Advanced Topics in Virtual Garment Simulation Part 1

B. Thomaszewski[1] , M.Wacker[1,2] , and W.Straßer[1]

[1]WSI/GRIS, University of Tübingen, Germany
[2]University of applied sciences Dresden, Germany

Abstract

For more than two decades, cloth simulation has been an active research area in computer graphics. In order to create efficient high-quality animations, techniques from many research fields have to be thoroughly combined. The ongoing interest in this field is also due to the multidisciplinary nature of cloth simulation which spurs development and progress in collision detection, numerical time integration, constrained dynamics, or motion control, to name just a few areas. Beyond the very basic approaches, the complexity of the material can be daunting if no guidance is given. It is therefore the goal of this tutorial to provide the reader with an introduction and a guideline to the relevant matter. In order to provide a concise review, we will focus on advanced topics in cloth simulation, shedding light on both theoretical and practical aspects. This will pave the ground for those willing to implement a contemporaneous cloth simulation system as well as researchers who consider to start working in this area.

1. Introduction

The physical simulation of deformable objects is a central research area in computer graphics. Problems emanating from this field are usually complex to model and pose significant demands on computational resources. This is particularly true for the more specific case of cloth simulation. Due to the thin and flexible nature of cloth, it produces detailed folds and wrinkles, which in turn can lead to complicated self-collisions. In order to tackle this challenge within the requirements imposed by computer graphics, specialised methods have to be designed for concrete applications. These can roughly be divided into two categories: applications for which quality is most important and those for which speed is the urgent demand. To clarify this distinction, the requirement for time-critical cloth simulation could read: *given a fixed timing (e.g. 25 frames per second) optimize the (visual) quality of the simulation*. An analogous formulation can be stated for the first case. Because every category stands in its own right, we will address both of these fields in this tutorial.

Since cloth simulation has been an active research field for quite a while now, there is a broad variety of different approaches. One objective of this course is therefore to give the reader an introduction to thoughtfully chosen matter and a guideline towards practical applications. Another is to introduce algorithms and tools necessary for creating high quality animations. Finally, a further goal is to supply the audience with techniques for accelerating computations and eventually obtain fast simulations. The latter includes models specifically designed for computational efficiency. These methods can again be divided into two categories: algorithms which are optimized for sequential real-time computations and those which exploit the parallel potential of current hardware.

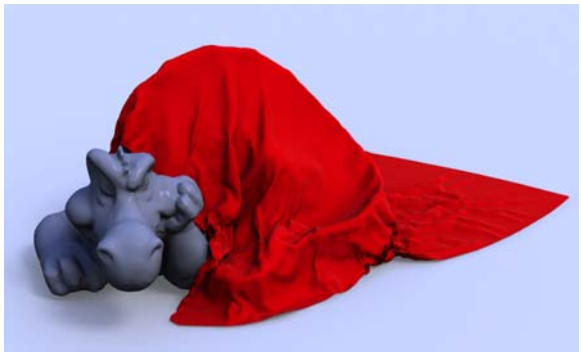

Figure 1: *The Eurographics 2007 Phlegmatic Dragon covered by a sheet of cloth.*

Because physically-based modelling has become the de

facto standard in cloth simulation, we will exclusively treat physically based methods in this tutorial and leave alternative approaches like geometry- or heuristics-based models aside.

1.1. Major Challenges in Cloth Simulation

Despite the comparably long history of cloth simulation this field can by no means be considered closed. For instance, the accurate modelling of real textile materials remains an open issue. Even if static properties can be measured quite accurately, textiles typically exhibit a high dynamical behaviour (e.g. hysteresis, damping, etc.) for which no accurate models are available yet. In the field of collision handling there is still room for improvement as well. Besides requirements like speed and robustness, error recovery is a point of particular practical relevance [VMT06a]. Furthermore, garment design and integration in CAD-like environments needs to be pushed forward if clothing simulation is to become a widely-used tool in industrial applications (see e.g. [CK05]). Likewise, electronic vending of clothing via the internet requires Virtual Try-on platforms specifically tailored to the individual needs of end customers. Finally, since high-quality animations can still be very time consuming, the need for designing faster techniques has not ceased. One way towards more efficient methods is to design, possibly from scratch, new algorithms that result in significant computational speed-up. Another is to develop or adapt algorithms which exploit parallelism on current hardware. In the course of this tutorial, we will address most of the aforementioned topics in great detail.

1.2. Overview

This tutorial assumes that the reader is already familiar with the basics of physically-based simulation. An overview of the state-of-the-art in this field is given in Sec. 2. The following sections of the first part are intended to supply the reader with the basics of contemporaneous cloth simulation. The third section gives an introduction to continuum mechanics and an example of numerical implementation. The way in which wrinkles and folds form depends mainly on the bending properties of the fabric. Because of its importance, a separate section (4) is devoted to this issue and practical ways of integrating bending into cloth simulation are discussed. In Sec. 5 we will describe how computations can be mapped onto parallel architectures. This includes generic modifications and extensions to existing algorithms in order to exploit potential parallelism. Both shared and distributed memory architectures will be considered, and we will address implementation related issues for both these settings.

The second part of these notes addresses the measurement of real world fabric parameters, including multi-layered textiles and seams, and their integration into cloth simulation. Furthermore, integrated Virtual Try-On applications are discussed, including models for parametrically deformable human bodies, body animation and motion retargeting as well as techniques for real-time cloth simulation.

2. State-of-the-Art in Virtual Clothing

In this section we will give an overview of the current state-of-the-art in virtual clothing. We will pay special attention to the topics covered in this tutorial. A more comprehensive summary of the evolution of this research can be found in [MTVW*05], Part 1.

2.1. Mechanical Models

For dynamically deformable surfaces, mass-spring systems [Pro95] and the more general particle systems [BHW94, VCMT95, EWS96] continue to be the most widely used simulation techniques in computer graphics. The popularity of mass spring systems is due to the ease of implementation and low computational costs. The accuracy offered by this method is, however, rather limited. As an example, simple homogeneous materials cannot be simulated consistently and the results highly depend on the specific mesh used in the simulation. If the reproduction of authentic material behaviour is desired (as, e.g., by the textile community), approaches based on continuum mechanics have to be used. Continuum-based approaches lead to a set of partial differential equations (PDEs), which have to be discretized in space and time. The spatial discretisation is usually carried out by means of finite differences (FDM) or finite element methods (FEM). Techniques based on finite elements and continuum mechanics (referred to as FE-approaches in the remainder) have not seen as much attention in cloth simulation as particle and mass spring systems. While we only mention the most relevant work here an extensive list can be found in [HB00]. Most of the existing FE-approaches are based on the geometrically exact thin shell formulation presented by Simo et al. [SFR89a]. Departing from the fully nonlinear theory, Eischen et al. [EDC96] proposed a cloth simulation method using quadrilateral, curvilinear elements. Because of the buckling behaviour of cloth, which can lead to divergence in the algorithm, an adaptive arc-length control is used. Etzmuss et al. [EKS03] presented a linear FE-approach based on the plane-stress assumption. Bending is treated separately from in-plane deformations and a co-rotational strain formulation is used to account for arbitrary rigid body transformations. The work presented in [TWS06] extends the concept of Subdivision Finite-Elements by Cirak et al. [COS00] to dynamic cloth simulation using a co-rotational strain formulation. As a middle course between simple mass spring systems and finite elements, Volino et al. [VMT05] proposed an accurate particle system which draws on notions from continuum mechanics but replaces the numerical discretisation by a more direct geometric formulation.

2.2. Numerical Time Integration

The mechanical model provides the means for computing internal forces due to fabric deformations. The dynamical evolution of the system (i.e., the trajectories of the nodes) is then determined by Newton's second law. In the discrete setting, the time dimension is decomposed into discrete time intervals and numerical integration methods are used to advance the system from a given state to its next state in time. Most generally, one distinguishes between two types of integration schemes: *explicit* methods compute the next state in time based on the current state derivatives, which are readily computed according to the mechanical model. Commonly used explicit integration methods are the second-order accurate Midpoint method used e.g. by Volino et al. [VCMT95] and the fourth-order accurate Runge-Kutta scheme used for instance by Eberhardt et al. [EWS96]. For computer graphics applications the numerical accuracy is usually less important than stability and robustness. As a well known fact, explicit schemes only provide conditional stability (see [HE01]). Since the differential equations resulting from cloth simulation are inherently stiff, explicit methods need small step sizes to guarantee a stable simulation. Consequently, computation times soon become excessive with increasing problem sizes. *Implicit* schemes do not suffer from this restriction since, in this case, stability is independent of the step size. Since the seminal work of Baraff et al. [BW98] implicit methods have therefore become predominant for physically-based simulations in computer graphics. Implicit methods include the unknown state of the system at the end of the time step in the update formula. Therefore, a system of (nonlinear) equations has to be solved in every time step. This process is one of the most time consuming parts in cloth simulation. Widely used representatives of this class are the first-order accurate backward Euler scheme [BW98] and the second-order BDF-2 scheme [HE01]. Recently, the Newmark family of integrators (including both explicit and implicit variants) found its way into computer graphics [BMF03, GH*03]. For a detailed overview and comparison of existing integration schemes and their efficiency for cloth simulation, the reader is referred to [VMT01] and [HE01].

2.3. Collision Handling

Besides the simulation of the mechanical properties of cloth the interaction with its environment has to be modeled as well. This involves the detection of any collisions and an adequate response to prevent the clothes from intersecting. The proper treatment of these two components (to which we refer as collision handling in the remainder) is a very complex task [THM*05]. While the physical simulation engine computes new states at distinct intervals only, collisions can occur at any instant in between such intervals. Algorithms based on continuous collision detection can handle these cases in a robust way, but are often very complex and time consuming. Therefore, the collision handling step is a major bottleneck in the simulation pipeline. Basically, detecting interference between two arbitrarily shaped objects breaks down to determining the interference between all primitives (i.e. faces, edges, and vertices) of one mesh with every primitive of the mesh representing the other object. With complex objects comprising thousands of faces, this naïve approach soon becomes too expensive due to its quadratic average run time with respect to the number of faces. A common way to accelerate the interference tests is to structure the objects under consideration hierarchically with bounding volumes. A bounding volume hierarchy (BVH) is then constructed for each object in the scene (including deformable as well as rigid objects) in a preprocessing step, using, for example, a top-down approach. In this case, a bounding volume enclosing the entire object is set as the root node of the tree representing the hierarchy. This node is then subdivided recursively until a leaf criterion is reached. Usually, the leaves contain one single primitive. Common choices for bounding volumes in cloth simulation are axis aligned bounding boxes (AABB) [van97, LAM01] or the more general discrete oriented polytopes (k-DOPs) [KHM*98, MKE03]. For treating self-collisions efficiently, it is necessary to adapt the general BVH algorithms to this special case. Measures related to the curvature of the surface can be used to quickly rule out flat, non-intersecting parts of the surface [VMT94, Pro97]. As another useful extension for self-collisions, continuous collision detection based on BVHs can be used, in which the exact contact points between two successive time steps are detected [BFA02].

Once the intersecting parts of a garment have been determined, an appropriate response has to computed in order to prevent the imminent intersections. A method well-suited for cloth simulation is the one presented by Bridson et al. [BFA02]. The essence of this method is to apply a stopping impulse to approaching triangles (i.e., to adjust their nodal velocities) whenever their distance falls below a certain threshold. However, even with such robust approaches there can be situations in which intersections cannot be prevented. An example for this are complicated multi-layer collisions or when cloth is pinched together due to character motion. In such cases special care is necessary in order to restore an intersection free state and to keep the simulation running [BWK03, VMT06a].

Despite the aforementioned acceleration techniques, collision handling remains a major bottleneck of cloth simulation. Recent developments aimed at further accelerating these algorithms by migrating computations to the graphics card [GKJ*05] or by exploiting parallel architectures [TB07, TPB07].

3. Mathematical and Physical Foundations

In this section, we will describe how a physically based model for deformable objects can be derived. As already pointed out in Sec. 2, mass-spring systems are still widely

used for cloth simulation in computer graphics. However, for authentic material mapping and hence realistic and reliable draping behaviour of cloth, as required e.g. by the textile community, one must necessarily resort to continuum mechanics. We will therefore restrict our considerations to methods based on continuum mechanics and begin with a brief review of the relevant foundations. As we will see, the central quantities in this theory are *strain* and *stress* which are related to each other via material or constitutive laws. We will only discuss some general material laws at this point and postpone more sophisticated models especially suited for textiles to the second part of these notes. Having established the governing equations, we will turn to an exemplary spatial discretisation.

3.1. Continuous Models

The structure of textiles or woven fabrics is clearly different from continuous media. However, modelling each single thread would certainly be an inefficient approach. We will instead approximate the garment geometrically with a polygonal mesh. If the regions of each polygon contains a sufficiently large number of threads (or weave periods) we can safely approximate the fabric as a continuous medium. Departing from a continuum model, we can derive a consistent spatial discretisation. Consistency here means that with increasing resolution the computed approximation converges to the actual solution of the continuous problem. This allows us to choose the spatial resolution in accordance to computational requirements without changing the properties of the cloth. In doing so, the dynamic motion of garments can be simulated efficiently and independently of discretisation throughout a broad range of resolutions. The first ingredient for such a continuum model is introduced in the next section.

3.2. Deformation Measures

In order to describe the equations that govern the (dynamic) behaviour of deformable objects, we consider the conditions that must hold in an equilibrium state. Generally speaking, a deformable object is in equilibrium if the internal forces due to deformation exactly cancel the external forces acting on its volume and boundary. The first step in this analysis is to compute deformation, which requires an appropriate measure.

A conceptional description of a deformable object embedded in three-dimensional is given by its *configuration mapping*

$$\varphi : \Omega \subset \mathbf{R}^3 \rightarrow \mathbf{R}^3 \ , \tag{1}$$

where Ω is its parameter domain. In dynamic scenes, where the object undergoes translation, rotation, and deformation this mapping also depends on time

$$\varphi : \Omega \times [0,\infty] \rightarrow \mathbf{R}^3 \ .$$

It is common to base descriptions of state and behaviour of deformable objects on an initial and a current configuration. For simplicity, the initial mapping is assumed to be the identity

$$\bar{\varphi}(x_1,x_2,x_3) = \varphi(x_1,x_2,x_3,0) = \mathrm{id} \ .$$

The configuration mapping can be given the interpretation of transforming positions of material points in the initial, undeformed state to corresponding positions in the current, deformed configuration (see Fig. 2).

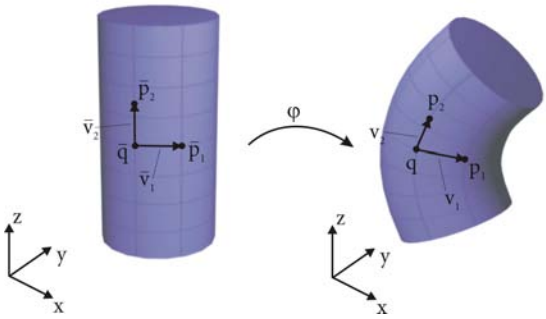

Figure 2: *Relative change of positions from undeformed to deformed configuration.*

For later derivations, it is convenient to express the current state in terms of a displacement field $\mathbf{u}$ from the initial configuration as

$$\varphi = \bar{\varphi} + u = id + u : \Omega \rightarrow \mathbf{R}^3 \ . \tag{2}$$

The displacement field itself is not an adequate measure of deformation, since invariance under rigid body motion such as translation is not given. A general deformation measure should capture the relative change between two elemental vectors in their initial and current configuration. Besides the obvious direct change in length, the angle formed by two vectors can change as well. One measure that takes both these characteristics into account is the scalar product. Consider the two vector pairs in the undeformed and deformed configuration in Fig. 2. The vectors are given by $\bar{\mathbf{v}}_i = \bar{\mathbf{p}}_i - \bar{\mathbf{q}}$, respectively $\mathbf{v}_i = \mathbf{p}_i - \mathbf{q}$. We can use a Taylor series expansion to express the vectors in the current configuration in an alternative form as

$$\begin{aligned} \mathbf{v}_i &= \varphi(\bar{\mathbf{q}} + \bar{\mathbf{v}}_i) - \varphi(\bar{\mathbf{q}}) \\ &= \varphi(\bar{\mathbf{q}}) + \nabla\varphi(\bar{\mathbf{q}}) \cdot \bar{\mathbf{v}}_i + O(\bar{\mathbf{v}}_i^2) - \varphi(\bar{\mathbf{q}}) \\ &\approx \nabla\varphi(\bar{\mathbf{q}})\bar{\mathbf{v}}_i = (\nabla u(\bar{\mathbf{q}}) - \mathrm{id})\bar{\mathbf{v}}_i \ . \end{aligned}$$

For later use, we define the deformation gradient F as

$$F = \nabla\varphi \ , \tag{3}$$

which can be given the interpretation of mapping vectors in the initial configuration to vectors in the current configuration. A general deformation measure can now be derived as

the difference of scalar products in the rest and current state:

$$\mathbf{v}_1 \cdot \mathbf{v}_2 - \bar{\mathbf{v}}_1 \cdot \bar{\mathbf{v}}_2 = \bar{\mathbf{v}}_1 \cdot (\nabla\varphi^T \nabla\varphi - id) \cdot \bar{\mathbf{v}}_2. \quad (4)$$

Using Eq. (2) we can identify from Eq. (4) the symmetric *Green* strain tensor as

$$\varepsilon_G = \frac{1}{2}(\nabla\varphi^T \nabla\varphi - id) = \frac{1}{2}(\nabla u^T + \nabla u + \nabla u^T \nabla u)\ . \quad (5)$$

An alternative expression can be obtained using the deformation gradient

$$\varepsilon_G = \frac{1}{2}(F^T F - id)\ . \quad (6)$$

For practical purposes, Eq. (5) can be written in indicial notation as

$$(\varepsilon_G)_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} + \sum_k \frac{\partial u_k}{\partial x_i}\frac{\partial u_k}{\partial x_j}\right)\ . \quad (7)$$

In this form, the entries of the strain tensor can be interpreted geometrically: diagonal components ε_{ii} measure the change in length in the direction of the x_i-axis and off-diagonal entries ε_{ij} measure the shearing between two axes. For small displacements, the nonlinear terms are negligible. This gives rise to the linear *Cauchy* strain tensor

$$\varepsilon_C = \frac{1}{2}(\nabla\mathbf{u} + \nabla\mathbf{u}^T)\ , \quad (8)$$

which is used in small strain analysis. Because of its linearity, the Cauchy tensor is very common in computer graphics. A closer look reveals that while being invariant under translations, rotational invariance is not given. In dynamic simulations, where the objects usually undergo large rotations, this is a severe restriction. A possibility to circumvent this problem is to extract the rotational part from the deformation field. The key observation is that, using polar decomposition, F can be written as

$$F = RU\ , \quad (9)$$

where R is a rotation and U is a pure deformation. The rotation can then be determined by finding the eigenvectors of $F^T F$. From this together with Eq. (6) it can also be seen that ε_G is invariant under rotations since

$$F^T F = U^T R^T R U = U^T U \quad (10)$$

due to the orthogonality of R. Once the rotation field R is known, the rotated linear strain tensor can be computed as

$$\varepsilon_{CR}(\varphi) = \varepsilon_C(R^T\varphi). \quad (11)$$

In the expressions derived so far we have implicitly assumed a Cartesian reference frame. We may think of garments or more generally deformable surface as 2-manifolds embedded in three-dimensional space. As such, a two-dimensional parametrisation of the surface is necessary to correctly establish differential measures like deformation. For instance, approaches based on thin shell formulations (see Sec. 4.4) rely on (parametrised) curvilinear systems. In such settings, all expressions will depend on partial derivatives of the parametrisation with respect to local coordinates. Although in a different way, common deformation measures are eventually recovered (see Sec. 4.4).

Strains in a deformable solid are accompanied by counteracting forces, which are the subject of the following section.

3.3. Internal Stress and Equilibrium

To describe the distribution of internal forces, we will first consider a differential surface element with area dA on a cross section of a deformable body as depicted in Fig. 3. Let $\mathbf{n}$ denote the normal of that element and let the resultant force acting on it be $d\mathbf{f}$.

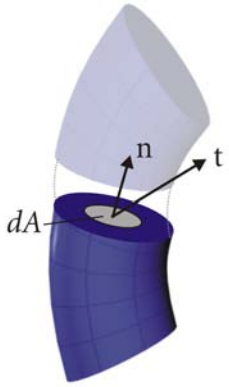

Figure 3: *Deformable solid in cross-sectional view. Traction vector* $\mathbf{t}$ *resulting from forces on area dA with normal* $\mathbf{n}$.

Then, the traction vector $\mathbf{t}$ is defined as

$$\mathbf{t} = \lim_{dA\to 0} \frac{d\mathbf{f}}{dA}\ .$$

Generalizing this expression for every normal direction leads to a mapping (or tensor) σ that maps every unit normal direction onto its traction vector,

$$\mathbf{t}(\mathbf{n}) = \sigma\mathbf{n}\ . \quad (12)$$

The tensor σ is called the *Cauchy* stress tensor. This symmetric tensor will be the subject of the next step on our way to the equilibrium equations.

Consider now a volume element V of a deformable body and let $\mathbf{f}$ be the body forces per unit volume acting on V. If we neglect forces due to inertia, translational equilibrium [†] implies that the sum of all forces acting on the volume V is

[†] as opposed to rotational equilibrium (see [BW97]p.103)

zero. Using equation (12), this can be expressed in integral form as

$$\int_{\partial V} \mathbf{t}\, da + \int_V \mathbf{f}\, dv = \int_{\partial V} \boldsymbol{\sigma}\mathbf{n}\, da + \int_V \mathbf{f}\, dv = 0\,, \qquad (13)$$

where $\mathbf{t}$ denotes the externally applied traction forces on the border ∂V. With the Gaussian divergence theorem, the surface integral of $\boldsymbol{\sigma}\mathbf{n}$ can be transformed into a volume integral,

$$\int_{\partial V} \boldsymbol{\sigma}\mathbf{n}\, da + \int_V \mathbf{f}\, dv = \int_V (\operatorname{div}\boldsymbol{\sigma} + \mathbf{f})\, dv = 0\,. \qquad (14)$$

Since this must hold for any enclosed volume, the (point wise) equilibrium equation of a deformable solid follows as

$$\operatorname{div}\boldsymbol{\sigma}(\mathbf{x}) + \mathbf{f}(\mathbf{x}) = 0\,. \qquad (15)$$

For dynamic simulation this equation has to be augmented by terms accounting for intertial forces. Furthermore, $\boldsymbol{\sigma}$ will also include viscous stress contributions due to material damping, leading to

$$\operatorname{div}\boldsymbol{\sigma}(\mathbf{x}, \dot{\mathbf{x}}) + \mathbf{f}(\mathbf{x}) = \rho\ddot{\mathbf{x}}\,. \qquad (16)$$

Here, ρ is the mass density and $\mathbf{x}$ denotes the vector field of positions. This continuous formulation is the starting point for numerical treatment. In Sec. 3.6 we will look at an exemplary spatial numerical discretisation. Before we continue with constitutive laws in Sec. 3.5 we will take a look at a hitherto neglected aspect of deformation.

3.4. Bending

As we have seen in the previous sections, the deformation modes in general (three-dimensional) continuum mechanics can be separated into stretching and shearing (cf. Eq. 7). These two deformation modes are *orthogonal* to each other: pure stretching does not lead to shear deformation and vice versa. In the special case of thin deformable surfaces, we intuitively identify *bending* as a further deformation mode. The question arises as to whether for bending deformation an analogous orthogonality relation holds. We will pursue this issue in the following.

The concept of bending cannot be derived from the three-dimensional, point wise view of continuum mechanics, since this standpoint is indifferent of *shape*. If we consider for example an elastic ball it is immediately clear that we cannot *bend* such an object. The same holds for other objects that do not exhibit a direction with significantly smaller size. If there is, however, such a direction like in the case of a thin plate or an elastic rod, bending deformations become possible.

As we can see from these examples, the ability to bend an object is closely related to its shape, or more precisely, to the proportion of its extents in the different dimensions. The ball is perfectly isotropic and has no intrinsic orientation. Therefore, the choice of a reference frame is arbitrary. While the geometry of the cube furnishes an intrinsic coordinate system, none of the dimensions is accentuated above

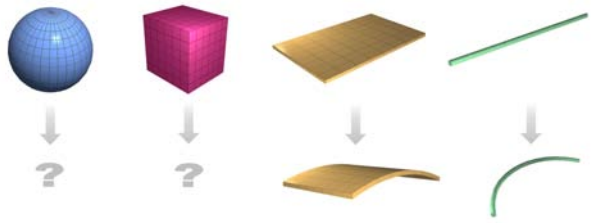

Figure 4: *Whether an object can be bent depends mainly on its shape. It is not clear how to bend a sphere or a cube. For a thin plate or a rod the notion of bending is intuitively clear.*

the others. In the example of the plate however, one direction can be distinguished, in which the lengths are clearly inferior to those in the orthogonal directions. The same holds for the case of a thin flexible rod. Note that the intuitive bending deformation of thin objects like the plate in the example is such that it causes as little in-plane deformation as possible. Hence, we may assume a bending mode which is orthogonal to the deformation modes of stretching and shearing. For the moment, we can most generally identify bending deformation of an embedded manifold as a change in curvature. We will return to this issue in a more specific discussion in the following paragraphs. In analogy to strain associated with stretching and shearing (to which we collectively refer as *membrane*-strains) we define the bending strain $\gamma_{\alpha\beta}$, $\alpha,\beta \in \{1,2\}$, where the components $\gamma_{\alpha\alpha}$ are normal curvatures in the directions of surface coordinates and $\gamma_{\alpha\beta}$, $\alpha \neq \beta$, is the torsional component.

