vertices are parallel to their corresponding limb joints. This ensures that every vertex of the same limb will fall down to the same side on the limb, even in case the direction of the limb is almost vertical. If the limb is perfectly vertical, the result is unpredictable. However, this case never happens in practice.

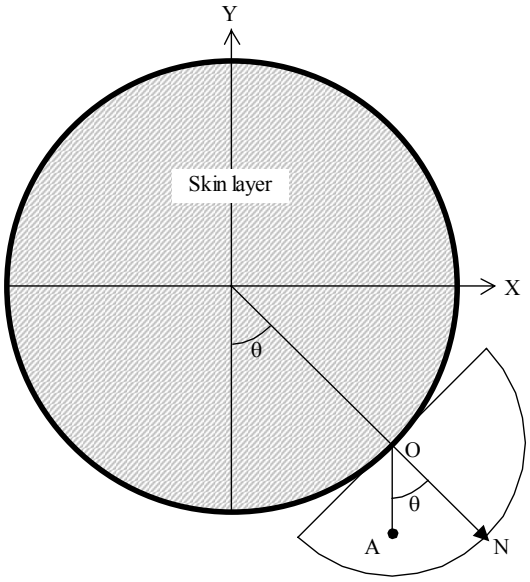


Figure 5: *Computation of the disc diameter.*

In this method, no collision detection on surfaces is necessary. Our particle system is very simple, each vertex moves independently. Therefore, the acceleration applied on vertices is constant. The stability of the system does not depend on the time step duration. The resulting motion of the garments provides the appearance of dynamic movements as well as the shape of real clothes. An example of a sleeve deformation is shown on Figure 6.

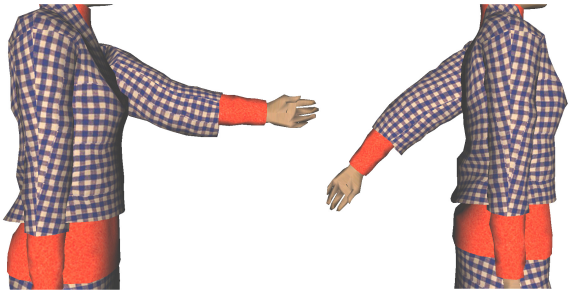


Figure 6: *Deformation of sleeves.*

3.2.2. Clothes on the Trunk

The hybrid technique described in Section 3.2.1 works well if the skin surface can be approximated by a moderately thin cylinder and the movement of the garment is more or less perpendicular to the skin surface. This assumption is true for the arms and legs; the friction prevents the garments to move along the limbs. In such cases, vertices are considered to move independently of each other. However, such an assumption cannot be made for garment regions around the trunk where, unlike the arm, the movements of the garment are driven not only by the torso but also by the potentially complex movement of the shoulder and the arm. To get an intuitive understanding of the problem, consider a shirt worn by a virtual human who is raising its arms. The whole garment around the torso part moves along the vertical direction as the arms are raised. This implies the necessity to simulate the elastic property of the tissue. Therefore, a dedicated method for cloth deformation on the trunk has been developed.

A simplified mesh composed of 42 nodes is animated using a simplified mass-spring system. The movement of these nodes is driven by two interactions: gravity and springs in connection with neighbouring nodes. Each of these nodes is a control point used to deform the cloth mesh on the trunk. These control points are uniformly placed on the trunk, they form a grid of 3x4x3 points. Six additional control points are attached to the shoulder. These control points maintain the grid mesh on the trunk and prevent the mesh from falling due to the gravity. Each control point is included in a half-sphere that is attached to the skeleton. The collision detection algorithm verifies if each control point is contained within its corresponding hemisphere. In case the control point is not in its hemisphere, a kinematical correction is applied on the position and the speed.

The cloth mesh is deformed with the Freeform Deformation method [BAR 84] using the position of the nodes of the simplified mesh. Our implementation uses a 3x4x3 control point lattice for the FFD and uses Bernstein polynomials as the basis functions. The example in shows the grid, composed of the 36 green nodes. The weight values associated to the vertices, deformed by the FFD, are pre-calculated.

3.4. Layer 3 "Floating Cloth"

Layer 3 is composed of vertices that freely float around the body. This will take care of cases, such as a large skirt floating around the legs. Any part on this skirt can collide with any part of the leg. The simulation of this layer uses a classical approach with particle system and collision avoidance.

* *Particle system*: We use a simple mass-spring system. The simulation is performed using the Implicit Euler Integration proposed by Barraf et al. [BAR 98] and Volino et al. [VOL 00]. The garment is modelled by a simple mass-spring system. We consider two interactions: gravity and forces applied by the springs joining the particles.