3.5. Constitutive Relations in Linear Elasticity

The previous section led to the formulation of equilibrium equations involving the stress tensor $\boldsymbol{\sigma}$. In general, the relationship between stress and strain can be of high complexity. Here, we will focus on situations where stresses in a body depend only on its current state of deformation. Materials that fulfil this requirement are called *hyperelastic*. Generally, for two different types of elastic material, the same deformation will result in different stresses in terms of magnitude and possibly direction. This relationship between stress and strain is described by the elasticity tensor $\mathcal{C}$.

In the simplest case, the relationship between stress and strain is linear. Note that the term *linear* appears in two different contexts. First, there is a geometric relationship between displacement and strain. A linear strain-displacement assumption results in the Cauchy strain tensor, as described above. Second, the dependence between strain and stress itself can be either linear or nonlinear. We will start with a linear material law, for which the stress tensor can be written as

$$\boldsymbol{\sigma} = \mathcal{C} : \boldsymbol{\varepsilon}\,, \qquad (17)$$

where tensorial notation was used. This formulation accounts for the fact that a component ε_{ij} can potentially have influence on every entry σ_{kl} of the stress tensor. Using the summation convention, the stress tensor can conviniently be expressed as

$$\sigma_{ij} = \mathcal{C}_{ijkl}\varepsilon_{kl} \ . \tag{18}$$

Therefore, the entry $\mathcal{C}_{ijkl}$ can be interpreted as linking σ_{ij} to ε_{kl}.

As the simplest model, a linear-elastic isotropic material is governed by only two independent constants. In this case, the elasticity tensor[‡] $\mathcal{C}$ reads

$$\mathcal{C}_{ijkl} = \lambda\delta_{ij}\delta_{kl} + 2\mu\delta_{ik}\delta_{jl} \ , \tag{19}$$

where λ and μ are the *Lamé* constants (cf. [BW97]). These constants can be used to express the well-known Young modulus E and the Poisson ratio ν as

$$E = \frac{\mu(3\lambda+2\mu)}{\lambda+\mu}, \qquad \nu = \frac{\lambda}{2(\lambda+\mu)} \ . \tag{20}$$

Fig. 5 illustrates the meaning of these constants. In this example, a simple elastic beam is subjected to a longitudinal loading along the x-axis. This loading leads to a deformation $\varepsilon_{xx} = \frac{l-l_0}{l}$ counteracted by the stress σ_{xx} in the same direction. In linear small strain elasticity, these quantities are related by

$$\frac{\sigma_{xx}}{\varepsilon_{xx}} = E \ .$$

In addition to the extensional strain in the direction of the

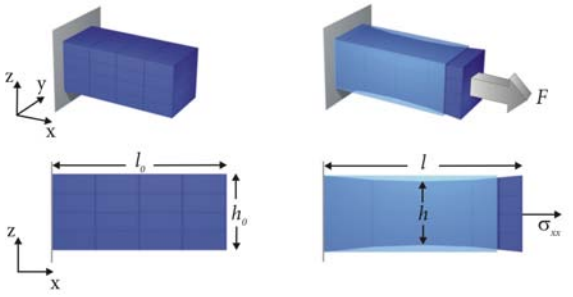

Figure 5: *An elastic beam is subjected to a horizontal loading in x direction. Longitudinal as well as transverse deformation can be observed.*

loading, we can usually also observe deformations $\varepsilon_{yy} = \varepsilon_{zz} = \frac{h-h_0}{h}$ in the orthogonal directions. Since this transverse contraction can be completely attributed to the axial loading, we can write

$$\varepsilon_{yy} = \varepsilon_{zz} = \frac{\nu}{E}\sigma_{xx} \ ,$$

(see also [Hau04]). For physical realism, ν has to be such that $0 \leq \nu \leq 0.5$. In the case of textiles, ν is usually smaller than 0.3.

The material model described above corresponds to Hooke's law in three dimensions. Because of its linearity and easy implementation, it is widely used in computer graphics. In order to model the basic in-plane properties of textiles, a more general material law is needed. In two dimensions, stress and strain are described by 2×2 tensors and the elasticity tensor consists of 16 components. Symmetry considerations[§] imply that

$$\mathcal{C}_{ijkl} = \mathcal{C}_{klij} \ ,$$

as well as

$$\mathcal{C}_{ijkl} = \mathcal{C}_{jikl}, \qquad \mathcal{C}_{ijkl} = \mathcal{C}_{ijlk} \ .$$

This results in an elasticity tensor of the form

$$\mathcal{C} = \begin{bmatrix} \mathcal{C}_{1111} & \mathcal{C}_{1112} & \mathcal{C}_{1122} \\ & \mathcal{C}_{1212} & \mathcal{C}_{1222} \\ \text{sym.} & & \\ & & \mathcal{C}_{2222} \end{bmatrix} \ ,$$

where components determined by symmetry have been omitted. This tensor is minimal in the sense that, assuming complete anisotropy, it cannot be further reduced. Hence, we can identify at most six independent constants describing an anisotropic, linear elastic material in two dimensions. For implementation purposes, vectorial notation is more convenient. As an example, the symmetric 2×2 stress tensor can be written as a 3-vector,

$$\sigma = \begin{bmatrix} \sigma_{xx} & \sigma_{yy} & \sigma_{xy} \end{bmatrix}^T \ .$$

An analogous elasticity matrix can be derived from the elasticity tensor (see e.g. [Bel00]) as

$$\mathbf{C} = \begin{bmatrix} \mathcal{C}_{1111} & \mathcal{C}_{1122} & \mathcal{C}_{1112} \\ & \mathcal{C}_{2222} & \mathcal{C}_{1222} \\ \text{sym.} & & \mathcal{C}_{1212} \end{bmatrix} . \tag{21}$$

As an example for textile simulation, one could use four of these constants: $\mathcal{C}_{1111}$ and $\mathcal{C}_{2222}$ are readily related to Young moduli E_u and E_v in the fabric's yarn directions *weft* and *warp*, which do not necessarily coincide with the coordinate directions x and y. Furthermore, $\mathcal{C}_{1212}$ is related to a shear modulus G and $\mathcal{C}_{1122}$ to transverse contraction coefficients ν_u and ν_v. With this material model, the stress-strain relationship can be written in matrix form as

$$\begin{bmatrix} \sigma_u \\ \sigma_v \\ \sigma_{uv} \end{bmatrix} = \frac{1}{1-\nu_u\nu_v} \begin{bmatrix} E_u & \nu_u E_v & 0 \\ \nu_v E_u & E_v & 0 \\ 0 & 0 & G(1-\nu_u\nu_v) \end{bmatrix} . \tag{22}$$

[‡] In this context, the formulation in terms of tensors is common but basically indicial, tensorial, and matrix notations are equivalent (see also [Bel00]).

[§] The elasticity tensor can mathematically be derived in terms of second order partial derivatives of a stored strain energy function (see [BW97]). The symmetry follows from the commutability of the mixed partial derivatives.

In a similar way to Eq. (22), the bending stress τ can be related to the bending strain γ as

$$\tau = \mathbf{C}_{bend}\gamma\,, \tag{23}$$

where $\mathbf{C}_{bend}$ models the bending properties of a specific material (see e.g. [VMT05]).

Because of its linearity and easy implementation, the material model described above is widely used for garment simulation. Beyond the small deformation range, the behaviour of most materials cannot reasonably be approximated by a linear model. In part 2 of the tutorial advanced constitutive laws are discussed, which capture fabric behaviour more accurately.

3.6. Spatial Discretisation with Linear Finite Elements

Having discussed constitutive models, we can now proceed with the discretisation of the governing equilibrium equations (14), respectively (15). We will use a formulation based on linear finite elements as an illustrative example [EKS03].

The finite element method is a procedure for the numerical solution of partial differential equations. The origins of this method can be found in structural mechanics, where the distribution of strains and stresses throughout a body in static equilibrium is sought. For the sake of brevity, we will not describe in detail how the final discrete equations used for implementation are derived. Instead, we will depart directly from the discrete setting. The reader interested in detailed derivations is referred to the standard text books [ZT00a] and [Bat96].

Assuming a decomposition of the domain into disjoint elements, the problem under consideration is formulated in terms of a displacement approach. This means, that for every node defined by the geometric decomposition, a displacement vector from its initial position is to be determined, such that the resulting final configuration is a solution to the elasticity problem. This process derives from a *variational principle*, the so called virtual work equation, which is mathematically rooted in variational calculus. In such a principle, an energy functional $\Pi = \Pi(\mathbf{u})$ is sought to be rendered stationary, in this case with respect to nodal displacements $\mathbf{u}$. In static elasticity problems, this functional is the sum of potential and strain energy[¶]. The functional is stationary if

$$\delta\Pi(\mathbf{u}) = 0\,, \qquad \text{for all variations } \delta\mathbf{u}\,. \tag{24}$$

From this, the (dynamic) virtual work equation can be derived as

$$\int_\Omega \delta\varepsilon : C(\varepsilon)d\Omega - \int_\Omega \delta u\; f\; d\Omega + \int_\Omega \delta u\; \rho\; \ddot{u}\; d\Omega = 0\,. \tag{25}$$

Here, the first term is due to the internal strain energy, the second one accounts for body forces per volume, and the last term represents inertia forces. This equation forms the basis for the subsequent finite element discretisation. An important observation on the way towards the discrete formulation is that, using further mathematical transformations, the virtual work equation per element can be expressed alternatively in terms of equivalent internal and external nodal forces. If we assume a linear relationship between displacement and strain, as well as between strain and stress, the problem can finally be expressed in the form of

$$\mathbf{K}\mathbf{u} = \mathbf{f}\,. \tag{26}$$

The matrix $\mathbf{K}$ is called the *stiffness matrix* and $\mathbf{f}$ is the vector of external forces per node. Visually, this equation states that in an equilibrium state the displacement of the nodes is such that the equivalent internal nodal forces (resulting from internal stress) exactly cancel the external forces. For dynamic problems, this equation has to be augmented by terms accounting for inertia and viscous forces. This results in a second order initial value problem which reads

$$\mathbf{M}\ddot{\mathbf{u}}(t) + \mathbf{D}\dot{\mathbf{u}}(t) + \mathbf{K}\mathbf{u}(t) + \mathbf{f} = 0\,, \tag{27}$$

where $\mathbf{M}$ is the mass matrix and $\mathbf{D}$ is the viscosity matrix. In this equation, the initial conditions $\mathbf{u}(0) = \mathbf{u}_0$ and $\dot{\mathbf{u}}(0) = \mathbf{v}_0$ are assumed, where $\mathbf{v}_0$ denotes the initial velocity. Note that the vector of nodal displacements $\mathbf{u} = \mathbf{u}(t)$ now depends on time. In the following, we will explain how to set up the stiffness matrix for a practical example, namely the plane stress analysis of two-dimensional elasticity. This particular case can be used as a basis for a continuum mechanics-based cloth simulator.

3.6.1. Plane Stress Analysis

Two of the most compelling advantages of the finite element method are its modularity and versatility. Once a general framework for the method has been laid out it can be applied to a broad range of problems. The specialisation on the actual problem follows through decisions on additional properties like material laws and, what is most important, the choice of an actual element type. As a concrete example, we will investigate the special case of plane stress analysis in this section. Being the simplest candidate, the linear triangular element with nodal displacement as only degrees of freedom will be used.

Plane stress analysis can be applied to elasticity problems that are inherently two-dimensional, i.e. where the in-plane deformation is predominant. This restricts the range of problems to settings which are essentially *flat* or which can at least be reasonably approximated with flat elements. Nevertheless, it is possible to use the plane stress assumption as a basis for cloth simulation (see [EKS03]). In the following, we will, for simplicity, assume that the object under consideration lies in the *xy*-plane. As the name already indicates, in the case of plane stress, the out-of-plane components of the stress tensor are zero, i.e. $\sigma_{iz} = \sigma_{zj} = 0$.

The geometry of the linear triangular element is given by

[¶] The extension to kinetic energy is omitted here for simplicity.

the coordinates of its three points $\mathbf{p}_1$, $\mathbf{p}_2$ and $\mathbf{p}_3$ (cf. Fig. 6). The three linear nodal shape functions are completely

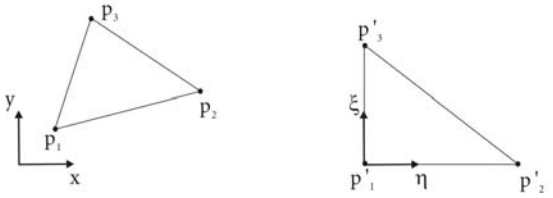

Figure 6: *Geometry of a triangular element (*left*) and its corresponding generic element (*right*).*

defined by the requirement

$$N_i(\mathbf{p}_j) = \delta_{ij} \ .$$

Further, the approximation of the displacement field over the element is uniquely defined by the nodal in-plane displacements $\tilde{\mathbf{u}}_x$ and $\tilde{\mathbf{u}}_y$ in the x and y-direction as

$$\mathbf{u} = \sum_i N_i \tilde{\mathbf{u}}_i .$$

The definition of the shape functions, which are depicted in Fig. 7, automatically ensures displacement continuity across elements. For this simple element, an explicit expression for

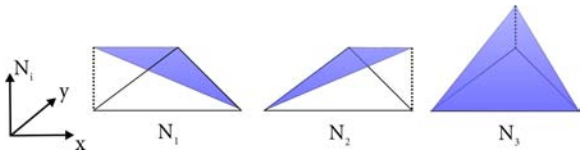

Figure 7: *The three linear shape functions of a triangular element.*

the shape functions in terms of the Cartesian coordinate can readily be derived. Nevertheless, it is instructive for the general case to take another approach. Notice that every triangular element can be transformed to a generic element as shown in Fig. 6. In this local space with coordinates ξ and η, the shape functions are trivially given by

$$N_1 = 1 - \xi - \eta, \quad N_2 = \xi, \quad N_3 = \eta \ .$$

Likewise, it can easily be verified that the shape function derivatives with respect to the local coordinates follow as

$$\frac{\partial N_1}{\partial \vartheta} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \quad \frac{\partial N_2}{\partial \vartheta} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \frac{\partial N_3}{\partial \vartheta} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \ ,$$

where $\vartheta = \begin{bmatrix} \xi & \eta \end{bmatrix}^T$ denotes the vector of local coordinates. For the subsequent formulations, the shape function derivatives with respect to the Cartesian coordinates are required. These are obtained by use of the chain rule as

$$\frac{\partial N_i}{\partial x_j} = \frac{\partial N_i}{\partial \xi}\frac{\partial \xi}{\partial x_j} + \frac{\partial N_i}{\partial \eta}\frac{\partial \eta}{\partial x_j}.$$

In practice, the necessary computation of

$$\left(\frac{\partial x}{\partial \vartheta}\right)^{-1}$$

can be accomplished without difficulty. As can be seen in the following, the shape function derivatives are indeed the only quantities actually needed in computation. Hence, there is no need for explicitly deriving the shape function expressions.

The next stage consists in formulating strain. In the case of moderately small deformations, a linear strain definition in terms of the displacement field is common. Over a single triangular element, the Cauchy strain is defined as

$$\varepsilon = \begin{bmatrix} \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{bmatrix} \mathbf{u} = \sum_{i=0}^{3} \begin{bmatrix} \frac{\partial N_i}{\partial x} & 0 \\ 0 & \frac{\partial N_i}{\partial y} \\ \frac{\partial N_i}{\partial y} & \frac{\partial N^i}{\partial x} \end{bmatrix} \tilde{\mathbf{u}}_i = \sum_{i=0}^{3} \mathbf{B}_i \tilde{\mathbf{u}}_i \qquad (28)$$

Once the strain is computed for an element, the stress follows by a simple linear relation (see Sec. 3.5).

Stiffness Matrix The point of departure for setting up the stiffness matrix is the elemental virtual work equation. Using some basic transformations, the integral term leading to the global stiffness matrix can be written in terms of elemental contributions as

$$\int_\Omega \mathbf{B}^T \mathbf{C}\mathbf{B}\mathbf{u} \ d\Omega = \sum_i \int_{\Omega_i} \mathbf{B}^T \mathbf{C}\mathbf{B}\mathbf{u}_i \ d\Omega_i \ ,$$

provided that $\Omega = \cup_i \Omega_i$ and $\Omega_i \cap \Omega_j = \emptyset$ for all $i \neq j$ holds. Hence, the stiffness matrix can be assembled in an element-wise manner

$$\mathbf{K}_{ij} = \sum_e \mathbf{K}^e_{ij}$$

where $\mathbf{K}^e_{ij}$ is the contribution of element e to the the entry of the global stiffness matrix linking nodes i and j. In the geometrically linear approach, the involved matrices $\mathbf{B}_i$ from equation (28) are constant over one element. Together with a linear elastic material law the above integral reduces to

$$\mathbf{K} = \sum_i \int_{\Omega_i} \mathbf{B}_i^T \mathbf{C}\mathbf{B}_i \ d\Omega = \sum_i \mathbf{B}_i^T \mathbf{C}\mathbf{B}_i t A_i$$

where t is the constant material thickness and A_i is the area of a triangular element.

Practical Considerations With the methods presented so far only static analysis is possible. However, the extension to dynamic simulation is not difficult as it consists mainly of the addition of inertia and viscous forces already mentioned in Eq. (16). The actual way in which this is accomplished depends on the actual numerical time integration scheme. It is worth noting that the stiffness matrix automatically supplies a means of evaluating internal forces in the current configuration. In this case a simple matrix-vector multiplication is sufficient. However, if elastic forces are not integrated implicitly (as in explicit schemes or certain variants

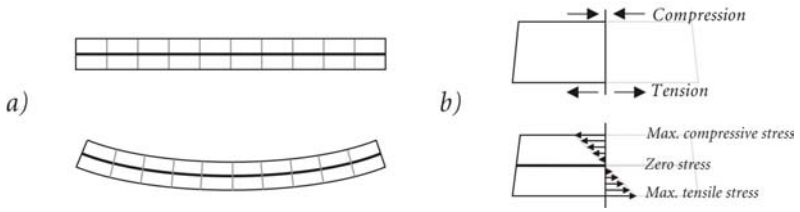

Figure 8: *Cross-sectional view of bending deformation with finite thickness.* a) *Bending leads to a combination of compressive (top layer) and tensile (bottom layer) deformations. The neutral axis (shown in bold) remains unstretched.* b) *Linearly varying stresses through the thickness imply the existence of a zero-stress axis, the* neutral axis*.*

of the Newmark algorithm [BMF03]), they can be computed directly without having to assemble the matrix itself.

4. The Importance of Bending in Cloth Simulation

Wrinkles and folds play an important role in the appearance of real textiles. The way in which they form depends mainly on the bending properties of the specific material type. While a remarkable amount of effort has been spent on precisely reproducing the in-plane forces, few existing models are concerned with an accurate and consistent way of modelling bending energy. Nevertheless, the characteristic folding and buckling behaviour of cloth highly depends on bending properties. The range of existing approaches to modelling bending is broad and we will investigate the most relevant methods in this section. From the field of engineering, the thin plate or thin shell equations, which will also be introduced in the following, are known to be an adequate approach to this problem. The associated minimisation problem includes fourth order derivatives with respect to the displacements, which in most of the frameworks are not readily computable. Therefore, a direct implementation of such energy-based methods is often avoided. However, many methods draw their inspiration from the theory of elastic beams or plates and we will therefore start with a review of the relevant matter.

4.1. Bending with Finite Thickness

Let us consider a thin plate and assume, for the moment, that the plate is cylindrically bent. In this case, we can – without loss of generality – restrict our investigations to a thin slice of the plate. Thus, we arrive at a geometry corresponding to the classical beam element (see e.g. [ZT00a]). In the cross-sectional view shown in Fig. 8 it can be seen that the bottom layer is stretched while the top layer has been compressed (cf. [Kee99]). We can reasonably assume that the maximum values of tension and compression occur on the boundary layers. If we further assume that the induced stresses vary monotonically between these maxima we arrive at an axis with zero stress, the so called *neutral axis* (see Fig. 8). is the primary parametrisation domain. In the following sections we will see that the theory of elastic beams and plates is essentially based on this dimension reduction.

4.2. Linear Elasticity of Beams and Plates

The simplest model corresponding to our interests is the one-dimensional, linear elastic beam. As we will see in section 4.3, many existing approaches to bending in cloth simulation rely on this model. In the corresponding elasticity problem, the central unknown is the lateral deflection w of the neutral axis. We begin the description by introducing the kinematic constraints which lead to the common model of the *Euler-Bernoulli* beam. In this model, the *Kirchhoff Assumptions* are used, i.e. lines that are initially normal to the neutral axis are supposed to remain straight (i.e. they do not bend), normal to the neutral axis, and unstretched. The deformed state of the beam can be described by the displacements u_0 and w_0 of the neutral axis and a rotation θ of the normal (see Fig. 9).

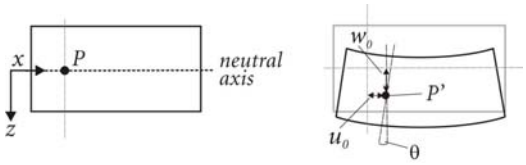

Figure 9: *Displacements u_0 and w_0 of the neutral axis and cross-sectional rotation θ for a deformed beam element.*

The horizontal and vertical displacements of any material point in the beam are given by

$$u(x,z) = u_0(x) - z\theta(x), \quad w(x,z) = w_0(x) \ ,$$

with the strains

$$\varepsilon_x = \frac{\partial u}{\partial x} = \frac{\partial u_0}{\partial x} - z\frac{\partial \theta}{\partial x} \ .$$

Because normal lines are assumed to remain unstretched, the strain ε_z in this direction can be neglected. Further, the requirement that normal lines remain perpendicular to the neutral axis implies

$$\theta = \frac{\partial w_0}{\partial x} \ . \tag{29}$$

Hence, we have for the transverse shear strain

$$\varepsilon_{xz} = \frac{\partial u}{\partial z} + \frac{\partial w}{\partial x} = -\theta + \frac{\partial w_0}{\partial x} = 0 \,. \tag{30}$$

With the strains defined, the stresses now follow with an appropriate constitutive law. If we assume a linear elastic material law we have

$$\sigma_x = \frac{E}{1-\nu^2}\varepsilon_x \,, \tag{31}$$

where E is Young's modulus and ν is Poisson's ratio. The bending moment around the horizontal axis is obtained as

$$M = D\frac{\partial \theta}{\partial x} = \frac{Eh^3}{12(1-\nu^2)}\frac{\partial^2 w_0}{\partial x^2} \,, \tag{32}$$

where the cubic thickness term h^3 is due to the moment of inertia. Note that for small deflections w_0, the term $\frac{\partial^2 w_0}{\partial x^2}$ is actually the curvature κ of the beam. This allows us to write the generalised stress-strain relationship of the Euler-Bernoulli beam in a clearer manner as $M = D\kappa$. The linear moment-curvature relationship is exploited by some approaches in cloth simulation to directly model bending forces (e.g. [VCMT95]).

In order to establish the governing equilibrium equations, we consider the forces acting on a differential beam element. In Fig. 10 loads and stress resultants on the beam are shown.

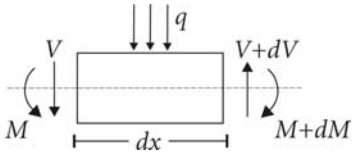

Figure 10: *Distributed lateral forces q, transverse shear force V, and bending moment M acting on a differential beam element of length dx.*

The beam is in equilibrium if the transverse internal force V (or shear resultant) and the external distributed load q are in balance. This leads to the equilibrium equation of the Euler-Bernoulli beam

$$\frac{Eh^3}{12(1-\nu^2)}\frac{\partial^4 w}{\partial x^4} = -q \,. \tag{33}$$

The above formulations directly carry over to cylindrically deformed plates. They can as well be translated to the general theory of (doubly curved) thin plates (see [ZT00b]). In engineering, thin plate elements are used to support lateral loads. Because curvature now occurs in both transverse directions, one speaks of the *neutral surface*, or simply *mid-surface*, in analogy to the neutral plane. Again, it is assumed that the stretch deformations of the mid-surface are negligible. Hence, the primary unknown is again the lateral deflection w. However, the deflection now varies in both x and y direction which renders the problem two-dimensional. For thin plates, the equilibrium equation is

$$\frac{Eh^3}{12(1-\nu^2)}\left(\frac{\partial^4 w}{\partial x^4} + 2\frac{\partial^4 w}{\partial x^2 \partial y^2} + \frac{\partial^4 w}{\partial y^4}\right) = -p \,. \tag{34}$$

This is a biharmonic equation involving fourth order partial derivatives. Investigating the corresponding strains it can be seen, that second order derivatives of the (lateral) displacement field are required. The thin plate equations have been used in computer graphics, too. For instance, they appear in a common minimisation problem from variational design (see e.g. [WW98]). Despite its demands on continuity, the thin plate approach can be used in physically based simulation. Since this theory does not take into account in-plane deformations it has to be augmented by an appropriate membrane model. This conjunction can be found in the class of *Kirchhoff-Love* thin shell theories (see Sec. 4.4).

4.3. Existing Approaches: From Springs to Shells

In this section we will report on bending models used in physically based simulation. In the seminal work of Terzopoulos et al. [TPBF87] a model for the animation of elastically deformable surfaces based on continuum mechanics is presented. The authors derive an elastic strain energy depending on the nonlinear metric and curvature tensors. The associated partial differential equations are discretised in space using finite differences on a regular quadrilateral grid. Although this approach is based on a physically sound theory, it was not widely adopted in the computer graphics community, due to its significant computational complexity. In the following years, approaches for the simulation of deformable surfaces like cloth mainly relied on particle and mass-spring systems. For modelling forces due to bending deformation, most of these methods use some kind of angular measure to approximate curvature. Breen et al. [BHW94] were among the first to use a coupled particle system in cloth simulation. The authors present an approach based on energy potentials for modelling the static drape of cloth. Departing from linear beam theory (see section 4.2), they first derive the bending energy between two successive edges in a rectangular discretisation. Curvature is approximated by fitting a circle through the three points involved in such a bending element. Using a biphasic curvature expression, Breen et al. model bending energy by approximating the nonlinear curves obtained from measurements with the *Kawabata Evaluation System* [Kaw80] numerically with quadratic fits. Once the energies are set up at the nodes, the gradients have to be computed to obtain nodal forces. The approach presented by Eberhardt et al. [EWS96] extends this work to the dynamic range. Computation times are greatly reduced through the use of sophisticated integration schemes. However, Eberhardt et al. do not approximate curvature but directly use the angle as a deformation measure.

Volino et al. [VCMT95] use a mass-spring system in-

spired by continuum mechanics. The basic bending element is formed by two adjacent triangles from the underlying unstructured grid (see Fig. 11). To determine curvature, a circle fitting inside the two triangles is found using the dihedral angle. The curvature over an element is then obtained as the inverse of the circle's radius. The authors point out that the curvature has to be limited to a certain maximum to prevent bending forces from reaching infinity. Using linear beam theory, the actual forces are deduced from the geometry of the involved triangles.

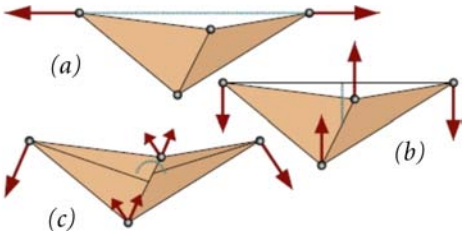

Figure 11: *Different ways of modelling bending forces on triangle meshes. Forces can be computed according to tensile crossover springs (a), based on the dihedral angle (b), or based on a weighted sum of vertex positions (c). (Fig. taken from [VMT06b])*

Baraff et al. [BW98] use the same basic bending element as in [VCMT95]. Following their proposed computational framework, a constraint expression for bending energy is derived. This essentially corresponds to an energy term which depends quadratically on the dihedral angle.

A different approach to cloth simulation was proposed by Eischen et al [EDC96, EB00]. Their method is based on the nonlinear shell theory derived by Simo et al. [SFR89a, SFR89b]. A four node bilinear element with nodal displacements and director rotations as the primary unknowns is used for discretisation (see [SFR89a]). In the context of shell theory, curvature is directly accessible through bending strains and does not have to be approximated otherwise. Like in [BHW94] Eischen et al. use measured data obtained from the Kawabata Evaluation System and use a 5th-order polynomial fit to approximate the curves. In sum, the approach leads to highly nonlinear equilibrium equations which have to be solved, for example, with the Newton-Raphson procedure. This solver is coupled with an adaptive arc length control to account for limit or bifurcation points in the solution due to buckling instabilities. The proposed method is limited to the static case and does not account for dynamic effects. For subsequent comparison, Eischen et al. [EB00] present a particle-based approach inspired from continuum mechanics. Like Breen et al. they use a regular quadrilateral discretisation but derive forces directly without explicit resort to energy potentials. However, their constructions are based on linear elasticity theory. Bending forces are derived using linear beam theory which again results in a linear moment-curvature relationship. The angle formed by two consecutive edges is taken as a direct measure for curvature. The authors state that the outcome of the two methods cannot be visually distinguished on the scale of the images they produced.

More recently, Choi et al. [CK02] proposed a bending model based on assumptions on the buckling behaviour of fabric. Starting from a standard quadrilateral mass-spring system, the basic bending element consists of an interleaved spring. The authors advocate that compressive in-plane forces on textiles lead to large out-of-plane deflections once a critical loading is reached. For the notoriously instable post-buckling state the buckled shape is predicted as a circular arc of constant length and curvature. With this assumption, the curvature can be computed analytically without any angle appearing. Hence, linear beam theory can be applied to derive the bending energy. Lastly, the authors derive expressions for force vectors and Jacobians at the nodes which allows the use of an implicit time integration scheme.

Bridson et al. [BMF03] proposed another derivation of bending forces for cloth simulation. Again, two adjacent triangles form the basic bending element. With the assumption that bending forces should neither cause in-plane deformation of the fabric nor lead to rigid body motions, they derive the directions and relative magnitudes for the four bending force vectors of an element. These vectors are then scaled with a bending stiffness constant and the sine of the dihedral angle (see Fig. 11, b). An additional scaling factor accounts for anisotropy of the mesh. For the numerical time integration, Bridson et al. suggest to use a mixed implicit-explicit integration scheme in which the (comparably small) bending forces can be handled in an explicit manner while viscous damping forces are treated implicitly. In this way computing the complicated derivatives of the bending forces is avoided.

In the bending model used by Etzmuß et al. [EKS03] forces are computed according to an approximation of the surface Laplacian. This idea is motivated by the fact that discrete mean curvature is closely related to the discrete surface Laplacian (see [MDSB03]). Using linear finite elements, the approximate Laplacian is computed for each element and projected onto the corresponding vertex normals. As usual, the element contributions are summed up to give the point wise Laplacian for every vertex.

While in most of the previous approaches curvature is approximated rather inaccurately, Grinspun et al. [GH*03] presented a method which is based on a sound curvature derivation. Their work extends existing cloth simulators to the range of objects for which the bending resistance is predominant. To this end, a discrete flexural energy potential is established using differential geometry. Again, the basic bending element consists of two adjacent triangles. The energy derives from an approximation to the squared difference of mean curvature in the current and initial configuration. The derivatives of the bending energy are intricate to com-

pute and hence the authors suggest the use of an automatic differentiation system.

Recently, Volino et al. [VMT06b] presented an approach which combines fairly good accuracy for representing quantitative bending stiffness with a simple and efficient computational procedure. In this method, a *bending vector* is computed that represents the bending of the surface through a simple linear combination of particle positions (see Fig. 11, c). This vector is then redistributed as particle forces according to the bending stiffness of the surface. It can be shown that this scheme preserves total translational and rotational momentum without the need of recomputing the distribution coefficients according to the current position of the particles. This leads to a very simple computation process which is entirely linear, and thus very well adapted to implicit numerical integration. In terms of computational simplicity and speed, this approach competes well with simpler spring models, since it requires only linear operations, which can conveniently be cast into matrix form. Additionally, the Jacobian of the bending forces is constant throughout the simulation, which is a considerable computational advantage when using implicit numerical integration methods. The authors demonstrated that the accuracy of their model is close to the more accurate normal-based method as used in, e.g, [BMF03]. Nevertheless, it does not require expensive operations like trigonometric functions or square root evaluations.

Assuming an inextensible surface and, thus, isometric deformations, Bergou et al. [BWH*06] derived a bending energy which depends quadratically on nodal positions. Similarly to [VMT06b], this leads to a computationally economic model with bending forces linear in positions and a constant associated Jacobian. An extensive analysis of this and other isometric bending models based on discrete curvature energies can be found in [WBH*07].

4.4. SubdivisionFE

As pointed out above, the thin shell equations are a good choice when physical accuracy is important. A corresponding finite element approach requires a C^1-continuous displacement field (to be exact the shape functions have to be in H^2). The main problem with this requirement is guaranteeing continuity across elements which usually necessitates the use of additional variables (e.g. nodal rotations and slopes). An elegant and convenient way to avoid this additional complexity is to use subdivision finite elements as a basis (see [COS00]). The following section gives a brief account of the approach presented in [TWS06], which utilizes this type of finite elements .

4.4.1. Thin Shell Mechanics

In the Kirchhoff-Love theory of thin shells the configuration mapping (2) is expressed in terms of the mid-surface parametrisation $\mathbf{x}(\theta^1,\theta^2)$ (see Fig. 12) as

$$\varphi(\theta^1,\theta^2,\theta^3) = \mathbf{x}(\theta^1,\theta^2) + \theta^3\mathbf{a}_3(\theta^1,\theta^2)\,, \qquad (35)$$

where θ^i denote curvilinear coordinates and $\mathbf{a}_3$ is the director field normal to the surface. In analogy to Eq. (2) we write

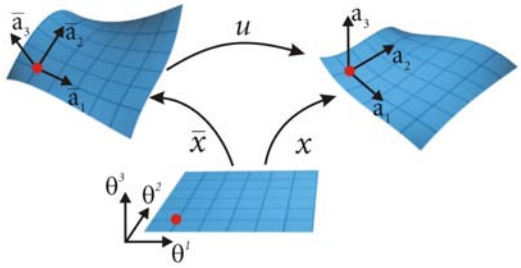

Figure 12: *A material point (red) on the shell's mid-surface with basis vector frame in the initial, reference, and current configuration (from left to right).*

$$\mathbf{x}(\theta^1,\theta^2) = \bar{\mathbf{x}}(\theta^1,\theta^2) + \mathbf{u}(\theta^1,\theta^2)\,. \qquad (36)$$

From this, tangential surface basis vectors can be defined as

$$\mathbf{a}_\alpha = \mathbf{x}_{,\alpha}\,, \qquad (37)$$

where the comma denotes partial differentiation. From hereon, we use Greek indices to denote variables, which can take on the values $\{1,2\}$ in order to distinguish from Latin indices, which can take on the values $\{1,2,3\}$. Moreover, the covariant tangent base vectors are given through differentiation of the configuration mapping as

$$\mathbf{g}_\alpha = \varphi_{,\alpha} = \mathbf{a}_\alpha + \theta^3\mathbf{a}_{3,\alpha} \qquad (38)$$

from which the surface *metric tensor* is derived as

$$g_{ij} = \mathbf{g}_i \cdot \mathbf{g}_j. \qquad (39)$$

Following Eq. (5) this leads to the definition of the Green strain

$$\varepsilon^G_{ij} = \frac{1}{2}(g_{ij} - \bar{g}_{ij}) = \alpha_{ij} + \theta^3\beta_{ij}\,, \qquad (40)$$

where α and β are membrane and bending strains, respectively. In the Kirchhoff-Love theory (cf. Sec. 4.2), the director $\mathbf{a}_3$ is assumed to stay normal to the surface, straight and unstretched,

$$\mathbf{a}_3 = \frac{\mathbf{a}_1 \times \mathbf{a}_2}{|\mathbf{a}_1 \times \mathbf{a}_2|}\,. \qquad (41)$$

Consequently, we have $\alpha_{3\beta} = \alpha_{\alpha 3} = 0$. The strains then simplify to

$$\alpha_{\alpha\beta} = \frac{1}{2}(\mathbf{a}_\alpha \cdot \mathbf{a}_\beta - \bar{\mathbf{a}}_\alpha \cdot \bar{\mathbf{a}}_\beta), \quad \beta_{\alpha\beta} = (\bar{\mathbf{a}}_{\alpha,\beta} \cdot \bar{\mathbf{a}}_3 - \mathbf{a}_{\alpha,\beta} \cdot \mathbf{a}_3)\,.$$

Departing from $\mathbf{a}_\alpha = \bar{\mathbf{x}}_{,\alpha} + \mathbf{u}_{,\alpha}$ and neglecting nonlinear terms, this can be recast to an expression which is linear in

displacements [COS00]. Resultant membrane and bending stresses follow as

$$n^{\alpha\beta} = \frac{\partial \Psi}{\partial \alpha_{\alpha\beta}}, \quad m^{\alpha\beta} = \frac{\partial \Psi}{\partial \beta_{\alpha\beta}}, \tag{42}$$

where Ψ is the strain energy density. The particular form of Ψ depends again on the specific material law used (see Sec. 3.5). As for the plane stress case, it is possible to use the corotational strain measure of Eq. (11), although the rotation field is determined in a slightly different way (see [TWS06]).

4.4.2. Subdivision-Based Finite Elements

Subdivision is a process for constructing smooth limit surfaces through successive refinement of an initial control mesh. As one of the most prominent examples Loop's subdivision scheme fulfils all prerequisites to serve as a basis for the thin shell discretisation. Besides the usual C^1-continuity inherent to subdivision surfaces, an important feature of this schemes is that the curvature of the resulting limit surface is L_2- or square integrable [RS01]. Due to this property, the subdivision basis functions can be used as shape functions for the FE-solution of the thin shell equations. In each step of this subdivision method, the positions of newly inserted nodes as well as those of old nodes are computed through a linear combination of vertices from the coarse mesh determined by the so called subdivision mask.

Only the immediate neighbours (i.e. the 1-ring) of a vertex have influence on this computation which gives rise to an efficient implementation. The process of subdivision itself can be considered as a linear operation and consequently be written in matrix form. It is therefore possible to directly derive properties like derivatives of the limit surface using an Eigenanalysis of the subdivision matrix. This yields simple expressions that can be computed efficiently. The resulting limit surface (or more generally, the limit field) can be evaluated at an arbitrary point in the interior of a patch, which is an important property for numerical integration. The key observation is that in regular settings (i.e. when all involved vertices have valence 6) Loop's scheme leads to generalised quartic box splines. In this case surface properties in one patch are completely defined by the 12 nodal values in the 1-neighbourhood (see Fig. 14) and the associated box spline basis functions N_i.
For instance, if we denote the local patch coordinates by θ^α, the limit surface can be expressed as

$$\mathbf{x}(\theta^1,\theta^2) = \sum_{i=1}^{12} N_i(\theta^1,\theta^2)\mathbf{x}_i, \tag{43}$$

where $\mathbf{x}_i$ are the nodal positions of the underlying mesh. In the same way, the displacement field interpolation is obtained from the nodal values. Additionally, differential quantities can be determined as

$$\mathbf{x}_{,\alpha}(\theta^1,\theta^2) = \sum_{i=1}^{12} N_{i,\alpha}(\theta^1,\theta^2)\mathbf{x}_i. \tag{44}$$

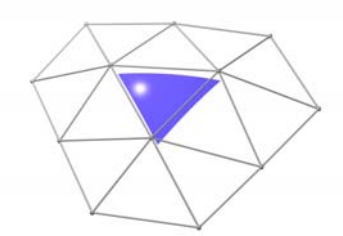

Figure 14: *1-ring neighbourhood of a regular patch consisting of 12 nodes.*

If the patch has an irregular vertex, the box spline assumption no longer holds and thus interior parameter points cannot be evaluated. For the following finite element discretisation, however, only quantities at the barycenter of the triangles are needed for integral evaluation. Hence, Cirak et al. required the initial mesh to have at most one irregular vertex per triangle. Then, after one subdivision step the barycenter lies again inside a regular patch (see Fig. 14). This process of subdivision and evaluation of the newly generated patch can again be expressed as a sequence of matrix multiplications.

Spatial Discretisation With the definition of the membrane and bending strains and assuming a linear elastic material (Eq. (17)) the internal energy from the virtual work equation (25) can be rewritten as

$$\int_\Omega \delta\varepsilon : C(\varepsilon) d\Omega = \int_\Omega \left(\delta\alpha^T \mathbf{H}_m \alpha + \delta\beta^T \mathbf{H}_b \beta \right) d\Omega, \tag{45}$$

where $\mathbf{H}_m$ and $\mathbf{H}_b$ are matrices corresponding to the membrane and bending part of the material law (see [COS00] for a complete derivation). Due to the linear strain interpolation, we have

$$\alpha(\theta^1,\theta^2) = \sum_i^N \mathbf{M}_i(\theta^1,\theta^2)\mathbf{u}_i, \quad \beta(\theta^1,\theta^2) = \sum_i^N \mathbf{B}_i(\theta^1,\theta^2)\mathbf{u}_i$$

for matrices $\mathbf{M}_i$ and $\mathbf{B}_i$ relating nodal displacements $\mathbf{u}_i$ to membrane and bending strain. This gives rise to a formulation of the complete system in the classical form of

$$\mathbf{K}\mathbf{u} = \mathbf{f} \tag{46}$$

with vectors of nodal displacement $\mathbf{u}$ and forces $\mathbf{f}$. The stiffness matrix $\mathbf{K}$ can be assembled in the usual element-wise fashion

$$\mathbf{K}_{ij} = \sum_e \int_{\Omega_e} \left(\mathbf{M}_i^T \mathbf{H}_m \mathbf{M}_j + \mathbf{B}_i^T \mathbf{H}_b \mathbf{B}_j \right) d\Omega = \sum_e \mathbf{K}_{ij}^e. \tag{47}$$

The integral in this equation can be evaluated using numerical quadrature. In this form, the above equations are only valid on regular patches. However, as mentioned above, in irregular settings one subdivision step is sufficient for evaluations at the barycenters. For a patch with an irregular vertex of valence N let $\mathbf{S}$ denote the the subdivision operator

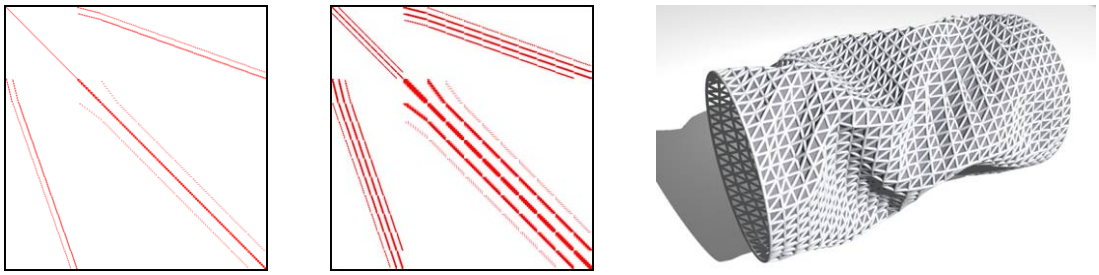

Figure 13: *A comparsion of the sparsity structures resulting from an ordinary finite element approach (*left*) and the subdivision FE-based method (*middle*) for a regularly tesselated mesh comprised of 1448 vertices (*right*).*

(see [COS00]). Further, let $\mathbf{P}$ be the projection operator extracting the 12 vertices corresponding to the central regular subpatch (Fig. 14, right). Then we can write

$$\mathbf{K}_{ij}^{e} = \int_{\Omega_e} \left[\mathbf{S}^T \mathbf{P}^T \left(\mathbf{M}_i^T \mathbf{H}_m \mathbf{M}_j + \mathbf{B}_i^T \mathbf{H}_b \mathbf{B}_j \right) \mathbf{P}\mathbf{S} \right] d\Omega \quad (48)$$

and thus simply include the conceptual subdivision step into the stiffness matrix.

To cover dynamic effects of moving and deforming objects inertia as well as viscous forces have to be included. This leads to a second order ODE in time analogous to Eq. (27). After transformation into a set of coupled first order ODEs, standard numerical time integration schemes can be applied (see [HE01]).

Discussion The setup process for the matrix $\mathbf{K}$ in Eq. (48) is very similar to a common FE-formulation. The evaluation cost for an element matrix in this approach, however, is slightly higher than for usual methods. This is mainly due to the increased connectivity of the elements: a regular patch contains a neighbourhood of 12 nodes where for standard approaches there are only three. This means that more entries have to be computed and written into the system matrix (see Fig. 13). As a consequence, the setup an resolution of the linear system is slower than for standard approaches like the one described in Sec. 3.6. To put this into the right context, processing times for collision detection and response can still be much larger. A striking advantage of this approach is that it alleviates the modelling of a broad range of different materials (see Fig. 15). For example, because the full curvature tensor is available (cf. Eq. (25)), bending properties can directly be controlled, allowing for complex anisotropic models. Since this approach is entirely based on the sound mathematical foundation of continuum mechanics, the simulation is largely independent of discretisation throughout a broad range of resolutions.

5. Parallel Techniques for Cloth Simulation

Up to this point we have mostly dealt with methods that were designed to exhibit as much visual realism as possible. Moreover, techniques for measuring and reproducing real world materials in an accurate way were discussed. This realism and accuracy, however, comes at a price: the computational costs can be very high and run times for realistic scenarios are often excessive. Most of the computation time is spent on two stages, time integration and collision handling. In the following, we will therefore examine these two major bottlenecks, which are present in every physically-based cloth simulation system.

Figure 15: Top: *Different types of folds on a garment's sleeve generated using a typical textile material.* Bottom: *Sequence taken from an animation of axial compression of a cylinder with a metal-like material.*

5.1. Implicit time integration

The ordinary differential equation resulting from the temporal discretisation of Eq. (27) are notoriously stiff. Due to stability reasons implicit schemes are widely accepted as the method of choice for numerical time integration (cf. [BW98]). Implicit schemes require the solution of a usually nonlinear system at each time step. As a result of the spatial discretisation, the matrix of this system is usually very sparse. There are essentially two alternatives for the solution of the system. One is to use an iterative method like the popular conjugate gradients (cg) algorithm [She94]. Another is to use direct solvers based on factorization. The cg-method is more popular in computer graphics as it offers much sim-

pler user interaction, alleviates the integration of arbitrary boundary conditions and allows balancing accuracy against speed. We will therefore focus on the cg-method in the following.

5.2. Collision Handling

For realistic scenes, the interaction of deformable objects with their virtual environment has to be modelled. This involves the detection of proximities (collision detection) and the reaction necessary to keep an intersection-free state (collision response). In the remainder, we refer to these two components collectively as *collision handling*. We usually distinguish between external collisions (with other objects in the scene) and self-collisions. For each of these types different variants of algorithms are usually used. Even with common acceleration structures, these algorithms are still computationally expensive. For complex scenarios with complicated self-collisions the collision handling can easily make up more than half of the overall computation time. It is therefore a second bottleneck for the physical simulation and hence deserves special attention.

Target Architecture The distinctive characteristic of parallel platforms is their memory architecture. In the following we will consider both distributed and shared address spaces. As exemplary representatives we choose clusters built from commodity components for the distributed memory setting and multi-core, multi-processor systems for the shared memory case. Although the basic ideas remain the same, the specific form and implementation of the actual algorithms depend on the target architecture. This is due to the fact that for different platforms, optimal performance is achieved in quite different ways as well. For example, one of the most important objectives on distributed memory architectures (DMA) is to minimize communication between the nodes of the cluster. While (inter-process) communication on shared address space architectures is not a costly aspect, care has to be taken for synchronization (including e.g. data access by multiple threads) as well. We will point out important differences at the appropriate places.

Programming Paradigms and Implementation Another distinguishing property of a given architecture is how it supports and favours different programming paradigms. On DMA machines a widely adopted strategy is that of single program multiple data (SPMD) where every node executes the same code but has different data to operate on. The communication in this case is usually message passing style, for example an implementation of the message passing interface (MPI). As a concrete example, the MPI-based numerical toolkit PETSc offers extensive support for SPMD style programming in scientific computation. Although it supports other approaches and can be run on many architectures, PETSC is especially well suited for large scale applications on DMAs. It can also be used in the shared memory setting although in this case, message passing is actually not necessary and synchronization can often be implemented more efficiently. Additionally, it is often desirable on these platforms to use a multi-threaded programming approach which is not supported by PETSc. However, most of the operating systems directly support this paradigm and there are numerous toolkits and frameworks offering convenient interfaces for thread-based programming. Another interesting alternative with growing acceptance is the OpenMP interface. This interface offers extensive support for exploiting loop-level parallelism. It is entirely based on compiler directives and is therefore highly portable. Moreover, the user does not have to care about thread accounting (i.e. creation, synchronization, termination) which makes this interface easy to use.

5.3. Parallel Solution of Sparse Linear Systems

As was stated above, one of the major computational burdens is due to the solution of a sparse linear equation system related which derives from implicit time integration. We assume that a sparse linear system of the form $\mathbf{Ax} = \mathbf{b}$ is to be solved up to some residual tolerance using the cg-method. The number of necessary iterations and therefore the speed of convergence depends on the condition number of the matrix $\mathbf{A}$. Usually, this condition number is improved using a preconditioning matrix $\mathbf{M}$ leading to a modified system

$$\mathbf{M}^{-1}\mathbf{Ax} = \mathbf{M}^{-1}\mathbf{b},$$

where $\mathbf{M}^{-1}\mathbf{A}$ is supposed to have a better condition number and $\mathbf{M}^{-1}$ is fairly easy to compute. The choice of an appropriate preconditioner is crucial because it can reduce the number of iterations substantially. The setup and solution of the linear system now breaks down to a sequence of operations in which (due to their computational complexity) the sparse matrix vector multiplication (SpMV) and the application of the preconditioner are most important. As a basis for the actual parallelization we will consider problem decomposition approaches subsequently.

5.4. Problem Decomposition

In the following explanations we use the compressed row storage (CRS) format for sparse matrices in which nonzero entries are stored in an array along with a row pointer and a column index array (see [Saa03]). The most intuitive (and abstract) way to decompose the SpMV operation into a number of smaller sub-problems is to simply partition the matrix into sets of contiguous rows. The multiplication can then be carried out in parallel among the sets. This simple approach has several disadvantages. First, the matrices we deal with are always symmetric (due to the underlying PDE). Hence, only the upper triangular part, including the diagonal, has to be stored. This leads to smaller memory requirements for the data as well as for the index structure. In its sequential version, the resulting numerical kernel is more efficient (cf.