* *Collision response*: Calculating collision detection between the cloth and the skin mesh would not be feasible in real-time, therefore we use a simplified model of the body. Given the assumption that the floating clothes are mainly skirts, the collision detection is calculated for the legs only and two cylinders model each leg. By simply calculating the distances between the skirt and the four leg segments, the collision detection can be performed. It is possible to further optimize by restricting the number of collision distances that we compute for each segment. During the pre-processing stage, a list of possible colliding vertices is defined for each segment. This list is defined by calculating the distance between the vertices and the legs and the normal orientation for each vertex on the skirt. The collision response consists of applying a kinematical correction to the position and velocity of the colliding vertices.

3.5. Computation Speed

Our method has been tested on several dressed virtual humans. Figure 7 illustrates how computational time changes with an increasing number of cloth triangles. The code was executed on a 1GHz PC with 512 MB RAM and a GeForce2 graphics card. The computational times do not include the deformation of the avatar skin and the real-time rendering. A simulation step corresponds to 0.04 seconds worth of real-time animation.

The deformation of Layer 3 is in average thirteen times slower than the deformation of the Layer 2. This is because Layer 2 uses an optimized model with a simplified collision detection method and no integration method. The deformation of Layer 1 is even faster because it uses skeletal deformation. The Layer 3 algorithm is able to simulate a maximum of 1,000 non-tessellated polygons in real-time. In most cases, this is sufficient to produce aesthetically pleasing results. Another advantage is that most of the clothes can be animated using Layer 2. Layer 3 is used in only a limited number of cases, such as large skirts or large trousers.

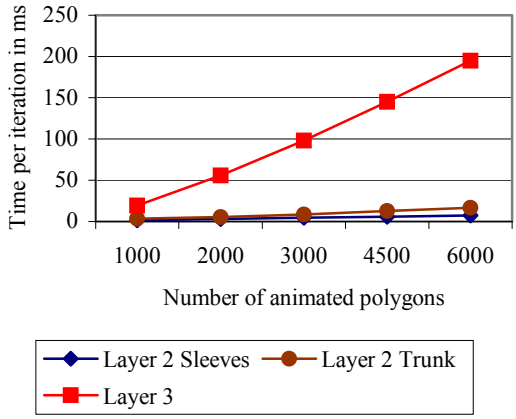


Figure 7: *Computing time per iteration versus number of triangles.*

4. Data-Driven Cloth Simulator

We present a data-driven method for simulating clothes worn by 3D characters in real-time. To effectively optimize the physics-based deformation, which is the bottleneck of the simulation, we use a coarse representation of the cloth mesh to drive the gross behaviour in simulation. We consider that the gross cloth behaviour is driven mainly by two separable contributions: the skeleton-driven movement of the character and the mechanical properties of the cloth. This consideration was partly inspired by the hybrid real-time simulation method proposed in the previous section [COR 02], where a hybrid deformation method is used to combine dynamic surfaces with skeleton-driven deformation (SDD). Unlike that method, however, our method exhibits significantly more efficient and realistic behaviour. This effect is achieved by focusing on the analysis of cloth movements in relation to its associated skin surface, and adopting a learning strategy. The idea is to use the analysis of the pre-simulated sequence to identify the region largely explained by joint movement and to replace the physics based simulation with geometric methods wherever possible.

In our approach, the key ingredients of the new technique are associated with different facets of cloth simulation: First, our novel collision detection prunes out unnecessary collision tests by tightly localizing potentially colliding regions through the analysis of the cloth movement in relation to the skeleton. Second, we use the pre-simulated sequence to approximate the dynamic behavior of the coarse mesh geometrically wherever possible. Finally, fine details such as wrinkles are also simulated in a data-driven manner, by using the pre-simulated cloth sequence as examples. Subsequently, real-time animation of fully dressed human could be generated, which would be suitable for applications such as games where visual plausibility is more important than accuracy.

4.1. Simulation of the Large-Scale Behavior

Due to the computational expenses of solving the full numerical system of the physics-based deformation, we seek simplifications by constructing a coarse mesh representation of the garment. The coarse mesh is used to deduce the large-scale behavior of the cloth in a data-driven manner, based on the input pre-simulated sequence. A number of optimization strategies are adopted: The two following sections describe a pre-processing that constructs and segments a coarse mesh representation into different region types. We then describe in the next two sections the spring-mass system and collision handling of the coarse mesh at each frame of the simulation. Also described is the runtime process.

4.1.1. Construction of the Coarse Mesh

We begin by constructing a coarse representation of the given cloth model that will drive the gross behavior of the simulated garment.