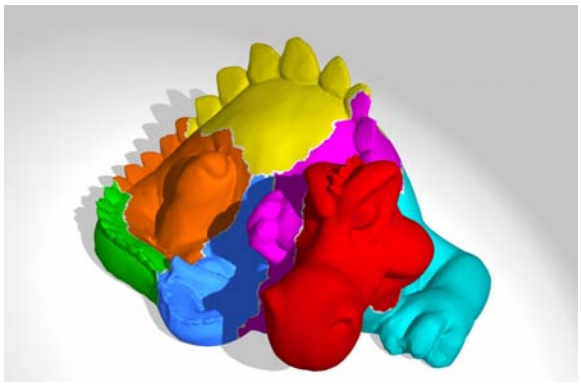

Figure 16: *The* Phlegmatic Dragon *model is decomposed into 8 disjoint regions (indicated by different colours). This partitioning serves as a basis for parallel processing.*

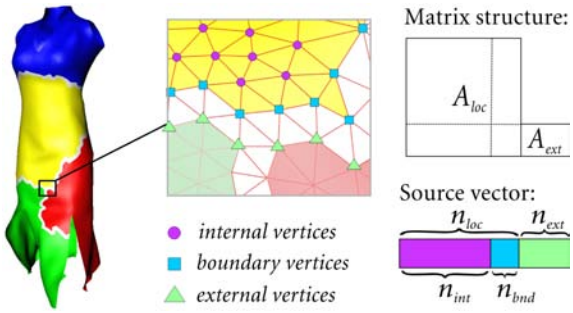

Figure 17: *Decomposition of a mesh into four disjoint partitions indicated by different colours. The associated vertex ordering leads to a special structure of the matrices and the source vector.*

[LVDY04]): it visits every matrix entry only once, performing both dot products and vector scalar products. However, the access pattern to the solution vector is not as local as for the non-symmetric case, i.e., entries from different sets need to be written by a single processor. The required synchronization would make a direct parallel implementation of the symmetric SpMV kernel inefficient. Another reason why the above decomposition is inadequate is that it does not take into account two other important components of linear system solution: matrix assembly and preconditioning. Methods based on domain decomposition are better suited for this case. They divide the input data geometrically into disjoint regions. Here, we will only consider non-overlapping vertex decompositions, which result in a partitioning P of the domain Ω into subdomains Ω_i such that $\Omega = \cup_i \Omega_i$ and $\Omega_i \cap \Omega_j = \emptyset$, for $i \neq j$. Decompositions can be obtained using graph partitioning methods such as Metis [KK96] in our case. An example of this can be seen in Fig. 16 and Fig. 17, which also shows a special vertex classification. This will be explained in the next section.

5.5. Parallel Sparse Matrix Vector Multiplication

Let $n_{i,loc}$ be the number of local vertices belonging to partition i and let V_i be the set of corresponding indices. These vertices can be decomposed into n_{int} internal vertices and n_{bnd} interface or boundary vertices, which are adjacent to n_{ext} vertices from other partitions (see Fig. 17). If we reorder the vertices globally such that vertices in one partition are enumerated sequentially we obtain again a partitioning of the matrix into a set of contiguous rows. The rows $a_{i,0}$ to $a_{i,n}$ of matrix A where $i \in V_i$ have the following special structure: the block $A_{i,loc}$ defined by $\{a_{lm} | l \in V_i, m \in V_i\}$ and lying on the diagonal of A is symmetric. The nonzero entries in this block describe the interaction between the local nodes of partition i. More specifically, this means that when nodes l and m are connected by an edge in the mesh, there is a nonzero entry a_{lm} in the corresponding submatrix of A. Besides this symmetric block on the diagonal there are further nonzero entries a_{le} where $l \in V_i$ is an interface node and $e \notin V_i$. These entries describe the coupling between the local interface nodes and neighbouring external nodes. The multiplication can be carried out efficiently in parallel if we adopt the following local vertex numbering scheme (cf. [Saa03]). The local vertices are reordered such that all internal nodes come before the interface nodes. For further performance enhancement, a numbering scheme that exploits locality (such as a self avoiding walk [OBHL02]) can be used to sort the local vertices. Then, external interface nodes from neighbouring partitions are locally renumbered as well. Let A_{ext} be the matrix which describes the coupling between internal and external interface nodes for a given partition. Notice that A_{ext} is a sparse rectangular matrix with n_{bnd} rows. With this setup the multiplication proceeds as follows

1. $y(0, n_{loc}) = A_{loc} \cdot x(0, n_{loc})$
2. $y(n_{int}, n_{loc}) += A_{ext} \cdot x_{ext}(0, n_{ext})$

The first operation is a symmetric SpMV, the second one is a non-symmetric SpMV followed by an addition. Both these operations can be carried out in parallel among all partitions. This decomposition is not only used for the SpMV kernel but also as a basis for the parallel matrix assembly, preconditioner setup and preconditioner application. Additionally, it can be implemented efficiently on both distributed and shared memory architectures.

5.6. Parallel Preconditioning

In order to make the cg-method fast, it is indispensable to use an efficient preconditioner. There is a broad variety of different preconditioners ranging from simple diagonal scaling (Jacobi preconditioning) to sophisticated multilevel variants. For the actual choice one has to weigh the time saved from the reduced iteration count against the cost for setup

and repeated application of the preconditioner. Additionally, one has to take into account how well a specific preconditioner can be parallelized. Unfortunately, designing efficient preconditioners is usually the most difficult part in the parallel cg-method [DHv93]. As an example, the Jacobi preconditioner is very simple to set up and apply even in parallel but the reduction of necessary iterations is rather limited. Preconditioners based on (usually incomplete) factorization of the matrix itself or an approximation of it are more promising. One example from this class is the SSOR preconditioner. It is fairly cheap to set up and leads to the solution of two triangular systems. For the sequential case, this preconditioner has proved to be a good choice in terms of efficiency [HE01]. However, parallelizing the solution of the triangular systems is very hard. Even if it is not possible to decouple the solution of the original triangular systems into independent problems we can devise an approximation with the desired properties. Let $\bar{A}$ be the block diagonal matrix with block entries $A_{ii} = A_{i,loc}$, i.e. the external matrices A_{ext} are dropped from A to give $\bar{A}$. Setting up the SSOR-preconditioner on this modified matrix leads again to the solution of two triangular systems. However, solving these systems breaks down to the solution of decoupled triangular systems corresponding to the $A_{i,loc}$ blocks on the diagonal. This means that they can be carried out in parallel for every partition. For reasons of data locality we use n smaller SSOR preconditioners constructed directly from the $A_{i,loc}$-blocks. Approximating A with $\bar{A}$ means a comparably small loss of information which in turn leads to a slightly increased iteration count. However, this increase is usually small compared to the speedup obtained through parallelization.

5.7. Optimizations

Besides the topics that were treated above a further aspect restricts the efficiency of a parallel implementation of the cg-method. Dense matrix multiplications usually scale very well since they have regular access patterns to memory and a high computational intensity. This is not true for the SpMV case. A typical SpMV algorithm using a matrix in the CRS format looks as follows:

Algorithm 1 Symmetric Spmv

```
1: for i = 1 to n_rows do
2:     start = ptr[i], end = ptr[i+1];
3:     for j = start to end do
4:         y[i] += dat[j] * x[ind[j]];
5:     end for
6: end for
```

It can be seen that the actual matrix data A_{dat} as well as the index structure *ind* is traversed linearly (with unit stride) while accesses to the vector's data x_{dat} occur totally arbitrary, i.e. the locality of these data accesses cannot be assumed. It can also be observed that the computational intensity per data element is rather modest. The performance of the SpMV algorithm is therefore mostly limited by memory bandwidth and cache performance. Because of this, it is important to improve data locality and thus cache performance. A good way to achieve this is to exploit the natural block layout of the matrix as determined by the underlying PDE: the coupling between two vertices is described by a 3x3 block – therefore nonzero entries in the matrix occur always in blocks. This blocked data layout already compensates for a lot of the inefficiency. An additional benefit can be achieved using single precision floating point data instead of double precision. This reduces the necessary matrix data (not including index structure) transferred from memory by a factor of two.

5.8. Parallel Collision Handling

From the parallelization point of view, the problem of collision handling differs substantially from the physical model. Collisions can be distributed very unevenly in the scene and their (typically changing) locations cannot be determined statically. This is why the naive approach of letting each processor care for the collisions of its own partition can lead to considerable processor idling, which seriously affects the overall parallel efficiency. Therefore, a dynamic problem decomposition is mandatory. Because bounding volume hierarchies (BVHs) are widely used for collision detection between deformable objects, it is reasonable to use this approach as a basis for a parallel implementation. As we have seen in a previous section, we can generally distinguish between two different types of collisions: external collisions and self collisions. To detect the first type we have to test our deformable object against every other (rigid or deformable) object in the scene. For the latter case, the deformable object has to be tested against itself. In this section, we will first lay down the basic algorithm for parallel collision handling. Subsequently, necessary adaption for both distributed and shared memory architectures will be addressed.

5.8.1. Basic Problem Decomposition

The recursive collision test of two BVHs can be considered as a depth-first tree traversal. For inducing parallelism, this procedure is implemented using a stack which holds individual tests of two BVs.

In the BVH testing procedure, expansion of a tree node results in two additional tree nodes each representing a refined BVH test. As in the sequential procedure, the first test is carried out by starting a new recursion level. However, before entering the recursion, the second test is pushed onto the stack. The traversal goes on downwards until a leaf is reached. Upward traversal begins by processing elements from the stack. In this way, all of the nodes in the tree are visited. Fig. 18 shows a snapshot of a BVH testing process along with the corresponding state of the stack. Note that, although we assume a binary tree in this example, an extension to general n-ary trees is possible (see [TPB07] for

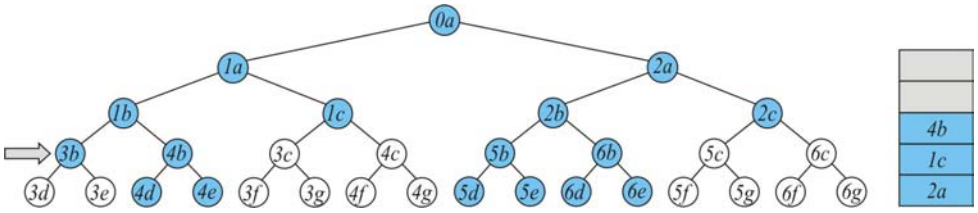

Figure 18: *Dynamic problem decomposition. The arrow indicates the current state of the BVH testing procedure. The stack on the right stores the root of untried branches.*

details). With a proper indexation of the hierarchies a single test can be represented using only a few indices and can thus be stored economically in appropriate data structures.

Tests which are recorded on the stack can be executed in one of the following ways:

- A test can be removed from the top of the stack and executed sequentially when the recursion gets back to the current level. Conceptually, this case is very similar to the procedure of the original algorithm.
- One or more tests can be removed from the bottom of the stack and executed by a newly generated (sub-)task. For each assigned test, this task executes a BVH testing procedure for which the considered test defines the root of a BVH testing tree.

In oder to prevent that tasks of too fine a granularity are generated, several tests can be removed at once from the stack and assigned to a single task. The actual decomposition and load balancing strategy depends on the specific platform and is discussed in subsequent paragraphs.

5.8.2. Distributed Memory

The first step for embedding the above algorithm into the SPMD framework is to construct a BVH. The physical modelling phase provides a decomposition of the input mesh into *local* meshes owned by nodes of the cluster. On each node, a BVH hierarchy is set up on this mesh using a standard top-down approach. Once this is done, the root nodes of the different processors can be combined to form a global hierarchy of the mesh. Since the structure of this hierarchy is kept constant throughout the simulation, this global hierarchy can safely be replicated once on every node. This replication late enables the location-independent execution of sub-tasks. While external collision can now be treated in the usual way, care has to be taken for self-collisions. Two different types of self-collisions have to be distinguished: collisions between sub-meshes of different processors and those that are real self-collisions on the processor-local mesh. For the latter case, existing techniques can be used since this corresponds to the usual self collision problem. For the case of inter-processor self-collisions, the corresponding BVHs are tested against each other, similar to the way standard collisions are treated.

All discussed BVH tests form a set of *top-level tasks* of the parallel collision handling method. For scenes where collisions are not uniformly distributed, the number of top-level tasks and also the amount of time to execute individual top-level tasks can differ considerably among the processors. Hence, a load balancing scheme which distributes the load among the processors evenly is mandatory.

Dynamic Problem Decomposition and Load Balancing As described above, every processor maintains a stack data structure where dynamically generating tasks are stored. For dynamic load balancing tasks must be transferred between processors at run-time. In distributed memory architectures, task transfers require explicit communication operations. Depending on the information associated with a task, task transfers can significantly contribute to the overall parallel overhead, especially when the computation to communication ratio of the tasks is poor. Finding a compact description of tasks is therefore crucial to minimize communication overhead. In our case, the cost for transferring a task is rather low.

Because the global BVH is replicated on every node, individual tests can be represented simply as an array of integers of the form $(obj1, dop1, obj2, dop2)$. The two BV hierarchies to be tested are identified by $obj1$ and $obj2$, and the root of the test is specified by $dop1$ and $dop2$.

The additional costs for building the copies of the BV hierarchies at the initialization phase are negligible since storage requirements are comparably low (say, only a few megabytes at max) even for complex scenes. The overhead during the course of the simulation is also insignificant: Updating hierarchies (at the beginning of every collision handling phase) requires one all-to-all broadcast operation to provide all processors the complete positions vector.

The dynamic load balancing process is responsible for triggering and coordinating task creation and task transfer operations in order to prevent that processors run idle. In order to achieve high scalability, a fully distributed scheme should be used. The distributed task pool model is a specifically well-suited basis. In this scheme every processor maintains a local task pool. Upon creation, a task is first placed in the local task pool. Subsequently, it can be instantiated

and executed locally, when the processor gets idle. It also might be transfered to a remote task pool for load balancing purposes. As discussed subsequently, task creation and task transfer operations are initiated autonomously by the processors.

To achieve a high degree of efficiency an (active) processor must be able to satisfy task requests from other processors immediately. Additionally, we must ensure that tasks with sufficient granularity are generated. In our case, this means that when a task is to be created, the stack must contain a minimum number of BVH tests which can be assigned to the task. In order to meet both requirements, we generate tasks in a proactive fashion, i.e. independently of incoming tasks requests. Tasks are generated (and placed in the local task pool) when the size of the stack exceeds a threshold value τ. Since task generation generally imposes an overhead (even when the new task is subsequently executed on the same processor) we increase τ linearly with the current size of the task pool σ: $\tau = a\sigma + b$. This simple heuristics enables us on the one hand to provide in a timely fashion tasks with a minimum granularity (determined by b) and ensures on the other hand that the overhead of additional task decomposition operations is compensated by an increased granularity of the resulting tasks. The parameter values a and b largely depend on the parallel architecture used and should be determined experimentally. Usually, choosing $a = 2$ and $b = 8$ delivers the best results.

A new task gets $\tau/2$ tests, where the tests are taken from the bottom of the stack. Tests are taken from the bottom of the stack for creating new tasks because such tests have a higher potential of representing a large testing tree since they originate closer to the root of the current testing tree.

For transferring tasks between task pools, a receiver initiated scheme is employed. When a processor runs idle and the local task pool is empty, it tries to steal tasks from remote pools. First, a victim node is chosen to which a request for tasks is sent. For selecting victims we apply a round robin scheme. If available, the victim transfers a task from its pool to the local pool. Otherwise the request is rejected by sending a corresponding message. In the latter case another victim node is chosen and a new request is issued.

For the actual implementation of the previously described methods, tools supporting distributed multithreading are needed. An efficient and convenient variant can be found in the parallel system platform DOTS [BKLwW99]. DOTS provides extensive support for the multithreading parallel programming model (not to be confused with the shared-memory model) which is particularly well suited for task-parallel applications that employ fully dynamic problem decomposition. In this way, the user does not have to care for task transfer and thread accounting explicitly, which is a great alleviation. Combining PETSc and DOTS, the collision handling algorithm can be implemented efficiently on DMAs. For a thorough performance analysis the interested reader is referred to [TB06, TB07].

5.8.3. Shared Memory

Compared to the approach for DMAs the basic parallelization strategy remains the same. Because communication is cheaper and thus dynamic collaboration between threads is less expensive, the shared memory setting enables us to set up heuristics exploiting temporal and spatial coherence. In this way, thread creation overhead can be controlled effectively.

Unlike in the distributed memory setting we do not have to care for load balancing explicitly. As long as there are enough threads ready for execution the scheduler will keep all cores busy. However, for problems with high irregularity, like parallel collision handling, it is generally impossible to precisely adjust the amount of logical parallelism to be exploited to the amount of available parallelism (i.e., idle processors). Especially on shared memory architectures, thread creation overhead can considerably contribute to the overall parallel overhead. Therefore, an over-saturation with threads has to be avoided as well.

In [TPB07] thread creation overhead is minimized on two levels. On the algorithmic level, a heuristics-based approach is used which prevents threads with too fine a granularity from being generated. On the implementation level, the process of thread creation and thread execution is decoupled. The next two paragraphs explain these optimizations in more detail.

Controlling Task Granularity For effectively controlling the granularity of a task, we need a good estimate of how much work corresponds to a certain task. The computational cost for carrying out a test in the collision tree is determined by the number of nodes in its subtree. Generally, this number is not known in advance. Because of the inherent temporal locality due to the dynamic simulation we can, however, exploit coherence between two successive time steps. After each collision detection pass we compute the number of tests in the respective subtree for every node in the collision tree using back propagation. This information is then used as a work estimate for tasks in the subsequent collision handling phase.

In this way, creating tasks with too small an amount of work can be avoided. Additionally, this information can be used to determine which tests should be carried out immediately. The error involved in the work estimation is usually very small. This can also be seen in Fig. 20 which shows a comparison of the estimated and the actual work load for 5.5 seconds of simulation for a representative test scene (see Fig. 19). The new task splitting scheme performs robustly and always keeps track with usual approaches. In most applications, this scheme even results in a significant performance

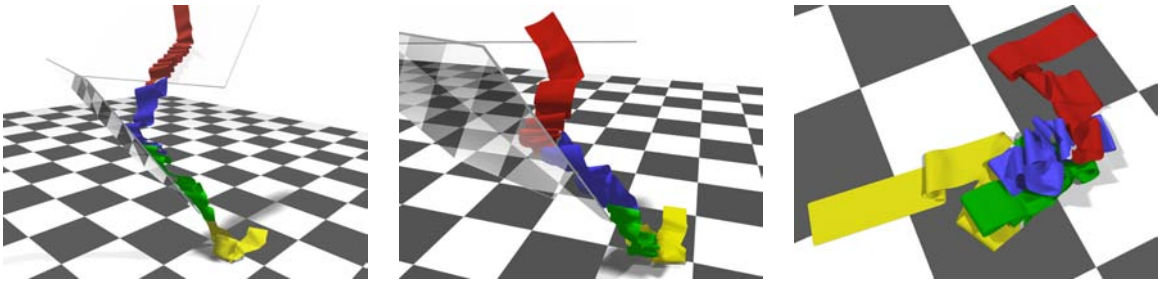

Figure 19: *Three shots from a representative test scene. A long (00.5m x 2.00m) ribbon consisting of 4141 vertices falls on two slightly inclined planes and slides onto the floor. Due to surface friction complex folds are formed as it slides over the planes. This again leads to complicated self-collisions which are reliably handled by the parallel collision handling algorithm.*

gain [TPB07]. This attests to the fact that temporal coherence in dynamic collision detection is a valuable source for performance improvements.

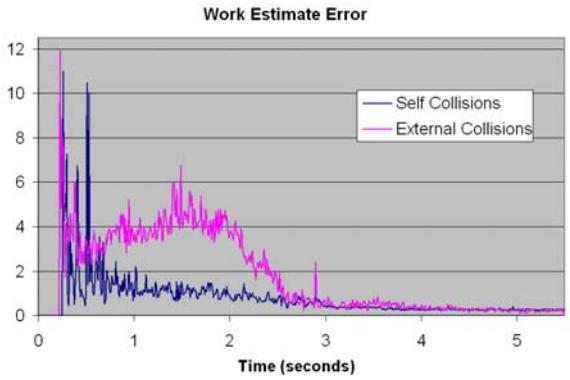

Figure 20: *Work estimate error for a typical test scene. The diagram shows the deviation from the actual amount of work over time in percent. Even in this very dynamic scene, the temporal coherence is high.*

Implementation For an implementation of the above algorithm, methods supporting the multithreaded programming are needed. Again, DOTS is an attractive candidate for this purpose. In order to ensure high performance on shared memory architectures, DOTS employs lightweight mechanisms for manipulating threads. Forking a thread results in the creation of a passive object, which can later be instantiated for execution. Thread objects can either be executed by a pre-forked OS native worker thread or can be executed as continuation of a thread which would otherwise be blocked, e.g., a thread reaching a synchronization primitive.

The above algorithm performs good and scales well even in challenging scenes (see Fig. 19). The actual speed-up that can be obtained depends, of course, on the specific scene and is in general the better the more work there is to be done in collision handling (see [TPB07]). A further aspect to note is the performance of the different task creation strategies. The naive stationary approach does not scale well when compared to the randomized version, which already shows quite a good performance. The work estimate approach performs very good and can, depending on the actual scenario, outperform the randomized version.

References

[Bat96] BATHE K.-J.: *Finite Element Procedures*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1996.

[Bel00] BELYTSCHKO T.: *Nonlinear Finite Elements for Continua and Structures*. John Wiley, 2000.

[BFA02] BRIDSON R., FEDKIW R. P., ANDERSON J.: Robust Treatment of Collisions, Contact, and Friction for Cloth Animation. In *Proceedings of ACM SIGGRAPH '02* (2002), ACM Press, pp. 594–603.

[BHW94] BREEN D., HOUSE D., WOZNY M.: Predicting the drape of woven cloth using interacting particles. In *Proceedings of ACM SIGGRAPH '94* (1994), ACM Press, pp. 365–372.

[BKLwW99] BLOCHINGER W., KÜCHLIN W., LUDWIG C., WEBER A.: An object-oriented platform for distributed high-performance Symbolic Computation. vol. 49, Elsevier, pp. 161–178.

[BMF03] BRIDSON R., MARINO S., FEDKIW R.: Simulation of clothing with folds and wrinkles. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA 2003)* (2003), ACM Press, pp. 28–36.

[BW97] BONET J., WOOD R. D.: *Nonlinear Continuum Mechanics for Finite Element Analysis*. Cambridge University Press, 1997.

[BW98] BARAFF D., WITKIN A.: Large steps in cloth simulation. In *Proceedings of ACM SIGGRAPH '98* (1998), ACM Press, pp. 43–54.

[BWH*06] BERGOU M., WARDETZKY M., HARMON

D., ZORIN D., GRINSPUN E.: A Quadratic Bending Model for Inextensible Surfaces. In *Fourth Eurographics Symposium on Geometry Processing* (Jun 2006), pp. 227–230.

[BWK03] BARAFF D., WITKIN A., KASS M.: Untangling cloth. In *Proceeding of ACM SIGGRAPH '03* (2003), ACM Press, pp. 862–870.

[CK02] CHOI K.-J., KO H.-S.: Stable but responsive cloth. In *Proceedings of ACM SIGGRAPH '03* (2002), vol. 21, pp. 604–611.

[CK05] CHOI K.-J., KO H.-S.: Research problems in clothing simulation. *Computer-Aided Design 37* (2005), 585–592.

[COS00] CIRAK F., ORTIZ M., SCHRÖDER P.: Subdivision surfaces: A new paradigm for thin-shell finite-element analysis. *Journal for Numerical Methods in Engineering 47* (2000), 2039–2072.

[DHv93] DEMMEL J., HEATH M., VAN DER VORST H.: Parallel numerical linear algebra. In *Acta Numerica 1993*. Cambridge University Press, 1993, pp. 111–198.

[EB00] EISCHEN J., BIGLIANI R.: Continuum versus particle representations. In *Cloth Modeling and Animation*, House D., Breen D., (Eds.). A. K. Peters, 2000, pp. 79–122.

[EDC96] EISCHEN J., DENG S., CLAPP T.: Finite-element modeling and control of flexible fabric parts. *IEEE Computer Graphics and Applications 16*, 5 (Sept. 1996), 71–80.

[EKS03] ETZMUSS O., KECKEISEN M., STRASSER W.: A Fast Finite Element Solution for Cloth Modelling. In *Proceedings of Pacific Graphics* (2003), pp. 244–251.

[EWS96] EBERHARDT B., WEBER A., STRASSER W.: A fast, flexible, particle-system model for cloth draping. *IEEE Computer Graphics and Applications 16*, 5 (Sept. 1996), 52–59.

[GH*03] GRINSPUN E., , HIRANI A., DESBRUN M., SCHRÖDER P.: Discrete shells. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA 2003)* (2003), ACM Press, pp. 62–67.

[GKJ*05] GOVINDARAJU N., KNOTT D., JAIN N., KABUL I., TAMSTORF R., GAYLE R., LIN M., MANOCHA D.: Interactive collision detection between deformable models using chromatic decomposition. In *Proceedings of ACM SIGGRAPH '05* (2005), ACM Press.

[Hau04] HAUTH M.: *Visual Simulation of Deformable Models*. Phd thesis, Wilhelm-Schickard-Institut für Informatik, University of Tübingen, Germany, July 2004.

[HB00] HOUSE D. H., BREEN D. E. (Eds.): *Cloth Modeling and Animation*. A K Peters, 2000.

[HE01] HAUTH M., ETZMUSS O.: A High Performance Solver for the Animation of Deformable Objects using Advanced Numerical Methods. In *Computer Graphics Forum* (2001), pp. 319–328.

[Kaw80] KAWABATA S.: The standardization and analysis of hand evaluation. *The Textile Machinery Society of Japan* (1980).

[Kee99] KEELER S.: The science of forming: A look at bending. *Metal Forming Magazine* (October 1999), 27–29.

[KHM*98] KLOSOWSKI J. T., HELD M., MITCHELL J. S. B., SOWIZRAL H., ZIKAN K.: Efficient collision detection using bounding volume hierarchies of *k*-DOPs. *IEEE Transactions on Visualization and Computer Graphics 4*, 1 (/1998), 21–36.