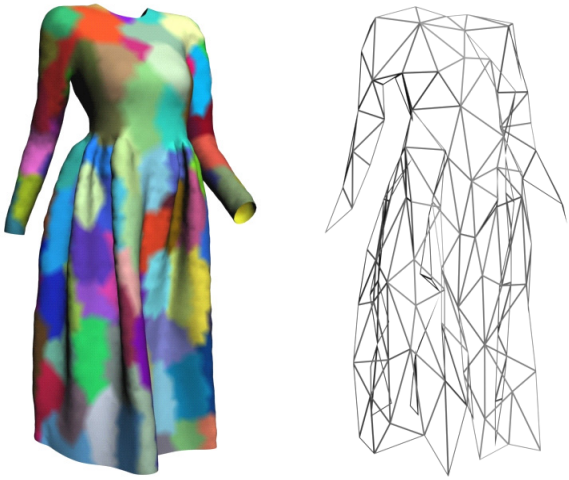


Figure 8: *The fine and the coarse mesh.*

It consists of two following steps: (1) The cloth surface is partitioned into a set of patches as shown in Figure 8. (2) A coarse mesh representation is obtained by combining a set of vertices in a patch into a single mass point located at the center.

4.1.2. Identifying Cloth-to-Joint Region Types

Next we carry out cloth-to-skin (or body) attachment through skin fitting, by which the skinning data on the cloth mesh are approximated in such a way that the skinning-driven cloth shape best fits the simulated cloth shape throughout the whole pre-simulated sequence. The basic idea is to use the pre-simulated results as examples and find the error-minimizing skin data through optimization (Figure 9). An optimization approach, such as the one presented by Mohr et al. [MOH 03], has been adopted here.

The residual values of the fitting provide useful information on how the garments behave in relation to the body. In our approach, three regions are identified from the residual values of the skin fitting process (Figure 10): those that potentially interact with several joints, those that are loosely attached to the skeleton and those that are rigidly attached to the skeleton.

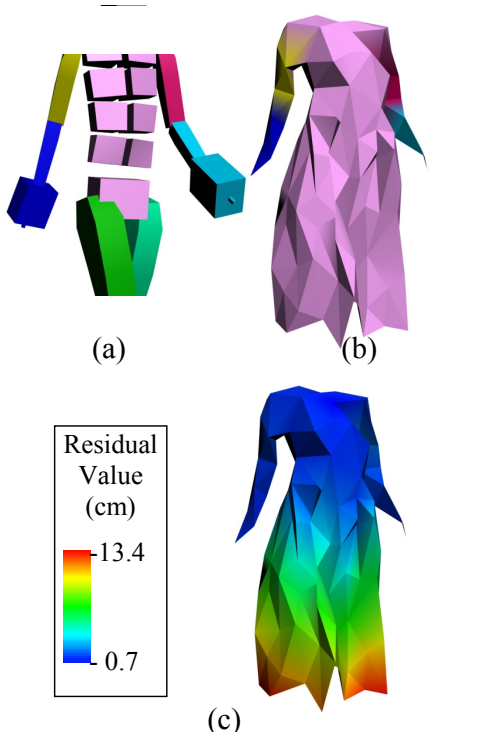


Figure 9: *(a) and (b) influence of the joints on the dress shown in colour, (c) quality of the fitting of the SDD data.*

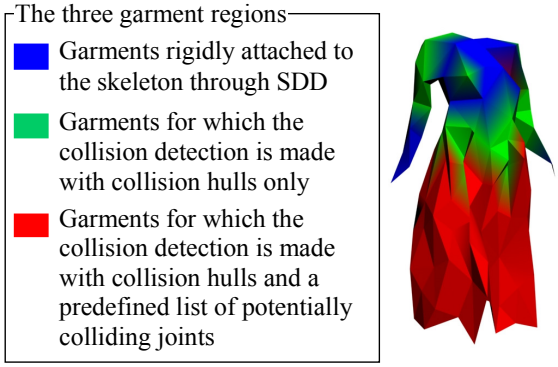


Figure 10: *The three regions computed on an analysis of the residual values.*

The deformation of tight regions is directly computed with the Skeleton Driven Deformation (SDD). The use of SDD for these regions makes it possible to reduce the number of mass points in the mass-spring system.

High residual values indicate much less dependency on a specific body region of the cloth movement. Therefore, an additional collision check is required to handle the interaction of the clothes with the whole body skeleton. A list of potentially colliding body patches is defined by selecting those that approach within a certain distance of the floating regions during the pre-simulated cloth sequence.

4.1.3. Data-Driven Post-Correction of the Coarse Mesh

At each frame of the simulation, we compute the coarse mesh by a mass-spring system with the implicit Euler numerical solver [BAR 98]. The simulation run on the

coarse mesh hardly reproduces the gross movement of the original cloth because the initial mesh has been significantly simplified (from 4000 to a few dozen vertices) and the topology has been modified. Moreover, unlike the simulator used for the pre-simulated cloth sequence, the simplified mass-spring model does not accurately simulate the bending and shearing properties of the fabrics [VOL 00].