[KK96] KARYPIS G., KUMAR V.: *Parallel Multilevel k-way Partitioning Schemes for Irregular Graphs*. Tech. Rep. 036, Minneapolis, MN 55454, May 1996.

[LAM01] LARSSON T., AKENINE-MÖLLER T.: Collision detection for continuously deforming bodies. *Proceedings of Eurographics '01* (2001), 325–333.

[LVDY04] LEE B. C., VUDUC R. W., DEMMEL J. W., YELICK K. A.: Performance models for evaluation and automatic tuning of symmetric sparse matrix-vector multiply. In *ICPP '04: Proceedings of the 2004 International Conference on Parallel Processing (ICPP'04)* (2004), IEEE Computer Society, pp. 169–176.

[MDSB03] MEYER M., DESBRUN M., SCHRÖDER P., BARR A. H.: Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and Mathematics III*, Hege H.-C., Polthier K., (Eds.). Springer-Verlag, 2003, pp. 35–57.

[MKE03] MEZGER J., KIMMERLE S., ETZMUSS O.: Hierarchical Techniques in Collision Detection for Cloth Animation. *Journal of WSCG 11*, 2 (2003), 322–329.

[MTVW*05] MAGNENAT-THALMANN N., VOLINO P., WACKER M., THOMASZEWSKI B., KECKEISEN M.: Key Techniques for Interactive Virtual Garment Simulation. In *Proc. of Eurographics 2005, Tutorial 4* (2005).

[OBHL02] OLIKER L., BISWAS R., HUSBANDS P., LI X.: Effects of ordering strategies and programming paradigms on sparse matrix computations. *Siam Review 44:3* (2002).

[Pro95] PROVOT X.: Deformation constraints in a mass-spring model to describe rigid cloth behavior. In *Proceedings of Graphics Interface (GI 1995)* (1995), Canadian Computer-Human Communications Society, pp. 147–154.

[Pro97] PROVOT X.: Collision and self-collision handling in cloth model dedicated to design garments. In *Proceedings of the Eurographics Workshop on Computer Animation and Simulation (CAS 1997)* (1997), Springer-Verlag, pp. 177–189.

[RS01] REIF U., SCHRÖDER P.: Curvature integrability of subdivision surfaces. *Advances in Computational Mathematics 14*, 2 (2001), 157–174.

[Saa03] SAAD Y.: *Iterative Methods for Sparse Linear Systems*, 2nd ed. SIAM, 2003.

[SFR89a] SIMO J. C., FOX D. D., RIFAI M. S.: On a stress resultant geometrically exact shell model. part i: Formulation and optimal parametrization. In *Computational Methods in Applied Mechanics and Engineering* (1989), vol. 72, pp. 267–302.

[SFR89b] SIMO J. C., FOX D. D., RIFAI M. S.: On a stress resultant geometrically exact shell model. part ii: The linear theory; computational aspects. In *Computational Methods in Applied Mechanics and Engineering* (1989), vol. 73, pp. 53–92.

[She94] SHEWCHUCK J. R.: An Introduction to the Conjugate Gradient Method Without the Agonizing Pain, 1994.

[TB06] THOMASZEWSKI B., BLOCHINGER W.: Parallel simulation of cloth on distributed memory architectures. In *Eurographics Symposium on Parallel Graphics and Visualization (EGPGV '06)* (2006).

[TB07] THOMASZEWSKI B., BLOCHINGER W.: Physically based simulation of cloth on distributed memory architectures. *Parallel Computing 33* (2007), 377–390.

[THM*05] TESCHNER M., HEIDELBERGER B., MANOCHA D., GOVINDARAJU N., ZACHMANN G., KIMMERLE S., MEZGER J., FUHRMANN A.: Collision Handling in Dynamic Simulation Environments. In *Eurographics Tutorials* (2005), pp. 79–185.

[TPB07] THOMASZEWSKI B., PABST S., BLOCHINGER W.: Exploiting parallelism in physically-based simulations on multi-core processor architectures. In *Eurographics Symposium on Parallel Graphics and Visualization (EGPGV '07)* (2007).

[TPBF87] TERZOPOULOS D., PLATT J., BARR A., FLEISCHER K.: Elastically deformable models. In *Proceedings of ACM SIGGRAPH '87* (July 1987), ACM Press, pp. 205–214.

[TWS06] THOMASZEWSKI B., WACKER M., STRASSER W.: A consistent bending model for cloth simulation with corotational subdivision finite elements. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA 2006)* (2006), Eurographics Association, pp. 107–116.

[van97] VAN DEN BERGEN G.: Efficient collision detection of complex deformable models using AABB trees. *Journal of Graphics Tools: JGT 2*, 4 (1997), 1–14.

[VCMT95] VOLINO P., COURCHESNE M., MAGNENAT-THALMANN N.: Versatile and efficient techniques for simulating cloth and other deformable objects. In *Proceedings of ACM SIGGRAPH '95* (1995), ACM Press, pp. 137–144.

[VMT94] VOLINO P., MAGNENAT-THALMANN N.: Efficient self-collision detection on smoothly discretized surface animations using geometrical shape regularity. In *Proceedings of Eurographics '94* (1994), Computer Graphics Forum, pp. 155–166.

[VMT01] VOLINO P., MAGNENAT-THALMANN N.: Comparing efficiency of integration methods for cloth simulation. In *Proceedings of Computer Graphics International 2001 (CGI 2001)* (2001), IEEE Computer Society, pp. 265–274.

[VMT05] VOLINO P., MAGNENAT-THALMANN N.: Accurate garment prototyping and simulation. *Computer-Aided Design & Applications 2*, 5 (2005), 645–654.

[VMT06a] VOLINO P., MAGNENAT-THALMANN N.: Resolving surface collisions through intersection contour minimization. In *Proceedings of ACM SIGGRAPH '06* (2006), vol. 25, ACM Press, pp. 1154–1159.

[VMT06b] VOLINO P., MAGNENAT-THALMANN N.: Simple linear bending stiffness in particle systems. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA 2006)* (2006), Eurographics Association, pp. 101–105.

[WBH*07] WARDETZKY M., BERGOU M., HARMON D., ZORIN D., GRINSPUN E.: Discrete Quadratic Curvature Energies. *Computer Aided Geometric Design (to* (2007).

[WW98] WEIMER H., WARREN J.: Subdivision schemes for thin plate splines. *Computer Graphics Forum 17*, 3 (1998), 303–314. ISSN 1067-7055.

[ZT00a] ZIENKIEWICZ O. C., TAYLOR R. L.: *The Finite Element Method. Volume 1: The Basis*, 5th ed. Butterworth Heinemann, 2000.

[ZT00b] ZIENKIEWICZ O. C., TAYLOR R. L.: *The Finite Element Method. Volume 2: Solid Mechanics*, 5th ed. Butterworth Heinemann, 2000.

Advanced Topics in Virtual Garment Simulation

Part 2

Real world fabrics and Virtual Try On

E. Lyard[1], C. Luible[1] , P. Volino[1], M. Kasap[1], V. Muggeo[1] and N. Magnenat-Thalmann[1]

[1]MIRALab, University of Geneva, Switzerland

1 Real World Fabrics (Accuracy of cloth simulations)

Fabrics are complex visco-elastic materials. They must have sufficient strength and at the same time they have to be flexible, elastic and easy to pleat and shape. Their simulation is not easy, as their behaviour is difficult to describe and predict. Nevertheless, computation algorithms have been developed over many years and evolved to such a level so that today we are able to not only simulate simplified, static clothes, but also complex dynamically moving garments, in the time frame, expected by the clothing industry[VM05]. But, not only advanced computational models are responsible for precise virtual garment simulations. Also exact input parameters play an important role for a correct reproduction of the fabrics mechanical behaviour. For instance, newly-developed computation systems finally allow the simulation of the non-linear fabric behaviour; but in order to truly reflect these characteristics in the virtual computation, we have to be able first of all to measure them appropriately. Experimental values for the main mechanical and physical parameters can be derived from standard fabric characterization experiments such as the "Kawabata Evaluation System for fabrics" (KES) [KAW80] and the "Fabric Assurance by simple Testing" (FAST) [MIN95]. However, both characterisation methods have not been designed for the purpose of deriving parameters for virtual simulations.

1.1 Mechanical properties of cloth

The mechanical behaviour of fabric reflects the nature and molecular structure of the fiber material constituting the cloth. The arrangement and orientation of the fibres in the fabric structure has its influence as well. Fabric fibres can be organized in several ways. The main structures are:

- Woven Fabrics: threads are orthogonally aligned and interlaced in an alternating way 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 fibres are arranged in an unstructured way, such as paper fibres.

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

- The nature of the fibre: wool, cotton, synthetic, etc.
- The thread structure: diameter, internal fibre and yarn structure, etc.
- The thread arrangement: woven or knitted, and particular pattern variation.
- The pattern properties: tight or loose.

These properties determine the stiffness of a material, its ability to bend, and its visual appearance. The mechanical properties of deformable surfaces can be grouped into four main families:

- Elasticity, which characterises the internal forces resulting from a given geometrical deformation.
- Viscosity, which includes the internal forces resulting from a given deformation rate.
- Plasticity, which describes at which point of deformation irreversible material changes occur.
- Resilience, which defines the limits at which the structure will break.

Most important are the elastic properties, which 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 be sufficient 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 to the evolution of deformation are also needed, and complete dynamic models including viscosity and plasticity should be used. Depending on the amplitude of the mechanical phenomena under consideration, 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 characterisation and modeling of mechanical phenomena.

While the linear laws described in the Section 3.5, Part 1, are valid for small deformations of the cloth, large deformations usually lead to nonlinear response of the cloth. In this case the dependence between stress and strain is no more linear. This effect is usually manifested as a stiffening of the cloth as the deformation increases. This can possibly be followed by rupture (resilience) or persistent deformations when 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 by nonlinear strain-stress functions.

For the practical measurement of mechanical properties of fabrics there are many different standards and instruments. This subject is discussed in detail in the next section.

1.2 Measurement of fabric properties

Each textile possesses typical characteristics, influenced by the textiles raw material (natural fibres, synthetics, etc.), yarn structures (degree of twist), planar structure (weave, knit) and finishing treatment, which are advantageous for some types of garments, but can be unfavourable for others, regarding garment comfort. For an optimal usage of each type of fabric, the garment and textile industry invented the concept of “fabric hand”, what is an assessment process, where each textile is evaluated regarding its quality and suitability. The fabric hand attributes can be obtained through subjective assessment or objective measurement. Objective fabric characterization methods measure and relate the major mechanical properties, in order to obtain comparable information about textiles. The applied physical tests analyse and reflect the sensations felt during the subjective fabric assessment, where the textile is touched, squeezed, rubbed or otherwise handled and describe them with a numerical value[AAT02]. Important fabric hand properties are flexibility, compressibility, elasticity, resilience, density, surface contour (roughness, smoothness), surface friction and thermal attributes, which are the result of a broad fundamental research on fabric properties[PIE30][LD61]. In virtual simulations, the main imitated mechanical properties are elasticity, shear, bending, density and friction. The KES system was developed in the 1970’s by Kawabata and constituted the first standardization of objective fabric assessments. Since, this method is widely used for the objective characterisation of fabrics, as well as for studies of fabric mechanical properties. In the late 1980´s the CSIRO Association in Australia realised the importance of a commercial measurement for wool fabrics and tried to offer a simpler and cheaper alternative to KES, the FAST method. Both, KES and FAST measure the same parameters; however different measurement principles are applied. The FAST method uses simpler procedures than KES and permits only a linear interpretation of the measured data, whereas KES provides a complete stress-strain profile for all measured characteristics. The measurements of both systems are conducted in the low force area, what corresponds to loads which a fabric is likely to undergo during garment manufacturing. Alternative, more flexible measurement devices exist for the measurement of tensile and hysteresis properties.

Elasticity tests are designed in a way to return the correlation between applied forces and corresponding fabric elongations. The FAST method measures the elasticity property at one load of 100 N/m along warp and weft direction. KES tests the tensile behaviour with an increasing force of up to 500 N/m also along weft and warp direction. After the tensile load attains the maximum force, the recovery process is recorded.

During shear tests, the required forces to change the angle between the orthogonal intersecting threads of textiles to certain extend are assessed. Whereas the tensile property is more influenced by the fabrics fibre composition and the yarn structure, the shear characteristic is mainly influenced by the fabric structure. Different measurement principles can be applied. The main standard method fixes the fabric between two clamps and applies opposite forces until a maximum angle (KES [KAW80]) or maximum force is attaint. Other methods measure the fabrics extension-compression in the bias direction (FAST [MIN95]) [BEH61].

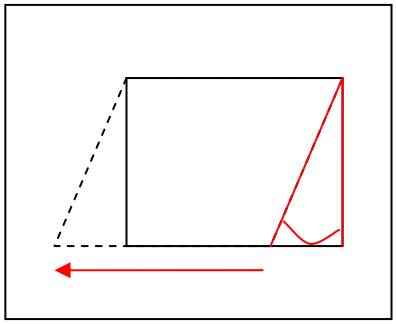

Figure 1: Angle-force method

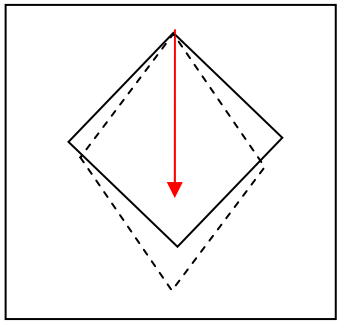

Figure 2: Extension in bias direction

Regarding fabric bending tests, there are two main categories. The first category measures the bending deformation under the fabrics own weight. The most well known method within this category is the Cantilever test, which uses the engineering principles of beam theory. A fabric is moved forward to project as a cantilever from a horizontal platform. As soon as the leading edge of the fabric reaches an angle of 41.5° to the horizontal platform the bending length is measured (FAST). Another method of this category consists in the folded loop method, where the fabric is fold back on itself and the height of the loop measured. The second category of bending tests is designed to return the moment-curvature relationship by measuring forces or moments (KES). Therefore, a fabric is fixed between two clamps and bent in an arc of constant curvature, where the curvature changes continuously and applied moments recorded.

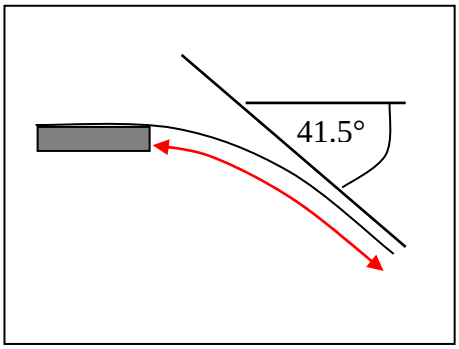

Figure 3: Cantilever method

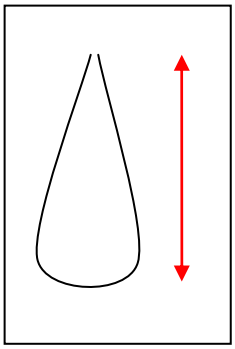

Figure 4: Folded loop method

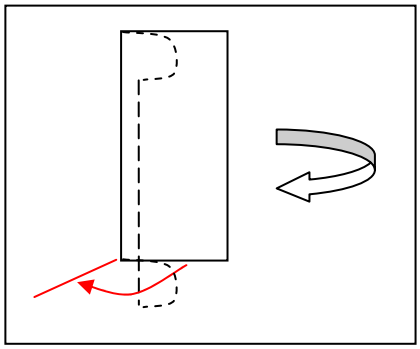

Figure 5: Moment-curvature method

In contrast to elasticity, shear and bending, the friction property is not an internal but external mechanical force, varying with each other contact material. Distinctions are made between static and dynamic friction. The static friction is related to the initial force, what is needed to overcome to start moving a material against another object or surface. The dynamic friction however occurs during the movement itself and is therefore generally lower than the initial static friction. There are several methods to assess friction. Within the standard fabric characterisation experiments (KES), the friction is measured by moving a piano-wire over the fabric at a constant force frequency. Friction is not measured by the FAST system.

1.3 Mapping to Computational Models

For virtual garment simulations not the calculated standard fabric hand values, but the actual measured empirical data is of interest, as therefrom important mechanical input parameters can be derived. For the actual derivation of fabric input parameters, a mathematical description of the measured data is needed. Depending on the complexity of the implemented computational model and the available amount of measured data, this mathematical interpretation can be linear (from FAST) or non-linear (from KES and alternative devices) derived.

The tensile behaviour of fabrics is strongly non-linear for most textiles. Strain-stress profiles of very elastic materials are particularly characterized by a curved envelope. Versatile computational models are able to simulate the nonlinear tensile behaviour and therefore ask for an adequate input data. Linear parameters derived from the measurement data from FAST, are correct in the low force area, occurring for example during static simulations, where the fabric is basically stressed by its own weight. However, linear parameters are incorrect for the simulation of higher stresses, as referring to them much lower loads are sufficient to achieve larger fabric elongations. Moreover, used in virtual garment fitting processes, linear parameters return a wrong feedback about the garment comfort.

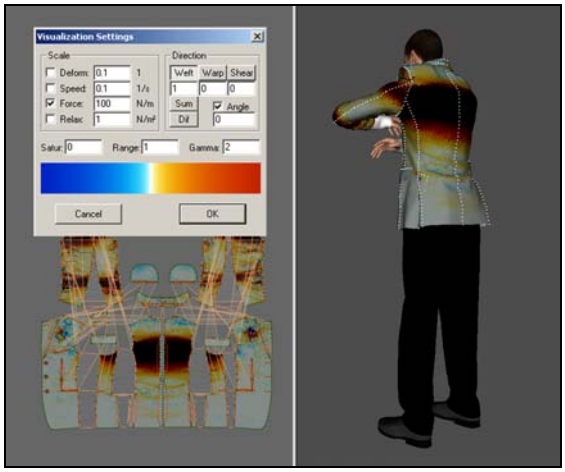

Figure 6: Simulation using KES tensile data

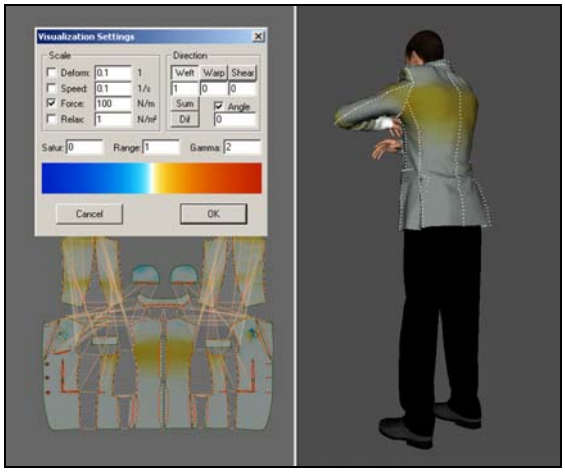

Figure 7: Simulation using FAST tensile data

At a first glance, the non-linear fabric parameters derived from the KES strain-stress envelopes seam to be better suited for the derivation of nonlinear tensile parameters. However the KES method is limited as well, as it returns the strain-stress profile only up to one maximum load. Dynamic cloth simulation is a much more complex issue. When the garment follows the movement of the mannequin, the fabric undergoes not only one but many deformations and relaxations of various low and high loads in different temporal distances. For that reason, derived parameters from KES are accurate for the one specific measured force (500 N/m), but not for various loads. Hence, for dynamic simulations, multiple load/unload experiments with different applied forces, which reflect what actually happens during the wear of a garment, are needed. Also aspects which are related to the simulation history such as plasticity and properties which are time related such as viscosity, become important input characteristics for dynamic garment simulations. Until today, the viscosity of textiles is a little investigated field of research and no standard measurement exists for the characterisation.

Depending on the type of fabric material, the shear behaviour varies from linear to non-linear. State of the art simulation systems use a nonlinear shear computation model. Therefore, depending on the fabric material, a linear or nonlinear mathematical description is needed (Figure).

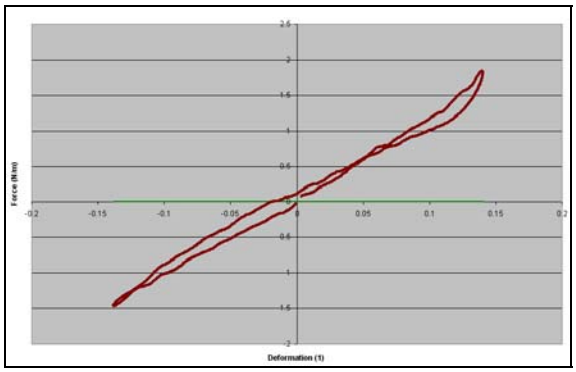

Figure 8: Linear shear strain-stress envelop

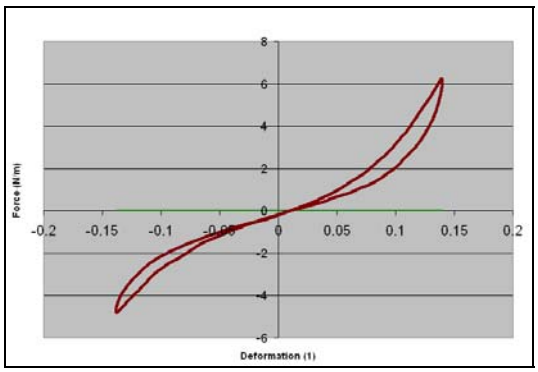

Figure 9: Non-linear shear-strain-stress envelope

Similar to the tensile property, the nonlinearity of the shear behaviour can be more accurately derived from complete strain-stress envelopes, whereas the more linear shear comportment can be accurately interpreted from single measured forces as well. However, in contrast to the tensile property, the error in the comfort feedback for simplified nonlinear parameter is smaller. This is related to the fact that regarding shear, generally lower forces are concerned.

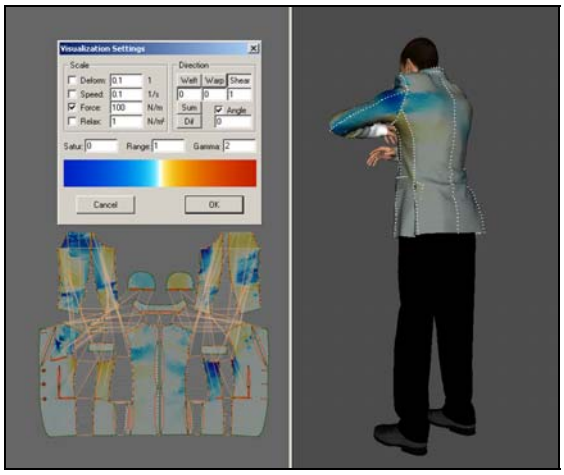

Figure 10: Accurate nonlinear shear parameter

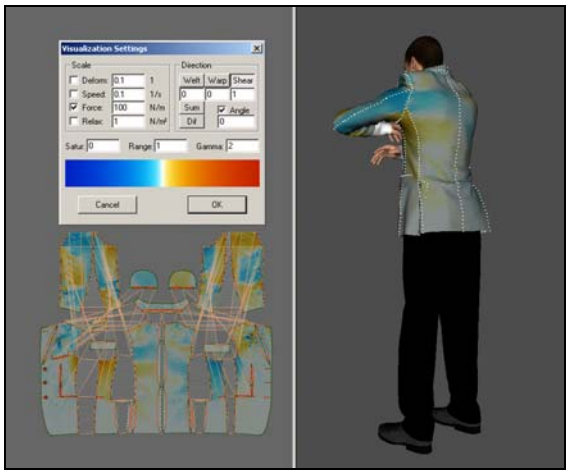

Figure 11: Simple linear shear parameter

Even in versatile computation system, the complex bending property is still linear modelled. Thus, a simple (linear) mathematical interpretation for the bending behaviour is precise enough. The bending rigidity is returned by standard measurement methods as characteristic value. As this measure is a description of the slope between two major points of the measured data, it is suited to be directly taken as linear bending characteristic. The comparison of the bending rigidity obtained from the FAST and the KES measurement systems shows a good correlation for the bending rigidity value.

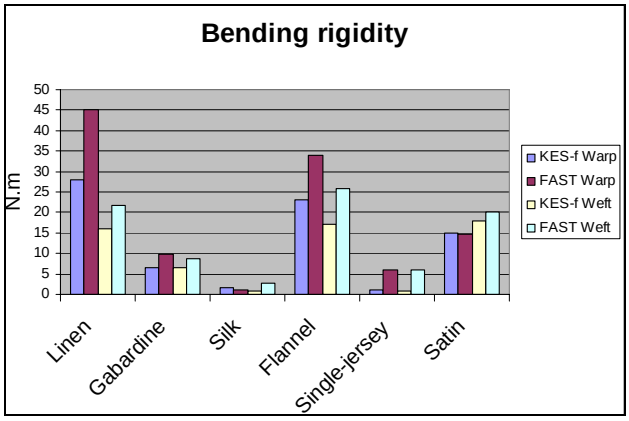

Figure 12: Bending rigidity FAST and KES-f

Figure 13: Virtual drape of the flannel fabric

1.4 Multi-Layered Fabrics and Seams

Regarding complete garments, not only the fabric characteristics are important for a mechanical accuracy and a good visual appearance. Additional clothing aspects, such as seams, interlinings and fabric fusing become important influencing factors for virtual computations and demand a separate examination.

Figure 14: men suits

Figure 15: fitting men suit

Figure 16: virtual simulated men suit

1.4.1 Seams

Conventional garments are generally not composed out of only one, but many different pieces of fabrics. On the one hand, this is due to the complexity of the shape of the human body with its curves and bulges, which need to be covered by a fitted garment. On the other hand, this is related to changing tendencies and trends, which constantly ask for new silhouettes. Hence, as a garment is composed out of multiple single surfaces and they have to be somehow connected subsequently, in order to form a complete garment. The mechanical behaviour of a single textile and the behaviour of a tailored garment out of the same fabric are different as the characteristics of the junctions of the single surface pieces influence the general comportment as well. For the combination of two fabric pieces there several different methods:

The traditional way of combining two fabric pieces is by applying sewing techniques, where two or more surfaces are linked together with threads. Hereby, distinction can be made between different types of seams, such as the plain seams, the welt seams, the welding seams, decorative seams, etc. Their mechanical characteristics vary according their amount of fabric layers and their amount stitches and topstitches. The plain seam consists of two fabric layers, the outer fabric and the seam allowance (Figure) and is mainly used for standard fabric assemblies. The welt seam is composed out of three or more fabric layers, the outer fabric and two times or more the folded seam allowance, completed by one or two top-stitches (Figure). It is mainly used for parts of the garment, where a lot of abrasion is expected and also where additional stability is needed, as for example at the inner side of a pair of jeans. Modern high tech textiles, especially water proof garments are welded instead of sewed, as the stitches would impact their performance. Decorative seams are stitching in various patterns on top of the outer fabric. Whereas the decorative seams influence less the mechanical behaviour, the seams containing multiple fabric layers can change the fabric comportment significantly. Therefore, if we want to accurately imitate those parts of the garment, the seam mechanics need to be measured and accurately simulated as well. The stiffness of seams is an additional parameter, which can be specified inside the simulation application for each seam. Their characteristics can be obtained with the above described fabric measurement methods.

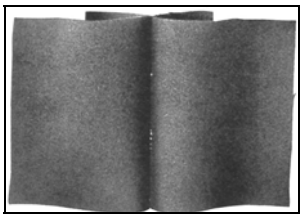

Figure 17: Plain seam

Figure 18: Welt seam with top-stitch

Another important aspect regarding accurate seaming simulations is the problematic of seam pucker. Seam pucker is the occurrence of unwanted small fabric wrinkles at a garment's seam, due to fabric gatherings caused by the sewing thread. Depending on the fabrics thickness and stiffness, this shrinkage can be up to 5% of the length of a seam. For an accurate virtual simulation, it is important to consider this parameter, as it influences the fit and especially the quality of the garment.

Figure 19: Effect of seam pucker

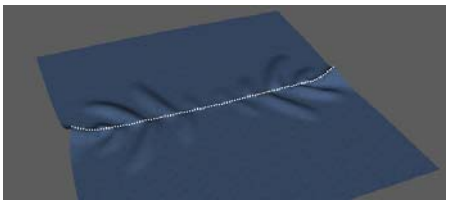

Figure 20: Virtual simulation of seam shrinkage

1.4.2 Multi-layered fabrics

In order to give to a garment more stability in some areas for functional or aesthetical reasons, a second fabric layers can be added. This second fabric layer is either permanently fixed to the outer fabric (fusing) or it is a loose additional textile (interlining). Regarding the permanently fixed fabric layer it is clear that the mechanical characteristics of the bonded combination of the two materials should be measured for their accurate virtual re-creation. However this can be easily accomplished by measuring not the single but the fused fabrics.

For the non-fixed fabric layers this is however a more difficult task. On the one hand the combination of both characteristics is needed, but on the other hand, the fabrics are single layers and for an accurate prototyping they should be treated separately. A simulation of two textile layers would cause two main problems. The multi-layer garment simulation asks for an important feature, inevitable for those challenging calculations, such as a powerful collision response method with stability and robustness. However, the high amount of polygons is visibly slowing down the simulation.

Today's collision algorithms also do not allow the simulation of various layers in small distances, such as the thickness of interlining fabric. Thus, because of the fairly unnatural distance of multilayer fabrics, the simulation looks unrealistic. Because of these two aspects it is suggested to also simulate the non-fixed fabric layers with only one virtual surface, using mechanical characteristics which are a combination of materials, even the simulation of an endless amount of fabric-layers is possible with today's simulation systems from a technical point of view.

We've just seen how to efficiently measure the physical properties of a cloth in order to produce high fidelity virtual simulations. This allows to estimate accurately the fitting of a given outfit on someone's avatar. However, if one wants to remain high fidelity, then the avatar must correspond exactly to the person's dimensions, thus the need to an efficient avatar deformation techniques, which is the topic of the next section.

2 Towards an Integrated Virtual Try On Application

2.1 Parametrically Deformable Human Bodies

3D human body models are the most important accessories of the computer graphics environments such as games, virtual modelling tools and virtual reality applications. Evolution of the computer graphic techniques and hardware technology let it possible to use more realistic models in those environments which use muscle and fat tissue deformation effects during animation. The primary characters of those environments, human body models, require specific techniques for real-time animation and realism. Because of their importance, specialized research areas focused on face, hand, skin, muscle modelling and skeleton attachment to generate realistic models which will improve the quality and realism of the environments. Human body models which are subject to all those techniques and environments are initially generated by 3D model designers or acquired by 3D body scanners like the ones from Human Solutions [HS:07]. These initial models are the template ones for the future deformation methods.

In recent 3 decades, human body deformation techniques evolved under the following three main headings: geometric, example-based, and physics-based. The geometric deformation techniques are not only the fastest but also the simplest ones. On the other hand result of the geometric deformation technique is not satisfactory to generate realistic models. Because of their computational simplicity and less requirements, it

is still the most preferable technique in real time animation. Example-based approaches are the most challenging field after effective use of the 3D body scanners. Based on a model database, the requested model interpolated from the database with appropriate parameters. Main constraint of this approach is that it requires number of post processed complex data as an input. Lastly, physics-based deformation is computationally the most complex method but it generates the most realistic results.

Initially, Lasseter [Las87] mentioned about the multi-layered approach for animation, which reduces the modelling complexity but increases the reality. Lately, to achieve higher degree of realism, multi-layered approach is used with the combination of deformation techniques to generate a body model with skeleton, muscle and fat tissue layers. From this point of view, Chadwick [CHP89] is one of the early initiator in this area with modelling muscle layer for deformation during animation. Recently, realistic body modelling techniques are based on multi-layered approach integrated with hardware acceleration for real-time computation.

2.1.1 Geometric Deformation

History of the geometric deformation starts with Barr [Bar81], where he applied basic transformations on super-quadrics. Recently these super-quadric ellipsoids are used to simulate the muscle effects on body model. After his successive work on super-quadrics, Barr [Bar84] developed hierarchical solid modelling operations that simulate twisting, bending, tapering like transformations on geometric objects. Blanc [Bla94] proposed the procedural generic implementation of these global deformation techniques.

Based on Barr's work, Sederberg and Parry[SP86] announced the Free Form Deformation (FFD) technique also today which is commonly used and have lots of extensions to solve specific problems related with deformation. As stated by the author "A good physical analogy for FFD is to consider a parallelpiped of clear, flexible plastic in which is embedded an object, or several objects, which we wish to deform. The object is imagined to also be flexible, so that it deforms along with the plastic that surrounds it."

In the first row, a set of original geometric models enclosed with a cube and the deformation of these models along with the enclosed cube could be observed. In the second row just the enclosing cube and the transformation over it illustrated. Using this technique with its basic form, people tried to generate complex (for that day) models.

Sederberg and Parry demonstrated a possible application of FFD method by modelling a handset from a stick. This approach used in many graphic applications for deforming complex models. One of the most interesting applications for that day is developed by Chadwick[CHP89]. Using robotics skeleton approach for animation of the human body model, he added muscle effect on top of the skeleton with FFD representation. For this animation frames, Chadwick used basic adjoining FFDs to achieve the real-time deformation. Parametric muscle deformation generated according to Denevit and Hartenberg parameters [HD55] related with the joint angles in robotics.

In the same time period, Thalmanns[MTT87] developed joint-dependent local deformation (JLD) operators to deform body model surface for animation. Each JLD operator is affecting its uniquely defined domain and its value is determined as a function of angular values of the joints under the operator.

Figure 21: JLD based animation of body models, Thalmanns[MTT87].

Since FFDs are based on a parallel piped cubic volume covering the model, they had some limitations. First, it is not possible to cover a complex model without sparse sub-volumes. Second but not the least, deformation with more freedom is not possible. Coquillart[Coq90] come with the extended version of the standard FFD method that is called ExtendedFFD. This new method uses non-parallel piped 3D lattices to include the shape of the deformation.

To have more flexibility on the deformation, Lamousin[LWN94] logically extends the FFD by mapping it on a non-uniform rational B_Splines(NURBS) and called this technique NurbsFFD. NFFD offers much more control on the model which is not achieved in the prior implementations. An interesting application of NFFD is demonstrated by the author with human leg model as an input.

Moccozet[MT97] even extends the EFFD by presenting a generalized method with techniques of scattered data interpolation based on Delaunay and Dirichlet/Varonoi diagrams. This is why it is called DirichletFFD. One of the advantages of this technique is the control of local deformations where it is crucial for 3D modelling and animation. Author implemented a multi-layer deformable hand model to simulate the intermediate layer between the skeleton and the skin.

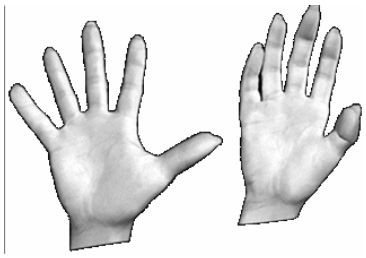

Figure 22: DFFD applied hand model deformation, Moccozet[MT97].

Originally human body model animation is generated from the point of view as for robots. This non-realistic motion implementation method improved with new deformation techniques. Early initiators of realistic motion generation, Thalmanns[MTT91], applied two methods for improving deformation during animation. For the body parts contacting with other objects, finite element method(FEM) is used, for the parts that are not contacting JLD[MTT87] used for simulating the natural behaviour of the human body model.

One of the earlier researches about anthropometric modelling of the human body model is introduced by Azuola [ABH*94]. First the human body model is segmented into groups according to the synovial [BPW93] joints. Deformation on the corresponding joint is constrained with its degree of freedom (DOF). Joints with one to three DOF selected with their movement limitations. According to the anthropometric measurement database, anthropometrically segmented body model deformed with FFD methods.

Jianhua[JTT94a] used a new approach for body model representation. Instead of polygonal representation, model divided into slices. Each slice is defined with parametric curves namely b-splines. Depending on the neighbouring joints distance and the normal vectors of the slices, collision prevented. Radius of each contour is scaled to achieve the muscular deformation effect.

Jianhua[JTT94b] extended contour based representation of body model with metaballs[HMT*85]. Like cylindrical representation, metaballs are used for smooth and detailed modelling.

One of the pioneering work in the anatomical modelling field is recognized by Wilhelms[WG97] work. Apart from body mesh, underlying muscle, skeleton and generalized tissues are also parametrically modelled according to the anatomical principals. Though detailed modelling of underlying layers of the skin, animation became more realistic compared to its preceding approaches. By his complete multi layered modelling work, Wilhelms make a big impact on the future body modelling techniques.

Similarly Scheepers [SPCM97] developed multilayered anatomically deformable models in the same time period. Tubular shaped bi-cubic patch meshes capped with elliptic hemispheres attached on both end of the corresponding skeleton. Depending on the corresponding joint angle, underlying muscle and fat tissue structures deformed the skin surface.

Dougles [DMS98] developed a system based on the anthropometric statistic of human face measurements in a population to model the face. Statistical results are used as geometric constraints on the specific part of the parametric surface. Because of its parametric property, variational modelling [GC95] technique is selected to deform the surface which satisfies the underlying constraints.

Singh [SF98] introduced a new deformation technique based on free form curve. Author inspired by armatures used by sculptures. In this approach with a single wire, direct manipulation for deformation on the model is possible. Also interacting multiple wires with accumulation is possible to generate more complex deformations.

While most of the geometric techniques inspired from FFD, Singh[SK00] in contrast proposed a surface oriented FFD. He implemented a control surface defined by a distance function around the surface. This new approach allows localization of the control lattice complexity for detailed deformation. Furthermore, this make the approach ideally suited to the automatic skinning.

Marinov [MBK07] present an efficient technique based on multi-resolution deformation on a high resolution meshes. Proposed deformation process handled in the GPU with the help of pre-computed deformation operators and the gradient information. By this way dynamic 3D model deformation succeeded with several times faster computation. After the deformation, for calculation of the new normal field, he provided neighbouring vertex and other required information as vertex attributes. For deformation process, basically affine transformation operators applied on the control points.

In contrast with the previous work on muscle modeling, Dong at al[DCKY02] take a further step on realistic muscle modelling. Using Visible Human Dataset[VH:95], horizontal human body slice images are post-processed to extract main muscle contours. After contour extraction, cubic B-spline surfaces are built to fit on the contours. Dong et al used a new deformation procedure for dynamic muscle shape forming. Previously generated muscle geometries attached on the skeleton, then according to the action lines and the joint angles new position of the muscle endpoints calculated. With the new muscle endpoints, direction of the tangent vectors reoriented to change the shape of the geometry.

One another approach for human body deformation is applying the sweep based method on the limbs [HYKJ03]. In this approach each limb is approximated by swept based ellipsoid which changes its size as it moves through the limb. Transition from each joint, the ellipsoid changes its orientation through the new

one. All ellipsoids interpolated to fit in the original model. Resulting approximated model processed with displacement map to reflect the original shape because of its smooth behaviour.

Recent geometric muscle deformation technique presented by Pratscher [PCLS05] by using multi-shell structured ellipsoids. Each shell has its own level of hardness for deforming the attached skin. Using number of heuristics, body mesh is partitioned into segments to determine the location of the muscles. User has the opportunity to customize the muscle connections, size etc. and those parameters saved under a musculoskeletal template then which can be applied on different bodies. Muscle mapping on the body with single or multiple pose is also considered.

Kavan at al [KZ05] proposed a skinning approach which is alternative to the standard linear blend skinning. They use spherical deformation methods to improve the side effects of the previous method. Main advantage of the method is to spherically deform the joint parts to overcome the collapsing-joint artifacts. Theoretical properties of the rotational interpolation are the basics of the spherical blend skinning and computationally require the similar system resources as the previous method.

Hyun[EHW*07] extended his previous sweep based approach for body deformation by adding GPU assisted collision detection for limbs during deformation. Given polygonal mesh approximated by control sweep surfaces and according to the joint angle changes, sweep surfaces deformed and overlapping parts are blended. Some anatomical features like elbow-protrusion, skin folding etc. are emulated in the GPU.

Cross sectional representation of the body model used by Zhaoqi at al[ZTS06] to generate skeleton-driven deformations. Even this approach seems to be very similar to the one proposed by Shen [ST95], Zhaoqi at al generalizes the cross sectional contours to preserve the original details of the body.

Park at al [PH06] developed a novel approach for skin deformation with visually realistic results. They attached very large number of markers (~350 markers) on a human body. With those markers they motion captured the body. Resulting data applied on a virtual model to reflect visual details such as muscle deformation. Since it requires huge number of markers it is not practical and easily repeatable but the good point would be that once the data captured it can be applied to other models.

Li[LW07] presented an approach for automatic creation of the human body skeleton with controllable parametric structures. Anatomical features of the body determined according to the contour based model data. According to the extracted features, appropriate size skeleton model is constructed. Over the skeleton model, multi resolution parametric sweep based surface is generated for controllable body model.

One of the earlier surveys on human body model CAESAR is also the subject to human body deformation field. Recently Wang [WR07] used this database for animating the static scanned data. This database is not only consisting of body models but also the corresponding landmarks and feature points of individual models. From this point of view, Wang at al attached h-anim skeleton on the models by using the landmark information. They developed a web based system for parametric manipulation of the models in this database.

Recent work for body animation is proposed by Aguiar at al[dATSS07]. Using 3D human body scanner, they rapidly attach a skeleton on the scanned model for animation. Their system tested with markers and marker free scanned data. They employ a Laplacian mesh deformation schema which is based on the marker information to compute the poses of the model.

2.1.2 Physics based deformation

One another approach for anatomically-based muscle modelling is coming from Nedel[NT98a]. In his approach muscles are attached on to the skeleton as usual but the muscles are represented by two layers called the action lines and the volumetric forms. Modifications on the action lines are automatically captured and used for further muscle deformation. Also mass spring systems are used for muscle modeling.

In his latter work, Nadel[NT98b] extended his previous approach by modelling the muscles with physical methods. To simulate physical deformation, he used mass-spring system with new kind of springs called "angular springs". By this way he achieved more control over the muscle volume to generate realistic visual effects.

Aubel[AT] proposed a new automated skin deformation method based on multi-layered human model. Deformations applied by considering the physiological and anatomical structure of the model's skin. 1D mass-spring systems used to achieve the muscle motion and deformation. Also the muscle layer covered with viscoelastic fat layer to generate dynamic effects during the animation. Based on this implementation Aubel[AT01] extended his method to let the designers have full interaction on the model by dynamically changing the muscle shape.

Capell [CGC*02] at al implemented dynamic skeleton driven deformation by using equations of motion of elastic solids. Volumetric finite element mesh is used to perform the deformation with few constraints. i.e. line constraints for skeleton representation and the edges of the volumetric mesh coincident with the bones. They linearize the non-linear motion equations to be solved over volumetric regions associated with each bone.

One another physics based muscle modelling approach presented by Teran at al [TSB*05]. It is one of the most detailed modelling approaches where muscle material heterogeneity also considered. Segmented visible human data set [VH:95] used to extract the real muscle shape format which will be the base envelope for the proposed model. Once the muscle shape constructed, it is filled with tetrahedrons to apply physical behaviour by means of FEM. Because the tendons and the muscle belly behave differently in its density, different parts of the model simulated with appropriate tetrahedron material properties. Result of the simulations became much more realistic then the previous developments.

Application of physical phenomena to improve the flesh parts of the body not only constrained to the muscles. Larboulette[LCA05] proposed a fast and simple technique to enhance the standard character skinning method by adding dynamic skin effects through the underlying skeleton. Method called rigid skinning is applied on the existing skinning information without modifying its kinematic deformations or other post processed data. Efficiency of this technique is coming from its real-time applicability of visco-elastic properties on the body parts.

One of the recent attempts for realistic muscle modelling is represented by Zhou at al [ZL05]. Geometric and physical modelling techniques are combined to simulate real muscle effect by using NURBS based Galerkin method. From visible human dataset[VH:95], 3D reconstruction of the muscle shape achieved by fitting the result into a Nurbs form. With simple FEM properties and a few control points it is possible to deform the muscle model during the animation.

While most of the researches are focused on the muscle modelling for realistic skin deformation, Capell at al [CBC*05] showed that the character rigging is possible to animate the model with high level parameters. Dynamic elastic bodies are subject to the application area of this method. Using force based rigging to deform the character, underlying collisions handled with a novel approach proposed by them.

2.1.3 Example Based Deformation

Since the human body scanners became widely used, it is possible to generate visually realistic models. Recent body scanner systems have the capability of capturing high resolution data along with the texture information. Apart from such benefits, these systems generate static models which require post-processing stages to let them available for further deformation and animation. In the last decade we observe key attempts to solve such problems and benefit from scanner systems high resolution data generation capabilities. An early research on this field carried out by Seo [SMT03]. She proposed a method consisting of 3 main steps for parametrically synthesize visually realistic human body models: Pre-processing the 3D scan data, function approximation of the parameterized modeller and the runtime evaluator. Together with set of scanned human body data, template model with appropriate skeleton attachment designed. For parametric deformation, specific landmarks over the template model determined. Same landmarks also specified on the scanned data set. Regarding the user specified model parameters appropriate scanned data found from the database. Finally template body model mapped on the resulting scanned data with skeleton adjustment and displacement mapping. Resulting mesh processed with refinement operator to handle irregular deformation of the template one while mapping stage. Seo's extended and complete version of the work can be found in Seo[SMT04].

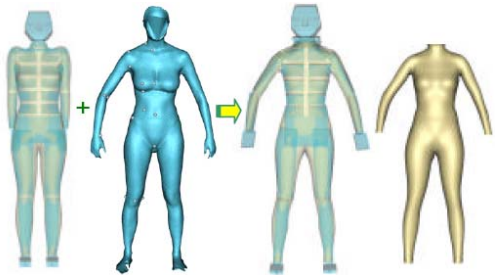

Figure 23: Mapping template model on the scanned data, Seo[33].

Similarly Allen at al [ACP03] developed an example based approach to transfer template body model on to a scanned data. They used 250 scanned data to demonstrate parameterization and reconstruction. With this approach it becomes possible to analyze the human body modelling applications like texture transfer, skinning transfer, shape analyze etc.

One novel extension to Seo[SMT04]'s method is attaching the real human skeleton model. Magnenat-Thalmann[MTYCS04] extracted the skeleton information from the visible human dataset[VH:95] where the proceedings extracted just the muscle tissue from the same dataset. With this approach instead of attaching and representing chopstick like skeleton models, author achieved to attach the skeleton with real characteristics. For different size models, skeletons scaled with the similar operations explained in their previous work.

Figure 24: Real like skeleton attachment, -Thalmann[MTYCS04].

Anguelov at al[ASK*05] presents a pose space deformation of the body model by using body scanner. Scanning the same person with different poses, deformation space generated. Two different body deformation models learned from the generated data: rigid and non-rigid. Proposed framework generates the desired body shape according to the parameters like angles of the joints and the eigen-coefficients of the shape. Different pose deformations captured from an individual is also applicable to others.

Since the hardware accelerated computation becomes more popular, researches shifted on the GPU based deformation techniques. Rhee at al[RLN06] is one of the earlier researchers who proposed real-time weighted pose space deformation technique. From a sufficient set of example of an articulated model they interpolate the displacements. Regarding this information skinning deformations parallelized on GPU to take the real-time efficiency.

After that section, we know how to deform a 3D character so that its dimensions match several criterions, for instance match someone's sizes. It is very likely that this person, who wants to try out garments, will want to see his/her avatar in motion. This isn't a trivial stage because as the avatar was deformed, its animation must be adapted accordingly. This problem is addressed in the next section, which focus on body animation and retargeting.

2.2 Body Animation and motion retargeting

Before dressing up a virtual character it must first be animated appropriately. This must be done is such a way that makes the cloth simulation possible. For instance, time derivatives of the body motion must be available so that the cloth can behave accordingly. The geometry of the body must be deformed smoothly so that the collision detection remains consistent, and the tessellation has to be uniform in order to maximize performances.

The most widely adopted way to create a virtual human goes through the modeling of a skeleton. This skeleton is the actual animated entity, and the character shape is deformed according to the skeleton pose.

2.2.1 Skeleton Animation

Skeletons that are used in character animation were conceptually borrowed from robotics. They are modeled through a collection of rigid links (or bones) connected together by joints (Figure 25). Each joint can have up to 6 degrees of freedom (DoF): 3 translations and 3 rotations, but they usually have between 1 and 3 rotational DoF in order to mimic the human skeleton. Skeletons are a rooted hierarchy, i.e. it starts from one node which has 6 DoF and goes hierarchically through the skeleton until an end effector is reached. This means that changing the orientation of a joint will modify the configuration of all the bones which are placed further down in the hierarchy. This root node is meant to place the character in the 3D scene, while the subsequent nodes will adapt its posture.

Such a model makes possible to manipulate limbs by simply tweaking 1 parameter value. For example, modifying the orientation of the shoulder joint will move the entire arm.

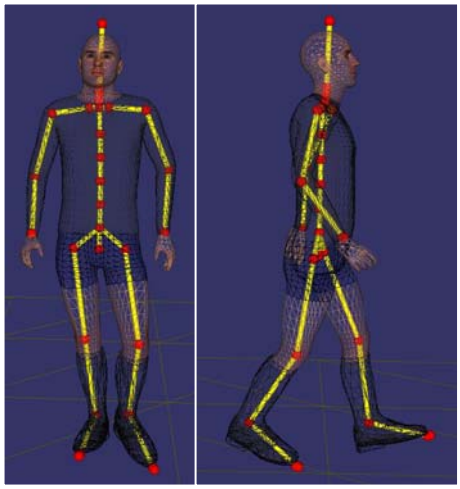

Figure 25: A virtual character in wire frame and the skeleton used to animate it. In yellow are the links (or bones) and in red the joints

There are various ways to create a skeletal animation. The most basic one, which is still widely used, is hand animation [Las87][Lass94]. A skilled animator will animate the skeleton using key frames, by carefully tuning the poses of the skeleton over time. This is a time consuming process, but it provides a total control over the final result.

Another way to generate animations is to use dynamic simulations. It works very well for motions without a specific goal, such as ragdolls [Kok04]. However the results are quite robot looking when addressing motions with a specific goal[HWBO95][YLS04]. The reason why these motions have a robotic look is because they all rely on finite state machine in order to convert the original goal to dynamic impulses, very much like the robotic community does. One way to overcome this issue is to add physics on top of a very basic motion. For instance, Liu and Popovic [LP02] proposed a framework which takes a collection of basic postures to be achieved by a character, and outputs a high quality motion by taking the physics into account. These methods have the advantage of relying solely on software in order to generate

the motion. It saves time compared to traditional animation, however such systems are very complex to develop, which makes their use quite limited.

The last way of creating motions is to record it directly on a real subject. This process is called motion capture (mocap), and it is probably the most widely used for commercial productions. Indeed, because the motion comes from a real performance, it ensures the highest level of realism for the resulting animations. There are several class of systems, and even though the most common systems are optical [Vic][MoAn] (Figure 26), several other approaches exist, such as magnetic trackers [Pol][Ass], exoskeleton [Meta] and more recently markerless captures [Orga] became commercially available.