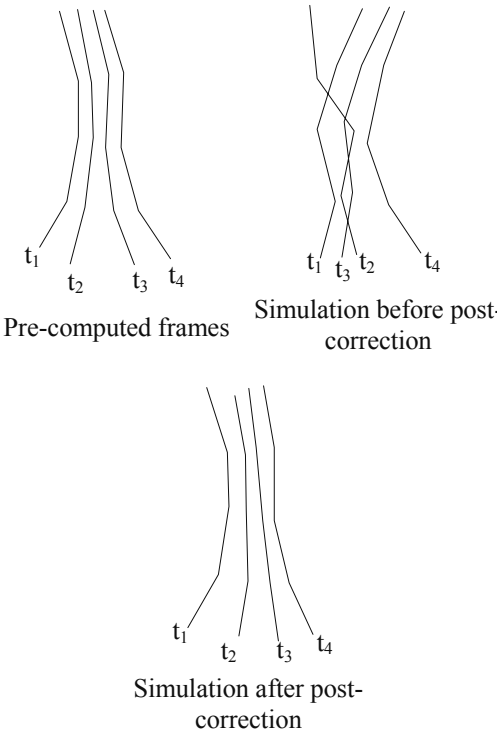


Figure 11: *Post-correction of the mass-spring system.*

We approach the problem by modifying the behavior of the mass-spring system through a fix-up process (similar to [MEY 00]) where the position and velocity of the coarse mesh vertices are modified in order to maintain the cloth shape as close as possible to the original one (Figure 11).

4.1.4. Collision Hulls

To prune unnecessary collision tests, we pre-compute what we term "collision hulls" that exploit the skin-to-cloth relation obtained from the pre-simulated sequence. These are built once at the beginning of the simulation (prior to the runtime simulation) after the SDD has been computed on the coarse mesh, using the pre-simulated sequence. At each pre-simulated frame, we calculate the difference between the SDD motion model and the pre-simulated cloth model in the local coordinate system of the SDD. After a sweep, we get a set of points that cover the path a patch takes during the simulation. The smallest convex hull (Figure 12 (b) and (c)) that contains all these points is generated for every patch using the "Quickhull" algorithm presented by Barber et al [BAR 96].

Collision handling at runtime consists of correcting the position of coarse mesh vertices after every simulation step so that they remain inside their respective hulls. Thus, collision detection returns to computing the inclusion of the particle in its associated hull; the Gilbert-Johnson-Keerthi algorithm [GIL 88] is ideally suited to this task. We used constrained dynamics [WIT 01] to handle the collision response (i.e. modification of position and velocity in response to collision detection). Collision detection is also computed between floating regions and skeleton joints.

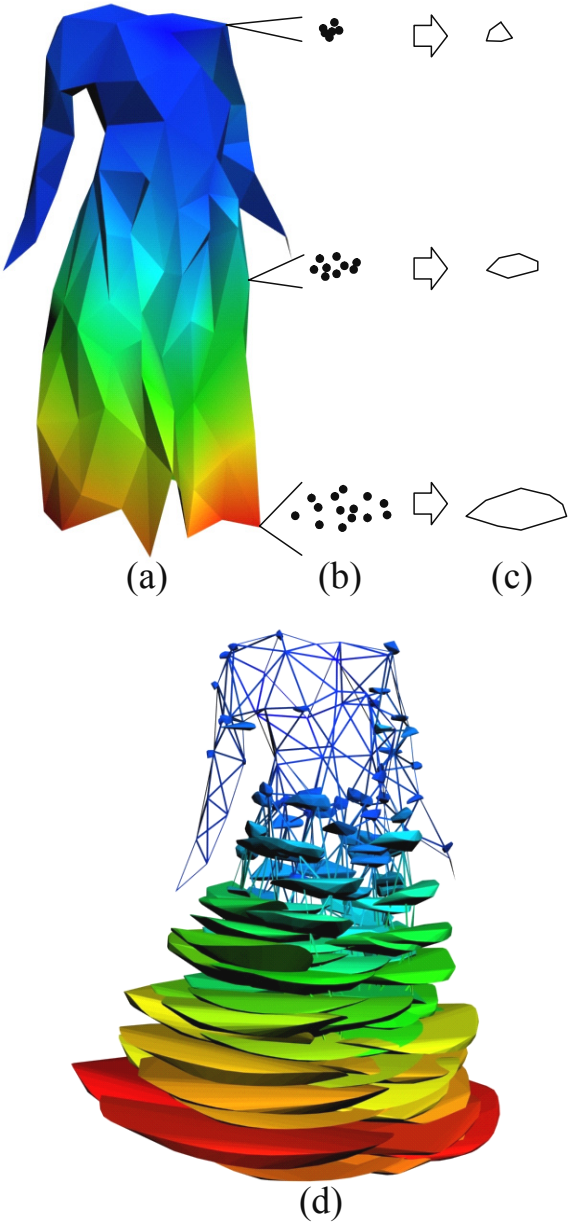


Figure 12: *Post-correction of the mass-spring system.*

4.1.5. Runtime Computations

The real-time computation of global cloth movements is obtained with a mass-spring system together with the collision response and post-correction described above. The runtime computation of the coarse mesh is obtained in the following order:

* Mass-spring computation

* Post-correction

* Collision response on hulls

* Collision response on floating garments

4.2. Generating Garment Details

So far we have shown the first part of our simulation, that is, the coarse level simulation. We continue now to describe the second part of the simulation, by which detailed cloth shape such as wrinkles or folds are depicted. Again, the main challenge here is obtaining the highest possible realism while maintaining acceptable computation load, in order to meet the real-time requirements.

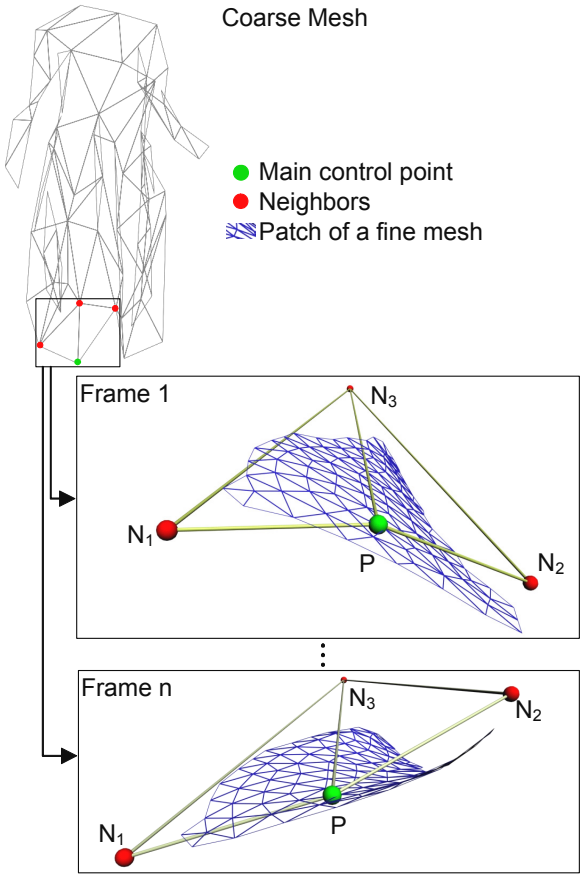


Figure 13: *Shape of the patch with respect to the positions of the control point P and its neighbours **N1**, **N2**, and **N3**.*

As recognized in earlier works [HAD 99] [KAN 01], wrinkles can be efficiently animated with a geometric method as they are geometric in nature. Unlike previous methods, however, our wrinkling function is not hand-drawn, nor geometrically approximated, but rather trained from on the analysis of the pre-simulated sequence.

In this work, we choose to represent the wrinkle displacement in the local coordinate system used for SDD. This makes our wrinkle parameterization invariant of all joints of higher hierarchy than the currently influencing joint.

Several techniques exist for shape interpolation using examples, such as RBF or parametric interpolation. We have used linear interpolation in which coefficients are defined by multi-linear regression on the pre-simulated animation, since it provides satisfactory results at a very low computation cost. For every vertex x in a patch, the interpolator function takes the associated mass point in the coarse mesh, and its neighbors as input. To calculate the position of x from the input, the wrinkle interpolator interpolates the positions of the coarse mesh points, weighted by coefficients determined the regression model on the pre-computed animation (Figure 13).

4.3. Results

Our simulator behaves fairly well on a wide variety of clothes, including those with highly stiff mechanical properties. Figure 14 show the pre-processing and run-time simulation results for a long dress, a skirt and underwear. Moreover, performance will increase due to the fact that the smallest number of triangles will be processed for the real-time rendering.

However, the method may introduce flaws in simulation for some tight clothes, due to the approximate handling of collision detection. For some body movements, the skin surface may slightly intersect the cloth surface. Similarly, the same problem may arise for self-collisions on clothes. This effect is particularly visible on the “Cocktail” dress that is made of two layers of tissues (see videos). The deletion of the skin triangles covered by the garment surface can partially correct this drawback.

Note that the cloth simulation is also restricted to clothes worn on bodies. While offering high computation speed, the cloth simulator cannot handle some cloth movements such as those appearing during dressing or undressing. More generally, the clothes are unable to interact with objects other than those that have been taken into consideration during the pre-processing phase. The list of objects that can potentially interact with clothes and the way these objects interact are defined at the pre-processing stage and cannot be changed during the real-time simulation. Finding a method to update the list of possible interacting objects automatically could be a subject for future research.