Figure 26: A Vicon MX camera, with infrared emitters and filter

All these systems deliver different kind of data, and a post-processing stage must take place between the capture and character animation in order to convert the captured data to a suitable format. Optical mocap only records the location in 3D of reflective markers placed on the subject's body. These markers do not exactly reflect the bones motion, thus it must be estimated from the markers data only. This is done via an optimization process which places the skeleton in the configuration that better match the markers positions. In order to do so, several commercial packages are available, would they be delivered with the mocap hardware[VIQ][MoCal] or purchased separately [MoBuild].

2.2.2 Body Animation

Once the skeleton is animated, the actual character's shape must be deformed accordingly. There exist a wide variety of approaches that address this problem with various deformation results and performances. The most widely used, even though quite old, remains linear blend skinning [Magn88]. It simply blends together the displacement of a vertex, as if it would be rigidly attached to a bone. It requires a bind pose, which gives the relative displacement between the character mesh and the skeleton, along with some attachment data that defines which bones must be taken into account to deform a vertex. It is formulated as follow:

$$v' = \sum_i w_i M_i B_i^{-1} v$$

With v the initial vertex position in bind pose, Bi the bind transformation matrix of joint i, Mi the current transformation matrix of joint i, and wi the weight of joint i ($\sum_i w_i = 1$). It has the advantage of being very simple to formulate and implement, however it generates artifacts when the deformations become too large. Indeed, $\sum_i w_i M_i B_i^{-1}$ no longer is a rigid transform matrix, thus artifacts such as collapsing elbow and candy wrapper may appear.

The hardest part of this method is to define the blending weights so that the deformations are smooth and purposeful. Various CG packages [Max][Digi][Maya] provide interfaces to define the weights, which still must be tuned by an experienced designer.

In order to address the artifacts caused by linear blend skinning, various approaches were proposed. Wang et al. [Wang03] puts variable weights depending on the current bones transformations, Kavan et al. [Kavan05] interpolate the transformations instead of the final vertex positions, and Mohr et al. [Mohr03] introduces pseudo joints which aim at refining the skeleton motion.

More recently, Kavan et al. [KCZO07] proposed to use dual quaternions [Cliff82] in order to represent translations, which makes possible to efficiently blend translations and rotations without creating a degenerated transform.
Dual quaternions are similar to regular quaternions, i.e.

$$\hat{q} = \hat{w} + i\hat{x} + j\hat{y} + k\hat{z}$$

only the coefficients w, x, y and z are dual numbers, i.e. they're of the form

$$\hat{a} = a_0 + \varepsilon a_\varepsilon \text{ with } \varepsilon^2 = 0$$

A translation (t0, t1, t2) can be represented by the unit dual quaternion

$$\hat{q} = 1 + \frac{\varepsilon}{2}(t_0 i + t_1 j + t_2 k)$$

and a rigid transform by the product $\hat{q}q_0$, q0 being a unit quaternion representing a rotation. The linear blending thus simply becomes:

$$\frac{w_1\hat{q}_1 + \ldots + w_n\hat{q}_n}{\left\|w_1\hat{q}_1 + \ldots + w_n\hat{q}_n\right\|}$$

$\hat{q}_i$ being the dual quaternion representing the transformations to be taken into account, and wi the corresponding blending weights.

This method has the advantage of taking care of the artifacts usually introduced by linear blend skinning, while remaining fast due to the use of quaternions for the interpolation.

Other trends for skin deformation exist, such as physically based [Hua06][Cap05][Guo05] or example based [Kry02][Sloa01][Rhe06][Jam05][Par06], however due to their complexity and computational cost they aren't well suited for real-time animation, and will not be reviewed here.

2.2.3 Motion retargeting

Within the context of a virtual try on application, a given motion will probably be used in order to animate various sizes of body. Because each body is meant to reflect the physiological features of the user, it is subject to non uniform scaling of its limbs, along with a drastic change of the limbs shapes. This calls for the use of a motion retargeting algorithm, because one given motion can correctly be applied to only one body which resembles the subject onto whom it was captured.
In most applications requiring motion retargeting, the virtual character must evolve in a surrounding environment, and possibly interact with it. Dedicated solutions, making use of either inverse kinematics [TGB00][BCHB03] or global optimization [Glei98][LS99] were proposed. All these methods make use of constraints, which are to be enforced while remaining as close from the original motion as possible.
For a virtual try on application, the goal here isn't to modify the motion so that it complies with a given set of constraints, but rather to change it in such a way that it remains as close as possible from the original, while removing ugly artifacts like foot skating. As this problem is included in the more general approaches cited above, they still can be used. What they do is to minimize a set of parameters while enforcing a set of constraints (foot planting, interaction with the environment…). This goal can be formulated as follow:

$$\text{Minimize} \quad f(x) = \frac{1}{2} x^T M x$$
$$\text{Subject to} \quad c_i(x) = 0, i \in N_e$$
$$c_i(x) > 0, i \in N_i$$

With M a diagonal matrix defining a weight on each parameter: the bigger the weight of a parameter, the harder it is to modify it. x the parameter vector, c(x) a set of constraints to be satisfied.
All the high frequency features of a motion clip should be kept by this optimization process because they are what makes a motion natural looking. Thus, the vector x isn't composed of the actual parameter values of the joints, but rather of spline control points which are used to defined the changes from the original parameter curves. Thus, the control points spacing will determine the minimal frequency that is to be added to the motion signal. If they're too far apart, then the retargeting goal might not be feasible, while if they're too close they may add high frequency features and degrade the quality of the animation. A good spacing for the control point is between 2 and 10 frames.
Constraint optimization algorithm are quite tricky to implement, but fortunately it is possible to convert this problem into an unconstrained one by using the penalty method to handle the constraints. The above objective function thus becomes

$$g(x) = f(x) + \sum_{i \in Ne} w_i c_i(x)^2 + \sum_{i \in Ni} w_i (\min(c_i(x), 0))^2$$

With wi constraints weight used to put more importance on such or such constraints. This can be minimized using a regular conjugate gradient method.

Besides regular kinematics constraints, one may wish to enforce dynamic constraints as well, so that the resulting motion better matches the body onto which it's being applied. Several approaches were proposed so far [Pop99][ALP04] which rely on global optimization for performing the adaptation. A recent work from Tak et al.[TK05] used a Kalman filter in order to modify the motion and enforce physical consideration. They used a measurement model H composed of a collection of functions taking into account the constraints imposed by the user:

$$Z = \begin{bmatrix} H_K(q, \dot{q}, \ddot{q}) \\ H_B(q, \dot{q}, \ddot{q}) \\ H_T(q, \dot{q}, \ddot{q}) \\ H_M(q, \dot{q}, \ddot{q}) \end{bmatrix}$$

With HK the kinematic constraints (e.g. specify an end effector location), HB the balance constraint (i.e. make the zero momentum point lie within the supporting area), HT the torque constraints (i.e. this muscle can exert at most this force), HM the momentum constraints (for flying phases).
As the entire motion clip is already available, the process model simply is the value of the motion parameter at time tk, and it doesn't depend on the previous state:

$$\hat{x}_k^- = [q_k \dot{q}_k \ddot{q}_k]$$

The prediction consist of taking a measurement $\hat{Z}_k^-$ calculated by taking samples around $\hat{X}_k^-$ and making them go through the measurement model H.
The final update is eventually given by :

$$\hat{X}_k = \hat{X}_k^- + K_k(Z_k - \hat{Z}_k^-)$$

With Kk the kalman gain, and $Z_k = H(\hat{X}_k^-)$.
This formula quite differs from the regular kalman update, and the details of the calculation can of course be retrieved from the original paper.

Such framework is quite complex to develop, and a fair amount of user interaction is required for each motion clip in order to clearly define the constraints. Moreover, unless the character drastically changes from the original captured subject, the physically related artifacts aren't much visible. Thus, the very first problem that must be taken care of is foot skating because it is most visible to the casual eye. Kovar et al.[KSG02] used once again global optimization for addressing this issue while Glardon et al.[GBT06] used inverse kinematics in order to deal with it. However, these approaches only take the skeleton into account and with our virtual try on application in mind, it seems important to also consider the skin motion [LM07]. The goal here isn't to enforce the feet at a particular location, but rather to keep the motion as close from the original one as possible while getting rid of the foot skating. In order to do to, and assuming that the part of the foot sole that must remain planted is known for the entire timeline, the appropriate displacement ΔR_t of the root node of the skeleton can be calculated as follow:

$$\Delta R_t = o(v_i, t) - o(v_i, t+\delta) + o(v_j, t+\delta) - o(v_j, t+1)$$

With o(v,t) the offset at time t of vertex v from the root, in world coordinates, vi the vertex of the character's mesh that must remain static until time $t+\delta$, and vj the one to remain planted after that time.

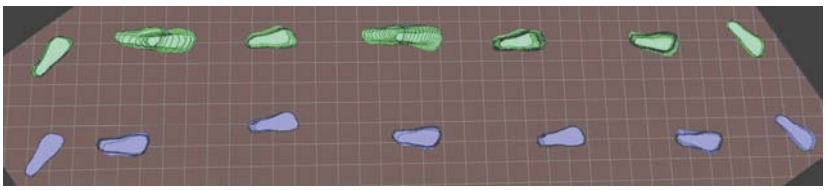

Figure 27: Footprints left by a virtual character before (top, in green) and after (bottom, in blue) foot skate cleanup.

This formula works very well for the horizontal translation of the character, as it introduces a drift according to the amount of foot skate that is present in the animation (Figure 27). However, in the vertical direction it also introduces a drift, due to the inaccuracy of the animation. In that direction, and as the motion of the character isn't to be modified, we can minimize the penetration of the feet into the ground and if the animation of the character is correct no penetration will remain, regardless of the character scaling.
Considering the heights ht of each static point throughout the animation, the offset to be applied to the root translation so that their mean becomes zero simply is $\Delta H = \sum \frac{h_t}{N}$, N being the total number of samples. Now considering $\overline{H}$ the average root height over the animation, its actual height Ht over the animation can be written as $H_t = \overline{H} + r_t$.

The scaling factor α which applied to the offsets rt minimizes the variance σ^2 of the static points around the ground floor is given by:

$$\alpha = -\frac{\sum r_t.(\overline{H} + l_t)}{\sum r_t^2}$$

Now that the avatars are able to walk around, they must be actually dressed up. High quality simulations of garment are very heavy in terms of computations thus they do not perform in real time. The next section outline how it is possible to adapt existing algorithms for garment animation, so that they perform with less accuracy, but in real-time.

2.3 Real-time cloth simulation

2.3.1 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 behaviour at different levels are necessary.
This study is not intended to integrate yet another more precise physical model of garment behaviour, 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.
Indeed, 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. Indeed, 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.

2.3.2 Techniques for Real-Time Garment Simulation

Fast Cloth Simulation

The first approach of real-time garment simulation is to optimize usual cloth simulation methods for better performance. This is usually carried out through the use of particle systems, which allow simple computation of approximate mechanical models.
Spring-mass systems are typically used in this context. When high accuracy is still needed, some particle systems allow the expression of viscoelastic materials with the accuracy of continuum mechanics [ETZ 03] [VOL 05], as described as follows.

Fast and Accurate Particle Systems

The goal of this model is to simulate the nonlinear and anisotropic behavior of cloth materials, which are typically described as strain-stress curves measured along the weft, warp and shear deformation modes. The major challenge is to find the best compromise between the high requirement for mechanical accuracy (quantitative accuracy with anisotropic nonlinear strain-stress behavior) and the drastic performance requirements of real-time and interactive applications.
One of the most efficient solutions is to take advantage of the particle system described in [VOL 05]. The mechanical model takes advantage is indeed based on continuum mechanics, but is still a particle system. Hence, it follows many of the properties initially found in finite elements [IRV 04]. Basically, the system evaluates the strain of each triangle element according to the position and speed of the particles, then uses the mechanical properties of the material for computing the stress of the elements, and converts back the stresses into equivalent particle forces.
While this scheme has strong analogies to first-order finite elements, we have however carried out some developments aimed at vastly improving the computation speed without too many sacrifices in the accuracy. Among these developments, computational simplifications are obtained by avoiding the computations required for the linearization of the strain and stress tensors [ETZ 03]. Furthermore, an adapted accurate computation of the Jacobian is implemented for ensuring numerical stability even is very severe deformations [CHO 02].
The main interest of this computational process is to offer very good computational performance while handling materials that possibly have nonlinear and anisotropic strain-stress laws, with accuracy in par with finite-element models. Yet, our system offers all the performance and flexibility related to particle systems, particularly trough the possibility of handling directly geometrical constraints such as collisions.

Bending stiffness should also be considered. However, bending forces are quite low in actual cloth materials, and through the use of large elements, it becomes quite useless to waste computation time evaluating forces that have almost no effect in the simulation. Still, when stiff bending forces are to be considered, new fast linear bending simulation schemes [VOL 06] may offer a very good computation compromise.

Figure 28: Accurate models of nonlinear anisotropic viscoelasticity are required to applications such as interactive prototyping.

Efficient Numerical Solvers

Numerical integration is another key issue to fast and efficient particle systems. While explicit methods guarantee dynamical accuracy at a very high computational time usually incompatible with real-time applications, implicit methods allow moving the trade line toward much higher computation performance at the expense of accuracy.

High-performance solvers [EBE 00] [HAU 01] [VOL 01] now focus on implicit methods for attaining the performance required for interactive models [COT 99] [JAM99] [HAU 03] [MEY 01] [MUL 00] [MUL 02]. However, their computational speed is obtained either through large compromises on the accuracy which, for low-order methods, affect the dynamic cloth motion realism through numerical damping, and for high-order methods, compromise the stability.

Since real-time application might deal with erratic animation behaviors (noisy motion tracking, skipped frames, etc), the BDF-2 method [HAU 01] might not be the best candidate, as this 2nd-order method might exhibit stability issues under severe deformation conditions particularly when dealing with highly nonlinear mechanical behaviors (collisions). Instead, adaptations of the simple Backward Euler method [VOL 01] seem to offer the best unconditional stability that is really necessary in the context of real-time applications. Meanwhile, accuracy can be significantly improved through the use of the Backward Midpoint variant [VOL 05].

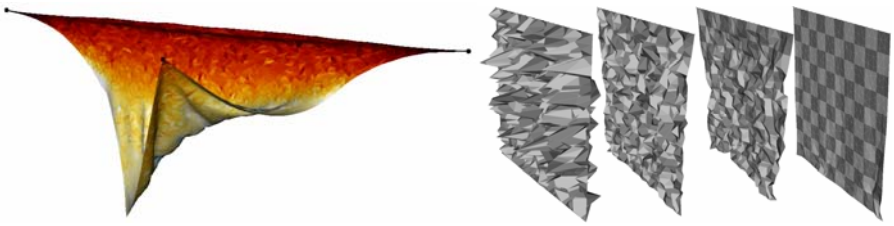

Figure 29: Stability tests on extremely deformed objects, and numerical convergence tests.

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. The most appropriate general solution is to use Bounding Volume Hierarchies, which take advantage of small-scale motion consistency of the objects during motion [MEY 00]. Specific acceleration techniques might approximate collisions and use graphics hardware to compute collisions on bump maps [VAS 01].

However, it is quite unlikely that general collision detection schemes offer adequate performance in the context of real-time simulation. Therefore, context-specific optimisations are usually required, for instance

by restricting collision detection only to the surface regions that have large probability to collide, and ignoring self-collision detection.

Integrating Body and Garment Animation

The main idea of hybrid approaches is to employ fast specific simulation techniques depending on the simulation context of particular regions of the cloth surface. A typical example, described in [COR 02], is described in the following.

The Hybrid Approach

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. 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.

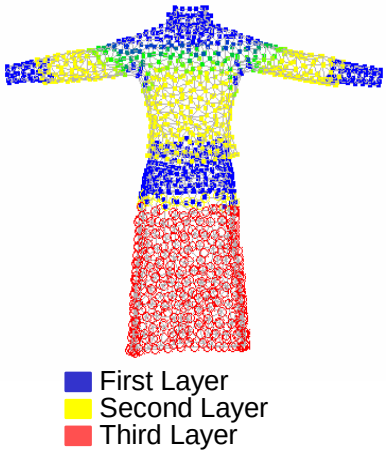

Figure 30: Garment segmentation

The Augmented Skinning Approach

Another approach is to base garment animation on directly on skinning animation. Hence, the garment is simply considered as a skin, uniformly animated like the rest of the body. However, this animation is complemented by a method which animates the skin according to mechanics, using the original skinned position for approximate collision processing.

In this approach, the garment has to be skinned, as would be the rest of the body surface. This is typically done through the use of automatic algorithms that extrapolate the skinning weights of the garment from the skinning weights of the underlying body surface. The automatic skinning extrapolation required an algorithm which tracks the relevant features of the body shape ruling the animation of any vertex of the garment surface. This algorithm can be designed by extending a proximity map (nearest mesh feature

algorithm) with additional visibility considerations for pinpointing the actual geometrical dependencies between the surfaces of the body and the cloth. Additional smoothness criteria should also be embedded so as to prevent any jaggy deformation over the garment surface. Further optimizations, such as the reduction of bone dependency count, are also performed for reducing the computational time of skinning animation. Collision data is obtained from the nearest-feature algorithm used in the skinning extrapolation scheme, and optimized with specific distance and visibility considerations.

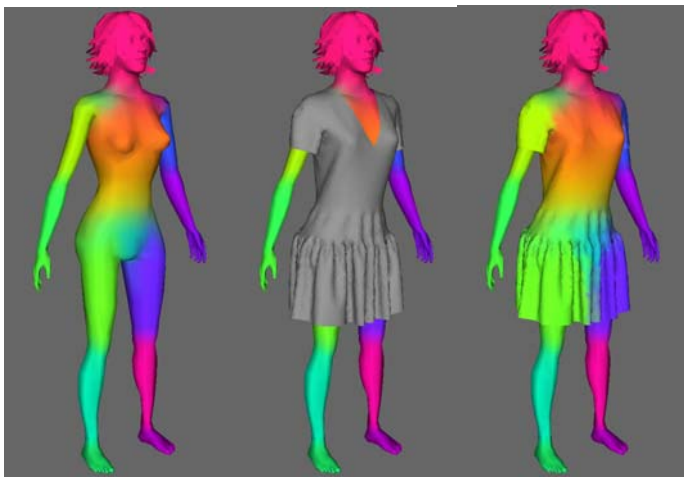

Figure 31: Extrapolating skinning information from the body surface to the garment surface.

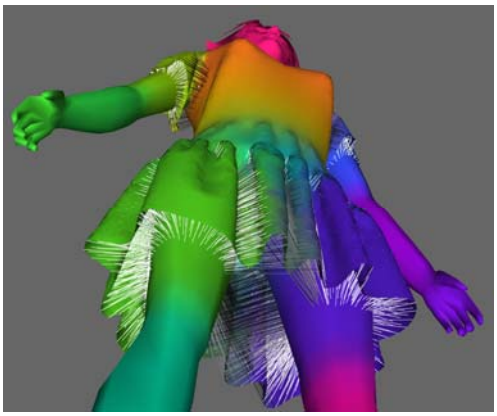

Figure 32: Collision data between the garment and the body extracted from proximity maps.

Data-Driven Animation

The main idea of this approach consists of developing a cloth simulator that can learn the cloth behavior through a sequence of pre-computed cloth simulation. This approach allows simulating the garment features such as wrinkles or gathers in real-time.

In [COR 04] is presented a data-driven method for simulating clothes worn by 3D characters in real-time. It divides 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.

To effectively optimize the physics-based deformation, which is the bottleneck of the simulation, a coarse representation of the cloth mesh is used to drive the gross behavior in simulation. It considers that the gross cloth behavior is driven mainly by two separable contributions: the skeleton-driven movement of the

character and the mechanical properties of the cloth. This consideration is 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 behavior. 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 this 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.
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.

Now we have all the pieces required for a Virtual Try On application: Cloth animation, body sizing and animation. Putting them together in a single, web based application can turn out to be very difficult if not well thought out in the first place. The next –and last- section gives guidelines on how to proceed.

2.4 An integrated Virtual Try On application

Internet is a field of interest which evolves perpetually, especially in the fashion industry. Its recent developments have made possible to distribute complex 3D data through the internet. Thus, streaming and 3D technologies can now be combined in order to create a Virtual Try On application (VTO). The VTO is a window on a virtual dressing room where users can define measurements of a 3D body, put clothes on it and make it walk around. Due to the limitations of the real time aspect and the web constraints, the integration platform needs to be optimized and well designed as presented in [CLSM01] [PLAM02].
In this section we present how to integrate all these elements in a web based application. Even though the web 2.0 allows for rich content and an increased bandwidth for data transfer, the existing stand alone applications must be redesigned in order to take into account the server-client architecture. Also, the data handling must be optimized so that the downloading time is reduced to a minimum. For instance, instead of adding many small details to the 3D models (Figure 33), state of the art rendering techniques such as CG shaders can be used in order to improve the visual quality of the whole application (Figure 34).

Figure 33: Real time dressed and animated virtual body.

2.4.1 A common platform

Several components must be put together in order to create a VTO application. A body animation system has to be created, along with a body deformer and real-time cloth animator module. Moreover, the system has to be fed by cloth designers, thus it must be possible to create content with the mainstream 3D modellers such as 3DS Max or Maya.

With this outlook, Collada (Collaborative Design Activity) seems to be a good candidate for data exchange between the content creators and run-time engine. Collada is an open file standard for interactive 3D applications currently being promoted by several major player of the industry, such as Sony or Khronos. The standard covers most of the features required for our VTO application: Scene graph hierarchy, materials and textures, animation, skinning and shaders. It has free plugins available for most of the 3D creation packages which makes it usable by the designers. The current features of Collada are well suited for body animation and rendering, but it lacks very specific schemes for the data related to the body customization and cloth animation. Fortunately, it is extensible with new specialized tags, which makes it possible to include also the data specific to a VTO application into one single Collada file.

Because the VTO is a web based application, it must be embedded into a web browser, so that a user can simply browse the internet in order to access it. This can be done through the ActiveX controls of internet explorer, which make possible to put any application (would it be 3D or not) into a web browser. Thus the core VTO application will be encapsulated into an activeX control so that no cumbersome installation process is required for the user.

There exist plenty of 3D engine, onto which a VTO application could be built: OpenSceneGraph, OpenSG, OpenInventor, openRM, OGRE… We choose to use OpenSceneGraph for several reasons: first of all, it is completely open source and free of charge, that way the final application can be distributed freely, and the maintenance is made easier by having a full access to the source code. Second, it has an off-the-shelve activeX control and thus the embedding into a web browser isn't a problem any longer. Last, the Collada plugin that converts Collada files into OSG data structure is available so that we can still benefit from the fancy shader effects as they were exported from the modelling tools.

Figure 34: Real time cloth animation (with FX shaders).

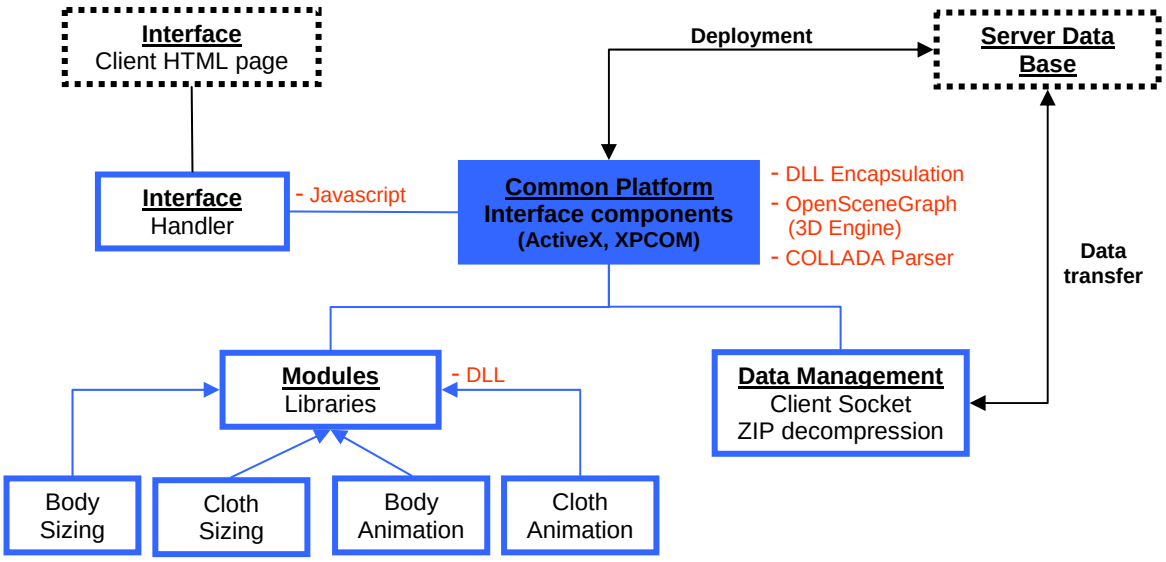

Figure 35: Relation between each component of the VTO application.

These three elements (COLLADA file format structure, Interface components (ActiveX and XPCOM) and OpenSceneGraph 3D engine) create our common integration web platform in term of structure and coding language. These elements constitute the core of our common platform (see Figure 35).

2.4.2 Web data management

Even though the internet bandwidth has drastically increased over the past decade, the content being transferred continues to become bigger and bigger. Thus, a data management strategy is essential for the VTO. As Collada is a text format only (i.e. no binary format was defined), these files will have to be compressed in some way before to be transmitted through the web. Because the text files themselves aren't too big (a few megabytes, without the textures), a simple zip compression is sufficient in order to ensure a quick download of the 3D content. For instance, the compression ratio obtained with WinZip 8.1 is more than 70% for a standard Collada file. That way, a 2 Mb file will be less than 600Kb after compression and the download time will be divided by 3.