5. Case Study: Mixed Reality Simulation in Early Pompeii

The LIFEPLUS EU IST project proposes an innovative 3D reconstruction of ancient frescos-paintings through the real-time revival of their fauna and flora, featuring groups of virtual animated characters with artificial life dramaturgical behaviors, in an immersive AR environment. The goal of this project is to push the limits of current Augmented Reality (AR) technologies, exploring the processes of narrative design of fictional spaces where users can experience a high degree of realistic interactive immersion. Based on a captured/real-time video of a real scene, the project is oriented in enhancing these scenes by allowing the possibility to render realistic 3D simulations of virtual characters in real-time.

Since antiquity, images were used as records of both events-lifestyles, as well as decorations. The possibility of reviving them, adds a new dimension in understanding our past. However, the recreation of historic environments for serious study, education and entertainment is not new, Arnold [ARN 00], although the methods for achieving the objectives have evolved considerably over time. Before the days of widespread books and printing, story tellers would conjure up visions of events and places, providing their listeners with an impression of realities (often augmented realities) elsewhere in time and space. Theatre, fine art and cinema have added to the richness of the explicit visual experience available to the viewer. They have made the interpretations of history more accessible to the general public, but at the same time narrowing the individual’s

scope for personalized, interactive experience and visualization of the description of it. Historical frescos are a unique arrangement of "mise-en-scene" elements that enhance the user experience by creating a set of compelling narrative patterns, alas however in a static, two-dimensional way. Mixed Realities, Milgram and Kishino [MIL 94], and their concept of cyber-real space interplay invoke such interactive digital narratives that promote new patterns of understanding in various contexts. In the context of cultural heritage sites such as the ancient city of Pompeii, would like to observe and understand the behaviors and social patterns of living people from ancient Roman times, superimposed in the natural environment of the city. In industrial environments, we would like to 'employ' virtual workers for training and maintenance reasons, in order to assist real ones. Since recently, AR Systems had various difficulties to manage such simulations in a fully interactive manner, due to hardware & software complexities in AR enabling technologies, Azuma et al [AZU 01].

5.1. Developments in Augmented Reality

A number of projects are currently based on AR integrated platforms, exploring a variety of applications in different domains such as medical, ART [ART 04], cultural heritage, Stricker et al [STI 01] and Papagiannakis et al [PAP 02], training and maintenance, Schwald et al [SCH 01] and Wohlgemuth and Triebfürst [WHO 00], and games, Thomas et al [THO 00]. Special focus has recently been applied to system design and architecture in order to provide the various AR enabling technologies a framework, Gamma et al [GAM 94], for proper collaboration and interplay. Azuma, [AZU 01], describes an extensive bibliography on current state-of-the-art AR systems & frameworks. However, few of these systems take the modern approach that a realistic mixed reality application, rich in AR virtual character experiences, should be based on a complete VR Framework (featuring game-engine like components) with the addition of the "AR enabling Technologies" like a) Real-time Camera Tracking b) AR Displays and interfaces c) Registration and Calibration.

5.2. Augmented Reality Framework Components

Our AR platform is based on the VHD++, Ponder et al [PON 03], component-based framework engine developed by VRLAB-EPFL and MIRALab-UNIGE which allows quick prototyping of VR-AR applications featuring integrated real-time virtual character simulation technologies. The framework has borrowed extensive know-how from previous platforms such as presented by Sannier et al [SAN 99]. The key innovation is focused in the area of component-based framework that allows the plug-and-play of different heterogeneous technologies such as: Real-time character rendering in AR, real-time camera tracking, facial simulation and speech, body animation with skinning, 3D sound, cloth simulation and behavioural scripting of actions. To meet the hardware requirements of this aim, a single DELL P4 M50 Mobile Workstation was used, with a Quadro 4 500 GL NVIDIA graphics card, a firewire Unibrain Camera or USB Logitech web camera for fast image acquisition in a video-see-through TekGear monoscopic HMD setup, for advanced immersive simulation. Our previous efforts were based on a client-server distributed model, based on 2 mobile workstations. To achieve the requirement of 'true mobility', a single mobile workstation is used in our current demonstrations, after improvements in the streaming image capturing and introduction of hyper-threading in the platform code.

5.3. Real-time Markerless Camera Tracking

Real-time markerless camera tracking presents two main problems, namely the absence of easily recognisable markers and a demand for high-speed computation. Using markerless tracking is often a necessity as applying markers within the tracking area is not suitable or possible (especially on cultural heritage sites). This then requires accurate tracking to be done with only natural features present, often with sharply varying light sources, shadows, motion blur and occlusions. Performing this tracking in real-time necessitates the use of algorithms specially adapted to fast operation, and thus disqualifies many algorithms that are perfectly suitable in offline applications.

The LIFEPLUS application utilizes the 2D3 camera tracking solution, 2d3 [2D3 04], based on the approach that the system should be able to self-initialize anywhere within the tracking environment without any intervention from the user. In effect this means that instead of calculating relative changes in rotation and translation, we calculate absolute rotation and translation for every frame. This has the advantage of avoiding the problem of drift, and also ensures instant recovery after tracking was lost due to excessive motion blur or occlusion.

Figure 14: *Virtual human clothing.*

5.4. Simulating Ancient Pompeii

Our early results (the project started 1st March 2002) span across the frame of non-real-time mixed realities

simulations to real-time VR interactive experiences. In greater detail, we have tested our methodologies-plug-ins, SDKs and platform in the following 3 example scene simulations featuring behaviored virtual Pompeian characters. In Figure 5, the screenshots depict part of our test rendered sequences, where virtual clothes, hair, speech, body and facial animation are tested using 3DSMax and MIRALab dedicated virtual human plug-ins.

Figure 15: *At the Pompeian bakery.*

Bibliography

[ALE 02] ALEXA, M., "Linear Combination of Transformations", SIGGRAPH 2002 Conference Proceedings, Annual Conference Series, ACM Press, Vol. 21(3), pp. 380-387, 2002.

[ARN 00] ARNOLD, D.B., "Computer Graphics and Archaeology: Realism and Symbiosis", 2000, ACM SIGGRAPH and EUROGRAPHICS: Interpreting the Past, pre-conference proceedings, pp. 10-21.

[ART 04] ART: Augmented Reality for Therapy, http://mrcas.mpe.ntu.edu.sg/groups/art/

[AZU 01] AZUMA, R., BAILLOT, Y., BEHRINGER, R., FEINER, S., JULIER, S., MACINTYRE, B., 2001, "Recent Advances in Augmented Reality", IEEE Computer. Graphics Application, vol. 21, 6, pp 34-47.

[BAR 84] A. H. BARR. Global and local deformations of solid primitives. SIGGRAPH 84 Conference Proceedings, Annual Conference Series, pages 21–31. ACM SIGGRAPH, Addison Wesley, July 1984.

[BAR 98] BARAFF, D., AND WITKIN, A., "Large steps in cloth simulation", ACM Transactions on Graphics, Proceedings of ACM SIGGRAPH, ACM Press, pp. 43-54, 1998

[BAR 96] BARBER, C. B., DOBKIN, D.P., AND HUHDANPAA, H.T., "The Quickhull Algorithm for Convex Hulls", ACM Transactions on Mathematical Software, ACM Press, Vol. 22(4), pp. 469-483, 1996

[BUR 93] R. L. BURDEN, J. D. FAIRES, "Numerical Analysis, Fifth Edition", published by PWS Publishing, ISBN 0-534-93219-3, 1993.

[COR 02] CORDIER, F., MAGNENAT-THALMANN, N., "Real-time Animation of Dressed Virtual Humans", Eurographics, Blackwell publishers, pp 327 – 336, 2002.

[CHO 02] K.-J. CHOI, H.-S. KO, "Stable but Responsive Cloth", ACM Transactions on Graphics, Proceedings of ACM SIGGRAPH 2002, ACM Press, 21, pp. 165-172, 2002.

[DES 99] DESBRUN, M., SCHRÖDER, P., AND BARR, A. H. "Interactive Animation of Structured Deformable Objects", In Graphics Interface'99 proceedings, Morgan Kaufmann, pp. 1-8, 1999.

[FLA 88] PRESS, W. H., FLANNERY, B. P., TEUKOLSKY, S. A., AND VETTERLING, W. T., "Numerical Recipes in C, The art of scientific computing", Cambridge University Press, 1988.

[GAM 94] GAMMA, E., HELM, R., JOHNSON, R., VLISSIDES, J., 1994, "Design Patterns: Elements of Reusable Object-Oriented Software", ed. Addison-Wesley

[GIL 88] GILBERT, E. G. , JOHNSON, D.W., AND KEERTHI, S. S., "A fast procedure for computing the distance between complex objects in three-dimensional space", IEEE Journal of Robotics and Automation, published by IEEE Press, 4(2), pp. 193-203, 1988

[HAD 99] HADAP, S., BANGARTER, E., VOLINO, P., MAGNENAT-THALMANN, N., "Animating Wrinkles on Clothes", IEEE Visualization '99. San Francisco, USA, IEEE Press, pp. 175-182, 1999.

[JAM 03] JAMES D. L., AND FATAHALIAN, K., "Precomputing Interactive Dynamic Deformable Scenes", ACM Transactions on Graphics, ACM Press, Vol. 22(3), pp. 165-172, 2003.

[KAC 03] KACIC-ALESIC, Z., NORDENSTAM, M., BULLOCK, D., "A practical dynamics system", ACM SIGGRAPH/Eurographics Symposium on Computer Animation, ACM Press, pp. 7—16, Jul 2003.