In practice, the data is decompressed directly by the VTO which embeds a Zip decompression module. The decompression time is less than 0.5 seconds which is negligible for the whole downloading process.

On other hand, the textures employed are not easily compressible like a simple text. This represents the main difficulty for the web data management. The current best solution is to integrate a streaming module for the textures. It permits to display the 3D shape before to display the whole object. As COLLADA integrates the standard XML format, we could use it to stream textures from URL paths. Our application will stream the requested textures and according to textures ID it will subsequently process texturing on the object. This approach seems to be a good compromise between high resolution model and downloading time.

In this tutorial, the client/server management side will not be developed. We can only say that a data base is the usual solution to manage data but the best interest is how to broadcast it. Currently, a solution could be the peer to peer broadcasting method. This method increases the data speed deployment but it is totally dependent on the number of connected peers.

Web Interface

The web interface of the Virtual Try On platform provides the user with all the functionalities needed to define his/her own avatar. The user can modify measurements of a generic virtual body to fit

his/her general physical aspect. Choose its garments (including size and texture) and visualize the virtual body walking in a fashion show room with the garments animated in real time.

The web interface needs to be simple and user-friendly. All users should be able to handle it easily and intuitively. It follows that the main functionalities of the Virtual Try On are divided in three modules. In the first one, the user provides his measurements in order to get a 3D model with his profile. In the second one, he can choose between different cloth categories, select a garment to get information about it and try it on via his 3D model. Finally, in the third module the user can see his model walking around and can interact with it by zooming on it or changing the view angle.

The interface content should be well-ordered. Indeed the web interface elements which are the most important of the Virtual Try On web based application should be highlighted. In our prototype we show up the fact that we can choose the exact measurements for the 3D model. Indeed this element is very important in this interface and should appear earlier. This is the reason why we put this measurements selection with the common sizes selection in order to create a submenu called Measurement. Naturally, this measurements submenu will interact directly with the web based application and with our body sizing module. Each slider represents a sizable body part of the virtual body (see previous paragraph on Parametrically Deformable Human Bodies method). Next, the users have to select the cloth itself and to be able to get relevant information on it. Regarding to our goal, this submenu permits to get a global view of the cloth data bank, to follow the user choices and to view the result on the virtual body. The third submenu is called Show Room and permits to show the virtual body deformed and the chosen cloth, through an animation. It is composed by two buttons which are used to play or to stop the walk animation. Finally, the proposed submenus sequence follows the natural logic proposed as to know the measurements following by the dressing choice and the show room. This corresponds to our modules sequence i.e. body sizing following by cloth sizing following by body and cloth animation.

The current interface is developed using Flash tool. The Flash Player developed and distributed by Adobe Systems (www.adobe.com) is a client application available for most common web browsers. It features support for vector and raster graphics, a scripting language called Action Script and bi-directional streaming of audio and video. This tool provides compatibilities with ActiveX and XPCOM components and can communicate through JavaScript by FScommand which are Action Script methods.

To conclude, the Virtual Try On regroups two main elements which are the web based application and the web interface. Our web based application is constructed on a common structure which is composed by the Collada file format, the OpenSceneGraph 3D engine, interface components and our in-house modules. Currently, this development is not finished yet, but currently follows the presented integration strategy above.

3 References

[AAT02] AATCC Technical Manual 2002, www.aatcc.org

[ABH*94] Azuola F., Badler N., Ho P., Kakadiaris I., Metaxas D., Ting B.: Building anthropometry-based virtual human models. Proc. IMAGE VII Conference (June 1994).

[ACP03] Allen B., Curless B., Popovic Z.: The space of human body shapes: reconstruction and parameterization from range scans. ACM Trans. Graph. 22, 3 (2003), 587-594.

[ALP04] ABE Y., LIU K., POPOVIC Z.: Momentum-based parameterization of dynamic character motion, SCA 2004.

[ASK*05] Anguelov D., Srinivasan P., Koller D., Thrun S., Rodgers J., Davis J.: Scape: shape completion and animation of people. ACM Trans. Graph. 24, 3 (2005), 408-416.

[Ass] http://www.ascension-tech.com/ the Ascension technologies website, accessed May2007

[AT] Aubel A., Thalmann D.: Realistic deformation of human body shapes. In Computer Animation and Simulation '00, pp. 125-135.

[AT01] Aubel A., Thalmann D.: Interactive modeling of human musculature. Interactive Modeling of Human Musculature. Computer Animation 2001, Seoul (2001).

[Bar81] Barr A.: Superquadrics and angle-preserving transformations. IEEE Computer Graphics and Applications 1, 1 (1981), 11-23.

[Bar84] Barr A.: Global and local deformations of solid primitives. In SIGGRAPH '84: Proceedings of the 11th annual conference on Computer graphics and interactive techniques (New York, NY, USA, 1984), ACM Press, pp. 21-30.

[BAR 98] D. BARAFF, A.WITKIN, 1998, Large Steps in Cloth Simulation, Computer Graphics (SIGGRAPH’98 proceedings), ACM Press, pp 43-54.

[BCHB03] BOULIC R., LE CALLENNEC B., HERREN M., BAY H.: Experimenting prioritized IK for motion editing. Proceedings of Eurographics 2003.

[BEH61] B. Behre, “Mechanical Properties of Textile Fabrics Part I: Shearing”, Textile Research Journal, Vol. 31, No.2, 1961, pp. 87-93

[Bla94] Blanc C.: Superquadrics a generic implementation of axial procedural deformation techniques. In Graphics Gems (1994), vol. 5, Academic Press, pp. 249-256.

[BPW93] Badler N. I., Phillips C. B., Webber B. L.: Simulating Humans: Computer Graphics Animation and Control. Oxford University Press, New York, NY, Oxford, 1993.

[CHO 02] K.J. CHOI, H.S. KO, 2002, Stable but Responsive Cloth, Computer Graphics (SIGGRAPH’02 proceedings), Addison Wesley, pp 604-611.

[Cap05] CAPPEL S., BURKHART M., CURLESS B., DUCHAMP T. POPOVIC A.: Physically Based Rigging for Deformable Characters, Symposium on Computer Animation 2005.

[CBC*05] Capell S., Burkhart M., Curless B., Duchamp T., Popovic Z.: Physically based rigging for deformable characters. In SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation (New York, NY, USA, 2005), ACM Press, pp. 301-310.

[CGC*02] Capell S., Green S., Curless B., Duchamp T., Popovic Z.: Interactive skeleton-driven dynamic deformations. In SIGGRAPH'02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques (New York, NY, USA, 2002), ACM Press, pp. 586-593.

[CHP89] Chadwick J. E., Haumann D. R., Parent R. E.: Layered construction for deformable animated characters. In SIGGRAPH'89: Proceedings of the 16th annual conference on Computer graphics and interactive techniques (New York, NY, USA, 1989), ACM Press, pp. 243-252.

[Cliff82] CLIFFORD W.: Mathematical Papers, London, Macmillan, 1882.

[CLSM01] Frédéric Cordier, WonSook Lee, HyeWon Seo, Nadia Magnenat-Thalmann, Virtual-Try-On on the Web, VRIC, Virtual Reality International Conference, Laval Virtual 2001, May 16-18.

[Coq90] Coquillart S.: Extended free-form deformation: a sculpturing tool for 3d geometric modeling. In SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques (New York, NY, USA, 1990), ACM Press, pp. 187-196.

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

[COR 04] F. CORDIER, N. MAGNENAT-THALMANN, 2004, A Data-driven Approach for Real-Time Clothes Simulation. 12th Pacific Conference on Computer Graphics and Applications, IEEE Computer Society, pp. 257-266.

[COT 99] S. COTIN, H. DELINGETTE, N. AYACHE, 1999, Real-Time Elastic Deformations of Soft Tissues for Surgery Simulation, IEEE Transactions on Visualization and Computer Graphics, 5(1), pp 62-73.

[dATSS07] de Aguiar E., Theobalt C., Stoll C., Seidel H.-P.: Rapid animation of laser-scanned humans. In IEEE Virtual Reality 2007 (Charlotte, USA, 2007), IEEE, pp. 223-226.

[DCKY02] Dong F., Clapworthy G. J., Krokos M. A., Yao J.: An anatomy-based approach to human muscle modeling and deformation. IEEE Transactions on Visualization and Computer Graphics 8, 2 (2002), 154-170.

[DES 99] M. DESBRUN, P.SCHRÖDER, A.H. BARR, 1999, Interactive Animation of Structured Deformable Objects, Proceedings of Graphics Interface, A K Peters, pp 1-8.

[DES 01] G. DESBRUNNE, M. DESBRUN, M.P. CANI, A.H. BARR, 2001, Dynamic Real-Time Deformations Using Space & Time Adaptive Sampling, Computer Graphics (SIGGRAPH'01 proceedings), Addison Wesley, pp 31-36.

[Digi] http://www.digimation.com/home/ Digimation, provider of Bones Pro 3, accessed May 2007

[DMS98] DeCarlo D., Metaxas D., Stone M.: An anthropometric face model using variational techniques. In SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques (New York, NY, USA, 1998), ACM Press, pp. 67-74.

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

[EHW*07] E. H. D., H. Y. S., W. C. J., K. S. J., S. K. M., B. J.: Sweep-based human deformation. The Visual Computer 21 (2007), 542-550.

[ETZ 03] O. ETZMUSS, J. GROSS, W. STRASSER, 2003, Deriving a Particle System from Continuum Mechanics for the Animation of Deformable Objects, IEEE Transactions on Visualization and Computer Graphics, IEEE Press, pp 538-550.

[ETZ 03] O. ETZMUSS, M. KECKEISEN, W. STRASSER, 2003, A Fast Finite-Element Solution for Cloth Modeling, Proceedings of the 11th Pacific Conference on -Computer Graphics and Applications, pp 244-251.

[GBT06] GLARDON P., BOULIC R., THALMANN D.: Robust on-line adaptive footplant detection and enforcement. The Visual Computer 22(3), 194-209, 2006.

[GC95] Gortler S. J., Cohen M. F.: Hierarchical and variational geometric modeling with wavelets. In SI3D '95: Proceedings of the 1995 symposium on Interactive 3D graphics (New York, NY, USA, 1995), ACM Press, pp. 35-*.

[Glei98] GLEICHER M.: Retargeting motion to new characters. SIGGRAPH 98.

[GRI 03] E. GRIMSPUN, A.H. HIRANI, M. DESBRUN, P. SCHRÖDER, 2003, Discrete Shells, Eurographics Symposium on Computer Animation, pp 62-68.

[Guo05] GUO Z., WONG K.: Skinning with Deformable Chunks, Computer Graphics Forum 2005.

[HAU 01] M. HAUTH, O. ETZMUSS, 2001, A High Performance Solver for the Animation of Deformable Objects using Advanced Numerical Methods, Eurographics 2001 proceedings, pp 137-151.

[HAU 03] M. HAUTH, J. GROSS, W. STRASSER, 2003, Interactive Physically-Based Solid Dynamics, Eurographics Symposium on Computer Animation, pp 17-27.

[HD55] Hartenberg R. S., Denavit J.: A kinematic notation for lower pair mechanisms based on matrices. Journal of Applied Mechanics 77 (1955), 215-221.

[HMT*85] H. N., M. H., T. K., T. K., I. S., K O.: Object modelling by distribution function and a method for image generation. Transaction for the Institute of Electronics and Communication for Engineers of Japan J68-D (1985), 718-725.

[HS:07] Human solutions, http://www.human-solutions.com, May 2007.

[Hua06] HUANG J., SHI X., LIU X., ZHOU K. : Subspace gradient domain mesh deformation, ACM Transactions on Graphics 2006

[HWBO95] HODGINS J., WOOTEN W., BROGAN D., O'BRIEN J.: Animating Human Athletics, SIGGRAPH 1995.

[HYKJ03] Hyun D.-E., Yoon S.-H., Kim M.-S., Jittler B.: Modeling and deformation of arms and legs based on ellipsoidal sweeping. In PG '03: Proceedings of the 11th Pacific Conference on Computer Graphics and Applications (Washington, DC, USA, 2003), IEEE Computer Society, p. 204.

[IRV 04] G. IRVING, J. TERAN, R. FEDKIW, 2004, Invertible Finite Elements for Robust Simulation of Large Deformation, Eurographics Symposium on Computer Animation, pp 131-140.

[JAM 99] D. JAMES, D. PAI, 1999, ArtDefo - Accurate Real-Time Deformable Objects, Computer Graphics (SIGGRAPH'99 proceedings), ACM Press, pp 65-72.

[Jam05] JAMES D., TWIGG C.: Skinning Mesh Animations, ACM Transactions on Graphics 2005.

[JTT94a] Jianhua S., Thalmann N. M., Thalmann D.: Human skin deformation from cross sections. Proc. Computer Graphics International '94 (1994).

[JTT94b] Jianhua S., Thalmann N. M., Thalmann D.: Human skin deformation from cross-sections. In Computer Graphics Int. '94 (27-1 1994).

[Kavan05] KAVAN L., ZARA J.: Spherical blend skinning: a real-time deformation of articulated models, Interactive 3D Graphics 2005

[KAW80] Kawabata S., "The standardization and analysis of hand evaluation (2nd edition). The hand evaluation and standardization committee", The Textile Machinery Society of Japan, Osaka, 1980

[KCZO07] KAVAN L, COLLINS S., ZARA J., O'SULLIVAN C.: Skinning with dual quaternions, Symposium on Interactive 3D Graphics. 2007.

[Kok04] KOKKEVIS E.: Practical Physics for Articulated Characters, Game Developers Conference, 2004

[Kry02] KRY P., JAMES D., PAI D.: EigenSkin: Real Time Large Deformation Character Skinning in Hardware, Symposium on Computer Animation 2002.

[KSG02] KOVAR L., SCHREINER J., GLEICHER M.: Footskate cleanup for motion capture editing. SCA2002.

[KZ05] Kavan L., Zara J.: Spherical blend skinning: a real-time deformation of articulated models. In I3D '05: Proceedings of the 2005 symposium on Interactive 3D graphics and games (New York, NY, USA, 2005), ACM Press, pp. 9-16.

[Las87] Lasseter J.: Principles of traditional animation applied to 3D computer animation. In SIGGRAPH '87: Proceedings of the 14th Annual conference on Computer graphics and interactive techniques (New York, NY, USA, 1987), ACM Press, pp. 35-44.

[Lass94] LASSETER J.: Tracks to Animating Characters with a Computer, SIGGRAPH94 Course

[LCA05] Larboulette C., Cani M.-P., Arnaldi B.: Dynamic skinning: adding real-time dynamic effects to an existing character animation. In SCCG '05: Proceedings of the 21st spring conference on Computer graphics (New York, NY, USA, 2005), ACM Press, pp. 87-93.

[LD61] J. Lindberg and B. Dahlberg, "Mechanical Properties of Textile Fabrics, Part III: Shearing and Buckling of Various Commercial Fabrics," Textile Research Journal, Vol. 31, 1961

[LM07] LYARD E., MAGNENAT-THALMANN N.: A simple footskate removal method for virtual reality applications, The visual computer, to appear.

[LP02] LIU K., POPOVIC Z.: Synthesis of Complex Dynamic Character Motion from Simple Animations, SIGGRAPH 2002.

[LS99] LEE J., SHIN S.: A hierarchical approach to interactive motion editing for human –like figures. Proceedings of SIGGRAPH 1999.

[LW07] Li J., Wang Y.: Automatically construct skeletons and parametric structures for polygonal human bodies. Computer Graphics International (2007).

[LWN94] Lamousin H. J., W. N. W.: Nurbs-based free-form deformations. IEEE Computer Graphics and Applications 14, 9 (1994), 59-65.

[Magn88] MAGNENAT-THALMANN N., LAPERRIERE R., THALMANN D.: Joint-dependant local deformations for hand animation and object grasping, Graphics Interface, 1988.

[Max] http://usa.autodesk.com/adsk/servlet/index?id=5659302&siteID=123112 Autodesk 3DSMax website, accessed May 2007

[Maya] http://usa.autodesk.com/adsk/servlet/index?siteID=123112&id=7635018 Autodesk Maya, accessed May 2007

[MBK07] Marinov M., Botsch M., Kobbelt L.: Gpu-based multiresolution deformation using approximate normal field reconstruction. Journal of graphics tools 12, 1 (2007), 27-46.

[Meta07] http://www.metamotion.com/gypsy/gypsy-motion-capture-system-mocap.htm

[MEY 01] M. MEYER, G. DEBUNNE, M. DESBRUN, A. H. BARR, 2001, Interactive Animation of Cloth-like Objects in Virtual Reality, Journal of Visualization and Computer Animation, Wiley, 12(1), pp 1-12.

[MIN95] Minazio P. G., “FAST – Fabric Assurance by Simple Testing”, International Journal of Clothing Science and Technology, Vol. 7, No. 2/3, 1995

[MoAn] http://www.motionanalysis.com/ the Motion Analysis website, accessed May 2007

[MoBuild] www.autodesk.com/motionbuilder Autodesk post processing software, accessed May 2007

[MoCal] http://www.motionanalysis.com/applications/animation/games/calcium.html, Motion Analysis Calcium post processing software, accessed May 2007

[Mohr03] MOHR A., GLEICHER M.: Building Efficient, Accurate Character Skins From Examples, ACM Transactions on Graphics 2003.

[MT97] Moccozet L., Thalmann N. M.: Dirichlet free-form deformations and their application to hand simulation. In CA '97: Proceedings of the Computer Animation (Washington, DC, USA, 1997), IEEE Computer Society, p. 93.

[MTT87] Magnenat-Thalmann N., Thalmann D.: The direction of synthetic actors in the film rendez-vous Montreal. IEEE Computer Graphics and Applications 7, 12 (Dec. 1987), 9-19.

[MTT91] Magnenat-Thalmann N., Thalmann D.: Human body deformations using joint-dependent local operators and finite-element theory. 243-262.

[MTYCS04] Magnenat-Thalmann N., Yahia-Cherif L., Seo H.: Modeling anatomical-based humans. In ICIG '04: Proceedings of the Third International Conference on Image and Graphics (ICIG'04) (Washington, DC, USA, 2004), IEEE Computer Society, pp. 476-480.

[MUL 02] M. MULLER, J. DORSEY, L. MCMILLAN, R. JAGNOW, B. CUTLER, 2002, Stable Real-Time Deformations, Proceedings of the Eurographics Symposium on Computer Animation, pp 49-54.

[MUL 00] M. MULLER, M. GROSS, 2000, Interactive Virtual Materials, Proceedings of Graphics Interface, Canadian Human-Computer Communications Society, pp 239-246.

[NT98a] Nedel L., Thalmann D.: Modeling and deformation of the human body using an anatomically-based approach. In CA '98: Proceedings of the Computer Animation (Washington, DC, USA, 1998), IEEE Computer Society, p. 34.

[NT98b] Nedel L. P., Thalmann D.: Real time muscle deformations using mass-spring systems. In CGI '98: Proceedings of the Computer Graphics International 1998 (Washington, DC, USA, 1998), IEEE Computer Society, p. 156.

[OBR 99] J. O'BRIEN, J. HODGINS, 1999, Graphical Modeling and Animation of Brittle Fracture, Computer Graphics (SIGGRAPH'99 proceedings), ACM Press, pp 137-146.

[Orga] http://www.organicmotion.com, the Organic Motion website, accessed May 2007

[Par06] PARK S. HODGINS J.: Capturing and animating skin deformation in human motion, ACM Transactions on Graphics 2006.

[PCLS05] Pratscher M., Coleman P., Laszlo J., Singh K.: Outside-in anatomy based character rigging. In SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation (New York, NY, USA, 2005), ACM Press, pp. 329-338.

[PH06] Park S. I., Hodgins J. K.: Capturing and animating skin deformation in human motion. ACM Trans. Graph. 25, 3 (2006), 881-889.

[PIE30] Peirce F.T., "The Handle of Cloth as a Measurable Quantity", Journal of the Textile Institute, Vol. 21, 1930

[PLAM02] Dimitris Protopsaltou, Christiane Luible, Marlene Arevalo, Nadia Magnenat-Thalmann, A body and garment creation method for an Internet based virtual fitting room. Computer Graphics International Conference Proceedings, Springer Verlag, pp. 105 -122. July, 2002.

[Pol] http://www.polhemus.com/ the Polhemus website, accessed May 2007

[Pop99] POPOVIC Z.: Physically based motion transformation, SIGGRAPH99

[Rhe06] RHEE T., LEWIS J.P. NEUMANN U.: Real-Time Weighted Pose-Space Deformation on the GPU, Computer Graphics Forum 2006.

[RLN06] Rhee T., Lewis J., Neumann U.: Real-timeweighted pose-space deformation on the GPU. Computer Graphics Forum 25, 3 (2006), 439-448.

[SF98] Singh K., Fiume E.: Wires: a geometric deformation technique. In SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques (New York, NY, USA, 1998), ACM Press, pp. 405-414.

[SK00] Singh K., Kokkevis E.: Skinning characters using surface oriented free-form deformations. In Graphics Interface (2000), pp. 35-42.

[Sloa01] SLOAN P., ROSE F., COHEN M.: Shape by Example, Interactive 3D Graphics 2001.

[SMT03] Seo H., Magnenat-Thalmann N.: An automatic modeling of human bodies from sizing parameters. In I3D '03: Proceedings of the 2003 symposium on Interactive 3D graphics (New York, NY, USA, 2003), ACM Press, pp. 19-26.

[SMT04] Seo H., Magnenat-Thalmann N.: An example-based approach to human body manipulation. Graph. Models 66, 1 (2004), 1-23.

[SP86] Sederberg T. W., Parry S. R.: Free-form deformation of solid geometric models. In SIGGRAPH '86: Proceedings of the 13th annual conference on Computer graphics and interactive techniques (New York, NY, USA, 1986), ACM Press, pp. 151-160.

[SPCM97] Scheepers F., Parent R. E., Carlson W. E., May S. F.: Anatomy-based modeling of the human musculature. In SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques (New York, NY, USA, 1997), ACM Press/Addison-Wesley Publishing Co., pp. 163-172.

[ST95] Shen J., Thalmann D.: Interactive shape design using metaballs and splines. In Implicit Surfaces'95 (Grenoble, France, 1995), pp. 187-196.

[TBB07] http://www.thinkbroadband.com/news/3049-world-broadband-statistics.html published originally by Point-Topic.com Tuesday 10 April 2007

[TGB00] TOLANI D., GOSWAMI A., BADLER N.: Real-time inverse kinematics techniques for the anthropomorphic limbs. Graphical Models 62, 353-388 (2000)

[THA 03] N. MAGNENAT-THALMANN, H. SEO, F. CORDIER, 2003, Automatic Modeling of Virtual Humans and Body clothing. Proc. 3-D Digital Imaging and Modeling, IEEE Computer Society Press, pp. 2-10.

[TK05] TAK S., KO H.: A physically-based motion retargeting filter, ACM Transactions on Graphics 24(1), 2005.

[TSB*05] Teran J., Sifakis E., Blemker S. S., Ng-Thow-Hing V., Lau C., Fedkiw R.: Creating and simulating skeletal muscle from the visible human data set. IEEE Transactions on Visualization and Computer Graphics 11, 3 (2005), 317-328.

[VH:95] Visible human project, http://www.nlm.nih.gov/research/visible, 1995.

[Vic] www.vicon.com, the vicon website, accessed May 2007

[VIQ] http://www.vicon.com/products/viconiq.html , Vicon IQ post processing software, accessed May 2007

[VM05] Volino P., Magnenat-Thalmann N., "Accurate Garment Prototyping and Simulation", Computer-Aided Design & Applications, Vol. 2, No. 1-4, 2005

[VOL 00] P. VOLINO, N. MAGNENAT-THALMANN, 2000, Virtual Clothing: Theory and Practice, Springer.

[VOL 05] P. VOLINO, N. MAGNENAT-THALMANN, 2005, Accurate Garment Prototyping and Simulation, Computer-Aided Design & Applications, CAD Solutions, 2(5), pp 645-654.

[VOL 05] P. VOLINO, N. MAGNENAT-THALMANN, 2005, Implicit Midpoint Integration and Adaptive Damping for Efficient Cloth Simulation, Computer Animation and Virtual Worlds, Wiley, 16(3-4), pp 163-175.

[VOL 06] P. VOLINO, N. MAGNENAT-THALMANN, 2006, Simple Linear Bending Stiffness in Particle Systems, Proceedings of the Symposium on Computer Animation 2006, pp 101-105.

[Wang03] Wang et al., Multi-weight enveloping: least-squares approximation techniques for skin animation, SCA2003

[WG97] Wilhelms J., Gelder A. V.: Anatomically based modeling. In SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques (New York, NY, USA, 1997), ACM Press/Addison-Wesley Publishing Co., pp. 173-180.

[WR07] Wang Q., Ressler S.: Generation and manipulation of h-anim caesar scan bodies. In Web3D '07: Proceedings of the twelfth international conference on 3D web technology (New York, NY, USA, 2007), ACM Press, pp. 191-194.

[YLS04] YANG P., LASZLO J., SINGH K.: Layered Dynamic Control for Interactive Character Swimming, SCA 2004.

[ZL05] Zhou X., Lu J.: Nurbs-based galerkin method and application to skeletal muscle modeling. In SPM '05: Proceedings of the 2005 ACM symposium on Solid and physical modeling (New York, NY, USA, 2005), ACM Press, pp. 71-78.

[ZTS06] Zhaoqi W., Tianlu M., Shihong X.: A fast and handy method for skeleton-driven body deformation. Computer Entertainment 4, 4 (2006), 6.