[KAN 02] KANG, Y.-M., CHO, H.-G., "Bilayered Approximate Integration for Rapid and Plausible Animation of Virtual Cloth with Realistic Wrinkles", Computer Animation 2002, Switzerland, IEEE Press, pp. 203, 2002.

[KAN 01] KANG Y.-M., CHOI J.-H., CHO H.-G., LEE D.-H., "An efficient animation of wrinkled cloth with approximate implicit integration", The Visual Computer Journal, Spinger-Verlag, 2001.

[MEY 00] MEYER, M., DEBUNNE, G., DESBRUN, M., BARR, A. H., "Interactive Animation of Cloth-like Objects in Virtual Reality". Journal of Visualization and Computer Animation, John Wiley & Sons, 2000.

[MIL 94] MILGRAM, P., KISHINO, F., 1994, "A Taxonomy of Mixed Reality Visual Displays", IEICE Trans. Inf. Syst., vol. E77-D, 12, pp 1321-1329.

[MOH 03] MOHR, A., AND GLEICHER, M., "Building Efficient, Accurate Character Skins from Examples", ACM Transactions on Graphics, ACM Press, Vol. 22(3), pp. 165-172, 2003

[PAP 02] PAPAGIANNAKIS, G., PONDER, M., MOLET, T., KSHIRSAGAR, S., CORDIER, F., MAGNENAT-THALMANN, N., THALMANN, D., 2002, "LIFEPLUS: Revival of life in ancient Pompeii", VSMM2002, invited paper.

[PON 03] PONDER, M., PAPAGIANNAKIS, G., MOLET, T., MAGNENAT-THALMANN, N., THALMANN, D., 2003, "VHD++ Development Framework: Towards Extendible, Component Based VR/AR Simulation Engine Featuring Advanced Virtual Character Technologies", IEEE Computer Society Press, CGI Proceedings, pp. 96-104.

[PRO 95] PROVOT, X., "Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behavior", Graphics Interface'95 proceedings, A K Peters, pp 147-154, 1995.

[RUD 00] RUDOMIN, I., MELN, M., "Multi-Layer Garments Using Hybrid Models", Visual 2000, Springer, pp. 118, 2000.

[SAN 99] SANNIER, G., BALCISOY, S., MAGNENAT-THALMANN, N., THALMANN, D., 1999, "VHD: A System for Directing Real-Time Virtual Actors", The Visual Computer, ed. Springer, vol.15, 7/8, pp.320-329.

[SCH 01] SCHWALD, B., FIGUE, J., CHAUVINEAU, E., VU-HONG, F., 2001, "STARMATE: Using Augmented Reality technology for computer guided maintenance of complex mechanical elements", 2001: eBusiness and eWork, proceedings, vol.1, section 1.4.

[STI 01] STRICKER, D., DÄHNE, P., SEIBERT, F., CHRISTOU, I., ALMEIDA, L., IOANNIDIS, N., 2001, "Design and Development Issues for Archeoguide: An Augmented Reality based Cultural Heritage On-Site Guide", ICAV3D 2001: Augmented Virtual Environments and 3D Imaging, proceedings, pp. 1-5.

[TER 88] D. TERZOPOULOS, K. FLEISCHER, "Deformable Models", The Visual Computer, Springler-Verlag, Vol.4 (6), pp.306-331, 1988.

[THO 00] THOMAS, B., CLOSE, B., DONOGHUE, J., SQUIRES, J., DE BONDI, P., MORRIS, M., AND PIEKARSKI, W., ARQUAKE: "An Outdoor-Indoor Augmented Reality First Person Application", 2000, ISWC 2000: Wearable Computers, proceedings, pp. 139-146.

[VAS 01] VASSILEV, T., SPANLANG, B., "Fast Cloth Animation on Walking Avatars", Eurographics, Blackwell Publishers, 2001.

[VOL 00] VOLINO, P., MAGNENAT-THALMANN, N., "Virtual Clothing - Theory and practice", Springer Verlag, ISBN: 3-54067-600-7, Dec 2000.

[WIT 01] WITKIN A., "Constrained Dynamics", Physically-Based Modeling SIGGRAPH 2001 Course Notes.

[WHO 00] WOHLGEMUTH, W., TRIEBFÜRST, G., 2000, "ARVIKA: augmented reality for development, production and service", DARE 2000: Designing augmented reality environments, proceedings, pp. 151-152.

[ZHA 02] ZHANG, D., YUEN, M., "A Coherence-based Collision Detection Method for Dressed Human Simulation", Computer Graphics Forum, Blackwell publishers, Vol. 21(1), pp. 33-42, 2002.

[2D3 04] "2D3: The creator of the Boujou Software Tracker",: http://www.2d3.com/.